Design and Self-Management of Wireless Networked Systems with Model-Driven Optimization

Von der Fakultät für Elektrotechnik und Informationstechnik der Rheinisch-Westfälischen Technischen Hochschule Aachen zur Erlangung des akademischen Grades eines Doktors der Ingenieurwissenschaften genehmigte Dissertation

vorgelegt von

Elena Meshkova, M.Sc.
aus Moskau, Russische Föderation

Berichter: Univ.-Prof. Dr. Petri Mähönen
Univ.-Prof. Dr. George C. Polyzos

Tag der mündlichen Prüfung: 15. Dezember 2014

Abstract

Wireless networked systems are becoming increasingly popular, with a growing number of deployments and diverse applications. This leads to increasing complexity of these systems, as due to the shared nature of the wireless medium the amount of network resources available is constantly decreasing. These networks have to operate in challenging and dynamic conditions, and accommodate diverse and often contradictory objectives stated by multiple stakeholders. Often wireless networks are integral parts of larger systems, like the Internet, and need to utilize existing imperfect software and hardware components, which results in additional heterogeneity and implementation challenges. All this makes design, planning, and run-time management of wireless networked systems a demanding task.

In this thesis we address some of these challenges through a model-driven optimization methodology, which is formalized using category theory. We state a meta-optimization problem based on requirements of network players, operational context, applicable models that consider for both parameter- and component-based solutions. In order to simplify the problem we propose applying the mapping functions of abstraction, transformation, and decomposition, while considering the aspects of information loss or disruption in the problem formulation. We show on selected case studies how the proposed methodology can be employed at both static (design, planning) and dynamic (run-time management) stages of a network's life-cycle. In particular, we demonstrate that this methodology is effective for a detailed design and implementation of a self-optimizing system for wireless home networks, and optimization of protocol stacks for wireless sensor systems with an ontology-driven framework.

Since modeling is a crucial aspect of network optimization, we also propose and investigate several types of models in this work. Directed labeled graphs and their network motifs are used for identification of the networking context in the CSMA/CA based networks. We show how spatial network structure affects characteristics of these graphs, such as node degree distributions and occurrence patterns of small network motifs. We also propose a novel graph edge labeling based on clustered correlation coefficients that capture network dynamics imposed by tunable network parameters. We argue that this robust metric can lead to faster network optimization in a variety of operational conditions. We also consider modeling of the temporal context. In particular, based on power spectrum measurements we obtain online hidden semi-Markov models of network activity patterns and apply them as part of a dynamic spectrum access scheme. Additionally, we exploit metaheuristic mechanisms for cross-layer optimization and network planning that aim at robustness, performance maximization and optimizability of the system. We investigate how the size and the structure of the state space, the availability of valid models and utility functions influence the convergence of these methods. Our results show that for network problems it is often better to invest in careful problem formulation and long execution of simple metaheuristics rather than going for their custom modifications besides the simplest ones.

The proposed approaches have being extensively prototyped or proven through simulation based experimentation. In particular we have focused on small-scale wireless networked systems that utilize IEEE 802.11 radio interfaces and wireless sensor networks. We have also experimented with the WARP software defined radio platforms.

Kurzfassung

Drahtlose vernetzte Systemen gewinnen zunehmend an Bedeutung, belegt durch eine zunehmende Zahl von Installationen und ihre diversen Einsatzgebiete. Einhergehend mit diesem Bedeutungsgewinn nimmt die Komplexität dieser Systeme immer weiter zu, da durch eine gemeinsame Nutzung des drahtlosen Mediums die Zahl freier Netzwerkressourcen immer weiter abnimmt. Heutige Netzwerke operieren häufig unter schwierigen und hochgradig dynamischen Bedingungen, und müssen vielfältigen und teils widersprüchlichen Anforderungen gerecht werden. Oft sind Drahtlosnetzwerke ein integraler Bestandteil größerer Systeme, wie zum Beispiel dem Internet, und müssen ungeeignete Software- und Hardwarekomponenten einsetzen, was zu zusätzlicher Heterogenität und Implementationsherausforderungen führt. Zusammengenommen macht dies den Entwurf, die Planung, und die Laufzeitverwaltung von drahtlosen Netzwerken zu einer herausfordernden Aufgabe.

In dieser Arbeit adressieren wir einige dieser Herausforderungen durch eine modelbasierte, auf dem Formalismus der Kategorientheorie aufbauende, Optimierungsmethode. Wir formulieren Meta-Optimierungsprobleme gemäß den Anforderungen der Netzwerkspieler, dem operationellen Kontext, und anwendbaren Modellen die sowohl parameter- als auch komponentenbasierte Lösungen berücksichtigen. Um das Problem zu vereinfachen, schlagen wir die Anwendung von Abbildungsfunktionen wie Abstraktion, Transformation, und Dekomposition vor, und berücksichtigen die Aspekte des Informationsverlusts und der Unterbrechnung in der Problemformulierung. Wir zeigen anhand ausgewählter Fallbeispiele wie die vorgeschlagene Methodik sowohl in statischen (Entwurf, Planung) als auch dynamischen (Laufzeitverwaltung) Stadien eines Netzwerkes genutzt werden kann. Insbesondere demonstrieren wir, dass unser Ansatz für einen detaillierten Entwurf und Implementierung von selbstoptimierenden Systemen für Heimnetzwerke geeignet ist, und für die Optimierung von Protokollstacks für drahtlose Sensornetzwerke mit einem ontologiebasierten Framework.

Da die Modellierung ein zentraler Aspekt der Netzwerkoptimierung ist, schlagen wir in dieser Arbeit verschiedene Modelltypen vor und untersuchen sie. Gerichtete beschriftete Graphen und ihre Netzwerkentsprechungen werden für die Identifikation des Netzwerkkontextes in diesen CSMA/CA-Netzwerken eingesetzt. Wir zeigen, wie die räumlichen Netzwerkstrukturen die Charakteristiken dieser Graphen beeinflussen, wie zum Beispiel die Verteilung des Verbindungsgrades und das Auftreten kleinerer Netzwerkentsprechungen. Wie schlagen darüberhinaus einen neuartigen Algorithmus zur Graphenbeschriftung vor, der die durch vielfältigen Parameter eines Netzwerks ausgelöste Dynamik erfassen kann. Wir argumentieren, dass diese robuste Metrik zu einer schnelleren Netzwerkoptimierung in einer Vielzahl von Einsatzszenarien führen kann. Wir berücksichtigen auch den temporalen Kontext der Modellierung. Insbesondere leiten wir online Hidden Semi-Markov Modelle der Netzwerkaktivitätsmuster mittels Leistungsspektrumsmessungen her, und wenden diese als Teil dynamischer Spektrumszugriffsschemata an. Außerdem nutzen wir metaheuristische Modelle zur schichtenübergreifenden Optimierung und Netzwerkplanung mit dem Ziel der Robustheit, Leistungsmaximierung und

Optimalität des Systems. Wir untersuchen wie die Größe und Struktur des Zustandsraums, sowie die Verfügbarkeit valider Modelle und Nutzenfunktionen die Konvergenz dieser Methoden beeinflussen. Unsere Ergebnisse zeigen, dass es häufig sinnvoller ist, in eine vorsichtige Problemformulierung und eine lange Ausführungszeit einfacher Metaheuristiken zu investieren statt auf nicht-triviale Anpassungen zu setzen.

Die vorgeschlagenen Ansätze wurden mittels Prototypen oder in Simulationen untersucht. Insbesondere haben wir uns auf kleinere drahtlose Netzwerke mit IEEE 802.11-Funkschnittstellen und auf Sensornetzwerke konzentriert. Wir haben auch Untersuchungen mittels der WARP Software-Defined-Radio Plattform vorgenommen.

Acknowledgements

The work on my PhD thesis caused me various emotions, ranging from frustration, despair, and boredom to a feeling of sensation from a well posed question, an interesting idea, obtained understanding of a system, or a good solution. I really enjoyed this stage of my life, education and work, and would like to thank all my colleges, friends and even mere acquaintances who helped and inspired me.

First of all, I would like to dearly thank Professor Petri Mähönen by whom I had luck to study. He is one of the rare people to have a vision, a crisp mind, enthusiasm, kindness, and patience to pursue worthy goals, and invite others to participate. I thank my teacher for his guidance, inspiring ideas, and extreme support. I also thank Professor George Polyzos for accepting to become my second examiner and spending his time with my thesis.

I thank Janne Riihijärvi — a husband, a friend, a colleague. If not for this wonderful loving person my life, both office- and home-wise, would be much more dull, and definitely I would know much less. I thank my parents, Olga and Sergei, who always believed in me, shared their experience, and never let me to give up. I thanks my grandparents who always cared and loved. I thank my friends and colleagues, whose discussions, cooperation and even smiles, heavily contributed to the fun and the gained knowledge in these years. Some of them are Junaid Ansari, Mitia Medvedev, Marina Petrova, Andreas Achtzehn, Jad Nasreddine, Krisakorn Rerkrai, Zhou Wang, Daniel Denkovski, Christine Jardak, Frank Oldewurtel, Wasif Wasood, Arham Muslim, Yassine Antir, Md. Mohiuddin Khan, Christoph Mülder, Dalibor Mladenovski, Tobias Schulte, Stephen Grünewälder, Ossi Raivio, Alexandros Palaios, Xi Zhang, and Borislava Gajic.

Contents

1

Introduction

This thesis contributes towards bringing together insights and techniques from networking and software engineering for efficient design and run-time optimization of networked systems. We suggest, partially formalize through category theory, and apply to selected case studies recursive model-driven optimization as a methodology for development of networked systems and its elements. In particular, we employ this methodology to design a self-optimizing wireless home network and perform pre-deployment optimization of wireless sensor networks using an ontology-based framework. As part of the related research on scalable network modeling, we study and propose the extension for labeled network motifs as graph-based approximations for wireless spacial models and network dynamics. We also experiment with Hidden Semi-Markov Models for temporal network activity modeling based on power spectrum measurements. Additionally, we investigate metaheuristic search techniques, primarily genetic algorithms and simulated annealing, in application to network planning and cross-layer optimization. In particular, we consider behavior of these methods with respect to changes in optimization objectives, structure and sizes of the search state spaces, availability of network models.

1.1 Motivation and General Approach

Much of the classical networking research has focused on problems characterized by precise requirements, tight constraints and fixed sets of inputs. Examples are transceiver design for higher spectral efficiency, development of particular protocols, e.g., a routing protocol with robust convergence properties, or a CSMA/CA MAC mechanism that optimizes medium access for specific Quality-of-Service (QoS) metrics. Such types of problems used to be the most relevant to engineering, as complexity of networked systems was quite low, and networks were rather homogeneous both in term of hardware and software components, as well as user demands and provided services. However, much of the earlier research has not specifically considered the system-level aspects of network development and run-time management. These become increasingly important as systems grow both in size and complexity, and become more dynamic and heterogeneous, thus increasing costs of maintenance, as well as an amount of investments required to further boost their performance [1–4]. The typical approach, over-provisioning of resources, can provide only a limited and temporary solution to the growth in performance and robustness demands especially in the area of wireless communications [5]. However, it cannot be efficient against critical system failures, resulting, for example, from emergent behaviors or human factors [6–8]. Additionally, flexibility and evolvability as two of the main requirements to a networked system are receiving increasing attention, and these requirements should a priori be accomplished through design and managements decisions, not over-provisioning [9, 10].

Because of these considerations, we believe that it is promising to take a *coherent* and systematic approach on all system life-time aspects [11, 12], requirements, design, de-

ployment, evaluation, and maintenance (run-time management). This allows to simultaneously consider limitations and opportunities provided by different development stages, thus reducing the overall solution complexity, minimizing hidden risks, and increasing performance. In order not to further increase development costs and system complexity while solving a problem of this scale a certain *generic methodology* should be followed. It ought to be applicable at different stages of network lifetime and scalable to address both the system as a whole, as well as its arbitrarily small elements (e.g., fulfilling the tasks of development of a simple protocol or a particular feedback loop control mechanism). Of course, our proposal relies on research in several fields and extensive prior work. In our work we do not claim for excessive novelty, as many aspects concerning the proposed methodology have been considered in detail in different research areas. However, in most cases this was done in isolation from the other approaches and/or in application to a different problem domain. In this section we provide only the key references that enable our approach, with more details being given in other chapters of the thesis.

Furthermore, compared to the traditional networking research we should also consider socio-economical aspects of a network life-cycle that dictate additional incentives for system development. These are, first, the design reuse, i.e., re-employment of already developed components and designs. Second, the fractal or recursive development that allows for design at multiple abstraction layers with preliminary analysis and verification until all components are implemented at their final form. The third, the semi-automatic traversal of the development life-cycle from specification to implementation enables both overview of the project as a whole, and its specific modifications [13]. Compared to traditional software systems, networks are characterized by a heavy influence of operational environment and changing stakeholder goals. They also have much more complex maintenance or run-time management phase than typical software.

In short the considered problem can be stated as defining and applying a generic design and run-time management methodology that leads to better networked systems taking into account interests of all the players, including both network stakeholders and developers. This system-wide task can be formulated as an *optimization problem*. Different aspects of system-level design based on optimization has been researched and advocated in networking by, e.g., Doyle *et al.* and Chiang *et al.* [14–16], and software engineering community [17, 18], including its embedded domain [13, 19, 20], where it serves as a central part of the system-level design.

The optimization problem involved can be stated in terms of *goals*, a *system model* that defines state space of alternative solutions in a given operational context, and applied reasoning tools or *optimizers* (see Figure 1.1). There exist multiple metrics for satisfaction of network players, i.e., their goals, that can be formalized through, e.g., *utility* functions [21, 22]. Different actions can be carried out on a system ranging from parameter settings to deployment of new components. These form the optimization space of the problem. Finally, various constraints can be introduced that limit the space of available solutions, such as regulatory requirements, a necessity to use a legacy technology, or a presence of already deployed system parts that can only be adjusted to a limited extent.

The optimization problem in its full form can be very complex. For example, a cellular network of a single operator can contain tens of thousands of base stations serving millions of users, with a single large city usually containing some 500 GSM and UMTS base stations shared between several operators [23, 24]. Typically satisfaction of a user is measured with at least three to five Key Performance Indicators (KPIs) [25], combination of which forms the utility function of a user. The utility can be maximized through, e.g.,

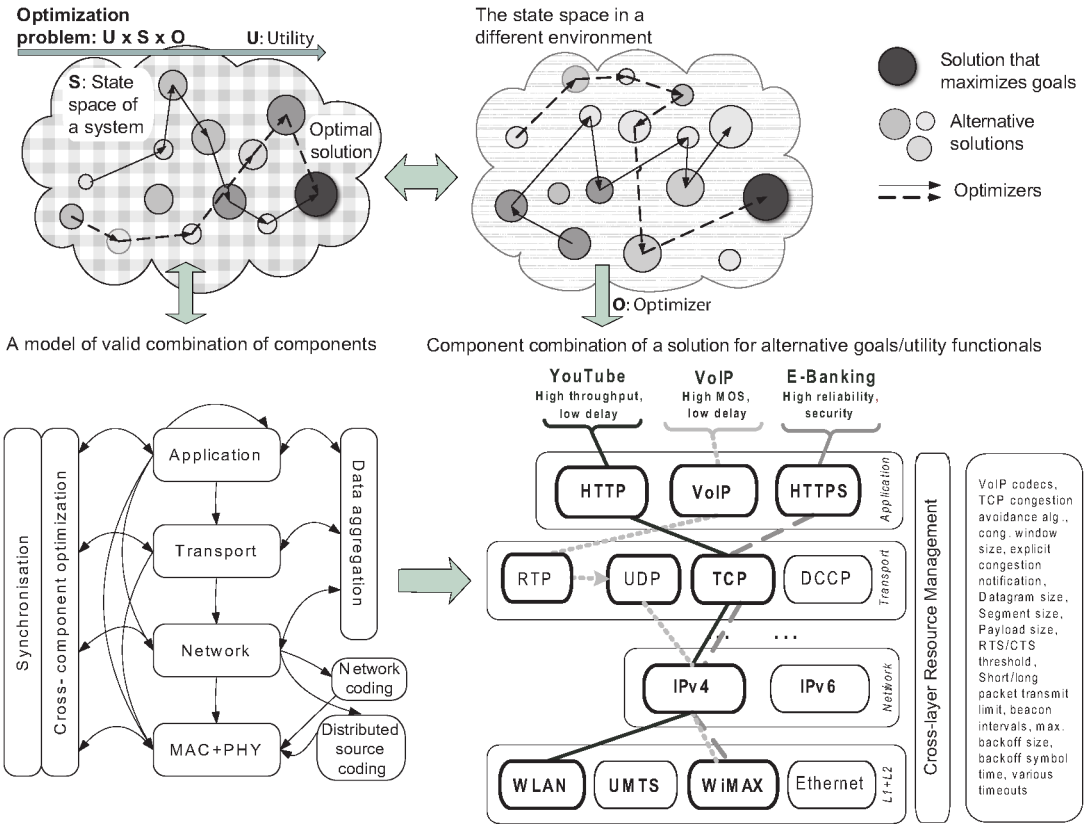


Figure 1.1: The schematic representation of some of the aspects of an optimization task, which on the high level of abstraction can be seen as a product space of a system model that defines a solution state space, a utility functional stating optimization goals and constraints, and an applied optimizer under computational and timing constraints.

tuning a number of predefined parameters on base stations and cellular terminals. This is a classical parameter-based optimization with all the design choices being made, and tunable parameters being well defined and very limited in number. However, this problem being approached straightforwardly in a joint manner already poses multiple complex goals as functions of the above KPIs, and has a dimensionality of hundreds of thousands of possible optimization actions (if only base stations and their parameters are considered). It explodes exponentially to hundreds of millions if user terminal settings are considered as well. The problem will become even more challenging if additional design choices are considered, such as options for deployment of new components, e.g., base stations or radio resource management software.

As we have seen in the above simple example, joint optimization problems can be of extreme complexity. *Mapping functions* can be used to make them more manageable. First, we can *abstract* the problem throwing away unnecessary optimization variables, considering various functionals of the original goals, as well as applying different models to capture relevant behaviors of the environment and the system. For example, instead of considering a complex functional of individual KPIs of users, e.g., the Mean Opinion Score (MOS) [26], we can utilize simplified aggregates of those, such as the call drop metric.

Representation and solution of the frequency assignment problem as the graph coloring problem is another instance of abstraction, with wireless nodes and their connections being approximated as graphs [27, 28]. Second, we can *decompose* the joint optimization problem into subproblems. For instance, the example on joint cellular system optimization can be effectively decomposed and abstracted into a number of distributed power control problems especially for networks operating using W-CDMA (Wideband Code Division Multiple Access) technology [29]. Here, the replacement of centralized collection of statistics with distributed deployment of simple control loops in practice achieves very good performance, low implementation complexity and reduced resource consumption. Third, for simple problems we can consider the *transformation* of the optimization problem into a close or equivalent representation that allows application of alternative reasoning methods thus, hopefully, resulting in better solutions. An example is the usage of complex (I and Q) numbers to represent the amplitude and phase modulation of a signal[1].

In majority of cases optimization tasks can be solved by applying various reasoning approaches, efficiency of which heavily depends on the nature of the problem. Therefore, in order to choose the most appropriate reasoning method, we need to obtain a better understanding on the optimization problem and its structure. For some limited scenarios we can use the "white box" approach, as we can obtain detailed and precise information on variables involved into the optimization task. In particular, for the physical layer (PHY) related problems physics and information theory form the foundation for understanding communication processes and provide tools to find effective abstractions, such as the Signal-to-Interference-and-Noise Ratio (SINR) metric. However, when problems span across multiple traditional protocol layers and involve third parties, "gray" or "black box" approaches have to be applied. These rely on no or limited amount of known information on the structure of the optimization task. The lack of data can be compensated by gathering measurements at the deployment sites or making educated guesses that often take the form of statistical models, which can be further applied either for immediate problem solution or its effective decomposition/abstraction. The main objective in the latter case is to find the minimum set of mapping functions that carry the largest amount of information on the original task, while reducing its complexity, so that the *optimality gap* is minimal. One of the key trade-offs that we can formulate based on the above discussion strives for balance between the amount of useful information on constituents of an optimization task and its complexity that effect the applicability and efficiency of available optimization methods. Less demanding the problem, simpler and faster are the means it can be solved with, but possibly with a cost on accuracy. In the extreme case the resulting gap might lead not only to sub-optimal performance of the system, but even to its critical failure. In other words, we face the trade-off on *amount of information and efficiency of its utilization*. This trade-off applies not only to the mappings of the optimization task, but also on the initial formulation of the problem, including the choice of tools and components that can be used for its solution, i.e. choice of variables and set of values these can take. Overall, an optimization task can be seen as a product, and, therefore, a space of trade-offs, of the goals and constraints, considered system models, and applied

[1] More formally the abstraction can be considered and many-to-one surjective relation. The decomposition is a function into a product space of other optimization tasks, which is not injective in general, but can be made so by introducing a component in the target space for keeping track of the potential information loss. These functions are non-invertible, but have their opposites, zooming (injective) and composition (surjective), respectively. The transformation is a bijection between different optimization problems. We further discuss on these mapping functions in Chapter 2.

optimizers. One should additionally take into account the costs imposed by exploration of these optimization possibilities incurred throughout development life-cycle, as these contribute to multi-dimensional constraints (or additional goals) of the optimization task. For example, the use of third-party components or other forms of *design re-use* principle can significantly low developments costs.

Optimization tasks in the networking domain are typically characterized by the presence of *uncertain or false information* of its objects and instances. *Information loss or disruption* can occur due to application of mapping functions, and initially inaccurate or poorly defined inputs. This influences the magnitude of the optimality gap for a particular optimization task, and might result in sub-optimal solutions, including critical system failures, e.g., as the result of an emergent behavior. There exist a number of mitigating techniques that were developed to combat such problems as the classical noisy inputs to the system [30] to malicious faults and Byzantine failure scenarios [31]. Further, an optimization task allows injection of new information into the system, e.g., development of new components, inputs, or derivation of more precise models. Generally, the information-centric aspect of decision-making is gaining popularity in networking and has been considered in, for example [32–35], while in part originating from economics and decision theory [36–38], where such important concepts as "Value of Information" (VoI) were formulated.

The optimization task and its mappings basically form a *meta-optimization* problem, which is characterized by a large and variable state space, applicable models and reasoning frameworks. The structure of such meta-optimization problems is very poorly understood and has been only studied in a limited fashion as, for example, part of the research on control of controls, or optimization of control loops for a plant [39]. Therefore, it is viable to first address the meta-optimization problem by generalizing and finding suitable variables, abstractions and models for this task utilizing existing pool of knowledge from networking and software fields in terms of, e.g., interfaces, components, technologies, and architectures. Next, when having insights from the field, it is important to study the benefits that relaxation of the above categories would bring in terms added flexibility, complexity and gains. The obvious example of this process is the traditional TCP/IP layering structure that is getting actively relaxed for the wireless environment as part of the research on cross-layer optimization [40]. As part of the incentive on improvement of understanding of the overall problem space, one can also study on a limited scale different joint optimization problems that were traditionally decomposed in a particular way (such as a classical PHY/MAC division). This can be done in a laboratory environment, e.g., applying exhaustive search, or metaheuristic methods [41] (see Chapters 4 and 5).

Clearly, application of mapping functions is an iterative process, as alternative abstractions of the optimization task have to be explored in case a desired solution cannot be achieved. During this process we should additionally consider alternative objective functions and constraints, as well as tools and models, which leads to multiple formulations of the optimization task and its subtasks, creating numerous abstractions of its variables. This inevitably leads to *recursion*, as it is likely that the exploration of only one alternative reasoning branch will not produce the desired optimization result. Recursion can also be viewed as one of the approaches to the formulated meta-optimization problem.

Summarizing, the focus of this thesis is the system-level design and run-time management of networked systems and their constituents. We believe that as networks become more complex and more demands are placed on them, more attention should be paid to this research direction in order to obtain coherent well-performing systems. We propose a

a model-driven optimization-centric methodology for these systems. The main aspects of this methodology are its formulation in terms of a meta-optimization task, and provision of general approaches to its solution. These approaches are based on utilization of mapping functions and models for the abstraction and the decomposition of the task, which are evaluated through the optimality gaps metrics. Consideration for those through category theory is our distinct contribution to the stated problem.

1.2 Major Contributions

Central themes in this dissertation are the tasks of system design and run-time management using optimization-driven reasoning, briefly outlined in the previous section. In the rest of the thesis we further specify, explore theoretically and in practice different aspects of the suggested approach. More specifically, we make the following main contributions.

Model-driven optimization for wireless networked systems. Based on the related work and our experience we propose a recursive model-driven optimization as a methodology for development and maintenance of networked systems. First, we formulate an optimization task and its constituents. Then we state that generally an optimization task should be treated from the meta-optimization perspective, i.e., it should be decomposed and abstracted to simple model-based representation. As these operations are generally bound to information loss compared to the original formulation, we also consider some of the involved risks. We argue that it is desirable to explicitly consider models utilized in a formulation of an optimization tasks as they, among other factors, bound the achievable performance, i.e., define the optimality gaps. Finally, we argue that category theory forms a natural framework for formalizing the relationships between the different optimization problems and their alternative formulations that arise in networking. Such essentially graph-based formalization extends the optimization decomposition research that has become popular for formally deriving protocols and deployment solutions from network-wide optimization problems. We discuss the suggested methodology on selected examples from wireless and fixed networking, including such popular architectural abstractions as layering, network planning tasks, and utility-based queuing [42].

Ontology-based optimization for Wireless Sensor Networks (WSNs) and home appliances. We apply the proposed methodology to realize an ontology-driven framework that aims at parameter- and component-based composition and configuration of protocol stacks for user-defined WSN scenarios [43–45]. The framework automatically invokes several modeling and decision-making modules to reason on parts of the task in the most appropriate form. For example, ontology-based reasoning is used to decide on alternative compositions of behavioral models and their mappings to the actual components. At the same time, performance results from multiple experiments are compared through the relevant database and utility-based results processing. In the case multiple experiments need to be done, e.g., to find an efficient parameter setting combination, an exhaustive search can be avoided by applying heuristics-based search that is based on probabilistic graph models [46]. The same ontological framework, although simplified, is employed for service composition for white good home appliances [47, 48]. Additional relevant works by the author include [49–51].

State-space Exploration and Optimization with Metaheuristics. We study the application of metaheuristics techniques for wireless network planning and optimization, and compare several popular "out-of-the-box" methods commonly applied to such problems in an isolated manner. Metaheuristics are widely applied for solution of networking problems, such as network planning planning, where the structure of the optimization state space is irregular and fit poorly to application of standard optimization techniques. We raise and answer the question of how exploitation of the knowledge on the structure of the problem and its search space can enhance quality of the search. We also incorporate selected probabilistic models, some of them being trained online, to enhance the selected metaheuristics, namely simulated annealing [46, 52–55]. The resulting methods enable to create utility-parameter causality estimates that enable effective cross-layer in Wi-Fi and wireless sensor networks.

Modeling of wireless networks with labeled network motifs and hidden semi-Markov models. We develop and study models based on graph approximations that can considerably reduce the dimensionality of the optimization task. These models allow capturing selected aspects of the operational context and represent the respective performance-parameter relations. We propose to use directed labeled networks motifs to capture principal components influencing network behavior taking into account both network dynamics [56] and its spatial structure [57]. We carry out the study on the example of ad-hoc networks. Our results show that only a limited number of subgraphs occur frequently in realistic spatial network deployment models. Moreover, network dynamics with respect to adjustable protocol parameters can also be captured using only a small number of edge labels. Our findings indicate that the proposed graph-based models are robust, and can be successfully employed for autonomous context identification, improvement of online optimization algorithms, and advanced network planning. We also investigate the applicability of Hidden Semi-Markov Models (HSMMs) for characterization of temporal context using empirical power spectrum samples gathered in the indoor environment [58, 59]. Based on our study we suggest and implement on WARP boards a MAC protocol that learns and utilizes these models online for efficient spectrum utilization.

Design of a self-optimizing system for wireless home networks. Finally, we apply the proposed recursive model-driven optimization methodology for development of the Home Cognitive Resource Manager (HCRM) [52]. The system aims to be maximally user-independent, account for multiple shakeholder goals, and provide cross-layer and radio-resource self-optimization functionalities. At the same time it should be evolvable and enable further experimentation in the field of self-optimization for wireless home networks. We specify the detailed requirements for such a system, choose an appropriate architecture, derive its detailed design, and, finally, implement the prototype. We opt for the agent-based design and the cognitive resource management architecture [60]. The resulting system features such functionalities as meta-optimization, multiple specific optimization algorithms including machine learning based approaches, control channel handling, a centralized policy server, interface virtualization, and utility-based reasoning [61–63]. We show that the HCRM is performing well in dynamic wireless environments. It can operate both on COTS Wi-Fi hardware, as well as software-defined radios.

1.3 Dissertation Structure

This thesis is organized in seven chapters. In this introduction we gave the general motivation for our work, sketched the proposed methodology and summarized the main contributions of the thesis. In Chapter 2 we describe in detail and partially formalize our methodology based on recursive model-driven optimization. In Chapter 3 we apply the proposed approach to create an ontology-based framework that partially automates design of wireless sensor networked solutions. In Chapter 4 we study several metaheuristic search techniques in application to the problems of network planning and run-time cross-layer optimization. We investigate the influence of the search state space, the stated optimization goals and the available network models on the performance and applicability of these methods. In Chapter 5 we study graph approximations, namely directed labeled network motifs, as models for capturing not only geometrical and spatial network structure, but also its dynamics. We also use HSMMs to estimate temporal ON/OFF activity patterns of wireless nodes based on observed power spectrum (RSSI) samples. In Chapter 6 we use the proposed methodology to design and implement the self-optimization system for wireless home networks. Finally, Chapter 7 concludes the thesis and makes recommendations for the future work.

In Chapters 3 and 6 we concentrate on system design aspects, whereas in Chapters 4 and 5 we mostly focus on network modeling. There exist prototype implementations of the most of systems and approaches described in this thesis, in particular of those discussed in Chapters 3, 4, and 6.

2

Model-driven Optimization for Wireless Networked Systems

The goal of this chapter is to describe and discuss in detail the proposed *recursive model-aware and optimization-driven methodology* suitable for both development and run-time management of networked systems. We start with a recap on the motivation of the work, which, in short, is the growing network complexity in answer to demands for higher diversity and quality of provided services. Then we outline and formalize the methodology that is based on the view of a network life-cycle and its constituents as an *optimization task* that needs to be formulated and solved by *modeling* the relevant aspects of the reality. We propose stating the formal grammar for the methodology using *category theory*, and describe key points from the extensive related research that led to definition of the methodology as it is. We use insights from the literature, as well as simple networking examples to illustrate our case. The high-level overview of the proposed methodology is given in Figure 2.1.

2.1 Our Focus

Most network optimization problems are too complex and/or poorly understood to be solved directly. Therefore, typically, splitting the solution process into two phases is beneficial. First, the original task has to be systematically *mapped* into a set of interconnected and simplified problems, which can be solved separately and contribute to the solution of the overall problem. That is the *meta-level* optimization phase. Second, each of these subproblems has to be addressed separately. This is not always straightforward. Often there exist a number of alternative formulations of the task that influence accuracy of the obtained solution and the imposed overhead. For example, the system model, the pursuit goals and constraints dictate applicability of the optimization methods.

The main challenge of the meta-level phase is to formulate and represent an original problem into a form that results in sufficiently good solution at a cost of available resources (computational, economical, and, even, political, if we consider for example Internet and cellular policy regulations). For this we propose to recursively apply models as a logic behind applications of the mapping functions, such as decomposition and abstraction. The recursiveness is dictated by the fact that we cannot fully predict solutions that would be obtained to each individual problem. For example, if a subproblem cannot be solved in its original formulation and has to be enhanced with additional inputs and assumptions, then the reformulation of the whole meta-optimization chain may be required. In this sense the meta-optimization essentially facilitates a top-down approach to the optimization task, and solutions of individual subproblems give the bottom-up feedbacks.

In order to practically apply the optimization-driven methodology there should be criteria to decide on viability of alternative mapping actions as part of the meta-optimization process, individual problem formulation, and their solution assessment. Unfortunately,

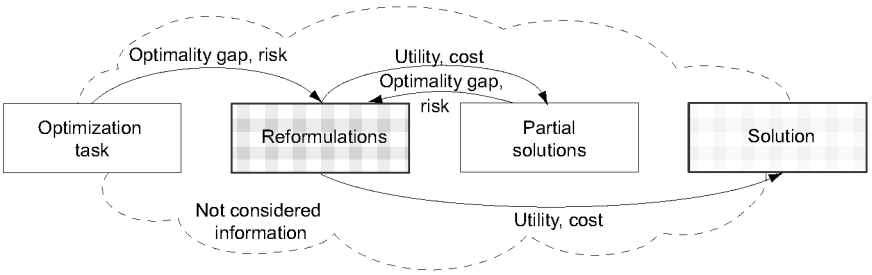


Figure 2.1: The general flow of the optimization-drive methodology.

currently there exist only very limited amount of work in this direction. In this chapter we provide the initial discussion on the topic, try to formulate and apply in practice the criteria that seems to be appropriate. This criteria is based on the notion of the *optimality gap* between different solutions and underlying problem formulations that can be seen as a function of the utility, the cost and the risk incurred by abstracting the problem. The latter factor is closely related to information contained by a problem statement and its loss during further reformulations.

In this thesis, and especially in this chapter, we attempt at defining and applying an appropriate generic design and run-time management methodology for networked system design. We do not claim for excessive novelty of our work, as in many research domain there exist approaches that aim at effectively formulating and solving similar intergraded problems (or their significant parts). Among them are operations research, different types mathematical optimizations, machine learning, control and decision theory, and many more. We merely want to combine, and, if needed, adjust, existing concepts, approaches, metrics and variables that, from our point of view, are useful for wireless system design and run-time management.

As an overall idea, we consider the network lifecycle as a huge joint optimization task that cannot be directly solved, and needs to be decomposed, approximated and further modeled at various levels of granularity in the coherent manner. The combination of solutions of the resulting optimization subproblems creates the desired well-performing networking system. We believe that such integrated approach to network development would avoid statement of irrelevant design and management problems, and, at the same time, allow identifying promising tasks, where the spent efforts are likely to bring the highest gains. The incentive for optimization task decomposition has been also highlighted in regard to both software and network systems [4, 15, 64–70]. The dangers of improper decompositions, such as the emergent behaviors and the long-term state dependencies that can result to poor system predictability, were discussed, e.g., in [6–8].

Clearly, the consideration of only technical aspects of networked system construction is not viable, though in this thesis we mostly consider only these criteria for evaluation of the proposed solutions. Many of the promising technical solutions cannot get implemented due to economical, political, and social considerations. Prompt examples are spectrum management policies that cause suboptimum spectrum utilization, but are economically or politically feasible for most of the stakeholders. Introduction of the secondary spectrum usage, facilitated by the cognitive radio research, is to a large extend delayed due to non-technical considerations of the major players [71]. The long-existing instability problem of BGP routing is also present due to operator privacy constraints rather than

absence of the technical proposals [72]. Similarly, the success stories of Wi-Fi and GSM, but fading of the WiMax were not purely based on the technological characteristics of these technologies, but also their ability to satisfy economical and social demands of users and operators [73, 74]. For example, WiMax could not directly support comfortable billing system for the operators. However, in this thesis we mostly consider the technical criteria for evaluation of the proposed solutions. Networked systems are typically also imposed heavy evolvability and adaptivity requirements.

Wireless networked systems[1] are focus of this thesis. These networks still pose a lot of research challenges, and are already economically successful. Their popularity results in denser deployments and leads to competition for constrained wireless network capacities, which reduces possibilities for over-provisioning, and increases a value of careful and scalable network design [1–5].

2.2 Methodology Description

We propose to treat a whole development lifecycle of a networked system, including its requirements specification, design, implementation, evaluation and maintenance (run-time managements) stages, as a *joint optimization task*. Ideally this approach allows exploiting the widest range of opportunities for development of a better system [75]. This methodology, relying on principles of optimization, mapping, modeling, and component based organization, is very generic and can be specialized, e.g., applying particular meta-models, to fit such well know principles as recursive development of software systems [76–78], model-driven engineering [19, 79–81] and goal-based software design [82–85], platform-based software design for embedded systems [67], and network optimization decomposition with horizonal and vertical layering [15].

We enrich the original approach with explicit considerations for applied modeling techniques and mapping operations, considering information-centric decision criteria, which in practice we capture as a utility- and cost-based optimality gaps with the consideration of the risk of the wrong estimation of these values. We propose applying *category theory* to formalize our approach, as well as general optimization-driven network problem solving. For this purpose, we give below a short introduction on the necessary theoretical foundations of category theory, as well as outline a research program for developing categorical definitions of the key concepts discussed here.

2.2.1 Ingredients of the Optimization Task

In this section we summarize the key ingredients of optimization tasks arising from the consideration of networked systems. We formalize this discussion using techniques from category theory, *defining* an optimization task as a “product” of the major individual ingredients, namely system models and their respective constituents, optimizers, utility functionals expressing the goals, and additional metrics consisting of, e.g., the cost and the optimality gap. This decomposition is not, of course, the only possible. We also do

[1] In relevance to our work we believe the term “networked systems” is more accurate than just “networks”, as it highlights that we consider networks as a collection of interconnected elements. Our aim in this work to create approaches that are scalable and can be applied not only to whole networks, but also its elements. Further we will use the terms “networked systems” and “networks” interchangeably as synonyms.

not include some of the minor aspects of the optimization problems into our selected decomposition. Nevertheless, we strongly believe that the collection of ingredients outlined here is very natural for a majority of optimization problems encountered in networking.

Compared to traditional software systems, network are heavily prone to influence of the outside world, in other words their operational conditions. This influence can be divided into two main categories, the inputs, including goals and constraints from network players, especially the stakeholders, and the environmental effects that distort the information flow in the system. The gray zone is alteration in signaling between the system and the players under influence of the environment. This knowledge can be highly useful in analysis and performance prediction of a system, provided that exist enough inputs and models to reliably capture this knowledge. Other networks that share the same operational space can both act as network players, e.g., require relay services from our system, and cause changes in the environment, for instance, by introducing wireless interference.

Careful modeling of the outside world is one of the keys to the optimized system behavior, which however, comes at a cost of additional complexity. In our work we consider that the world cannot be directly influenced by a system, it can only passively observed. However, definition of the world might change during system life-time or even as a result of mapping actions, e.g., decomposition of the optimization task. For example, if at the requirements stage we ask the user to redefine her requirements/goals to the system, this can be treated as an actuator input to the "component" user, which might give as an output a different task that will lead to the simpler system development. However, at the later stage these goals cannot be changed, therefore the user becomes part of the "world". Similarly, during the decomposition process certain parts of the system can be viewed as stakeholders to relation to other components. For example, classically the application layer protocols are treats as "customers" to the transport layer.

Execution of the system model that incorporates the aspects of interest on both the operational environment and the system itself (i.e., the whole "world") results in the state space of alternatives for the optimization process. The state space can be viewed as a set of solutions that carry specific characteristics, which change depending on the state of the environment and the system. Depending on the task these alternatives can be perceived as black, gray or white boxes, with each of these views allowing for different number of further optimization opportunities. For example, commercial, contrary to open-source or self-developed, elements often cannot be decomposed and their performance can be predicted only on the basic of direct measurements or high-level analytical models. However, the developments costs of these elements are low, while reliability is typically high.

We take a generic *component-oriented* view on networked systems [64, 65, 86]. In this context, we treat a system as a dynamic collection of components and their configurations linked in a specific way, i.e., via a process for which internal and external information flows are prone to influence of the external operational environment. Another important part of a component, besides its configuration, is the *interfaces* that both on logical/behavioral and structural levels dictate compatibility between the component. Combination of components and processes create a solution or upper-layer component. As discussed later depending of the level of abstraction a component might be modeled to represent only its behavioral or structural properties [87]. As stated by [21] an alternative to configurable component-based definition of the system model and the resulting search state space is the parameter-based optimization. We further discuss on component- and

parameter-based design in Chapters 3 and 6.

The optimization process itself can be performed using a number of alternative methods, which we name *optimizers*. Optimizers perform the search functionality on a system model or over the state space of possible solutions it provides. They can be also executed to create new solutions, e.g., by combining components in a new way following a particular model, i.e., create a new system model. As we demonstrate in Chapter 3 optimizers can indicate the necessity for new components or modification of existing ones. Recently search methods, mostly metaheuristics-based, have gained attention in application to software development [18]. They can be regarded as optimizers as well. Protocols perform optimization to maximize such metrics as, e.g., throughput [22], i.e., they execute specifically designed optimizers for these purposes.

Summarizing, both, for the original optimization task, as well as for some of its abstractions we a likely to encounter one of the following situations. (a) Existing variables cannot provide a satisfactory solution to the problem [large amount of information that cannot be effectively utilized]. (b) We do not possess enough information to reason on particular object, e.g., we do not know how a component behaves in particular operational conditions in conjunction with other components [lack of information or uncertain information]. (c) The information provided about the object is not correct or conflicting, e.g., its characteristics are obtained applying an erroneous mathematical model, or results of testbed trials and mathematical modeling are highly diverse [false information]. It is a crucial task to identify these various types of information-centric critical points and the take appropriate actions to mitigate these problems. This problems to a large extend are dealt with as part of the meta-optimization process.

2.2.2 Optimization Goals and Performance Metrics

Recapping, we want to find such a system *solution* that would optimize for *goals* stated by multiple network stakeholders for a given range of *operations conditions* under imposed *constraints* (see Figure 2.2). Goals fulfillment metric is measured through *utility* functions. The variables of this top-level formulation can be perceived as classes that encapsulate other simpler entities. It is often useful to reason on components of such a composite metric increasing dimensionality of the problem rather than using a joint functional. Obviously, there also exist constraints in terms of, e.g. regulations, hardware and software capabilities levels, as well as maximal costs, on levels of achievable performance, which get accounted for either as part of the goal statement, a system model, or even an optimizer (e.g., timing and computational boundaries on solving the task). Additionally, as a priori in most cases the solved optimization tasks are built on the *representation* of the world, not the world itself, there exist risks that deployed/executed solutions in reality would behave differently to expected, e.g. we will encounter emergency (mis)behaviors. We believe that it is also useful to explicitly acknowledge this risk, and do not incorporate it inside into other metrics. The risk metric is helpful to account not only for deployment uncertainties, but also for uncertainties connected with reusing of solutions for other optimization problem inside of the already formulated task as part of the meta-optimization approach, e.g., combining results of disjoint optimizations into one solution. One should not forget to also consider the *costs* of development and management (e.g., information gathering) while considering alternative approaches. For example, use of third-party components or other form of design re-use may significantly lower the developments cost.

Variables of the task can take a limited set values that define the optimization state

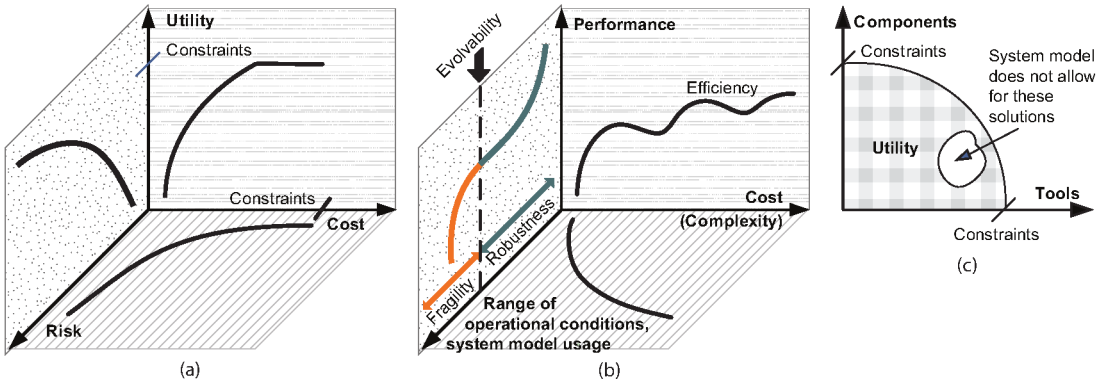


Figure 2.2: Illustration of the influence of the main variables of the optimization task.

space. There is an obvious tradeoff between network performance, its overall complexity, and the range of supported operational conditions, see Figure 2.2, which can be captured through the *robustness* and the *fragility* metrics [14, 88–90]. As the optimization task is generally *dynamic*, i.e., prone to run-time changes, the fulfillment of the *robustness* requirements has to be often addressed either through over-provisioning, or by enabling *evolvability* of a system. System complexity buys a certain increase in the utility as well as in the range of operational conditions, however this dependency is not linear and tends to saturate, or even degrade until overcome by upgrades and deployment of additional components that result in higher complexity and increase performance [7, 8, 88, 89]. Basically we observe the variant of the robustness/complexity spiral [88], which can be reformulated in terms of desired utility, range of supported operational conditions and complexity. The fragility region, see Figure 2.2, denote the range of operational conditions and the virtue for which the system was not designed and where it cannot operate well. Here the chances of unpredictable, mostly poor behavior, rapidly increase, which might completely paralyze the system [14]. One of the key approaches to increase robustness is to implement a particular functionality multiple time at different system levels. For example, network reliability can be improved through implementation of ARQ techniques across different layers of the protocol stack. In biology and other related fields, this phenomenon is called *degeneracy* of the respective architectures.

Doyle et al. [14] linked in their *Robust-Yet-Fragile (RYF)* principle the complexity, robustness and fragility aspects of a system. The RYF principle states that only a certain amount of complexity is required to keep the system robust under a given range of operational conditions, with the fragility risks increasing hopefully only outside of these operational boundaries. However, if we consider evolvability of a networked system, its constant development in order to meet increasing user needs, it becomes impossible to stay at the portrayed maximum, even if it is reachable (which, in most cases, is not the case). The tradeoff between the complexity and robustness becomes a spiral, where the deployment of new services or scale expansion increases the complexity and/or decreases the robustness of a system resulting in further development changes [88]. There exist a number of theoretical works [14, 15, 88, 91, 92] that utilize the RYF principle for better understanding the mathematical mechanisms underlying Internet and their exploitation for the more robust design of the network elements.

Generally, the more demanding is the stated goal the more costly typically would be

the corresponding solution. On the other side easy-to-achieve goals would result in to suboptimal systems that do not exploit many of the performance improvement possibilities. Additional danger lies in fuzzy goal statement, which might result in wrong design decisions. The same danger is imposed by distorted formalizations of the goal, as a result of reformulation of the optimization task. Yet another complication lies in dynamics of user definition/perception of the utility, where user levels of satisfaction can vary for the same Key Performance Indicator (KPI) values based on his prior experience from a system or services offered by the competitors [93]. For example, users tend to perceive much more strongly drops in the experience levels of services compared to the similar levels of quality increase. They also tend to quickly get used to good performance, even it is more than they pay for, and compare performance drops to these recent utility levels rather than absolute ones [94]. In networking these aspects are partially studied in the domain of Quality-of-Experience (QoE) and Quality-of-Service (QoS) research [95–97].

Though in practice utilities are governing almost any network functionality, there exist several reasons why they have not been yet widely adopted explicitly in network management. Utilities are difficult to design to accurately reflect interests of both users and providers. Their particular forms, e.g., multi-objective, non-convex, are not suitable for all optimizers. The tradeoffs between the complexity introduced by utilities and additional gains compared to conventional simple metrics, like QoS classes, have not been totally proven. Rather there exist strong indications that utilities are the right high-level goal formalization, which can benefit from the hierarchical mapping, e.g., in same QoS classes [42]. Generally any stakeholder cares only for a limited number of metrics. For instance, for a user, after Odylzko [98], the three primary metrics are *cost*, *time-to-wait*, and *quality-of-content*. The latter metric includes considerations for both the content and its quality. Other metrics, such as power consumption that is particularly valuable for wireless users, do not have a decisive value until the primary objectives are sufficiently fulfilled. A provider might adjust user utilities by influencing the functions of the above metrics, for example, by defining the shape of the cost function or normalizing the time-to-wait expectations to ensure fairness.

2.2.3 Mappings and Meta-optimization

The joint original formulation of the optimization task might be unsuitable for obtaining a satisfactory solution. It can be too complex to be directly solved, lack, or have false inputs. The optimization task should be *reformulated*, e.g., enriched with details or abstracted, until it can be solved either autonomously or by a human. Solution of the original task through reformulations can be regarded as a *meta-optimization* process, which operates on five basic *mapping* functions: *abstraction/zooming, decomposition/composition*, and *transformation* (see Figure 2.3).

Mapping functions alter the range of values variables of an optimization task can take, and change the list of considered variables (this also includes also the models that variables store and similar information). Decomposition enables representation of the task as a set of subtasks that can be solved separately to be utilized later as part of the original formulation. The composition function is opposite and aims at combining several tasks and their solutions towards a joint optimization. Abstraction actions allow omitting certain variables or their selected values during the reformulation process. Relaxation of the SINR maximization task that allows for application of convex optimization is one of the examples of the abstraction process (see Figure 2.4). Zooming is the opposite

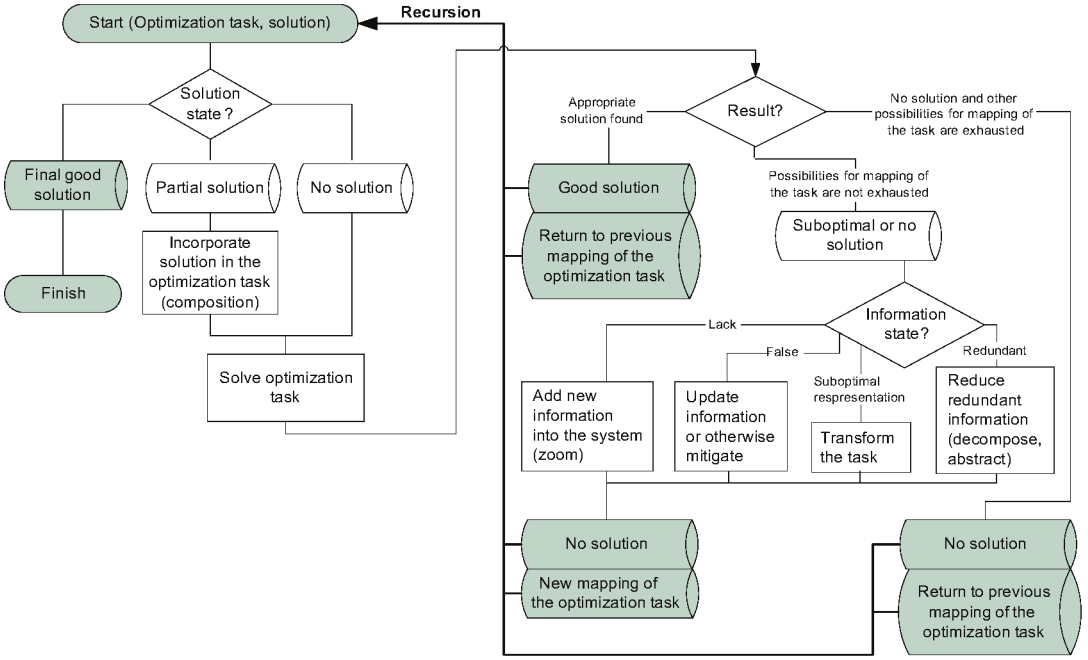


Figure 2.3: Flowchart of the methodology with the meta-optimization component.

function. Decomposition and abstraction are typically used as a part of the top-down design approaches, whereas composition and zooming are often needed for the down-top development strategy. Transformation can be seen as a combination of abstraction and zooming, where variables and their values are not omitted or added, but transformed to a different representation.

Clearly, blind application of the mapping functions to the optimization task creates an NP hard problem. Therefore, first, there should be metrics to quantify the usability of the current abstraction of the optimization task. We propose to utilize the approach based on set of metrics from economics and decision making theory, such as "*Value of Information*" (VoI), "expected value of perfect information", "expected value of sample information", "expected cost of information", etc., [36, 37, 99], part of which are utilized in networking [32–34]. Each abstraction can be characterized using three basic metric discussed above, utility, cost and risk. Generally, one may define *optimality gap* as a measure of difference in the utility or the cost achievable between two abstraction. Similarly, the *information gap* identifies the amount of information lost/gained due to application of certain mapping functions on the optimization task that might lead to change of, risks, e.g. evaluation/deployment, for a particular task. Based on the gap one may decide on application of particular mapping functions on the optimization task and its variables.

One should mention that to the cost function one might attribute not only the resources need to solving a particular abstraction, but also savings in costs that other relevant abstractions might gain in the solution process, i.e., the cost might be a joint metric to other optimization tasks. This correlates well with the known statement [100] that "information is costly to produce but cheap to reproduce" or even classical division into capital and operational expenses. For example, a particular task is formulated to require to development of reusable components with well defined interfaces. If these components

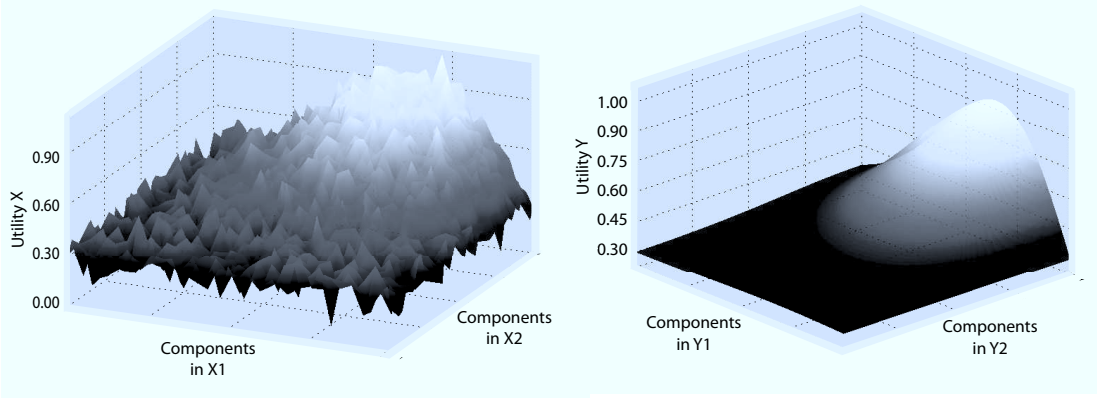


Figure 2.4: Goals vs. components as a search profile. Utility Y is a smoother variant of Utility X. Utility X requires for metaheuristics search due to performance profile of its state space, whereas Utility Y supports for fast convex optimization, which comes at a price of lower optimum achievable on the state space Y compared to the X.

are not utilized to solution of other problems their costs are high. However, if they will be widely reused the overall development costs decrease.

2.2.4 Models and Meta-optimization

Everybody is implicitly aware that we operate not on the actual system and the target operation environment, but their *representations* that is closely related to the notion of modeling. For us *model* is a very broad term that includes not only abstract representation of, e.g., a particular protocol or problem, but also the inputs and outputs used to interact with the model or build models-of-models (meta-models) [79]. Principles from Model-Driven Engineering (MDE) [83, 101, 102] can be applied for design and management of networked systems by recursively modeling their different aspects (also from different research perspectives) and reasoning on these models.

Models can be perceived as established meta-entities that guide the optimization process, stopping it from becoming an exhaustive search (as an extreme of the combinatorial optimization problem) with a vast amount of states. They guide, i.e., give constraints and devise logic, behind application of the mapping functions. Examples of high-level models are different decompositions of development lifecycles, architectural principles, such as layering, and distinct architectures. For instance, development of the architecture of the system, design of its components, their implementation and evaluation can be viewed as decomposition of the original problem following the model that corresponds to the simplified waterfall development life-cycle [11]. One may state that various forms of network development life-cycles is nothing, but a particular form of decomposition of the optimization task that proved to be useful. We can also apply horizontal and vertical layering as models to perform decomposition the optimization problem [15].

Models also denote application of various reasoning methods and connected representation of information of the optimization task, such as statistical and Bayesian models [103] for casual and world modeling, information gap decision making [104], convex optimization, etc. Views on a networked system as an interconnected set of configurable components or an element with a number of adjustable parameters are also models.

We continue the our discussion by providing an example on how inadequate modeling can lead to a network failure and/or suboptimal performance. Taking a view on a networked system from only perspective of one research (sub)field without careful considerations might lead to wrong conclusions on its organization and behavior. For example, traceroute [105] observations on the Internet topology lead to the erroneous conclusions on the scale-free structure of core of this network, and the beliefs that there are few key failure points in this system [106]. Later it was shown by Willinger and others that this conclusion does not hold [107]. Based on the hardware engineering considerations and additional measurements and these authors in [108] suggested their own more precise metric capturing the self-similarity in the behavior of the core.

The danger lies not only in lack of interdisciplinary view, but also in wrong scale or type of modeling applied to the particular problem. For example, the research on cross-layer optimization [109] aims to mitigate limitations of the TCP/IP layering model in application, for instance, to wireless networking. The last, but not least, is the famous example on BGP oscillatory behavior [110]. Due to lack of information exchange the Autonomous Systems are allowed to independently define such BGP policies so that no convergence to stable routing occurs. BGP policies also allow for contradictory statements in their configuration files due to the limited support for consistency checking [111]. These problems can be represented and resolved through the modeling-driven approach, as suggested by [112, 113].

Summarizing, on the meta-level our basic objects are optimization tasks. These objects are characterized by the utility, cost and risk metrics and their estimates. They are acted upon using mapping functions, which application patterns are encoded by models. We end up with an *recursive* process, where solution of joint optimization task is treated as a meta-optimization process which aims at mapping of the original problem into, generally, a set of effectively solvable subproblems. Their solutions then need to be evaluated and, if viable, incorporated into prior abstractions, in the end achieving the desired overall solution of the stated problem. As mentioned, the optimization task can be formulated to a network as a whole, as well as for its arbitrary constituents and subproblems arising during the development life-cycle. Now we work towards formalization of the proposed viewpoint using category theory.

2.3 Towards Grammar of Meta-optimization with Category Theory

In this section we work towards formalization of the concepts introduced above. This is required to enable application of mathematical reasoning methods and exploit specific optimization possibilities arising through partial automation of the development life-cycle of a system [66, 67, 114]. Otherwise application of the suggested meta-optimization methodology would be limited only to informal human reasoning.

We suggest applying *category theory* [115–117] as a grammar for describing the problems of network system design and maintenance that can be considered as optimization tasks. Our work is inline with the recent rapid expansion in applications of category theory outside of the purely mathematical domain, for example, to system biology and human cognition [118–120], physics and its connection to topology, logic and computation [121], and, most interestingly to us, software engineering [122–124]. Additionally the recent work by Kozen et al. carries a promise for automation of reasoning on categorical constructs [125]. Further in this section we first shortly introduce the basics of category theory. Then, based on the relevant prior work [124, 126–134], we outline the

categorical approach to meta-optimization in the context of network system optimization. Finally, we discuss the different categorical constructs arising from the meta-optimization formalization in more detail.

2.3.1 Introduction to Category Theory

Category theory is a branch of mathematics that aims to find commonalities and analogies between many of its areas, provide abstract constructs and definitions that will be applicable to many of the mathematical fields[2]. It is often regarded as the "mathematics of mathematics", as it builds a systematic framework that allows relating different areas of mathematics. Category theory can be seen as a tool for deriving a language to describe and unite in a single framework similar phenomena even though those might require different algorithmic approaches and mathematical apparatus to be described and reasoned upon. It provides the means to re-formulate problems from one area of mathematics to another, which might simplify the solution process. Additionally, category theory offers convenient graph-based representation of its operations, which makes it comfortable to construct and track complex dependencies between objects of interest.

2.3.1.1 Constituents of a Category

Category theory relies on the key abstraction appropriately called *categories*. A vast majority of mathematical structures can be seen as categories or their constituents. A category consists of two classes, *objects* and *morphisms* that define various relationships between objects. A set of all possible morphisms between objects A and B is denoted as $Mor(A,B)$. There exist several notations used in category theory. The one reminding small diagrams we find especially comfortable. For example, the expression $A \xrightarrow{\alpha} B$ or $\alpha : A \to B$ denotes the morphism α from the object A (called as a source or a *domain*) to the object B (named as a target or a *codomain*).

Morphisms must satisfy two axioms. First, morphisms should support the *composition* operation that is required to be *associative*. For instance, the morphisms from objects A to B, B to C, and C to D are denoted as α, β and ν, respectively. The associative relations between A to D in terms of morphisms can be written as

$$(\alpha \circ \beta) \circ \nu = \alpha \circ (\beta \circ \nu) \tag{2.1}$$

Second, for each object A there must exist a special morphism called *identity* that is typically denoted as id_A. The identity on A id_A being applied on the morphism α results in the same morphism α.

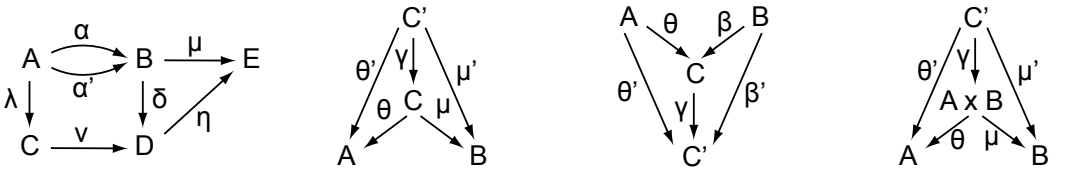
$$\alpha \circ id_A = \alpha \tag{2.2}$$

The order of operations in the category theory is important. For example, if θ is the morphism from C to A then in order not to affect the original mapping θ the identify i_A should be applied as follows,

$$i_A \circ \theta = \theta \tag{2.3}$$

There exist a number of special types of morphisms, such as monomorphisms, epimorphisms, bimorphisms, etc. For us the most relevant is the *isomorphism* that is a mapping $\alpha : A \to B$ so that there exists $\beta : B \to A$ with $\alpha \circ \beta = id_A$ and $\beta \circ \alpha = id_B$. The operation β is the *inverse* morphism of α. An arbitrary morphism has only one inverse, if any.

[2]For the introduction of category theory we heavily rely on the material provided by R. Geroch [115] and less extensively on the works of M. Fokkinga [116] and J. Adamek et al. [117].



a) Example b) Product c) Direct sum d) Catertesian product

Figure 2.5: Sample diagrams in category theory (adapted from [115]).

To graphically represent how morphisms are applied on objects the *diagrams* were introduced. A diagram is said to *commute*, i.e., to be valid, if any two objects in the diagram, and any two morphisms between those objects are equal. The example diagram in Figure 2.5a commutes if $\alpha = \alpha'$, $\nu \circ \lambda = \alpha \circ \delta$, and $\eta \circ \delta = \mu$. Many statements in category theory can be formulated and illustrated using diagrams that commute.

Further we give two other definitions relevant for our discussion. Figure 2.5b illustrates the notion of a *product* of objects A and B that is the object C together with the morphism θ from C to A, and the morphism μ from C to B. For a definition of product to hold there should exist any object C' with the morphisms θ' ($C' \xrightarrow{\theta'} A$) and μ' ($C' \xrightarrow{\mu'} B$), so that there would be a unique morphism γ so that the respective diagram in Figure 2.5b commutes. The definition of a *coproduct* or a direct sum is somewhat opposite. Figure 2.5c shows the respective diagram. Here the direct sum of the objects A and B is an object C with the morphisms θ ($A \xrightarrow{\theta} C$) and β ($B \xrightarrow{\beta} C$) holding if the following property is fulfilled. There exists any object C' with morphisms θ' ($A \xrightarrow{\theta'} C'$) and β' ($B \xrightarrow{\beta'} C'$) so that a unique morphism γ could be defined as shown in the commutative diagram in Figure 2.5c. A product, similarly to the direct sum, consists not only of the objects, but the morphisms as well. Clearly, the definitions of the product and the direct sum can be applied not only to two, but arbitrary many objects. Generally a category is not required to satisfy a property that any of its two objects have both a direct sum or a direct product. However, most of the common categories do fulfill this property. Product and coproduct belong to the more general class of *limit* and *colimit* constructs in category theory (see Figure 2.6). Different types of these diagrams are very useful in application to our proposal as they allow to categorically define all the mapping operations, including decomposition, composition, and abstraction of a problem with the application of a particular model.

2.3.1.2 Category of Sets as an Example

In order to define a category one should specify its objects and morphisms. If we consider objects to be ordinary sets and the morphisms to be their mappings, we would obtain the *category of sets* provided that the associative composition and the identity morphism conditions are satisfied. We again consider the example of three objects, now sets, A, B and C. We define composition of morphisms as the composition of mappings, i.e., the rules that assign elements between sets, e.g., from set A to set B. In this case the mapping from A to B that is α composed with the mapping from B to C that is β, i.e., $\beta \circ \alpha$ in the category theory terms, is equivalent to the mapping of any element a from the set A to the element of C that is $\beta(\alpha(a))$. The mapping of compositions is associative for the sets, i.e., the first condition to be a valid category is fulfilled. The identify condition is satisfied as well, as the identity mapping i_A is the mapping of any element of the arbitrary set A onto itself.

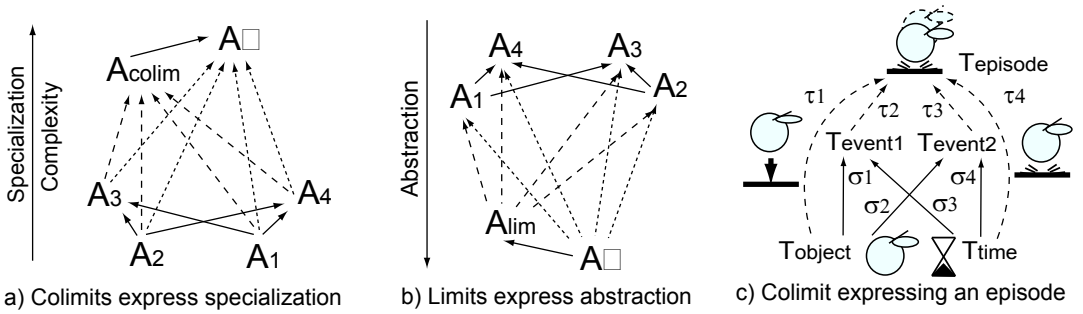


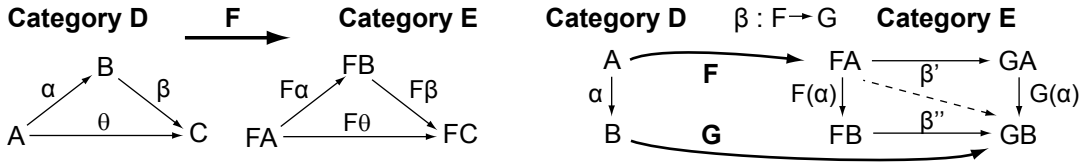
Figure 2.6: Sample diagrams of limits and colimits. Figures (a,b) illustrate one of the basic limit and colimit diagrams showing only the commuting objects, but not morphisms. Figure (c) shows a categorical definition of the concept of an episode as a colimit of four objects that have structured relations. The two basic objects/concepts ("object" and "time") map to the event objects, with morphisms from all of them required to formulate a notion of an episode (adapted from [134]).

In the category of sets the notion of the direct product is equivalent to the Cartesian product. The Cartesian product allows to build new sets from the existing ones by defining members of the new set as a combinations of elements of old sets, with only a single element taken from each and every original set. The example of a cartesian product is the two-dimensional coordinate plane that is constituted from the set of x-axis points and the set of y-axis points. The two sets A and B have a cartesian product $A \times B$ that consists of all pairs of their elements (a,b), when there is the mapping α from the cartesian product $A \times B$ to the set A, as well as the mapping β from $A \times B$ to B $(A \times B, \alpha, \beta)$, see Figure 2.5d. Similarly, the direct sum in the category of sets is equivalent to the disjoint union, i.e., $(C \cup_d D, \theta, \delta)$, where C and D are the original sets, $C \cup_d D$ is the disjoint union, and θ, δ are the respective mappings. The mapping θ is from the set C to $C \cup_d D$, where the element c is mapped to the element $(c, \mathbf{1})$. The mapping δ is done similarly to θ. The category of sets, while somewhat simple, serves as typical example of a category. Often objects in a category theory can be though of as sets with some additional structure on top, and the morphisms can be considered as structure preserving mappings.

2.3.1.3 Operations on Categories and Construction of New Categories

Considering the meta-optimization aspect, not only definitions of categories and operation on its objects are important, but also the operations on categories, and construction of new categories from the existing ones. The central concepts for operation on categories are *functors* (see Figure 2.7a). These are the mappings from one category to another that preserves the categorical *structure*, i.e., the objects and the morphisms of one category are respectively mapped to the objects and the morphisms of the another. The common notations of the functor $\mathbf{F}$ applied on the category $\mathbf{A}$ to the category $\mathbf{B}$ is $\mathbf{F} : \mathbf{A} \to \mathbf{B}$ or $\mathbf{A} \to \mathbf{FB}$. This essentially means that, first, each object in category $\mathbf{A}$ gets mapped to the category $\mathbf{B}$ as $F(A)$. Second, each morphism in $\mathbf{A}$ $A \xrightarrow{\alpha} A'$ is mapped to a morphism in $\mathbf{B}$, $F(A) \xrightarrow{F(\alpha)} F(A')$, so that the valid definitions of the composition $A \xrightarrow{\alpha} A' \xrightarrow{\beta} A''$ and the identities are preserved, i.e.,

$$\begin{aligned} &\mathbf{F}(\beta \circ \alpha) = \mathbf{F}(\beta) \circ \mathbf{F}(\alpha) && \text{for morphisms } \alpha \text{ and } \beta \text{ in } \mathbf{A} \\ &\mathbf{F}(i_A) = i_{\mathbf{F}(A)} && \text{for each object A in } \mathbf{A} \end{aligned} \tag{2.4}$$



a) Functor, structure preseving mapping b) Natural transformation, putting two functors together

Figure 2.7: Sample diagrams illustrating the notion of a functor and natural transformations (adapted from [135]).

The functors suiting the above definition are called *covariant* functors. *Contravariant* functors are in a sense opposite to the covariant functors. They have the same identity definition, but a different composition rule for $A \xrightarrow{\alpha} A' \xrightarrow{\beta} A''$, that is

$$\mathbf{F}(\alpha \circ \beta) = \mathbf{F}(\alpha) \circ \mathbf{F}(\beta). \tag{2.5}$$

For the *contravariant* functor the morphism $A \xrightarrow{\alpha} A'$ gets mapped to the category **B** as $F(A') \xrightarrow{F(\alpha)} F(A)$ instead of $F(A) \xrightarrow{F(\alpha)} F(A')$ as for the covariant functor. Covariant and contravariant functors, as well as the direct product and the direct sum, are key examples of *duality* typical to category theory. In category theory virtually any definition has its "opposite", which allows to inverse any action performed on a category. This applies even if information loss occurs due to the application of *forgetful functors*. (Additionally, bifunctors and multifunctors are also defines in category theory that are functors of two or more arguments, respectively.)

Forgetful functors can "loose" some structure or properties defining a category. For example, a group can be turned into a set if a forgetful functor that drops all the operations performed on the group is applied. Similarly, the mapping of the category of vector spaces into the category of the Abelian groups is performed by a covariant forgetful functor that "forgets" how to multiply vectors by numbers, but does not forget how to add vectors. In Chapter 3 we implicitly apply forgetful functors to map the detailed description of software modules or the deployment scenarios, which can be considered as two categories, into their ontological representations that reside in the other category.

The original categories can be restored by the introduction of the *"free"* constructs that are left adjoints of the forgetful functors. (Adjoints are functors that have an inverse relation to the given functor.) Consider an example from the programming, where an example of the forgetful functor application would be a mapping of an integer to a boolean value. The functor would basically perform an if statement checkup, assessing if the integer variable is not equal to 0 in that case assigning it the "TRUE" value (1 in the internal representation). Otherwise the boolean variable would be assigned "FALSE" (0 in the internal representation). To typecast a boolean variable back to an integer requires definition a free integer variable that has a value of 0, i.e., a mapping back to the category of integers. Then the memory, one byte, where the boolean variable is stored will be copied to the respective lowest bit of the newly created integer variable, which concludes the mapping process.

Besides applying functors explicitly a new category can also be formed as a "subset" of an existing one. A *subcategory* **A'** is constructed from the selected morphisms and objects of the parent category **A**. The *full* subcategory preserves all the morphisms of the

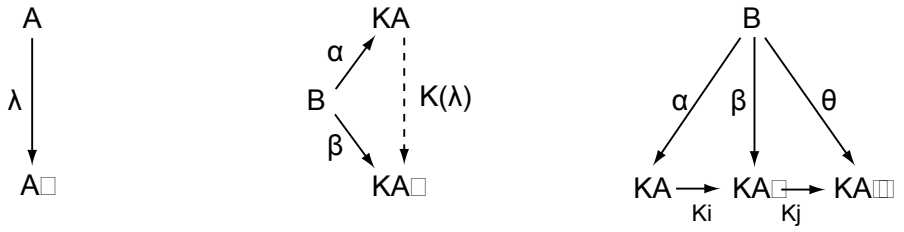


Figure 2.8: Commutative diagrams from approximation of an NP-hard optimization task by application of different models (adapted from [126]).

chosen objects from the parent category.

Category theory can be considered to have layered hierarchical organization. A category with its object and morphism classes can be seen as a basic level of the hierarchy. Higher level categories include other categories and the respectively involved functors as their objects and morphisms, respectively. *Natural transformations* can be considered as morphisms in a category of functors (see Figure 2.7b). They allow operations on functors and between functors. The same basic categorical operations, such as products and coproducts, apply for both the basic and the higher order categorical constructs.

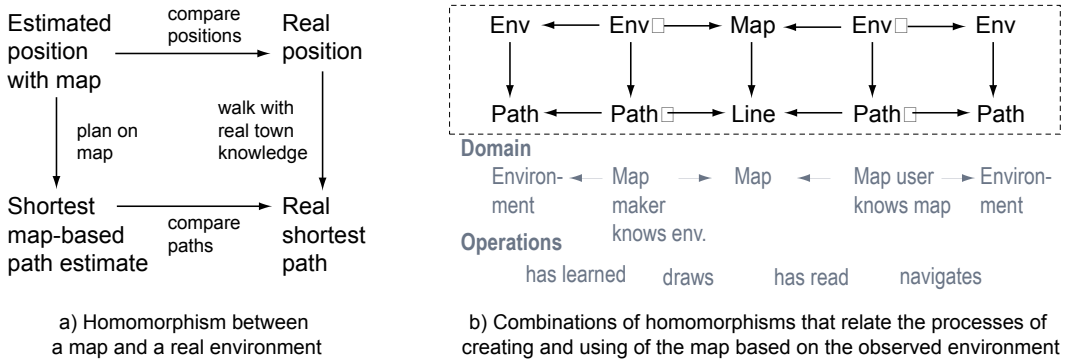
2.3.2 Reviewing and Applying Relevant Categorical Constructs

Category theory relies on small number of basic concepts with few required properties. It is elegant, dual, scalable as the theory has a clear hierarchical organization. Based on particular specifications, often expressed in the form of commutative diagrams, a system of particular categories and the respective functors can be defined. Coherent grammars that can be formulated using category theory have become popular in non-mathematical fields as an attempt for finding a unified upper layer of abstraction of the diverse concepts, methodologies, and techniques used in those areas. These grammars can formalize and define the correct ways of translation between different (often alternative) constructs without losing their structural organization. In application to relational databases this, for example, allows for changes in specification a database schema that automatically leads to a coherent reorganization of the respective data stored in the database [129]. There also exist proposals on the categorical specification of functional and object-oriented programming concepts [136, 137], component-based design [138, 139], and model-based engineering [123]. Most of these works are motivated by the similar scenarios for modification of the structure of a system, translation between different layers of abstractions (e.g., an architectural specification and a concrete realization), migration between the applied models. Typically, the authors, first, define categories and functors, and prove their basic properties. Then they construct the appropriate diagrams, often employing the specific types of limits and colimits, and prove that those commute for the defined functors, categories, their objects and morphisms. Often the suiting derivatives of the basic category theory, such the shape or the comma category theory, are applied, as those already define many useful properties and diagrams that can be applied for the particular problem.

The commutative diagrams based on specific limits and colimits are also applied for formalization of processes in other areas of research, such as ontology engineer-

ing [130, 135], multi-agent modeling [140], or information fusion [133]. Figures 2.8–2.10 display some of the commutative diagrams suggested as part of the categorical formalization for different fields. Figure 2.8 shows the basic of model-based approximation of an NP-hard optimization task [126]. In their work [126] authors suggest that the categorical formalisms can be used to define a hierarchy of approximations to the original optimization tasks based on applied models. Such a hierarchy allows to evaluate alternative approximations/models so that the most appropriate one can be chosen considering the computational constraints and the required accuracy of the solution. It also clearly signals for applicability of different models for particular optimization tasks. In the diagram 2.8b **B** is the category of objects of interest, **A** is the category of archetypes or models used for approximation, and **K** is a category of functors for comparison of the original objects with the modeled/approximated objects. Therefore, an approximation of a particular object B in **B** is the pair (α, A), where A is a model in **A** and α is a morphism $\alpha : B \to KA$. The approximations (α, A) and (β, A) can be compared though the morphism λ, so that $K(\lambda) \circ \alpha = \beta$ and the respective diagram on the top part of the figure commutes. Figure 2.8c shows a more complex development of the same idea, where the object B in **B** is approximated though a number of morphisms α, θ, β and models A, A', A'' in A. The objects KA to KA', KA' to KA'', and KA to KA'' are compared by functors Ki, Kj, and $Kj \circ Ki$ respectively. The authors of this work rely heavily on categorical shape theory [141]. Its basic idea is the provision of the categorical constructs that formalize and relate the parts of the process of studying an original collection of objects in category using their approximations. These are obtained through application of the set of modeling/approximation methods. From our point of view the approach in [126] is very useful to our model-based meta-optimization task described in the beginning of the chapter. It suggests a categorical construct for the abstraction function, where the one network optimization task is approximated by the other as a result of application of some model. For example, using such a categorical construct we can substitute a throughput maximization task in a wireless CSMA/CA network that directly employs throughput feedback from spatially distributed nodes with one using graph-based network motifs, which reduces the amount of feedback required to make optimization decisions (see Chapter 5).

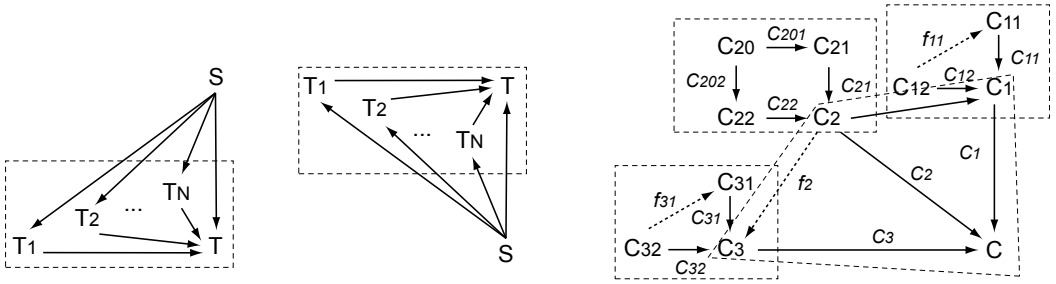
There also exist a recent suggestion on applying the categorical constructs for model-driven engineering [123]. The authors show how the selected categorical patterns, such as monads, are directly applicable for modeling of everyday routines encountered in model-driven design, e.g., model comparison, transformation and synchronization, up to the very concrete example of the update operation. The authors generalize their approach by introducing the category theory based concept of meta-modeling, which defines the theory of description and operation on models that are, in turn, applied as design patterns for software development. This work is one of the many that uses specific forms of limits for merging of structured objects. One other example application of the category theory along with multi-agent modeling defines a formalism for spacial communication with maps [142]. The authors showed that if a combination of homomorphisms, i.e., structure preserving morphisms, holds then there is an equivalence, with respect to reaching of the desired goal, between a navigation done by the agent using a map, and a path taken by the agent with the perfect knowledge of the environment (see Figure 2.9a). The authors also showcased that with a complex homomorphism one can define the process of map construction and its further exploitation so that the desired goal would be reached (see Figure 2.9b). These examples can be clearly related to the comparison of the solutions provided by the original optimization problem and its approximations. If the quality of



a) Homomorphism between a map and a real environment

b) Combinations of homomorphisms that relate the processes of creating and using of the map based on the observed environment

Figure 2.9: Commutative diagrams with homomorphisms that correspond to the situation when both the map-induced navigation and navigation with perfect prior knowledge of the real environment lead to the same goal. Also a combination of homomorphisms that include the step of creation of the map are shown (adapted from [142]).



a) Decomposition

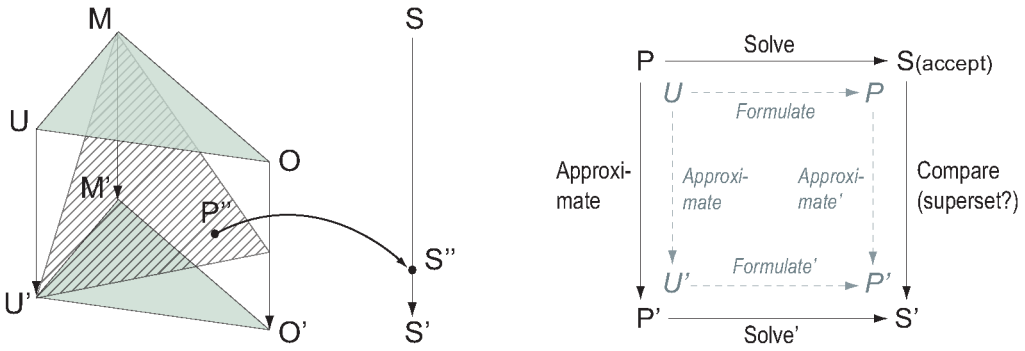
b) Composition

c) Layers of component models

Figure 2.10: Commutative diagrams (a limit and a colimit) corresponding to the decomposition and composition operations. Also the commutative diagram of the complex multi-layer component structure is displayed with both structural (c) and functional (f) morphism between subcomponents (adapted from [139]).

the solution provided by an approximation satisfies the requirements to a solution of the original problem then the task formulations can be declared equivalent. For example, if required minimal per-user throughput can be achieved based on the solution of the SINR maximization task conducted in a simulator then the online search for the optimal configuration in a real network does not have to be performed even if, in principle, it can provide better results at the price of the increasing network complexity. Figure 2.9b can be also related to the problem of model formation and exploitation. The corresponding diagram commutes only if the trained model actually corresponds to the environment in terms of structure, as well as contributes to the solution of the problem under study.

Besides abstraction, other operations as part of the meta-optimization that are the zooming as the inverse of the abstraction, decomposition and its inverse composition, and transformation are also defined using limits and colimits. The examples of relevant operations in relation to the architecture-centric modeling [139] are shown in Figure 2.10. The displayed diagrams correspond to the general cases of composition and decomposition, as well as multi-layered component-based design with both structural and functional relations between the components. Depending on the types of morphisms or functors used,



(a) An optimization task and its approximation as a product space of utilities, optimizers, system models and the approximation of those. Any point in this product space maps to a particular optimization problem formulation, which is then mapped (solved) to a particular solution.

(b) A commutative diagram of an approximation of an optimization task that leads to an approximate, but still acceptable solution. The same categorical diagram, being re-labeled, can apply to multiple networking problems, e.g., convex relaxation of SINR maximization problem.

Figure 2.11: Commutative diagrams of approximations in application to an optimization task and a meta-optimization problem.

different mapping between the optimization task formulations are possible, as discussed, e.g., in [143]. For instance, application of forgetful functors immediately leads to the abstraction operation which cannot be inverted, i.e., lead to zooming only with addition of a free construct. If categories are isomorphic, i.e., their functors are fully invertible we are dealing with the transformation operation.

2.3.3 Categorical Structure of Meta-optimization

In this section we give a categorical definition of a network optimization task, and the process of meta-optimization. We provide several basic commutative diagrams showing main stages of solution of an optimization task, and show how exactly the same diagrams, just with different morphisms (functors) apply to diverse network problems.

A fundamental component in all of the considered examples of networking problems is the category of optimization problems. In the most basic form this arises as a product category of three individual categories. These are the system model (typically subsets of Euclidean spaces, products of discrete sets, manifolds, or similar), the utility functional, including the constraints and the cost factors (with the utility and the cost residing in the category of maps from state space to real numbers, and constraints being Boolean functions on the state space), and the specified optimizer (typically iterated functional program on the above). Several variations of these concepts are, of course, possible. For example, we can incorporate exploration history into the optimization problem by considering not only the present state of the system model but an entire path (making the utility a functional instead of a function), or consider the state space an estimated part of the problem instead of being fixed.

Figure 2.11a illustrates the optimization task as an element of the product category that is mapped to the particular solution, which is in turn is the approximation of the ideally obtainable result. The diagram underlines the major trade-offs between the utility functionals, the choice of optimizers, the system models, and their approximations, by

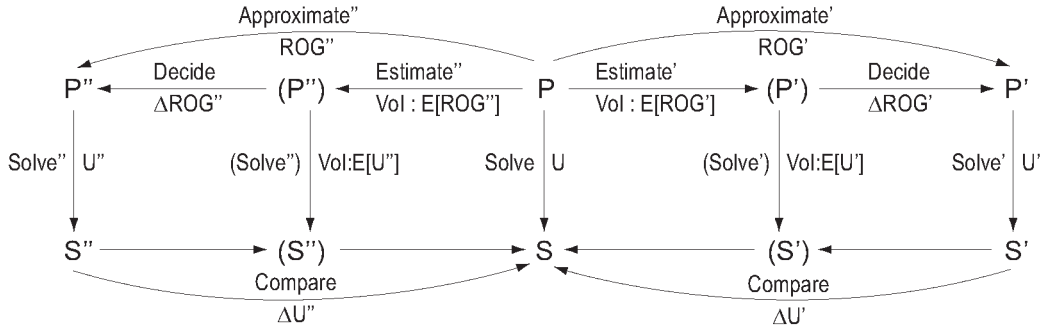


Figure 2.12: A commuting diagram, showing two alternative approximations of an optimization task including associated ROG, and utility metrics, as well as their VoI related estimates that effect the mapping decisions.

mapping the various formulations of the problem in thus arising three-dimensional space to the line of solutions ranging from the idealistic one to the most approximate. It is important to note that any approximation, including the non-optimal solution process (e.g., time-bounded), potentially leads to an optimality gap in terms of the utility and the cost. Additionally the optimality gap carries the risk of being re-evaluated once considered as part of the solution to the original problem. This cumulative metric is referred to as ROG further. Estimates of these values have a direct connection to the VoI considerations.

We shall now move from the plain optimization tasks to the overall meta-optimization problems. Figure 2.11b illustrates the latter, with approximations of different kinds represented as morphisms in the category. This is a very general diagram that arises from many key wireless and fixed networking problems problems. For example, in several cases one can try to obtain similar results by either approximating the optimization problem (say, using convex relaxation of a non-convex optimization problem, such as SINR maximization), or by changing the underlying utility function of the user. If this is possible to within an approximation factor encoded into the definition of the morphisms, this diagram would exist as a commutative diagram in the category. Otherwise the given relaxation does not have simple interpretation as a changed user utility. Having a clear utility-based description of the approximation involved would be very valuable, since that enables the network operator or vendor immediately assess the perceived impact of the approximation on the satisfaction of the users. However, the possibility of such an approximation is not clear yet and remain an active research question.

Figure 2.12 shows categorical construction for the evaluation of an approximation of a problem in two different ways, and the comparison between the related VoI, utility, and ROG estimates. The diagram commutes if and only if both solutions provided by approximations are acceptable. Additionally, the estimates of ROG and utility are to be accurate enough for the decision on the viability of the problem approximation to be taken, otherwise the left- and the right-most parts of the diagrams would never exist. The wrong VoI estimates clearly lead to inaccurate meta-optimization decomposition decisions and, therefore, most probably, suboptimal solution of the optimization task both in terms of achieved utility and incurred costs. Generally, two way approximation is common in networking, and is applied for comparison of two models (as we also discuss in the later parts of the thesis). For example, in wireless networking context the choices are often whether to use full spatial model of node locations for solving a given problem, or to approximate

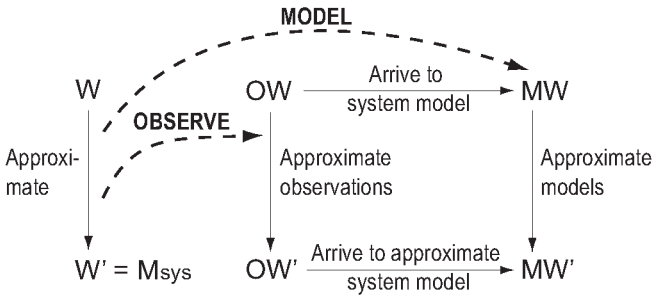


Figure 2.13: Mapping of the real environment and the networked system (the world) into the system model with two functors. These correspond to the observation operations (basically defining the sensory process), and the application of models to represent the reality. If both functors are valid then the diagram commutes, where the order of application of morphisms is irrelevant to the obtained system model (i.e., first the sensory inputs are defined and then a valid model capturing these is found, or first the model is chosen and then the sensory inputs that suit to train its parameters are established).

the network structure with a graph model. In the time domain similar questions can be posed related to the use of full ON/OFF models as alternating renewal processes, or just focus on simpler statistics of such models, for example the duty cycle of the transmitters.

Figure 2.13 illustrates how the construction of the system model of the real world is related through the choice of observation parameters and applied modeling techniques. The original world and the system model are represented as separate categories, with two functors carrying out the mappings shown in the diagram. If this diagram commutes in the relevant functor category, it is possible to construct successive approximations of the underlying system model from observations and vise versa. An example of this will be encountered later in the thesis when we study the construction of time-domain ON/OFF models of transmitter activity patterns (see Chapter 5). There within a fixed modeling framework statistics of the subsampled and noisy observations are used to construct likewise approximate models, but which become successively more accurate as more observations are made. The model approximation side of the figure corresponds then to coarse-graining the ON/OFF model to estimate just the duty cycle of the transmitter instead of more complex spectrum use behaviors.

Figure 2.14 is less abstract as the above ones, and sets an example of a commutative diagram that represents simplified data flow of the classical TCP/IP network stack. Classically data is exchanged only between adjacent layers though carefully defined interfaces with limited amount of variables. Protocols on the same layers are typically aware of only of the state of the same protocol on the other nodes. Such rather restricted scheme is fine tuned to work well in fixed TCP/IP networks. However, the respective categorical diagram, in our case for simplicity constructed only for the transports and the link/MAC layers (see Figure 2.14), indicates that additional information exchange is possible, and even required for the diagram to commute. These additional morphisms are dropped to the fixed networks as the performance enhancements and additional robustness they offer lead to minimal utility gains at the price of significantly higher complexity and, therefore, cost. However, in the case of wireless networks such an exchange enabled by the cross-layer design schemes and additional middleware, might become justifiable. (The particular schemes are still a subject of active research with respect to different network

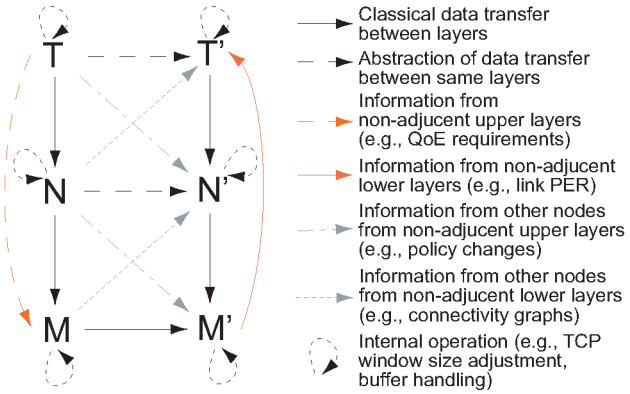


Figure 2.14: A categorical diagram corresponding to a simplified representation of information exchange between transport (T), network (N) and link/MAC (M) layers.

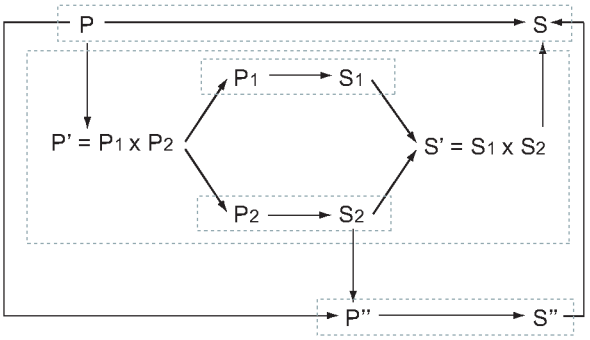


Figure 2.15: An example of composite decomposition and approximation of an optimization task, where the communicating diagram would indicate equivalence of different approximate solutions to the acceptable one obtained as a solution of an original problem.

types and deployment scenarios. One classical example, which demonstrates that indeed additional information exchange needs to be considered is the TCP over wireless problem. Here the TCP algorithm reduces its window size in reaction to packet losses over wireless link, as it assumes congestion on the network layer. The problem is mitigated, if the protocol's system model and its inputs are altered, e.g., the nature of the underlying link or, better, the cause of the delay/packet loss is indicated [144, 145]. From the diagram we can see two architectural ways for sharing information between layers. First of these consist of adding additional state variables to each of the layers (composition), whereas the second alternative consists of definition of explicit interfaces for cross-layer information sharing. Both of these have, of course, been proposed in the literature on an *ad hoc* basis.) Stating more formally, if the original decomposition into MAC, network and transport layers no longer results in a product in the optimization problem category, the additional information represented through the extra morphisms is needed to bridge the optimality gap, which becomes significant. In such context the "degree of forgetfulness" (the VoI criteria) for the relevant functors carrying out the original decomposition becomes the key metric to consider and study.

Finally, Figure 2.15 shows a general version of the decomposition that includes the perfect decomposition and the subsequent composition of the solutions to the subprob-

lems to form a solution to the original task. These arise as the categorical definitions of products and direct sums. Key example for decomposition in the context of the classical TCP/IP stack is the division of the network resource management into routing and end-to-end rate control. Under suitable assumptions on flow durations and the utilities involved, this forms a decomposition of the more general resource assignment into two distinct parts, each with their own solutions, which can be composed on the system level to yield a solution to the original problem. More similar examples can be found from [15]. Additionally, the diagram shows an alternative possibility of using the partial solution S_1 to reformulate the original optimization problem (P into P'') and subsequently solve it. This diagram would commute only if the resulting solutions S'' and S' (from the perfect decomposition) are in the set of acceptable solutions S.

2.4 Conclusion

In this chapter we considered the optimization-centric view to networked system design and run-time management. We suggested, also based of extensive related work, that many of real network optimization problem would benefit from the meta-optimization approach, which includes the problem decomposition, abstraction, and relation between models that is central to our methodology. We then provided a concise overview of the different aspects of system design arising from this viewpoint, including the various desirable features any meta-optimization based approach should have. We defined and discussed in detail the key constituents of optimization tasks, including utilities of the different stakeholders, policies as constraints, system models, optimizers, and various relations between these especially with respect to obtaining approximate solutions.

In the second part of this chapter we proposed the use of category theory as a highly promising approach to formalizing the meta-optimization task. The key concepts from category theory needed for formalizing network optimization tasks are the graphs associated with the objects and morphisms in the optimization problem categories, with their limits, colimits, initial and terminal objects. Each morphism typically has a concrete meaning that can be mapped into a real world concepts, such as information exchange between layers. If a particular categorical diagram does not commute, this usually becomes an indication to study the related decomposition/composition problem further. Morphisms can be thrown away applying, e.g., probabilistic reasoning or otherwise knowing that arising of the particular functionality/problem is very unlikely (which might create problems with further evolution of the system, as with TCP over wireless). Same diagrams apply to many problems of different nature, for instance, interface definition vs. model application, though both can be mapped to the information exchange/transformation as a general umbrella. The categorical framework is natural for alternative representation of the constituents of the optimization task and meta-optimization process.

The key topic for future work will be to further elaborate and detail the category-theoretic approach, especially from the automated reasoning point of view. In the rest of this thesis we contribute to the different major areas of the meta-optimization task, including study of different optimization techniques, use of ontologies as precursor to category-theoretic reasoning in networks, as well as forming and approximation of system models for wireless communication systems. These different lines of work are brought in part together in the end when we describe the design and implementation of an extensive home networking self-optimization prototype using some of the key parts of the optimization-centric design approach.

3

Ontological Reasoning for Wireless Sensor Network Stacks

In this chapter we apply elements of the proposed optimization-driven methodology to enable semi-autonomous design of protocol stacks for Wireless Sensor Network (WSN) scenarios. Wireless sensor networks are characterized by a large number of non-standardized protocols, varying application requirements and stringent resource constraints. Therefore, these types of networks will particularly benefit from a semi-autonomous framework for designing and managing a network stack targeting at optimal performance. To the best of our knowledge, no such framework currently exists. We suggest and describe the design of an ontology-driven software system, called CONFab, for protocol stack composition and configuration according to user-specified scenarios and requirements [44]. For us an additional reason to experiment with ontologies is their consideration as a promising candidate for practical implementation of a special case of categorical reasoning discussed in the previous chapter.

We derive, the extensible ontology that sets and relates main concepts required to describe a WSN and, in particular, a protocol stack of its nodes. The same ontology defines the structure of a knowledge base and serves as the backbone for the process of derivation of the optimal protocol stack. We take a service-oriented approach to the stack design, where a particular network is treated as an interconnected collection of configurable components that provide certain services and functionalities. The framework automatically chooses, "wires", and configures components from a pool of alternatives, to form a stack that maximally fulfills user requirements. It learns using the feedback from deployed systems, as well as expert and user inputs to predict performance of a network. For decision making the knowledge base is primarily based on Description Logic (DL). The additional reasoning is also enhanced with a number of plug-ins that enable, among others, utility-based optimization, metaheuristic search and autonomous component-based protocol stack combination. After the network deployment its performance can be further fine-tuned using a run-time adaptation components, which configuration can be also devised based on data gathered as part of the CONFab operation.

We use a set of commonly known MAC and routing protocols to validate the framework on the Indriya testbed in different user-specified application and deployment conditions [146]. The results indicate that CONFab with its component-based approach is able to construct optimized protocol stacks and thereby yields increased quality of service and network lifetimes. As the core of the suggested system is an ontology-driven knowledge base, in Appendix B we further explain ontology-enabled reasoning. As an example we use a problem of composition of a functional set of services for a given home environment, concentrating on services accommodated by white home appliances.

3.1 Introduction

The task of automation of network design and management, the vision of cognitive wireless networking [147] or autonomous computing [148] cannot be realized without setting a shareable common "vocabulary" for the involved systems. Ontology [149] is a natural candidate for this purpose. Ontologies are used in several domains of computer engineering, including the Semantic Web, to specify a conceptualization in the context of knowledge sharing, thus enabling semantic search and information retrieval [150–152]. There also exist recent promising efforts on standardization of an ontology conceptualizing low-layer aspects of cognitive radios [153]. By conceptualization we understand an abstract, simplified view of the environment we want to represent, i.e., a model. Various qualitative logical operations can be carried out on a item of ontology. A correctly formulated ontology allows for checking and guaranteeing consistency, but not completeness of the stored information with respect to queries and assertions formulated using the vocabulary defined in the ontology. An ontology along with a database and a reasoning engine constitutes a Knowledge Base (KB) that employs description logic [154] to make deductions on consistency of knowledge and filling in the missing gaps in logical constructs.

In the context of networked system design and optimization an ontology can serve as a basis for *qualitative* modeling of the system and its environment. In this chapter we employ an ontology to represent knowledge on network stack organization [14, 15, 155, 156] that we consider from component/service-oriented point of view [64, 157]. The proposed ontology also captures in a structured manner a pool of available components and scenario descriptions that include user goal statements. Based on this ontology we create a knowledge base that serves as a backbone for the tool called CONFab (Component based Optimization for Networks with deployment Feedback). This extensible software system is capable of semi-autonomous pre-deployment composition and configuration of a protocol stack for a particular deployment and application scenario according to quantifiable performance metrics. The goal of the system is to choose from the pool of existing software components the best ones and form a valid scenario-specific protocol stack so that the user requirements are maximally satisfied. The developed tool follows the optimization methodology described in Chapter 2, with the ontology also defining the backbone of the meta-optimization. The decisions on protocol stack composition and configuration depend on user scenario specifications, expert inputs, and processed feedbacks from existing network deployments. The overall system architecture is shown in Figure 3.1.

In order to map the performance data to the KB, enable finer grained solutions and advanced reasoning, we introduce a number of plug-ins that, for example, facilitate quantitative analysis on data, utility-based decision-making, and comparison of component graphs that map to protocol stacks. For the same reasons we perform partial inference on the KB through dedicated queries. Such a modular approach maps to the concept of meta-optimization and allows capturing and operating on descriptions at various levels of granularity, provides mapping and defines relations between different categories and sets of objects. This allows, for instance, non-experts to define high-level goals that are then mapped to protocol performance metrics by the framework. Similarly, mechanisms accommodated through such an architecture enable stack performance predictions for scenarios on which no prior experience is available, also through active protocol stack optimization state space exploration via scheduling and deployment of new experiments [43, 46, 51]. CONFab learns from past experiences and also utilizes expert inputs for decision making. It operates on already existing protocol stacks, provides recom-

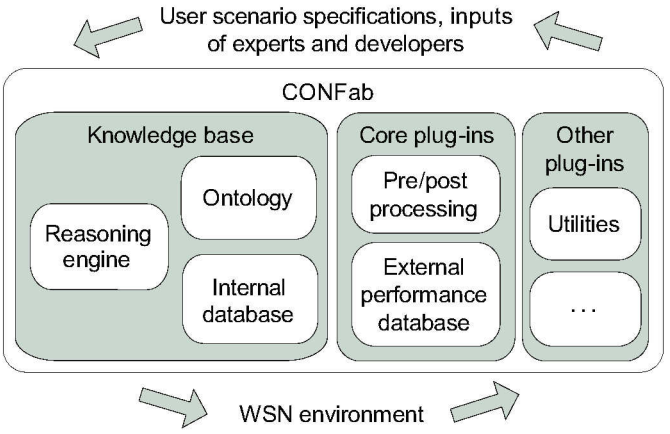


Figure 3.1: The overall architecture of the CONFab framework, showing inputs to the system, its main blocks, and sample plug-ins [44].

mendations on possible wirings of components and their configurations. It can also find commonalities between different stacks to facilitate the creation of new ones. CONFab has a modular design and can be enriched with additional functionalities realized either as plug-ins to the system, or through additions to the structure of the knowledge base via extension of its ontology.

We validate CONFab on component-based and monolithic implementations of well-known MAC and routing protocols on the Indriya testbed [146], with some plug-ins being evaluated separately. We show that CONFab with its component based approach is able to predict, propose, and construct efficient protocol stacks, improving performance and network lifetime. Therefore, it eases and partially automates a suitable network stack development process. For example, by employing component-based design and CONFab processing we derived a new protocol stack that provides up to 37% improvement in energy efficiency in medium sized networks (30–70 nodes) as compared to the best monolithic stacks without compromising packet delivery rates.

The rest of the chapter is structured as follows. In this section we review main concepts and related work for wireless sensors networks and ontological reasoning. We also describe the tools used for CONFab implementation. In Section 3.2 we formalize service- and component-based design principles to WSN protocol stacks and elaborate on the choice of component granularity for our task. We then describe the structure of the ontology as part of the discussion on CONFab design in Section 3.3. We also describe some of the selected plug-ins. The validation experiments for the framework are described in Section 3.4. Finally the chapter is concluded in Section 3.5.

3.1.1 On Wireless Sensor Networks

A wireless sensor network is an ad-hoc, typically multihop, networked system that primarily gathers information on phenomena of interest in a particular deployment environment. WSNs are utilized in applications serving, for example, military, medical, environmental, home, security, and emergency management purposes [158]. These networks are, therefore, required to operate efficiently in a wide range of deployment conditions, under diverse application loads, and meet highly varying QoS/QoE requirements. The resource-

constrained nature of WSNs also often dictates an additional optimization goal that is the longer network lifetime, i.e., minimization of power consumption. These networks fall in the domain of rapid prototyping [50]. There exist multiple alternative protocols without any standardized protocol stacks, like TCP/IP. All these factors indicate that the likelihood of having uniform protocol stack with default parameter settings for these networks is low. Thus, like in the case of embedded systems in general, we require deployment- and scenario-specific solutions [159]. At least parameter-based protocol stack adaptation is to be performed. It is likely that component-based optimization will also be required to construct a dedicated protocol stack either from existing protocols or smaller components. *Protocol stack composition* therefore emerges as one of the *core design tasks* for wireless sensor networks. Even partial *automation* of this process reduces economical and personnel expenses needed for this task. Thus, WSNs can be considered as a perfect "playground" for autonomous service/component-oriented network design and management research[1]. If these techniques are successful for WSNs they can be applied to other suitable network types such as cognitive wireless networks.

For CONFab evaluation, we choose to experiment with TelosB motes [162] that run TinyOS operating system [163]. This operating system follows the component-oriented philosophy, thus being a natural tool for our research. We view a protocol stack as a collection interconnected *components* of various granularity, ranging from simple modules, such as timers, to complete protocols or even whole stacks. These components carry general and scenario-specific characteristics. They can be interconnected to provide desired services. The design space for WSN protocol stacks can be viewed as a multitude of alternative components, their configurations, and interconnections (or "wirings"). Thus, we may have to operate on a large and complex optimization state space with vast numbers of design possibilities.

The natural drawback of components-based design in its immature stage is the lack of interface standardization, which leads to emergence of many of alternative suggestions that are poorly compatible with each other. In this work we do not address this problem. We implement our components or create wrappers for the protocols based on our prior experience [49, 50, 164] and other related research [165–168]. Our point is not to suggest new interfaces or middleware, but to create a framework that matches meta-data of those as the part of protocol stack composition process. Further, we shortly review related work in WSN community.

A large number of independent protocols have been proposed for WSNs, including MAC [169], routing [170], and other layers [164, 171, 172]. While there exist multiple performance evaluations of individual protocols, understanding on the overall behavior of the whole protocol stacks under diverse application loads and environmental conditions is lacking. There are, of course, proposals for configurable solutions imposing a fixed system architecture, like TinyDB [173], or requiring common middleware or an isolation/harmonization layers [174, 175], or support for specific meta-language and code mobility [176]. Research on the cross-layer design also targets at the same objective [109]. However, there again exists no framework that compares these approaches, and navigates a user to the one that is most appropriate for the given deployment scenario. If we

[1] We use the terms "service" and "component" next to each other. In our understanding, as explained later, services are high-level abstraction of component interfaces. For one, service can be realized by several alternative components which, in turn, map to concrete realizations in software modules. For example, routing services are mapped to such WSN components as S4 [160] and CTP [161] protocols.

take a heavy assumption that there are common meta-data characteristics imposed on existing WSN components, protocols, overall stacks and frameworks, it will be possible to deduce only a small subset of suitable solutions that are viable for a particular deployment. This will allow increasing the reuse factor of existing components and decrease economic overhead of WSN design. The more specific and structured the component meta-data is, the more feedback can be gathered on the deployment scenarios. The more common interfaces are shared between WSN software modules, the better solutions such a framework can provide.

There exist proposals that aim at easing the development and run-time configuration of network stacks, limitedly utilizing information on application/user requirements and employing component-based principles [166, 167]. Programming models [177, 178] are another alternative for comfortable WSN development. However, these works often lack testbed implementations. Compared to CONFab they also do not learn from the experiences of actual deployments. Furthermore, besides harmonization of interfaces, these lack a methodology and tools that allow to capture high level network description, and relate protocol stack formations, code synthetics and performance evaluation. Our framework allows all these through integration with our prior works [43, 51].

At the current state we address only the very small, but crucial, part of the stated problem. Using ontology we structure and semi-autonomously reason on component meta-data, as well as gathered network statistics. We only address the task of protocol stack design that is shared by all relevant WSN nodes. Alternative protocol stacks are built either from existing routing and MAC protocols, or their alternative component-based implementations. An additional motivation for our work is the related research, which shows that careful construction of protocols and their interfaces can significantly improve the overall network efficiency [64, 69, 157, 179], also due to extended run-time adaptability. There also exist an open question of efficient protocol stack decomposition [70] and cross-layer design even for classical TCP/IP networks [15, 180]. There are several examples of practical component-based realizations for media access control and routing protocols for wireless sensor networks [165, 168, 181–183]. These approaches though aiming at modular design and run-time adaptivity, still do not address the problem of autonomous (pre-deployment) protocol stack composition.

3.1.2 On Ontological Reasoning

As mentioned in the beginning of the chapter ontologies are widely used in software engineering. Computer scientists use the term *ontology* to specify a conceptualization [149] in the context of knowledge sharing. By conceptualization we understand an abstract, simplified view of the system and its operational environment. Every knowledge base explicitly or implicitly uses some sort of conceptualization. An ontology can be used to set up a taxonomy of a certain domain, describing its main concepts and relations between them. Various logical operations can be carried out on items of an ontology. Ontologies are relatively easy to extend and maintain. Well-known meta-modeling frameworks such as UML and SDL [184, 185] can provide alternatives to description of network services and corresponding modules, but do not naturally support extensions for autonomous reasoning and construction of component stacks based on a complex set of characteristics. The conventional data programming models like tuple spaces [186] and publish/subscribe [187], on the contrary, do not address the main challenge of automatic service oriented network composition. They simply provide models of network organization and can be treated as

separate components that require special wiring models.

Initially we approached the problem of semi-autonomous protocol stack composition directly utilizing feedback from prior deployments stored in a relational database [43]. However, we have quickly discovered that such an approach is not scalable. It leads to complex stored procedures that need to be continuously adapted as new network characteristics are added or a new structure of component wiring is considered. This process was lengthy and error-prone. We migrated to ontology-enabled knowledge base, where these relations and characteristics got encoded using the description logic, and stored in OWL [188]. This made the system much more maintainable and adaptable as we have observed during the further development process that required refinement of the original version of the ontology. Migration to the ontology-enabled KB has forced us to concentrate on qualitative as opposed to quantitative reasoning. However, as we explain further, application of hierarchical reasoning mitigates this drawback. On the contrary, such an approach improves maintainability and modularity of the framework.

There are a few proposals for using ontology-based reasoning for WSNs. Most of them, e.g., [189], aim at providing a sharable semantic to integrate WSNs with other networks, their client applications and enable more efficient gathering of sensory information. In particular, [190] is closer to our goals and provides a high-level description of a network and its performance parameters. Unfortunately, this proposal lacks implementation and evaluation results. Furthermore, it does not explicitly foresee the possibility to use feedback information and learn from previous deployment experiences.

3.1.3 Tools

The OWL Web Ontology Language [188] is used to define ontologies. It is an outcome of a collaborative research effort led by W3C. OWL is based on XML. The language, among others, allows to carry out such logical operations as definition of relations between classes (e.g., disjointness or inclusion), statement of equivalence of classes, provision of different characteristics of properties, like symmetry or transience.

We have chosen to use OWL DL, a sublanguage of OWL, that possesses the maximal expressiveness while retaining computational completeness as compared to other ontology description languages, like RDFS [191] or other OWL flavors. The alternative ontology languages, such as CoRaL [192] or RDF [193], are either more limited, difficult to use as part of a knowledge base, have less support and available extensions, or they are very domain specific. Ontologies specified in OWL DL are reusable and interoperable, i.e., a specific ontology context can be reused in multiple domains and understood by different systems, which, in turn, enables automated reasoning. Finally, OWL is one of the most popular and standardized ontology languages, and most likely will become *de facto* standard for the semantic research in the near future. In our work, the OWL-based knowledge base stores objects (individual) that reflect meta-data on components (for protocols and stacks), services that they provide, and scenarios that also carry the user requirements. The support ontology classes contain individuals defining values that characterize (through properties relations) the above entities. As the knowledge base operates on description logic, i.e., it performs qualitative reasoning, these values have to be quantized/discretized in a finite number of categories. Also additional plug-ins have to be invoked to perform this discretization, as well as advanced quantitative reasoning.

Currently we use several tools in our work. Most of them are realized on Java or have Java-compatible interfaces. For realization of the knowledge base we employ Protégé [194],

FaCT++ [195] reasoner and Jena library [196]. Protégé is a visual environment for creating and visualizing ontologies developed at Cambridge University. It features a reasoning API that is used to access an external DIG description logic interface compliant reasoner [197], thus, allowing inference about classes and individuals in the ontology-enabled knowledge base. The inferencing services include, among others, estimation of consistency of ontology classes, and derivations of objects that belong to a particular class. We have chosen FaCT++ [195] for the inference engine as from our experience it provides the best performance among most well known open-source reasoners.

The standard reasoners like FaCT++ or Pellet [198] are very comfortable to use for test developments. However, they do have their disadvantages. First of all they still have a limited functionality, i.e., some rules cannot be evaluated with the existing open-source reasoners. Another problem is that they are not always as fast as needed. Therefore, extra functionality was realized using either Jena library or existing Protégé plug-ins like SWRL [199] which relies on proprietary programming language Jess [200] designed for development of expert systems. In our work we have decided for Jena as it is an open source library, though it requires a lower-level programming compared to Jess. We used this interface to connect to our plug-ins realized in Java, C#, Matlab and R that are further described in Section 3.3. As said, in the wireless sensor network domain we experiment with TelosB motes [162], which use Texas Instrument's CC2420 packet radio, run TinyOS [163] and are programmable using nesC.

3.2 Describing a Protocol Stack

In this section we first state the parameter- and component-based optimization problem for network protocol stack composition. Then we discuss on component granularity and service-based abstractions for effective autonomous reasoning.

3.2.1 Optimization-centric Problem Statement

Adapting the optimization task formulation from the previous chapter to the CONFab task we state the following. We consider *networks* as complex dynamic systems consisting of interdependent software and hardware elements, each possibly effected by the external environments, serving to satisfy a combination of goals directly or indirectly stated by network stakeholders. Network performance is a measure of satisfaction of these goals. A *scenario* comprises all aspects of the operational environment, stakeholder goals, applications behavior, network and system characteristics. Thus, the aim of network development can be formulated as *building of a system that shows the best network performance for a given scenario*.

Development of a networked system, especially a WSN, involves assessment of alternative design and configuration solutions, clearly resulting in an optimization problem. This optimization problem is constrained by the presence of limited resource including technological and economical aspects. For example, there is a limit on the software and hardware components, human efforts and time that can be used for the given task. More formally, we seek a solution

$$S_{sc.} = \underset{s \in S}{\arg\max} N(s \mid O), \tag{3.1}$$

where $S_{sc.}$ is the optimal solution for a networked system for a given scenario, selected from the collection of allowable constraint satisfying systems S, and measured in terms

of network performance N in an environment O.

Our system currently operates only with the software elements and decides on a common protocol stack for all nodes in a WSN. Therefore, in the context of our work a solution s is a network protocol stack, consisting of *components* c and their wiring w. The collection of components c might be fixed, or form as a part of the optimization process considered. Finally, each component might have a collection of *configurable parameter settings* p. Systems can therefore be depicted as

$$s \in \Big\{ \big(\{c_i\}, \{p_j \mid c_i\}, w(\{c_i\}) \big) \mid w \text{ is valid wiring} \Big\}, \tag{3.2}$$

where the validity of the wiring w in terms of the interfaces of the components. We assume that there exists a pool of components, alternatives for their combinations and parameter configurations. There are also possibilities for both alteration of already existing modules or development of new ones.

We decompose the network performance metric into the application performance $P(s)$, and degree of satisfaction of additional constraints $C(s)$, e.g., with respect to consumed hardware resources in our implementation. Currently we assume that the hardware used for WSN network is specified as a part of the scenario description that is not the subject to further development and change during the design process. The performance metrics can be perceived as functions of protocol-level Key Performance Indicators (KPIs), such as goodput, latency, or power consumption. We also split the description of the operational environment in the scenario into environmental and application related parts, denoted by O_E and O_A, respectively. Therefore, the optimization task from (3.1) becomes

$$S_{sc.} \approx \arg\max_{s \in S} N\Big(\big(P(s), C(s) \big) \mid O_E, O_A \Big), \tag{3.3}$$

where the approximation is due to the limitation of the considered goals, as well as the operational and system characteristics as compared to the more general original formulation. The problem (3.3) is the approximation of the original optimization task (3.1).

Clearly the problem (3.3) cannot be solved directly, and we have to operate within the meta-optimization procedure. In CONFab the solution to the optimization task is gained by employing multi-stage and partial reasoning, thus enabling abstraction and decomposition operations (Chapter 2). For example, we separately invoke behavioral and structural reasoning, and execute multiple plug-ins that operate on objects of different complexity and accuracy of representation. The framework also naturally supports scalable computation, where relevant methods can be applied for both batch data processing, as well as on a limited scale for fast result acquisition. Support for reasoning on different component granularity allows for stack construction from monolith solutions, separate protocols, and smaller software modules. The number of tunable parameters may also vary.

3.2.2 Components and Protocols for Network Stack Design

Solution of the optimization problem leads to a particular network stack that provides optimal (in the sense of maximizing N) performance in a given scenario. Through (3.2), this stack can be decomposed into a set of components, their interfaces and the configuration settings that need to be realized. This decomposition task is challenging in practice and only a few works exist that aim at solving this problem, however, while making heavy mathematical modeling assumptions [15].

We suggest merging top-down (decomposition task) and bottom-up approaches to derive components for a network stack design. Using decomposition of a stack, we seek to create generic and re-useable components, conditioned on the available development resources. (Obviously, we also can employ existing components, such as protocols, though their interfaces might need harmonization either through direct editing or additional wrappers.) We then apply forward engineering as part of CONFab for employing these generic components to solve a scenario-specific optimization task. If desired results cannot be achieved through a permissible combination of already existing components, new ones need to be developed and added to the pool of existing components.

We take the classical software engineering approach and decompose a network stack into components derived on *behavioral* and *structural* basis. The goal of the decomposition process is to create highly re-usable components with inter-connectable interfaces. The granularity of these components is selected so as to facilitate the composition of protocol stacks suitable to diverse applications and deployment scenarios. Additionally, it is desirable to strike a balance on component description so that, form one side, minimal expert knowledge into logic of a stack is required from a user. From the other side, the component meta-data description is to be useful for the optimization task formulation.

The protocol stack from the behavioral perspective can be viewed as a flow of *services* hosted by logical units (*functionalities*). We consider the following core high level services: data communication, forwarding, addressing, (sub)network value assessment, media access, radio transreceiving, and adaptability. These services are hosted by functionalities such as stack unit, routing, MAC, decision-making, and radio unit. Components and interfaces are structural reflections of these elements. Our experience indicates that designing services and realizing components at a very coarse granularity restricts their reusability, maintainability and performance due to increased inner complexity. One simple example is a monolithic realization of a protocol, which offers multiple services. Such an approach can result in redundancy, duplication, of provided services, as well as limit their sharing, i.e., decrease reusability. For example, both MAC and routing protocols might benefit from link quality estimates. For strictly layered protocol stack this functionality needs to be either implemented on both layers or an additional cross-layer module is needed. Additionally, current protocol implementations often do not use unified interfaces, which can result in loss of crucial information between layers [201].

Obviously the absence of unified interfaces also limits the opportunities for autonomous stack composition. Generally the larger is the component size the more complex and, therefore, less reliable it is. However, it can host more services and implement them more effectively. Additional adaptivity of large component can be achieved either through exposing a large number of configurable parameters or using complex adaptive algorithms. Both of the options require very careful initial design, increase component complexity, decrease its maintainability and ease-of-use. On the contrary, very fine grained stack decomposition may lead to complex component interdependencies that results in high wiring and reasoning overheads.

3.2.3 Protocols and Attributes Chosen for Design Validation

In order to validate our approach, we have selected well-known and widely used WSN MAC and routing protocols. The three chosen MAC protocols, BoXMAC-2 [202], X-MAC [203] and TrawMAC [204]), are based on the preamble sampling principle. The routing protocols, CTP [161] and S4 [160], use the ETX (Expected Transmission Index) metric for

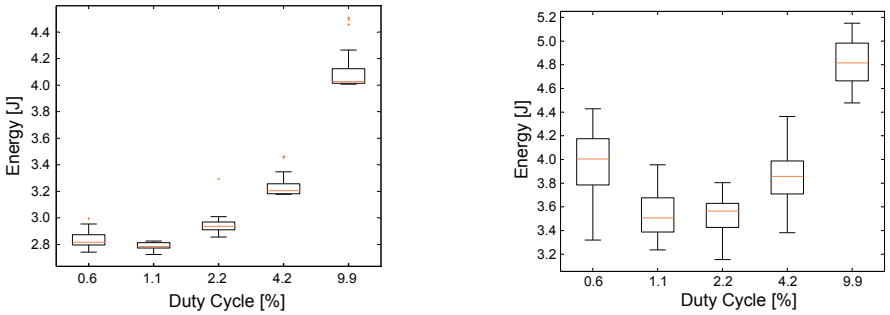


Figure 3.2: Sample total energy consumed by each node in a network during the 5 minute experiment. The panel on the left shows the result for the 20-node network with low packet application rate (0.01 pkt/s). The panel on the right shows the results for the 40-node network with high packet application rate (0.2 pkt/s). Packets are generated in a bursty manner by all nodes in a network besides the gateway. Both experiments execute monolithic implementations of CTP routing protocol and BoXMAC-2 MAC protocol.

link estimation. Though the behavior of the selected protocols vary significantly depending upon the network and traffic conditions, most of them share common functionalities. We use two realizations of these protocols as inputs for CONFab reasoning. One is based on their monolithic implementation based on openly available code, another is the component-based realization that we describe further. Our components have certain configuration parameters, which include duty cycle (or sleep interval), beaconing rate, and routing decision metric. Platform specific features can also be supported such as hardware ACK or address recognition on the CC2420 radio chip. In this work we actively experiment with the duty cycle parameter, as it is known to have a crucial effect on one of the most important WSN KPIs that is the lifetime of a network. Besides, this parameter is known to have non-linear behavior [205], see Figure 3.2, which makes it even more interesting for experimentation[2].

As part of the scenario description a user has to specify the optimization goals, capabilities of a system, and scenario characteristics. Currently we employ a limited set of scenario specification parameters. We mainly specify the expected deployment size of a network, application datarate and its behavior (e.g., bursty or constant application packet rate, single- or multiple-destination traffic, etc.), as well as the type of power supply to derive expected network lifetime. There are also other scenario parameters, such as expected mobility rate and hardware platforms, but for those we use default values and they were reasoned upon using only artificial examples, while testing the KB and its ontology.

There exist multiple KPIs that influence network performance as perceived by a user. For wireless sensor networks the key ones include the lifetime of a network, application throughput and delivery ratio, and resilience to node failures. Other metrics supported by CONFab include RAM/ROM usage, which is we employ as a constraint on the available TelosB node memory. However, the results presented in Section 3.4 focus on Packet Receive Ratio (PRR) and power consumption metrics, which are one of the most important KPIs influencing the network performance and lifetime.

[2]The boxplot representation summarizes numerical data through few statistics. The horizontal lines of the box represent the median, the first and the third quartile of the data. The whiskers define the data points at the 1.5 times of the interquartile range from the box edges. The boxplot may also show outliers of the portrayed numerical data.

3.2.4 Protocol Decomposition

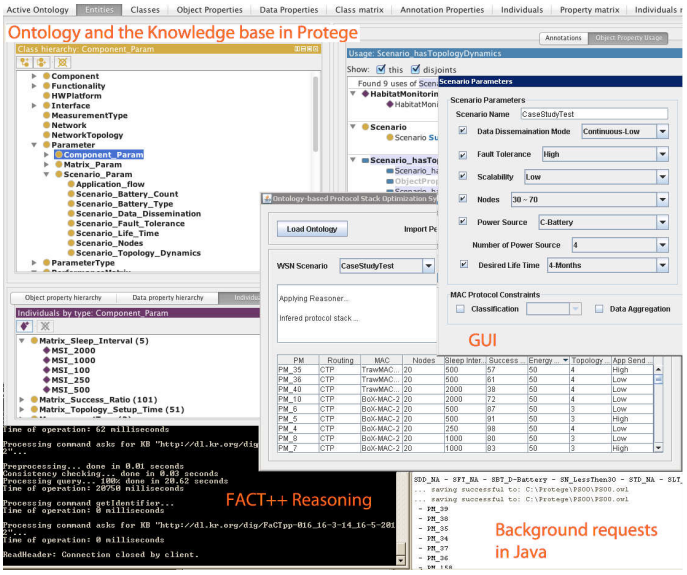
Next, we describe our take on protocol decomposition that is based on our and our colleagues' earlier research [50, 165]. This is not the only decomposition approach that can be accommodated by CONFab. With appropriate ontology adjustments (in terms of valid interrelations between different components and services, and additional meta-data), many of the related works reviewed above can also be incorporated while keeping the core ontology and CONFab structure the same.

We found it useful to reason not only on the behavioral and structural abstractions of a protocol stack, but also consider two levels of component granularity. The primitive level components in the case of MAC protocols typically include the low-level hardware dependent functionalities that provide platform independent interfaces such as timers, random number generators, carrier sensing, send-/receive frames, etc. Primitive components enhance protocol development due to their reusability and shared interfaces [165]. However, due to low granularity, it is not viable to directly use them for semi-autonomous protocol stack design based on user inputs and deployment feedbacks. Typically more than ten of such components are used to realize a MAC protocol [165]. These components display complex interdependencies, and it is highly challenging to autonomously access their individual contribution to the high-level performance. However, these elements may be combined to form a complex component that offers higher level service(s) on which semi-autonomous reasoning in CONFab is viable. As an example, medium access service provided by MAC protocols can benefit from distinction of a *preamble sampling* and/or *neighborhood sleep schedule* functionalities which are common to many of WSN protocols [169]. Similarly, CONFab can reason on different routing services such as the tree- or the cluster-based routing. Our high-level decomposition of routing protocols includes 1) a *beaconing* component, which broadcasts protocol specific knowledge set; 2) *forwarding engine* that calculates efficient routes based on indicated routing metrics; and 3) *routing daemon* storing the routing information. These components have compatible interfaces, and different solutions can be realized using these three basic building blocks. For instance, the link estimator module, which provides the expected transmission index (ETX), or the node energy detector component enable alternative routing metrics.

The results from Section 3.4.2 indicate that our approach to decomposition allows realizing different protocol behaviors without loosing performance as compared to most of the monolithic implementations. In fact, the component-based realization allowed us to easily change the functionality of one of the protocols used in our case studies (S4 [160]), which lead to the improvement in energy efficiency by approximately 50 %. At the same time granularity of the established services, developed components and the complexity of their interconnection logic remained simple enough to be easily accommodated for automatic reasoning such as enabled by in CONFab.

3.3 CONFab Design and Architecture

CONFab may be seen as a knowledge base with a number of support modules. The knowledge base structure is defined by the ontology, which also dictates how inference will be done on the KB entities. The inference process allows determining alternative protocol stack compositions using information on possible wirings of behavioral (functionalities + services) and then structural (components + interfaces) modules. During this process the meta-data of these entities is *qualitatively* checked against user scenario specification.



(a) Sample CONFab screenshots that include the GUI, the Protégé development environment, the FaCT++ server, and some of the Java background processing.

```
<Scenario rdf:ID="HabitatMonitoring">
  <Scenario_hasBatteryCount rdf:resource="#SBC_4"/>
  <Scenario_hasFaultTolerance rdf:resource="#SFT_Low"/>
  <Scenario_hasLifeTime rdf:resource="#SLT_4-Months"/>
  <Scenario_hasNodes rdf:resource="#SN_31-70"/>
  <Scenario_hasTopologyDynamics rdf:resource=
                                  "#STD_None"/>
  ...
</Scenario>
<owl:ObjectProperty rdf:ID="Scenario_hasNodes">
  <rdfs:domain rdf:resource="#Scenario"/>
  <rdfs:range rdf:resource="#Scenario_Nodes"/>
  <rdfs:subPropertyOf rdf:resource=
                        "#Scenario_Object_Param"/>
  ...
</owl:ObjectProperty>
<Scenario_Nodes rdf:ID="SN_31-70"/>
<Scenario_Nodes rdf:ID="SN_GreaterThen70"/>
<Scenario_Nodes rdf:ID="SN_LessThen30"/>
<Scenario_Nodes rdf:ID="SN_NA"/>
<owl:Class rdf:ID="StackComposition">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasStack"/>
      <owl:someValuesFrom rdf:resource="#Stack"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasServices"/>
      <owl:someValuesFrom rdf:resource=
                              "#SendReceive"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  ...
</owl:Class>
```

(b) Sample OWL entities. Only few illustrative parts of the definitions are given.

Figure 3.3: Sample CONFab screenshots and a short example of the OWL code [44].

The main CONFab support modules provide two functionalities. The first is the GUI used to enter scenario specification and display framework recommendations on protocol stack composition. The second module aims at post-processing of data from network deployment sites, and its automatic addition to the KB, thus enabling *feedback* for CONFab. This functionality allows the framework to enhance the quality of its recommendations as more relevant inputs become available for the system. CONFab is an extendible system. It can be enriched by additional functionalities realized as specialized plug-ins, which can be implemented in any framework and connected via a Java-interface. The ontology can be also altered, which leads to changes in the structure of the KB and enables additional integrated reasoning capabilities. Currently implemented plug-ins are described in Section 3.3.2. The overall CONFab architecture is shown in Figure 3.1.

3.3.1 Ontology, Knowledge Base and Reasoning

The core element of CONFab is the knowledge base (KB), the structure of which is defined by the ontology, and reasoning is done using an inference engine (see Figure 3.3). The KB employs the description logic. It captures in a structured manner information on the WSN environment and the task of protocol stack design. This includes data on pools of available functionalities, services, components with specifications of their interfaces and configurable parameters, description of the logic of their valid wirings, and the metadata. The latter includes estimates on the performance of whole stacks conditioned on particular operational conditions. These inputs to the KB can be provided by experts (e.g., component definitions and their characteristics), users (e.g, scenario descriptions), or they can be obtained through post-processing of feedback from the deployment sites, which is stored separately.

As said, it is the ontology that gives structure to the above information and enables

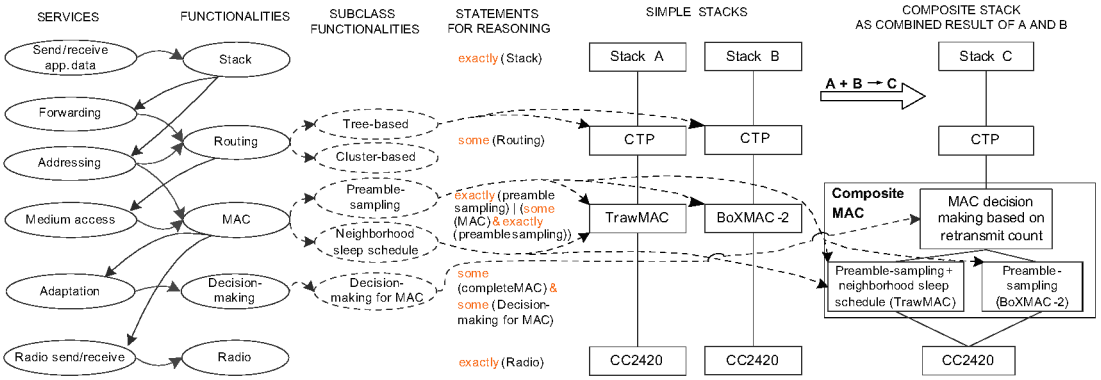


Figure 3.4: Sample relations between classes and reasoning statements that lead to different stack compositions [44]. The displayed stacks are among the ones used during the evaluation process. Stack C is the combination of stacks A and B. The stack combination functionality is enabled through sub-scenario definitions and labeled graph comparisons.

inference on it[3]. Recapping, an ontology has three main types of entities: classes, properties, and individuals. Classes and properties have a hierarchical organization. Individuals are instantiations of classes. Classes are basically defined by their properties, the values of which are result of logical qualitative operations on other classes and individuals[4] For example, in Figure 3.3b we show some of the property values that define the habitat monitoring scenario. The inference process ensures the consistency of data in the KB, by, e.g., classifying individuals, finding equivalent class definitions, and adding missing property statements into subclass definitions of a class. In case a conflict arises on the ontology structure that cannot be resolved by the inference engine it is reported to a user. There also exist possibilities for partial inference on selected subset of classes. For example, we can invoke a form of the service-oriented design by executing inference only on the level of functionalities and services, i.e., perform behavioral reasoning. From the other side we can perform structural reasoning tracking inter-component dependencies and checking for interface compatibility. Combined inference on behavioral and structural ontology classes and their objects (individuals in OWL terminology) leads to derivation or identification of the protocol stack that is likely to perform well for a given scenario. Once the whole inference process is done on the KB and it is found to be consistent, the queries to the KB can be formulated using the Jena [196] or SPARQL [206].

The inference engine of the KB operates on sets of values, without making distinctions between those falling into one particular range. In other words, a qualitative reasoning on network performance is performed on a range of values (see Figure 3.3). For example, currently, our scenario definition has two basic characteristics that define expected application rate {“low” (< 0.01 pkt/s), and “high” (> 0.01 pkt/s)} and number of nodes in a network {“low” (<30 nodes), “medium” (30-70 nodes), and “high” (>70 nodes)}. We also discretize performance values, e.g., power consumption, in a similar manner. Generally, hierarchical discretization, partial inference and querying allows splitting the reasoning process into several stages. This provides possibility to offload computation as a back-

[3]The ontology we propose is a more complex version of the ontology for software configuration management presented in Appendix B.

[4]OWL ontologies support not only objects, but also data and annotation types of properties. But these require less explanations and do not define the core of the CONFab ontology.

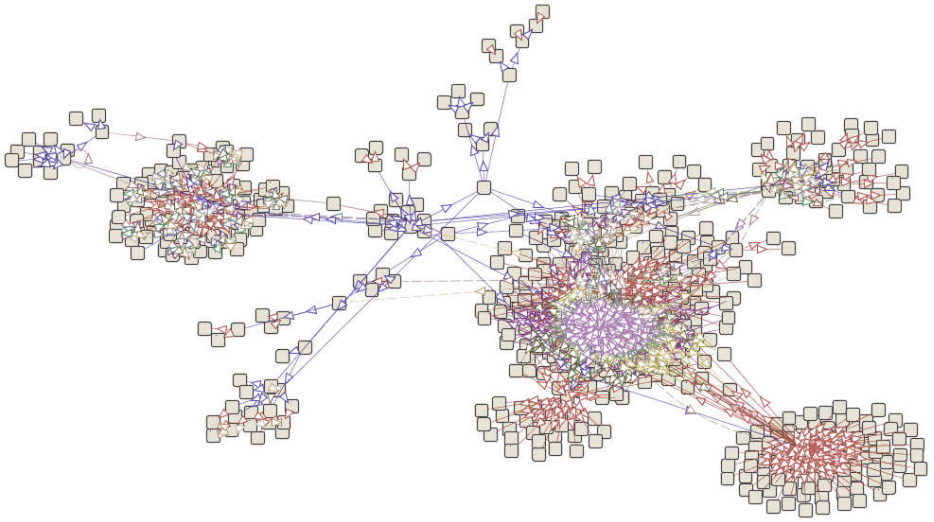


Figure 3.5: A snapshot of the ontology, where rectangles correspond to ontology classes (most of which are automatically generated as result of post-processing of experiment results), and arrows denote property relations.

ground batch processing, and reuse results of prior inference for other scenarios.

Hierarchical organization of ontology is employed to organize such relations as "a protocol is a subclass of component, MAC is a subclass of protocol, and routing functionality has subclasses of tree-based and cluster-based routing" (see Figure 3.4). It is also used to reason on particular individuals. For example, for each scenario we create the respective component, service and functionality subclasses, in which individuals suitable to this scenario are classified (in OWL one individual can belong to several classes simultaneously). Such structure, also enabled through transitive and inverse property relations, allows us to perform scenario-specific queries.

Figures 3.4, 3.5 and 3.6 illustrate major entities of the ontology and show selected intermediate results of the reasoning process. Though Figure 3.5 shows a large number on ontology classes, it is not difficult to maintain and evolve this structure, as most of the entities are automatically created in response to either a creation of a new scenario or feedback data from a deployment site. Ontology classes and subclasses capturing functionalities and components loosely follow the OSI layer structure, with each layer being further subdivided according to its functional specifics (see Figures 3.4 and 3.6). CONFab starts from the upper-level behavioral reasoning on required services, followed by deduction of functionalities that provide those services. They are then mapped to components and interfaces, respectively, which leads to a valid protocol stack definition, which can be finally mapped to relevant TinyOS/nesC components. These stack components can then be automatically combined through appropriate interfaces and deployed using the system and the corresponding XML-based meta-language as is in [51, 165].

The overall process is shown in more detail in Figure 3.7. The displayed flowchart also shows how some of the plug-ins being invoked during the CONFab decision process. As we discussed earlier, it is difficult to model in detail the influence of a separate component on a network stack without consideration for its interdependencies with other elements of a stack. Therefore, on the first stage of reasoning we operate on functionalities and components using only expert inputs, that state if a certain elements is at all suitable for,

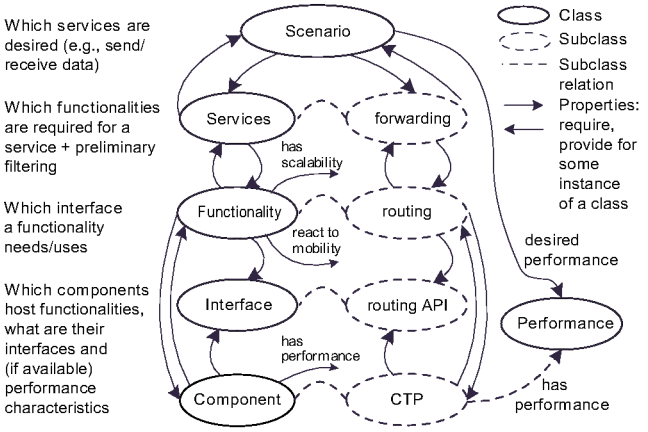


Figure 3.6: Sample classes with their chosen properties and their interrelations [44]. The figure displays sample behavioral classes (services and functionalities) and structural classes (interfaces and component) with with their chosen properties and their interrelations. The composition of a stack is guided by specifications captured as part of the scenario description.

e.g., large-scale deployment or high mobility conditions. For our implementation, we, for instance, decide against the CTP routing protocol if we have to deliver packets to multiple nodes, as this protocol typically serves only one destination (a gateway). We also consider the compatibility of interfaces. The result of this stage of reasoning is a valid list of protocol stacks with their components and parameter configurations. These provide services required by a user and fulfill basic scenario requirements. On the second level of reasoning we are doing performance-specific filtering. We also bring additional scenario details into the picture, such as considering protocol stacks with memory footprint fitting the deployment hardware (TelosB motes). Here we still employ qualitative assessments. After the set-based performance evaluation is performed, if enabled, we execute qualitative utility-based reasoning on the remaining stacks. Currently we are processing the average performance metrics that are stored in the KB. Later we may consider enquiring a database with raw data for more sophisticated post-process. We tried to decompose reasoning into stages significantly different by amount of utilized scenario information. Using the suggested scheme we can, as a background process, generate all possible protocol stack combinations that are compatible on service- and interface levels. Later we can consider for reasoning only these configuration, and consecutively filter them based on particular scenario requirements.

When providing recommendations to the user, CONFab lists not only stacks that differ in components, but also in their parameter settings. Consideration for parameters are important, as they, first, severely impact protocol stack performance. Second, behavior of typical monolithically implemented protocols can only be adjusted only through their parameter configuration settings. A typical list of protocol stack configurations for a particular scenario can be very large, until there are strict constraints either on allowable range expected performance values, or number of configurable parameters and granularity of their values. Recommendations to the user are listed in decreasing order of expected performance and can be reordered on the basis of specific KPI metrics (see Figure 3.3a). In one of our plug-ins we introduce mechanisms for autonomous probing of suggested recommendations in the target deployment environment. This plug-in is based

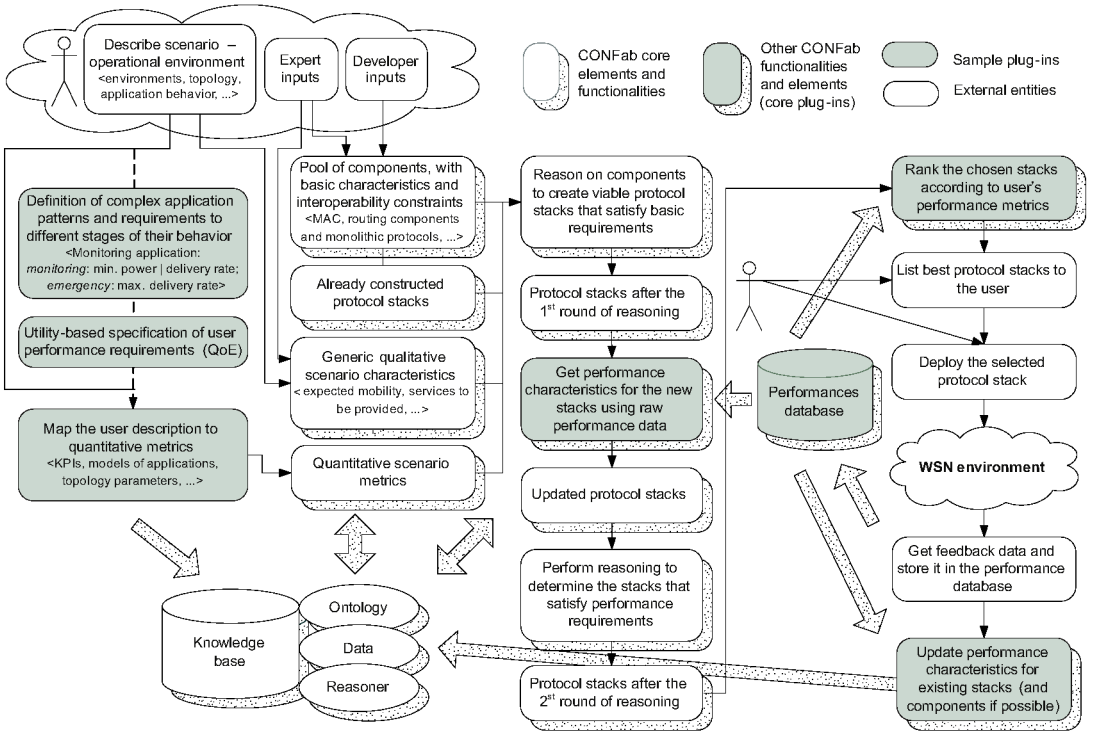


Figure 3.7: The decision/reasoning logic followed by CONFab.

on metaheuristics and avoids exhaustive search. Currently in order not to explode the optimization search space we severely limit the number of parameters considered. For our main evaluation we explore only the duty cycle parameter. For the plug-in experiments we consider seven protocol parameters at multiple layers. The stack configurations are also limitedly checked for compatibility in run-time distributed behavior. For example, typically we cannot have different duty cycle settings on network nodes as this will likely prevent them from communicating. However, we can set the maximum MAC-layer retransmission count on each node individually. The run-time distributed adaptivity of a parameter is indicated through a dedicated property.

We conclude the discussion on CONFab design with a short description of how performance results from existing deployment are recorded in the knowledge based and how they are utilized to provide recommendations on protocol stack composition. Currently the feedback data that is gathered from deployment sites is stored in the KB as a new object in case if such stack-scenario combination was not encountered before. If a different deployment trial is added for the same scenario and the protocol stack combination (including its parameter configuration) then the database storing the raw results is queried to obtain average values for all existing trials. This average result, after classification into one of the qualitative performance metric classes, updates the ontology individual, which stores data on the corresponding stack/scenario deployment. The non-discretized average values are also recorded by the same individual in the dedicated data properties. The same procedure applies when choosing stack recommendations for the user. All the stacks fulfilling user requirements for the same or more conservative range of scenario characteristics are shown. Later they can be either sorted according to their performance metric in the GUI (see Figure 3.3), or additionally filtered based on scenario-specific util-

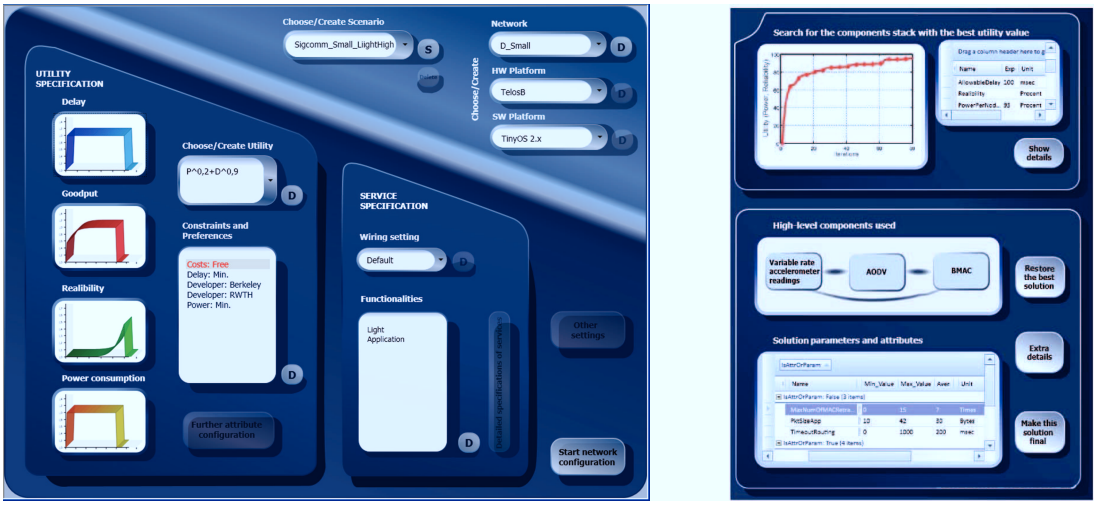


Figure 3.8: The screenshots of a variant of the GUI with utility function setup.

ity functional and recorded averages. These are the basic mechanisms of how CONFab employs experience from prior deployments or expert feedback. We acknowledge that such data processing is rather rudimentary. However, the current implementation allows demonstrating the applicability of the CONFab approach. In future we plan to enhance this side of the proposed framework. Next, we discuss some of the additional modules developed for CONFab.

3.3.2 Selected Plug-ins

Based on our prior research we implemented several plug-ins for the CONFab. These are 1) the utility-based ranking and additional quantitative processing of KB data, 2) the graph-based comparison of existing stacks and their merging, 3) the autonomous scheduling of new experiments with varying protocol stack configurations that is based on a genetic algorithm and aims to find an optimally performing stack, and 4) the automatic WSN stack deployment onto our customary testbed and performance feedback collection [43, 51]. Below we describe in detail the plug-ins #2 and #4.

3.3.2.1 Active Machine Learning based State Space Exploration

The core of CONFab recommends a set of protocol stacks (components and their parameter values) based on the information contained in the KB. However, the framework cannot directly compile and deploy these stacks. Neither it can devise a sequence of experiments, through which these stacks would be probed to find the best performing for the particular deployment scenario. In other words the core of CONFab does not support the *autonomous* active exploration of the optimization state space. It can only be performed through the user involvement that is needed to choose from the list of suggestions the desired stack, find appropriate software code, and schedule its deployment into a WSN. Later, the deployment feedback would also have to be imported in CONFab.

We have developed three plug-ins based on our prior work [43, 51] that enable automation of these processes. *First*, we have enabled utility-based estimation of network performance based on combination of functions of observed KPIs. The right panel of Figure 3.8 shows a screenshot of the GUI, where a user can define functions that interpret

values of a particular KPI, such as power consumption, delay, goodput, and reliability, and then combine them in a single functional. The left panel of the figure shows a part of the GUI that depicts a sample performance of the system, and provides the possibility for active autonomous exploration of the WSN protocol stack optimization state space [43].

Second, we developed a testbed, which also includes a database back-end and a scripting language [51]. Compared to other analogues, this testbed enables rapid autonomous early in-house evaluation of new WSN development over a range of scenario and protocol stack parameter settings[5]. This is enabled by simultaneous scheduling of multiple experiments to run on WSN nodes that operate on no-overlapping wireless channels. Below we provide some details on the testbed system operation.

A testbed experiment description includes a graph that defines the network topology, and a compilable nesC code that implements the desired stack. Each stack can be reconfigured using tunable parameter settings. These settings are exposed through the Makefile and in the end adjust variable values in the corresponding nesC code. An experiment description is first recorded in a database, then it gets translated to the testbed configuration, and, later, to a nesC Makefile as the scheduled experiments are getting deployment onto the testbed. Currently, we impose a MAC level filtering to generate topology, which clearly imposes a bias in evaluation, but allows for more experiments to be run it parallel. Later we plan to extend the testbed with mechanisms for estimation of real network connectivity and then schedule experiments based on this information.

The core component of the testbed system is a database that records, among others, paths to the compiled protocol stacks, a range of the configurable parameters values, deployment feedback data, state of motes in the testbed including their operational frequencies, and schedules of experiments to be run on these motes. For instance, if for any reason a scheduled experiment cannot be executed, e.g., due to temporary failure of nodes, it is re-scheduled for a later repetition, which is reflected in a database. Feedback information is gathered through TelosB USB connections at run-time. It does not have a strict format. Rather the order of sent fields, their types and sizes have to be specified through the database, so that this data can be parsed and recorded in the database on-the-fly. Basically any middleware or interface abstraction can be used for reporting of the data. For example, we have used the UDAE data access engine [207]. We also employed UDAE to demonstrate the possibility for run-time control over network configuration, as well as for early studies on run-time adaptability [43]. This testbed was important to us, as it allowed to realize active autonomous exploration of the optimization state space, which is difficult on such public entities as the Indriya WSN testbed due to their busy schedule. The testbed gave us an option of performing an exhaustive search among all suggested protocol stacks and their parameters on a number of topologies for a given scenario specification.

Third, we adopted a genetic algorithm, that belongs to the class of metaheuristic search methods, for exploration of the optimization state space of a WSN scenario. Sometimes CONFab does not have enough inputs to recommend the appropriate protocol stack for the user. This can happen, for example, if the given scenario is very different from the results of others recorded in the KB, or there is no evaluation data for a new WSN component. In such cases one can do active exploration of the optimization state space through multiple deployments of different protocol stacks that are scheduled to find a

[5]For a fixed set of components, any component-oriented optimization problem is clearly isomorphic with a parameter-based one, with the parameter space enumerating the different possible wirings of the components together with their respective individual parameter settings.

near-optimum configuration while avoiding exhaustive search. The core of CONFab is queued to provide a list of protocol stacks and their configurations that are deployable for a given WSN scenario. This set constitutes the optimization search space. Using the testbed described above we schedule for execution members of this search space, and then feedback the evaluation results to the KB. For the exploration strategy we have chosen to use a Generic Algorithm (GA) [21, 208]. The incentives behind choosing this metaheuristic were 1) high efficiency of this algorithm for practical tasks, 2) the ability to operate in non-linear optimization state space that is likely to occur due to non-linearity such parameters as the duty cycle, or possible emergent behavior of a protocol stack, 3) the ability (though not a requirement) to directly benefit from multiple simultaneous deployments as facilitated by our testbed, and 4) the offline nature of the task, where additional possible trials imposed by the population size do not have such a strong impact on users as in the case of the online optimization.

In Chapter 4 we also discuss, compare, and apply a genetic algorithm as one of the alternative optimizers for searching the optimization state space for the task of network planning. However, there we primarily use numerical simulations to generate the evaluation data. In this chapter we rely on the empirical results. First, we shortly recap on the algorithm and explain how it is adapted for the CONFab purposes. The genetic algorithm is a probabilistic metaheuristic search method that repeatedly alters a set of mathematical objects so that the derived objects earn improved performance as compared to their parents. In the terminology of the algorithm the objects are individuals that are typically defined through an array of variables — chromosomes. A set of objects is called population. Each new population can be also denoted as a generation. Individuals for a new generation are derived from individuals (their chromosomes) of a prior population, using such evolutionary-inspired operations as crossover and mutation. Goodness of each individual is evaluated according to its fitness value. Better performing individuals have higher probability to be considered for derivation of the next generation. Genetic algorithms can be seen as a combination of greedy (crossover, inheritance/selection) and adventurous (mutation) exploration strategies.

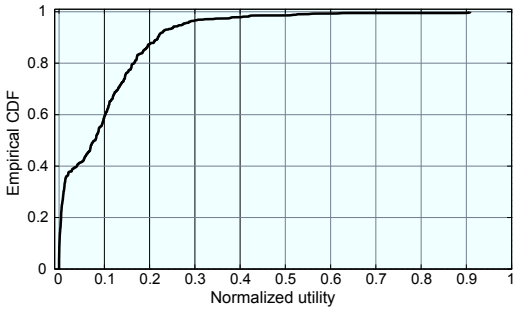
As said for our task the optimization state space is defined by a set of protocol stacks and their configurations selected by CONFab. We convert this set, first, in the tree-based representation in its canonical form [209], and then represent this tree as an array of integers, i.e., move to the parameter-based representation. For the genetic algorithm these parameters, the members of the array, become the chromosomes of the individuals. We perform the optimization using the GA realization in the MATLAB Optimization Toolbox [208], just in our implementation chromosomes can take only integer values to which the respective parameters values are mapped. We use the default MATLAB settings for most of the GA parameters, experimenting only with the population sizes, number of elite members, as well as the percentage of crossovers and mutations. The GA fitness function is the network utility functional. The initial population is defined randomly.

We have evaluated the application of the genetic algorithm on a small WSN scenario of network size of eight nodes and random connected network topology. In this evaluation we did not try investigate stacks that have alternative protocols or components, as they are discussed in other parts of the chapter. We used a single stack that includes BoXMAC-2 [202] MAC and MultiHopLQI [210] routing protocols, and a custom application that sends temperature sensory data at a regular intervals. There are in total seven tunable parameters in this stack. They affect the number of sensory readings that can be concatenated in one packet, which influence the frequency of generation of application-layer

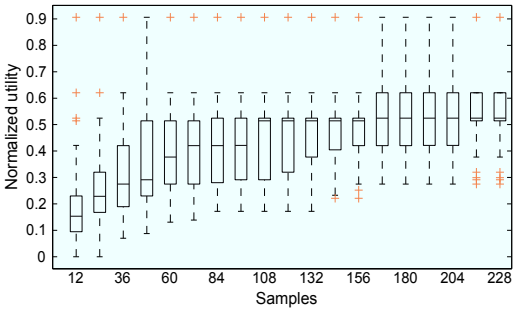
packets, the duty cycle, the beaconing interval and the timeout for the routing protocol, MAC's unicast acknowledgment, the maximum retry limit, and the backoff interval. Each parameter can take two or three values, resulting in 432 different permutations. This dataset is a subset of data studied in Chapter 4, Section 4.4. The utility functional that we consider is based on a normalized combination of power consumption and goodput. The network utility U is defined as $U = \min\{1, (|1 - P/P_0|^+ \cdot G)/U_{max}\}$, where P network power consumption, P_0 is maximum considered power consumption below which the utility is non-zero, G denotes goodput and U_{max} is maximum desirable network utility. Figure 3.9a displays the cumulative distribution function (CDF) of this utility. The overall results from running the state space exploration using different setting of the genetic algorithms are shown in Figure 3.9. As we see, all modifications of the genetic algorithm, even the one with the population size of two and no-crossover function (see Figure 3.9b), are able to stably find the best 3% of the solutions (0.5 normalized utility) in less than 108 samples, that is, 25% of all permutations. The best results are achieved for the larger population size of 12 individuals with both mutation and crossover operations present. Here the best 3% of the solutions were on average obtained with 48 deployments, that is, about 11% of the total number of permutations (see Figure 3.9d). These results show that larger population sizes allow improving the achieved results. However, the drawback of such an approach is the requirement for a quite significant for our task number of trials/samples that need to be done before a further exploration decision can be made. This makes the genetic algorithm less appropriate for cases of online network optimization, as there users might get annoyed with long sequences of state space trials that are likely to degrade momentary network performance. Thus, in Chapter 4 we investigate and modify another metaheuristic algorithm, simulated annealing, that requires only one sample to make next optimization decision using empirical data gathered in heterogeneous wireless networks. However, still, genetic algorithms are likely to be efficient for the online optimizations where a relatively long exploration phase is viable. For example, as a future extension, the proposed GA-based mechanism can be used for centralized run-time WSN management, where CONFab would be applied to suggest network settings and protocol stack configurations as part of the run-time performance optimization process. This is a viable application scenario as WSNs often can tolerate performance degradations caused by reconfigurations in the periods when the phenomena of interest are unlikely to occur. Additionally, these networks are typically cooperative, and the delay-tolerance for many applications is high. Thus, there exist no obstacles to network-wide deployment of a new configuration, and, if it is not performing well, reversion to the old setup with the resending of the data accumulated during the trial period. The network might also easily generate test data, and forbid new explorations in case if the likelihood of obtaining of crucial sensory date increases.

3.3.2.2 Composite Scenarios and New Stacks

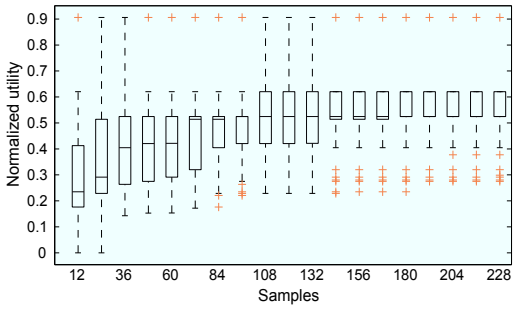
CONFab allows for two types of scenario definitions. In the basic case, we do not capture dynamics in the system behavior, and only state a range of values a certain scenario parameter can take. For example, we can define that the application rate changes between low and high, i.e., $\text{AppRate} \in \{\text{low}, \text{high}\}$. Such a definition does not provide details on the expected network behavior. It only allows predicting performance results for a certain stack based on other deployments. However, CONFab is also capable, albeit in a limited fashion, of addressing dynamics in scenario behavior, including simple variations in goal



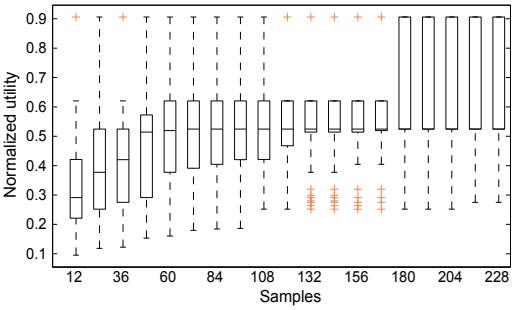
(a) Cumulative distribution function of utility over parameter value permutations obtained empirically from a testbed.



(b) Results from running the genetic algorithm with population size of two (one elite member, no crossover, and one population member undergoes mutation).



(c) Results from running the genetic algorithm with population size of eight (two elite members, from the rest 70% of crossovers and 30% of mutations).



(d) Results from running the genetic algorithm with population size of twelve (two elite member, from the rest 80% of crossovers and 20% of mutations).

Figure 3.9: Results of active optimization state space exploration on empirical data with different settings of the genetic algorithm. Each version of the genetic algorithm was executed 200 times to obtain average performance results.

statements or monitored changes in the wireless environment (caused, for example, by the diurnal cycle). This is enabled through sub-scenario definitions that are processed by the graph-comparison plug-in. This plug-in allows users not only to define a range of appropriate scenario settings, such as expected application rates or number of deployed nodes, but also to explicitly state that, e.g., the current scenario comprises 30% of behavior of the scenario A and 70% of the scenario B as illustrated in Figure 3.4.

Such scenario definitions can be exploited for derivation of composite stacks, which aim at optimizing performance in different scenario conditions and switching between those, thus facilitating an *adaptive* behavior. For example, we evaluate in Section 3.4 a scenario where an application switches between two, high and low, transmission rates. A user defines this scenario as a combination of the two sub-scenarios that differ in application rates. CONFab starts the reasoning process, and first deduces the list of recommended stacks for each of the sub-scenarios, duly sorted according to the predicted performance. The next step is to determine which stacks from these can be combined together through the introduction of a separate Decision-Making (DM) component. The decision-making component acts as a decision tree node (basically a collection of if state-

ments), determining according to the observed operational conditions when each branch of the component graph should be executed. The stacks are combined based on their labeled direct graph representations, where vertexes and edges correspond to components and their wirings, respectively. The two graphs are compared, and where their directed paths diverge in the merged version an empty vertex is added, which corresponds to a DM component. To realize this processing we added a plug-in implemented in R that uses the "graphcomp" package [211]. The output of the plug-in is a new protocol stack with additional decision-making components. These are either selected from a pool of interface-compatible alternatives already present in the KB, or in absence of the appropriate DM component, its desirable characteristics are indicated to the user. In the latter case a new appropriate DM component has to be implemented, and its description added to the knowledge base. Clearly the obtained stacks later undergo the same validity checks in the KB as described above.

In this work we considered two relevant scenarios. For the first scenario with the two iteratively changing high and low datarates CONFab has suggested to combine the energy efficient TrawMAC and the reliable BoXMAC-2 MAC functionalities using an additional decision module (see right part of the Figure 3.4). We could easily realize this suggestion using component-based implementation of these protocols. We found that switching between the respective MAC functionalities can be triggered by a combination of the retransmission count metric and the unicast/broadcast packet indications. Therefore, we obtained the CompositeMAC protocol. The CTP/CompositeMAC stack turned out to be the most efficient for our experiments. For the second scenario we considered the setup where 50% of the traffic is sent to the gateway, and the rest being forwarded to the other random nodes in the network. For this application the stack with the composite (CTP/S4) routing and the BoXMAC-2 was considered. For the given network network size and application rate the stack did not perform significantly better than the simpler alternatives, providing a counter example, which shows that the composite behaviors do not always lead to the improved results. Further these scenarios are discussed in Section 3.4.3.

3.4 Validation and Performance Evaluation

We validated CONFab using the empirical data gathered on the Indriya [146] TelosB [162] testbed at the National University of Singapore during 700+ hours of experiments (see Figure 3.10). As stated in Section 3.2.3 we studied three MAC and two routing protocols, considering both their monolithic and component-based implementations. Using the component-based design we also rapidly prototyped three new protocols employing the developed MAC and routing components. One of them is the improved version of the S4 routing protocol [160], which displayed significantly better performance then its predecessor. Another is a new effective MAC protocol called *CompositeMAC* that combines functionalities of the TrawMAC [212] and the BoXMAC-2, see Section 3.4.3. The third is an experimental routing protocol that combines functionalities of the S4 and the CTP.

We evaluated the CONFab and the related WSN modules using the settings summarized in Table 3.1a. We altered the duty cycle parameter to all of the protocol stack combinations, as well as the option for hardware acknowledgement for the TrawMAC [212] to show that CONFab can operate on both parameters and components. The available duty cycles range from 2.2% to 9.9% with a channel sensing time of 11 ms. We considered the scenarios with four different application behaviors and four different network sizes. All nodes besides a gateway generate traffic with the packet sizes 16 bytes. They are im-

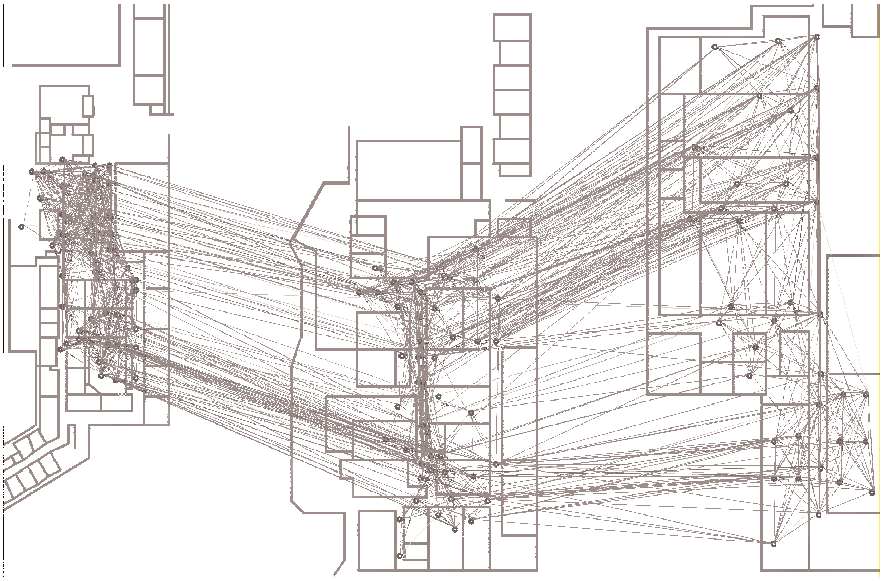


Figure 3.10: An overview of the Indriya TelosB testbed at the National University of Singapore [146]. The 127 node testbed is deployed over multiple floors. Experiments are mostly done on nodes of the floors shown at the left and center parts of the figure.

posed a small random delay to avoid simultaneous generation of application packets. All the experiments were conducted for a duration of 15–30 minutes, and split into the three phases. The first third of experiments was dedicated for node bootstrapping and routing topology setup. Application data was sent during the second part of the experiments. Finally, in the third part applications no data was generated, and the network traffic was constituted from unsent packets on nodes' caches or control messages. The data of the state of the node and its performance was logged every two seconds, with the reporting of the send and received packet at different protocol layers was done. Each protocol stack combination was deployed in a specific scenario for three times. Performance results were gathered using additional nesC software modules that monitor energy consumption similarly to the approach suggested in [213]. These results were later in part fed into the CONFab to build its knowledge base, and in part were used to evaluate framework's reasoning. In the next sections we first present the results on overall CONFab functioning. Then we show the performance of the specific components and stacks developed as part of the work on CONFab.

3.4.1 Protocol Stack Composition Using Performance Prediction

We evaluate the protocol stack composition suggested by CONFab for a scenario that is not already present in the KB. We model it after the habitat monitoring testcase [214] with the network size of 50 nodes and application datarate of 0.02 pkt/s, see Table 3.1b. The scenario goal is set to provide the high PRR of at least 90 %, and maximize the network lifetime. In order to recommend protocol stack for a particular scenario CONFab, first, maps the scenario characteristics and requirements into the ontology classes that reflect protocol-, not user-KPIs. For example, the desirable network lifetime with specification of the number and the type batter sources is mapped to energy consumption over a specific time period (for instance 5 minutes). If these inputs are numerical they are discretized

Evaluated scenario variables	Values as specified by the user and (actually deployed)
Application data rate	{Bursty} U {Low, High, Low-High} (0.01 and 0.1 pkt/s)
Number of nodes	<30, 30-70, >70 (20, 40, 80)
Scalability	No pre-defined topology changes
Optimization goals	PRR, Power
Routing protocols	S4, CTP, S4-CTP Composite
MAC protocols	BoxMAC-2, TrawMAC with/without hardware ACK, CompositeMAC
Components of interest	Link Estimator, Decision Making
Duty cycle [%]	9.9, 4.2, 2.2, 1.1, 0.6

(a) Main scenario characteristics and protocols used in the evaluation section.

Scenario Parameter	Selected (Deployed) Values
Application data rate	Bursty Low (0.02 pkt/s)
Optimization goals	Lifetime, Reliability (PRR, Power)
Robustness	High (translates to PRR)
Scalability	Low (No changes)
Number of nodes	30-70 (50)
Application PRR, >	80%
Power source type	C-battery
Number of power sources	4
Network lifetime, >	4 Months
Mobility	None

(b) Main characteristics of the habitat monitoring scenario.

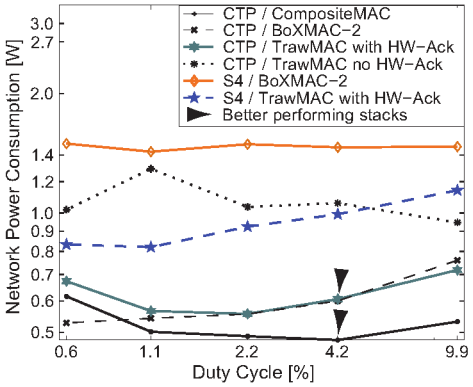
Table 3.1: Summary of the main scenario characteristics for Section 3.4. For some parameters the tables show both the values as entered by the user, as well as (in brackets) either their translation to the protocol/scenario settings, or the actual values used during the evaluation. For example, the desired network lifetime attribute, the number and the capacity of the battery is translated to the energy consumption metric.

in to the respective qualitative classes to enable CONFab reasoning. Otherwise, if the inputs are already given as class values (e.g., medium network size with low application rate) this step is omitted. Then, the framework queries the KB for performance results in similar deployments, i.e., scenarios with characteristics falling into the same or nearby classes. In our case, there are records for 40 and 80 node deployments with low and high application rates of 0.01 and 0.2 pkt/s. Combinations of these values relate CONFab to four different scenarios, which results are queued. Their union are considered to be solution to the problem. Otherwise, if utility based reasoning is enabled, it combines and orders the performance results for these scenarios in a weighted manner in order to obtain more accurate the prediction estimates. Further, the possible solutions are filtered to fit the scenario constraints, and the respective ordered list of recommendations is shown to the user. We have introduced the weighted approach, since solely using the quantitative ontology-based reasoning makes the framework vulnerable to definition of the quantization class boundaries. However, if these are wisely defined or adjusted on the scenario specific basis, this modification can be omitted, or used to further improve the robustness of the solutions.

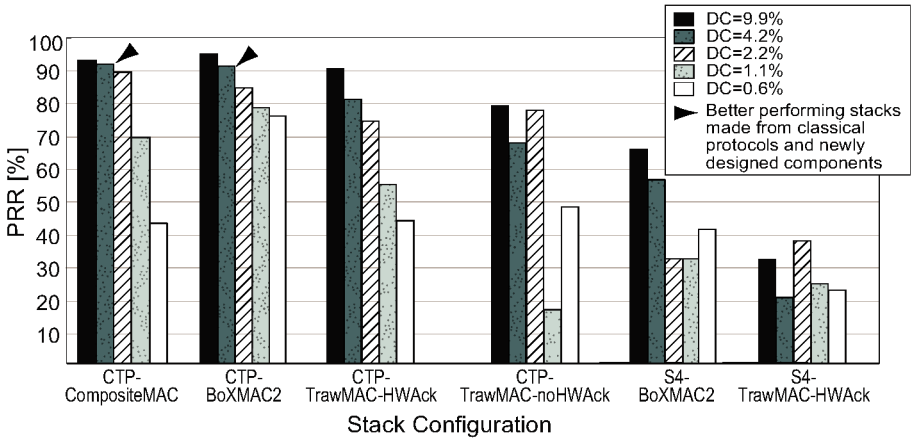
Figure 3.11a shows the table of protocol stack configurations recommended to the user that reflects both the expected and the actually obtained performance for these stacks. The table displays not only the sample protocol stack configurations that are expected to provide the PRR above 90%, but also some of the stacks, which PRR exceeds 80%. This is done to more fully illustrate the performance of the framework. The table shows the predicted power consumption and the application level PRR at the suggested duty cycle values for different protocol stacks during the stable operating regime (after the network topology and the routing paths have been established). These values are shown under *Stack*, *DC* and *Expected PRR* and *P* columns, respectively. The results obtained from actual deployments of the recommended stacks for the given scenario are shown in the last column in the table. It is worth noting that besides the monolithic combination of protocols, CONFab also suggests the component-based protocol stack, that is, the CompositeMAC together with CTP. As we see the actual (“real” in the table heading)

	ID	Stack ROU	Stack MAC	DC [%]	Expected PRR [%]	Expected P [W]	Real Perf. PRR [%]	Real Perf. P [W]
All stacks considered	► c10	CTP	Composite MAC	4.2	>90	<0.6	92	0.48
	c11	CTP	Composite MAC	9.9	>90	<0.6	90	0.50
	c09	CTP	Composite MAC	2.2	>80	<0.6	93	0.53
Not decomposed stacks considered	► p14	CTP	BoXMAC2	4.2	>90	<0.7	92	0.60
	p20	CTP	TrawMAC	9.9	>90	<0.8	91	0.72
	p12	CTP	BoXMAC2	9.9	>90	<0.8	95	0.76
	p18	CTP	BoXMAC2	2.2	>80	<0.6	84	0.56
	p16	CTP	TrawMAC	4.2	>80	<0.6	81	0.61

(a) The CONFab predicted estimates and the actual performance metrics for the selected stacks.



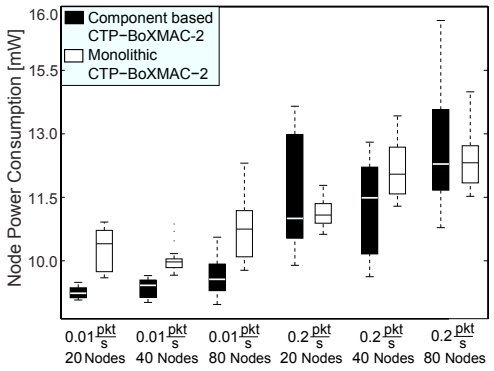
(b) Network power consumption of different protocol stacks at different duty cycles.



(c) Application packet success ratio of the selected protocol stacks at different duty cycles.

Figure 3.11: Performance estimations by CONFab and the actual deployment results in the habitat monitoring scenario for selected protocol stack configurations [44].

and the expected performance results differ. However, they follow the same tendency as predicted by the CONFab, i.e., CTP/CompositeMAC protocol stack with the DC of 4.2% still outperforms all other alternatives, and out of the monolithic implementations the CTP/BoxMAC-2 with DC of 4.2% or 9.9% also perform the best as expected. The overall results obtained from deployments of the recommended stacks, as well as other available protocol stack combinations for the scenario of interest are summarized in Figures 3.11b and 3.11c that show the average network power consumption and the application packet reception ratio. These data confirm that CONFab is able to predict relative the protocol stack performance with high accuracy. These results also show that the protocol stack with the highest expected performance has a component-based realization, and includes the newly suggested CompositeMAC protocol (see Section 3.4.3). At a duty cycle of 4.2% it outperforms the best monolithic protocols based stack (CTP/BoXMAC-2) by showing 27% improvement in energy efficiency at the same PRR rates of above 90%.



(a) Average per node power consumption. Error bars correspond to 5th and 95th percentiles of the data.

Network size	Application rate [pkt/s]	PRR of CTP [%]	
		Monolithic	Components
20	0.01	97.2	97.1
40	0.01	93.4	94.6
80	0.01	84.4	86.8
20	0.2	89.9	90.8
40	0.2	80.5	82.7
80	0.2	72.3	75.5

(b) Average application packet reception ratio.

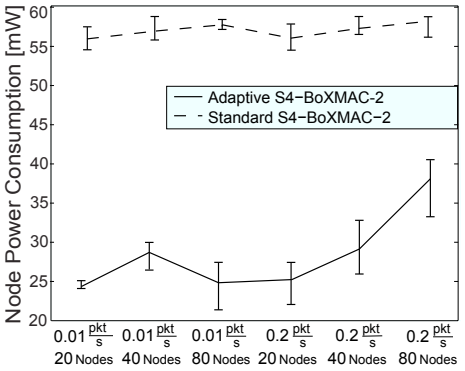
Figure 3.12: Results showing the behavior of the CTP/BoXMAC-2 stack, considering its monolithic and component-based implementations [44]. Performance is obtained over the whole operational period, which includes the stages of the network setup, stable operation, and when no applications are running, rather packets left in buffers are sent.

3.4.2 Component-based Protocol Implementations

Above we demonstrated that CONFab can successfully operate on protocol stack configurations that include both monolithic and component-based protocol stack realizations. In this section we extend this discussion further by providing selected evaluations of our component-based protocol implementations. First, we aim to show that the respective designs lead to performance comparable to monolithic implementations. Second, we discuss on how component reuse can enhance protocol's behavior at minimal implementation overhead.

Figure 3.12 illustrates the first statement, where performance of the monolithic and component-based protocol stacks CTP/BoXMAC-2 are compared for scenarios with varying traffic and network sizes. For both stacks the duty cycle parameter is set to 2.2%. As we see the PRR displayed by these stacks does not vary by more than 4%. The variations in the per node power consumptions are higher, but in the worst case their averages do not differ by more than 14%, with the mean difference being less than 10%. The memory footprints for the monolithic and the component-based implementations are comparable, and the performance variations are within the expected limits for different implementations. We have also evaluated the other protocols that were re-implemented using components, and the respective results also do not show high deviation from the standard implementations. Therefore, as was shown by other researchers [165–168] as well, the price of decomposition can be paid for WSNs.

The next aspect of the discussion is component re-usability. As discussed earlier for CONFab we need relatively large components that are viable for the service- and functional-based abstractions, so that autonomous reasoning which aims at protocol stack composition can effectively utilize these entities. In Section 3.2.4 we have already named the abstractions and the main components that we consider in the current CONFab ontology. Here we demonstrate a vivid example of benefits of re-usability of our components for the S4 routing protocol [160]. The standard realization of the S4 uses periodic beaconing, i.e., each node periodically broadcasts ETX metrics of its neighbors to establish bid-



(a) Average per node power consumption.

Network size	Application rate [pkt/s]	PRR of S4 [%]	
		Standard	Adaptive
20	0.01	95.1	94.5
40	0.01	48.3	51.2
80	0.01	12.7	16.4
20	0.2	93.0	84.8
40	0.2	11.2	18.1
80	0.2	6.6	9.5

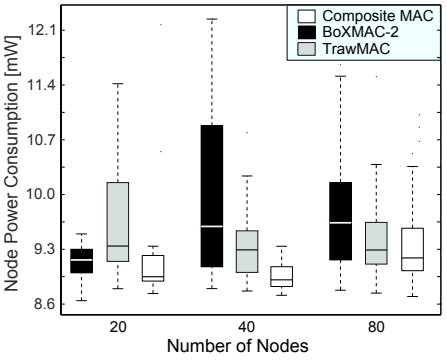
(b) Average application packet reception ratio.

Figure 3.13: Results showing the behavior of the S4/BoXMAC-2, considering the standard S4 implementation with the periodic beaconing and its modification with the adaptive Link Estimator component [44]. Performance is obtained over the whole network operation period, which includes the stages when the network is setup, operating stably, and when no applications are running, rather packets left in buffers are sent.

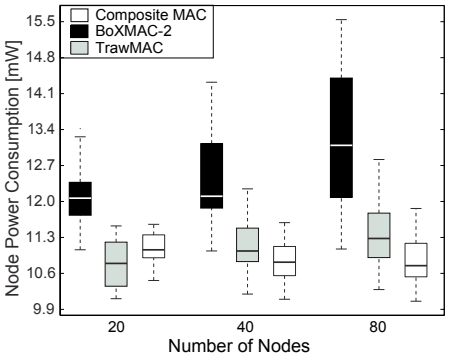
irectional link qualities. Obviously significant bandwidth can be saved if the rate of these control messages can be adapted based on observed network conditions. The CTP routing protocol [161] piggybacks the respective metrics to each broadcast transmissions, thus tying the link quality exchange to the other traffic in a network. This approach, though possibly causing additional routing delays in cases of highly unstable links, works well on practice in typical indoor WSN deployments. Therefore, we have substituted the *Link Estimator* component of S4 with the one for the CTP. Figure 3.13a shows that for the new version of the S4 the average power savings of about 50% are achieved for a range of scenarios with network sizes varying between 20 and 80 nodes and the applications rates of 0.01 and 0.2 pkt/s. The packet reception ratio remains comparable for both variation of the protocol as evident from Table 3.13b. This table also shows that as the network size grows the PRR values severely decrease. This is caused, in part, by the absence of queues in S4, which leads to packets drops under a heavy network load as was also indicated in [160]. The S4 example clearly indicates that careful interface design enforced by the component-based design approach increases re-usability in the system even for large modules, which may lead, as demonstrated above on the example of the S4, to considerable performance improvements at a minimal additional implementation costs.

3.4.3 On Composite Functionalities and Component Granularity

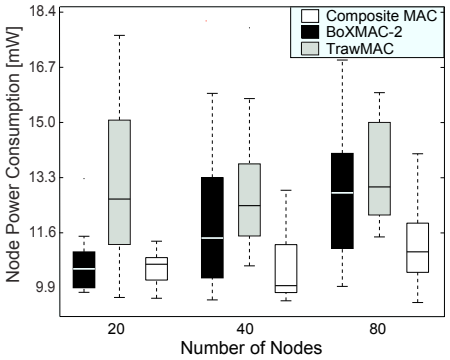
In this section we further evaluate the performance enhancement possibilities provided by the component/service-based design as facilitated by CONFab. We illustrate how the CONFab plug-in for the graph-based protocol stack comparison can be utilized. We also, based on discussions in [215] and in Section 3.2, evaluate usability of components of various granularity for autonomous CONFab reasoning. We consider two case studies. In the first one we study possible incentives, abstractions and decompositions for combining the BoxMAC-2 and the TrawMAC protocols into a new CompositeMAC protocol. In the second case study we define a scenario for CONFab that leads to combination of the S4 and the CTP routing functionalities.



(a) Per node power consumption for *low* application rate during stable operating regime (after the network setup).



(b) Per node power consumption for *high* application rate during the network setup phase.



(c) Per node power consumption for *high* application rate during stable operating regime (after the network setup time).

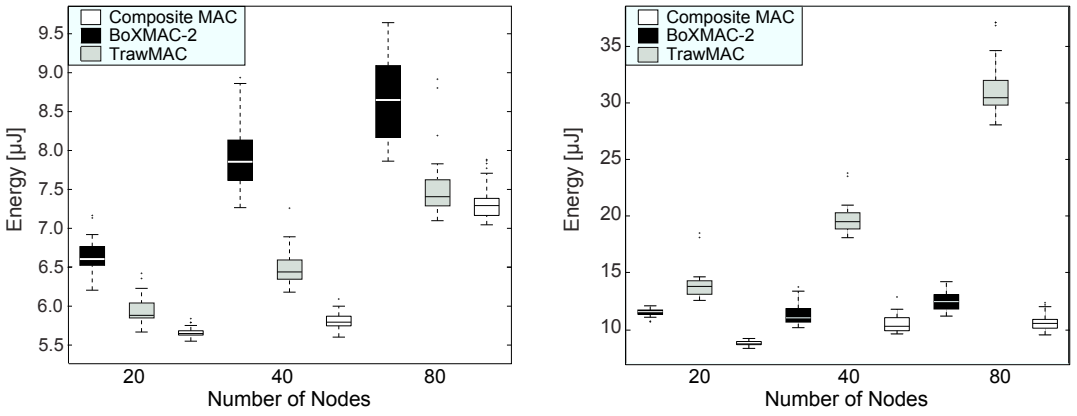
MAC	Netw. size	Average # retransmissions per node	PRR
BoxMAC-2	20	0.30	90.8
TrawMAC	20	1.34	87.4
Composite MAC	20	0.16	89.3
BoxMAC-2	80	0.09	77.5
TrawMAC	80	0.65	54.2
Composite MAC	80	0.06	71.6

(d) Average PRR comparison during the stable operating regime (after the network setup).

Figure 3.14: Comparison of BoXMAC-2, TrawMAC and CompositeMAC. Application rate of 0.2 pkt/s is used for a network consisting of 20, 40 and 80 nodes. CTP is used as the underlying routing protocol. The duty cycle is 2.2% [44].

3.4.3.1 CompositeMAC

The design of TrawMAC [212] suggests its high energy efficiency as compared to such state-of-the-art protocol as BoxMAC-2 [202]. Our experiments have only partially proved this statement. Figure 3.14 illustrates that for low application rates, as well as during the network setup phases TrawMAC is indeed more energy efficient than BoxMAC-2. Moreover, when network load is small their PRRs are comparable. However, when applications generate heavier traffic TrawMAC performance drops both with regard to PRR and power spending, see Figures 3.14c and 3.14d. After studying the protocols in detail we discovered that, indeed, TrawMAC [212] considerably reduces idle listening and overhearing performed by network nodes. However, it also causes *deafness* of potential receivers in situations that can be explained by the classical hidden terminal problem. The TrawMAC sender notifies all overhearing nodes of duration of its transmission forcing them into sleep. Therefore, in cases of hidden terminals, packets from other senders will be simply lost as their receivers would be sleeping, and retransmission will be triggered.



(a) Low-traffic rate.

(b) Low+high traffic rate.

Figure 3.15: Energy consumption per node per successful packet for a network with variable traffic rate [44]. After the topology setup, the application traffic rate is 0.01 pkt/s for a duration of 300 s. Later on, the application traffic rate is changed to 0.2 pkt/s for 200 s. CTP is used as the routing protocol. The duty cycle used is 2.2%. For low application rates the PRRs are similar for these protocol stack combinations, so they are not shown.

In reality such behavior is not optimal as receivers can still successfully decode packets even in the classical hidden terminal cases if the SINR of the relevant signals are high enough, which is obviously not allowed by the TrawMAC. Therefore, this MAC protocol is the most useful in very dense close-to-mesh network deployment scenarios with most nodes hearing each other, or in scenarios when the traffic rates are low enough so that the deafness situation is overall avoided.

Ideally CONFab should be able to identify situations when protocols, such as TrawMAC, performs suboptimally and suggest alternative stack configurations for such operational contexts. For this three basic questions are to be answered within the system:

- What are the right contexts (subscenarios, observations, their metrics and frequency) to detect possibilities for protocol stack improvement?
- What is the right component granularity to enable efficient stack reconfiguration?
- What are the right scenario models that allow to comprehensively study protocol stack components and their configurations?

Clearly, each of these questions requires a comprehensive research study by itself, and CONFab, being only an enabling platform that aims demonstrating feasibility of the semi-autonomous protocol stack optimization, cannot address them in detail. However, our system does provide interfaces and structure in order to enable complex solutions targeting these questions. In the current implementation we attend these questions in a simple manner to demonstrate that appropriate mechanisms can be incorporated into the framework.

We illustrate these on example of the TrawMAC and BoxMAC-2 MAC protocols. Based the assumption that we know the behavior of the two MAC protocols, as well as on the above analysis of TrawMAC, we can opt at development of a new improved MAC protocol either by going deeply into development or redesign, or by recombining basic functionalities of existing protocols, which following the component-based design would create some code/memory footprint overhead, but save development efforts. For the first case

we cannot use CONFab. For the second we can ask CONFab to combine the a subset of protocol stacks that use these protocols and obtain a component-graph of a new stack, as described in Section 3.3.2.2. The resulting component graph consists of a chosen routing protocol (e.g., CTP), and a dummy decision unit that switches between the TrawMAC and the BoxMAC-2 behaviors, see Figures 3.4 and 3.16a. The combination of the decision unit and two MACs can be defined as a new *CompositeMAC* functionality.

A similar stack recommendation from CONFab would be also obtained in case if we define a composite scenario, where all application traffic is directed to a single gateway, and the traffic patterns periodically change between low and high application datarates. For such a combined scenario will force the system to compare performances in the two subscenarios with low and high application traffic (see Figure 3.15), which would result in CONFab trying to combine the respective MAC protocols. Unfortunately, currently CONFab does not consider the problem of scenario decomposition into aspects that might highlight the difference between protocol behavior conditioned on the set of protocol considered. Therefore, until a scenario is formulated as a collection of appropriate subscenarios it misses the arising optimization opportunities. For example, current CONFab does not distinguish between network bootstrapping and stable operation phases, as well as broadcast and unicast traffic, and, thus, can neither obtain nor analyze data presented in Figures 3.14b and 3.14c, which provided us hints on decision criteria for switching between the MAC behaviors. It is an interesting future work to extend CONFab so that the framework can perform such an analysis. The required modeling could be, for example, based on the structure and dynamics of network connectivity, sensory events, and the resulting traffic. Care should be taken that such subscenarios (contexts) are not too small (to avoid combinatorial explosion), and the transition phases and mixtures between the subscenarios provide predictive, not harmfully emergent, behaviors. This task is also closely related to another future work on more accurate scenario characterization through active network probing.

The CompositeMAC protocol require efficient merger of the TrawMAC and BoxMAC-2 implementations, and the creation of a functional decision making unit. We found it efficient to switch to the TrawMAC behavior if there is a broadcast message to be transmitted or a unicast message is sent for the first time. In case of retransmissions the BoxMAC-2 protocol is used (see Figure 3.16a). Additionally, the decision making unit includes a wrapper over the two MAC protocols. It defines the shared data fields and structures that contain sleep schedules of neighboring nodes, and identify which MAC scheme should be used for a certain packet. The problem of the functionality merger is tightly coupled with the task of finding right component granularity, which was discussed in Section 3.2 and for which possible quantifiable metrics are provided in [215]. Figure 3.16 shows three different decompositions and abstractions of CompositeMAC. Figure 3.16c represents the close-to-implementation level of abstraction, with individual modules having a direct mapping to nesC software modules of high granularity. It is typically unproductive for such system as CONFab to operate on this level of abstraction. The analysis on a large number of components and their interfaces that have complex, but mostly pre-defined wirings, “pollutes” and complicates the reasoning rather than opens new valuable configuration possibilities. Services provided by these components are valuable when reasoning on the MAC implementations, but there are very few of them (like *Neighbor Info*) that can be used for cross-layer protocol stack optimization. Mostly with such an abstraction only a combined set of components provides meaningful service from the point of view of the protocol stack composition, which clearly indicates wrong level of granularity for CON-

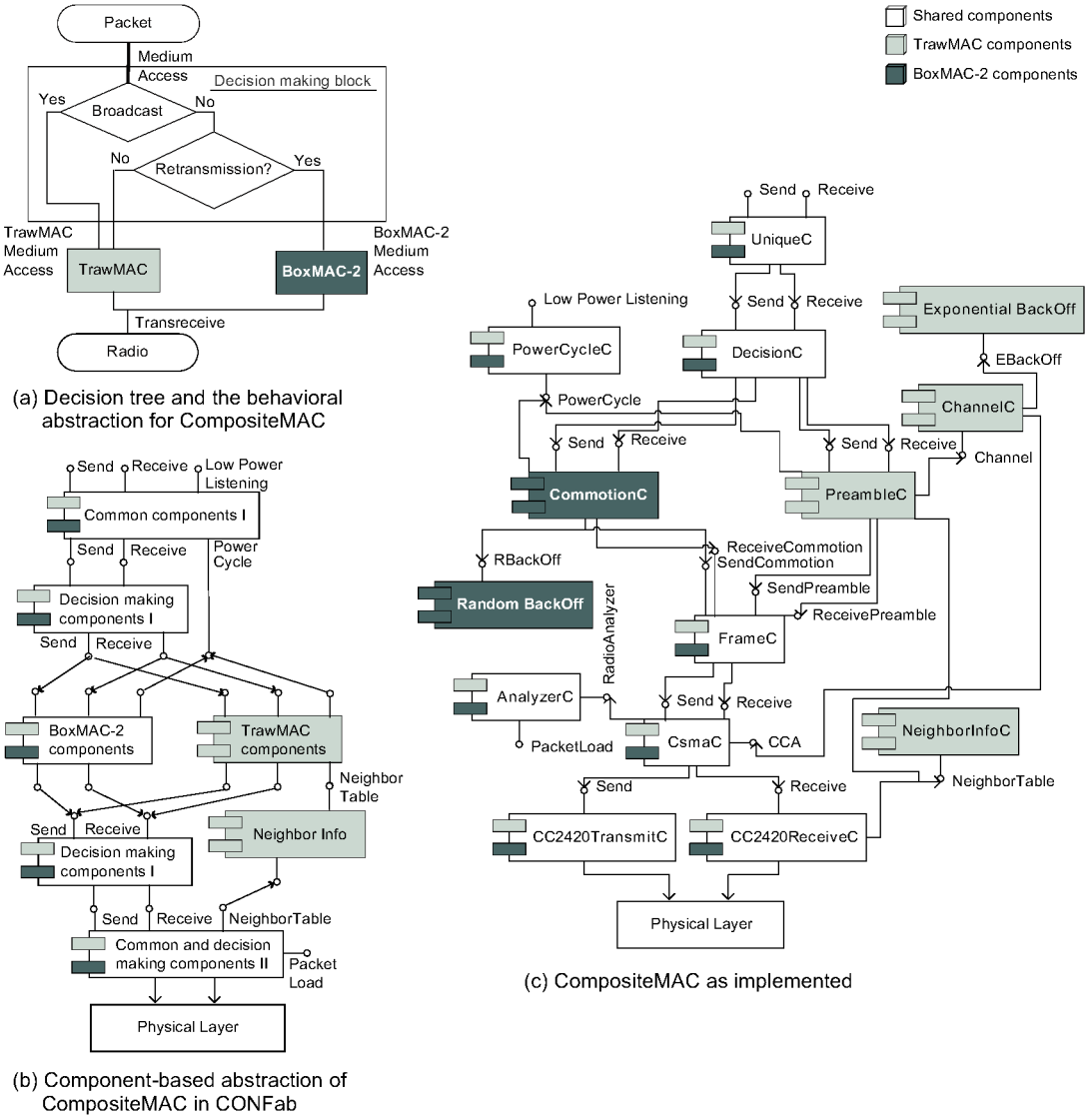
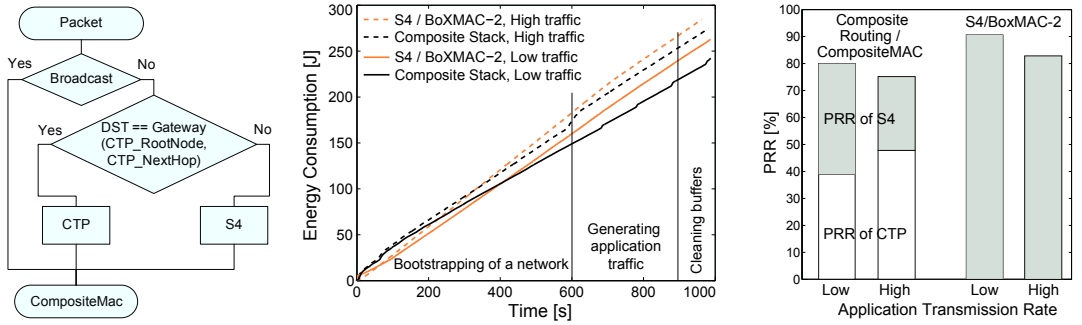


Figure 3.16: Components of CompositeMAC at various level of abstractions (subfigure (c) based on approach proposed in [216]).

Fab's purposes [215].

From the other side the abstraction in Figure 3.16a that shows the decision logic of switching between TrawMAC and BOXMAC-2 is appropriate for behavioral (functional) abstraction, but cannot be used for components representation, as it is simply not detailed enough, and does not reflect on inner organization of the software module. In Figure 3.16b we show the intermediate abstraction of the CompositeMAC protocol that we consider to be appropriate for CONFab (with the *Neighbor Info* module being shown as an example of an exposed component useful for the cross-layer reuse). This abstraction has small number of components that possess a limited number of clearly defined interfaces, which also maps well to the behavioral representation important for the first stage of the semi-automated protocol stack design. We used this representation granularity on define MAC components in the ontology.



(a) The decision tree for composite routing.

(b) Overall network energy consumption.

(c) Overall network PRR.

Figure 3.17: Performance of the protocol stack that includes composite routing and CompositeMAC as compared to the S4/BoxMAC-2 protocol stack for the scenario, where 50% of the traffic is directed to the gateway with the rest being addressed to random nodes.

The evaluation results for the CTP/CompositeMAC stack are provided in Figures 3.14 and 3.15. The stack performs very well both in the scenarios with homogeneous and periodically changing traffic. For these scenarios we see that the performance gains over the *best* of the MAC protocols are up to 15% for high traffic rates, see Figure 3.15b. For many of the evaluation cases considered here, the PRRs rates of the protocol stacks with three different MAC solutions are very similar, and, therefore, we do not separately show them. We only display in Figure 3.14d the situation of a heavy traffic load, which results in different PRRs for the three protocol stacks, with the CTP/CompositeMAC combination performing better than the CTP/TrawMAC, but slightly worse than the CTP/BoxMAC-2.

Similar reasoning was applied for defining granularity of components for routing. The resulting choices allowed us to easily consider a modified *Link Estimator* component for S4 that led to 50% of performance gains as explained in Section 3.4.2, and combined routing behaviors of S4 and CTP so that their implementation along with CompositeMAC a one protocol stack would fit a memory footprint of a TelosB node (see Section 3.4.3.2). We stress one more time that though the current ontology structure of CONFab is efficient it reflects only one possible decomposition and representation of a protocol stack. Generally this ontology-enabled knowledge base can be extended or modified to reflect and reason on almost any of the existing WSN alternative solutions.

3.4.3.2 Composite Routing

As part of the evaluation of component-based design and CONFab semi-autonomous reasoning we considered another scenario that leads to combining of the protocol functionalities, namely composite routing. In this scenario 50% of application packets are targeted to a gateway with the rest sent to other random WSN nodes. This scenario is inspired by the consideration for a network setup, where nodes besides providing sensory feedback, and also actively collaborate with each other, e.g., perform data aggregation or share a sensing schedule. As we saw from Figure 3.13b the performance of the S4 routing protocol due to lack of a packet buffer quickly degrades as network size grows. From the other side the CTP routing protocol cannot send packets to multiple hosts. Therefore, the indicated scenario forces CONFab to recommend a protocol stack that has the joint routing functionalities. The packets sent to the gateway are forwarded by CTP, otherwise S4

is used. The corresponding decision tree is shown in Figure 3.17a and the merge of the routing functionalities is done without re-design and re-implementation of the respective protocols. The performance of the resulting stack is shown in Figure 3.17 for a scenario with network size of 20 nodes and variable application rates. We see that for both low and high application rates the observed performance gains are not significant for such a small network, as the decreased power consumption is also combined with longer network setup times and lower packet delivery ratio, which is caused by the design flaw of S4 (absence of buffers). These results indicate that though certain opportunities for autonomous optimization arise through component recombination by CONFab, provided performance gains do not always justify increase in the complexity and the additional implementation efforts. Further, inefficient protocol stack configurations obtained due component reconfiguration can also significantly increase the optimization state space. This factor cannot be resolved by such systems as CONFab, but can be mitigated through aggressive shaping, exploration and reasoning on the respective state spaces. Additionally, one foresees extension of CONFab with functionalities allowing for the framework to derive statistics of components and stacks performance over a range of operation conditions that would enable a user to make informed decisions which components should be considered by the framework further on.

3.5 Conclusions

In this chapter, we have presented a framework, called CONFab, that employs ontology-enabled knowledge base for semi-automatic scenario-specific composition and configuration of network protocol stacks for wireless sensor networks. This optimization problem is solved based on user-defined goals, prior deployment experiences and expert inputs, which are conceptualized and related using the ontology. The ontology also captures the major behavioral and structural aspects of the protocol stack design, thus enabling high-level reasoning on the corresponding entities recorded in the knowledge base. CONFab operates both on already predefined protocol stacks, as well as their components such as protocols or smaller modules. The framework suggests viable “wirings” and configuration settings chosen from a pool of alternatives to generate a protocol stack that is expected to maximally fulfill the user requirements for a given deployment scenario. The core of CONFab operates using description logic and performs qualitative reasoning. The framework also allows for modifications of its knowledge base, and supports additional software plug-ins, which enables extensibility of the system. Our plug-ins allow the framework to learn from its experience, actively explore the optimization state space, employ utility-based reasoning, utilize composite scenario definitions, and compare and merge graph-based protocol stack representations.

As part of this work we have also considered the problem of effective decomposition of protocol stacks into components that provide clearly defined services, manageable interfaces, and are suitable to autonomous reasoning. We have identified common high-level MAC and routing abstractions that, we believe, allow for effective decomposition of protocol stack functionalities. These were used to eliminate redundancies in protocol stacks, leading to improved performance and reduced development overhead. We experimented with the corresponding software implementations of these abstractions, which indeed confirmed the efficiency of the proposed approach.

The CONFab framework has been extensively evaluated. Besides separate testing of the suggested plug-ins, extensive performance evaluation studies of 700+ hours have

been conducted on Indriya TelosB testbed for the core of the framework. First, we thoroughly studied the performance characteristics of the well-known MAC and routing protocols and their component-based realizations. Then we fedback part of this data to CONFab to get its recommendation for a range of scenarios, which have proven to be successful. Our results demonstrate that CONFab is capable of autonomous reasoning on functional protocol stack composition, and with appropriate plug-in it can automate the whole process of searching for a well performing stack in a given deployment scenario. We have achieved over 37% increases in network lifetimes using a composite component-based protocol stack obtained with CONFab as compared to the classical layered implementations without degrading the packet delivery ratio.

In future, we plan to explore and formalize the hierarchical decomposition of a networked system to determine several levels of component granularity and network node clustering that would improve interfacing and meta-data specification of these components. This work would further advance the research on network lifecycle automation, to which the CONFab framework contributes. For the same purpose we plan to extend our ontology to capture a networked system in more detail, preferably incorporating several alternative approaches to network conceptualization. This would truly enable application of the meta-optimization and lead to such grand visions as Knowledge Plane and Cognitive Wireless Networking.

4

State-space Exploration and Optimization with Metaheuristics

In this chapter we show that metaheuristics are a powerful tool for solving network optimization problems with unknown or irregular search state spaces[1]. We discuss on advanced metaheuristics, where information on the network is incorporated into the search to increase its effectiveness. We demonstrate that they can be especially useful for the overhead-constraint applications such as the online optimization. However, our results also indicate that for the traditional offline scenarios the investment of efforts in development of advanced versions of metaheuristics, though leading to the improved results, might be less efficient as compared to the efforts directed at more careful formulation of the problem (e.g., its modeling, decomposition, and simplification of the state space).

We consider network planning and cross-layer optimization problems, and experiment with the following metaheuristic techniques: a genetic algorithm, simulated annealing, particle swarm optimization, and a memetic algorithm. We show that these methods already provide acceptable results when directly applied "out-of-the-box", i.e., they can be used "as is" by the wireless researchers that have no prior experience in metaheuristics. We demonstrate that they are robust towards poorly defined search state spaces, especially regarding their size. Additionally, we comment on performance of the particular search methods with respect to the considered network topologies and utilities, explaining how the inner organization of metaheuristics influences the obtained results.

We study in detail how the simulated annealing (SA) mechanism can be used for both offline and online the cross-layer parameter-based optimization in small and medium-sized wireless sensor and Wi-Fi networks in absence of the relevant context information [43, 46, 53–55]. We enhance this method with simple probabilistic models that relate the observed performance and the expected impact of changes in protocol parameters. These models can be trained online or obtained from prior historical data. We also suggest modifications to the exploration-exploitation strategy of the SA that in the self-optimization scenarios allows limiting negative user experience during the mechanism's training. We evaluate the proposed approach in the Qualnet network simulator, as well as in real Wi-Fi and WSN testbeds [51]. We show that in the considered scenarios the proposed modifications to the simulated annealing enable faster convergence of the network configuration to its near-optimum values. The algorithm also enables the re-use of the trained models in similar networking scenarios. Our experiments strongly indicate that relatively basic modifications to simulated annealing can provide a considerable boost to the achieved quality of experience metric of the network.

[1] In this thesis when discussing metaheuristic search methods, we use interchangeably this full term, as well as the alternatives "metaheuristics", "metaheuristic methods, techniques or algorithms", and "metaheuristic searches".

4.1 Introduction

As discussed in Chapter 2, each optimization problem has its own state space. The structure of that state space coupled with the utility functional dictates which algorithms are applicable and efficient for the solution of this task. Two main purposes can be persuaded when analyzing the state space of the problem. First, we may want to methodologically study (sample) the state space in order to reduce dimensionality of the problem, and obtain valid models of its key features that would lead to considerable simplification the optimization task, solutions of which would still display acceptable utilities [217, 218]. Second, we might aim at finding the near-optimal solution over this state space without any, or only secondary, interest in its structure. In this chapter we are primarily concerned with this second task, and apply metaheuristics as one of the most general and unrestrictive state space search methods.

Metaheuristic techniques are perceived as useful tools for resolving the network optimization tasks, which are characterized by the either irregular search spaces with multiple local extrema, and/or have an unknown, poorly modelable structure. The latter can be perceived as the black- or gray-box types of problems, which cannot be modeled so as to result in a reliable solution by the standard optimization approaches. We found metaheuristics to be a valuable tool for obtaining the first understanding of the problem and its search state space, which is viable for future modeling. For example, in this chapter the probabilistic models trained as part of work of the SA indicated strong linear parameter-KPI dependencies. These were later found to be efficiently captured in form of the correlation coefficients, which led to the construction of robust models of influence of *dynamics* of parameter changes on KPIs [56] (see Chapter 5, Section 5.3). Metaheuristics are suitable for efficient exploration of large optimization state spaces, which cannot be efficiently decomposed and approximated (abstracted) using existing information on the system and its operational environment. These methods are applied to the tasks, where greedy or exhaustive exploration is not viable.

There exist a significant number of wireless networking problems, where metaheuristic techniques could be applied. The examples of such tasks are cellular network planning problems [219], run-time cross-layer wireless network optimization, especially its distributed variation [220], scenario- and application- specific network protocol stack design (see Chapter 3), and joint PHY/MAC online optimizations especially for OFDMA systems [221]. Large state space of wireless networking problems are especially challenging in their practical or empirical domain, as there many of the typical modeling assumptions do not hold [222, 223]. Further, in Section 4.2 we shortly survey the related work on applications of metaheuristic methods to the wireless network problems.

The metaheuristic techniques are not ideal. Though in practice they perform very well for a wide range of problems, these methods do not provide finite, as opposed to asymptotic, convergence boundaries. Therefore, in some cases, metaheuristics can both deliver poor solutions, and take a long time to converge. Multiple works [21, 224] point out that these methods might need careful tuning to perform the best for a given optimization problem. In our experience the tuning of the methods' parameters improved the obtained results, but also the unmodified versions of the studied metaheuristics mostly performed well. However, we support and provide the indicative results for the other common opinion, which states that these techniques are invaluable to the initial assessment of the problem state space.

The classical exploration strategies can be also enhanced with feature detection and dimensionality reduction mechanisms to improve their convergence rates and/or quality of achieved results. Often the corresponding dependencies can be successfully stored and exploited in form of probabilistic models. Probabilistic models are often thought to be particular suitable for wireless environments due to their stochastic nature. These models can be either adapted from the similar already solved problems or learned online during the exploration process. We study the potential of such an approach for online and offline parameter-based cross-layer optimization. We show that advanced metaheuristics, in our case the modified versions of simulated annealing, that actively employ network models and prior data are indeed useful tools for improving efficiency of the search. However, these methods pay a price of losing the generality typical for the classical search techniques. We consider such fast converging metaheuristics to be viable candidates for the online applications, i.e., these methods can be applied as part of self-optimization especially if the context information required for utilization of the classical resource management optimization techniques is not available.

In our opinion, a combination of the network planning and cross-layer optimization problems provides an interesting case study on the applicability of both the basic and the advanced versions of metaheuristics for network optimization. These problems allow demonstrating the behavior of metaheuristics in a number of important situations encountered when designing and managing networks. The network planning problem, discussed in Section 4.3, allows investigating the scalability and the robustness of these methods. The case for online cross-layer optimization, studied un Section 4.4, allows investigating the convergence rate of the method is more important that the optimality of the result. Moreover the online and offline considerations of this scenario allows studying how easy it is to modify the search to be problem specific, and how efficient such a solution can be. Before proceeding further to exploration of metaheuristics in application to these two optimization problems we provide a short overview of the studied search methods and their application to wireless networking.

4.2 Survey of Applications of Metaheuristics in Wireless Networking

Heuristics and *metaheuristics* play an important role in the field of Artificial Intelligence (AI) [21, 225]. These methods have also found a wide application in wireless networking. For example, they are discussed in the numerous surveys of application of the AI and machine learning methods for wireless networks [226–231]. In this section we briefly introduce and further survey applications of these families of techniques, obviously concentrating on the metaheuristic methods.

Heuristics can be seen as theoretically unverified/unproven metrics or methods, basically “rules of thumb”, that in most cases can, but do not guarantee to, rapidly find a near-optimal solution to a problem. There exist a large range of heuristic searches for exploration, or finding of a near-optimal solution in state spaces, where application of classical, e.g., convex optimization techniques is not possible, and utilization of the exhaustive search is not desirable or feasible [21, 225, 229]. Compared to classical optimization algorithms, heuristic methods do not impose specific requirements of the relief of the search, e.g., ask for its convexity. They can be applied on problems that are characterized by both continuous and discrete search space structures. Many of the heuristic methods, especially the ones of the iterative nature, can avoid local minima and, thus, search in highly irregular reliefs. Generally well-formulated heuristics yield useful ap-

proximate solutions to NP-complete problems, such as graph coloring with the DSATUR heuristic [232]. Properly formulated, they may display hard boundaries on the convergence time, though still they do not guarantee optimality of the result until the problem follows well-defined structural conditions. In cognitive radio networking and generally wireless networking different search mechanisms based on heuristics, e.g., variants of greedy searches [233], have been widely applied [231, 234]. Often machine-learning algorithms are utilized to find good heuristics for the search algorithms [235–237].

In general *meta*heuristics (as opposed to plain heuristics) can be defined as iterative high-level processes that guide subordinate heuristics to efficiently converge to high-quality solutions. Whereas heuristics are typically closely tied to a particular problem at hand, and cannot readily be generalized beyond their domain of applicability, metaheuristics provide general computational prescriptions that can be applied to a wide variety of optimization problems. Similar to heuristics, metaheuristics are approximate and usually non-deterministic methods that provide little or no guarantee of convergence to the optimal solution, or even a good enough local optima. However, as said, metaheuristic search methods usually yield in practice excellent results and have accordingly become popular in solving hard optimization problems that cannot be solved otherwise. Well-known examples of successful application of metaheuristics to non-networking problems are the embedded system design [13, 20], and the printer circuit board automatic component allocation [238].

As with many other AI methods heuristics and metaheuristics may evolve online to improve results of the search [239–241]. Often such approaches are named as *advanced* or *hybrid* methods. The same terminology is used to define other modifications of the classical search techniques. For example, advanced metaheuristics can also make use of domain-specific knowledge in the form of previous search experiences (embodied in some form of memory) to guide the search. Advanced methods may also include components of other metaheuristics, or exchange information with each other. They are often used as parts of hybrid solutions to improve the search in the particular area of interest [242]. For example, a combination of particle swarm optimization and simulated annealing is used improve network coverage as part of the dynamic network planning problem in [243]. The term “hybrid” is also employed when metaheuristics and machine-learning methods are combined together. Metaheuristics and machine learning share a lot in common [244]. Machine-learning models incorporated inside a metaheuristic search can sometimes be extracted and re-used to direct another optimization search. Similarly, good heuristics can significantly improve the convergence rate and quality of models produced by machine learning [245, 246]. For example, Pendharkar used genetic algorithms to train neural networks that can predict customer churn in cellular systems [247]. Similarly, hybrid metaheuristics to train neural networks for channel prediction in application to the MIMO scenarios have been proposed as well [248]. A large part of work presented in this chapter deals with advanced or hybrid metaheuristics. Further, we briefly describe several popular metaheuristic methods, majority of which are used as part of the comparative “out-of-box” evaluation in Section 4.3. In Table 4.1 some applications of these methods in wireless networking, including sensor, cognitive and ad-hoc domains, with sample references are given.

Genetic algorithms (GA) [275] are, probably, the most well known representatives of the family of the *evolutionary* methods, which mimic the process of natural evolution. A genetic algorithm typically operates on parameter-based representation of the optimization state space with imposed additional constraints. It relies on strings of chro-

Table 4.1: Popular metaheuristic methods and their applications.

Metaheuristics	Applications and references				
	Network planning	Routing and localization	Cross-layer optimization	PHY, online multicarrier transceiver configuration	DSA and spectrum management
Evolutionary methods	[249–253]	[249, 250]	[254, 255]	[256, 257]	[258]
Swarm intelligence	[259–262]	[262–267]	[268]	[269]	[258]
Local neighborhood search	[270–272]	[259, 273]	[46]	[273]	[274]

mosomes with each of them encoding one candidate solution to the problem, so called individual. Individuals form a population on which operations inspired by biological processes are conducted. These are inheritance, selection, crossover, and mutation. Each individual is assessed according to its fitness function, i.e., a utility functional. Bases this assessment the individual as a whole (elite individual) or its chromosomes participate into formation of a new population (generation). This process of evaluation of individuals and their evolutionary-inspired adaptation for a new population is repeated multiple times, which typically leads to gradual improvement of the solution quality. One of the popular criteria for algorithm termination is the absence of significant performance improvement of new solutions as compared to the one already found for a preset number of generations. Another common criterion is a maximum number of generations reached. Genetic algorithms are very flexible. They have a number of tunable parameters, as well as possibilities for adaptation of its basic mechanisms, e.g., providing possibilities for incorporation of machine learning models. Wide applicability and good performance of genetic algorithms led to the development of genetic programming, where tree-like representations of the state-space are explored, which gives possibility of the search space structuring in more program-like (if-then) manner. Graph form state representations are used in evolutionary programming. A mixture of both linear chromosomes and trees is explored in gene expression programming.

Genetic algorithms and other evolutionary computing methods have been widely applied in the domain of cognitive wireless networking and generally wireless networking. The survey by Kampstra et al. provides a review on applications of evolutionary techniques to the field of telecommunications in general, structuring it after the application problems, which range from network planning to dimensioning, routing, and dynamic spectrum access/frequency assignment [249]. *Memetic* methods are a family of hybrid metaheuristics that use genetic or other evolutionary methods for the global search and another technique for the local utility improvement [276]. The stages of global and local searches alternate. Memetic algorithms have seen applications to wireless sensor networking [252, 253], as well as cognitive radios [257]. Additional sample applications of genetic and memetic algorithms can be found in Table 4.1.

Particle swarm optimization (PSO) is a popular metaheuristic search technique that belongs to the class of *swarm intelligence* methods [277]. The algorithm mimics social behavior of large gatherings of animals or birds, where behavior of an individual (a po-

tential solution, a particle) is influenced by other individuals already residing at better positions. The PSO, similarly to genetic algorithms, relies on multiple copies of potential solutions, i.e., it forms a population to avoid being stuck in local minima. However, instead of relying on the crossover or the mutation operations, the PSO particles (individual potential solutions) operate on their own best known positions in the search space, as well as on the the entire swarm's best known position in order to probabilistically decide on the "speed" and "direction" of their further "movement" in the state space. This iterative search method has been a successfully applied to many research problems. However, as for many metaheuristics, its performance is dependent of the initial choice of parameters. Like for the other techniques discussed in this section, there also exist multiple variations of this basic metaheuristic.

Ant colony optimization is another metaheuristic belonging to the class of swarm intelligence methods [278]. It is inspired by the behavior of ants in finding their paths from the colony to the food sources. The algorithm avoids getting stuck in local minima by using multiple ants (or agents) to traverse the solution space and find locally productive areas. Ant colony optimization is particularly useful in solving tasks where no global or up-to-date perspective can be obtained, and therefore the other, in general more effective methods cannot be applied. It outperforms simulated annealing, tabu search and genetic algorithms in dynamic environments, as it can adapt continuously to the changes in real time. The swarm intelligence methods have been especially popular in application to wireless sensor networking [262] and ad-hoc routing [264–266].

Simulated annealing (SA) [279] is one of the oldest probabilistic metaheuristics that can be loosely described as a process of heating and controlled cooling process of a metal, i.e. annealing, which aims to improve its quality by affecting its microstructure. This mechanism belongs to the class of local neighborhood searches [280]. Simulated annealing can be viewed as a subset of genetic algorithms with only one generation present, which undergoes continuous mutations. However, in contrary to the genetic algorithm the simulated annealing introduces the concepts of temperature and reheating. The temperature controls the processes of adaptation of configuration parameters ("genes", "electrons in an atom") that define the optimization state space of a problem. The higher is the temperature the more actively gene values are changing, and the "further" in terms of considered distance metrics from each other they can get. When the temperature drops, as a result of the cooling process, very few of the parameters can change, and these changes would not be any more so significant. Therefore, when temperature is high the optimization state space is searched aggressively, and when it drops primarily the local neighborhood of the best solution obtained so far is searched. If the temperature decreases below a certain threshold the search stops. The mechanism probabilistically accepts newly found solutions, which allows the system to move consistently toward lower energy states, yet still jump out of local minima. The probability of acceptance in the classical version of the simulated annealing depends on both the current temperature value, as well as the degree of change between the old and new utility values. The algorithm also foresees the mechanisms of reheating, where the temperature can be raised again, which also allows escaping local minima. Simulated annealing allows the flexible definition of the function of temperature cooling that is typically dependent on the number of search iterations or the performance improvement. Another specific of simulated annealing is that this mechanism makes decisions based only on a single sample of the explored state space, which can be especially useful for situations when long exploration phases may have significant negative performance impacts, e.g., in cases of run-time optimization

with active users. As in the rest of the chapter we extensively experiment with this metaheuristic, we provide the outline of this algorithm in Appendix C, which also illustrates the simplicity of this technique.

Tabu search [281] is another example of the local search metaheuristics. This algorithm uses memory structures, tabu-lists, forbidding the use of certain values in the search. Typically formation of the tabu-lists is based on the search history. One example of a prohibitive rule is a tabu imposed on visiting the parts of the state spaces that were traversed very recently. Tabu lists containing the prohibited values are very effective, though a very good solution that just happens to be forbidden might be missed. To overcome this problem aspiration criteria are often introduced. They allow to override the tabu state of a certain solution and include it in the allowed set. Tabu search algorithms vary primarily by the way in which the tabu and the aspiration criteria are defined.

Simulated annealing has been successfully applied to similar problems as other metaheuristics. These include parameter-based cross-layer optimization [46], power control [282], localization [273], and network/radio resource planning [271, 272]. Tabu search was used, for example, to perform clustering in wireless sensor networks [283], or solve WSN network planning tasks [260, 270]. For all these problem domains repeatedly the questions are raised concerning the randomness of the solution, i.e., how similar would be the delivered results after running the metaheuristic multiple times, what is the desirable number of search iterations, and if there are other less heuristic alternatives exist.

The family of metaheuristic searches is a very powerful tool that can be applied to a very wide range of problems. They are especially popular for all forms of the network planning problems and the corresponding parameter optimizations (e.g., location, power, frequency, association). The strength of these methods is that they are likely to find a good result in sufficiently high number of iterations even when nothing about the problem, and the corresponding utility behavior on its search space is known. However, when the structure of the problem gets better understood these become not the most optimal tools. The discovery of the structure of the problem often allows to alter the optimization process, e.g., to effectively decompose it or shrink the search state space. This leads to the increase in the convergence rate without significant degradation of the quality of the solution provided. Modeling of the problem also allows for simpler heuristics to be applied. Furthermore, with sufficient approximations the classical optimization techniques can be used [284], which leads to provably optimal optimization decisions (subject to the accuracy of the approximations employed, of course) that can be obtained very fast. Alternatively, machine-learning methods can be applied to derive robust reflex optimization algorithms, which are especially useful for the online optimization.

4.3 Exploring Metaheuristics on Example of Network Planning

In this section we aim to demonstrate that out-of-the-box applications of metaheuristic techniques indeed result in close-to-optimum solutions, sometimes, at a price of long convergence times. For our study we consider a simplified network planning problem. We experiment with the genetic algorithm (GA), the participle swarm optimization (PSO), the memetic algorithm (MEM), and the simulated annealing (SA). For the first three methods we use existing realizations in R packages `rgenoud` [285], `hydroPSO` [286] and `Rmalchains` [287], respectively. Simulated annealing is realized as described in Appendix C. Our comparison aims at providing additional evidences for applicability of these search methods for solution of network optimization tasks. We also comment of

the likely cause of the difference in the performance of these methods depending on the particular optimization task formulation. Further we shortly introduce the formulation the network planning optimization task, describe the considered scenarios, and provide the basic settings of the metaheuristic techniques. After that we present and discuss the results of our work.

4.3.1 Scenario and Metaheuristcs Settings

In the network planning scenario we distribute a population of users over a certain area that continuously download data utilizing all available throughput, i.e., they saturate their links. The goal of the optimization task is to determine the number, the location, and the power settings of femto base stations (or access points) that would maximize the utility of a network. All nodes are sharing the same operational frequency range. We employ the Xia-Bertoni propagation model [288], and consider a number of alternative utility functions, user node distributions, and two network performance metrics to demonstrate how metaheuristics perform on quite different optimization state spaces.

Network performance is established using either Shannon capacity limit, or LTE-like mapping of SINR values to MAC throughput using the coding and datarate as described in [289]. Throughput of users is combined according one of the three utility functions. In the first we obtain the mean network performance. The second utility considerers performance received by the 5% of the users that experience the worst quality of service. This utility is similar to the max-min fairness, but in our case we consider not the worst user rather the worst N users to avoid the situation, when one poorly placed node severely effects the optimization of the whole network. The third utility is a weighted sum of the first two utilities with 99% of the contribution being from the poorly performing nodes. For assessment of performance results we use utility values normalized over the maximum obtained during all the evaluations runs in a given scenario.

We consider three different user location distributions after the Poisson and the SSI (Simple Sequential Inhibition) point processes. There are approximately 50–450 users allocated on the areas ranging from $50 \times 50\,\mathrm{m}^2$ to $1000 \times 1000\,\mathrm{m}^2$. Each metaheuristic runs for 1,000,000 evaluations. The population size for the genetic, the particle swarm optimization, and the memetic algorithms are 300 genes/particles. The simulated annealing experiences reheating every 50000 iterations. By default all the algorithms operate on real numbers. For the genetic algorithm we also experiment with the version defined on integers. For the PSO, the GA and the MEM algorithms we use the default parameter setting provided by the R packages. For the SA the significant parameter of the exploration rate of the search dimension at the maximum temperature is set to follow the normal distribution with zero mean and the standard deviation being 8% from the maximum coordinate setting and 15% from the maximum power setting.

The array of optimized variables that define the dimensionality of the optimization state space, also called as a gene or a particle for specific metaheuristics, is organized as follows. First, we choose the maximum number of femtocell (alternatively refereed to as access points (APs)) to be placed. This number N can vary between 5 to 40. Each access point is defined by three variables, which are its X and Y coordinates, and the power settings. Therefore, the size of the array, the number of state space dimensions, is $3 \cdot N$. The gene sequence is orders by as follows. First, we place X coordinates for all the stations, followed by all the Y coordinated, and N power variables defining transmit power of each of the station. The transmit power can vary from -120 dBm to 23 dBm,

where the number of the transmission power levels a femtocell can choose from is limited to 2, 7, 10, or 48. The lowest power level of -120 dBm indicates that the femto station is off, it does not have clients, and, therefore, does not need to be included in the final solution. The other transmission powers vary from 0 dBm to 23 dBm.

By altering of the number of stations N, and the granularity of the power settings, we adjust the size of the state space to be searched. In the smallest case when we are considering only 5 APs with two, on or off, transmission power levels deployed over the area of 2500 m^2 with one meter placement granularity, we are already facing the state space of size $3.125 \cdot 10^{10}$. Clearly, the size of the state space influences the quality of the solution found and the convergence time of metaheuristics. A metaheuristic search would converge fast on a small state space, but this state space just might not include the optimal solution. On the other hand on a very large state space the optimal or near-optimal solution might not be found. In the next subsection, along with other results, we illustrate this *tradeoff between the size and granularity of the state space*.

4.3.2 Evaluation Results

We continue with the comparison of results obtained when running alternative out-of-the-box implementations of metaheuristic searches. Figure 4.1 shows the graphical notation as well as lists the different scenario settings for which the simulations have been conducted. Figure 4.2 gives a representative example of the evaluation results for the different metaheuristic optimizers and for the different state space sizes. Due to the large number and complexity of the scenarios it is naturally difficult to draw conclusions that would hold for all of the planning problems considered. Nevertheless, several important trends can be identified, and a number of individual observations made on how the different metaheuristics, state space choices, and utility function selections interact. As a general conclusions, the genetic algorithm with both integer or floating point based operations provide the most stability. Over all scenario permutations it shows on average the best performance. The memetic algorithm performs typically well on small state spaces with the uniform distribution of clients, especially if the Shannon mean capacity utility is used. This is clearly because the problem structure for these scenarios is more smooth locally, which can be exploited by the local search component of the memetic algorithm. The particle swarm optimization performs stably on the smaller state spaces, and tends to degrade with the increase of dimensionality. We believe that this is due the suboptimal version of this algorithm for the given problem (called SPSO [290]) and the utilized rate of the mixture of the local and the global particle information. The same figures, as well as the overall results, show that the simulated annealing on average shows good performance, especially on the state spaces with small number of power levels. It often converges very fast and performs similar to the other more complex metaheuristics. We believe that though this algorithm is less sophisticated then the others and cannot benefit from joint particle/gene intelligence, it gains an advantage from its simplicity. Its rather transparent structure allowed us to quickly implement it, and tune its few parameters (number of iterations per reheating and the state space exploration rate) to suit the given optimization problem. It is also the most memory efficient method.

We also see that the convergence rates are significantly impacted by the choice of the utility function involved, as well as the performance metrics/models used (see Figures 4.2). The mean Shannon capacity metric results typically in slower convergence except in the case of the memetic algorithm due to the reasons discussed above. This is

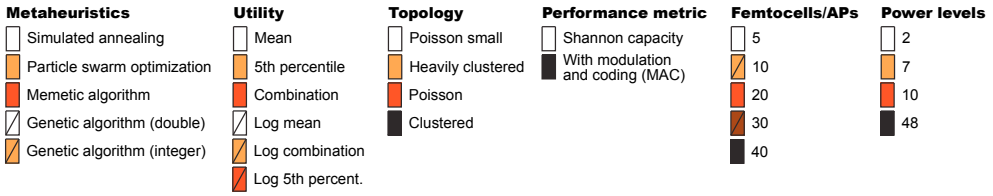


Figure 4.1: Legend of the experiments performed and their major characteristics.

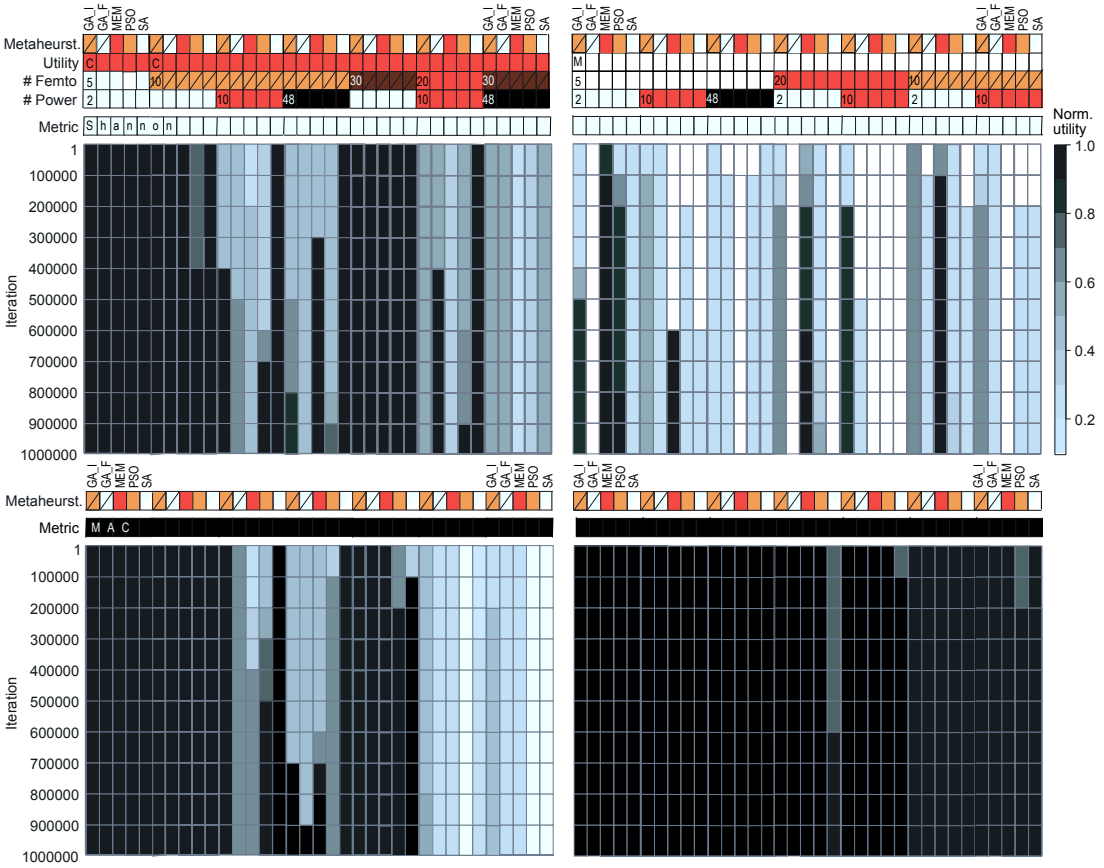


Figure 4.2: One of the extreme examples, showing a small Poisson population deployment. The network planning is carried out using the Shannon capacity based utility metrics.

probably due to more irregular relief of the Shannon based search state space as compared to the one induced by the MAC performance metric. The latter, due to the introduction of modulation and coding effects on the SINR values, discretizes the utility function values, and also makes the marginal utilities non-concave. This leads to less variability in the search relief, thus leading to increased convergence rates of all studied metaheuristics. This effect is especially prominent for the mean utility function.

The differences in arising problem structure between the cases of mean capacity, the 5th percentile, and the combined utility are illustrated in Figure 4.3. Figures 4.3a-e show the utilities for the evaluations performed by the genetic algorithm on the three different utilities using the Shannon and the MAC performance models for the heavily

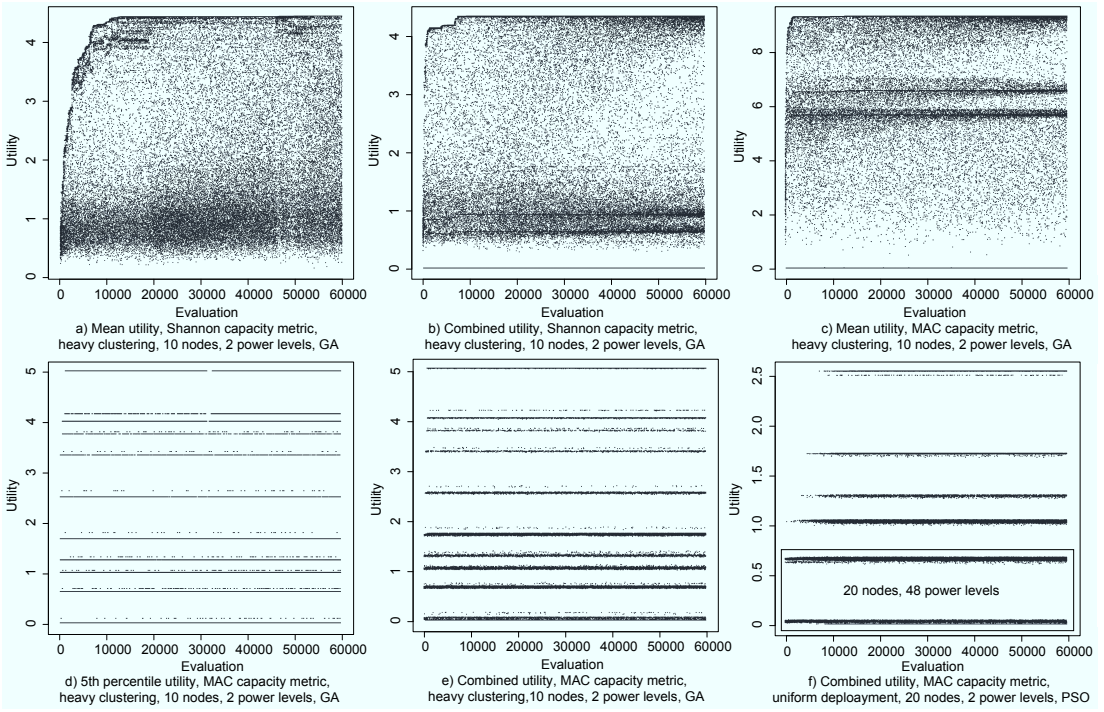


Figure 4.3: Sample influence of the utility, the performance model (metric), and the node deployment on the optimization process. Utilities are normalized to a comparable range.

clustered node deployment. We observe a strong difference between the mean and the 5th percentile utilities for the MAC model. In the first case the evaluation results are spread over the whole landscape with only very limited amount of structure being present (additional two lines besides the maximum). For the latter utility all evaluation results are located in several very narrow regions, which on the Figure almost look like lines. The combined utility results in the search with evaluations appear as diffused lines, due to the heavy component of the 5th percentile. Figures 4.3a-b illustrating the impact of utilities on the Shannon model indicate that in this case the contribution of the state space structure from the model is stronger than from the utilities. Figures 4.3e,f also illustrate how the node deployment model, as well as the size of the state space affect the evaluation process. If we compare the number of “lines” (clusters) between those two figures we observe that in the case of uniform node deployment there are fewer clusters with the one corresponding to the best performance being at larger distance from the rest, indicating that this problem is more difficult to search and additional adjustment of the algorithms is required. We made a conclusion on the influence of the number of clusters on the results obtained by the metaheuristics based on other observations made for the tasks with the same utility/performance model assumptions. The above conclusions also hold for other simulated scenarios for which illustrations are not given. As Figure 4.3f shows with increasing of the state space, i.e. increasing complexity of the problem, the number of clusters decreases. The discovery of the structure of the state space of an optimization problem can be utilized to adjust the behavior of metaheuristics in terms of exploratory jumps, i.e., parameters of the local and global searches.

In Figure 4.4 a comparison of the performance of particle swarm optimization and

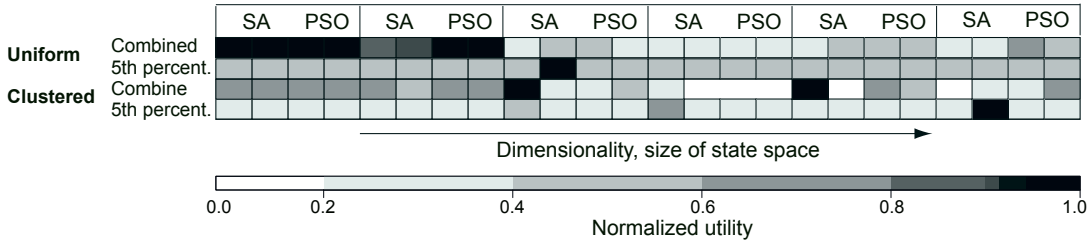


Figure 4.4: Performance of particle swarm optimization and simulated annealing algorithms over the logarithmic form of utilities over the uniform and the clustered user node deployments for the modulation/coding performance metric. The 5th percentile and the combined utilities are compared at 1,000,000 evaluations. The maximal absolute utility values that are guaranteed for each user for the combinations presented in the table are {14.2, 7, 19,14} Mbps for {combined/uniform, 5th/uniform, combined/clustered, 5th/clustered} utility/deployment scenarios. Results from two sample runs of the algorithms are shown to illustrate the possible diversity of the performance.

simulated annealing algorithms over the logarithmic form of utilities over the uniform and the clustered user node deployments for the modulation/coding performance metric is shown. From the results it is clear that just having the 5th percentile as the utility is very challenging for all the optimizers. The combined utility function enables the search to converge to significantly higher levels of the utility more often especially at smaller state space dimensionality. Moreover, the absolute performance results if we consider the five percent of users obtaining the worst service are mostly higher, sometimes as much as twice, for the combined utility as opposed to the 5th percentile metric. This difference is again induced by the relief of the search state space. The 5th percentile utility flattens the relief making it difficult for the metaheuristics to advance. The combined utility while having a strong 5th percentile component does not flatten the search state space by incorporating a contribution from the mean utility. The metaheuristics can regularly advance, which leads to ultimately better results.

Most metaheuristics converge within the prescribed maximum number of iterations on appropriate utility function coupled with the right performance metric and the search state space dimensionality. In the case of careful problem formulation many methods converge even in time of few hundreds of thousands of iterations (see Figures 4.2 and 4.5). However, even on the extremely suboptimal formulations (e.g., in terms of large state spaces) metaheuristics provided the acceptable results, though at the price of multiple re-runs. Generally we observed that that it might be often advantageous to run multiple problem instances either sequentially or in parallel for shorter times, rather than individual runs over very long time spans. This is clearly due to the probabilistic nature of the metaheuristics involved.

Figure 4.5 illustrates the structure of the solutions obtained from the metaheuristics for different optimization state spaces, employed performance metrics, and utilities. One can observe the clear changes between the initial stages of the search and the ultimate results obtained. Overall, for the scenario of $1\,\mathrm{km}^2$ heavily clustered node placements, the deployments are remarkably similar for the different variations of the utility functions used. This indicated that the structure of the state space is heavily influenced by the node deployment structure, which suppresses the influence of other optimization task

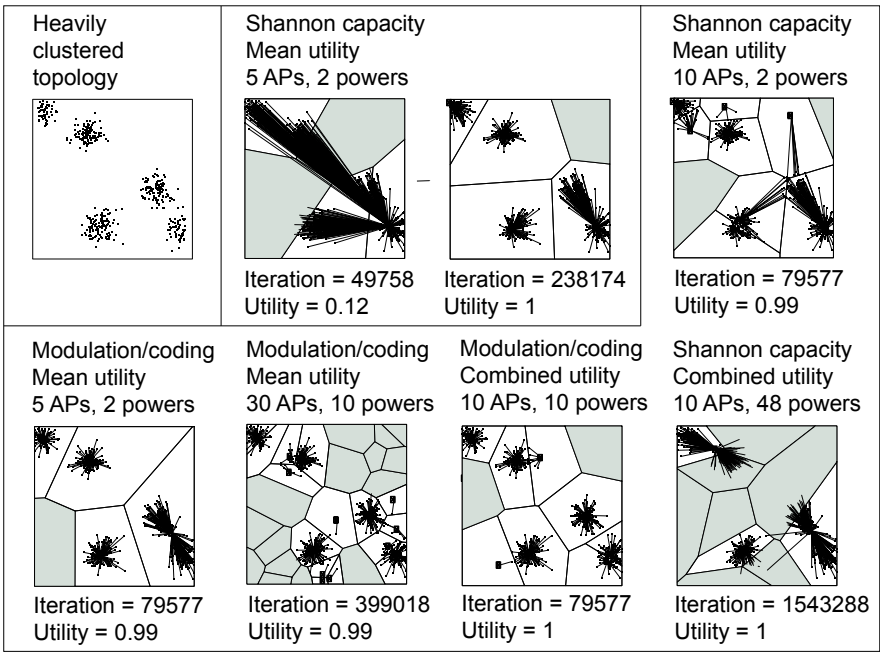


Figure 4.5: Example deployment / power allocation solutions found over the range of the optimization state spaces, employed performance metrics, and utilities. The sample results obtained by different (PSO, SA, and float-based GA) metaheuristics are displayed.

factors, such the utility formulation. In particular, the results obtained using the simple assumption of the Shannon capacity are quite similar to the more complex case of realistic modulation and coding effects being included. We also see that results for the very large state space (with large number of power levels available combined with large AP counts) are similar to the ones with smaller state spaces. Therefore, if a certain scenario is known to have a strongly featured search relief then both simpler utility functions, as well as underlying system models (that influence the performance metrics used) can be employed. Overall such scenarios tolerate more “sloppy” optimization task formulation.

Observing the results of Figure 4.5 in more details, we do notice subtle differences. If we consider the secondary goal of minimization of the number of deployed APs to lower the fixed costs then the deployment on the low-right panel with only 3 APs used is the best. However, this requires transmission power adjustment. This functionality may also incur additional costs. Otherwise with the default power level four nodes need to be deployed. Thus Figure 4.5 illustrates a typical question of the justification of additional technology investment (flexible power settings) considering the potential savings. The obtained results also return us to the discussion in Chapter 2 on appropriate optimization task formulation, which, e.g., takes into the account the goals of economical feasibility, which might get translated in the restriction on the available solution techniques. In general we have found that it is highly beneficial to use small amount of initial exploration of the state space for basic metaheuristics parameter tuning, as well as to sometime reformulate the original task (e.g., introduce the combined utility in our case) to improve the convergence. These results highlight further the interrelationship discussed in Chapter 2 between the problem formulation in terms of utilities, models, and the optimizer used.

4.4 Parameter-based Cross-layer Optimization and Hybrid Simulated Annealing with Probabilistic Models

In this section we discuss the problem of deriving an efficient advanced metaheuristic method for a specific networking problem [43, 46, 54, 55]. We consider the parameter-based cross-layer optimization task that can be solved both offline and at runtime. The online cross-layer optimization contributes the network self-optimization process. For metaheuristics we study simulated annealing as it is quite a simple method and, therefore, its modifications are easy to follow. We aim to realize such a hybrid version of the chosen metaheuristic search that converges to the sufficiently good solution significantly faster than the classical version of this algorithm. We explore how the exploitation of probabilistic models as part of the metaheuristic can help achieving the stated goal. For models we consider probabilistic relations between the utility values, which are a function of observed KPIs and the parameter configuration of multiple protocols across the stack. We explore models of various complexity ranging from considerations for empirical probabilities of increasing the utility, to parameter-utility correlation coefficients, and Bayesian graphical models. We also consider the influence of more generic and simple modifications to the simulated annealing, such as the dynamic adjustment of the reheating behavior, or the consideration for the ideally desirable utility values as part of the cooling schedule.

We test the variants of the considered metaheuristics in small ad-hoc, and multi-access scenarios setup in Qualnet network simulator [291]. We also deploy and compare these algorithms in the real Wi-Fi and WSN testbeds, and study the behavior of the versions of the algorithm on the empirical data both during the offline and the online optimization. Further in this section we shortly describe the system model and the assumption taken, as well as the considered scenarios. We recap on the simulated annealing algorithm and point out the places of its possible modification. We then discuss the considered probability models. Finally, we evaluate the resulting versions of advanced simulated annealing on the specific scenarios, considering both the effects of introduction of the probabilistic models, as well as simple adjustments to the basic algorithm.

4.4.1 Properties of the Considered Cross-layer Optimization Task

The cross-layer optimization task can be seen as a autonomic feedback control loop [148], with a metaheuristic optimizer acting as an intelligent agent [21, 46]. The general case of the cross-layer optimization is illustrated in Figure 4.6. The sensory inputs to the system are measured KPIs, or attributes, that serve as variables of a utility function that is used to estimate performance of a network. The actuators are parameters of protocols composing a network stack. Therefore, we deal with the utility maximization task, with the search state space of this optimization problem being a multi-dimensional landscape where each dimension corresponds to one adjustable protocol parameter. Each parameter may take a fixed number of predefined states or values.

We assume that there is no prior knowledge on the influence of parameter settings to attribute values. In other words network context information, such as topology or interference conditions, are unknown. Therefore, the model of utility-KPI-parameter relations has to be *learned* as part of the execution of the simulated annealing based optimization. In this work we do not learn the structure of the model, but rather the *probabilistic* relations between the variables of the model. We also explore the impact of exploitation of

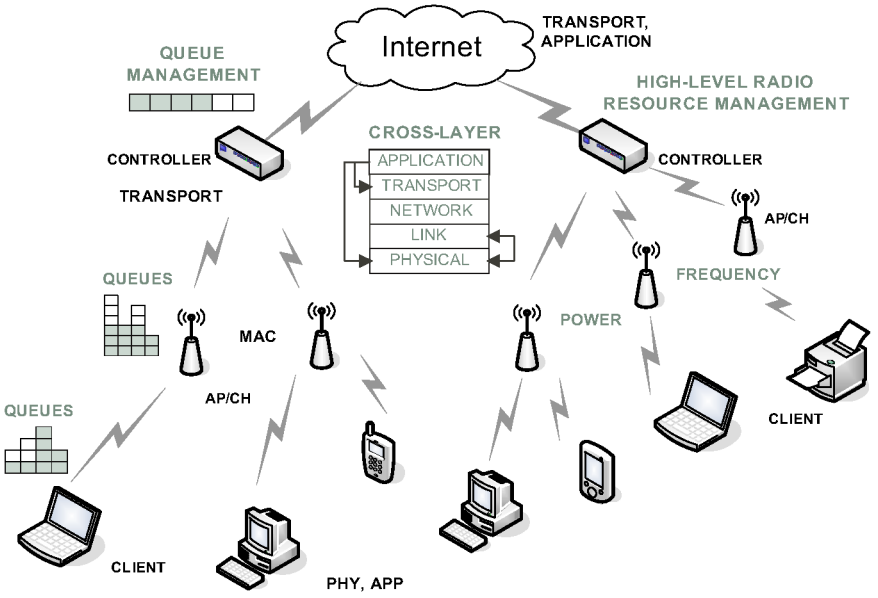


Figure 4.6: Overview of the general cross-layer optimization scenario for TCP/IP based wireless networks.

the pre-trained models in the hybrid simulated annealing methods, but this is a complementary, not the major, part of the work.

Furthermore, for simplicity we take a centralized view on the problem and concentrate only on "vertical" cross-layer optimization [40]. In other words, all inputs to the nodes are known to the decision making entity, and there is a centralized control over all nodes in a network. All the nodes share a single utility function, which all the participants contribute to and strive to maximize. Moreover, unless stated otherwise, we assume that all relevant nodes use the same parameter values. In future our work can be easily extended to include "horizontal" or inter-node optimization as well, but then obtaining of the exhaustive state space evaluation as the ultimate solution comparison point would be unrealistic. We also simplify the optimization problem by considering only simultaneous changes to protocol parameters, though, obviously, in reality these adjustments can be done at different rates. For example, switching between different access points or networks is likely to take longer than the change of a single MAC parameter. Such considerations will lead to more effective run-time optimization than the results shown below, and can be employed for the future extension of this work. We believe that in spite of the heavy simplifying assumptions discussed above our formulation of the cross-layer optimization problem is still relevant, and can serve as a viable case study for experimentation with metaheuristics.

The task of network optimization from a protocol stack configuration perspective carries the following *specifics*. First, there typically exists a *definite starting point of search* – the *default settings* of the parameters of the protocols that form the stack. This state in the search space usually already provides the above average utility in standard operational conditions. However, the default settings can lead to low utilities in cases of abnormal network behavior, which could be caused by a change in network topology, drastic increases in network load, or interference. The activation of the metaheuristic

search can lead to learning of such protocol parameter settings that result in significant performance increase. However, this process also carries a danger of decreasing system performance during the exploration phase without providing any guarantees that the utility gets improved in a reasonable time frame. This discussion is especially relevant if such an optimization is executed online with active users being present in a network.

Second, a self-optimization process to which the cross-layer optimization belongs has *two search phases*: *static and runtime*. The *static* configuration stage can be regarded as a result of long-term learning, i.e., the optimizer has a long history of network performance, it can observe and experiment with network settings for a long time without the need to keep high average utility during the training period. This stage can take place at times of low network load, e.g., during night times or at the stages of pre-deployment network testing. The static optimization allows obtaining the best network settings for a selected range of operational conditions at a cost of the system not being fully available for the exploitation. Such an approach may miss studying some traffic patterns and/or operational conditions that would occur when the system gets exploited for real, which might also lead to suboptimal network configuration.

The *runtime* optimization has more constraints. Here the average utility values must be kept high for user satisfaction. This requirement, basically, results in short exploration phases for the optimizer. For the same reason the system must react fast to changing network conditions (utility drops), which again leads to short periods of learning (exploration, reheating). Additionally, a network might encounter better operational conditions, for which a certain change in protocol settings would lead to significant utility increase. Such opportunities should be recognized by the optimization algorithm.

Ideally, a network optimization process incorporates both of the offline and online search phases. First a static training is conducted, for example, to limit the number of possible parameter set permutations and determine the default settings. Then the online optimization process is continuously running as a background process on the working system adjusting network settings if required. In our work we consider these two search phases as separate scenarios, with the offline explorations being primarily conducted for the WSN case study.

4.4.2 Versions of Hybrid Simulated Annealing

As said, the *simulated annealing* algorithm is a popular metaheuristic method that is known to be effective for “black box” type of problems [279]. We have chosen this technique as a case study, first, for its simplicity. It is quite easy to introduce a modification and track the effect of changes on this method. If proven to be successful the corresponding adjustments can be introduced to other metaheuristics, while already having an understanding of possible impact of the changes. Second, the SA potentially allows the use of a smaller number of evaluations, compared to, for example, genetic algorithms [292] that employ parallel searches to fill in the candidate population. This can be important for the online application. Simulated annealing has also been shown to have good asymptotic convergence properties. The pseudocode of this method is given in Appendix C.

4.4.2.1 Problem-specific Modifications

During our test runs we have been observing that the faster a stable near-optimum solution is achieved, the more likely the SA tries for other, significantly worse parameter combinations. This is due to the impact of the temperature factor that opts for highly varying

parameter sets at early stages. Therefore, in addition to considerations for the actual temperature and the dynamics of utility changes, we introduced an option of specifying a desired, ideal, utility value. The knowledge of the *ideal utility* allows to the optimality gap, and, based on this value alter both the probability of jumping to a newly discovered state, and the dynamics of the temperature decrease. The latter is especially useful in the online scenarios, where it is often feasible to stop the search as soon as a near-optimum solution is found in order not to disturb the user with further re-configurations. Surprisingly enough we observed that often it is the knowledge of the ideal utility and its heavy exploitation in the metaheuristic that brings higher gains than usage of sophisticated probabilistic models.

In order to overcome short-time variations in utility, introduced by fluctuations of the operational environment, we apply a system of two sliding windows to pre-process or smooth the measured data points. The sliding windows track if the variance of the utility values has stabilized. The first window smoothes a small number of data points, in order mitigate the short term performance fluctuations. Its output serves as an input for the second sliding window., which estimates when the network behavior becomes stable after re-configuration. For the online application the utility is continuously monitored. If the system notices a sudden drop in utility that cannot be contributed to short-time variances, but indicates a change in the network context, the SA is probabilistically restarted with a reheating temperature proportional to the utility drop. A static memory of the best parameter values discovered over time is used to boost up the algorithms performance by letting it first try those combinations. However, if these parameter settings perform far from the expectations, the used probabilistic model is nulled. The learning starts anew, as we suppose that the operational context has considerably changed, and the earlier model is not applicable anymore. Additional temperature thresholds are defined for the regular reheating phases, which are aimed at a limited additional exploration of the search space and detection of improved networking conditions. The reheatings are scheduled in periodicity and duration to minimally disrupt the users. Obviously, such rather conservative exploration strategy for the online scenarios delays finding of the optimal solution, but, as we observe, it limits the negative experience of the user caused by poorly performing network settings. Additionally, the default protocol settings that are always pre-defined in any OS introduce the baseline and the starting point for the search. We can always revert to these settings in the case if the search is unsuccessful.

4.4.3 Considered Probabilistic Models

Classical metaheuristic search methods, including simulated annealing, do not include learning. For example, SA can store the states with best utilities, but cannot deduce from them the most promising directions of search using long-term knowledge. Introduction of the learning mechanisms capable of such an analysis as part of the metaheuristic search has been widely researched and resulted in development of such hybrid techniques as linkage learning genetic algorithms [293, 294], application of racing algorithms for metaheuristics tuning [295], Bayesian optimization algorithm [296] that utilizes Bayesian modeling to estimate distributions of promising solutions to improve the process of generating new search states as part of the evolutionary computation. However, such techniques have not been widely applied in the wireless communications research.

We contribute to this line of research by developing and assessing the performance of hybrid versions of the simulated annealing method that incorporate different probabilistic

models. We investigate how more aggressive usage of the problem specific knowledge influences the optimization. From one side, we expect better results and faster convergence of the search. From the other side, the probabilistic models might have only limited applicability with respect to changing operational conditions and application scenarios. This can lead to suboptimal investigation of the search state space, which may in turn lead to the poor algorithm convergence, due to the increased risk of being stuck in the local maxima, as well as introduction of additional unjustified complexity.

We have chosen *probabilistic models* for representation of the sensor-actuator dependencies that are induced by the feedback loop running the metaheuristic-based optimizer. The choice for the probabilistic modeling is determined by the stochastic nature of many of the wireless networking components observed cumulatively, e.g., user behavior, interference and propagation conditions. Somewhat similar to the approach introduced in [296], we use these models to give a "direction" of the SA search. Basically, we assign probabilities to parameters to be chosen for the alteration, as well as probabilistically determine which value should the chosen parameter take. In the algorithm provided in Appendix C the function *neighbor* incorporates these models. It also outputs the settings of the adjusted parameters. We have examined three classes of models, namely, the simple ones that consider only basic marginal probabilities, the correlation coefficients based, and the basic Bayesian networks [297]. These three models consecutively grow in complexity, and pose increasing requirements to the training duration in order to produce useful estimates, but the quality and the accuracy of the corresponding predictions increases as well. These probabilistic models combined with the classical random search strategy, and the variants of the annealing mechanism result in a number of hybrid variants of this metaheuristic.

4.4.3.1 Random Search

In the basic version of the simulated annealing, the random function, as well as the temperature and the utility dynamics, i.e., the annealing/cooling process, determine the number of parameters to be altered, as well as the magnitude of changes of the particular parameter configuration. The classical simulated annealing is very robust against being stuck in the local maxima. (In all of the discussed search strategies we always use a random function to introduce uncertainty in the search process in order to avoid local maxima. The random factor influences the choice of the new state to be explored, as well as its acceptance as a new starting point for a new iteration of the search. We would not explicitly mention this component further.)

4.4.3.2 Marginal Probabilistic Search

This simple probabilistic model reflects marginal probabilities of increasing the utility if a certain parameter setting is getting altered. For each parameter we keep statistics indicating if its increase or decrease has positively or negatively influenced the utility. The ratio of these numbers determines the likelihood of the parameter value to be either increased or decreased. This statistic is then applied to indicate the promising direction in the parameter value change. The rest of the simulated annealing algorithm stays the same. The proposed metric is applicable only in cases where parameter values can be put into a sorted order. For example, the retry limit parameter can be accommodated into this scheme, but the TCP variants/flavors cannot. (Our experiments with variants of this statistics being applied for choosing the set parameters to be tuned did not lead to

sufficient performance increase in the studied scenarios, and therefore the corresponding version of the SA would not be studied further.)

4.4.3.3 Correlation-based Search

The second model we consider is based on the correlation coefficients between parameter settings and utility values. High correlation values indicate a strong linear dependency of the parameter values and the utility. Again, such a metric can be applied only to parameters with sortable/quantifiable setting. To incorporate this model the SA is modified to probabilistically choose a set of adjustable parameters based on their correlation coefficient values, with the size of this set being determined by the cooling scheduler. For high temperature values the parameters with higher correlation values are more likely be chosen for adjustment, as we typically want to get to the promising search neighborhood as fast as possible. As the temperature decreases the local search starts to matter, and the parameters with weaker correlation values have higher chances to be chosen for exploration. (The additional variant of the simulated annealing we considered also included consideration of the correlation coefficients to determine the direction and the magnitude of change in a single parameter reconfiguration. However, it did not perform significantly better then the version with the correlation-based choice of parameters and the marginal probability based their value selection. This is, probably, due to the small range of per-parameter settings specified as part of the optimization task statement).

In our experiments the application of the correlation-based model has led to significant performance improvement only if there were sufficient amount of samples available to perform training and establish the correlation coefficients. Otherwise, no notable difference as compared to the previous search strategy was observed. This is explainable as the use of correlation coefficients heavily alters the parameter search landscape, speeding the convergence of the search. Obviously, use of wrong correlation coefficient estimates would both slow the convergence, and overall worsen the quality of obtained results. We obtained the largest performance gains from utilizing the trained correlation coefficients to set the upper boundaries on the probability of a certain parameter to be considered for the SA optimization. This modification was effective in combination with both the standard and most of the advanced SA versions.

The correlation trends of parameters and application-specific utilities tend to stay the same for a considerable range of operational conditions (including different topologies and application traffic patterns, see Chapter 5, Section 5.3). However, if the networking environment experiences a drastic change, it is likely that the influence of parameters via attributes to the final utility changes as well. Therefore the established correlation coefficients would be invalid. For example, if a system was trained on a wireless topology that does experience hidden terminal problems, then the RTS/CTS mechanism would have only negative impact. However, if there is a topology change that leads to the hidden terminal problem, and the UDP traffic is run on the relevant nodes, then the RTS/CTS mechanism becomes useful and the corresponding correlation coefficient values would change completely [298]. Similarly, high wireless traffic might require both advanced QoS-aware queue management, as well as the increase of the MAC retry limit parameter or the maximum backoff to improve network reliability. Low network loads do not typically benefit from adjustment of these parameters.

Inspired by our experiments with parameter-utility correlation coefficients conducted as part of the work on experimentation with the simulated annealing, we continue

investigating this topic in Chapter 5, Section 5.3. There we establish that the models based on the parameter-KPI correlation coefficients are, indeed, very promising for wireless CSMA/CA networks. They allow efficient and *robust* capturing of the influences protocol parameters have on observed performance values. Considerations for protocol parameter dynamics allows for robust clustering of the operational contexts on the basis of their sensitivity towards protocol parameter changes. The promising extension of the work taking into the account the results from this chapter, as well as Chapter 5 would be the derivation of the sampling approaches that lead to obtaining reliable context-aware correlation coefficient models, while avoiding exhaustive probing.

4.4.3.4 Bayesian Graphical and Composite Models

We also consider models based on Bayesian networks [297, 299, 300]. Surprisingly, these models do not perform notably better than the simple probabilistic search. As a representative example we have chosen the trivial Bayesian graphical model without hidden nodes with the connection from parameters to the KPIs and then to the utility. The values of the KPIs and the utility are quantized into several (five) classes to enable conditional probability reasoning of the form: "Up to X parameters can be changed. What X parameters do I need to change and to what values, provided that I want to obtain the best possible utility with probability Y, and the values of $(N-X)$ parameters are fixed to the values $(x_1,\ldots,x_{N-X})$?". The higher the probability Y, the more probable it is that the taken actions lead to the expected results. The value X, that is the number of parameters to adjust, is determined by the annealing mechanism as a function of the current temperature, the utility and the random component.

Unfortunately, if not sufficiently trained, the Bayesian model being incorporated in the SA can result into the search getting stuck in local maxima. Our experiments have confirmed this expectation, as we observed the slow convergence and poor performance of this variant of hybrid SA. Basically, both the Bayesian and the correlation models, though the latter to the less extent, face the same dangers. These problems cannot be effectively solved by simple introduction of higher randomness in accepting proposals from these models, as the convergence is still affected by the application of untrained models. After some experimentation, we found that the best way to utilize these models is to use their estimates only, if those have high certainty. Therefore we developed the *composite models*. First, to obtain a sufficient amount of samples we run only the SA with the simple marginal probabilistic model. After each parameter was altered on average three times, we start to use for 50% of the actions either the correlation or the Bayesian network model. The probability of using the simple marginal probabilistic model goes down to 10%, as more samples are gathered using the other models. The answers provided by the Bayesian network are utilized only if their certainty is significant (e.g., above 30%). We note that the number of different Bayesian models is very large, and our conclusions apply for the particular model studied here. Future work is definitely needed to extend these conclusions towards different ways the Bayesian models can be applied in such scenarios.

The composite mechanism is applied as follows. In the beginning, the algorithm asks for possible re-configurations and probabilistically chooses the ones from their ranked list, where the sum of probabilities for improving the utility class (e.g., from "normal" to "good" or "very good") is higher than the threshold. If the required number of parameter value alterations is not reached, then the algorithm requires for a list of parameters and their values with negative performance expectations, which certainty is also above the

threshold. It then subtracts these from the rest search state space, and run the SA with the simple marginal probabilistic model on the remaining parameter permutations.

For the correlation model we basically do the same. We choose the parameters to adapt if only their correlation coefficients values are indicative (above 0.2 or below -0.5). If there are more parameters to change as provided by the correlation model, then the rest are chosen randomly using the simple marginal probabilistic variant of the SA. Obviously, as said earlier, any action proposed by these search strategies would be accepted only on a probabilistic basis, and otherwise substituted by the random exploration.

4.4.4 Scenarios and Evaluation Results

We evaluate the performance of the SA and its hybrid variants in several scenarios. As a motivation, we demonstrate convergence of the search technique in the online scenario with a complex non-linear utility function, where a YouTube video is downloaded by the multi-access node in changing operational conditions. This scenario is run in the Qualnet simulator. We also study the online performance of the SA optimizer in a Wi-Fi testbed. We consider two testbed setups, a three hop wireless network, and a six node access network with one access point.

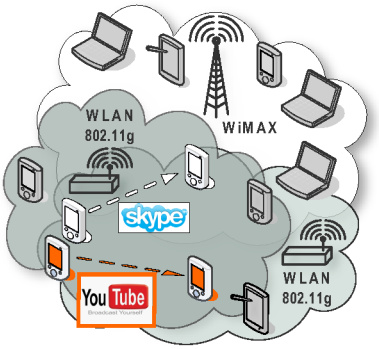
Next, we compare different versions of hybrid simulated annealing in the online and the offline scenarios. In the online scenario the cross-layer optimization is executed for a multi-access node running a VoIP application. The network load for the respective networks varies over time. This scenario utilizes the Qualnet simulator. The offline scenario is based on wireless sensor network testbed [51], which was already shortly described in Chapter 3. There, the effectiveness of the optimization conducted with several variants of hybrid simulated annealing is compared across several topology setups. On this scenario we also study the feasibility of pre-training of probabilistic models.

All the variants of the metaheuristics are realized in MATLAB with the additional interfaces developed to fetch results from the Wi-Fi and WSN testbeds [51], as well as the Qualnet simulator [291]. The optimization relies on the feedback loop that queries the systems every N milliseconds and gets information on node performance. Using the same interface, parameters of the nodes are adjusted as a result of the decision process. All the experiments that are described below were conducted at least 20 times to ensure the consistency of obtained results.

4.4.4.1 Motivational Examples: SA-based optimization in the Wi-Fi Testbed, and YouTube Traffic Simulations with Complex Utility

Before exploring further the gains provided by possible enhancements to the simulated annealing, we demonstrate the application of the basic version of this algorithm to online cross-layer optimization. First, we consider a simulation scenario, and then apply this metaheuristics on a Wi-Fi testbed.

Bulk video traffic download (YouTube). This scenario demonstrates that the metaheuristic algorithm can find good solutions in very short times also when it operates on complex, irregular shaped, utility functions. We consider a setup with an individual connection over which the YouTube video is downloaded. The download traffic is shaped as observed in [301], with the reported typical video file size of 10 MB and a video duration of about 4.15 minutes. Contrary to the original study we increase the average size of the downloaded files to 105 MB, which is representative for high quality movie trailers. The user can access two broadband wireless networks with the maximum datarates



Protocol	Parameter	Values
TCP	Delayed ACKs	ON, OFF
MAC	cwMin	2, 5, 20, 40
	cwMax	255, 511,1023, 2047
	Retry limit	1, 3, 5, 7
	RST/CTS	ON, OFF
MTU, and correspondingly adjusted packet size settings at the upper layers		500, 1000, 1500, 2000
Access method		802.11a, 802.11b

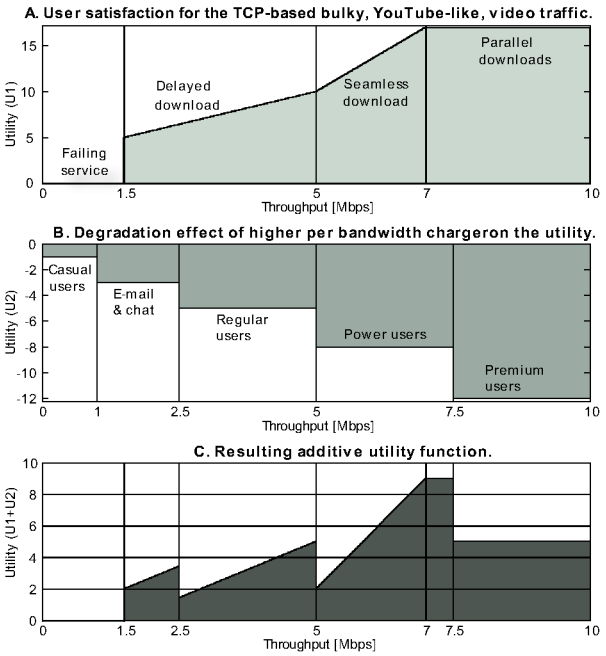


Figure 4.7: The summary of the YouTube scenario [54]. The figure includes, first, the illustration of the YouTube and the VoIP multi-access scenarios, where protocol settings of an individual connection are optimized. Second, it reflects the possible protocol parameter settings for the YouTube scenario, which result in 2048 permutations. Third, the utility of the YouTube application as a function of throughput and cost is provided.

of 11 Mbps and 54 Mbps. The adjustable protocol stack parameters and their possible settings are summarized in Figure 4.7. This scenario is realized in the Qualnet simulator.

The considered utility metric aims to produce predictable and low-varying bandwidth usage to avoid high peak loads on the operator (see Figure 4.7). The utility function depends on goodput and cost. However, both of them change non-linearly. The goodput's influence on the final utility depends on the expected time required to download the file. The time is calculated with assumption that the goodput stays constant through the download. The longer the download time the less the user is satisfied. In our case we assume the file of 105 MB has to be downloaded in less then 4.15 minutes for high user satisfaction. Therefore, we assume that if the expected goodput is less than 1.5 Mbps the user leaves the network as she experiences too high delays, so the utility is zero. At the other side fast downloads with goodput of more than 7 Mbps result in the highest satisfaction. For the cost component we consider the per-data-rate price plan. It can be advantageous to network providers as the flexible charging motivates more efficient utilization of network capacity and, therefore, leads to higher returns on investment. The cost function we assume models a goodput-dependent price plan for network usage. The higher the utilized datarates, the more the user has to pay.

In Figure 4.8[2] the simulation results obtained when running the YouTube scenario are provided. We compare the two node behaviors. In the first one the user node can only switch between the wireless interfaces, which is done in the ideal manner, i.e., at

[2]Most of the graphs plotted in this section are a smoothed and averaged version of the results received, post-processed for visualization clarity.

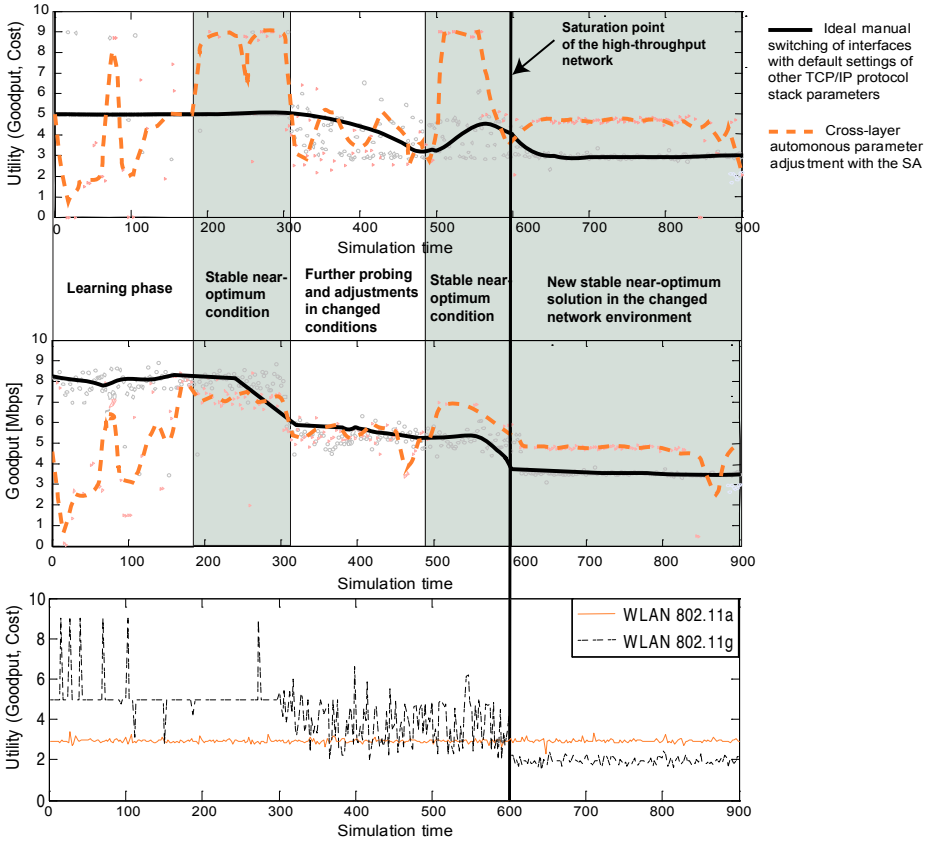


Figure 4.8: The performance of the network optimizer based on the simulated annealing for the YouTube scenario [54]. The performance is accessed in terms of the utility and the goodput. Figure shows smoothed averages from the large number of simulation ensembles, some of the raw measurements points are depicted on the background to show variance range of data. The lowest part of the figure demonstrates the default performance achievable is the node stays constantly connected to each of the wireless interfaces.

any point of time the better performing interface is chosen (the black line in the figure). Other protocol parameter settings stay at the default levels. In such setup the interface has to be switched only once, at the time of 600 seconds, when the performance of the 802.11a network severely degrades. The second behavior is determined by the simulated annealing algorithm that operates on the extended set of the TCP/IP parameters. As we see from Figure 4.8 after each of the learning periods (0-180 seconds, and 310-480 seconds) the optimizer manages to find the parameter settings that significantly improve of the default performance (the orange vs. the black line). Also after the networking conditions change one time at 600 seconds, the simulated annealing does not only switch between the wireless interfaces, but also finds additional protocol parameters settings that further improve the performance.

We also see that the proposed utility function indeed allows to limit the network load. This is especially notable in the periods from 180 to 310 seconds and around 550 seconds. These results confirm our earlier claims that the complex utility metrics, if carefully designed, can be beneficial for both providers and users. In our example the user receives

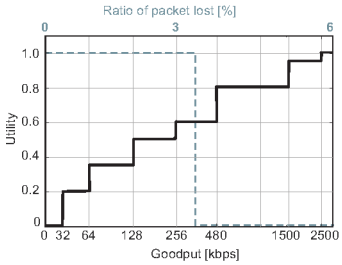
Table 4.2: The protocol stack parameters and their values for the Wi-Fi testbed scenario, 2520 permutations per application type. (Boldface indicates the default settings.)

Protocol	Parameter	Values
Download	Datarate	256, 512, 1000, 2000, 4000 kbps
Streaming	Datarate	276, 660, 1000, 2200, 3000 kbps
TCP	Congestion control mechanism	**Reno**, Scalable, BIC, HTCP, Hybla, Vegas, Westwood
MAC	RTS/CTS thres.	**ON**, OFF
	Fragmentation (if > 500 bytes per fragment)	**1** (none), 2, 3
MTU and correspondingly adjusted packet/segment size settings at the upper layers; (MAC fragmentation is applied to these MTUs)		500, 1000, **1500**, 2000
ISM 2.4 GHz Wi-Fi channel (shared by all the nodes)		1, 5, **9**

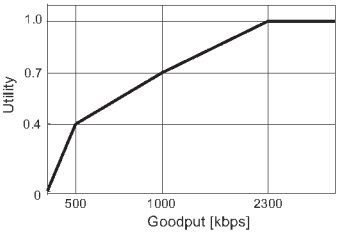
a satisfactory quality of service, and her node does not opt for more bandwidth, which allows the provider to allocate the spare network resources to the other clients. Potentially such a strategy allows the provider not to loose her customers due to congestion, and the users can have their primary interests satisfied with the decreasing number of handovers. Overall, in our scenario, the use of the advanced utility and the protocol stack cross-layer optimization leads to the average performance increase of 24%. The gains in the stable operation phases vary between 65% and 80%.

Wireless Testbed Scenario. Having shown that simulated annealing is effective in the simulator, we further demonstrate that this metaheuristic can be successfully applied online in the real networking environments. We consider two Wi-Fi network setups. First, we set up the multihop scenario where data is sent over three hops. Second, after [54], we shortly consider the variants of the access point scenario with several client nodes interacting with the access point. There is an interference from the another Wi-Fi AP on the channel 6. We experiment with two types of traffic, both emulated using the iperf application [302]. The first is the videostreaming traffic that is characterized by a stepwise utility and hard constraints on the allowable packet losses, see Figure 4.9a, and the other is the YouTube traffic that has a utility approximating a log function, see Figure 4.9b. The overall network utility is defined as a sum of the utilities of the individual nodes.

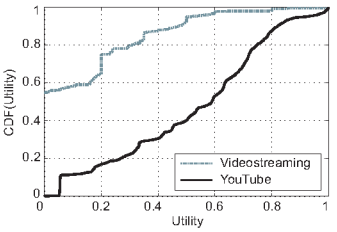
The adjustable parameters for this scenario are summarized in Table 4.2, including the standard TCP/IP stack operating on a set of parameters exposed by the Linux operating system. As with the parameter-based cross-layer optimization as a whole, only few parameter changes in this scenario can lead to significant improvement in the performance. Some of these influences are context dependent, for example, the channel switching is only effective if there is an interference present on some of the frequency ranges. It is the job of metaheuristic to identify and adapt those protocol stack parameters that lead to significant performance improvement in the studied scenarios. In our scenarios we alter the application layer datarate to emulate different Quality-of-Experience (QoE) provided by the invoked application. We specifically set the datarate parameter for the videostreaming application to take less values then the number of steps defined by the corresponding utility function Figure 4.9a, as often the control of the application cannot be performed at the desired level of granularity. We can also adjust the MTU size, the TCP congestion control flavors, the MAC configurations, and the transmission channel settings. The MAC



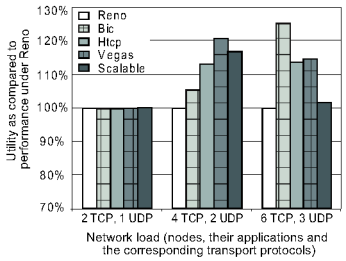
(a) Utility of the UDP-based videostreaming as a function of goodput and packet losses.



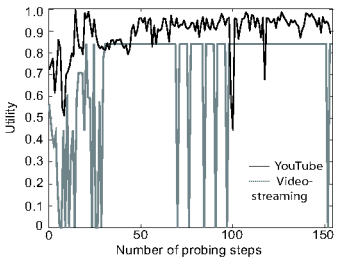
(b) Utility function of the TCP-based YouTube application.



(c) Cumulative distribution function of utilities for the multihop subscenario.



(d) Influence of the TCP flavors on the wireless access network with mixed traffic.



(e) Sample performance in the multihop subscenario with cross-layer per network optimization.

Multi-hop scenario	Mean performance (variance)		
	Default (with best app. rate)	Per network	Per node
Videostreaming (UDP-based)	0.84 (0.12)	0.824 (0.15)	0.825 (0.13)
YouTube (TCP-based)	0.77 (0.14)	0.978 (0.17)	1.026 (0.17)

(f) Summary of the performance achieved in the multihop scenario for per-node and per-network optimization.

Figure 4.9: Illustrations for the Wi-Fi testbed scenario [53]. (Please note the logarithmic scale for figures (a) and (b).)

layer configurations include the activation of the RTS/CTS mechanism, and the packet fragmentation. In order to enable cross-layer optimization on the per node basis we have provided an individual virtual application server to each client, as Linux does not support parallel protocol stacks with different settings. In other words, each client has it own server that runs its own protocol stack that shares the same configurations as the client protocol stack.

In order to establish a baseline for our evaluation we conducted an exhaustive search of parameter value permutations with all the nodes sharing the same configuration. The sample CDF distributions for the UDP- and TCP-based traffic in the multihop ad-hoc scenario are shown in Figure 4.9c. The sample runs of the metaheuristics algorithm are shown in Figure 4.9e, and the summary of the results for the multihop scenario are displayed in Table 4.9f. All utility values are plotted in normalized form to facilitate comparisons, with 100% (1.0) utility values corresponding to the best settings discovered through the exhaustive search with the protocol settings being common for the whole network. The initial settings used to start the SA search correspond to the default configuration of the TCP/IP stack. The application datarates are setup to provide the maximum utility for the default underlying stack settings (see Table 4.2). By comparing the per-node and the per-network optimization results, we can observe how a mixture of the size of the search space space, and the strength of the influence of a particular parameter, determines the gains in the utility as compared to the default protocol values. Generally, the option of the cross-layer optimization with the protocol settings being common for the whole network is interesting for the client-side configuration in a homogeneous user environment,

e.g., when all nodes in the wireless network support the same range of settings, and experience similar operational conditions. The cross-layer optimization conducted on the per-node basis can be promising in highly non-homogenous network setups.

For the multihop UDP scenario the default TCP/IP configuration with the best values of the application-layer settings obtain a good utility of 84.2% from the maximum. Introduction of the cross-layer optimization performed either on the per-node or per-network granularity does not lead to further utility improvement. This is due to the fact that influence of additional protocol stack parameters, such as the MTU size, is not prominent. During rather short exploration phase (on average 15 to 40 probing steps) the increase in the search state space and the correspondingly obtained utility gains balance out the losses incurred due to the poorer application rate settings. Moreover, the per-node adaptation, though operating among others on such an effective parameter as the per-hop channel assignment, requires about as twice as long initial exploration phase to achieve the comparable performance, as the state space of this optimization is much larger. Though, of course, potentially, the channel assignment, the rate control, the retry and contention settings, adjusted on the per-node basis can further improve the obtained network performance [303].

In the multihop TCP scenario the situation is different. The cross-layer optimization leads to much better results as compared to the default settings (77% vs. 97.8%), which is not surprising considering the well known suboptimal performance of the TCP Reno congestion control in the wireless environment [145], see Figure 4.9d. At the price of slower convergence rate and higher variance of the achieved results, the per-node optimization leads to further utility improvement that exceeds our best exhaustive search estimate obtained with the common settings shared all the nodes (102.6%). This is explainable as for each wireless hop individual settings that increase link reliability might be required due to the variations in the interference conditions, which leads to a different datarate/coding scheme used (in our experiments we let this setting to be auto-configurable). The biggest performance boost comes from the per-node channel allocation and reaches up to 25% compensating suboptimal settings of the TCP flavor. However, these gains are not as high as one may expect due to the elastic nature of the utility functions, and the suboptimal channel allocation, which simply could not be sufficiently better on the very large state space to be explored only in a limited number of exploration steps (less than 100).

The results obtained for the wireless AP scenario with the six client nodes running either the TCP- or the UDP-based traffic are inline with the previously obtained results [54]. The cross-layer optimization allowed to increase the network throughput by at least 13% while maintaining a high level of fairness. In this scenario the application-layer traffic shaping played a significant role as the dynamic adjustment of the UDP traffic was required to improve the network fairness in the cases of varying external interference. Basically, the application-layer parameter tuning led to similar effects that can be achieved by QoS mechanisms invoked on the network or the MAC (IEEE 802.11e) protocol layers. This, again, is inline with the discussion in Chapter 2, indicating degeneracy of TCP/IP, which leads to both the flexibility and the robustness of this architecture.

Obviously, a reasonable shaping and decomposition of the cross-layer optimization state space, which, for example, includes disjoint per node channel allocation, followed by the per network TCP and MAC parameter settings, and application rate adjustment, would result in much better optimization results than brute force application of simulated annealing demonstrated above. Still, to our surprise, the direct online application of this metaheuristic and the analysis of its search progress allowed to both improve the

performance of studied networks, as well as to find and obtain an initial understanding of the structure of the promising regions of the optimization state space. (Though much of the latter, such as importance of the non-overlapping channel assignment, transport protocol aware traffic shaping, or suboptimality of some of the TCP flavors for the wireless conditions, is, of course, obvious to any networking expert. However, as we partially demonstrate in the other parts of the chapter, the results obtained with metaheuristic mechanisms can also be useful for the scenarios that are less straightforward for the expert opinion.)

4.4.4.2 Evaluation of Hybrid SA on the VoIP Simulations and Wireless Sensor Testbed

In this section we evaluate how different hybrid versions of simulated annealing perform when applied to the task of parameter-based cross-layer optimization. For the online version of the problem we consider the scenario when a single node runs a voice-over-IP call, and has an option of accessing several wireless networks, with stacks that can be additionally optimized. For the static offline optimization we take the scenario where wireless sensor networks are configured to provide the most energy efficient performance, while satisfying the basic constraints on other KPIs, such the delay and the packet losses. The WSN scenario is interesting as there the influence of individual parameter settings is not so widely studied as for the Wi-Fi networks. Besides some of the parameters, such as the duty cycle, are known for their non-linear influence on the network power consumption [205]. The WSN scenario also allows us to experiment with the application of pre-trained models as part of metaheuristics. For this we consider the models obtained when optimizing different WSN topologies.

VoIP Multi-Access Scenario. The first scenario deals with runtime optimization, where a user initiates Voice-over-IP (VoIP) calls that can be served by a number of wireless networks, IEEE 802.11a WLAN, IEEE 802.11g WLAN, and WiMAX, see Figure 4.7. This setup is quite similar to the YouTube scenario from the previous subsection. Again we consider an optimization of an individual connection under the varying interference conditions conducted in the Qualnet simulator. The optimized parameters are the same as for the YouTube scenario, with the exception of the TCP settings, which is irrelevant as the VoIP does not rely on this transport protocol. However, the set of values for each parameter to take was altered, so that the total number of permutations was around 2000. The most notable change is the option of using even bigger packet size of 3000 bytes. Second, instead of having the RTS/CTS mechanism to be simply switched on or off, we used after the Qualnet realization, the threshold based on the packet size to activate this mechanism automatically. The incentive is as follows. In the case of the hidden terminal it may be suboptimal to use the RTS/CTS for small control packets, but this mechanism can be useful for big data packets. The values we use are (50, 600, 1600, and 10000), where the first value basically corresponds to the constant use of the RTS/CTS, and the last one to the OFF state. In the third part of our simulation the performance degradation in IEEE 802.11g network is exactly caused by newly added node that is situated so to create the hidden terminal problem for the optimized VoIP node.

The utility for the VoIP scenario is a function of the obtained Mean Opinion Score (MOS) and the cost of the network access (see Figure 4.10a) with maximum achievable value of 3.3. The WLAN access points are free of charge, but their performance varies over time, and towards the end of the scenario they become saturated. The user has to pay for WiMAX usage, but the capacity of this network satisfies all her requirements.

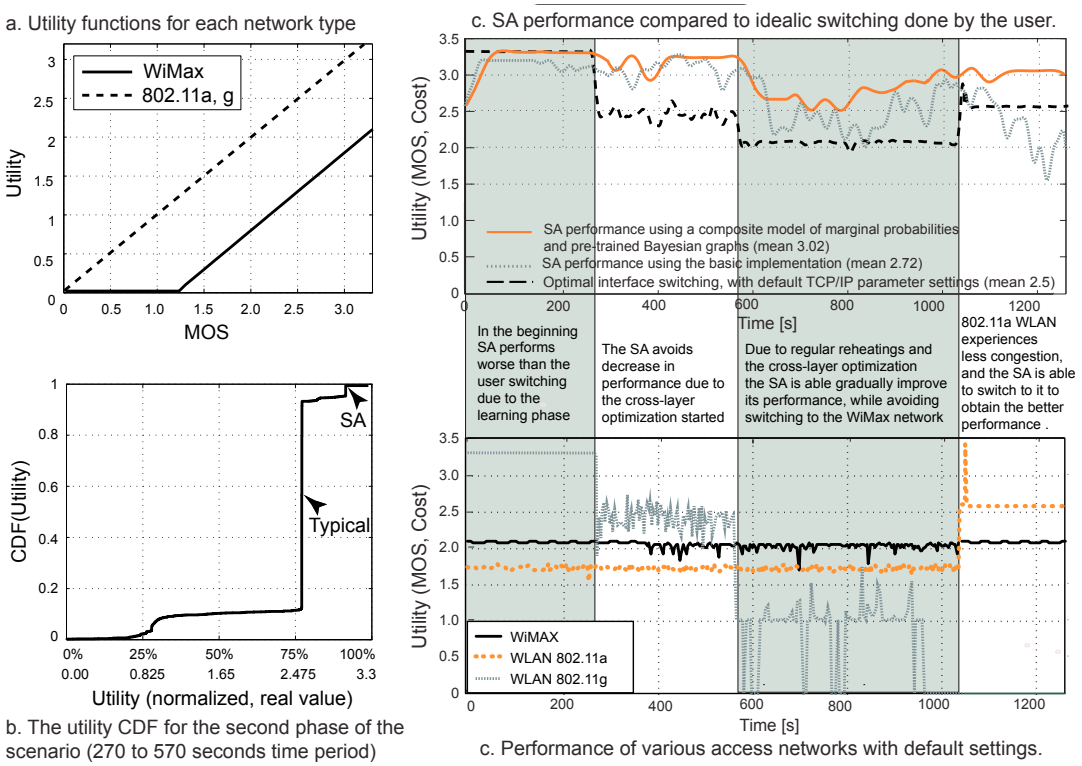


Figure 4.10: Performance of the basic simulated annealing in the VoIP scenario [46].

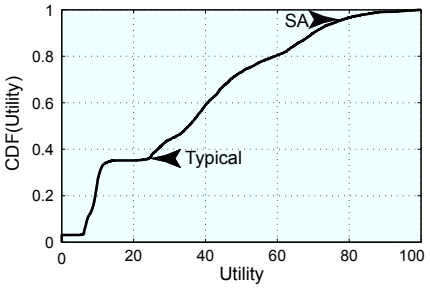
The mean opinion score specification is taken directly from the Qualnet simulator, as a function of delay, packet loss and datarate. For simplicity we assume seamless vertical handover [304]. Figure 4.10b shows the cumulative distribution function of the utility of all parameter permutations for the operational context for the second part of the scenario (approximately 250–600 seconds).

Under normal circumstances the user will always stay connected to the free access points and switch to WiMAX only when the MOS drops very low. Utility values provided throughout the scenario by different wireless networks with default parameter setting is shown in the down part of Figure 4.10c. The upper part of Figure 4.10c shows the ideal device behavior in terms of interface switching provided the parameter setting are kept as the default level, as well as the results obtained by executing both the basic and the hybrid versions of the simulated annealing. The hybrid variant uses the composite probabilistic model, which employed Bayesian graphs. We show these two versions of the SA to underline the possible variations in the behavior of the optimizer caused by different strategies in exploration of the optimization search state space. Overall, the obtained data indicates that the SA-based cross-layer optimization results in better performance than can be obtained through pure interface switching, even if the latter is always performed optimally. With the simulated annealing we achieve the average utility values of 2.72 and 3.02 for the basic and the hybrid versions as compared to 2.5 with the interface switching, which corresponds to 14% and 21% of improvement, respectively.

Wireless Sensor Network Scenario. As we want to study the applicability of metaheuristics to a wide range of wireless networks, we have constructed a second scenario featur-

Protocol	Parameter	Value
Application	Packet frequency (s) (one reading per 0.2 s)	0.2, **0.8**, 1.4
Routing	Beaconing interval (s)	5, **20**
	Timeout (s)	**0.8**, 1.5
MAC	Max. retry limit.	1, **4**, 7
	Duty cycle (%)	5, 10, **100**
	Backoff length	5, **35**
	Acknowledgments	ON, **OFF**

(a) The optimized parameters and their values. (Boldface indicates the default settings.)

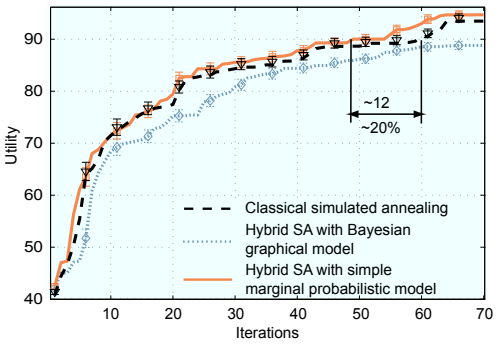


(b) Sample cumulative distribution function of the normalized utility values for the eight node random WSN topology.

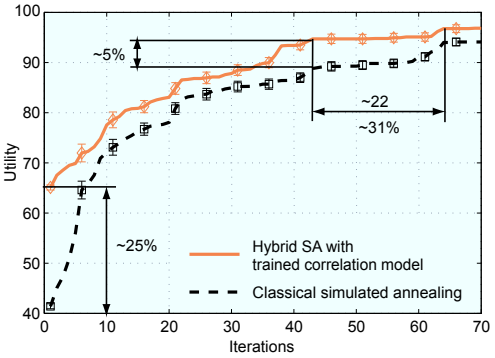
Figure 4.11: Wireless sensor network scenario. The considered parameter settings, and the sample utility CDF [46].

ing low-power radios. For prototyping we have chosen wireless sensor networks (WSNs). The performance characteristics of these networks are highly different from broadband wireless networks of the previous scenarios. As signals of low-power radios are easily disrupted, these networks are known to be extremely sensitive to the deployment environment, and therefore, it is of interest to conduct extensive static, pre-deployment tests of these networks. Most of these networks are foreseen to have miniature nodes, operate on batteries, and work autonomously with low traffic loads for a long periods of time. Therefore, the extension of network lifetime, i.e., reductions of power consumption and energy efficiency becomes one of the primary performance goals for WSNs. Additionally, as these networks are foreseen to serve for a wide range of very diverse applications, there exist no widely accepted a well-performing protocol stack for them, and multiple protocols might be considered for deployments. For this case a testing tool for evaluation of combinations of these protocols, which scales well needs to be developed. In Chapter 3 we proposed the CONFab framework for this purpose. There we have already shortly considered application of metaheuristics, in particular the genetic algorithm, for the purpose of autonomous reconfiguration of the protocol stack for specific deployment conditions (see Section 3.3.2.1). Here, we continue this work on an extended set of scenarios focusing on how this optimization can be conducted more efficiently if another metaheuristic search, namely simulated annealing, is modified.

We have evaluated the performance of the SA-based optimization algorithm in a WSN testbed over a wide range of scenarios using different utility functions and network topologies. Network sizes varied from four to twelve nodes with linear, mesh and random topologies. Utilities were functions of power, reliability, goodput and delay. As a representative example we use averaged results of these tests with the utility as a function of power consumption with a constraint on the reliability (maximum number of packets lost). The adjustable protocol stack parameters and their values are summarized in Table 4.11a. As in the previous scenarios functionalities of some of the parameter settings overlap. For example, switching off of the MAC unicast acknowledgement makes the maximum retry limit configuration meaningless. We limited total number of parameter value permutations to 432, since we wanted to evaluate performance of the algorithm on scenarios where the exhaustive search is possible. The cumulative distribution function (CDF) of this utility is shown in Figure 4.11b. Our experiments show that the sigmoid shape of the CDFs as depicted in Figure 4.11b can be considered representative for many network



(a) Performance of the basic version of the SA, and its hybrid variants based either on the simple probabilistic marginal model or the Bayesian network.



(b) Performance of another hybrid SA model that utilizes the pre-trained correlation coefficient model.

Figure 4.12: Performance of the variants of the simulated annealing during the offline cross-layer optimization for the wireless sensor network scenarios [46]. The eight node random topology is considered.

stack optimization problems. As can be seen, high utility values can only be obtained in 3-5% of the cases.

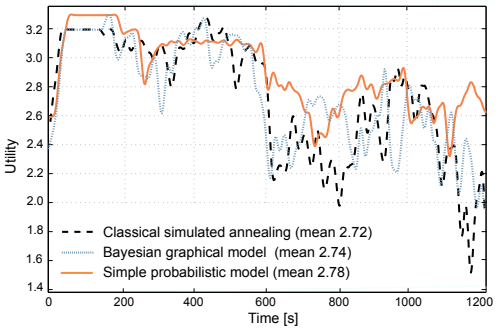
The static optimization of the WSN scenario provides even better results as compared to the online VoIP scenario (see Figure 4.12). The basic SA converges to 90% from the maximum after approximately 60 iterations. We reach 80% of the maximum within 30 iterations. The well performing hybrid variants of the simulated annealing reach those values even faster especially if the appropriate pre-trained probabilistic models are available. Further, we provide a detailed evaluation of a hybrid versions of simulated annealing on both of the considered scenarios.

4.4.4.3 Performance of the Simple Marginal Probabilistic Model

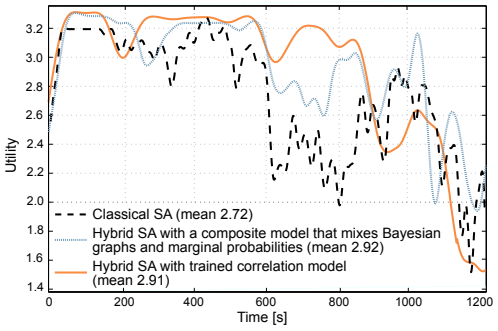
The simple marginal probabilistic model is computationally effective and usually performs better than the basic SA (see Figures 4.12a and 4.13a). The average convergence rate improvement in the static scenarios is around 7% with the local convergence gains reaching up to 20% (for the utility of 90% from the maximum). This variant of the SA also leads to a slight increase in the overall utility values achieved. For the runtime scenarios the utilization of this model leads to an average improvement of only 7%, however the local performance gains are higher and reach up to 30%. The gains over the basic simulated annealing are not surprising as a “direction” of adjustment of parameter values is given. As the same time this search strategy is still quite robust against local maxima, due to the fact that the disjoint influence of the considered parameters on the utility functions is typically monotonic. One may draw a parallel of this version of the search to the PSO with one particle, with the marginal probabilities dictating not the velocity of the particle, but the direction of its movement.

4.4.4.4 Performance of the Correlation Model

Additional insights into the parameter-utility correlations were gained during the preliminary trials. We experimentally confirmed that indeed different optimization actions on



(a) Performance of the basic version of the SA, and its hybrid variants based either on the simple probabilistic marginal model or the Bayesian network.



(b) Performance of two other hybrid SA models. One utilizes the pre-trained correlation coefficient model, another is a composite model that uses the Bayesian graph and the simple marginal probabilistic model.

Figure 4.13: Performance of the variants of the simulated annealing during the online cross-layer optimization for the VoIP multi-access scenario [46].

parameter configurations are required depending on the utility. As shown in Table 4.14a) correlation coefficients calculated for one utility can rarely be applied to a different one, if weights of the contributing KPIs significantly change. For example, the maximum retransmit limit setting has much less effect on power consumption of nodes than on their message delivery reliability (see Table 4.14a). However, there also exist key parameters, such the duty cycle that always heavily contribute the stack performance irrespective of the stated utility. Other parameters, e.g., the beaconing interval, have a significant influence only with respect to changes in operational conditions, otherwise they do not have a strong utility-specific contribution.

The direct application of the untrained correlation-based model typically did not lead to the performance improvement as compared to the basic version of SA. However, the trained models were useful, especially when they were applied to proportionally restrict the maximum probability for a parameter to be chosen in the search space. In our experiments, we calculated correlation coefficients on the data of several optimization runs. Though correlation coefficients obtained are not gathered over the whole parameter space, i.e., no exhaustive search was done, we still obtain notably better results than in case when all the parameters are treated equally (see Figures 4.12b and 4.13b). In the wireless sensor network scenario we improve the convergence rate for more than 30% if we consider the solution that provides 94% of the maximum utility. If we consider the results obtained only after few iterations then improvement is around 25%. In the VoIP scenario the improvement is significant only in the first two thirds of the scenario. In the last part, when the operational conditions drastically change, so that the new correlation coefficient model is required (which is unavailable), the performance becomes comparable to the basic version of SA. This again, indicates the robust, but still context and utility specific nature of this model. The correlation coefficient based model is also computationally efficient.

4.4.4.5 Performance of the Bayesian Graphical and Composite Models

We confirmed that the complex models, such as the one based on Bayesian graphs, start to work well only if they are trained on a rather large set of samples, i.e., these models are

Parameter	Correlation coefficient (scaled to the most influential parameter)		
	U(Power, Reliability)	U(Goodput, Reliability)	U (Reliability)
Packet frequency	0.56	1.0	0.61
Beaconing interval	0.02	0.16	0.3
Max. retry limit	0.09	0.02	1.0

(a) A limited set of correlation coefficients of parameters to several utility function in a WSN scenario.

Temperature	VoIP Utility	WSN Iterations to 90% of U_{max}
Boltzmann	2.78	72.0
Exponential	2.71	67.0
Linear	2.79	55.0

(b) Performance of different cooling schedulers in SA.

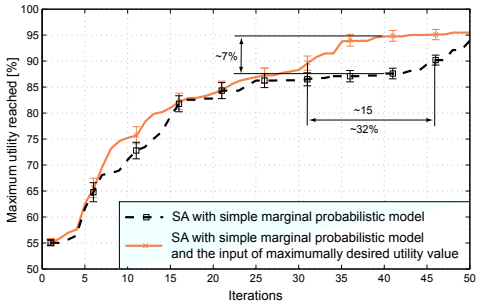
Figure 4.14: Further comparison of the effects of the modifications to the basic SA on the example of the VoIP multi-access and the WSN scenarios [46].

either pre-trained or work as part of the simulated annealing for a large number of search iterations, see Figure 4.10. Otherwise, as shown in Figures 4.12a and 4.13a, such models could perform even worse than the basic SA version, due to their tendency to get stuck in the local maxima. These results hold especially when very detailed data is extracted from the model. The situation improves by mixing the simple and the complex probabilistic models. For example, in the VoIP scenario the average performance increases from 2.72 to 2.92. Unfortunately, based on our experiments, we could not derive a composite model that would have be reliably applicable for a wide range of scenarios. Additionally, the computational costs for such complex graphical models are also quite high and these methods can be used only restrictively in resource-constrained environments. In future work we plan to study this problem further to hopefully find effective combinations of simple and complex graphical models to balance the local neighborhood and the global exploratory searches.

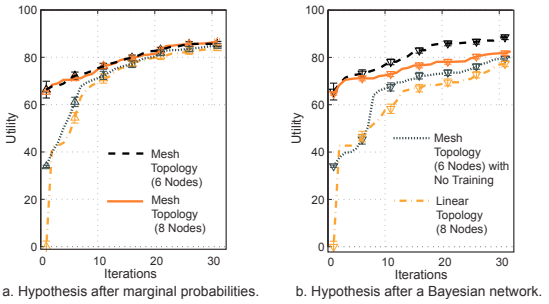
4.4.4.6 Effect of the Different Cooling Functions, and Utility-based Cooling

For the VoIP and the WSN scenarios we experimented with different cooling scheduler utilizing the Boltzmann, exponential and linear functions. Results summarized in Table 4.14b indicate that the linear temperature decrease leads to the best performance. Furthermore, we investigated the gains achievable if the information of the desired maximum utility is available to the simulated annealing. First, we used the maximum utility value to modify the probability of taking a new state space position as a starting point for a new simulated annealing jump. The closer is the utility value of the particular position to the maximum (desired) utility the more chances there are that this position is accepted for further exploration. Therefore, with this modification the probability of acceptance depends not of only on the temperature and utility change dynamics, but also the maximum utility. Second, the maximum utility value influences the dynamics of the search. The closer is the current utility to it the faster is the cooling process. This is especially useful for the online version of the algorithm. Similarly, the reheating period is set to be inversely proportional to the difference between the current and the maximum utilities, as if the utility is already high enough then the more likely it is the further re-configurations would harm, rather then improve the network performance.

Figure 4.15a shows the effect of exploitation of the utility-enabled version of the simulated annealing on the twelve node wireless sensor network scenario. One can observe the significant improvement to both to the convergence rate (by 32% to reach the 90% of the maximum utility), as well as the total utility achieved by the algorithm (7% higher



(a) Performance of the hybrid simulated annealing with simple marginal probabilistic model in case if the information of maximum desired utility is available.



(b) Effect of applying models, trained on different topologies, as part of the advanced simulated annealing search. The displayed results are for the six node mesh network.

Figure 4.15: Influence of additional information on performance of SA [55].

utility after the 40th iteration). We explain such significant gains by the more aggressive state space exploration that such version of the metaheuristic produces. The fast cooling provoked by the knowledge of the maximum/desired utility leads to more aggressive probing of the state space, i.e., more reheating cycles are initiated during the same amount of state space probings. Furthermore, the knowledge of the maximum utility allows more accurately determining the promising SA neighborhoods for further exploration, which is realized, as described above, though the modification of the mechanism of the acceptance of a new SA state. The results reported here are typical in our experiments. However, there are naturally situations arising (and observed in the simulations as well) where the additional aggressiveness does not always pay off.

4.4.4.7 Effect of Pre-training of Models

We conclude this section by discussing performance gains that can be achieved if the basic version of the simulated annealing is enriched with long-term historical insights provided by different probabilistic models. Care should be taken when applying trained models as part of metaheuristics. Our experiments confirm the intuitive assumption that the more specific is the trained model or the hypothesis, the fewer network conditions it is suitable for. We have trained hypotheses on three different WSN topologies (the eight node linear network, the six node and the eight node meshes). We used these hypotheses to optimize of the network exhibiting one of these topologies (see Figure 4.15b). As expected, the version of the simulated annealing with the models trained on the most different topology (the mesh vs. the linear network) performs even worse than the version of the system with no training at all, especially for the small number of iterations. However, results obtained using the hypothesis from a similar network (another mesh) are notably better leading to higher utility gains in the beginning of the optimization.

The simpler the probabilistic models used for learning the smaller the difference in the results obtained are after a large number of search iterations, as the simple hypotheses are capable of faster re-learning on new data. More sophisticated models converge faster by approximately 15 iterations provided that that sufficient amount of appropriate training data is available. Otherwise, they achieve worse results than the simple models. We observed similar behavior in the VoIP scenario as well. By varying the network load, instead of the topology, we determined that the advanced simulated annealing performs

better with untrained models rather than the trained ones in cases of drastic and frequent changes in network conditions. This, again, underlines the importance of the task of context identification and its further clustering.

4.5 Conclusions

This chapter concentrated on metaheuristic search methods. First, several popular metaheuristics were introduced, as well as a short review of their applications in wireless networking was provided. Then we considered a simplified network planning problem to demonstrate that even out-of-box metaheuristics implementations can be successfully applied to solve wireless optimization problems.

In the second part of this chapter we considered scenarios for both run-time and off-line parameter-based cross-layer optimization. We explored several versions of hybrid simulated annealing that make use of probabilistic models. We observed that, indeed, probabilistic models through exploitation of the long-term history are capable of significantly improving the quality of the search. For example, utilization of the correlation coefficients between parameters and a particular utility function boosted the convergence rate of the system up to 30% for the static optimization, also increasing the utmost utility value reached for the run-time scenario as well. Even the simple probabilistic graphical model almost always increases the convergence rate by 5%-10%. However, due to the fact that the usage of complex probabilistic models heavily alters the search, poor or inappropriate training of these models can lead to the search being stuck into local minima. We observed this effect, for example, on the Bayesian graphical model. Further, we discovered that if we employ a desired, ideal, utility value as an additional input, and utilize it to tune the cooling schedule, as well as to decide on the “length of the exploration jump” then the SA can converge more than 30% faster also improving the overall quality of the solution. Therefore, simple modification to the metaheuristic search may provide improvements comparable to the sophisticated methods.

The modification to the basic versions of the metaheuristics can certainly lead to considerable convergence gains and overall better utilities. However, it is not clear if the received gains would always pay off, as the compared to the benefits obtained by careful problem formulation. The results obtained in this chapter, as well as in Appendix A and Chapter 5, Section 5.3 indicate that the proper problem statement, the abstraction and decomposition of the resulting task, might indeed provide better results. From our point of view it makes sense to do simple adjustments to the basic metaheuristic search techniques and use automatic tuning mechanisms. It might be suboptimal to spend time on developing complex hybrid metaheuristics unless they can benefit from the pre-trained models or they are applied to a very specific scenario such as the online optimization in the absence of context identification and context-aware models/optimizers. Metaheuristics in general are viable for providing a better understanding of the problem. In some cases, e.g., very likely in a joint network planning task, they are the best optimizer candidates. These techniques may be also useful for the initial training of the models.

5

Simplifying Optimization Problems with Graphs and Hidden Semi-Markov Models

In this chapter we focus on simplification of the optimization task through derivation and application of appropriate networking models. In particular, we derive models that are based on *graph abstractions* or *network motifs* for capturing spatial and dynamic principal components of the operational context [56, 57]. We also propose methods for obtaining temporal models based on *Hidden Semi-Markov Models* that are used for online estimation and modeling of ON/OFF network activity patterns and their duty cycles based on observed power spectrum (RSSI) samples [59].

Graph-based models are popular in wireless, especially Wi-Fi, networks, as in a large number of cases they allow quantizing distance-dependent interactions between wireless nodes into a small number of classes. This quantization immediately reduces the dimensionality of the optimization task. We propose to use labeled network motifs as a general class of graph-based models for a wireless environment that consider only a neighborhood of limited size. For example, the popular connectivity, contention, and conflict graphs belong to this class of models [305]. We approximate spatial wireless network models as graphs with multiple edge types, and study the structure of their network motifs. We also use graphs to capture network dynamics imposed by tunable network parameters.

The rest of the chapter can then be attributed to the area the cognitive radio network research. We consider the problem of temporal online modeling of ON/OFF network activity patterns based on power spectrum samples. The resulting models can be utilized as part of the dynamic spectrum access or PHY/MAC protocol optimization. We propose and compare two methods of various complexity that estimate distributions of the spectrum power levels on a channel, in the simplest case distinguishing the “OFF” and “ON” states that correspond to the empty and the busy channels, respectively. The first method is based on Hidden Semi-Markov Models. The other directly operates on marginal power distribution of received signals.

5.1 Introduction

As we have discussed in Chapter 2 modeling crucially influences the formulation and the solution of an optimization task. In this chapter we present our work on the two types of models, namely the graph-based abstractions and hidden Semi-Markov models, that were successfully adapted to compactly represent several principal components of the wireless operational context. Application of the corresponding models is particularly promising in the Wi-Fi networking environment.

In Sections 5.2 and 5.3 we consider directed labeled (multi-dimensional [306]) graphs. In particular, we study the *spatial* aspect of contention graphs, arguing that neighborhoods of a node in wireless, especially Wi-Fi, networks can be modeled as network motifs without losing the accuracy of spatial models. Our numerical experiments show that in

reality there exist only small number of network motifs, small subgraphs, that reoccur frequently [57]. Therefore, when developing distributed optimization algorithms, e.g., MAC protocols, that require only localized information one can mostly consider just a limited subset of graphs, which considerably simplifies development of the respective solutions.

In addition to geometric network structure we also use labeled graphs to approximate *network dynamics* imposed by tuning of network parameters [56]. We show that a simple linearization of network performance functionals results in natural correlation metrics for influence of parameter changes on performance. Based on extensive simulations we demonstrate that these correlation metrics have a high degree of robustness against perturbations in node locations, leading to simple graph structures. We also discuss the relationships between the arising graph approximations of network dynamics and the commonly applied connectivity and conflict graphs. Our results indicate that the labeled network motifs formed by approximating dynamics combined with the connectivity structures are promising candidates for autonomous context identification, thus enabling faster and robust wireless network optimization.

In Section 5.4 we aim at online estimation and modeling of ON/OFF network activity patterns and their duty cycles based on observed power spectrum (RSSI) samples [59]. These models are used to improve dynamic spectrum access strategies, i.e., they allow to identify and predict transmission opportunities at different frequencies in the scenarios of network co-existence and primary-secondary spectrum usage. This information can be also used to optimize configuration PHY/MAC protocol layers, e.g., to adjust sensitivity thresholds or transmission powers. The study is done on empirical data gathered indoors using WARP SDR boards [307] or TelosB wireless sensor nodes [58]. We consider two mechanisms for online modeling and further signal classification. One employs Hidden Semi-Markov Models (HSMM) and the Viterbi algorithm. The other is based on the empirical marginal distribution of the received readings. Our results show the high accuracy and relatively fast training of the HSMMs at the price of considerable complexity of these models, especially if multiple power levels, with some being close to the noise floor, are to be identified. The density-based models are simpler, but they have applicability limitations, and tend to be inaccurate for complex scenarios, and more vulnerable to inappropriate initialization. In Chapter 6 we further apply online the HSMM/Viterbi approach for improvement of a spectrum-agile MAC protocol as part of the self-optimization system for home wireless networks.

5.2 Connectivity Graphs and Spatial Wireless Network Models

Graph-based models are utilized for a number of networking problems, ranging from routing and load balancing in fixed networks to channel allocation and scheduling problems in wireless systems. While for fixed networks graph models are, in a sense, exact, for wireless networks they should be considered as an approximation. The connectivity structure of wireless networks is of the *spatial* nature [57, 222, 308, 309], and performance of these networks fundamentally depends on dynamically changing distance and fading environment. However, spatial models can become quite complex, and typically require computationally demanding processing to extract information from them. It is desirable to find simpler computationally efficient analogs that both reflect network connectivity aspects, as well as portray other metrics of interests at multiple protocols layers. As an alternative, computationally effective *graph approximations* of wireless network structures have been developed, as it has been observed that often actual performance characteristics

depend only loosely on node locations.

Graphs are popular models for capturing structure and dynamics of wireless networks at different protocols layers. There is an extensive prior work on application of graph-based models in the wireless domain. Much of the work in the literature has a limitation of mostly being based on models that utilize random graphs, or graphs constructed as solution to specific performance optimization problems [14, 310–314]. Small network motifs representing localized connectivity information and their use for network optimization have been discussed in [298, 315–318]. The multi-dimensional aspect of graph approximations for wireless networks is being only limitedly studied, e.g., in [306, 319], mostly again as solutions to very specific types of problems. More practically, passive estimation of conflict graphs was explored in [320]. Other works, e.g., [321–324], discuss on power and rate adaptation in order to practically address hidden/exposed terminal problems, as well as to avoid "gray zones". Thus, these works limitedly concern the network dynamics. However, only low-level key-performance indications and raw SINR values are used for parameter adaptations. The latter two references also provide example usage of conflict graphs with multiple channels or directional antennas. However, they only exploit conflict graphs, and do not generalize nor embed the knowledge on optimization into them. Summarizing, the referenced research on graph-based modeling of wireless networks is related to either a) connectivity and contention representation, b) consideration of local neighborhoods, and/or c) capturing of network dynamics imposed by tunable network parameters[1]. We propose to unite all these types of graph approximations as *directed labeled network motifs*. Having a common class for these models allows us to generalize on the wireless graph approximation research and employ the achieved results in a systematic manner. Directed labeled network motifs with low number of states belong to the class of simplest models. If constructed appropriately they can be directly applied instead of full spatial models for similar applications, but with significantly reduced complexity. The price to be paid is some loss in accuracy of a model, and therefore, sometimes, sub-optimality of an optimization decision.

The directed labeled network motifs can benefit from two characteristics. First, the vertices and edges of these graphs should have a limited, relatively small, number of states. Second, only small subgraphs of the full graph reflecting the spatial and, possibly, dynamic structure of a wireless network should be considered. Therefore, the network motifs do not have a large number of vertexes and edges. These models aim to capture just primary, most influential, network properties [298, 315–317], which makes them *robust* with respect to measurement and modeling errors. Similar observations, arguing on the role such small subgraphs in arising dynamics, were also made in other research fields [325]. Of course, nothing stops us from increasing granularity and complexity of graph states, as well as their size. However, this might lead not only to higher computational overhead and rapidly increasing complexity, but also to modeling inaccuracies can result in sub-optimal optimization decisions and even emergent behavior. In the next two sections we study and argue on usefulness of the labeled network motifs. Our results strongly indicate feasibility of such an abstraction at least for the CSMA/CA networks.

Models based on graphs, though appealing, have a deficiency of abstracting away the broadcast property of networks. It is easier to forget that we operate in wireless environment if we use graphs (which might not be bad in itself, and can lead to generality

[1] To the best of our knowledge the capturing of network dynamic aspect in wireless graph-based modeling is not widely studied, and we can claim for novelty in introducing this consideration through parameter correlation metrics [56].

of solutions developed for both wired and wireless networks). It is also somewhat less natural to consider on graphs the solutions that exploit the broadcast property of the medium, such as network coding, multicasting, etc. There exist several alternatives to graph-based abstractions, for examples, hypergraphs and Voronoi diagrams [326–329]. However, it is more difficult to operate on these abstractions, and they are more computationally demanding. Overall, further research on applicability of these models for particular wireless network problems is desired.

Overall, one may consider that spatial objects and simple graphs are are *two extremes* of multi-dimensional network connectivity and dynamics modeling. As discussed in Chapter 2 the particular choice of the appropriate model is dictated by the optimization problem itself, i.e., the trade-off between accuracy of a model, its usability, and the imposed overhead.

Further in this section, after the focus of this chapter on simplification on an optimization task, we consider simplification of spatial network models with graph-based representation. In particular, we focus on the *structure of the connectivity graphs* arising after deployments of wireless network nodes, which positions are determined by particular spatial wireless network models. We study neighborhood graph structures with three edge types that typically arise as outcomes of different spatial network models, especially the *Geyer model* fitted to the real outdoor Wi-Fi deployment [330]. This differs our work to the classical research in the areas of *random geometric graphs* [331] (some of which are referred to as *unit disk graphs*), which is quite limited in our context beyond the unrealistic assumption on uniformly random distribution of nodes with binary labeling of their edges. For graph characterization we utilize two metrics. The degree distribution metric often provides good first-order approximation of the performance characteristics of a wireless network, e.g., with respect to the likelihood of encountering contention. We also consider the small network motifs [325] for a typical node of interest.

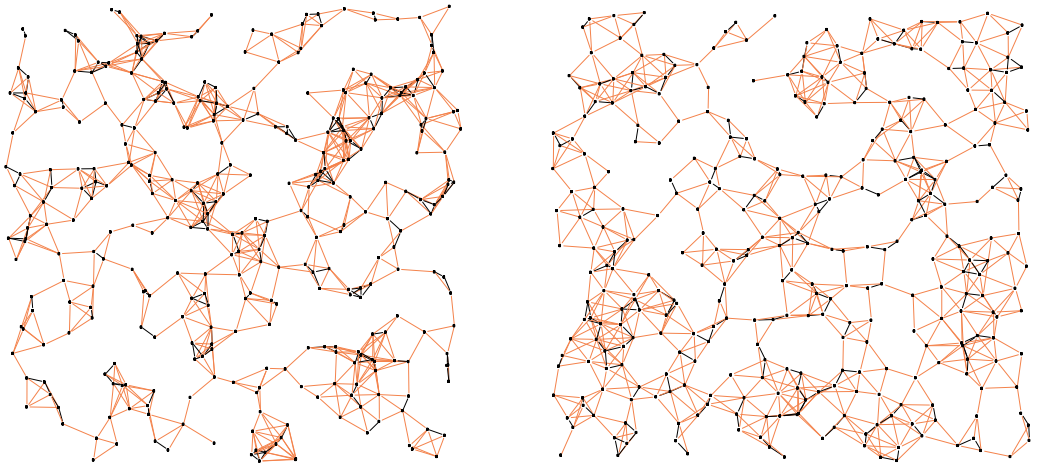
5.2.1 Considered Models of Spatial Network Structure

Spatial wireless network models are based on *stochastic geometry of point processes* [332, 333], and, typically, deal with distribution of node locations in some region (see Table 5.1). The simplest and the most popular of these models are based on the *homogeneous Poisson point process*. Though these models are easy to use for analytical calculations, the distribution of nodes generated after this type of process is unrealistic with respect to real wireless deployments, as the latter differ significantly from the uniformly random nature of the Poisson model, see [24, 334, 335]. Network nodes tend to have either more *clustered* or *regular* positioning. Clustering is dictated by natural human groupings or technical constraints, such transmission range of an AP. Infrastructure planning, e.g., in cellular systems, makes the nodes to be positioned more regularly. Riihijärvi and Mähönen in [222] have suggested to employ *Geyer saturation process* that is a type of Gibbs point process to accommodate for both clustering and regular aspects of node positioning, which led to the good fits to the empirical data. The Geyer spatial wireless network model was successfully shown to fit well different empirical datasets of wireless node positioning. Adjustment of its parameters can also approximate this model to the poisson, the clustered (e.g., based on Thomas point process), and the regular (e.g., based on Simple Sequential Inhibition (SSI) process) spatial models. The overview of the point processes relevant to the current study is given in Table 5.1. Figure 5.1 also shows sample deployment locations of network nodes resulting from application of the Poisson

Table 5.1: Definition of selected point processes.

Point process	Explanation
General informal definition	A point process N can be informally defined as a random collection of n locations $\{x_i\}_{i=1}^{n} \in D^n$, where both the individual locations $x_i \in D$ and their total number $n \in \mathbb{N}$ can be random.
Homogeneous Poisson	For this model the number of nodes $N(A)$ in a region $A \subseteq D$ follows a Poisson distribution with parameter $\nu\|A\|$, where $\|A\|$ denotes the area of A, and $\nu > 0$ is called the *density* of the process. The point counts $N(A_1)$ and $N(A_2)$ are independent random variables, in case regions A_1 and A_2 are disjoint.
Simple Sequential Inhibition (SSI)	The SSI depends on two parameters, the total number of points $n = \nu\|D\|$, and a *hard core radius* r which defines a minimum allowed separation between the points. The locations are generated by throwing points one at a time uniformly on D, while enforcing the condition that no two points can be closer than distance r apart. The process is stopped either when all n points have been placed, or their density exceeds the one allowed by r.
Gibbs	A Gibbs point process is defined by prescribing a density function $f(\boldsymbol{x})$, which assigns to each point configuration $\{x_i\}_{i=1}^{n} \in D^n$ a likelihood of that configuration to arise compared to the case of a homogeneous Poisson point process. In particular, if $f(\boldsymbol{x}) = 0$ the corresponding point configuration can never occur. If $0 < f(\boldsymbol{x}) < 1$ the given configuration can occur but is less likely to do so than in the Poisson case. Finally, if $f(\boldsymbol{x}) > 1$ the given configuration is more likely to occur than for the Poisson process.
Geyer	Geyer saturation process is a type of the Gibbs point process. Its each point x_i of the pattern $\{x_i\}_{i=1}^{n}$ contributes a factor of $\beta\gamma^{\min\{\zeta, s_r(x_i,\boldsymbol{x})\}}$ to the density. Here $\beta > 0$ controls the density of the points, $\gamma > 0$ the type of interaction (whether clustered or regular), $s_r(x_i,\boldsymbol{x})$ denotes the number of points in $\boldsymbol{x}$ that are closer to x_i than distance r, and ζ is the saturation threshold bounding the contribution of the individual points and ensuring that the arising density is integrable. The Geyer model is a special type of the general Gibbs models that suits well for our purposes, as it provides both a fair amount of flexibility, and accurate fits to the empirical data. It is also analytically tractable.

and the Greyer point processes. In the rest of the section we base our work primarily on fitted Geyer models as a representative example of a realistic spatial wireless network model. We also analyze node deployments after the Poisson and the SSI models as they are classical in the community, and allow to get an insight on the influence of regularity on network performance.



(a) Poisson spatial wireless network model (b) Geyer spatial wireless network model

Figure 5.1: Sample network node positioning as result of application of Poisson and Geyer processes. The spatial models are approximated as graphs. Nodes that can communicate are linked by black edges, if they can sense each other the edge color is orange.

5.2.2 Network Motifs as Approximation of Spatial Models

In this section we consider connectivity graph models. The connectivity models and their derivatives, contention, conflict or interference graphs model various aspects the connectivity and the awareness structure of the wireless nodes. Often the edges of the respective graphs are labeled to encode if two nodes can communicate, perform well on application layer[2], or only sense each others' transmissions (the latter for IEEE 802.11 networks corresponds to carrier sensing). These graphs have been widely employed as relatively low-overhead easy-to-use models for wireless network optimizations, which employ such methods as scheduling, network coding, frequency allocation, congestion and contention control [92, 298, 305, 315–317, 320, 322, 336, 337]. The connectivity graphs constructed after such a labeling determine to a large extent the behavior of a network provided that the relevant protocol parameter settings stay the same and external environmental conditions do not drastically change. For our study we consider such a case, with connectivity graphs determined only by particular underlying spatial network structures. Therefore, in the context of this work graph approximation are used to construct connectivity graphs based on point processes introduced above.

We formalize *graph approximation* as a function that maps the point process N to the connectivity graph G_N as follows[3]. The vertex set of G_N corresponds the points in the spatial network model N. There are $|d_k|$ edge label values, which encode connectivity, sensing and, if needed, other types of interactions between nodes. In this work an edge can be assigned two values, either "connectivity" or "sensing". These values are set to be

[2]There exist a certain ambiguity on the definition of this range as the respective quantization threshold is often set not on the node's ability to exchange signals reliably on the PHY level, but it denotes acceptable levels of performance on the MAC or even the application layers. This makes connectivity graphs similar to the ones built based on the application performance in the line-of-sight propagation conditions.

[3]The same formalization applies to the next Section, where we consider connectivity graphs as a possible explanation to boundaries, context, of observed variations in parameter-KPI relations.

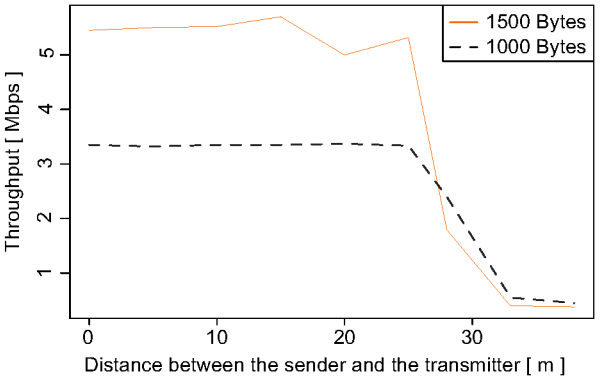


Figure 5.2: Throughput of a UDP connection over a IEEE 802.11b Wi-Fi link as a function of distance between the transmitter and the receiver for two different frame sizes (empirical results, adapted from [57, 338]). The figure displays three operational zones.

threshold-dependent on distances between the nodes. Therefore, the edge between two vertices x_i and x_j is then assigned a label of the integer type $(n+1)-\sum_{k=1}^{n}\mathbf{1}(\|x_i-x_j\|\leq d_k)$, where $\mathbf{1}(\cdot)$ is the indicator function, $\|x_i-x_j\|$ denotes the distance between x_i and x_j. The label value of $n+1$ indicates that x_i and x_j are not adjacent in the arising graph and no edge is drawn between them, i.e., the respective nodes do not interact in any way. This formalization is a generalization of the definition of a random geometric graph [331], which typically assumes the underlying Poisson point process and one edge type[4].

Our graph approximation relies on the assumption that there are no significant performance and connectivity dependencies between relative positions of the nodes as far as the distance between them stays the same. In most cases especially where channel conditions have a strong line-of-sight components this assumption holds (see Figure 5.2). In selected scenarios [321] it is more appropriate not to abstract away, but separately label the so-called "gray zone" that denotes unstable, rapidly decreasing performance as an additional edge type of the graph. In this section we do not take into the consideration that nodes may be equipped with directional antennas, or propagation conditions might significantly differ depending on where nodes are place and what separates them. For scenarios, where these assumptions do not hold with sufficient accuracy, other statistics has to be used additionally, for example the SNR/SINR metric [339, 340], which is further explored in the Appendixes D and E.

5.2.3 Scenario Setup

Next we analyze the structure of network motifs of connectivity graphs as an approximation for the spatial wireless network models discussed above. We focus on graph characteristics that are known to have a heavy impact on the performance of the modeled networks [298]. We also show that the graph-based models depict the differences in network structures created after different point processes.

In this work we consider networks mimicking the *indoor* and *outdoor* Wi-Fi access point deployments. We consider three, Poisson, SSI and Geyer point processes. Table 5.2

[4] In wireless networking literature random geometric graphs are also sometimes called *unit disk graphs* and *quasi unit disk graphs*, in case there is probability assigned to the edge existence.

Scenario	Model	Parameters
Outdoor Wi-Fi	Poisson	$\nu = 1.35 \times 10^{-5}$
	SSI	$\nu = 1.35 \times 10^{-5}$, $r = 100$
	Geyer	$\beta = 3.3 \times 10^{-5}$, $\gamma = 0.4$, $\zeta = 2$, $r = 150$
Indoor Wi-Fi	Poisson	$\nu = 2.7 \times 10^{-4}$
	SSI	$\nu = 2.7 \times 10^{-4}$, $r = 30$
	Geyer	$\beta = 6.8 \times 10^{-4}$, $\gamma = 0.65$, $\zeta = 2$, $r = 80$

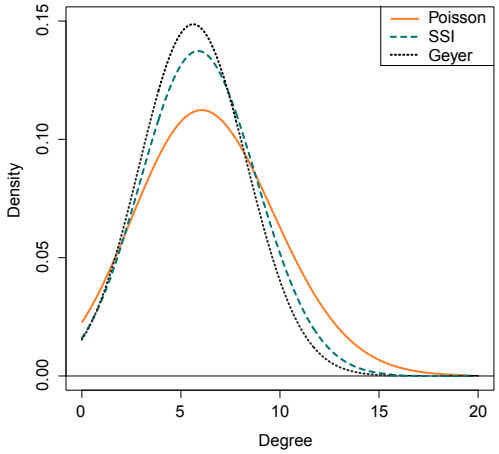
Table 5.2: The considered scenarios and corresponding models and parameter values.

shows parameters of the respective spacial wireless models for each of the scenarios. For us the Geyer model for the outdoor scenario is of key interest, as it fits accurately to the empirical data of the Google Mountain View Wi-Fi network deployment [222]. Parameters of other models are correspondingly adjusted after the Geyer model to yield the same overall network density and otherwise similar spatial structure. The indoor scenario differs from the outdoor one by denser node deployment. As said in this section we consider the network motifs that corresponds to the connectivity graphs and have only two types of edges, which correspond to distances of $(d_1, d_2) = (200\,\text{m}, 400\,\text{m})$ for the Outdoor Wi-Fi scenario, and $(d_1, d_2) = (50\,\text{m}, 100\,\text{m})$ for the Indoor Wi-Fi scenario. The results discussed below are obtained for 10^4 Monte Carlo simulation runs, with generated networks having a node in the center of the simulated area to avoid edge effects. We mostly consider this central, "typical", node that formally corresponds to the estimates of the functionals of the Palm distribution of the underlying point process [332, 333]. Further we analyze the degree distributions of this typical node, and also study the occurrence of different three- and four-node network motifs, where this node participates. We use the `nauty` toolkit [209, 341] to classify these motifs into their respective isomorphism classes (i.e., compare these graphs with respect to their connectivity and edge types), since only those have impact on the network dynamics. We used `R` as a tool for generation of network spatial models, graph-based approximation and their processing [342].

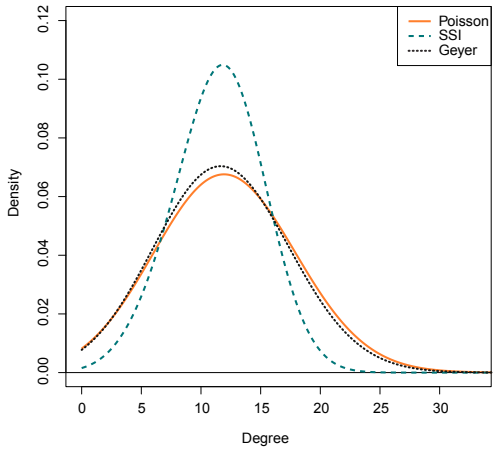
5.2.4 Results

First we analyze the degree distributions of the obtained graph approximations shown in Figure 5.3[5]. As we observe, the differences between the degree distributions of the graphs constructed after the studied spatial models is significant. The degree distributions after the Geyer model for the Outdoor Wi-Fi scenario differs considerably from the distributions derived from both the SSI and the Poisson models. The Geyer-based graphs in this scenario have the highest edge concentration, whereas the Poisson-based models have the most edge variability (see Figures 5.3a,c,d). For the individual degree type distributions the SSI models becomes closer to the Geyer models. This could be explained by the fact that outdoor deployments tend to have quite regular, planned, structure. In the

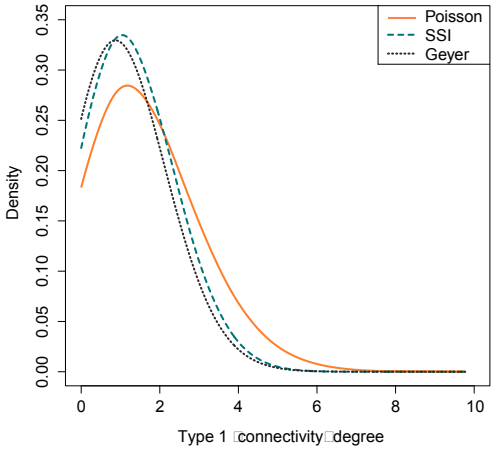
[5]The degree distribution is defined on the integers. However, we show here smoothed kernel density estimates so that it is easier to observe the differences between the degree distributions arising for the considered models.



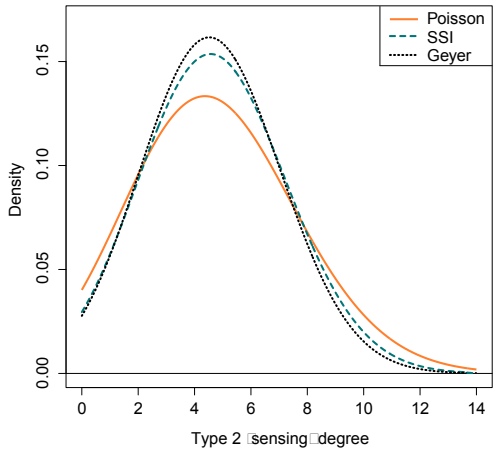
(a) The overall degree distribution for the Outdoor Wi-Fi scenario.



(b) The overall degree distribution for the Indoor Wi-Fi scenario.



(c) Distribution of type one "connectivity" degrees for the Outdoor Wi-Fi scenario.



(d) Distribution of type two "sensing" degrees for the Outdoor Wi-Fi scenario.

Figure 5.3: Sample degree distributions of connectivity graphs based on various spatial wireless network models [57].

indoor scenario, on the contrary, the degree distribution of the graphs based on Geyer and Poisson models are similar, as for these networks, the planning component is, typically, smaller and the concentrations of network nodes is higher (see Figure 5.3b).

The degree distribution of a connectivity graph forms an important metric. For example, it indicates how likely it is for wireless nodes to experience contention and gain access to a channel. However, this metric is not sufficient to deduce, for instance, the probability of a hidden terminal problem to arise in a network, i.e., it does not contain information on *interactions between individual nodes*. These kinds of results could be obtained from analysis of network motifs, essentially small subgraphs, that efficiently capture information on the neighborhood of a "typical" node (a node of interest). Therefore, we further study the occurrence of three- and four-network motifs for these nodes. We concentrate on such a small motif sizes, because, as was discussed in the beginning of the chapter, analysis of small subgraphs is already likely to cover most of the conflict cases

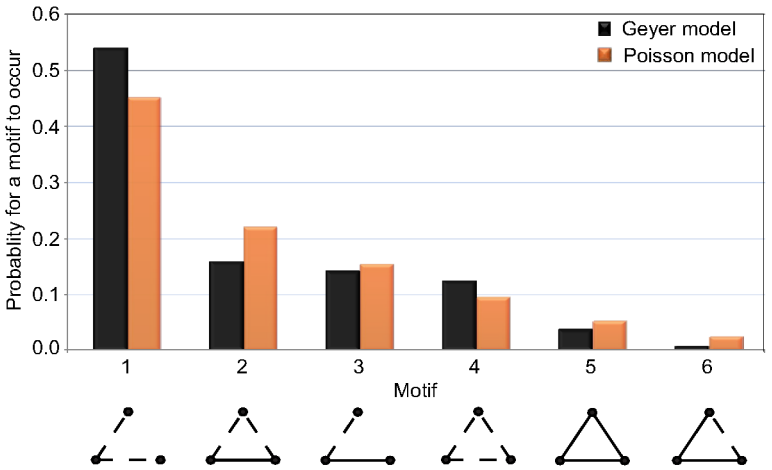
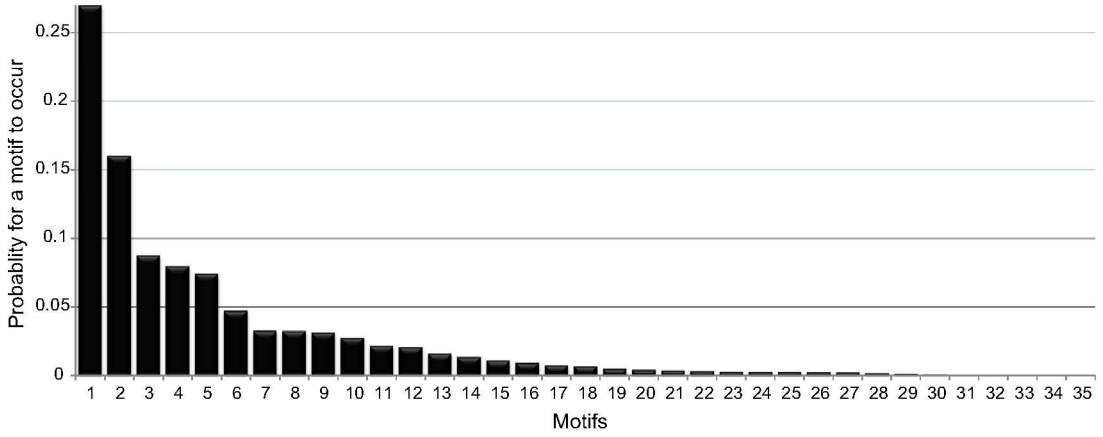


Figure 5.4: Histograms for the Geyer and Poisson models for the occurrence of three-node network motifs for the Outdoor Wi-Fi scenario. The illustration of the motifs is given as well. Black solid lines denote "connectivity" edges, and dashed red line correspond to "sensing" edges [57].

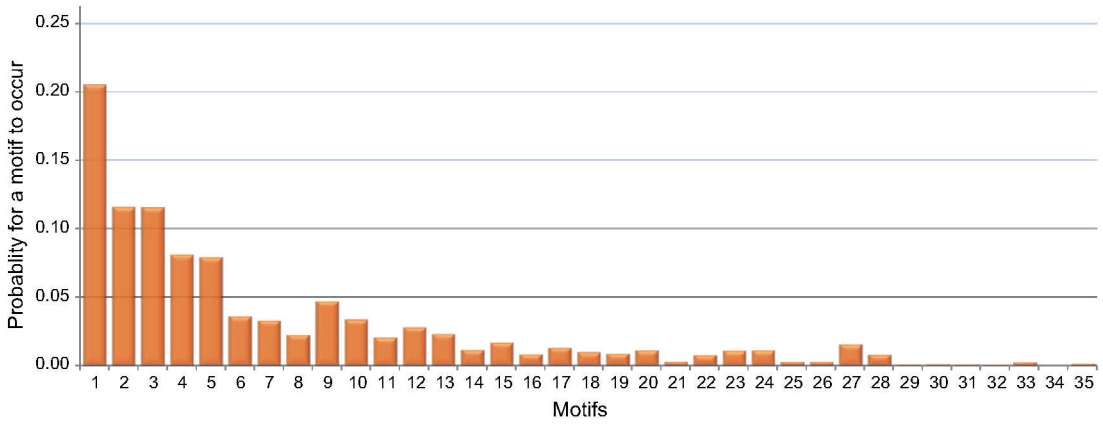
that can arise in a network, which require unique mitigation actions. Normally, there is an exponentially diminishing probability that with larger subgraphs we will encounter a situation that cannot be perceived as a combination of smaller motifs, and efficiently dealt with based on the analysis of those.

Here we analyze motifs arising from Poisson and Geyer spatial network models for the outdoor Wi-Fi scenario (see Figures 5.4 and 5.5). We consider this to be a representative example, as, first, for this scenario we operate on models that fit well to the empirical data. Second, these two spatial wireless models display the most diverse behavior with regard to the degree distribution metric. Thus, they can be considered as two extremes between which network motifs patterns after spacial models will reside. As a metric characterizing network motifs we consider the probability of occurrence of different type of the subgraphs that are summarized in the form of the rank-frequency plots. Figure 5.4 shows the probability of occurrence of the three-node network motifs. Out of seven possible network motifs six are actually encountered. The seventh subgraph that corresponds to the hidden terminal case is never generated due to the specific choice of distance thresholds that define the values of the edge labels ("connectivity" or "(only) sensing"). In the future we plan to experiment with these thresholds to see how distribution of motifs gets affected due to changes in labeling. Currently, we do not expect that changes in these thresholds, if they are reasonable, will significantly affect the major results that are the highly diverse frequency of appearance of the network motifs, and a noticeable dissimilarity of these patterns between the Poisson and the Geyer models.

Figure 5.5 that displays the results for the four-node motifs confirms these conclusions. For the four-node subgraphs only 35 motifs are encountered, which is slightly over half of all the possible ones [317]. Only nine of these occur with more than 3% probability. We also observe that some of the motifs, e.g., #9 and #12, have significantly different occurrence probability with respect to the underlying spatial models. Overall, these results support the assumption that in practice many of the wireless network deployments and solutions can to be built primarily considering behavior of a small subset of the theoretically possible network configurations.



(a) The probabilities of different four-node motifs arising for Geyer spatial wireless network models.



(b) The probabilities of different four-node motifs arising for Poisson spatial wireless network models.

Figure 5.5: Histograms for the Geyer and Poisson models for the occurrence of four-node network motifs for the Outdoor Wi-Fi scenario. The numbers on the X-axis denote different non-isomorphic graphs (motifs).

5.2.5 Short Summary

In this section we considered the problem of dimensionality reduction of models used as part the optimization task that concern PHY/MAC-level connectivity and interference of wireless nodes. We used connectivity graphs as an approximation of different spacial wireless network models, including a realistic Geyer model for Wi-Fi APs distribution. We discovered that of all the possible localized network motifs that encode interaction patterns, only a small portion of subgraphs arise frequently. We also observed that the underlying spatial models influence the occurrence of different motifs, and their probability distributions change significantly for different spacial node structures. Our results confirm the earlier claims that the graph-based approximations are efficient models that allow decoupling the performance evaluation of a protocol design for a wireless network from the underlying spatial model. Moreover, they indicate that, in principle, only a small portions of these have to be studied for a given node neighborhood to develop and/or evaluate with high degree of trustworthiness a certain network solution, thus, further significantly reducing the complexity of such a task. Though our study was made based

only on the connectivity graphs, we believe that the similar conclusions would apply to all localized network motifs based on distance-dependent KPIs.

5.3 Capturing Protocol Parameter-based Dynamics with Graphs

In this section we explore how graph approximations can be used to capture *network dynamics* induced by changes in tunable network parameters. We argue that *parameter-performance correlation metrics* obtained through simple linearization of network performance are appropriate for this goal. We support our arguments by providing results of extensive simulations that indicate the validity of our approach. They also show that connectivity graphs combined with the parameter correlation metrics are valuable to context identification, and can lead to effective and robust autonomous network optimization.

As discussed above, the currently applied graph approximations are almost universally based on the concepts of connectivity and conflict graphs [305]. These can be thought of as encoding *constraints* how resources can be used in the network. For example, for CSMA-based networks contention relations expressed through a conflict graph dictate which configurations of transmitters can be active at the same time. The accuracy of such models relies on the observation that radio propagation in typical Wi-Fi deployments often feature strong large line-of-sight components, and the "gray" area, where performance is unstable and rapidly decreases with distance, is very narrow and can often be neglected. In this sense the contention relations are *robust* with regards to small changes in distances between the nodes. It is precisely the robustness property that enables graph-based approximation of the relationships between wireless nodes. This feature is further illustrated in Figure 5.6 that displays the performance of a point-to-point link as a function of the distance between the nodes for different wireless channel types. As mentioned above, we see that in the absence of fading, or when the line-of-sight (LOS) component is strong, the dependency of performance on the distance is weak, and it can be quantized only in three states. However, in non-line-of-sight conditions this no longer completely holds, and the graph approximation with small number of edge types might become inaccurate. One should note that such situations can be prevalent for many present non Wi-Fi systems. Even in the case of MIMO enabled Wi-Fi we expect that small-scale fading will play more dominant role that needs to be accounted for in graph-based approaches.

In this section we suggest a different approach to graph-based wireless network approximation that, among other features, enables robust quantization with smaller number of states for more complex operational conditions. We focus on *network dynamics* as opposed to simple structural relations between the nodes. Classical contention and conflict graphs are typically used in the literature to study the expected performance of a particular protocol or a protocol stack with *fixed* parameter settings. We focus instead on the impact different tunable parameters of the protocol stack for a given *context*, which might be defined by a number of variables such as node positions, propagation environment, application behavior, network optimization goals, etc. We aim to describe these in a similarly robust manner as has been done for contention relations in the literature. We argue based on classical dynamical systems theory that various linearizations of network dynamics should naturally yield robust metrics related to statistical correlations between performance and parameter values. Based on extensive simulations we show that this is indeed the case. We believe that such approximations of network dynamics have a number of interesting applications. As an example, they enable fast and robust context-aware network optimization based on observed properties of connectivity graphs. Next, we

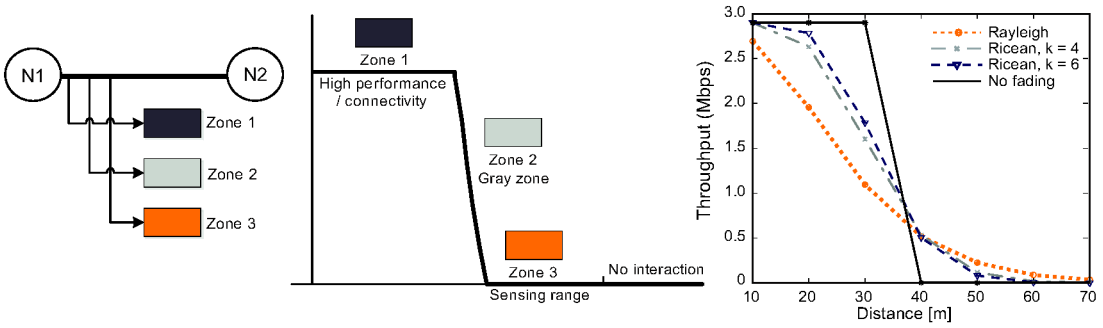


Figure 5.6: Possible labeling of the graph edge depending on observed performance [56]. The left and the central panels consider the example of the three distinct performance behaviors being observed as in the case for channels with the strong line-of-sight propagation component. The panel on the right shows dependency of performance on distance between two wireless nodes also for other channel conditions.

discuss on the connection of the network optimization problem with dynamical systems techniques. We also describe our simulation setup, and present the obtained results.

5.3.1 *Network Optimization and Models of Dynamics*

We consider a classical utility-based optimization framework for network performance [15]. We denote by U the overall network utility, in practice defining the global optimization target as a statistical average from which random environmental contributions such as fading have been averaged out. In most scenarios U would be expressed as a sum of individual utility functions for different data flows or users, but this level of detail is not needed here. Now, in a given environment, and for a given set of node locations, U is a function of a collection of *configurable* parameters $\boldsymbol{p}$, i.e., $U = U(p_1, \ldots, p_n)$. These parameters include, for example, transmit power, use of RTS/CTS mechanism, transport protocol settings, and so on. Then around any given operating point $\boldsymbol{p}_0$, the dynamics of U under changes in the parameter values can be expanded as Taylor series following standard arguments used in control theory and theory of dynamical systems. Formally this is done as follows. Let $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n) \in \mathbb{Z}^n$ be a multi-index with standard definitions $|\boldsymbol{\alpha}| \equiv \sum_i \alpha_i$, $\boldsymbol{\alpha}! \equiv \alpha_1! \cdots \alpha_n!$, $\boldsymbol{p}^{\boldsymbol{\alpha}} \equiv p_1^{\alpha_1} \cdots p_n^{\alpha_n}$ and

$$D^{\boldsymbol{\alpha}} f(\boldsymbol{x}) \equiv \frac{\partial^{|\boldsymbol{\alpha}|} f}{\partial x_1^{\alpha_1} \cdots \partial x_n^{\alpha_n}}. \tag{5.1}$$

Then the Taylor series reads

$$U(\boldsymbol{p}) = \sum_{|\boldsymbol{\alpha}|=0}^{\infty} \frac{D^{\boldsymbol{\alpha}} U(\boldsymbol{p}_0)}{\boldsymbol{\alpha}!} (\boldsymbol{p} - \boldsymbol{p}_0)^{\boldsymbol{\alpha}}. \tag{5.2}$$

Usually the highest degree of robustness in optimization and control is achieved by using the linear terms as a basis for optimization decisions (i.e., $|\boldsymbol{\alpha}| = 1$). As the result (after normalization and appropriate choice in the range of parameter values considered) we obtain terms closely related to *statistical correlation coefficients* between U and p_i.

The application of the correlation coefficients can be understood as follows. Suppose the network is observed to perform poorly, measured as a low overall network utility. In

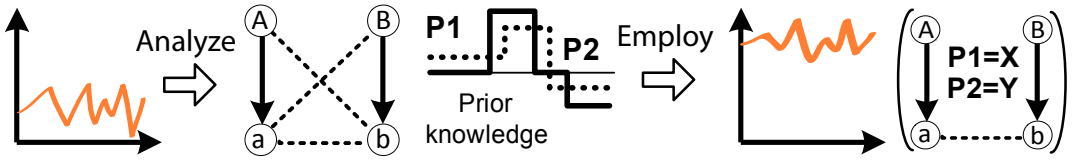


Figure 5.7: Illustration of the problem statement [56]. In which context we do operate (e.g., with respect to the current connectivity graph) and how it can be optimized (e.g., by tuning protocol parameter settings)?

order to improve this performance, complete optimization problem could be solved at network configuration, looking for the global optimum of $U(\boldsymbol{p})$. This can, however, be very costly due to the large number of parameter combinations and the typically complex global structure of the optimization problem. An intermediate choice is to respond to performance degradation by tuning only few parameters, perhaps even one at a time, based on the above-shown linear approximation and conflict/contention graphs identified for the given operational point. This process is illustrated in Figure 5.7. We determine the current operational context either through direct observation of selected Key Performance Indicators (KPIs) or by obtaining a connectivity graph [320]. For this context, as result of prior training or modeling, we have the estimates of the correlation coefficients. Parameters with large correlation coefficients would be the ones to be adjusted first as they are likely to lead to the fastest performance improvement. Others may be tuned later if desired. The parameter adjustments are likely either to change the underlying connectivity graph, e.g., avoid the hidden terminal problem by lowering transmit power, or alter the packet dynamics by, for instance, enabling the RTS/CTS mechanism. The suggested models or rules are useful if the correlation coefficients are robust against small changes in network configuration, and the optimality gap between such a simplified optimization scheme and the global optimum is small enough. This turns out to often be the case, as we show below based on our simulations.

5.3.2 Simulation Setup and Results

Our simulation scenario together with a more detailed graph-based view of the performance evaluation problem is illustrated in Figure 5.8. We consider two pairs of nodes, and the distance between the transmitters increases throughout the simulation time. The nodes communicate using IEEE 802.11a Wi-Fi. We systematically study the performance of the network by employing the QualNet commercial packet level network simulator [291]. The overall goodput as a function of all combinations of parameter and simulation settings is used as the main performance metric. We derive the correlation coefficients between parameter values and throughput as discussed above. We also record the performance gains that can be obtained by tuning different combinations of parameters from their default values. We consider the following changes in the simulation scenario, line-of-sight (LOS) propagation conditions and channels with the Rayleigh fading model, TCP and UDP traffic (FTP and CBR (Constant Bit Rate) applications, respectively), internode distances between transmitters (between 2 m and 202 m), with receivers being separated by 50 m. The tunable parameters are transmit power {10, 15, 20 dBm}, datarate {36, 54 Mbps}, MAC maximum retry limit {1, 4, 7}, use of the RTS/CTS mechanism, maximum contention window {511, 1023, 2047, 4080}, and maximum fragment size {500, 1000, 1500 bytes}.

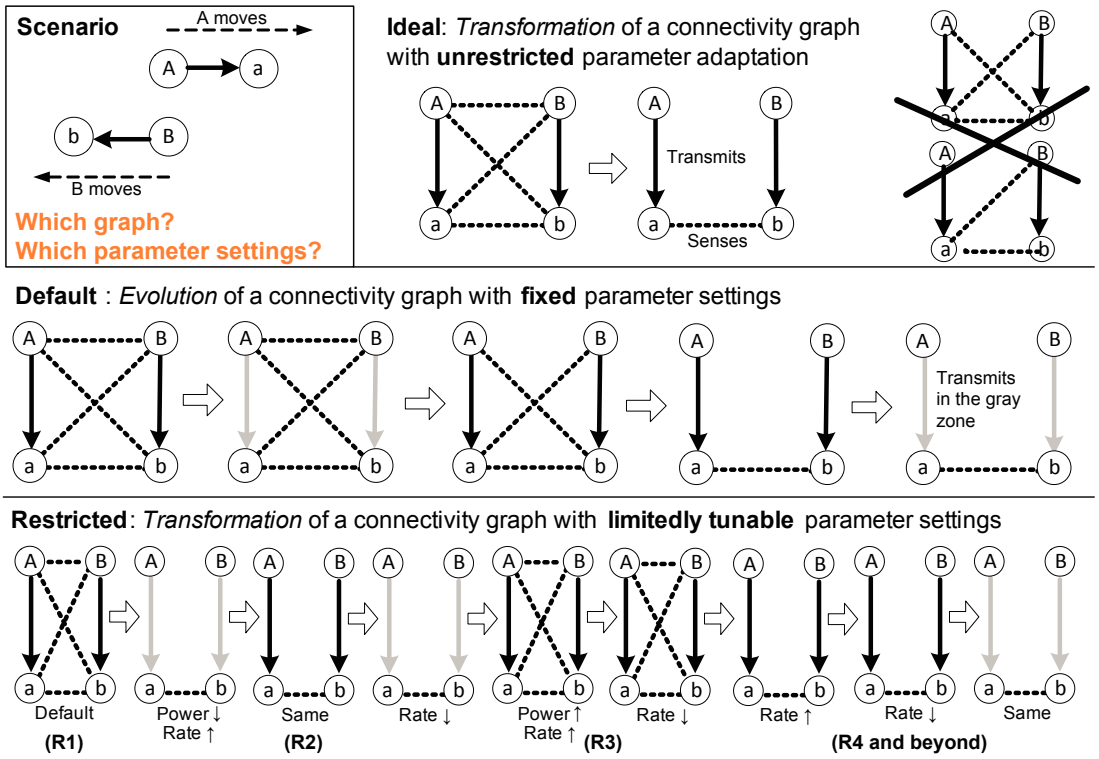
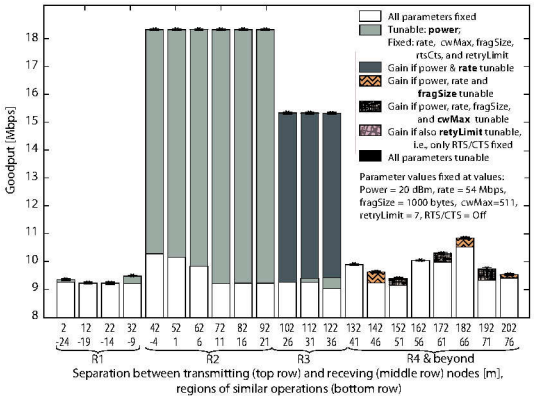
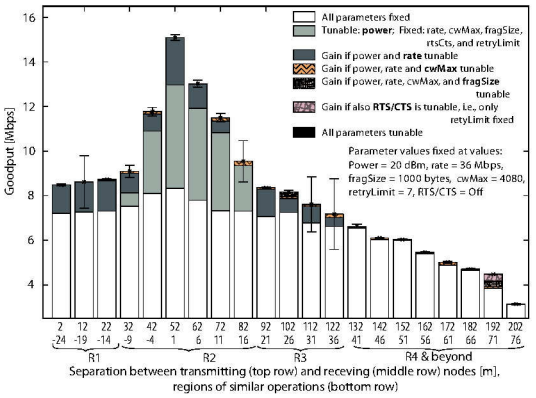


Figure 5.8: Scenario overview. Two transmitting nodes are horizontally moving away from each other. The connectivity graph evolution, which is typical for such movement with fixed parameter settings, is shown in the top panel. The lower panel displays the graph transformation if parameter setting can be limitedly tuned. Here the parameter adjustments allow avoiding hidden terminal (HT) problem and selected gray zones, for which poor performance is observed. (The R labels denote distinct operational regions that display distinct performance dynamics as shown in Figure 5.9.)

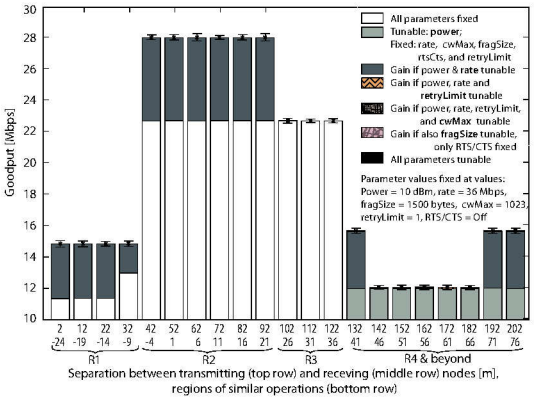
Let us first look at the overall performance numbers. Figure 5.9 illustrates the impact of changes in parameter values on the range of considered scenarios that include UDP and TCP traffic in line-of-sight and Rayleigh fading channel conditions. From the results we, first, see that for all the scenarios the default settings that maximize the total performance over the whole range of distances are different. The contributions to the achieved performance from tuning the parameters also vary, both with the regard to increase in the performance, and feasibility of adjustment of a certain parameter. However, we can also clearly see that the network performance is maximized with *very few* parameters. In vast majority of cases changes it is the transmit power that has the largest impact on achieved performance. The datarate change is another important aspect. The large impact of these parameters is explainable as they influence the network connectivity structure. For example in the operational region $R2$ decrease in transmit power changes the network connectivity structure so that the node pairs do not interfere each other. When distance between the transmitter and the receiver grows ($R3$) this connectivity configuration can be preserved by reducing the datarate parameters, which subsequently lowers the sensitivity threshold. For Figure 5.9d the large effect of the retry limit parameter as opposite to the datarate can be explained by the utility metric we are using for UDP traffic, that



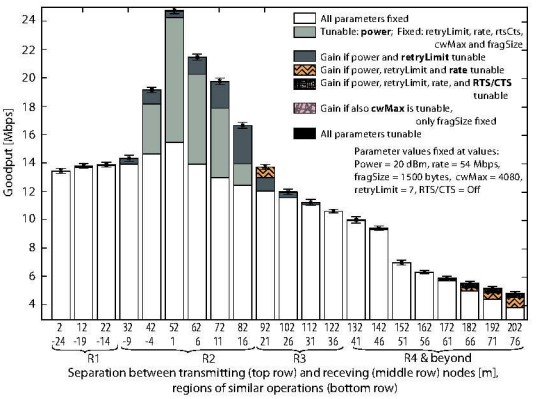
(a) The performance of TCP traffic in LOS propagation conditions without fading.



(b) The performance of TCP traffic with rayleigh fading model applied.



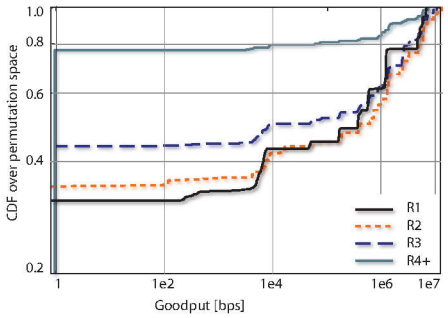
(c) The performance of UDP traffic in LOS propagation conditions without fading.



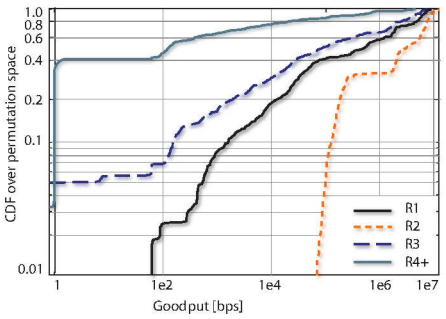
(d) The performance of UDP traffic with rayleigh fading model applied.

Figure 5.9: Performance of UDP and TCP traffic in scenarios with LOS channel and Raighleigh fading [56]. Figures show the overall achieved performance with fixed parameters that were chosen to maximize performance over the whole considered range of distances. They also show performance is increases if these parameters become tunable.

is goodput maximization without any consideration for packet losses. Due to Rayleigh fading, as opposed to the LOS conditions, the small retry limit is effective only at small distances between the transmitter and the receiver, at larger distances even for UDP the number of packet losses becomes too high. Clearly, the TCP traffic is organized differently, and there any packet losses are harmful, which leads to the retry limit being set high. The analysis of Figure 5.9 shows that though we have considerable diversity in effect of parameter settings on different scenarios and operational conditions, the performance of the concrete operational point could be significantly increased with very small number of actions. From these figures it is also clear that the results that can be achieved by parameter adjustment depend only loosely on the actual distances between the nodes, making graph approximation of these relationships feasible. Finally, in order to emphasize the distinctness of the regions $R1$–$R4$ given in Figure 5.9, Figure 5.10 shows the cumulative distribution functions for the throughput in the UDP and the TCP cases for the different operational regions. One can observe different behavior of the curves both with respect to scenarios and the operational regions within them. This indicates that identification of

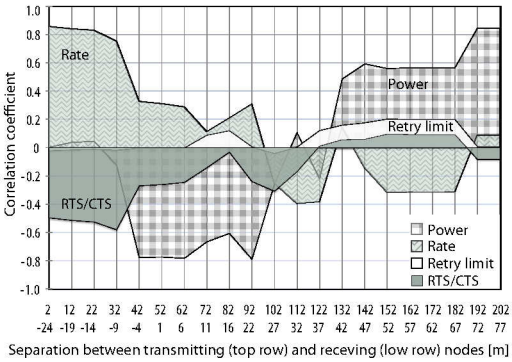


(a) UDP traffic in LOS propagation conditions, with no fading model applied.

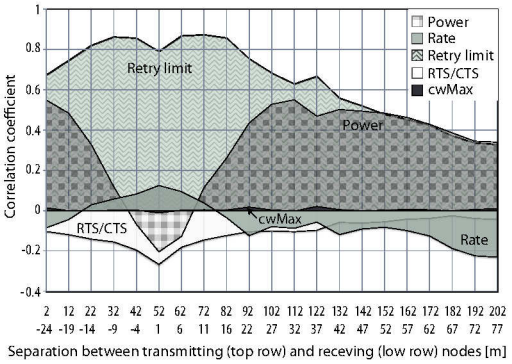


(b) TCP traffic with Rayleigh fading model applied.

Figure 5.10: Cumulative distribution functions (CDF) for parameter permutations obtained for UDP and TCP traffic for different abstractions regions of the performance graph [56] (R labels in Fig. 5.9).



(a) UDP traffic in LOS propagation conditions, with no fading model applied.



(b) TCP traffic with rayleigh fading model applied.

Figure 5.11: Correlation coefficient value for all parameter permutations [56].

operational context is needed to enable effective performance optimization.

Given the strong dependency between performance and just few of the possible parameter values, the simple linear approximation procedure discussed above should work well unless there are significant non-linearities present in the dynamics of the network. Figure 5.11 shows the correlation coefficients for the influence of different parameters on throughput for all parameter permutations. We see that the strongest correlations (either positive or negative) are with respect to either the rate or the power. Further, the correlation coefficients are seen to be quite robust against small changes in the distances between the nodes. Significant behavioral changes are rather observed for the regions that loosely correspond to a certain type of connectivity graphs and their possible transformation ($R1 - R4$). Therefore, very simple greedy optimization scheme would be able to keep the network performance at a high level, provided the correlation information is available.

Of course, the situation is not totally so straightforward with respect to parameter influence. First, there exist parameters that have a non-linear performance effect, such as the duty cycle in the WSN networking [205] (also see Chapter 3). Second, there are always constrains on tunability of a certain parameter. Let us consider the hidden terminal

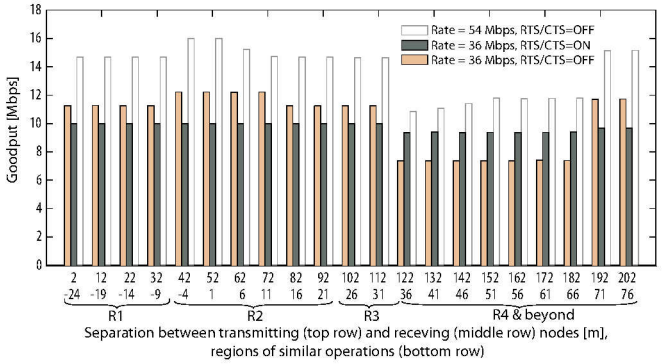


Figure 5.12: Network performance and fairness achieved by tuning the datarate and the RTS/CTS parameters for the UDP traffic in LOS propagation conditions, with the transmit power being fixed to 20 dBm and the retry limit to 7.

problem that arises for all four scenarios. It is in very few cases that we see that switching of the RTS/CTS handshake or maximization of the retry limit does have a greater effect than adjustment of other parameters. The only exception is the region $R4$ of the scenario of UDP traffic with Rayleigh fading, see Figure 5.9d. However, if, for example, we consider a subset of the UDP scenario with LOS channel, Figure 5.12, we see that changes in the datarate allow to mitigate the hidden terminal by increasing the number of send packets and reducing the sensitivity threshold without employing the RTS/CTS mechanism. Yet, if for some reason the datarate parameter cannot be adjusted then the RTS/CTS mechanism is beneficial. This simple case illustrates one should consider effect of parameters on network dynamics not only with regard to environmental conditions and application patterns, but also the optimization state space defined, in our case, by the set of adjustable parameters.

More comprehensive results, showing the effect in changes of a certain parameters, conditioned on other parameter settings (and their tunability) are given in Figures 5.13 and 5.14. This is done using correlation coefficients between the goodput and various parameters. The same figures also illustrate the robustness of particular parameters with respect to changes in channel conditions, inter-node distances, traffic/utility patterns and behavior of other parameters (optimization state space). They figures display correlation coefficients obtained for all individual parameter permutations color-coded on the basis of their behavioral similarity. They also show the correlation coefficients *clustered* into five classes. These values correspond to each of the colored groups, and can serve as additional labels for connectivity graphs to indicate the likely network dynamics with respect to parameter changes. Figure 5.13 gives detailed results for the case of transmit power being altered. We see that for *all* of the parameter permutations the results are naturally divided into *few* different robustly defined zones, and strong correlations can be observer for majority of the cases. This indicates that for many cases even if we are not in full control of the network, e.g., there is an autonomous datarate control, based on correlation coefficients we can choose the behavioral strategy that would maximize the performance. Figure 5.14 on the other hand shows sample results, where parameters were found not to have a strong and/or similar performance impact. Figure 5.14c shows that the maximum contention window size parameter does not have significant effect on performance in exception of 4 out of 576 permutation cases (the exceptions are drawn in red color in the figure). Figure 5.14b displays that for the TCP scenario, again with few

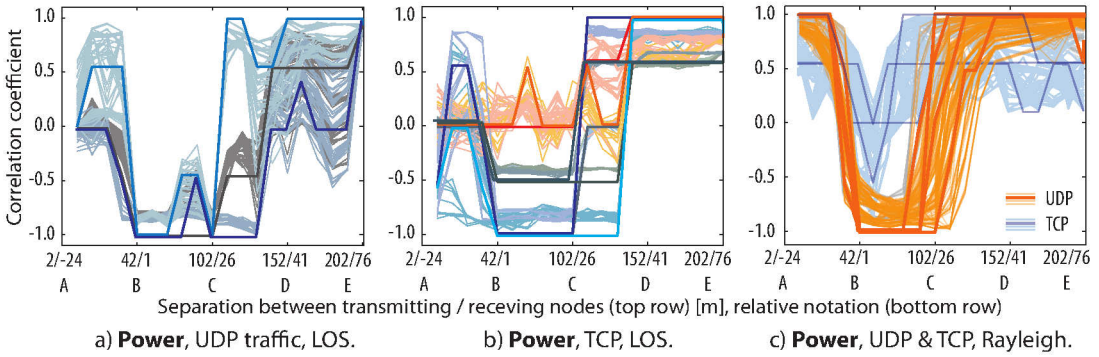


Figure 5.13: Correlation coefficient of the transmit power and obtained performance for all parameter permutations [56].

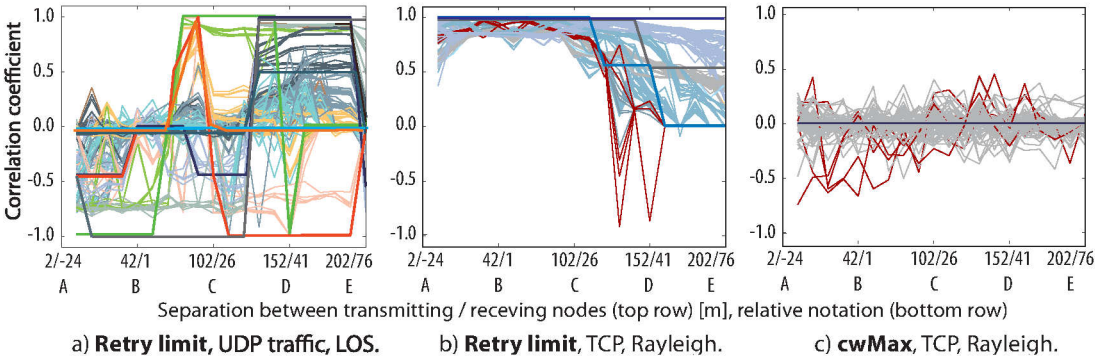


Figure 5.14: Correlation coefficient of the retry limit and cwMax MAC parameters and obtained performance over all range of simulated inter-node distances and parameter permutations [56]. Similar behaviors are color-clustered. Bolder lines denote the respective *clustered* correlation coefficients that later can be used for labeling the corresponding operational conditions (e.g., graph connectivity approximations).

exceptions, the retry limit parameter should always be set to the maximum value of seven, as this increases the reliability of the link. This influence is particularly strong for smaller inter-node distances. Figure 5.14a shows a higher diversity of parameter correlation coefficients behavior with respect to other network settings, with two noticeable trends. For the first group of the parameter permutations the tuning of the retry limit either does not have a significant effect, or its setting to the maximum is required, i.e., overall, maximization of this parameter is the right optimization choice for the given connectivity graphs. For the second group the influences of the retry limit parameter fluctuates, i.e., it should be considered as an optimization knob. The other notable trend is that for the less influential parameters the correlation coefficient behavior may differ between traffic profiles and channel conditions (see Figure 5.14a,b), i.e., they do not display such a robust dynamics as in the case of highly influential settings (see Figure 5.13).

Based on the observed results, we hypothesize that if one chooses to conduct network optimization solely based on the connectivity graphs the following algorithm is beneficial. One should perform network optimization such as to achieve minimally connected labeled network graph, however, avoiding motifs that contain the hidden terminal problem. If we consider more complex utility metrics, such as fairness, then one should also avoid con-

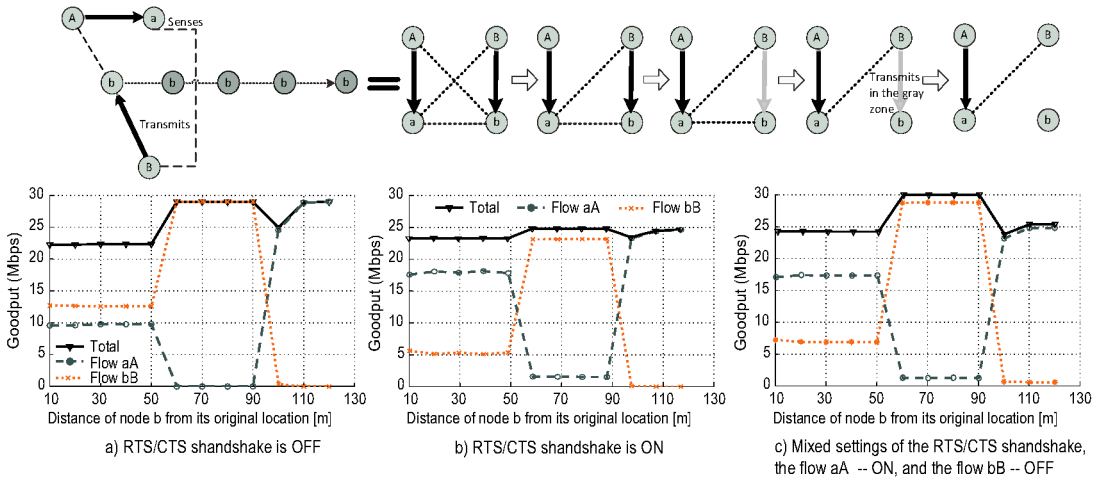


Figure 5.15: The asymmetric setup of wireless nodes. The position of nodes a, A and B are fixed. The receiver b moves away from the transmitter B so that first the hidden terminal problem occurs, which then is worsened by the gray zone connectivity for the bB connection. Before the bB link breaks the node b stops sensing the a node as well. One can observe that performance of such asymmetric network graph can be only limitedly improved by tuning the RTS/CTS parameter, even if it is adjusted on the per flow basic.

nectivity graphs that lead to asymmetric behavior, which were extensively analyzed by Garetto *et al.* [298]. (In Figure 5.15 we show that performance of the asymmetric network can in part be mitigated through parameter adjustment, however it does not completely solve the problem, and it is desirable to seek alternative solutions that would change the underlying connectivity graph). In other words, well-connected graphs are better than hidden terminal or asymptotic graphs, but worse than the loosely connected constructs. Additionally, zones of unstable low performance, so-called gray zones, should be generally avoided even if this results a less favorable connectivity graph. These zones rarely significantly outperform the other alternatives, but they often lead to highly varying, non-robust, performance. After the desired labeled connectivity graph is obtained further parameter adjustments are to be applied to enhance utility-specific link properties. Our study indicates that in many cases (e.g., operational regions $R1 - R4$) such decomposition of the reasoning process works. However, in the region border cases, such as the far-off part of the region $R4$, and generally complex environmental conditions, the optimization problem has to be solved in a joint manner.

The performance-parameter correlation coefficient metric that we proposed captures network dynamics, i.e., it can be directly related to the obtainable transformations of network connectivity motifs, as well as further link improvements. Being linear this metric avoids being tightly linked to small changes in utility functional, which makes it robust. Therefore, for efficient optimization it is beneficial to, first, identify operational spatial context by obtaining a connectivity/conflict graph. Then the dynamical context is to be obtained, i.e., parameter correlation values for the given utility functional and the state space of available parameter changes. After that parameters are to be adjusted following the order of correlation coefficients. If the optimization state space is large the proposed parameter correlation coefficient metric can be utilized as part of a metaheuristic mechanism to obtain faster a near-optimal solution to the given problem (see Chapter 4).

5.3.3 Short Summary

In this section we proposed the novel modeling approach that captures dependencies between obtained performance and optimizable entities with respect to a range of operational conditions. Graph approximations studied before concentrated only on the connectivity and the contention relations, and do not portray the dynamics from changes in configurable network parameters. We have suggested using parameter-performance correlation coefficients to capture this dynamics. Our experiments have confirmed that this is, indeed, an efficient and robust metric, which can, combined with graph-based approximations, lead to effective autonomous wireless network optimization.

5.4 Temporal Modeling with HSMMs for Spectrum Activity Patterns Estimation

In this section we experiment with Hidden Semi-Markov Models (HSMM) for online estimation and modeling of spectrum usage patterns of wireless networks and their respective duty cycles. We also consider a simpler mechanism as an alternative to complex HSMM-based modeling. Both of the proposed algorithms analyze power spectrum samples in time domain as sensed by WARP SDR boards [307] or TelosB wireless sensor nodes placed at different distances from the transmitters [58]. The first, simpler, algorithm utilizes the estimated density function of the marginal distribution of the received readings to perform clustering of the power samples into the classes that correspond to the ON/OFF activity states. The second algorithm uses hidden Semi-Markov processes and the Viterbi algorithm [343] to model and estimate the radio activity patterns of signal sources. The algorithm learns statistical models for each activity state, along with the probability distribution of the holding times, i.e., how long the sensor is likely to sense a corresponding power level. Application of these models results in better signal classification. Unlike the first algorithm the HSMM-based method can be used for both signal classification, and estimation of the state duration that corresponds to a certain power level. It also performs significantly better when sensor nodes operate in complex propagation environment receiving low RSSI values. However, the first simple algorithm suffices in standard operational conditions for modeling medium and strong signal sources.

We study signals obtained from one or two transmitters, with activity patterns following the gamma distribution. We analyze both strong and weak power spectrum samples. We also consider for a mobile signal source. Additionally, we show possible benefits of cooperative sensing for correctly estimating the ON/OFF activity patterns with the hidden Semi-Markov processes using data obtained at different sampling rates from another node. In the rest of the section we discuss the incentives for ON/OFF modeling and the chosen estimation algorithms. Then we describe the experimental setup, and analyze the performance of the chosen algorithms on the empirical data.

5.4.1 Motivation

Our work is belongs to the research areas of Dynamic Spectrum Access (DSA) and Cognitive Wireless Networking (CWN). One of the primary problems in this domain is the efficient utilization of available spectrum. For this efficient models addressing spatial and temporal aspects of spectrum usage are need, so that they can be applied in specific pro-

tocols, e.g., at the MAC layer, or as part of Radio Environment Maps (REMs) [344] and Radio Resource Management (RRM) tools [345–347].

The spectrum information may be both relatively static, e.g., for the TV white space reuse scenarios, or much more dynamic and less predictable, such as in the scenarios with DSA enabled networks [348] and femtocells [349]. These are especially challenging in indoor environments due to the complex propagation characteristics and interference patterns. Therefore, for capturing of the operational context probabilistic models based on real measurements are the promising candidate models for these type of scenarios. Using these models different nodes in a network can act independently or collaborate in order to create spatio-temporal maps of the environment that are valid for specific time period and propagation conditions. In our work we address the temporal aspect of spectrum usage, using power spectrum (RSSI) measurements as inputs for our models and algorithms.

Energy detector based sensing or *power spectrum sensing*, also known as radiometry, is the most common popular type of spectrum sensing method used for interference and primary user detection in the wireless and, in particular, DSA and CWN communities [350–354]. The method is characterized by low implementation and computational complexity, and has low power consumption. It is also generic and requires no knowledge on the signal type of the interfere or the primary user. The received power signal, often provided in the form of RSSI (Received Signal Strength Indication) samples, is used to detect the amount of energy received in a certain frequency band for conventional wireless networks. This metric allows to predict the amount of interference that is likely to be encountered in a particular band. In DSA applications this information is often used to estimate the presence of primary users in the frequency band. It can be also used for dynamic channel assignment or link performance optimization actions.

When the level of energy in the band is measured, there are several ways to interpret the results. One way is to compare the output of the energy detector with a threshold, which depends on the noise floor [355]. However, this approach is difficult to apply in setting of the threshold levels in the cases of low-power noisy signal, i.e., under low signal-to-noise ratio (SNR). This is especially challenging if the observed low-power signals change their power levels over short periods of time as in the case of mobility. In such situation it is difficult to differentiate noise from the primary users [350], which is required in the most of the DSA scenarios. The other alternative is to dynamically determine the thresholds for detecting a useful signal, which is one of the application of the models and the algorithms discussed in this section.

Summarizing, one can distinguish between two basic application scenarios for temporal power spectrum sensing data:

- estimation of active (ON states) or inactive (OFF states) of different transmitters, including estimation of several ON power levels, i.e. power-level clustering,
- modeling and prediction of the occupancy time for a particular frequency band.

The first scenario basically corresponds to detecting *instantaneous* channel occupancy. The information on the power levels and their average duty cycles can be used to perform run-time optimization of PHY- and MAC-layer parameters depending on the encountered amount of interference [321]. Solutions for the second scenario allow to optimize Dynamic Spectrum Access (DSA) by utilizing temporal power spectrum data including *historical* information. Earlier work on the so-called MAC-layer sensing [345, 347] has shown that

knowledge of either the *duty cycle* or the complete distribution of ON and OFF periods can be used to considerably enhance the efficiency of DSA algorithms depending on the detailed time-domain structure of spectrum use. Ideally, the second application scenario would allow to identify multiple signal sources in a network, predict their activity patterns and, therefore, effectively schedule transmissions of secondary users. Depending on the application scenario, mechanisms of different complexity are to be applied for data processing. Further, we propose two algorithms, with each of them being appropriate for different environmental/spectral conditions and target different application scenarios.

5.4.2 Algorithms for Temporal Estimation of Power Spectrum Usage

Next, we describe each of the proposed algorithms and their spectrum occupancy models. Our aim is to show that simple models and algorithm are often sufficient for good environmental conditions. However, when the environment becomes more challenging to estimate and more information on it is required, the sophisticated models and algorithms, e.g. probabilistic graphical models, become justifiable.

5.4.2.1 Density-based Threshold Detection Algorithm

The density-based threshold detection algorithm is simple, computationally efficient, and is applicable to fulfill the goals of the first application scenario described above. The density function of the marginal distribution of the collected data is used to determine thresholds to distinguish between several power levels, i.e., network *activity states*, that are later used for online classification of power samples. The algorithm determines number of power levels in which the samples are clustered. It requires the user to set the minimal signal duration of a transmitter, and the minimal expected difference between observed power levels. Using these, the input parameters settings for a moving average algorithm are determined in order to smooth the obtained samples. After the data is processed with this simple filter, the algorithm estimates the number of local maxima of the density function that are above a certain level to determine the initial number of the activity states. At the second stage the algorithm finds the *thresholds* between these states, which later will be used for classification of the samples. This is done by finding of the local minima of the density function obtained on the unprocessed raw data. Also the ON/OFF thresholds should yield a clear separation between the power levels. Additionally, the number of these minima should be one less than the number of the clusters. If the above criteria on local minima are not satisfied the algorithm gradually increases the number of samples over which the data is averaged until the density function of the processed samples does not yield the required result. Typically noisy or weak power spectrum samples require such iterative filtering. However, it decreases data granularity, and, in the extreme case, might prevent detecting short ON activity bursts. Additional drawback of this algorithm is its dependency on initial and cut-off thresholds mentioned above. In case of a different environment with sufficiently different levels of noise in samples, these might need to be adjusted for the efficient functioning of the algorithm.

5.4.2.2 HSMM- and Viterbi-based Algorithm

The ON/OFF pattern estimation algorithm that utilizes hidden Semi-Markov models is more complex. It is applicable in cases of severe propagation conditions, heavily subsampled data, and multiple transmitters (if we want to distinguish between multiple “ON”

power levels). The derived mechanism can also be utilized to enable the second application scenario described above. The algorithm we propose (a) clusters received samples in the power levels corresponding to, in general case, the OFF and several ON network activity states, (b) learns and captures their statistical characteristics as HSMMs, including the temporal aspects, and (c) uses these models for online classification of the incoming readings. For example, in the case of a single transmitter this mechanism estimates the distributions of power samples that correspond to the ON and OFF states. It also characterizes the lengths of these activity periods. The model is used to estimate online if incoming samples belong to the ON or OFF activity states using the Viterbi classifier.

The proposed algorithm models the activity patterns of transmitters utilizing the *alternating renewal process* or the *Semi-Markov ON/OFF process*. It uses the following transmission model. Each transmitter at any given time is either active (ON state) or inactive (OFF state). While being active the transmitter is assumed to send signals continuously with a constant power. The durations of active and inactive periods for each transmitter are assumed to be independent of each other. The earlier measurements in outdoor conditions have indicated that often for several different transmitter technologies the lengths of successive ON and OFF periods are independent [345, 346, 356]. The durations of ON/OFF periods are presumed to follow general distributions $f_{\text{on}}(t)$ and $f_{\text{off}}(t)$ with mean values μ_{on} and μ_{off}, respectively.

If the states of the different transmitters were directly observable, the estimation of the ON and OFF distributions would easily be solved using usual parametric or non-parametric statistical techniques. In practice this only occurs if there is a single transmitter of high enough power that it can be reliably distinguished from ambient interference and noise [346]. However, especially in indoors, fast fading and hostile propagation environment often cause the measured received powers from an active transmitter to fluctuate heavily. The signal can often disappear into the noise and interference for short periods of time. We overcome these difficulties by considering a *Hidden* Semi-Markov Model (HSMM) [357]. The true state of the system is still assumed to follow a Semi-Markov ON/OFF process (or in the case of multiple transmitters, a product of such processes, resulting in one OFF and several ON states). But instead of making observations on that process directly, we assume that we observe at given times a received power that is random with the distribution depending on the actual state. In networking HSMMs were successfully applied to estimate confidence intervals of packet loss measurements in Internet [358]. In the field of dynamic spectrum access these types of graphical models were applied on empirical data for offline power spectrum activity estimation by Geirhofer *et al.* in [346]. However, in their work only one transmitter was considered, the observed differences between the spectrum power samples corresponding to ON/OFF levels were high, the training of the models was done offline (i.e. there were a lot of spectrum samples available) and, therefore, the challenge for state estimation was minimal. In our work we apply HSMMs for much harder problems.

The general structure of the employed HSMM is shown in Figure 5.16. The model requires specification of a) *number of the states* in the HSMM graph, b) *initial probabilities* of the states, c) *transition probabilities* between the states, d) *sojourn times* spent in each of the states (corresponding to $f_{\text{on}}(t)$ and $f_{\text{off}}(t)$ for a single transmitter case), and e) the *emission distributions* of the received power conditional on the state of the system [343]. For the initial classification of the power levels, estimation of their empirical averages and standard deviations, we use the density-based threshold detection algorithm described above. The number of states of the HSMM is set to be equal to the

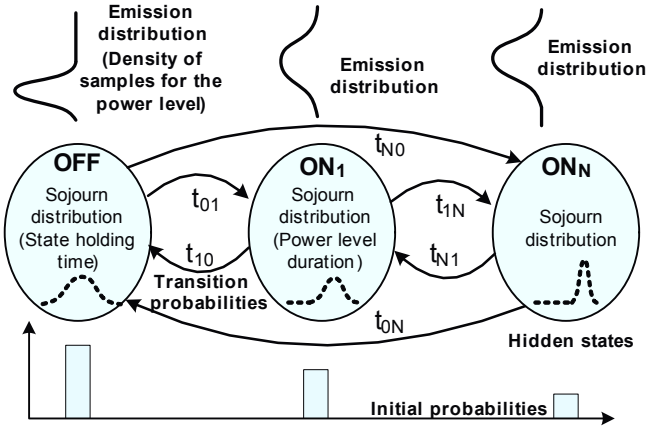


Figure 5.16: Generalized hidden Semi-Markov model used [59]. The figure shows three states of the HSMM corresponding to the OFF activity state of the network and two ON states that are characterized by different power levels.

number of resulting power classes. The statistical characteristics of these power classes are used to set initial values the emission distributions of the HSMM. We use equal initial state probabilities for the HSMM. The transition probabilities between all HSMM states are initially set to be even as well. However, during the model training, which we describe later, they get automatically adjusted. We applied normal emission distributions throughout, but these can easily be replaced by more general distributions to cover the effects of, for example, non-Gaussian interference or complex fading environment. The first zero of the autocorrelation function of the received power was used to determine initial values for the parameters of the sojourn distributions whenever parametric model was considered. We chose the gamma distribution as a parametric model for the sojourn time distributions due to its flexibility. The scale parameter θ was chosen to have initial value ranging from 1 to 5, while the shape parameter was set as $k = \mu\theta$, where μ is the time scale obtained from the analysis of the autocorrelation function as discussed above.

In order to avoid problems of the previously described threshold-based algorithms, there needs to be a mechanism for adjustment of initial values preset before based on the training samples. To refine the initial values of emission and sojourn distributions and adjust state transition probabilities, the method of finding the maximum likelihood iteratively using the *Expectation Maximization (EM) algorithm* [357] is typically applied. In particular, we used the specific R package, described in [359], for estimating hidden Semi-Markov models.

Next we train HSMMs to classify the incoming RSSI readings, and then utilize these models online by employing the Viterbi algorithm, as proposed in [343], to estimate the most likely sequence of hidden states from measurements. This information is used to classify the incoming power readings in one of the states of the system (in general case the OFF or one of the ON power levels). We chose to employ the Viterbi method as this is maybe the most well known and powerful decoding algorithm that relies on maximum of the available historical data. Therefore, application of this algorithms is likely to provide the best classification results possible, which is important as one of the main goals of this work is to explore feasibility of hidden Semi-Markov modeling for temporal spectrum modeling. In the future simpler classification techniques can also be explored, e.g., the ones based on maximum likelihood, to make the algorithm less computationally intensive.

5.4.3 Results

In this section we evaluate the proposed algorithms. First, we shortly describe the experiment setup from which we obtained the data samples. Then we invoke the two algorithms on this empirical data emulating their online application, and analyze the obtained results.

5.4.3.1 Experimental Setup

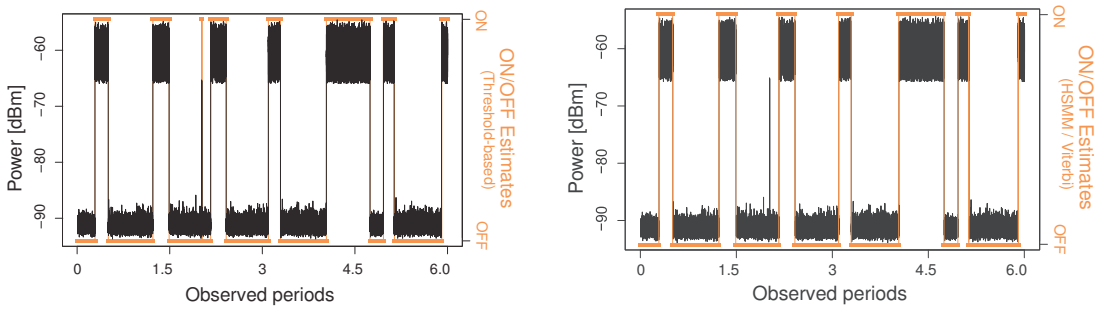
To evaluate the proposed algorithms we utilized our large spectrum measurement testbed installed in the indoor office environment as described in Appendix D [58]. For our experiments we used two types of sensing devices, namely WARP SDR boards [307] originally developed by Rice University (which are commercially available from Mango DSP) and Crossbow Inc. TelosB platforms [162]. Sensors were located in five different rooms covering over 240 m^2, with one supporting wall dividing the space. The testbed operated in the 2.4 GHz frequency range. We have used in total 60 TelosB nodes and 9 WARP boards. However, for this work we analyzed the data from only a small portion of those. After the measurement traces were collected we have chosen for the analysis the two traces for each type of sensors that correspond to the boards receiving the highest and the lowest power readings from the transmitters. To estimate the performance of the proposed methods we fed the traces as incoming online samples to our algorithms realized in R [360]. Before proceeding to the presentation of the obtained results, we provide a short description of sensor boards and transmitters used to generate the measurements.

Our spectrum sensor choice clearly falls into two distinct classes with WARP board being a high end SDR platform providing extremely fast spectral data samples, while the TelosB represents a low-end device for spectrum sensing. We have programmed the WARP boards to provide high rate spectrum measurements with a bandwidth of 22 MHz with frequency of 740 μs. The MAX2829 [361] radio transceiver provides a 10-bit (Received Signal Strength Indicator) RSSI, which is converted into dBm power levels. There are three RF gains supported on the chip and since the RSSI is measured after the RF gain, we selected the most sensitive range showing a linear behavior in the sensitivity range from -100 to -20 dBm. TelosB nodes are equipped with CC2420 radio transceiver from Texas Instruments [362], which has a built-in energy detection (RSSI) feature. The RSSI values can be read as an 8-bit wide register averaged over a duration of 128 μs. The sensed channel bandwidth is 5 MHz. These values can be stably reported to the PC with frequency of at least one reading per 3.5 ms.

We have employed TelosB nodes and a D-Link 2.4 GHz DWL-G700AP Wi-Fi access point as signal sources. The TelosB nodes were used to generate complex signal pattern distributions, produce a mobile signal source, and serve as multiple signal sources. We used the transmitter test mode of CC2420 on TelosB node to generate modulated carrier signals [362]. The chip provides a continuous 5 MHz wide signal with most of the power concentrated in a bandwidth of 2 MHz [58]. The transmit power was set to 0 dBm. A Wi-Fi access point is packet frame based and cause gaps in the transmission, we use it in a beaconing mode.

5.4.3.2 Applying Proposed Algorithms on Measurement Results

We apply the two proposed algorithms for the estimation of the ON/OFF activity patterns produced by one or several TelosB nodes, as well as a Wi-Fi access point. In this work, we consider three different scenarios for estimating ON/OFF activity patterns. First,



(a) Received power levels and the estimated states with the threshold-based algorithm.

(b) Received power levels and the estimated states with the HSMM/Viterbi algorithm.

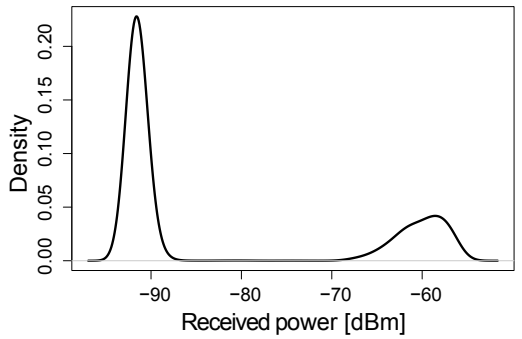
Figure 5.17: Single transmitter scenario [59]. The WARP board receives a strong signal. The receiver is located at the distance of 0.5 meters from the transmitter. Both algorithms for training use sample gathered over 16 periods.

we study the scenarios for single transmission source with a TelosB node generating a narrow-band signal. Then we study a mobile signal source from a TelosB node placed on an electric toy train using a fixed duty cycle, see Appendix D. Finally, we consider two signal sources: 1) two stationary TelosB node, and 2) a combination of a mobile TelosB node and a beaconing access point. For the single transmitter scenario we analyze the readings for two WARP boards that received the strongest and the weakest signals. For other scenarios we process the measurement traces only from the weakest receiver. We annotate the obtained results in terms of the number of periods observed by the receivers rather than in seconds. One period is defined as a sum of the average ON and OFF durations and corresponds to one duty cycle in the mean. In this work one period is equal on average to 5 seconds.

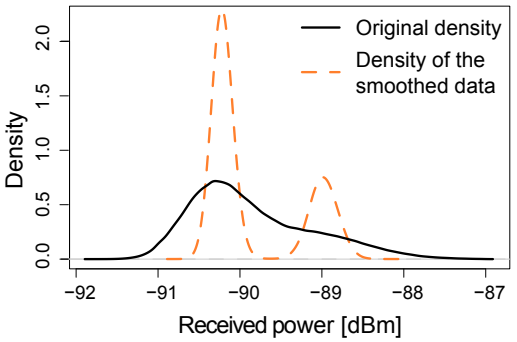
5.4.3.3 Single Static Transmitter

First we study the estimation and modeling of ON/OFF activity patterns obtained from a single transmitter. A TelosB node generated a 5 MHz signal with gamma ON/OFF distribution, with a duty cycle of 25%, an ON period with mean duration of 1.25 s, and variance of $0.16 s^2$. First we apply the proposed algorithms to process the RSSI readings of a WARP board receiving strong signals. The results are shown in Figure 5.17. The density-based threshold detection algorithm correctly identifies two classes for the power readings corresponding to the ON and OFF states, and uses the threshold level set to -69.95 dBm to correctly classify the incoming samples (see Figure 5.17a). The algorithm does not require any data smoothing. As shown in Figure 5.17b the HSMM/Viterbi algorithm also correctly estimates the ON/OFF states of the incoming samples based on the learned Semi-Markov model. Moreover, this algorithm is capable to correctly process erroneous/noisy samples, such as the one received around the second period, and classify them the OFF state (see Figure 5.17), as it takes the temporal structure of the activity states into the account. The first algorithm operates only on the thresholds, which leads to the signal misclassification. Both algorithms successfully classify the samples, when the difference between the noise level and the power received from an active transmitter is high.

The threshold-based algorithm requires additional processing when the data from the node sensing a weak signal is analyzed. In our study this is the case, when on average

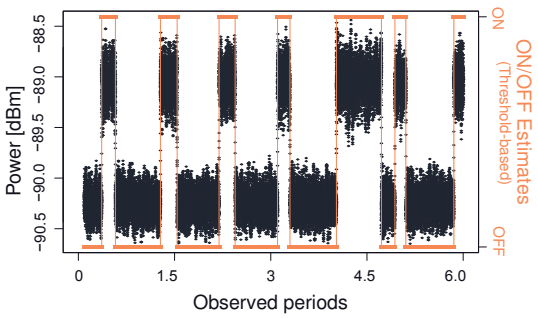


(a) The WARP board receiving the strong signal in the single transmitter scenario.

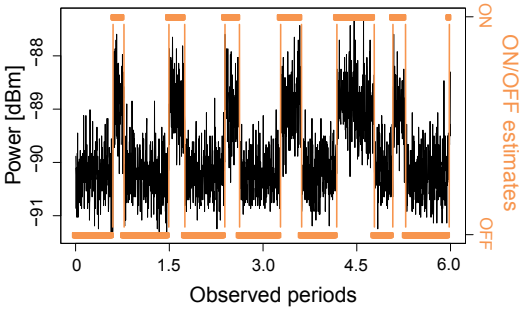


(b) The WARP board receiving the weak signal in the single transmitter scenario.

Figure 5.18: Probability density functions obtained for the power spectrum samples received by the WARP board in the two considered scenarios [59].



(a) Received power levels and the estimated states with the threshold-based algorithm that uses the moving window of 12 samples.



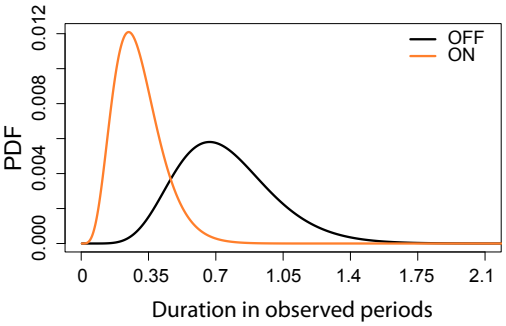
(b) Received power levels and the estimated states with the HSMM/Viterbi algorithm.

Figure 5.19: Single transmitter scenario [59]. The board receives a weak signal. It is located at the distance of 11 meters from the transmitter across the semi-concrete wall. Both algorithms for training use sample gathered over 16 periods.

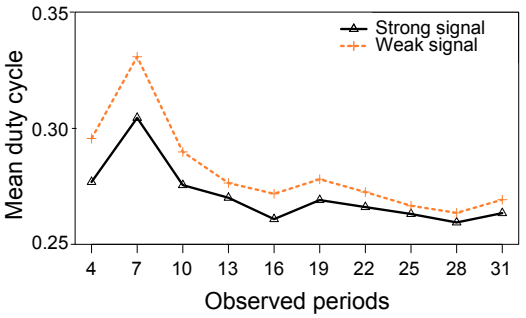
the difference between the ON/OFF samples becomes less than 2 dB. If no data filtering is applied then samples are misclassified as the two power levels are very close to each other. The algorithm could achieve correct classification of the samples with the application of the moving average filter with the window size of 12 samples, and setting the threshold to -89.64 dBm. High window size means that the heavy smoothing of the data is required, which can potentially lead to signal losses. The initial and the resulting density functions along are shown in Figure 5.18. The results of the threshold-based classification are displayed in Figure 5.19a.

The results show that in this scenario the HSMM/Viterbi algorithm performs well without application of additional filtering. It achieves accurate state estimation based on the learned Semi-Markov model, as shown in Figure 5.19b. The sojourn distribution for both of the nodes sensing the strong and the weak signal are the almost identical. This additionally indicates that the HSMMs can successfully applied for the receivers observing only weak signals. There exist certain performance degradation in terms of the required training samples exists for the weaker signal, but it does not lead to significant error in the ON/OFF period estimates (see Figure 5.20b-d). The figure also shows the

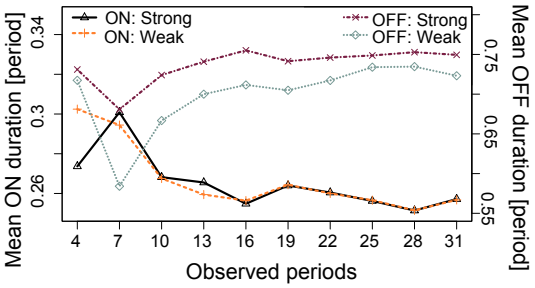
fast convergence of the parameters of the Semi-Markov models to the almost true values of the original signal distribution. Overall, the deviation in the estimated ON and OFF periods and duty cycles from the true values in the worst case does not exceed 3.5%. (The true original values are 0.25 and 0.75 of a period for the ON and the OFF duration, and, respectively, 25% for the duty cycle.)



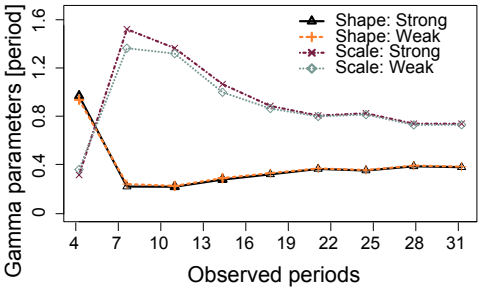
(a) The sojourn distribution for the HSMM for the WARP board receiving a weak signal.



(b) Mean duty cycles estimated after the model training as the function of the duration of the input time series given in periods.



(c) Mean ON/OFF periods estimated after the model training as the function of the duration of the input time series given in periods.



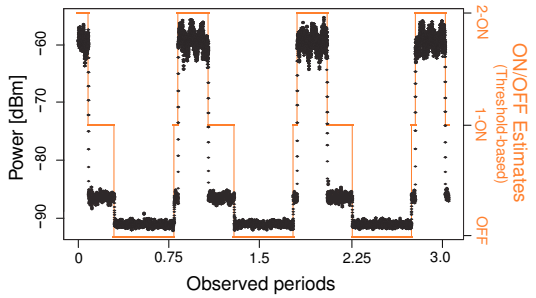
(d) Estimates of shape and scale parameters of the sojourn time distributions for two WARP boards as the function of the duration of the input time series.

Figure 5.20: The summarizing plots on the parameter estimates of the HSMM for the single transmitter scenario following the gamma signal distribution [59].

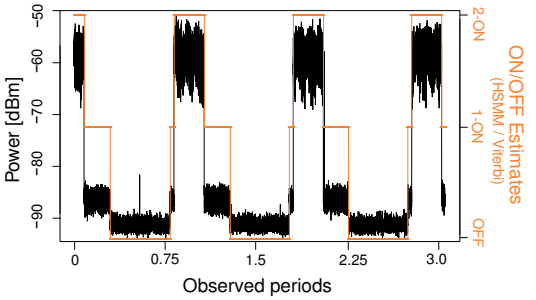
5.4.3.4 Two Static Transmitters

We have considered two scenarios with dual transmitters. In the first scenario the signal sources are two TelosB nodes. One node generates a 50% fixed duty cycle with an ON and OFF durations of 2.5 s. The other node follows a gamma distributed 25% duty cycle, with an average ON period of 1.25 s and variance rate of 10%. For the two transmitter scenario we have chosen to analyze data from the WARP board that receives strong signals from one TelosB node and very weak signals from another TelosB transmitter, as shown in Figure 5.21. The weak signal can be easily misclassified into the OFF state, as on average its difference to the noise floor does not exceed 1.5 dB.

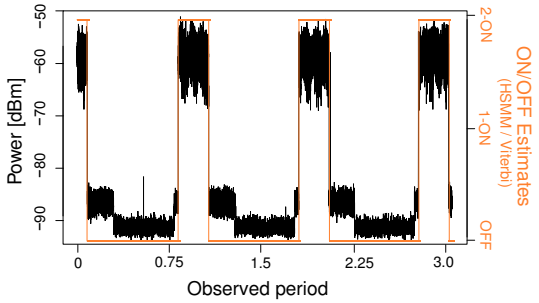
To obtain acceptable classification results for this scenario the density-based threshold detection algorithm has to apply a moving average filter with the window size of 10



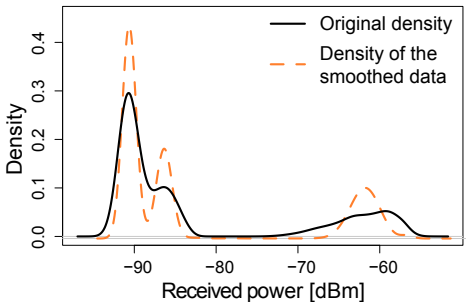
(a) Received power levels and the estimated states with the threshold-based algorithm that uses the moving window of 10 samples.



(b) Received power levels and the estimated states with the HSMM/Viterbi algorithm that uses the three-state HSMM model.



(c) Received power levels and the estimated states with the HSMM/Viterbi algorithm that uses the two-state hidden Semi-Markov model.

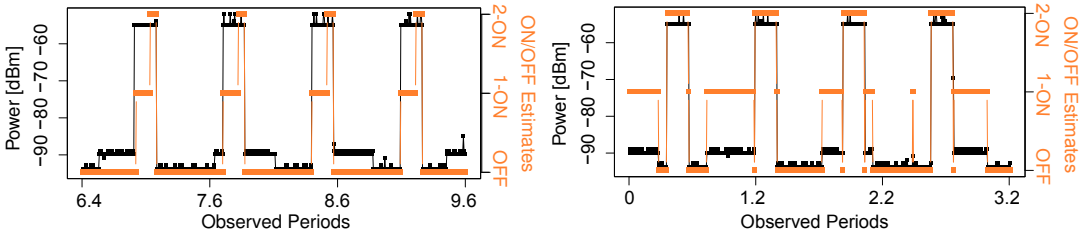


(d) Probability density functions obtained for the power spectrum samples received by the WARP of interest.

Figure 5.21: Two static transmitters scenario [59]. The WARP board is receiving a weak signal, and is located at the distances of 9 and 11.5 meters from the two transmitters across the semi-concrete wall, respectively. The training samples are gathered over 16 periods.

samples. The algorithm correctly determines that the samples have to be classified into three power levels (OFF, 1-ON and 2-ON states) and deduces the classification thresholds of -89.3 dBm and -78.4 dBm. The results of the online state estimation with these thresholds are shown in Figures 5.21a. The HSMM/Viterbi algorithm also correctly determines that the Semi-Markov model requires three states. Further training of the model is successful and the Viterbi algorithm performs correct clustering of the incoming samples without the need for data pre-processing, see Fig 5.21b. For comparison, we trained the two-state HSMM on the same set of data. With the two-state model the Viterbi algorithm completely misclassifies 1-ON state into OFF state for the weak power spectrum samples, as shown in Fig 5.21c. This indicates that the HSMM/Viterbi approach is useful for modeling and estimating the state of complex power spectrum time series, if the number of states for the Semi-Markov models increases correspondingly. However, this of course leads to an increase in processing time and computational overhead.

We also used the two transmitter scenario to demonstrate the possible benefits from cooperative nodes behavior. Two nearby WARP nodes performed the sensing at different rates, one 50 times slower than the other. If we estimate the three-state HSMM directly from the slower sensing device then we may arrive to the wrong estimations, as shown



(a) Received power samples and the state estimation on the three state gamma model trained by the same with slow sampling.

(b) Received samples and the state estimation on the three state gamma model obtained from the nearby node, which performs fast sampling.

Figure 5.22: Illustration of benefits of model sharing between the two receives for the two TelosB nodes transmitting scenario, sub-sampling case [59].

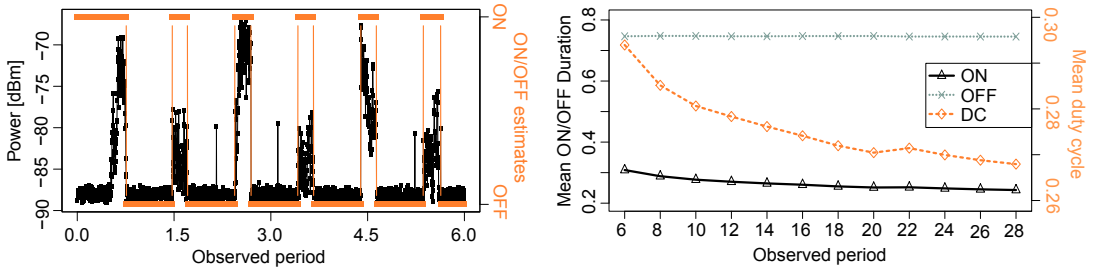
in Figure 5.22a[6]. Here the OFF estimate includes the low-power level of the ON signal. However, if we import the trained three-state model from the faster sampling sensor node to the slower sampling device then the states estimations made by this device become much more accurate, as shown in Figure 5.22b.

5.4.3.5 Mobile Signal Source

The results described above indicate that in a stationary environment the variation of the received power at individual nodes is relatively small for wideband 20 MHz receivers. We shall now illustrate the case of a mobile transmitter. These experiments were carried out using a TelosB node as a signal source placed on top of a model train that goes in ellipsoids around a part of a wall. The node transmits with 25% duty cycle. As a representative receiver, we have chosen the TelosB node that senses a bandwidth of 5 MHz. As clearly visible from Figure 5.23, while fast fading clearly severely distorts the results, we were able to obtain valid HSMM models, which application resulted in quite accurate state estimations. (In this and the following scenarios we do not anymore describe results obtained with the threshold-based algorithm. Due to large variations in received power readings and complex required analysis, we could not obtain feasible classification results with this algorithm without significant additional manual tuning, which basically makes it inapplicable for autonomous online application. However, we still use this algorithm to initialize the HSMM modeling.)

In the second mobility scenario we again use a mobile TelosB node with the same 25% duty cycle, but this time we have added a beaconing Wi-Fi access point. This time we employ the WARP board as a receiver. Our purpose is to study if it possible to correctly identify the signals form the Wi-Fi AP as belonging to the ON state. The additional difficulty is that the WARP board skips some of the beacons, which results in some irregularity in observed signals from the AP. As is evident from Figure 5.24a the two-state gamma model misinterprets the Wi-Fi signals as the OFF state. However, the three-state

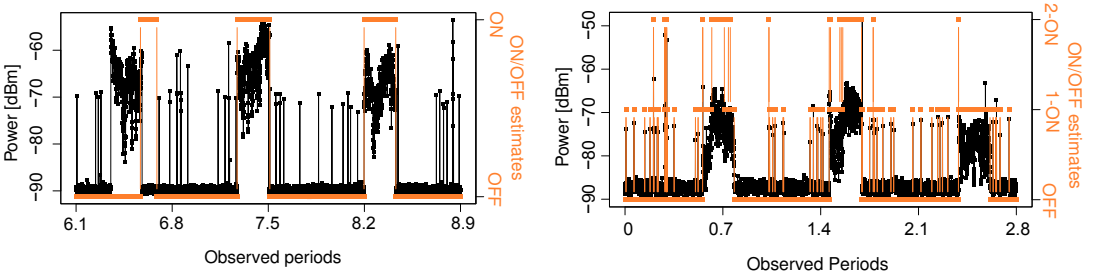
[6]Some of the figures in this section displaying sample observations of power signals and their classification, e.g., Figures 5.22 and 5.24, show signals taken either from different sensor boards or observed at different point of time. Therefore, the displayed data points might slightly differ in their subfigures, especially with respect to the x-axis. Such misalignment resulted that we took different subsets of the data samples for display of the classification result after application of the studied models. This choice was rather random, and neither results or conclusion are affected.



(a) Received power levels and the state estimation on model trained after 18 periods are observed.

(b) Estimates of ON/OFF period and the duty cycle as the function of number of training samples given in periods.

Figure 5.23: The mobile transmitter example. The receiver is located at approximately 7 m across the concrete wall from the transmitter.



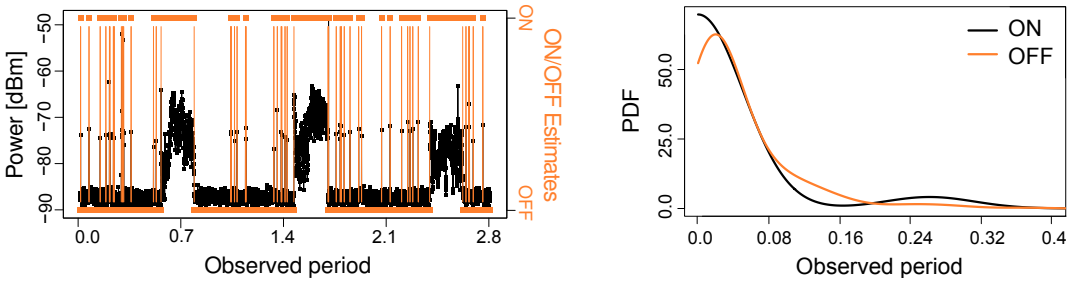
(a) Received power levels and the state estimation on the two state gamma model.

(b) Received power levels and the state estimation on the three state gamma model.

Figure 5.24: The two transmitter scenario with the moving train and the beaconing Wi-Fi access point. Measurements are shown for a WARP board located at the distances of 7 m across the concrete wall from the mobile transmitter. The access point is operating approximately 20 m away from the testbed area.

gamma model suits very well for this scenario, both the TelosB and the Wi-Fi signals are identified to belong to one of the two ON states.

As we previously have observed it is crucial to define the correct number of states for the model. We decided to further experiment with a different type of the HSMMs, namely a k-smooth non-parametric model. This model does not require the specification of the sojourn distribution and might converge to almost arbitrary estimates [359]. Therefore, it might be applied initially to decide to the type of a parametric distribution. It could also be utilized to decide if new states have to be added to the HSMM. The drawback of this model is slow convergence and, sometimes, wrong classification due to overfitting. We applied the two-state the k-smoothed non-parametric model for estimation of the network activity states to the two transmitter scenario with a mobile source described above. Figure 5.25 displays the results obtained with this HSMM and the respective sojourn distribution. The state classification results in better outcome than for the two-state gamma model (see Figure 5.24a), as the Wi-Fi source is identified with an ON state. If we consider the sojourn distributions of the k-smoothed non-parametric HSMM, Figure 5.25b, we see that the distribution corresponding to the ON state of the model has two local maxima. The first maximum is close to duration of a single AP power sample, where is the other is at about 0.26 of a duration of one observed period, which roughly corresponds to the



(a) Received power levels and the state estimates.

(b) The respective sojourn distributions.

Figure 5.25: Illustration of application of the two state k-smooth non-parametric model for the two transmitter scenario with one mobile source and an AP.

time a moving TelosB transmits. Therefore, in principle, several local maxima in a sojourn distribution, might serve as an indication that number of vertices of the HSMM should be increased to obtain better classification with parametric models. We plan to explore this hypothesis as part of the future work of automatic identification of vertexes for the HSMM in complex scenarios, where direct analysis of marginal signal density distributions does not produce satisfactory results.

5.4.4 Short Summary

In this section, we have applied hidden Semi-Markov models for modeling power spectrum activity patterns in the indoor environment. These models were then utilized as a part of the proposed algorithm for online classification of incoming samples. This mechanism was compared with the simpler approach, namely a density-based threshold detection algorithm. This algorithm can be successfully applied online to classify power samples and determine the threshold to distinguish between the ON and OFF states for both strong and weak signals. However, for the weak signals it requires strong smoothing of the data which might prevent it from detecting short transmission sequences. On the contrary it has been shown that the ON/OFF activity patterns can be successfully modeled using the hidden Semi-Markov models without significant loss in data granularity. These models can be further used to accurately classify online the incoming power samples with the Viterbi algorithm, even if there are occasional erroneous samples present. We have also shown possible benefits from intra-node model/information sharing for more accurate spectrum activity estimation on the example of the sub-sampled data. Furthermore, we have successfully applied the HSMM-based algorithm to classify signals incoming from a moving transmitter, as well as of multiple transmitters. In Chapter 6 the autonomous RRM framework for home wireless networks, called Home Cognitive Resource Manager, successfully applies the hidden semi-Markov models for estimation of activity patterns of different 2.4 ISM channels, and utilizes this information to guide the sensing process performed by a MAC protocol [363].

We plan on investigating the applicability of HSMMs to the cases where the assumption of independent durations of ON and OFF periods is strongly violated. We also want to consider alternative ways to construct the models, e.g., utilizing prior information on the number of primary sources, and then training a separate model for each of them.

5.5 Conclusions

In this chapter we suggested new graph-based models for effective approximation of wireless network optimization problems, with particular focus on Wi-Fi based systems. We showed how small labeled network motifs can be constructed to capture both *spatial and dynamic* network properties. These models reflect the principal components of network behavior, and may, therefore, indeed serve as *efficient and robust* means to derive operational contexts, and provide valuable inputs for network optimization. First, we studied occurrence patterns of conflict graphs, more precisely their three and four-vertex motifs, that arise from different underlying network topologies. These topologies are built after selected spatial wireless network models, with one of them, the model based on Geyer point process, accurately reflecting real deployment of access points in the Google Mountain View Wi-Fi network. We observed that only a small number of network motifs occur frequently in these topologies. Moreover, though the underlying spatial models result in different rank-frequency distributions of the occurancy of the motifs, the set of frequently encountered subgraphs is the same for all of the studied spatial models. This fact indicates that the number of the operational scenarios required to be evaluated as part of an optimization task can be often limited to graphs which occur frequently. Thus, the optimization task can be significantly simplified with respect to the number of operational conditions that have to be considered.

In the second part of the chapter we proposed to use labeled network motifs to capture, through quantization of parameter-performance correlation coefficients, the influence of protocol parameters on observed network performance in changing operational conditions. Our experiments on small ad-hoc networks have demonstrated that such models are very useful for simplification and dimensionality reduction of an optimization task. Furthermore, based on our results we could also establish relations between models capturing network dynamics and connectivity graphs. We plan to extend this work by considering larger networks with varying topologies and more complex utilities.

In the third part of this chapter we applied hidden Semi-Markov models, as well as a simpler and more computationally lightweight techniques that directly utilize the estimated density function of the marginal distribution of the received readings for temporal online characterization of ON/OFF power spectrum activity patterns. Here we observed the classical trade-off between complexity of a method and its efficiency/applicability. The HSMMs were performing very well in all the considered scenarios. The models were accurate even when obtained for poor wireless conditions with ambiguous sensing results, and in situations where complex temporal deductions were to be made (e.g., several “ON" power states had to be distinguished). However, the simpler method, though somewhat vulnerable to incorrect initial settings, was performing equally well in less complex scenarios with clearer incoming power signals. Our study has demonstrated that both of these newly proposed methods have their own exploitation scenarios. However, only the more complex HSMMs can be made robust towards significant variations in the operational conditions.

6

Designing a Self-Optimization System for Wireless Home Networks

In this chapter we discuss and develop a self-optimizing system for Wireless Home Networks (WHNs), testing and applying some key aspects of the methodology suggested in Chapter 2. The system is named Home Cognitive Resource Manager (HCRM)[1]. We concentrate on the design and implementation stages of the development life-cycle. We derive the specific goals for the system based on careful analysis of user requirement to WHNs, which lead to adoption of the agent-based perspective on system design [85]. We believe this approach to be very useful for most self-management concepts, including Cognitive Wireless Networks (CWN) [365, 366]. We take the Cognitive Resource Manager (CRM) [60, 367, 368] as an architecture of individual agents. This architectures serves as a "constraint that de-constrains", i.e., it allows to achieve high system flexibility, while providing structural constraints to ensure desirable levels of robustness [14]. After designing the HCRM using the above architectural guidelines with detailed architecture being discussed in the latter part of Section 6.2, we discuss system implementation and evaluate its performance.

The resulting prototype supports, in an integral manner, dynamic pursuit of goals and policy regulations formulated by network stakeholders using multiple control loops and optimization techniques that employ utility- and policy-based reasoning. The system operates on an extensible parameter configuration state space. It acts on multiple protocol layers including dynamic channel assignment, which is enabled through interface virtualization. The integrity of the system is maintained through conflict resolution and prioritization functionalities. Our prototype operates on wireless nodes supporting both COTS Wi-Fi transceivers and WARP SDR platforms [307]. For optimization we use either predefined decision trees or experimental implementations of two Machine-Learning (ML) techniques, namely Multi-armed Bandit (MAB) algorithm and Hidden Semi-Markov Models (HSMMs). The system has been successfully evaluated on multiple wireless nodes where several multimedia streams and data flows are competing against each other and external interferes. We conclude this chapter with lessons learned from prototyping of this self-optimizing Radio Resource Management (RRM) system for WHNs.

6.1 Introduction

As previously discussed in Chapters 1 and 2, the increasing complexity of future wireless networks leads to the requirement for self-organization. This is true especially in home networking where users are typically not networking professionals and cannot be expected to perform complex optimization and management tasks. Additionally, Wi-Fi networks,

[1] The papers relevant to the HCRM design that the author has been contributing to are [52, 61–63]. The prototype to large extent is one of the results of the joint work on the European ARAGORN project [364] that involved contributions of several academic and industry partners.

in contrast to cellular systems, are typically constructed from heterogeneous vendor equipment and software, which raises the issues of middleware and interfaces compatibility, as well as possible emergent behavior from combinations of these units. In this chapter we describe the design and prototype implementation of the self-optimizing system for these networks. It is built to fulfill two main tasks. First, we want to demonstrate that using the discussed architectural principles one can create a viable self-optimizing system for wireless home networks[2]. Second, we aim at creating a framework that would be general and flexible enough to address complexity and evolvability issues in these networks, especially during the run-time management phase. Therefore, we decided that the framework should 1) allow experimenting with a number of optimization algorithms, 2) act on multiple protocol stack layers utilizing various degrees of cooperation between nodes, 3) be extendible to support different types of wireless hardware.

We shall now further motivate our work by describing the likely development of wireless home networks in the near future. The popularity of wireless communication systems for home networking is rapidly growing along with the user demand for diverse high quality telecommunications services. This results in a dense wireless environment with fast channel variations and uncontrolled interference, which significantly complicates the design of wireless systems. In particular, a notable trend in Consumer Electronics (CE) is the increasing number of Wi-Fi enabled devices, like such as cameras, TVs, and other digital media appliances. In 2014, the shipments of CE devices for home usage equipped with WLAN interfaces excluding computers and mobiles devices is forecast to exceed one billion units [3]. The increasing adoption of Wi-Fi creates new usage models, like wireless content sharing and streaming between home media appliances. However, current network management solutions do not sufficiently assist non-professional users to efficiently setup and maintain IEEE 802.11 networks. For example, networks are often configured to operate on a single fixed channel, usually selected at system startup or at periodic intervals, and route traffic through an Access Point (AP), which might be suboptimal especially for dense and dynamic wireless environments. Additionally, there is an increasing number of radio technologies that operate on the same frequency range, such as the family of IEEE 802.11 radios, IEEE 802.15.4 low power devices, and Bluetooth, creating co-existence challenges [373, 374]. The such technologies as IEEE 802.11n and IEEE 802.11ac increases delivered throughput at a price of expansion of the bandwidths used. A survey of Cisco Systems, Inc. indicates that 54% of wireless deployments are already affected by wireless interference [375]. As another illustration, it was shown

[2]To avoid the ambiguity in the terminology we clarify our understanding of the terms self-optimization and autonomous (radio) resource management. Self-optimization is part of *self-management* that belongs to the concept of autonomic computing [369, 370]. Self-optimization seeks to autonomously perform mostly parameter-based optimization in order to improve system (network) performance. Radio-resource management [371] done autonomously pursues similar goals, but with explicit respect to wireless and cellular networks. Modern notion of autonomous RRM could be considered as a part of self-organizing network paradigm [372]. Radio-resource management typically applies to PHY, MAC and partially network layers of the protocol stack and aims at achieving of the optimal use of the involved resources. Radio-resource management is part of network resource management, which is a very wide concept (e.g. includes context management), and if done autonomously would involve not only self-optimization, but also other properties of self-management. With respect to the HCRM we utilize the terms of self-optimization, resource management and radio-resource management interchangeably to improve the flow of the text. We consider that WHNs are edge one or two hop networks where resource management decisions heavily involve cross-layer optimization, especially on the lower protocol layers. Additionally, the HCRM also performs some of the self-configuration functionalities, as it follows imposed policy constraints.

in [376] that in large cities the average density of Wi-Fi networks was $140-1800$ APs per km^2 already in 2007. Clearly, the local concentration of APs can be much higher [334], and is expected to increase in the future.

Because of these issues, there is a need for a system that *autonomously manages wireless home networks fulfilling changing stakeholders' needs in dynamic operational conditions*. This system should be *minimally complex* and provide high levels of *flexibility and evolvability* to incorporate the likely short- and long-term changes in user demands, operational environment, and list of desired services or applications. These goals can be addressed to different extent both at design and the run-time stages of network operation. In this chapter we describe a framework called Home Cognitive Resource Manager (HCRM) that aims to fulfill the above goals. It primarily focuses on radio-resource management aspects of the control of wireless home networks.

The HCRM is closely related to the fields of home and cognitive wireless networking [365]. The general concept of cognitive radio is considered in a number of works [32, 60, 147, 377], with an overview of the recent prototyping efforts given in [378]. Alternative self-optimization/self-management approaches for wireless environments [379–381] and, specifically, for wireless home networks [382–384], either have a different focus, do not consider the DSA aspects, or just present a certain resource allocation scheme, e.g., [385, 386], without addressing the system level aspects of the problem. The work on virtualization of wireless interfaces, e.g., presented in [387, 388], has not been incorporated as part of a complex self-management system until recently within the same context as our present work [175, 364].

Further in this chapter we discuss how complexity can be addressed during the lifecycle of a networked system, demonstrate on the developed prototype solution the feasibility of such an approach, at the same time pointing out encountered challenges. In particular, in Section 6.2 we describe the design of the HCRM. We discuss the high level requirements for the system, and their mapping to the lower-level system-specific objectives. We build the HCRM based on agent-based design and cognitive resource manager architectural principles [60]. Section 6.3 provides further details on the system design. In Section 6.4 we present selected evaluation results. Finally, Sections 6.5 and 6.6 draw conclusions with explicit summary of the lessons learned during the work.

6.2 Deriving Design

Typically, WHNs are deployed in individual apartments, and consist of a home gateway (an access point) and multiple wireless clients that, depending on the executed applications, might require either Internet and/or mutual ad-hoc connectivity (see Figure 6.1). Users have diverse requirements and access rights to services provided by a network. The deployment conditions might result in a situation where the same spectrum is shared not only between multiple users of one network, but also between different WHNs.

6.2.1 Challenges and Constraints

Some of the key challenges WHNs face are the diversity of hosted services and stringent requirements on their quality, the competition for constrained network capacity, the ample dynamics (environmental, user, and inter-component), and the resulting poor performance predictability. Suboptimality of the strictly layered TCP/IP stack without cross-layer optimization capabilities, imperfectness of software and hardware, and, finally, non-

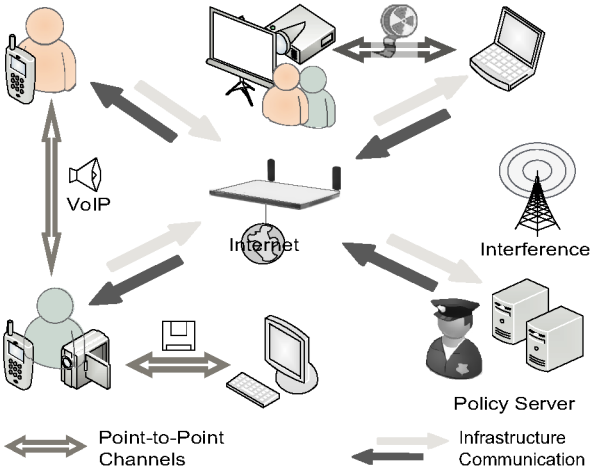


Figure 6.1: A wireless home network scenario (adapted from [62]).

professional users with the lack of expert support personnel contribute to the encountered challenges. Most of these challenges can be mitigated, at least partially, through the development of effective self-optimization techniques of various levels of complexity.

The design work on the HCRM can be decomposed into two major phases. The first is the derivation of functional requirements for the autonomous Resource Management (RM) phase, success of which is measured through stakeholder satisfaction. The second is the system design that constrains the achievable efficiency of self-optimization. In other words, the HCRM system design aims to serve as "constraint that de-constrains" the self-optimization process [89, 389, 390].

We proceed by stating the main requirements and constraints imposed on a system such as the HCRM, whereas the description on how these objectives are met by the system are given further in the chapter. (In this work we focus only on their technical goals, without explicit consideration for socio-economical aspects.) The requirements and the constraints are summarized in Table 6.1. Typical stakeholders of a home network are users, network and service providers. Objectives of a network provider can be approximated as a tradeoff between efficient usage of wireless resources, e.g., spectrum, and provision of high quality of service to users. Individual users aim to maximize performance provided by a network that can be perceived through Quality-of-Experience (QoE) metrics, while complying to external (operator) constraints. Implicit consideration for socio-economical aspects leads to the definition of constraints such as a limit on the available spectrum pool, as well as the prioritization of some users and applications. Users, and to a lesser extent providers, have dynamically changing objectives. They should be conveyed to the system accurately in order to achieve the best optimization results provided that the correspondingly imposed overhead and complexity are tolerable [42]. As discussed earlier, there exists a tradeoff between expected advantages provided by the information input to the system, the ability to accommodate and efficiently utilize this data, and the imposed costs, e.g., in terms of signaling overhead and processing complexity. One should consider not only absolute gains in performance, but also robustness of the system both with regard to variability of inputs and optimization actions, as well as emergency/fragility risks that can increase due to growing networked system complexity.

As part of the framework design we define a range of its possible inputs. One category

Table 6.1: Summary of the high-level system requirements.

By operators • Efficient use of network resources • User satisfaction • Compliancy to imposed constraints and policies	The HCRM must be able to operate in a wide range of indoor environments ranging from stand alone houses to dense neighborhoods with multiple close-by apartments with interfering wireless devices. The possibilities for network pre-planning are very limited. Network resources, e.g., spectrum, should be used efficiently in order to accommodate for the largest amount of users, while still satisfying their demands. The support for policy constraints and preferences imposed by various network stakeholders, e.g., operators, is also required as this allows imposing restrictions on resource usage, user behavior, and enforcing prioritization.
By users • Maximization of QoE and robustness in a foreseen range of operational conditions • Predictability and stability of performance • Indications of changing goals and constraints • Minimal involvement, i.e., maximal automation	Provision of high and stable Quality-of-Experience (QoE) to users is a must. The system should provide stable, satisfactory and predictable/explainable performance results for the user in changing conditions, taking into the account dynamics in application and even data flow requirements [42, 391],. Users should be able to indicate their changing goals or preferences, as well as to impose local policies (provided there is not conflict with operator policies). The system has to aim for minimal user involvement.
Additional system requirements • Minimal complexity • Evolvability and flexibility • Operation of COTS (Commercial off-the-shelf) hardware and Windows OS • Experimentation platform for self-optimization in WHNs • Policy conflict handler	Robustness-complexity tradeoff [14] dictates for minimization of system complexity, while satisfying other system requirements. Evolvability and flexility requirements to the HCRM lead to the demand for adaptivity of the corresponding optimization task, especially its goals and solution state space. The state space can adapt with respect to (a) available parameters (actuators) and sensors, (b) stated goals/constraints/policies, and (c) supported radio hardware platforms. The HCRM aims to demonstrate the applicability of self-optimization in context of resource-management on already existing low cost COTS hardware and widely spread Operating Systems (OS), as well as its extensibility to other hardware, e.g., software-defined radio boards.

of the inputs to the HCRM can be generalized as *operational conditions*. They include characteristics of the wireless environment, e.g., spectrum use, observed Key Performance Indicators (KPIs), and the inner state of the node, such as its protocol parameter settings. The latter can get adjusted both due to actions of the HCRM as well as other entities. For example, the inner state can be altered due to hardware and software updates,

autonomic IEEE 802.11 MAC rate adaptation mechanism, etc. Additionally, information on operation of external networks can be signaled to the system.

Goals and constraints is another category of HCRM inputs. We chose to lower the HCRM complexity by decomposing its objectives into two classes: global, slow varying, goals expressed using *policies*, and local, faster evolving, aims captured by *utility functions*. It should be noted that global goals affect local objectives. The explicit prioritization between goals decreases network complexity, and therefore increases its robustness. The system must meet constraints imposed by, e.g., equipment capabilities, timing requirements, and regulatory bodies.

The overall system utility can be seen as a result of a tradeoff between its efficiency, i.e., absolute displayed performance for a given set of resources, and robustness. Robustness can be viewed as a measure of performance variability as a result of changes in inputs caused, for example, by environmental instability, variations in traffic patterns or goal statements, or re-configuration actions [390]. Additionally, the notion of robustness includes explicit protection from critical system failures and drastic performance degradation for a wide range of operational conditions. Robustness can often be improved by duplication of important functionalities at different parts of the system, like duplication of the reliable traffic delivery functionality at the link (e.g., IEEE 802.11 MAC protocol) and the transport (e.g., TCP protocol) layers. However, such mechanisms while improving stability of the system add to its overhead. Moreover, networks that directly provide services to users, as opposed to machine-to-machine networks, need to be consistent in displayed performance trends over long periods of time.

Some of the constraints we impose on the developed system aim to demonstrate that efficient self-optimizing system for wireless home networks can be built without extensive re-design of excising software and hardware components. Therefore, the HCRM must utilize existing re-configuration possibilities provided by the TCP/IP stack to effectively manage data flows and wireless links. It is required to operate on Windows OS, and transmit on popular overcrowded ISM bands using both COTS hardware, such as IEEE 802.11a/g radio interfaces, and experimental Software Defined Radio (SDR) boards. The latter hardware, for instance, commercially available Wireless Open-Access Research Platforms (WARPs) [307], allow exploring additional adaptation possibilities provided by newly developed protocols and dynamic spectrum access methods. For example, we report results of experimentation with one the MAC protocols [363] implemented for WARPs, where optimization decisions are empowered by Hidden Semi-Markov Models (HSMMs) [59]. In this work we also experiment with application layer re-configurations, where selected applications allow for the HCRM to take control over their parameters, thus increasing the state space of promising optimization actions. We also modify the driver of the standard IEEE 802.11a/b/g card to explore the effect of the adjustable channel bandwidth, as this might increase the efficiency of spectrum utilization.

The earlier discussion on the constraints and the evolvability allows defining the type of the optimization state space for the HCRM. It is feasible to have the *parameter-based* actuation state space for run-time adaptation as it is simpler than component-based optimization [21] . Most of the foreseen actions on management of wireless links and data flows, including TCP/IP protocol stack reconfiguration, can be easily parameterized. The requirement of the flexibility of the parameter-based optimization leads to the definition of generic interfaces. For example, a common interface to various wireless technologies, like WLAN or UMTS, enables low-layer plug-and-play reconfigurability of a system [392].

The HCRM is expected to vary its optimization logic (decision and modeling algorithms),

including accommodation for both non-cooperative and cooperative behavior of wireless nodes. For this the *component-based* optimization must be also enabled, where appropriate methods are activated (both through off-line loading and on-line switching) to reach effective RRM decisions in varying operational conditions. Thus, HCRM solves the *meta-optimization task*, where the same resource management problem can resolved using different optimizers or a combination of those. The HCRM operates on *hybrid solution state space*, with actions on networks being primarily parameter-based, and the composition of the optimization logic being component-based. Almost any network management system that foresees large flexibility of its operational logic typically operate on a hybrid solution state space. The first analysis for the need of a hybrid solution state space for wireless TCP/IP networks, though not explicitly formulated in these terms, to our knowledge was suggested by Mähönen et al. in [60].

6.2.2 Finding an Appropriate Architecture

The requirements and constraints formulated above require an architectural approach that evolves beyond traditional radio resource management [371], as high complexity and centralized architecture of the latter is excessive for the smaller and less cooperative home networks. From the other side the level of self-organization typically present in current Wi-Fi home deployments is not sufficient either. Therefore, instead of directly relying on either of these popular designs, we decided to derive the HCRM architecture based on the basic *agent-based* design principle [85], while still taking the best practices from the existing small-scale wireless resource management architectures. We used the *CRM* architecture [60] from the area of cognitive networking [147] to map of the derived agents' functionalities into software modules that form a coherent self-optimizing system.

6.2.2.1 Agent-based Approach for Self-optimization in Wireless Home Networks

A feedback loop in its general meaning, not in the strict sense of control theory [39], lies at the core of many proposals for self-management, such as knowledge plane [393], cognitive wireless networking [147], autonomic networking [148], and self-organized networking [372], see Figure 6.2c-d. The concept of agents that basically implements feedback loops was independently introduced in multiple research fields, e.g., artificial intelligence [21] (Figure 6.2a-b), software engineering [394], and game theory [395]. A set of multiple agents that are not strictly incorporated as part of a single system and are able to interact and reason independently, are studied as multi-agent systems [396]. Due to the fact that agents are common to many research areas, and their basic characteristics have significant commonalities, it is relatively easy to combine and adopt their specific functionalities. In our case we consider mapping of the selected features of the intelligent agents [21] for realization as part of the software agents [394].

In cognitive networking the role of learning is widely discussed and compared to traditional decision-making techniques that rely on pre-determined (trained) models. This discussion maps to the field of artificial intelligence, which, as said, also has the notion of agents, referred to as "intelligent agents". According to Russel and Norwig in [21], there exist *learning* agents that, as the name implies, adapt and learn the optimum behavioral strategy, and *reflex* agents that act according to pre-defined models, see Figure 6.2a-b. The general consensus in the field is that learning agents are typically more complex and provide increased fragility/robustness risks, as compared to traditional reflex agents. Reflex agents can be built upon the results, i.e., models, obtained by the learning agents

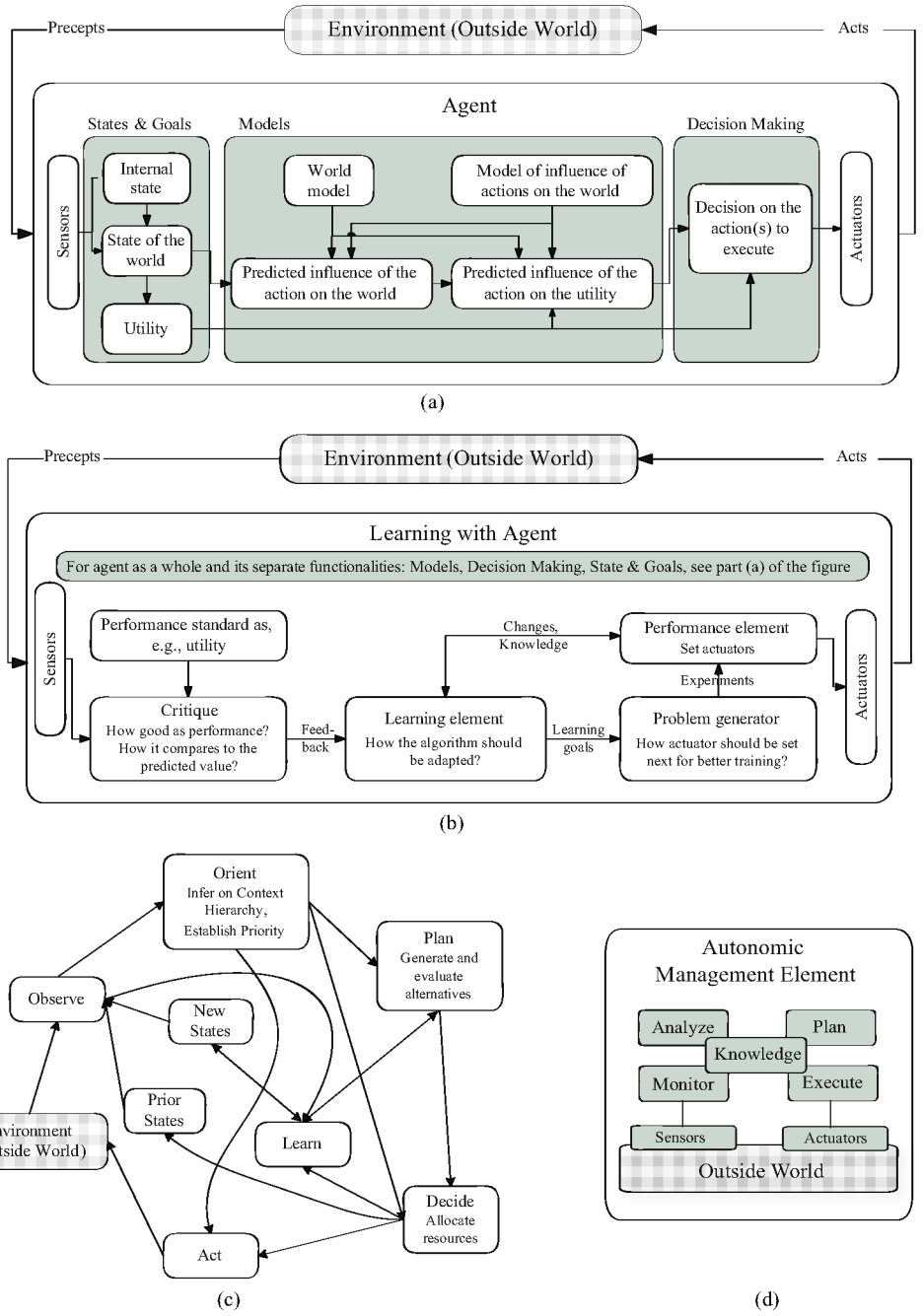


Figure 6.2: Sample control loops for self-management, (a) Generalized representation of an (intelligent) reflex agent, (b) a learning agent, (a,b) are extended from [21], (c) Cognitive Radio Cycle, adapted from [147], (d) an autonomic element, after [148]. An agent as a whole or its internal modules, e.g., models, can incorporate learning functionalities to adapt its behavior. The figure aims to demonstrate that different control loops share a lot in common, and therefore common design practices, such as agent-based architectures, may often be shared between them.

after their training is completed. We consider this viewpoint to be very much applicable to networking. Learning agents should be directly applied to the systems with active users only in cases when they are absolutely needed, e.g., when these systems operate in very diverse and unpredictable conditions to re-utilize results of prior training, or where no prior training is possible. Otherwise, they can be used for building network models, which are later exploited by other optimization modules, i.e., reflex agents.

The concept of an intelligent agent maps well to the notion of agents as understood in software engineering. Generally, a software agent is a module or a program that acts on behalf of a user or another agent. Software agents are reactive, autonomous, persistent, i.e., can self-execute if a set of pre-defined conditions is reached, and capable of coordination and communication with other agents. These characteristics distinguish agents from other software constructs, such as services, components, and objects [396]. In our work we realize modeling, utility, decision making and learning parts of intelligent agents [21] as software agents [394] in the context of wireless home networking. Depending on the interdependencies, timing and communication requirements posed by functionalities of an intelligent agent, they are realized either in a single software agent, or distributed into a system of those.

We argue that agent-based design is highly suitable for the HCRM, as it allows for a system to be flexible, evolvable, and address multiple tradeoffs faced during the self-management process. We advocate that it is beneficial to take an *agent-based goal-oriented view* [85] when designing self-optimization systems. This approach allows mapping of initial requirements to desired system behaviors, understanding their interfacing, and deciding which of those functionalities can be enclosed as agents, thus enabling autonomous system behavior. It is important to note that agents may both decrease and contributed to a system's complexity. From one side they enforce modular design, which reduces the complexity. From the other side, as agents enable the system to react on external events in distributed manner, which may lead the fragility through hidden (mis)behaviors, such as deadlocks or oscillations. The distributed architectures are typically more complex as the centralized ones.

We emphasize that we employ a multi-scale modular agent design, where simple agents can be parts of more complex ones, performing for them specified functionalities. We consider, on a high level of abstraction, that a whole home network solves the optimization problem of dynamically maximizing stakeholders satisfaction, while fulfilling the necessary constraints. Depending on active decision strategy, this problem is solved taking semi-distributed or centralized solutions. The main element of our framework is the HCRM agent that runs on each node of a network (see Figure 6.3), which is discussed in detail in Section 6.2.3. But first we shortly discuss the CRM architecture which strongly influenced the HCRM agent design.

6.2.2.2 The CRM Architecture for Self-optimization in Wireless Home Networks

Self-management and especially self-optimization on lower protocol layers for wireless network is often studied as part of cognitive radio or cognitive wireless networks research. Therefore, we can consider the related architectures in application to our goals. Cognitive Resource Manager (CRM) [60, 367, 368] is one of the alternatives [256, 397–400] that have been suggested as a flexible architecture for implementation of the self-optimization in networking systems. The CRM architecture implements the Mitola's cognitive cycle [147] for wireless networking. It achieves a good balance between being

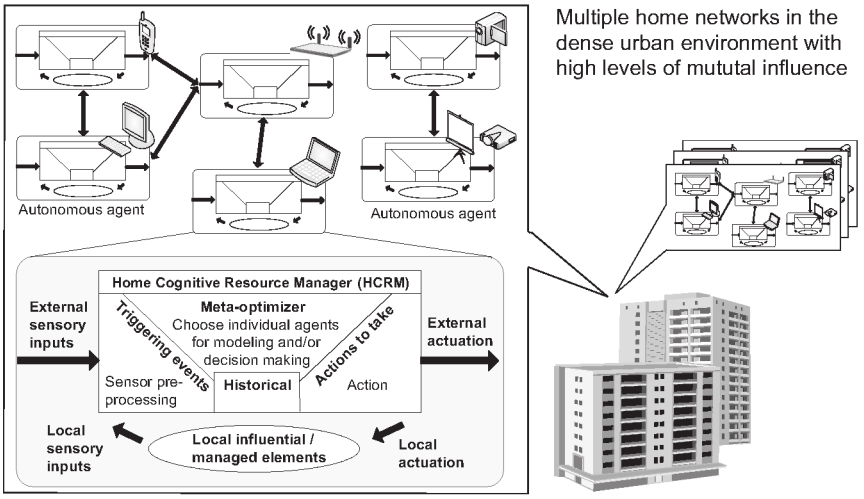


Figure 6.3: Conceptual agent-based design of the HCRM [52].

generic in component structure and accommodating for wireless field specifics, such as explicitly prodiving for interactions with layered protocol stacks or external policy entities. Figure 6.4 shows a slightly adapted version of the CRM architecture, and illustrates the relations between major parameters of the HCRM design task (operational conditions, constraints, solution state space, goals) and the CRM. As we see, the characteristics of the design task straightforwardly map to the items in the CRM architecture, which indicates appropriateness of this architecture to our task. Additionally, the CRM is already structured as an autonomous entity, it is split into a number of interacting components, and it communicates with its operational environment using a limited set of unified interfaces. It is straightforward to do an agent-based decomposition of the CRM. The description below clearly indicates that the CRM can be viewed as a set of coordinating agents, interface-providing agents, agents that answer for cooperation between nodes, and, finally, learning agents that employ particular toolbox algorithms. This maps well to the classification of software agents suggested by Nwana [394].

The CRM architecture distinguishes three interfaces, CAPRI, ULLA and GENI [175], to interact with the protocol stack. It also foresees the existence of external policy entities that contain regulatory constraints to the system. Policies represent a set of rules, constraints and preferences imposed by different network stakeholders, including regulatory bodies. The CRM also defines a module for cooperation with the external network nodes, and the sensory component to get spectrum information on the environment. Internally, the architecture is split into three blocks. The core coordinates all the activities, gets sensory data, and decides on actions to be taken. The knowledge base stores data required for decision making. The toolbox and libraries block contains algorithms required for the perception and the modeling of the outside world, and establishing the causal relations between the actuation state space and the sensory inputs. It can also accommodate for optimization/decision making algorithms. Therefore, the CRM is designed to have a hybrid actuation state space which fits to one of our main architectural requirements. We conclude that the CRM can be adopted as the functional architecture for the HCRM agent. Its structure and interface definition may serve as an architectural constraint that defines the backbone for autonomous radio resource management in WHN

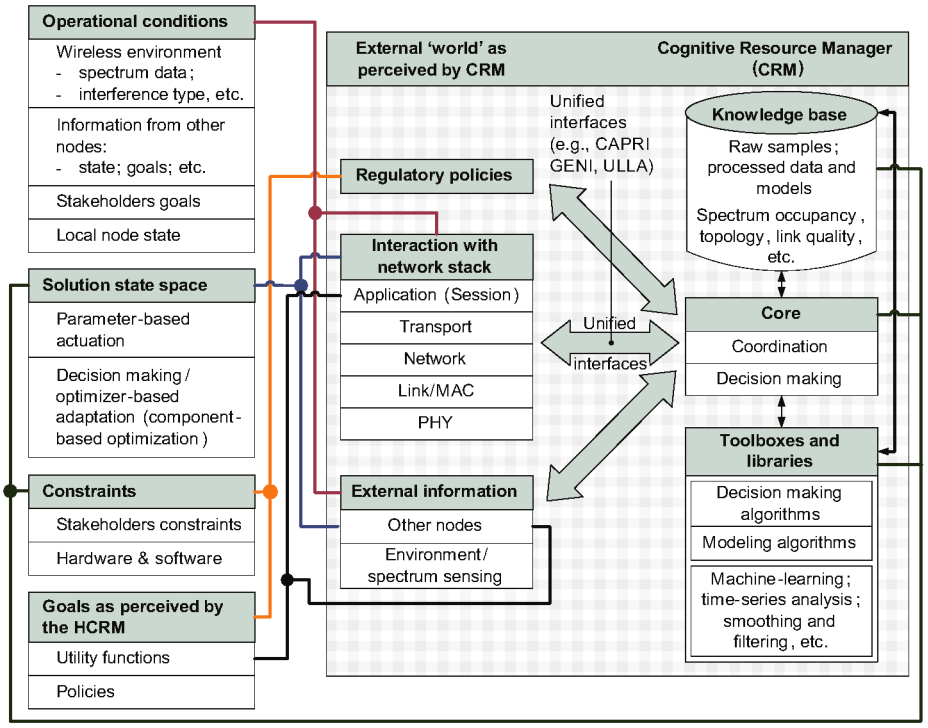


Figure 6.4: The overview of Cognitive Radio Manager (adapted from [60]) and mapping of this architecture to the main variable constituting the task of HCRM design.

environment, thus, in the end de-constraining and opening possibilities for efficient and flexible self-optimization.

6.2.3 Defining HCRM Functionalities and Architectural Decomposition

We continue designing the HCRM framework by further defining its major functionalities, and mapping those to agents and other supporting components, as summarized in Figure 6.5. There exist multiple architectural approaches for decomposing a system into a set of interconnected modules or even agents. One could apply, for example, the RCS (Real-time Control System) architectural approach [401] or goal-oriented requirements engineering [84]. To design our system we utilize the general principles common to these approaches such as minimal interdependency of modules, clear statement of goals, straightforward information flow, simplest possible interfaces, etc. We address the goal of performance maximization by expressing concretely the objectives of network stakeholders. As said, one of the major performance criteria, Quality-of-Experience (QoE), is stated using *utility functions* that depend on Key Performance Indicators (KPIs) of the protocols, such as throughput or delay. We also foresee to directly use selected KPIs to improve the response time of the system and avoid critical system failures. For example, in our implementation if sudden strong interference is detected on a channel, its traffic is immediately moved to an interference-free channel if possible. Simultaneous use of both utilities and selected KPIs adds redundancy into the functionality of the HCRM improving its robustness, while not increasing the overhead as all these inputs have to be reported to the system anyway. We use *policies* to express rules and constraints imposed on the

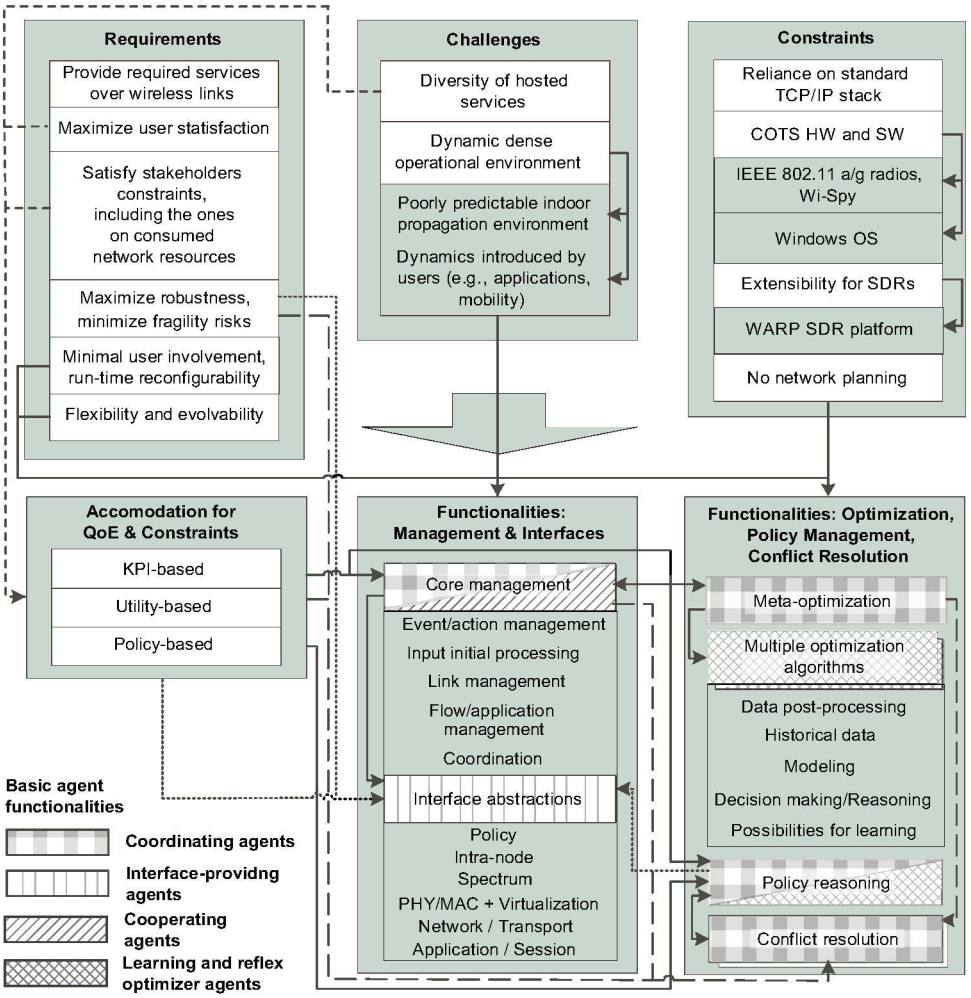


Figure 6.5: The figure shows how major requirements, challenges and constraints for the WHN self-optimizing system are used to derive the list of functionalities to be supported by the HCRM as part of the the design process. The functionalities that are viable to realization as particular types of agents are pattern-coded. The general intuition for developing separate control loops for these functionalities are their ability to independently interact with the environment and other modules providing gains to the overall system.

system by various stakeholders. Network policies, especially in the cognitive wireless domain, tend to have a hierarchical structure that depends on relations between network stakeholders. For example, FCC regulations apply to all wireless networks operating in the USA, while operator policies apply to only their networks and users. In our system policies can be defined either centrally or locally by regulators and users respectively. A policy framework is a logically centralized entity, which is efficient for propagating of slow varying optimization goals if they are to be considered throughout the network. Applications and user priorities are examples of coarse expressions of goals that could be used locally to adjust utility functions.

The support for multiple radio-access technologies, cross-layer optimization and additional spectrum sensing enables the reconfigurability of the network, thus contributing to

the goals of performance improvement and efficient resource usage. This flexibility can be utilized through the definition of unified protocol *interfaces and virtualization* of network adapters. These functionalities are also required to ensure *minimal user involvement* and enable evolvability of the HCRM for further experimentation with self-optimization in WHNs. For instance, to support new wireless technologies, we might require both a common interface enabling low-layer plug-and-play system reconfigurability, as well as an extension to the operational logic for better performance. The *meta-optimizer* functionality, which incorporates multiple performance optimization algorithms with varying requirement on sensing and actuation, also contributes to the above goals. It also further extends opportunities for experimentation on self-optimization.

The *core manager* addresses the robustness of the system by incorporating the key functionalities of the HCRM that realize fail-safe system performance. For instance, it bootstraps and maintains data flows and links, ensures temporary movement of the flows to the control channel during reconfiguration actions, and does basic post-processing of events and data. We decided to introduce a control channel on which an access point operates as part of our solution, as that is a viable way to ensure the basic robustness of CSMA/CA-based wireless networks. The control channel also serves as a backup channel for the traffic and ensures continuous, though generally not quality-wise optimal, user connectivity. The assumption of a control channel availability is valid as mostly at homes nodes have, first, full time AP connectivity, and, second, are equipped with multiple wireless interfaces which allows simultaneous maintenance of both AP-infrastructure and ad-hoc point-to-point links.

The robustness of the HCRM can be further improved through *conflict resolution*. Conflicts can occur due to, e.g., statement of competing goals or race conditions arising due to contradictory decisions reached within the network. Currently our system handles conflicts rather rudimentarily and improvement of the corresponding functionality is definitely possible. Still, even at this level, the corresponding functionality requires implementation at the three HCRM agents, the policy reasoner, the core manager, and the meta-optimizer (see Table 6.2). Conflicts in the HCRM are solved by either fixing the operation logic, or prioritizing events and alternative optimization mechanisms.

The tradeoff between efficiency, flexibility and robustness offered by the HCRM is addressed by allowing its developers and users to decide on the *size of the state space*, the complexity of the *optimization algorithms*, and the *alignment of objectives*, and *information sharing* between nodes. The latter decision defines the amount of cooperation required between nodes and how centralized their control should be. For our system we realized cooperation between nodes using a AP-centric common control channel. However, for data transfer we primarily use point-to-point links. They typically provide higher performance until the nodes are so far apart or encountered interference is so high that it better to use the AP as a relay node. The HCRM establishes ad-hoc links and redirects traffic between ad-hoc and AP links as part of the optimization process.

We have defined the HCRM functionalities in a loosely hierarchical manner so that they can be directly mapped to the hierarchy of interrelated components as depicted in Figure 6.5. The major functionalities (i.e., core management, interfacing, optimization, conflict resolution, and policy reasoning), are realized as agents, as they all encapsulate well-defined functions, act asynchronously at different time-scales, and can benefit from independent operation. Software-wise it is clear that each of these elements requires one or more threads. Moreover, the first two functionalities directly correspond to the core and interfacing blocks of the CRM architecture, and others jointly map to the toolbox

Table 6.2: Summary on conflict resolution in the HCRM.

Module	Incoming events with priorities		Behavior
Policy Reasoner	Policy changes	High	Resolve conflicts within local policy configurations (typically using the local and server ontology reasoners, and related data exchange). Forward the actions that the HCRM is required to perform to the core manager.
Core Manager	1. Policy changes propagated from the policy reasoner	High	If there is a policy-related conflict or critical system events, such as detection of link failures or new flows, all the related traffic is moved to the control channel. Further actions on this traffic are performed after the meta-optimizer decisions.
	2. Critical events (e.g., link failure)	Normal	
	3. Data and utilities estimates	Low	
	4. Actions for the meta-optimizer	Same as the driving events	
Meta-optimizer	Same as for the core manager	Same as for the core manager	The meta-optimizer reacts on the incoming events according to their priorities. If a higher priority event appears while the other events are being handled either by the meta-optimizer or actuated by the core manager, the latter is canceled. Events of the same priority cannot be serviced before execution of the previous event is acknowledged. Only the latest events of the same type (caused by the same sensory data) are considered. The LIFO servicing principle applies.

and knowledge base blocks. The core managers, the meta-optimizer and the interfacing blocks are complex agents that perform multiple functionalities and incorporate other agents, as they require processing of events and actuation at multiple time scales.

6.3 Implementation Architecture

In this section we give an overview of the HCRM design and implementation. Based on the prior analysis, we derived the component-wise design of the HCRM, which is depicted in Figure 6.6. As said, the *interfaces* abstract away differences between protocols of the same families enabling homogeneous interaction format with other protocols, external devices and nodes [175]. Their sensory inputs are the KPIs. The actuators are protocol settings, overall management actions on data flows and ad-hoc wireless links. Monitoring of both low and high layer KPIs allows for adequate and rapid reaction by the HCRM to performance changes. For example, low-layer KPIs, like the link load, allow quick detection of unfavorable channel conditions. However, if the high-layer utility-based metrics indicate user satisfaction with current performance then no actions will be taken, especially if their costs are high. For example in our implementation the channel switching action can take up to few seconds, and, therefore, should be avoided until required with

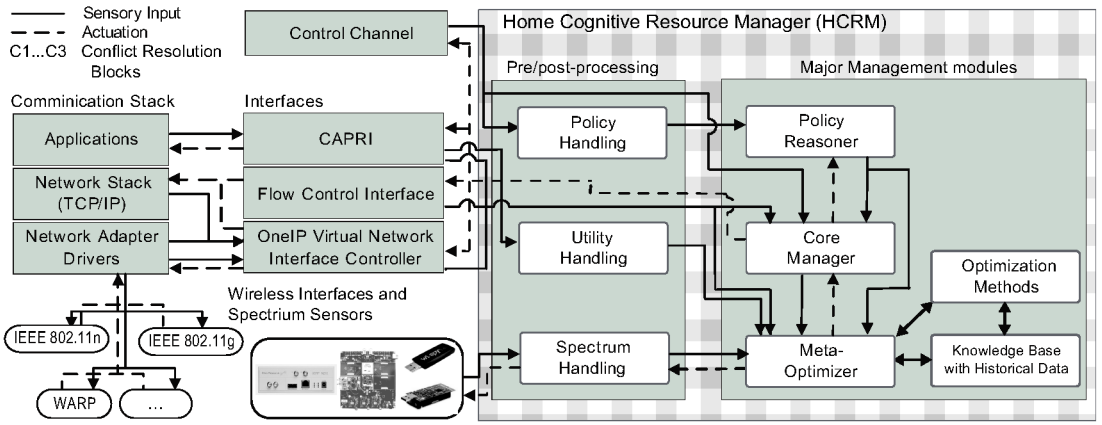


Figure 6.6: Overview of the architecture of the HCRM including the corresponding agents and their interactions with the communication stack via specially developed interfaces [52].

certainty.

Two of the HCRM interfaces are of particular interest. The *virtual driver wrapper*, called OneIP, minimizes user involvement in network interface configuration. It abstracts the interfaces through a single IP address and enables automatic adjustment of the corresponding MAC/PHY parameters. The *Common Application Requirement Interface (CAPRI)* [61,402] allows expressing the Quality-of-Experience requirements provided by applications and their data flows, using dynamically updatable utility functions that are expressed as combinations of KPIs. CAPRI is logically placed on the session layer. The session layer can be perceived as another hourglass waist besides the IP layer of the TCP/IP stack. We believe that this is the right layer for applications to register utilities for their data flows, which can be conveyed to the control plane mechanisms using either a framework like the HCRM or designated QoS protocols, such as IntServ [403] or DiffServ [404]. The CAPRI interface is also used to adjust the settings of individual applications, if the latter support this option. In our implementation we can, for example, adjust maximum datarates of the iperf application [302]. The spectrum sensing functionality is accessed through another interface that provides data in a unified format from a number of external spectrum sensing devices, such as commercially available from MetaGeek low-cost Wi-Spy spectrum analyzers [405] or SDR platforms, such as the USRP2 boards from Ettus Research [406]. Combination of the spectrum sensing interface and the virtual driver wrapper perform the role of the ULLA interface of the original CRM architecture [175].

Policy-based reasoning is introduced on examples of the preference and spectrum policies. The first ones allow setting user and application priorities. Policy preferences are used together with utility functions to express objectives of the stakeholders more fully and weight them according user/application priorities. The spectrum policies regulate usage of frequency bands up to the per-device, user, application and time-of-the-day granularity [407]. Generally policies are employed, either, to impose restrictions on optimization states, e.g., that involve channels forbidden for transmission, or set priorities. The policy reasoning utilized CoRaL language [192], and it is realized using three main building blocks shown in Figure 6.7. These are (a) the centralized server that is the repository of all policies updatable by both local and external stakeholders, (b) the

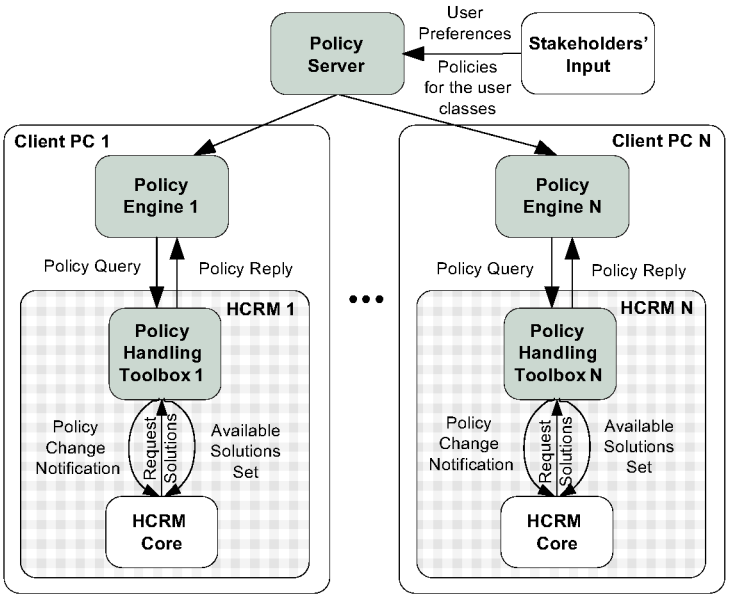


Figure 6.7: The HCRM policy framework includes the central policy server, the policy engine and the policy toolbox residing locally on the client nodes [52].

ontology-based policy reasoner acts as an agent that analyzes incoming policy updates for integrity, and (c) the related module that forwards policy updates to the core manager of the HCRM. For example, it can move the traffic from the forbidden bands to the control channel. Communication with the policy server is done through the control channel.

The *core manager* is one of the main agents of the HCRM. It ensures the robustness of HCRM against critical failures and provides its skeleton functionalities. The agent detects new application flows, establishes and manages ad-hoc point-to-point links. It handles both periodic and asynchronous commands between the protocol stack, other HCRM nodes, spectrum analyzers, the policy framework and the meta-optimizer. For example, it detects new applications and initializes the configuration of their data flows, monitors status of available wireless interfaces, and performs seamless channel handovers. This is done by either moving data flows directly between ad-hoc links, or through the control channel if a new point-to-point link is to be built, which requires time. The ad-hoc links are handled using a sub-module, the link manager that performs intra-node handshakes and probes for stability of links. The core manager also filters the control traffic on which the optimization actions are not viable. It can detect critical events related to ad-hoc links and data, e.g., breaking of a link (no data received) or data flow cancelation, react on them and signal to other agents. Actions and events triggered by this agent are assigned the next highest priority after the policy-based decisions, as they influence not the optimization quality, but the connectivity of the network. The core manager additionally monitors the actuation commands from the meta-optimizer to avoid configuration conflicts (see Table 6.2). The core manager realizes its coordinating functionalities relying on interrupts and buffers.

The HCRM supports both cooperative and non-cooperative node behavior, e.g., to reach a centralized decision on channel usage. In the case of semi-cooperative opera-

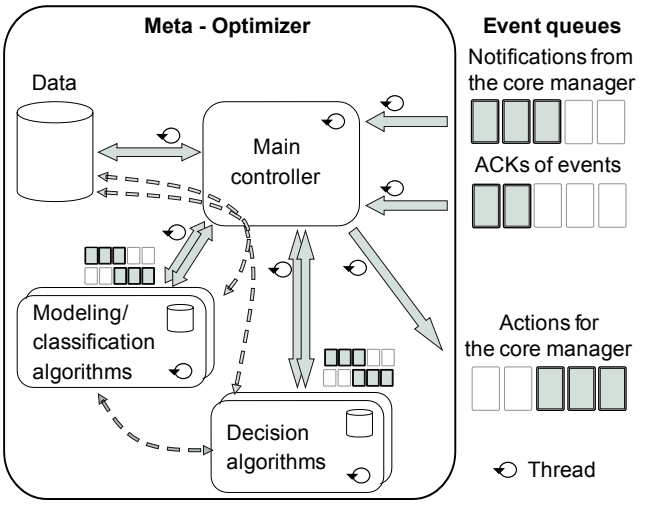


Figure 6.8: Overview of the meta-optimizer agent, showing its main components.

tion only the nodes that share the same point-to-point link exchange link information[3]. Otherwise all nodes share information enabling centralized decisions. Though information exchange is natural in our WHN scenarios as all nodes are sharing the same control channel, it might be suboptimal to pursue centralized optimization due to considerable overhead and considerable delays in large deployments. The decomposition, where nodes jointly decide on transmission frequencies, and determine how to share this capacity locally between the involved nodes, is viable. The HCRM can also opt for both utility- and solely KPI-based reasoning.

The *meta-optimizer* module is another complex agent. It is an event-driven component that enables a hybrid optimization of radio resource usage for wireless home networks. It is modular in design and is capable of dynamically invoking different algorithms to change its functionality. The meta-optimizer stores historical data, can host a number of data post-processing algorithms, maintains queues and corresponding support modules to address conflicts that might arise while deriving optimization decisions. The overview of this agent is given in Figure 6.8. As previously discussed, we choose only to adapt protocol parameters as actions for optimization algorithms. The configuration options for our parameters can be dynamically adjusted depending on, for example, executed optimizers and operational conditions. Optimization modules may rely on Machine Learning (ML) techniques or be of deterministic nature (reflex agents). For the latter modeling and decision making functionalities are typically united in one decision tree. Next, we describe this module more concretely, also providing some of the software implementation details in Appendix E, as this agent serves as a very simplified example of implementation of the meta-optimization concept introduced as part of the proposed optimization-driven methodology.

The meta-optimizer agent went through multiple re-design cycles during the development of the HCRM. It started from the state of being a simple large decision tree with multiple flags to switch between different control options (e.g., wether to enable inter-

[3]We do not discuss here in more detail the bootstrapping of the HCRM, in particular how the different nodes discover information on each other at startup. This can be accomplished using service discovery techniques, with more details given in [408].

node collaboration, or the decision-making partially based on machine learning). This implementation allowed to test and partially enhance the rest of the HCRM modules. In parallel we developed an advanced version of this agent. It complies to the main principles of the CRM architecture, enable flexibility and evolvability of the platform with respect to hosting multiple optimization mechanisms and supporting (e.g., post-processing) techniques. We believe that the current design is sufficiently good, and satisfies our original requirements. However, it lacks the support of reacting to incoming events at very high rates. This is due to the fact that the module is realized as a normal application software, basically in user space, without having special links to the kernel space and the ability to invoke some of its decision-making algorithms directly on the hardware. Therefore the maximum rate of decision making is limited, and no hard real-time guarantees can be given. The recent developments in the SDR domain, e.g., the TRUMP framework [409], circumvent these limitations, by allowing invocation of different simple optimization algorithms directly on the SDR board itself. In the current realization we utilize only some possibilities offered by the TRUMP when experimenting with WARPs. We use a machine learning algorithm to model the spectrum occupancy of multiple channels, and based on these models provide relatively slow, on the order of seconds, additional sensing guidance to the utilized MAC protocol [363].

The meta-optimizer is notified with three classes of events: new data, policy adjustments, and critical system changes. New data is forwarded to selected optimization algorithms, which, depending on the required resources and execution time, might be realized as separate agents or be just normal components within the meta-optimizer. These modules then either suggests some re-configuration actions, or updates (causal/world) models, or does nothing. The incoming data also holds information on the current network state, e.g., configuration of the parameters of the TCP/IP protocols and links. If this new state is different, due to some external adjustments, as assumed by the meta-optimizer then the information on state of the system is updated and optimization actions are invoked. Policy changes also affect the actuation state space, e.g., restrict use of certain transmission channels, or otherwise forbid some parameter configuration values. Policy and critical changes, such as a detection of a new data flow, an update on spectrum availability, or a link failure, force the meta-optimizer to invoke the fastest and the most robust algorithms, such as simple decision trees, to arrive at the optimization solutions as fast as possible.

The optimization-related functionality is split into two parts: modeling/classification and decision-making. The classification or modeling modules establish the “model of the world” that the decision making algorithm utilizes to determine optimization actions. The results of modeling and classification of sensory inputs can also be used to invoke optimization algorithms. For example, if the HCRM observes that utility values are high and the link conditions are moderately good then there is no need to run a re-configuration algorithm, until it is a modeling or learning component. Worsening of the utility values would, however, directly lead to execution of an optimization algorithm.

From our experience we noticed that the logical division of an optimizer into modeling and decision making should not always be followed too literally. Many algorithms integrate modeling, classification and decision-making functionalities as they are designed to work with only certain models or information representations. Therefore, we also support a possibility for designing dedicated optimization algorithms that integrate of the above functionalities, and directly operate on a selected set of raw data inputs.

Each of the optimization algorithms is realized as a separate component using a com-

mon parent class, which provides interfaces for execution of these separate modules, access to their data storage, and an interface to the meta-optimizer. This allows the meta-optimizer to invoke these algorithms in a unified manner, and for them in turn to access (but not modify) the meta-optimizer data. Such approach ensures minimal duplication of information, while data management conflicts are avoided. Individual optimizers and the meta-optimizer interact in the event-driven manner.

As said, the optimization and data processing modules can be encapsulated in the meta-optimizer as simple objects and invoked by its own reasoning thread. They can also serve as separate agents that are initialized using a thread pool of the meta-optimizer. A thread pool is used to limit the number of simultaneously run algorithms. Each agent has its own containers, i.e., queues, that store incoming events and proposed optimization actions. The optimizers communicate by sharing access to their methods through the meta-optimizer. Such a software construct also enables component re-use.

We have also introduced, though we are not currently using, an effectiveness field that characterizes the goodness of an optimizer. This score can influence the choice of the algorithm the meta-optimizer will execute. This parameter is to be estimated based on the long-time history of exploitation of these modules. We also have timers which can halt the optimization algorithm basically forcing it to immediately stop the processing and, if meaningful, declare results.

To allow more flexibility in the parameter state space manipulation, we have also developed a number of classes to store and dynamically modify the value ranges of allowed parameter settings. Based on the layered protocol architecture we have introduced a number of wrappers to ease the manipulation on these objects. Our class hierarchy allows not only to restrict a certain parameter state from being used, e.g., due to changes in a spectrum policy, but also to update at run-time the number of states that a certain parameter can take. As an illustration of the low-level HCRM design, further implementation details on realization of the flexible state space organization, are provided in Appendix E.

6.4 Performance Evaluation

In this section we describe sample results obtained from part of the HCRM demonstrated at the DySPAN 2011 conference [63] (see Figure 6.9), as well as its further extensions. We provide results on three scenarios that 1) run reflex agents based on the pre-determine decision tree algorithms on the COTS hardware, 2) execute a Multi-Arm Bandit (MAB) based learning agent on the COTS hardware, and 3) exploit the HSMM based agent using the WARP SDR boards.

As discussed above, we implemented the HCRM on a Windows platform using low cost COTS hardware such as WLAN IEEE 802.11a/g cards with Atheros chipset and Metageek Wi-Spy spectrum analyzers [405]. In order to perform MAC parameter settings and channel width adaptation on COTS hardware, we use NETGEAR WAG511 32-bit PCMCIA NICs (Atheros chipsets 5211 and 5212) [410] and a customized device driver. Each node is equipped on two radios and, typically, an external spectrum sensor. However, in future by using, e.g., VirtualWiFi [411] the node may rely on one radio. To show the flexibility of the platform we also extended the prototype to support WARP SDR platform [307], which runs a distributed MAC protocol similar to the one described in [363].

The parameters adapted during the autonomous resource management process are the central channel frequency and the channel width (realized as suggested in [412]), the traffic redirection and the traffic shaping using Class-Based Queueing (CBQ), and the

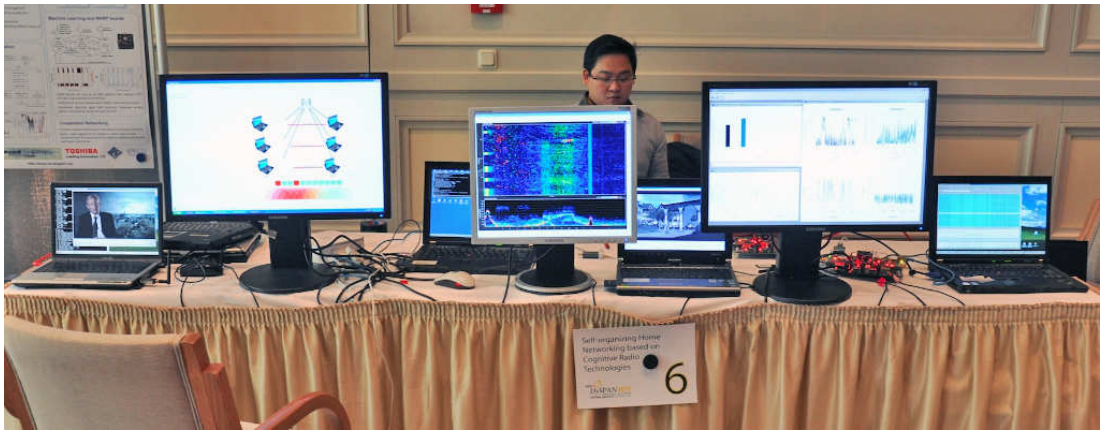


Figure 6.9: The demonstration of the HCRM running at the IEEE DySPAN 2011.

datarate adaptation for selected applications. In the third scenario when WARP platforms are used as secondary interfaces the machine learning based optimizer additionally adjusts the scheme of sensing channels for transmission based on prior observations, as explained in Section 6.4.2. The channel width adaptation option is not supported for the WARP platforms. The influential KPIs are throughput, link load, packet error rate, power spectral density for a certain frequency range (RSSI samples), and, of course, the resulting utilities. Additionally, the nodes exchange their configuration data in a limited manner for more efficient decision making, e.g., to find a common ad-hoc transmission channel.

6.4.1 Scenario with Reflex Agents on the COTS hardware

In order to evaluate the overall HCRM system we set up a test network consisting of four nodes organized into pairs that transmit variable multimedia and download data in the same radio range, see Figure 6.10. The network is prone to external interference. The nodes are equipped with IEEE 802.11g cards and commercial Wi-Spy spectrum analyzers. The scenario aims to show that the HCRM can indeed perform the declared functionalities of ad-hoc point-to-point link setups, seamless handovers, inter-node communication, effective interference avoidance, and cross-layer protocol stack optimization.

First we tested the HCRM acting on the set of configuration parameters that by default is supported by the Windows OS, the COTS Wi-Fi interface cards, even without the presence of sensing devices. The HCRM performed correctly, according to the expectations. The system gained performance advantage over the standard home Wi-Fi deployments due to ad-hoc links setup functionality and flexibility in channel allocation. We skip reporting of these results as this scenario has a lot in common with the more interesting setup discussed below, which allows the HCRM to additionally benefit from spectrum sensing, flexible channel bandwidth assignment, reconfigurability of applications, and utility-based reasoning.

The HCRM executes two decision tree algorithms as agents that run in parallel at the meta-optimizer. These algorithms utilize pre-defined thresholds that were obtained as results of extensive measurements and verified applying the multi-arm bandit ML-based training algorithm. They perform optimization based on either the utility metric or the low-level KPIs. The utility-based optimizer operates on the moving averages of

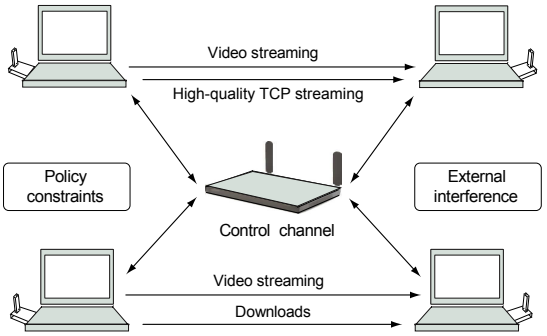


Figure 6.10: The scenario setup with the COTS hardware (Wi-Fi and WiSpy spectrum sensors) running the reflex agents.

the utilities of separate data flows and combines them in a weighted way according to user preferences as indicated through policies. Performance changes are identified by detecting significant drops in the observed utility values, or timeouts. The latter are used to probe if operational conditions have improved and, for example, higher datarates are allowed for an application [61]. (In [61] an alternative utility-based reflex agent is presented.) The purpose of the second agent is to react fast to critical levels of observed KPIs, primarily the link load. This algorithm aims to take actions based on few momentary readings. It can, for example, avoid data losses by moving traffic to the control channel from the point-to-point link in the cases of very high sudden interference. This action cannot be determined so fast using the utility readings, as, first, they come from the upper layer, and, second, the corresponding decisions are taken on the basis of several samples. The KPI-based algorithm operates on very conservative thresholds in order not to overreact and cause too many reconfigurations, or interfere with the utility-based decisions.

Figure 6.10 the HCRM's actions in changing operational conditions using the annotated graphs of goodput and the waterfall spectrogram of energy distribution over 2.4 GHz band. The normalized utility, due to the policy-based application priorities, is

$$U_{\text{netw}} = U_{\text{pair1}} + U_{\text{pair2}} = \big((U_{\text{HD}} + 6U_{\text{D}} + 3U_{\text{UDP}})/10 + U_{\text{UDP}}\big)/2, \tag{6.1}$$

where U_{HD} is the utility of High Definition (HD) video over TCP in the form of the step function, U_{UDP} is the utility of the streaming video over UDP with maximum datarate of 2 Mbps, and U_{D} is the logarithmic utility function of the download application (see Figure 6.12). Channel 1 is used as the control channel and it is therefore forbidden (through policies) to establish ad-hoc links on the involved frequencies. We use VLC video streaming application running over UDP as the video service [413]. Other traffic is generated using iperf [302].

On the right side of spectral diagram of Figure 6.11 we observe that the first pair of nodes starts the video transmission, which is moved after intermediate probing from the control channel to the Channel 11, as it has the least amount of interference. The channel width is assigned to be 5 MHz, as this bandwidth is enough to accommodate for the 2 Mbps datarate required by the video. Later the second pair of nodes also starts a low quality video transmission. According to the other goal pursued by the network on efficient utilization of network resources, the second pair of nodes is co-located with the first pair on Channel 11. In the time period #2 the second pair of nodes introduces two

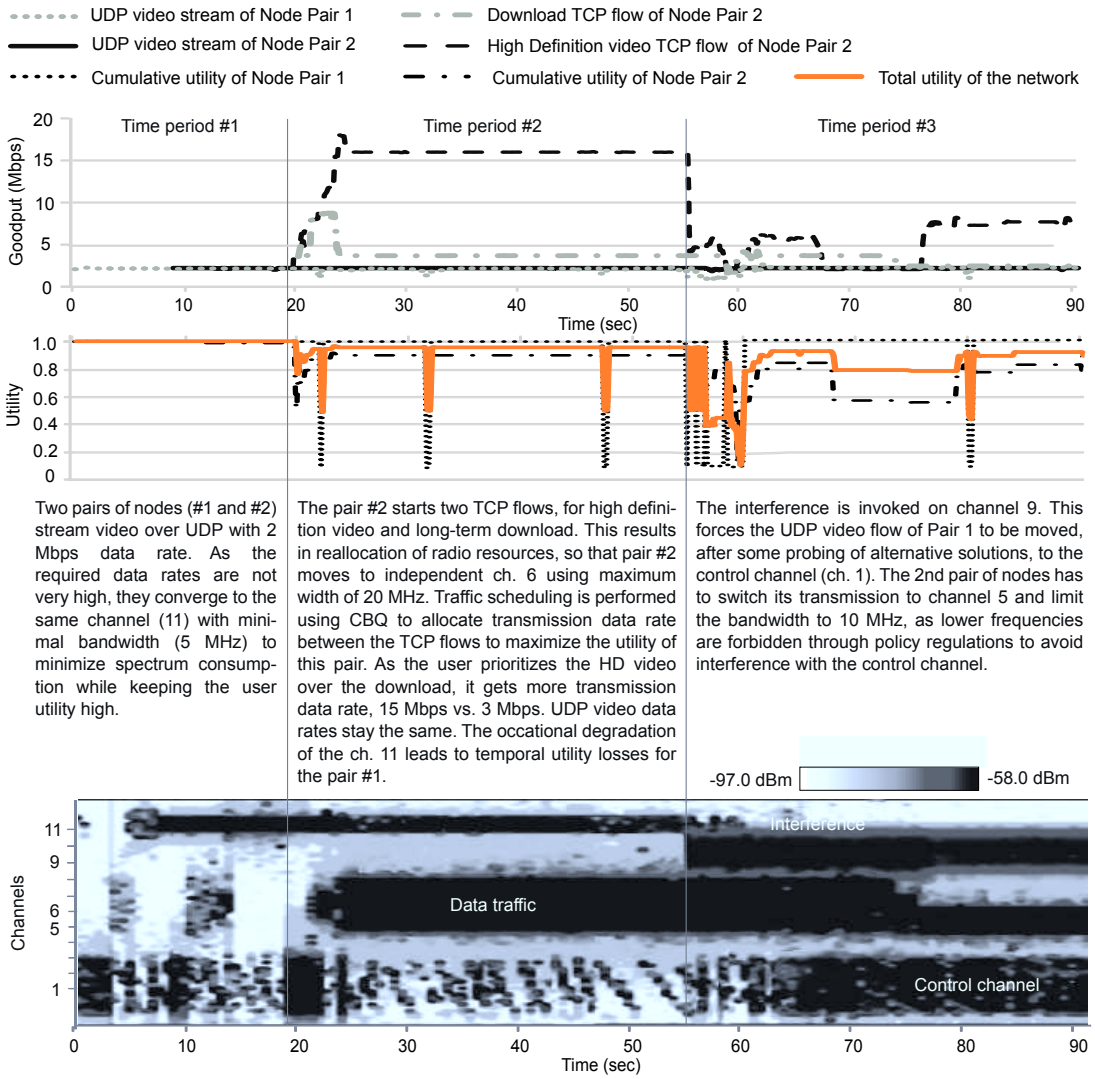


Figure 6.11: Performance of four node network in the dynamic environment with changing number of applications [52]. The figure displays (from top) the goodput achieved by the data flows, the overall network utility and the individual utility functions of the nodes, and the waterfall spectrogram of power distribution in the 2.4 GHz band[1,2].

additional TCP transmissions, one with the higher priority than the other. The HCRM of the transmitting node detects that more bandwidth is required due to degradation in utility, and thus decides to move the transmission to non-overlapping Channel 6 and allocate it a wider bandwidth. First the bandwidth of 10 MHz instead of 5 MHz is probed, and then it is increased further to 20 MHz. This HCRM agent also adapts the application datarate of the low priority TCP-based traffic via the CAPRI interface. In the time period #3 we introduce interference created by a noise generator [414] on the Channel 9. All the HCRM agents detect the interference by evaluating the link load and the packet error rate. As a result the first pair of nodes decides to move, after some unsuccessful probing, to the control channel. The second pair of nodes switches to Channel 5, shrinking the band-

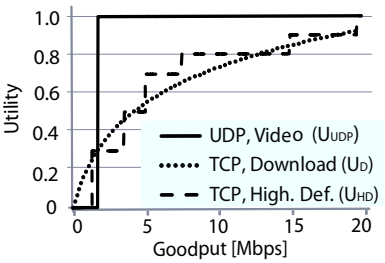


Figure 6.12: Utilities as functions of throughput for applications executed in the reflex agents scenario [52] (see Figure 6.12)

width due to the policy regulations (not to overlap with the control channel). It further reduces the maximum datarate for low priority TCP traffic this time executing CBQ option as no further datarate reduction option are supported by the application. (In our scenario in order to demonstrate both application and network-layer traffic control possibilities, and not to increase the number of active data flows so that the example can be easily followed, we configured the application to support smaller number of available datarates as compared to the network layer. This is a reasonable assumption as often the network layer is more flexible in managing the traffic as compared to many applications.) As we observe in the end of the scenario the network reaches the stable state with sufficiently high overall utility.

It should be noted that both channel expansion/shrinking and channel switching are made automatically by the HCRM without user intervention. These actions are kept transparent for upper layer applications as well. The optimization performed by HCRM does not break any connection at the application level. Short-term reductions in utility for the first pair of nodes are the result of wireless channel variations, and these interruptions are too short to invoke optimization actions, e.g., channel switching. The slow performance of the system is explained not by the additional delay introduced during HCRM decisions, but by hardware constraints of the utilized wireless cards on establishing the ad-hoc connections especially in poor environmental conditions. To our knowledge newer wireless cards can switch channels with much higher speeds, and more careful implementation of the ad-hoc mode on those cards generally overcomes this deficiency. There also exist suggestions on further optimization of these functionalities [415].

The conducted experiments show that the HCRM and its agents are successful in seamlessly and autonomously handling dynamic variations in external interference patterns and changes in the mixture of applications executed by the nodes, while maintaining high levels of the overall network utility.

6.4.2 Experimenting with Machine Learning and the WARP Boards

After demonstrating that the proposed self-optimizing system correctly operates according to the initial requirements to the framework using the COTS hardware and simple optimization algorithms, we experiment with some of the ML-based techniques. As we have discussed machine learning methods can be applied to train models for reflex agent, especially when relations between network performance, observed, KPIs, and the state space of possible actions are difficult to estimate and reflect using standard mathematical models. Reinforcement learning methods are also promising for dynamic and unpredict-

able operational environment. Previously, we have considered control mechanisms that were based on fixed decision trees. They typically provide very good performance in standard scenarios using predefined sequences of actions. However, they are not programmed to take “atypical“ actions that might be optimal only in specific environmental contexts. We have explored machine learning based control mechanisms to operate in non-standard network environments. Additionally, we used these scenarios to demonstrate that the HCRM can easily and effectively accommodate different types of optimizers on example of these methods, as machine learning is often named as potential enablers for self-optimization in wireless networks.

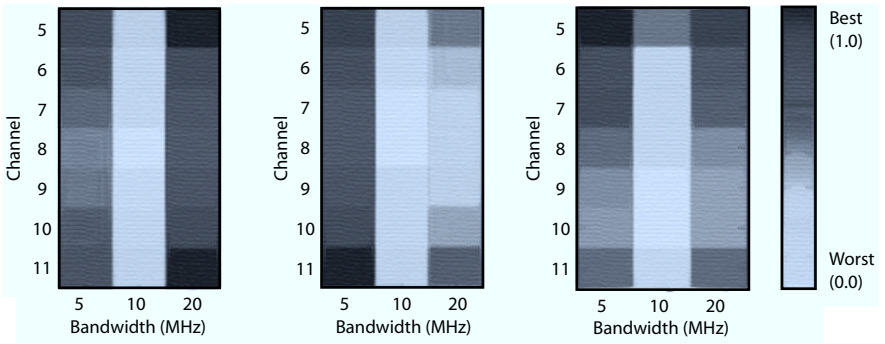
The considered scenarios involve only two nodes and external interference. For clarity we chose to experiment only on channel settings. The ML algorithms we use are realized in MATLAB and R in contrary to the main HCRM code implemented in C++. Therefore, we developed additional TCP socket-based interfaces between the HCRM meta-optimizer and the ML components, with the HCRM acting as the server to those modules. Additionally, the second machine learning algorithm uses WARP boards for build ad-hoc links. In order to use WARPs we have extended the core manager and the virtual driver to support Ethernet as a secondary interface for point-to-point link establishment. This improvement allows the HCRM to interact with any SDR platform that can communicate to the host PC via Ethernet, ensuring high datarate, low delay communication.

6.4.2.1 Multi-arm Bandit for Online Learning

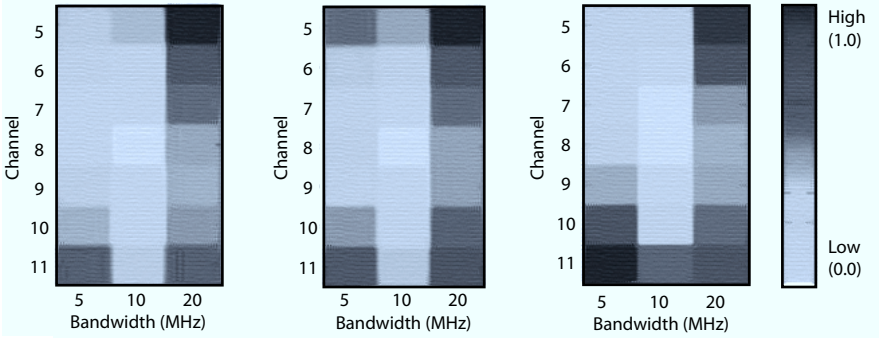
First, we applied a version of the Multi-Armed Bandit (MAB) to determine the best parameter settings and the observed utility in different operational contexts. In the considered MAB algorithm dependencies between the arms are modeled using a Gaussian process [416]. This is a reinforcement learning algorithm, where each action, i.e., parameter re-configuration, has a stochastic reward. With multiple tries the rewards for different actions changes proportionally to the achieved performance, leading to the convergence of the algorithm to the stable near-optimum solutions after a number of iterations.

Our scenario operates on a *spectrum-aware* utility that is a function of goodput and utilized channel bandwidth, with motivation coming from the efficient use of network resources (see Appendix E, Section E.3). The operational context is primarily defined by the positions of the sender and the receiver that are in different rooms of an office building. Challenge of this scenario is that the observed performance does not only depend on the distance between nodes, but also on the encountered external interference, number of separating obstacles (walls), the features of the Wi-Fi interfaces. Not all of these environmental parameters are easy to estimate with commonly available KPIs, and we, therefore, face a scenario with hidden factors. Moreover, relations between these factors and the observed performance are likely to be highly non-linear, and machine learning methods suit well here.

As actuators we considered the channel switching, and the channel width adjustment. The assumption by Chandra et al. [412], which is partially confirmed by our experiments, states that for high distances between nodes in Wi-Fi networks it is beneficial to use narrower channels, e.g., 5 MHz. Such configuration leads to higher received powers, and, thus, improves throughput between nodes in the “gray transmission zone” as compared to employed standard 20 MHz bandwidth. We indeed observed gains if not in the throughput, then in the spectrum-aware utility, for the narrower channel of 5 MHz at the longer distances (e.g., more than two offices apart with one of the separating walls being a



(a) Performance prediction for (from the left to the right) short, far and very far distances, respectively.



(b Prediction uncertainty for (from the left to the right) short, far and very far distances, respectively.

Figure 6.13: Predictions of the multi-arm bandit algorithm after 50 probing samples. Utility is a combination of achieved goodput and utilized channel width.

supporting wall). However, this result applies only for very specific set of propagation conditions and distances. The problem is that with the increasing distance the chances of beacon losses used for the ad-hoc links setup are also increasing. This leads to the higher link setup times, and, in severe cases, might prevent the network from establishing a connection at all. We believe that the part of the encountered problem also lies in the particular design of the Wi-Fi card used and the respective driver realization. The narrower channel width, which in fact implies a higher per hertz transmission power, only partially mitigates this problem.

This “bubble” of the emergency risk was spotted when we ran the MAP algorithm for the first times, and obtained seemingly unexplainable results in which the algorithm got stuck trying the same combinations, and did not converge even after many tries. After, as described above, a feasible explanation was found, we had to adjust the MAB algorithm, basically recursing the development of this module. We added to the reward metric of the MAB algorithm a penalty for the long setup time and inability to establish an ad-hoc link. We then trained the algorithm on several scenarios with nodes situated at different distances from each other. This time the algorithm behaved as expected, and final trends shown in Figure 6.13 were observed in less than 50 iterations. These numbers exceed the state space of 21 combinations, as the same states were re-visited multiple time to ensure the stability of observed conditions. The obtained results in Figure 6.13 match closely our expectations. As the distance between the nodes increases the algorithm switches from favoring the channels with 20 MHz bandwidth to the ones with 5 MHz bandwidth. In

addition, the algorithm detects a weak interference on Channel 8 thus reducing the probability for these nodes to operate on the nearby channels. As the distance between nodes increases even further to five offices between the nodes and having an additional weak interference on Channel 11 (due to beacons from another AP), the MAB algorithm starts to converge slower though still providing the expected result of narrowband transmission on Channel 5.

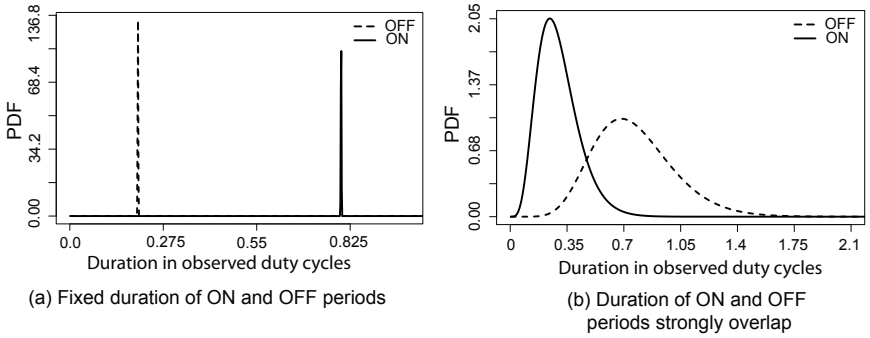
On example of the chosen MAB algorithm we show that the reinforcement learning techniques can indeed be applied in practice for the discovery and the adaptation to an operational environment. However, this algorithm converges quite slowly and, therefore, should be used with care at run-time. As mentioned before, the exploration phase of the reinforcement techniques will lead to momentary performance degradations, thus making them less favorable to a well-design reflex agent that operates on a good "hand-chosen" threshold-based decision tree algorithm. Therefore, the methods such as the MAB algorithm are the most applicable for the initial discovery of the operational environment, and its outputs can be utilized to set the thresholds for the reflex algorithms. This confirms the statement by Russel and Norwig in [21] on the desirability of utilization of the reflex agents for run-time system adaptivity, and employment of sophisticated learning agents for training of those.

6.4.3 Hidden Semi-Markov Models for Estimating Network Activity Patterns

Next, we consider the optimization agent that implements the algorithm for estimations channel occupancy models, i.e., distribution of its network activity patterns, with corresponding power levels clustered as either the OFF or several ON states. Our algorithm is based on Hidden Semi-Markov Models (HSMM) and utilizes history of RSSI samples [59], see Chapter 5, Section 5.4. We use this algorithm to provide long-term guidance for the decentralized spectrum-agile MAC protocol [363], which runs on WARP SDR boards [307] that serve as alternative secondary wireless interface in our implementation. In this scenario we do not use additional sensing devices. All estimations of spectrum conditions are done by the WARP boards.

The original spectrum agile MAC protocol [363] already shows efficient run-time self-optimization behavior based on short-term temporal dynamics of the spectrum. Its performance can be further improved by providing the information on the likely behavior of the channels. One way to provide such a guidance is to find distributions of power levels, i.e., ON/OFF network activity patterns, at different channels. These distributions indicate how long a particular node or a group of those are likely to occupy a certain channel, and how long a media is likely to be idle. They are then converted into dynamically changing weight functions that influence channel access probabilities for the MAC protocol.

We are able to identify the distribution of ON/OFF transmission periods by training at run-time Hidden Semi-Markov Models (HSMM) based on measured power (RSSI) samples. The obtained models of the ON/OFF transmission periods or the extracted duty cycle information is then forwarded to the MAC protocol. Based on this data and the current power spectrum values the MAC protocol adjusts its sensing and transmission patterns. If the distributions of OFF and ON periods have low variance (see Figure 6.14a for the extreme case of such behavior), i.e., the duration of the ON/OFF periods can be determined with high probability, we utilize this data to dynamically adjust channels weights. Otherwise, duty cycle estimates are more suitable as they still allow to reason on typical channel availabilities, while avoiding uncertainties connected with direct exploit-



(a) Fixed duration of ON and OFF periods

(b) Duration of ON and OFF periods strongly overlap

Figure 6.14: The estimated PDFs of the sojourn distributions of the ON and the OFF periods normalized to the average length of a duty cycle as estimated by the proposed HSMM algorithm. The illustrations for the the fixed and the variable duty cycles are given (see Chapter 5, Section 5.4, for more details).

ation of the ON/OFF distributions if the ON and the OFF density functions significantly overlap (see Figure 6.14b). Such utilization of the ON/OFF modeling information is inline with the work of Wellens et al. [356]. In both cases training of the HSMMs is useful. As HSMMs are capable to take the historical information into the account, the clustering of incoming signals into the ON/OFF states can be done more reliably as compared to simpler threshold-based methods, see Chapter 5, Section 5.4. (This is especially valuable if the interference temperature or noise threshold is not available.)

Overall, as demonstrated at [63], and shown in Figure 6.15 the HSMM-based ML algorithm can be successfully employed online to enhance the MAC performance. The figure displays a sample performance of the scenario, where a pair of the HCRM nodes tries to transmit on either of the three heavily interfered channels. This interference is caused by the primary nodes, i.e., the HCRM transmissions cannot disturb them. (The primary traffic is generated by the TelosB nodes as described in Section 5.4 and Appendix D). As we see even when the primary nodes transmit almost simultaneously on Channels 1 and 11, still, the HCRM nodes utilize both channels to minimize the control channel occupancy. (In this scenario the control channel is configured to operate on the 5 GHz ISM band). One can observe that the algorithm takes advantage of the small differences in the transmission schedules of the primaries. For example, the top of the figure shows that the transmission from Channel 1 is preemptively moved to Channel 11 before the primary interfere becomes active. Channel 6 is almost never used due to high duty cycle of the encountered interference there.

6.5 Discussion

Before concluding this chapter we summarize our experience on the development of self-optimization system for wireless home networks.

Software development aspects. Our experience with the HCRM framework confirmed the main principles of software design. We encountered recursiveness in the HCRM development lifecycle regarding both the whole system and its individual modules. Clear interface abstraction and careful class hierarchy design fully payed off. For instance, in the early development stage we had to urgently adjust a module and its interfaces. As a result later we spent considerable time developing additional function wrappers. On the

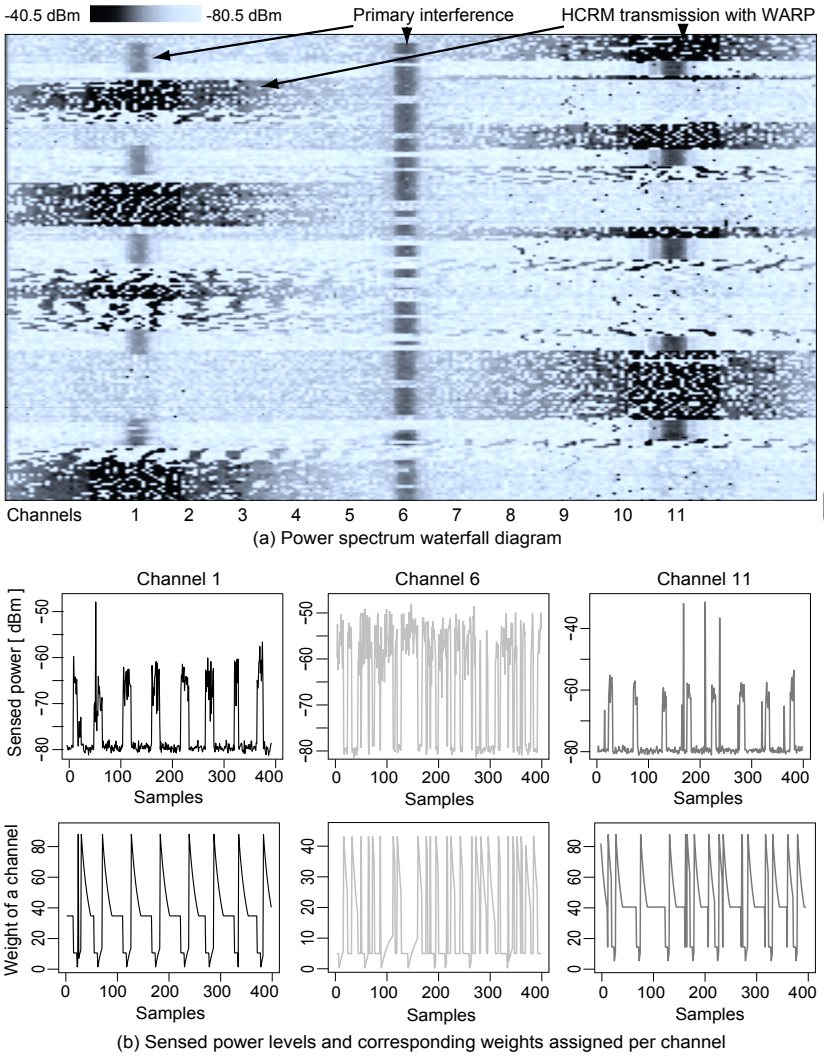


Figure 6.15: Illustrating the performance of the MAC protocol running on WARP boards that utilizes the ON/OFF network activity estimates based on HSMMs provided by the HCRM ML optimizer. Figure (a) displays the sample temporal behavior of the primary interfering nodes, and the secondary HCRM nodes. Figure (b) shows sample sensing results for Channels 1, 6, and 11, as well as the resulting weighing of these channels that affect their access probability.

contrary, extension of the HCRM to support the WARP SDR platform empowered by the OneIP interface happened rapidly and almost without any implementation problems due to the careful design of this component.

The application of the *optimization-based agent-centric* view on the HCRM design helped to uncover and explain some hidden design risks (e.g., robustness considerations, conflict resolution mechanisms) that were not considered at the initial stage of the framework development. Our prototype fulfills the design goals state in the beginning of the chapter. It provides both the seamless autonomous radio-resource management for home networks, allows for easy extension with new optimization algorithms and support for

new SDR boards. Still, we are not completely satisfied with the built system, as it is somewhat bulky and, to a certain extent, its implementation is quite complex. During the implementation of the HCRM we also encountered that the problem of conflict resolution concerning, e.g., event handling, actuation and prioritization between optimizers is much more complex and interesting than we expected and calls for additional research. We plan to revisit the design of the system in the future in the context of developing efficient self-optimized femtocell solutions following the same design and architectural principles as presented in this chapter, as well as in the thesis overall. Specifically, using the approach proposed in Chapter 2 we want to address at the design phase the issues of maintainability and extensibility keeping in mind the clear tradeoff in system complexity and robustness.

External hardware and software impact. Both the core of the system, as well as optimization algorithms required changes due to limitations or unexpected behavior of the COTS hardware (unstable ad-hoc link setup process) and features of Windows XP OS (e.g., there are no packet drops in low layer protocol buffers, and from the traffic shaping mechanisms only CBQ is easily available). Clearly Windows, being a non real-time OS, in combination with COST hardware cannot provide time guarantees on reaction of the system on incoming events. Therefore, we faced a risk of temporally inconsistent network configuration. Those could be accommodated in optimization mechanisms either through thorough testing in case of reflex solutions, or by introducing hidden influence factors in the structure of learning agents. Additionally, it is also possible to extend the HCRM to invoke some optimization algorithms on SDR platforms, thus providing tighter and faster control over MAC/PHY mechanisms.

Reflex and learning agents, RRM and CRM. Our experience when developing the HCRM confirms that reflex agents are more efficient than learning agents in cases when a network operates in a small range of predictable operational conditions. However, learning components are important when they can be used as training mechanisms to shape the reflex algorithms. In these terms the CRM architecture as opposed to traditional RRM architectures shows its strength, especially if it is extended with appropriate update and module loading mechanisms. However, the tradeoff between efficiency of simple resource management solutions, which have careful modular design and can be easily upgraded, and complex cognitive approaches that are more flexible at runtime, and provide better control over a network at the price of greater complexity, has to be studied further still.

6.6 Conclusions

In this chapter we have applied the proposed optimization-driven development methodology for the design and implementation of the self-optimization resource management system for wireless home networks, called Home Cognitive Resource Manager (HCRM). Fulfilling the requirements of multiple stakeholders in a dynamical environment it delivers high and robust performance. At the same time the framework design allows for easy extension and experimentation with new SDR platforms and optimization algorithms, including machine-learning approaches. The HCRM prototype and its evaluation confirms that the combination of an agent-based approach and CRM architecture is highly suitable for the development of evolvable self-optimized systems, especially the ones falling in the domain of cognitive wireless research.

7

Conclusions and Future Work

In this chapter we provide concise conclusions for the thesis and outline possible directions for future work. We comment only shortly the key contributions without actual detailed summaries, since those we already given in the ends of each preceding chapters.

7.1 Summary of Contributions

Contributions of this thesis are related to the problem of *optimization of wireless networked systems* either at the design or run-time management phases. First, we proposed a novel approach on model-driven optimization-centric methodology and its formalization using *category theory*. Secondly, we applied elements of this methodology to create a framework for semi-autonomous pre-deployment protocol stack optimization for wireless sensor network scenarios. In particular, we were first to apply *ontology-driven knowledge base* to capture the optimization problem definition and context, and solve it using feedback from prior network deployments while incorporating knowledge from theoretical models of protocol performance. As part of this work we were also the first to conceptualize in detail the task of software configuration and composition for white home appliances and wireless sensor networks using ontologies. The derived ontologies separately consider behavioral and structural elements of network design, account for specifics of operation environments, and accommodate for high-level characteristics of both hardware and software entities.

Third, we enriched popular *graph-based network models*, such as connectivity and conflict graphs, with *aspects of network dynamics*. For example, we showed that correlation coefficients between protocol parameters values and obtained performance characteristics are a robust and effective metric for context identification. This metric enables efficient application of graph-based models for network optimization in case of varying channel conditions, traffic types, and optimization state space, which has not been studied in the literature before. The parameter correlation coefficient metric is also closely related to the dynamic range of network motifs that can be encountered at a given operational point. We have studied this aspect as well, and discovered that for a given operational context the set of encountered network motifs is rather limited, which is again important for consideration of these models for the optimization process. We also applied hidden semi-Markov models to estimate ON-OFF network activity patterns based of spectrum power measurements received by heterogenous sensing nodes. We took the state-of-the-art in this field further by studying applicability of these models for online use on realistic empirical data. We have also realized a MAC protocol for WARP software defined radio boards that uses these models.

Fourth, we studied in detail how additional information on structure of a network, as well as size of the solution state space influences efficiency of *metaheuristic* methods, especially simulated annealing, when applied to the problems of network planning

and parameter-based cross-layer optimization. Our experiments indicate that for the off-line tasks careful formulation of the optimization problems, including utility statements, combined with naive or minimally adapted metaheuristic optimizers can often be more efficient then efforts spent on the heavily problem-specific customization of a metaheuristic search, which is executed on a poorly defined state space. For the online optimization metaheuristic modification are more viable, and can bring considerable performance gains, while causing very limited momentary user disturbance. As part of considered modifications to the basic simulated annealing, we explored both utilization of correlation coefficient metrics and Bayesian graphs to capture causal parameter-KPI relations.

Finally, we were the first to demonstrate in practice the applicability of *cognitive radio principles for home networking*. We designed and implemented a system that integrates utility-based network self-optimization, and cross-layer radio resource management, including dynamic channel access. Its software structure imposes "constraints that deconstrain", providing support for multiple radio-ends and optimization algorithms in such structured manner that it makes the system valuable for both robust user servicing and flexible and extendible experimentation.

This thesis has a system design and modeling focus, topics which are not easy to subject to quantitative analysis. However, throughout the thesis parts of the newly proposed models or insights were implemented as alternatives or improvements to existing mechanisms. Such works typically resulted in achieved more than 30% performance improvements, which supports strongly the validity of the ideas that formed the core of the thesis. Additionally, most of the proposed systems were prototyped, and, therefore, validated not only using simulations, but also empirically.

7.2 Future Work

We have contributed to several research directions in wireless networks, and undoubtedly further marginal improvement are possible for all of the proposed approaches. However, the largest impact can result from the work in three areas. First, the idea of joint consideration for network design and run-time management should be promising especially for the industry, where the question of clean slate design vs. hardware updates and gradual improvements through software updates is particularly important. To answer it correctly the decomposition and optimization approaches to network development lifecycle have to be revisited to account for specifics of network system design as compared to more established traditional software design. We believe that category theory can be very promising for formalization of this problem. Additionally, the research on evaluation metrics beyond performance maximization, such as robustness and evolvability, is required in directions of formulation, evaluation and explicit incorporation into the reasoning process. Second, application of ontology-enabled knowledge base for network design and management remains heavily under-explored area, with limited contributions concerning such specific areas as policy and low-level protocol layer management. We believe that conceptualization of the networking through ontologies can really enable such power visions as knowledge plane and cognitive networking. Third, simple probabilistic network models, especially graph-based approximation, that take into account network dynamics should be explored further in the context of different wireless networks, including both Wi-Fi and LTE.

A

Illustrating Utility – Optimizer Tradeoff

Following the discussion in Chapter 2, in this appendix on the examples two utility-based network management mechanisms we illustrate the utility – optimizer tradeoff of the optimization task statement. We aim to show that the balanced design of utilities that take into the consideration the incurred computational and communication overheads, as well as the capabilities of an optimizer can lead to significant performance gains as compared to the traditional approaches. We consider the case studies based on the numerical simulations of the class-based advanced queue management, and the workload/cost balancing between datacenters. Our experiments show that the proposed approaches are robust and can lead up to 30% performance gain.

A.1 Introduction

As discussed, complexity of network management is constantly growing along with better understanding that overprovisioning cannot solve all problems of the broadband Internet especially with regard to delay-sensitive traffic, not to mention the last hop wireless networking [417–419]. The discussion on viability of utility-based Quality-of-Experience (QoE) provisioning and pricing (with strong economical considerations) gains momentum, with the main arguments being the revenue model, as well as the cost of implementation and deployment of new techniques. The latter argument is likely to become less relevant with the development of software-define networking that is expected to considerably reduce the update costs [420]. In this appendix we support the argument for the utility-enabled network service differentiation. We advocate that correctly formulated and simple utility-based models can bring significant additional capacity gains (in terms of QoE), without posing excessive computational and networking overhead. Clearly, the consideration for utility and policy definitions allow businesses to define more unambiguously what are the goals of the company to the network management and the operational levels. They also let users to state their time-varying goals more precisely. Additionally, this allows for more dynamic change of these rules of the game. Moreover, reformulation of utilities on the *session-level* basis is likely to further boost QoE of network stakeholders, as this approach allows to consider continuously varying objectives induced by the changes in the mixture of data flows executed by the users and their varying relative importance. As part of this discussion, we next suggest possible forms of utilities for different caching-based and downloading applications. We use these utilities and their approximations to show that achieved performance gains are high, and are likely to overweight the additional overhead imposed on a network [47][1].

[1] We shall not focus here on the problem of efficient transporting and encoding of utility abstractions. However, there exist a number of approaches that can be used to address this issue as discussed, for example, in [421, 422].

A.2 On Utilities and Optimizers

Network Utility Maximization (NUM) [15, 22, 423] is one of the most popular general formulations of a network optimization task, see Chapter 2. Utility-based maximization, though potentially very beneficial, can impose high overhead being performed online. For example, a router has to potentially handle tens of thousands of data flows with continuously changing utilities of both convex and non-convex shapes. The NUM problem statement has to be simplified and appropriately adjusted for a particular optimization task. In this appendix we jointly consider on two ways to abstract an optimization problem that is *approximation of the utility functions*, and utilization of *more primitive optimizers*. One way to gradually approximate utilities is to use the hierarchical approach. The current Internet is a key example, though not formalized, of the hierarchical approach. There we have a clear distinction between the ISP policy-based goal statements, user performance metrics, and the KPIs-based optimization at distinct protocols layers. The hierarchical approach is enabled through meta-optimization as discussed in Chapter 2. Many of the network entities (protocols performing congestion control, queue management, load balancers, etc.) can be viewed as optimizers that operate on mostly predefined and fixed utility functions. The logic of these can be changed if needed.

The effectiveness of the optimization solution can be quantified by the optimization gap that in relevance to our task can be defined as the difference between the maximal utility achieved between the ideal optimizer on the ideal utility set and the practical optimizer for simplified utility functionals

$$\epsilon = |\underset{\Theta}{U(\mathbf{u}_i(\mathbf{a}_i))} - \underset{\Theta'}{U'(\mathbf{u}'_i(\mathbf{a}'_i))}|, \tag{A.1}$$

where ϵ is the obtained difference in utility, the first part of the function denotes the ideal solution, and the second part reflects the optimization solution under study. The utility functional U depends on the set of utility functions $\mathbf{u}_i$, which, in turn, are functions of attributes/KPIs $\mathbf{a}_i$. The optimizer used is denoted with Θ. In practice the "ideal" variables in this equation are often substitute by the best known ones.

We are clearly facing a joint optimization problem, where each of the above constituents influences on the choice of others. For example, in order for the classical techniques, like convex optimization, to be applied the utility functions typically need to be convex, monotonic and twice continuously differentiable. Their functional is often assumed to be a weighted aggregation. Such hard constraints allow for fast convergence of the algorithm in the case of a centralized solution, and guarantee convergence in the distributed case [15]. If we allow for a arbitrary utility functions we would require, for example, metaheuristics based optimizers, which do not guarantee convergence to the optimum, but in practice typically perform well [46, 424]. The choice of the utility functional affects both on the type of optimization solution we want to achieve (e.g. maximization of utility vs. fairness), and the stability of the obtained results [425]. Therefore, depending on the problem, the choice of both utility abstractions and optimization techniques has to be addressed carefully. Further in this appendix, we use numerical simulations to show how the basic performance metrics, resembling the ones used today, can result in large optimality gaps that can be significantly reduced using the alternative simple utility-based approaches we propose. They do not cause large additional network overhead.

A.3 Evaluation

There have been several attempts to incorporate utilities into existing network infrastructure reusing existing network mechanisms. Well-known examples that implement Network Utility Maximization (NUM) principle are the TCP flow control mechanisms such as FAST or re-ECN [426, 427]. We show that two classical network management tasks, namely the queue management and the power-aware load balancing, can also benefit from utility abstractions.

A.3.1 Class-Based Queuing

For this scenario we, first, derive the model how application-layer utility can be expressed through TCP metrics. Then we utilize this model to generate utilities for data flows which need to be services using class-based queuing approach.

A.3.1.1 Application Utility Functional through TCP metrics

As discussed in Chapter 2 the major components defining the application layer utility are cost, time-to-wait, and quality-of-content. The mapping of requirements of an individual application to the transport layer depends heavily on the underlaying transport mechanism[2]. The TCP-based traffic is influenced by two KPIs, *throughput* that varies over time and the *time* that is required to establish a TCP connection, i.e., the utility of the data flow over this transport protocol can be expressed as

$$U_{DF} = f(D_s, Q, G, S), \tag{A.2}$$

where U_{DF} is the utility of the data flow, D_s is the connection setup time, Q is the service quality indication, S is the size of the data to be transferred, and G is the achieved goodput. Kelly *et. al* in [22] showed that TCP optimizes for elastic traffic. However, depending on the application importance the logarithmic functions can be scaled differently. In Figure A.1 the utility for a web page is more steep than of a file download reflecting user priorities. The requirements for the connection establishment delay are similar as the user tolerates much longer setup delay for a large file than for a news page. Within the same application the utility forms can change on the per-flow basis. For example, due to the different value of the content, on a single web-page a value of the text might be sufficiently higher than of the picture (as we typically see today). However, the provider of the content, based on the price the user pays, might adjust this utility to fetch the advertisement banner first. The trade-off in the value of the content between users and providers is another aspect that the utility-based approach allows to formalize. One example of the specific utility definition of the TCP data flow is

$$U_{DF} = \left[\gamma_1 \cdot \log(-D_s + \gamma_2)\right]^+ + \left[\frac{Q \cdot S}{\phi_1 \cdot \log(G + \phi_2)}\right]^+, \tag{A.3}$$

where D_s is the maximum allowable setup time, and the γ_1, γ_2, ϕ_1, ϕ_2 are scaling coefficients that allow to express user preferences. Other terms are as defined in (A.2). The equation (A.3) also allows to impose the throughput based constraints on the utility function. For example, IPTV functions only if a certain level of throughput is provided. Such

[2] One possible interface to indicate application requirements to the transport layer in terms of utility functions was proposed Riihijärvi *et. al* in [61].

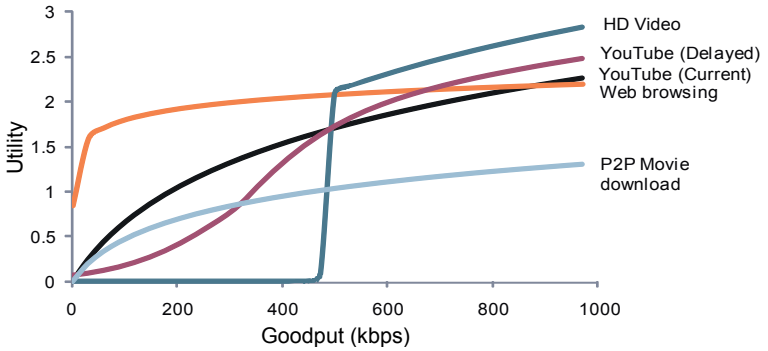


Figure A.1: Possible shapes of the utilities for download-based traffic [47].

applications do not gradually benefit from the increase in throughput, but rather behave in a step-like manner typical for streaming applications. There can also exist mixed application types where the classical logarithmic form is not followed for the whole bandwidth range. For example, a delay in the middle of download of an on-demand video might require a higher bandwidth for a part of the download so that the user does not have to wait in the middle of watching of a video stream, see Figure A.1 “YouTube (Delayed)”.

Other popular application types like voice or videoconferencing rely on UDP transport protocol that is mapped in the throughput domain as a step function due to the codec rates [428]. In terms of the utilities the UDP traffic is generally more difficult to manage than TCP as it has more strict requirements on other transport-level attribute dimensions, such as delay, reliability and goodput. For example, for gaming applications, that often rely on UDP, the propagation delay for the server is particular important [429]. For wireless networking utility-mapping to the link and physical layers is useful, as here significant packet losses can be minimized using low-layer network management.

A.3.1.2 Results

In the queue management scenario we assign data flows to the queues of an interface and allocate these queues a certain bandwidth, as it is done in class-based queuing, e.g., CBWFQ [430]. The overall optimization task is to maximize the overall user performance. After (A.3) the NUM problem can be formulated as

$$U_{CBQ} = \max \sum_{i=1}^{N_u} \left[\phi_{1,i} \log(B_i + \phi_{2,i})\right]^+, \tag{A.4}$$

where N_u is the total number of users to be served by the router, B_i is the bandwidth assigned to a queue (an individual flow or a group of data flows), and others as in (A.2). The $\phi_{1,2}$ coefficients are related to the executed application type.

Class-based queuing techniques typically allow for a very large number of queues defined per interface, e.g. a CBWQF-enabled router supports up to 64 queues per policy. In principle, there is no technical limit on the number of queues that can be defined. Therefore, for the reference solution of the NUM problem we allow for each data flow to have a separate queue in order to obtain the optimal results. However, in practice, such an approach is not feasible because of the computational overhead. Clustering of the data flows together leads to smaller number of queues, which reduces the processing burden. The clustering can be regarded as abstraction of individual flows into groups.

Table A.1: Simulation parameters for the CBQ scenario.

Parameter	IPTV	YouTube	Updates	HTTP
Traffic mix	10%	20%	20%	50%
$\mu_{\phi 1}$	1	0.75	0.5	0.25
$\sigma^2_{\phi 1}$	0.25	0.25	0.25	0.25
$\mu_{\phi 2}$ (kbps)	600	400	200	0
$\sigma^2_{\phi 2}$ (kbps)	800	400	400	200

We compare three utility abstractions that define data flow clusters, that are the *fair queuing* approach, the *application-based* clustering, and the *utility-based* clustering. The fair queuing algorithm treats equally all the flows assigning then the same bandwidth, i.e., it requires only one queue. Being extremely simple, it does not support QoS metrics, which might lead to severe degradation of user experience, e.g., in the case of streaming applications. The application-based queuing assigns flows to the queues based on their application type. This approach is widely used today. However, it cannot accommodate for the varying importance of the data flows. We propose to directly utilize utility functions to sort data flows in queues. In this example we group the functions coefficients into four clusters using a fast fuzzy c-mean clustering method [431]. In our approach we cluster to group together the most similar data flows, which leads to the efficient queue management. The performance of these three utility abstractions along the optimal results are shown in Figure A.2. The utility form-based method enables up to obtain 30% performance improvement compared with the conventional methods under study. This abstraction allows achieving on average 85% performance from the maximum, whereas the alternatives lead to about 60% of the optimum performance.

The job of the optimizers is to allocate bandwidth to the queues. As we cannot guarantee the convexity of the resulting utility functionals, we employ the constrained nonlinear optimizer that also used the genetic algorithm to overcome local maxima [208, 275]. Previously we used the ideal optimizer that is not constrained in time to find the optimal solution. Figure A.2c demonstrates the performance degradation experienced when the timing constraint of 500 msec is imposed on the optimizer. The largest decrease is experienced if we try to do per-flow bandwidth assignment, as the algorithm simply cannot find the good local maxima in a given time. Other approaches perform better, suffering on average only 10% performance decrease as compared to the use of the ideal solution.

We simulated the router interface that has 100 Mbps of bandwidth in MATLAB. There are up to 1000 data flows to be serviced simultaneously. We consider the mixture of four application types, high-demanding IPTV applications, semi-delay tolerant YouTube traffic, HTTP data with high regard to low traffic rates, and low priority downloads, as summarized in Table A.1. As discussed we assume that utility of a data flow depends on both shakeholders momentary priorities and the application type, which is reflected through the normal distributions of coefficients ϕ_1 and ϕ_2 in (A.4).

A.3.2 Power-Aware Load Balancing

We now show the viability of utility abstractions for the power-aware load balancing using a simplified network model. The utility functional takes into account interests of both users and datacenter providers. Recent studies [432, 433] have highlighted the significant

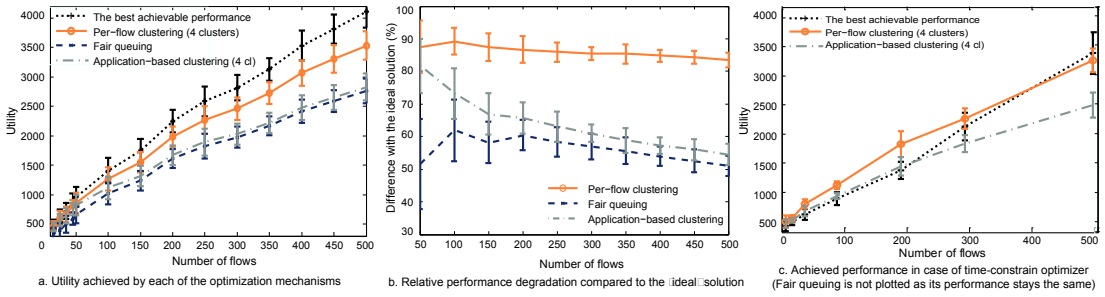


Figure A.2: Performance of optimizers with related utility abstractions for the class-based queue management problem [47]. (Each simulation setup was executed 50 times; the error bars indicate standard deviation of the values.)

impact of energy consumption on the Total Cost of Ownership (TCO) of datacenters. The figures show that energy cost may outrun the fixed cost of initial investment in the datacenter infrastructure. This shift in the perception of energy cost has driven new research in the area of energy-aware load balancing among datacenters with heterogeneous cost drivers. We propose to further extend this view by taking into account the fixed cost of datacenters and the performance implications of different user load scenarios.

Classic economic models optimize by means of minimizing the marginal unit cost. We can model the total cost of a datacenter (DC) as being composed of fixed cost, e.g. the price of servers and maintenance costs, and variable costs such as the energy costs imposed by serving a number of users[3]. The cost per unit can be defined as a utility function with

$$U_{\mathrm{DC}}(N_{\mathrm{DC}}) = -\frac{F_{\mathrm{DC}} + P_{\mathrm{DC}}(N_{\mathrm{DC}}) \times c_{\mathrm{DC}}}{N_{\mathrm{DC}}}, \tag{A.5}$$

where F_{DC} accounts for the fixed cost, c_{DC} is the cost per unit of power consumption and N_{DC} are the number of users served by the particular datacenter. As depicted in Figure A.3a, increasing the load on the datacenter will increase its efficiency due to the positive economies of scale, but the subadditivity may be reversed if the additional energy costs per user exceed the marginal fixed costs. In [434], a typical scenario for webservers is shown where the load largely increases if the number of users served exceeds a threshold. To model these effects we assume an exponential energy consumption function

$$P_{\mathrm{DC}}(N_{\mathrm{DC}}) = e^{\gamma_{\mathrm{DC}} \times N_{\mathrm{DC}}} - 1, \tag{A.6}$$

where γ_{DC} is the inverse energy efficiency factor.

In the global perspective, we additionally take into account user satisfaction for a particular service. For the sake of simplicity, we model the users to be only sensitive to the delay D imposed for data transfer[4] from the datacenter to the user without correlation to the number of other users served by the datacenter. The user utility function, after (A.3), is defined as

$$U_{\mathrm{user}}(D) = \left[\gamma_{u1} \cdot \log(-D + \gamma_{u2})\right]^{+}, \tag{A.7}$$

[3] We assume that all service requests impose the same load on a datacenter.

[4] For example, such a utility can be feasible for the mice traffic.

Table A.2: Main values for the load balancing scenario.

Parameter	Value
Number of users	13000
Number of data servers	3
Number of regions	3
Latency requirements differences	up to 20 times
Energy price difference	up to 10 times

with delay-tolerance coefficients of γ_{u1} and γ_{u2}. Clustering users coming from the same region, we define the optimization problem as an assignment problem

$$U_{\text{NUM}} = \max_{S}\Big(w_1 \cdot \sum_{i=1}^{N_{\text{DC}}} \sum_{j=1}^{N_{\text{user,DC}}} s_{i,j} U_{\text{user}}(D_{i,j}) + w_2 \cdot \sum_{i=1}^{N_{\text{DC}}} U_{\text{DC,i}}\left(\sum_{j=1}^{N_{\text{user,DC}}} s_{i,j} N_{\text{total}}\right)\Big), \tag{A.8}$$

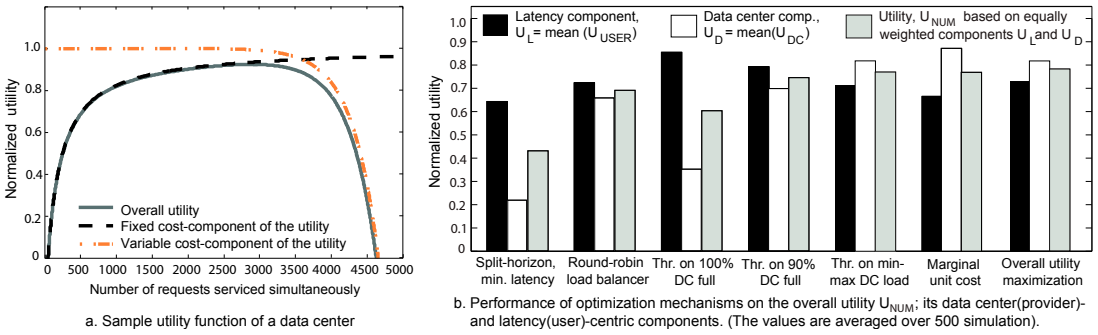
where w are the weight coefficients, S is the assignment matrix assigning a fraction of $s_{i,j}$ of the total system users to datacenter j, with i being the user cluster. Generally we assume the total number of users N_{total} to be unknown to the assignment mechanism.

Load-balancing strategies among datacenters often found in practice are based on adjusting the replies of DNS servers to the request of a particular service [435]. We evaluate several strategies that are often deployed in practice, and compare them to the three proposed utility-based schemes. Table A.2 depicts main parameter values we use.

In the *split-horizon* method, a reply to a service request is based on the estimated replay delay. However, due to the lack of feedback from a DC, the basic version of this algorithm may result in the request drops if a datacenter is overloaded. The standard *round-robin* strategy, where requests are equally balanced among the servers, also performs suboptimally and loses requests if the DC load is high. If the feedback from the datacenters is available to the DNS then more sophisticated assignment strategies are possible based on different abstractions of the utilities, and the data on a number of current requests served by a DC. Datacenters can send *thresholds* to the DNS specifying the maximal load levels. We show results obtained when we assign users requests to the datacenters that are the nearest, but only until a threshold is reached (100% and 90% from the maximum load). The strategy based on flexible *min-max thresholds* that are derived from the analysis of the form of the DC utility function performs very well. It is simple, computationally effective, and requires no utility function knowledge from the assignee.

We also studied performance of two algorithms where utility functionals are known to the DNS. In the provider-centric strategy a user is assigned to the datacenter which lowers the most its *marginal unit cost*. The more holistic approach operates on the *overall utility functional* for each incoming request. This strategy provides small performance gains as compared to other approaches, but requires an estimate of the total number of requests N_{total}, which imposes additional communication overhead. Generally these two methods, although more powerful and computationally demanding, do not provide considerably better performance as compared to the min-max threshold strategy, which we consider as the most preferable for our scenarios.

The results in Figure A.3 show that the utility-based approaches typically outperform other strategies. They can also adjust to the changing network objectives, flexibly switching between user- or provider-centric strategies and provide more balance solutions. Other strategies perform better only in the scenarios that they were designed for. For



a. Sample utility function of a data center

b. Performance of optimization mechanisms on the overall utility U_{NUM}; its data center(provider)- and latency(user)-centric components. (The values are averaged over 500 simulation).

Figure A.3: Performance of the optimization strategies for the power-aware load-balancing scenario [47].

example, the split-horizon strategy performs well on the user-oriented scenario, but fails everywhere else, and the data-center oriented optimizers cannot accommodate for the latency optimization.

A.4 Conclusions

In this appendix we considered the possible trade-off between granularity of stated goals (utility functions) and the complexity of required optimizers in presence of limited networking and computational resources and services. We placed our discussion in the context of the on-going debate on network neutrality vs. economy- and QoS-driven pricing of Internet. Our results support the popular claim that at least minimal differentiation of services can provide significant performance gains. Moreover, they showed that these gains could be further increased if the crude utility-based differentiation is applied in a dynamic manner.

B

Composing and Maintaining Services for White Appliances in a Home Environment using Ontology

In this appendix we show how an ontology and a related database can drive the composition of a functional set of services for devices in a given home environment, i.e. serve as a basis for *Software Configuration Management* (SCM) [47]. Our main purpose is to explain the main principles of ontological modeling and reasoning, as well as to argue on usability of this technology. We hope that using this example it will be more comfortable for the reader to assess the system for semi-autonomous WSN protocol stack design that is the main focus of Chapter 3.

The concept of an intelligent home, foreseeing co-existence of multiple devices, faces many challenges. Edwards and Grinter [436] have pointed among others the following issues: the partial, non-instantaneous, introduction of new technology at home, the interoperability issues, the high reliability demands, and the need for self-configuration. The solution of the above issues is impossible without a solid model describing the operational context of a home environment, its systems and their related interdependencies. Such a model, for example, ensures that services provided by devices installed at home answer user needs, the underlying software is secure, consistent, and does not cause conflicts or malfunctioning neither of other home services or the hardware itself.

We propose to use *ontology* as a core for such a model, as it enables fulfillment of the above tasks by setting a common vocabulary and relating concepts that are required to describe the relevant aspects a home system. This ontology is a core of a Knowledge Base (KB) that captures and reasons on a home environment in an autonomous manner, requiring human intervention only in the exceptional cases. For example, the system we developed automatically performs the following actions before installation a new service onto a device. It maps the service to the appropriate hardware-specific software, and estimates if any supporting modules and services are to be additionally installed. It further ensures integrity and conflict-free functioning of new and old software within the home network based on known interdependency issues, checks user rights to download the software, schedules download times, and assigns a priority to the download that would not to overburden the network. The KB is developed as part of the COMANCHE system [48], which aims to provide a software configuration management infrastructure that can be used in the home automation domain. To the best of our knowledge, the ontology-enabled KB system, when proposed [47], was the first to capture and reason upon services, related devices and software in a home environment, especially with respect to white home appliances. In the following we explain the structure of the proposed ontology, describe a sample use case, and provide the result of the reasoning performed on this scenario.

B.1 On Software Configuration Management

In order to enable Software Configuration Management (SCM) in the home environment, there needs to be a model that reflects devices installed at home, their provided services, and the corresponding software. Examples of devices constituting a home environment are computers, TV sets, white home appliances, various sensors and actuators, basically everything that can have a running software and be a part of a network. Additionally, the information on user profiles, accessible services, and the software components that implement them should be stored as well. The constructed model should reflect and ensure correct reasoning on interdependencies of all entities within a home environment, and resolve arising conflicts.

B.1.1 COMANCHE Framework

One of the possible approaches to the software configuration management is provision of a remote utility for massive servicing of home environment. This utility would keep track of the state of devices and networks installed at homes, provides remote maintenance and update, including secure access and installation of new software services. Subscription to such a utility can improve the overall level of user satisfaction from using "smart" appliances, and reduce the need to physical maintenance of devices installed at homes, thus, lowering the cost either for the housekeeping companies, or the users themselves.

The COMANCHE [48] framework is one example of such an approach. The system aims to effectively organize, discover, and exploit the tremendous amounts of attribute information needed for software configuration management in the home environment. The main development is a close collaboration and includes industry partners such as Alcatel-Lucent, Indesit and Gorenje. The framework follows service-oriented architecture, and, therefore, conceptualizes through the ontology the surrounding environment primary considering services that different objects (users, devices, software, other services) provide or utilize. COMANCHE is a centralized system, where knowledge concerning each individual home environment are stored centrally, which ensures the high level of data integrity and security. COMANCHE provides autonomous management of a home environment, and sends requests for platform managers for manual assistance in problem resolution only when it cannot resolve configuration conflicts. The latter functionality is enabled through reporting of unresolved conflicts by the reasoning engine of the KB. In order to balance the computational load, reduce the delay reaction time of the system, as well as to enable limited operation without Internet connectivity, some of the local activities of the system can be performed by a home gateway.

The COMANCHE system can be portrayed as a classical feedback loop with inputs being the status of a home environment, user commands, (software and service) updates provided by companies manufacturing or servicing the devices. These inputs can be obtained through both push and pull actions depending on technical convenience. The outputs of the system include the direct actions toward devices (e.g., installation of the new software update), enquiries send to service and device providers, and maintenance/conflict alerts to users, or local housekeeping/repair firms.

B.1.2 Evaluation Scenario

Before proceeding further with description of the ontology and related knowledge base, we describe a set of typical tasks to be carried out by the software configuration system.

In this appendix it serves as an *evaluation scenario* for our system. We consider the task of installation and, later, update of a new device that is a washing machine. First, once installed, the device registers itself with a home gateway. The gateway, that is the central management unit for the home, checks if it has the management software for this device, and if necessary installs it, maintaining the consistency of this software to the agents of other devices deployed at home. Once the home gateway updates itself with information concerning the new washing machine, for example, its hardware capabilities and default software installed, it propagates the changes on this home environment to the central management system. Later, when a user wants to change something in the installed devices, e.g., install a new power management function or advanced washing program onto the washing machine, the central KB will be queried to estimate interdependencies for the desired services, and establish a secure site from which the needed software components can be fetched. For instance, as part of this process, it needs to be ensured that the power management service is compatible with all other devices at the home network, so that a common power management can be enabled. The system also verifies that a new washing program can be supported by the particular washing machine, i.e., its hardware capabilities are sufficient, and it has been installed with all underlying services that the new program requires. Finally, the SCM software issues a command to the home gateway on scheduling of the installation and verification procedures for the respective required software modules.

B.2 Design of Home Ontology and its Knowledge Base

In this section we describe the design of the *home environment and context ontology*, which is one of the core components of COMANCHE framework (see Figure B.1). This ontology models a particular home environment, establishes interdependencies between potential or already installed services, related software modules, and hosting devices. Additionally its classes describe, for example, events encountered by devices or contain user profiles. But first, before proceeding with the detailed description we shortly discuss on inner organization of an ontology and the related work.

B.2.1 On Ontologies and Related Work

An ontology define the structure of a knowledge base that enables reasoning on specific objects related to the environment. It builds a model by defining and relating *classes* that conceptualize the world. It is also possible to relate *individuals* that populate a class. The relations between classes and individuals are established using object *properties*. Data properties are mostly used to encode quantifiable meta-data, or the data that is not represented are individuals of other classes. Both classes and properties can have hierarchies. Multiple complex relations can be established between these three basic entities. For example, properties may be marked as inverse of other properties, be transitive, etc. The inference is done based on these encoded relations, which basically results in classification of the entities, and leads to derivation of additional property relations. The inference can be done on the ontology as a whole, or a specific specific set of its classes or individuals. The latter is useful when, for example, a new individual, e.g., a washing machine, is added onto the existing ontology, and one wants to deduce to which classes it belongs to without spending resources on computing the whole ontology anew. If inference cannot be correctly performed, i.e., there a mistake in logic, the system reports

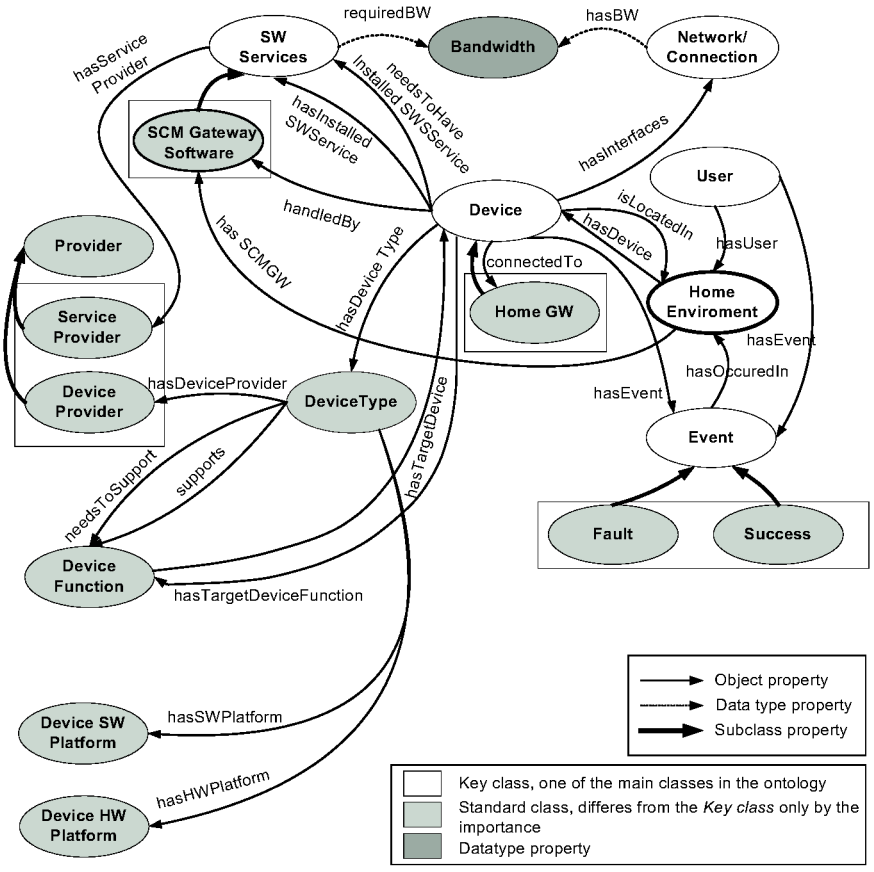


Figure B.1: Home Environment and Context Ontology [47].

on possible conflicts. The knowledge base consists of a populated ontology with the inference engine, which has not only classes and properties, but also contains individuals stored either in the database or as an XML (OWL) file.

Prior to this work [47] there existed several ontologies that describe smart home environments [437–440]. However, often these ontologies were not shown to serve as the basis for a knowledge base, i.e., did not operate on individuals, or they did not allow to model complex relations between services, the software that realizes them, users, hardware and the providers. Our ontology models these relations, and it even incorporates some elements of the actual home network modeling like the bandwidth requirement for different software services. Additionally, the suggested ontology is developed according to the requirements received from the leading home appliance producers such as Indesit and Gorenje. It is interesting to note that in spite of the complexity of the described environment our ontology is still manageable due to the heavy hierarchical structuring of both classes and properties. The proposal by Merabti et al. [441] has a prototype implementation and is closer to our ontology, but focuses on virtual services and home multimedia applications. The research by Bonito et al. [442], published in parallel with our work, is very promising and can complement our development. It concentrates on monitoring and controlling state of smart home using a designed ontology DogOnt.

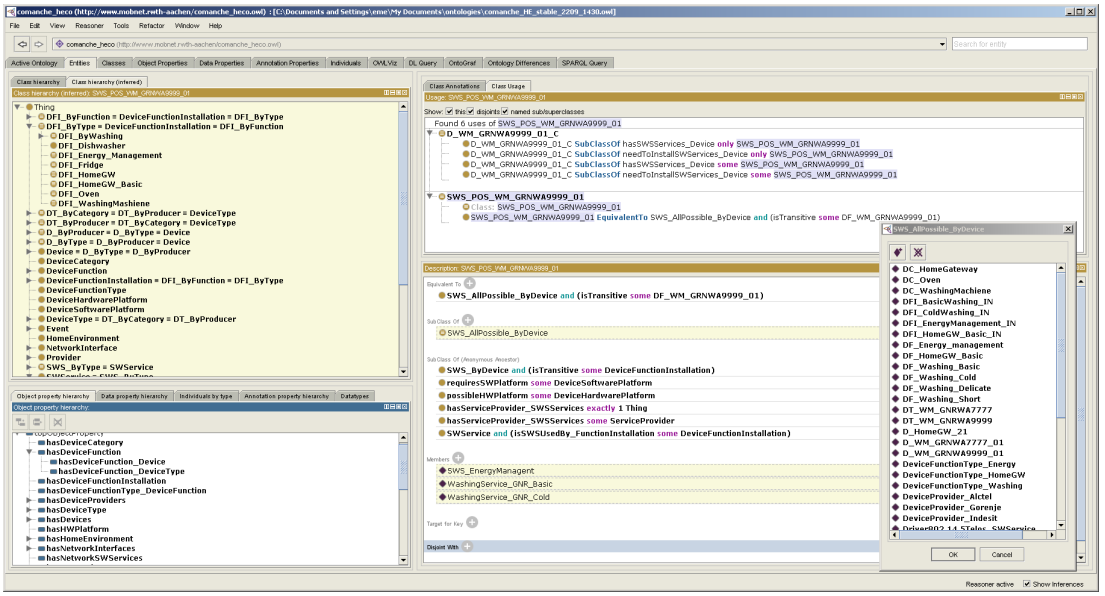


Figure B.2: Sample screenshot of the home ontology developed in Protege. Yellow boxes on the left highlight inferred classes and entities (individuals). The figure also displays inferred services that are installed onto a particular washing machine.

B.2.2 Design and Evaluation

As in Chapter 3 we realized the ontology in OWL [443]. The knowledge base is developed using the Protégé [444], and the reasoning is done using the FaCT++ [195]. Dynamic creation of ontology subclasses is done using Java and Jena. The overview of the ontology is given in Figure B.1 and its sample screenshot in Protégé is given in Figure B.2. There also exist multiple subclasses that are not shown in the basic ontology diagram. They are derived based on the rules imposed onto the ontology. For example, subclasses of the `device function` and `software service` classes record all the functions and the corresponding services that an appliance of a certain type might perform. The lowest level subclasses are created to link individuals, such as devices or services, to facilitate comfortable querying. For example, in our ontology there exist classes for a particular Gorenje washing machine that lists the supported and the installed services. The inference of this data requires the property relations to the class of describing the washing of the relevant type, from which the list of the supported services is extracted which then undergoes the constraint validations specific for this particular home environment. Overall, our ontology is realized following the OWL-S ontology [445] that allows for a flexible and rich set of object interdependencies. Further, in order to simplify the ontology maintenance we have used a hierarchical property structure with a number of parent, so-to-say abstract, properties. Next we shortly describe each of the main classes of the proposed ontology, as well as relations between them.

One of the main entities of the proposed ontology is the `home environment` class, which lists the home users and the installed devices, including the home gateway. The `device` class represents a specific appliance placed in the home environment, such as a washing machine. The profile of the device includes its model/type, the link to the event repositary, the list of functionalities and the respective software services, as well as the

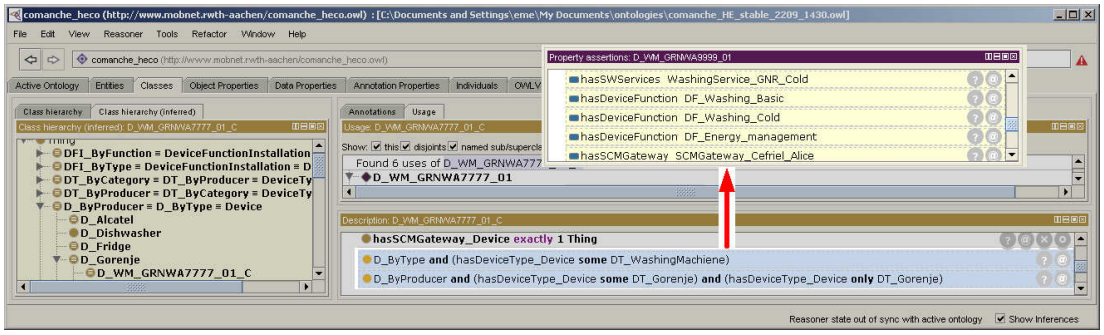


Figure B.3: Inferred list of functionalities that can be hosted by a washing machine D_WM_GRAWA9999_01.

link to the specific *SCM gateway software* that handles the appliance. The `device type` class describes the model of the device, such as Gorenje washing machine GRAWA9999. The class contains the list of functionalities a device can support, the producer's name, the description of the hardware and software platforms that run on the device, as well as the list of supported network interfaces. Those entities are created as instances of separate classes. Such a construct allows, for example, to automatically derive which services could potentially be hosted by a certain device type. They also allow deriving the list of functions that can performed by the device, which are later mapped to the list of installable services (see Figure B.3). `Home gateway` is one of the child classes of the device class. It handles all the incoming and outgoing network traffic, therefore it has to support extended security functionality, and handle heavy traffic loads. It also has to have special secure registration procedure with the SCM software gateway as compared to the ordinary home appliance, provided that the SCM gateway is hosted by the other device. It also has to register with the central SCM entity.

In this work, as compared to CONFab, we concentrate on the high level of software abstraction, thus dealing only with services and functionalities/functions. Services are getting installed onto the devices and they provide the desired functions. One functionality can be provided by the services of different vendors. Both functions and services have inner dependencies. For example, in order to install a delicate washing program one needs to install the basic, the cold water and the short programs. The ontology we created allows to encode such relations. The simple example of the inference performed on the KB is given in Figure B.4. In order for a device to perform any service, the software implementing this service is to be installed. In order to determine if a service can be installed on the certain type of device, besides mapping of the service to a certain device function, also the hardware and the software profiles of the device model have to be stored, which is enabled through `device software` and `hardware platform` classes. They capture such parameters as RAM, ROM, processing capabilities, peripherals, UI capabilities, etc. The `network interface` class is used to specify the network capabilities of a certain device type. This information is used to schedule servicing of different nodes/devices in a network. Additionally, through the software `provider` class the list of companies that host trusted software modules can be obtained. Device producers can be enquired to the list of trusted provider entities. The `event` class keeps tracks of events occurring in the home environment, proposed maintainable actions and their outcome. This information is used to schedule further actions and to resolve arising conflicts.

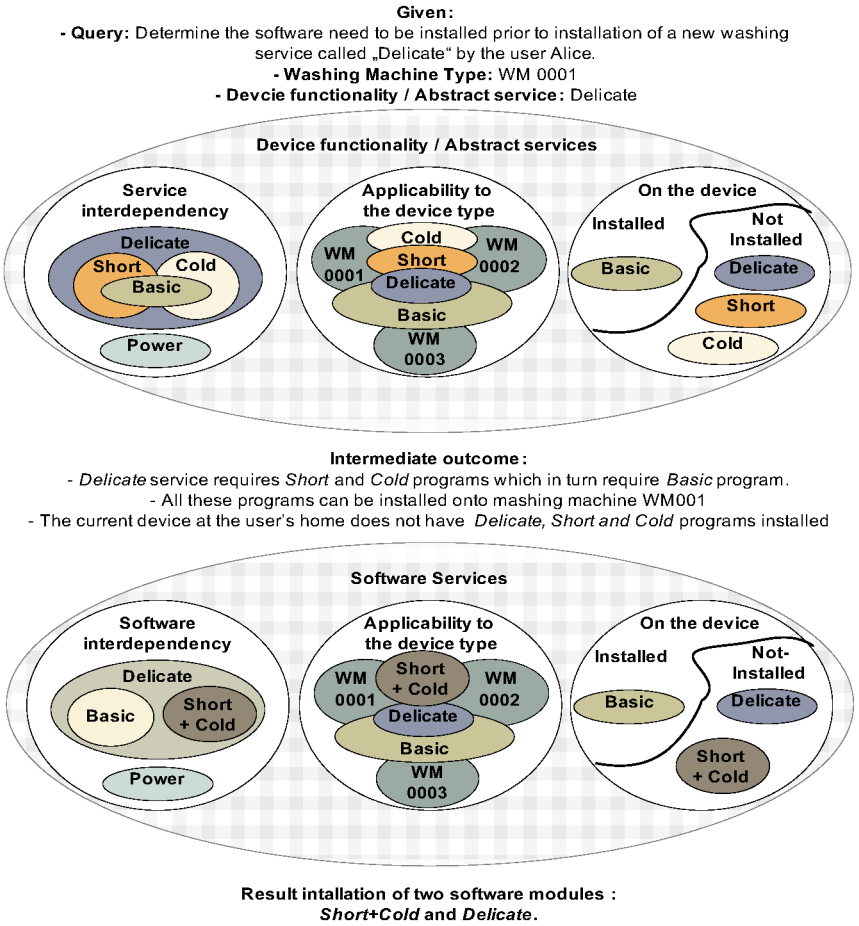


Figure B.4: Inference example: determination of the software modules need to be installed to enable the desired functionality [47].

Overall, as we see the task of conceptualizing the home environment from the viewpoint of software configuration management results in relatively complex home environment and context ontology that includes, besides the upper level classes described above, also numerous automatically generated subclasses and rules that dictate classification of individuals into these classes. Our experience and results indicate the suggested ontology and the related KB can effectively facilitate installation and management of both functionalities and the respective software services on the home devices ensuring secure and conflict free functioning of the home environment. The KB also responds to changes in the home environment by updating old or installing new services, resulting in the seamless configuration of the home network. Additionally, the system allows keeping track of the events happening in the home network in order, for example, to assist the providers in the remote maintenance of the home. The ontology can be also easily extended to incorporate new rules and new types of data. The described work indicates that ontology is one of the promising approaches for establishing of a common vocabulary and a basic for conceptualization for modeling, reasoning on, and optimization of a networked system.

C

The Outline of the Simulated Annealing Method

In this appendix we provide a pseudocode of the simulated annealing metaheuristic that we used for the experimentation in Chapter 4. Depending on the application of this method its functions *initTemp*, *decreaseTemp*, *neighbor*(), P and *timeout* are redefined throughout the chapter. Obviously, the inputs corresponding to the problems space search S spaces and the utility function U differ as well. The metaheuristic aims at maximizing the given utility function. It is presented on the next page.

Note that for the online exploration this search is exploited iteratively, with the pauses, corresponding to the exploitation phase of the algorithms. These, as well as the duration of the search itself are influenced by the currently monitored network utility values, as well as other possible indicators that might signal changes in the operational context. Therefore, if utility values are high simulated annealing is likely to be executed rarely, just to check if better parameter settings can be found, or to detect an improvement in the operational conditions, for which a new search might lead to sufficiently improved performance. However, if a drastic drop in the utility is detected then it is likely that the operational conditions have worsened, and the metaheuristic should be executed immediately to find a configuration that minimizes bad user experience.

Algorithm 1: Pseudocode of the simulated annealing search method.

Data: **S**, **U**, u_{ideal}, n_{sMax}, n_{r0}, n_{rb0}, t_0, t_{min}
Result: s_{final}, u_{final}
begin
  $s \longleftarrow \text{random}(S)$; $u \longleftarrow U(s)$; // Initialize the state as a random point in the state space, calculate the corresponding utility.
  $s_b \longleftarrow s$; $u_b \longleftarrow u$; // Initial "best" solution per re-heating period and its utility.
  $s_f \longleftarrow s$; $u_f \longleftarrow u$; // Initialize the overall, final, "best" solution and its utility value.
  $n_r \longleftarrow n_{r0}$; // Number of reheats to be executed.
  $n_s \longleftarrow 0$; // Initial number of state evaluations is 0. If the maximal utility observed is not improved for n_{sMax} evaluations the algorithm is halted.
  $i_{rb} \longleftarrow \text{TRUE}$; // If current reheating cycle takes the best available solution as the start.
  while ($n_r > 0$ *and* $f(u, u_{ideal}) == FALSE$ *and* $n_s < n_{sMax}$) **do**
    // Till the max. number of reheats expires, or desired utility would be reached that is a functions of the current u and ideal u_{ideal} utility values, or no improvement in the utility would be seen for n_{sMax} evaluations. If the ideal, desired, utility value is not known, the algorithm can still operate, but would converge slower.
    $t \longleftarrow \text{initTemp}(t_0, u_f, u_i, n_r, n_s)$; // Set the initial temperature. This could be just a constant t_0, or additionally be dependent on on the current utility values and the reheating count.
    if $i_{rb} == TRUE$ **then**
      // Is the solution found in this reheating period is better than the prior results?
      $s \longleftarrow s_f; u \longleftarrow u_f$
    else
      $s \longleftarrow \text{random}(S)$; $u \longleftarrow U(s)$;
    $i_{rb} \longleftarrow !\, i_{rb}$;
    while ($t > t_{min}$ *and* $f(u, u_{ideal}) == FALSE$ *and* $n_s < n_{sMax}$) **do**
      // While temperature is above threshold t_{min}, and utility is not good enough, and there are less then n_{sMax} evaluations without utility improvement.
      $s_n \longleftarrow neighbor(S, s, t, u, u_i)$; // Pick some neighbor. This is a model specific function. In the basic variant of SA the direction of the jump is random, and its length is directly proportional to the temperature, the current utility value (or dynamics in its improvement). We also consider utilizing the difference between the current and the ideal utilities if the latter is known.
      $u_n \longleftarrow \text{U}(s_n)$; // Evaluate its utility.
      if $u_n > u_b$ **then**
        // Is the new set better?
        $s_b \longleftarrow s_n; u_b \longleftarrow u_n$
      if $\text{P}(u, u_n, t) > \text{random}()$ **then**
        // Should we move to it?
        $s \longleftarrow s_n; u \longleftarrow u_n$ // Yes, change state.
      $t \longleftarrow \text{decreaseTemp}(t)$ // One more iteration done and temperature is to descrease.
    if $u_b > u_{final}$ **then**
      // Is the solution found in this reheating period is better than the prior results?
      $s_{final} \longleftarrow s_b$; $u_{final} \longleftarrow u_b$; $n_s \longleftarrow 0$
    else
      $n_s \longleftarrow n_s + 1$ // Increase the state evaluation count
    $n_r \longleftarrow n_r - 1$ // Decrease the counter of reheats.
  return s_{final}, u_{final}; // Return the best solution found.

D

Experimental Spectrum Sensor Testbed for Estimating Indoor Radio Environment

This appendix describes the testbed used to gather the empirical data on temporal aspects of spectrum use for the work presented in Chapter 5, Section 5.4. This testbed was constructed to empirically investigate and model the temporal and spacial characteristics of the radio environment for indoor scenarios that later could be used as inputs to Radio Environment Maps (REMs) [446]. The system is a network of over 80 heterogeneous wireless spectrum sensors with significantly different measurement capabilities that are deployed in an office building consisting of multiple rooms. Power spectrum activities of the transmitting nodes are monitored by the sensing devices that report RSSI (Received Signal Strength Indication) samples observed on a certain frequency (spectrum) range. Besides estimation of the temporal activity patterns we used the testbed to study indoor propagation conditions. The observed phenomena strongly indicate that the development of both spacial and temporal models of indoor environment, and the related optimization algorithms, which employ on empirical power spectrum data, is challenging, as both heterogeneity and specific characteristics of spectrum sensors and non-linearity of propagation conditions are to be taken into the account.

D.1 Testbed Design and Deployment

In the area of cognitive wireless networking the optimization problems that aim at *dynamic characterization and management of the indoor radio environment* [354, 447] using Radio Resource Management (RRM) solutions become increasingly important. In this context, Radio Environment Maps (REM) paradigm has been suggested as a support for RRM tools by providing a set of models that characterizes the radio environment as accurately as possible [447]. Construction of static optimization algorithm and models of the indoor environment are difficult due to, in part, complex dynamic characteristics of this media in terms of propagation and interference [448–450]. Dynamic environmental characteristics based on real time measurements are possible candidates to facilitate the efficient functioning of RRMs and REMs [451, 452]. Therefore, we decided to construct a testbed, where heterogenous sensing devices deployed over a large area, in order to facilitate the research on dynamic modeling on the indoor wireless conditions.

We carried out the testbed deployment in a typical office space to study the implications of device sensitivity, sampling rates, and the spatial dependencies on REMs in a representative indoor environment [58]. The testbed spans over the area of $240\,\mathrm{m}^2$ and occupies several rooms with both non load-bearing (paper) and bearing (semi-concrete) walls. It allows to realistically study the indoor propagation conditions as perceived by three types of sensor devices. These device types, namely USRP2 [406], WARP [307] and TelosB [162] nodes, display a wide range of sensing and processing capabilities, and to some extent reflect the characteristics that can be expected from a realistic hetero-

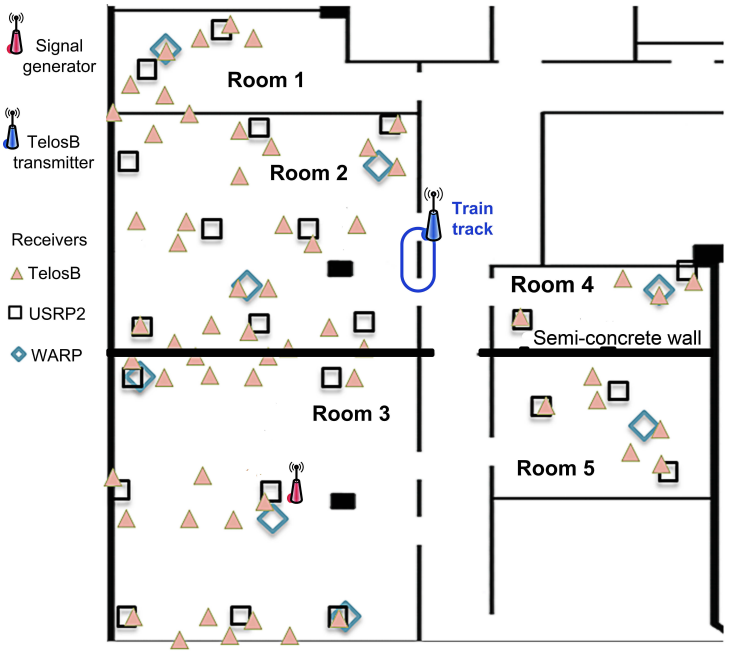


Figure D.1: The deployment setup of different devices in an area of 12 m × 20 m [58].

geneous network deployment. The testbed allows investigating the effect from both controlled and real user mobility as (a) some of the signal sources can be deployed on top of an electric toy train, and (b) a part of the testbed operates in a laboratory that is regularly used by a varying number of people. Therefore, the testbed fulfills the major requirements for enabling the experimental research of indoor radio environment, which are, the *diversity of the environment* (e.g. different structures of the walls, diverse furniture placements in the rooms) and the *space* covered by the testbed; *heterogeneity of spectrum sensors*; support for *different user activity and mobility* patterns; and operation over *multiple frequencies*, enough to accommodate the several independent channels [453].

In our deployment setup, we have used 60 TelosB nodes, 20 USRP2 boards and 8 WARP boards. Figure D.1a shows the testbed deployment map in the office building consisting of five rooms. We have also deployed embedded PCs interfaced to these devices. The embedded PCs are part of the office LAN infrastructure so that the measurements can be controlled centrally through a remote machine. Furthermore, being part of the backbone, all the machines are synchronized using the Network Time synchronization Protocol (NTP) [454]. Figures D.2 and D.3 show a typical deployment setup on an office table and an electric toy train that carries a sensing or transmitting device. We have developed the sensing applications [455, 456] with flexible controlling APIs for all types of spectrum sensors, and the corresponding software scripts to enable efficient remote testbed control that includes conducting multiple repetitive experiments and gathering of data in a systematic manner.

All the considered spectrum sensors operate on 2.4 GHz ISM band. Rice University's WARP SDR boards [307] and USRP2 boards from Ettus Research [406] have a sensing bandwidth of 22 MHz and 20 MHz respectively. Crossbow Inc. TelosB platforms [162] clocked at 4 MHz are transceivers with sensing bandwidth of 5 MHz. It should be noted

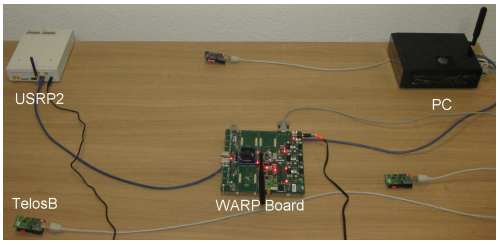


Figure D.2: Snapshot of a typical setup on an office table with a WARP board, a USRP2 board and three TelosB nodes attached to a PC, which is accessible remotely via LAN [58].

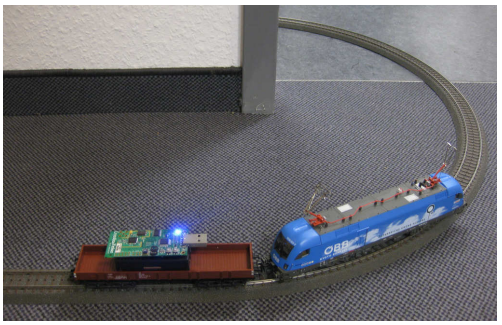


Figure D.3: Snapshot of a TelosB node used as a signal source with 0 dBm transmit power, placed on a toy train. This setup was used to study the effects of mobility in a controlled manner. The train track allows repeating the movement over and over again exactly at the same path. Furthermore, the train track is set so that it goes behind the wall to study the effects of walls on the indoor signal propagation characteristics.

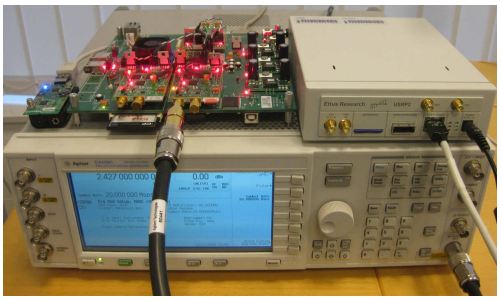


Figure D.4: The setup used to profile sensing devices with the signal fed directly over the coaxial cable. We considered different power levels and the whole range of operating frequencies. We used five devices of each type to study the variations in displayed performance between the devices.

that WARP and TelosB boards report the spectrum readings directly in terms of RSSI values, whereas USRP2 boards use two input channels for I and Q samples, which are then converted on the host PC into the power readings. Our choice of the sensor types clearly falls into three distinct classes with different sensitivity levels and hardware capabilities. The WARP boards are high end SDR platform providing extremely fast spectral data samples. The USRP2 boards are medium grade SDR platforms. The TelosB platforms rep-

Table D.1: Measured sensitivity ranges and execution timings for different sensors.

Spectrum sensors	Calibrated linear measurement range		Onboard execution timings (without PC communication)	
	Min	Max	Spectrum sampling	Channel switching
WARP	-90 dBm	-20 dBm	1.4 μs (RSSI)	22 μs
USRP2	-75 dBm	-25 dBm	50 ns (IQ)	200 μs
TelosB	-90 dBm	-10 dBm	70 μs (RSSI)	740 μs
Spectrum sensors	Timings for spectrum sensing with communication delay to the PC for different frequency bandwidths			
	Interface	5 MHz	20-25 MHz	2.402-2.482 GHz
WARP	Ethernet	–	30 μs (22 MHz)	132 μs
USRP2	Ethernet	2.02 ms	0.53 ms (20 MHz)	5.23 ms
TelosB	UART	3.5 ms	9.1 ms (25 MHz)	25.9 ms

resent low-end devices for spectrum sensing. Table D.1[1] summarizes the limits of sensing capabilities of all device types, which are dictated by their hardware capabilities and the software applications that were developed for this testbed.

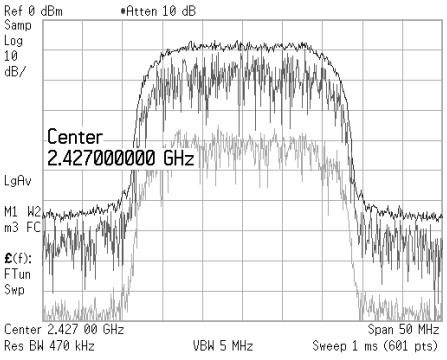
We have carefully profiled (see Figure D.4) and then calibrated to the linear operating range the considered types of spectrum sensors by feeding a referenced signal from an Agilent E4438C signal generator directly over a coaxial cable (see Table D.1). We have observed that the inconsistencies and biases among the devices of the same type are acceptably low. We also observed that USRP2 devices show strong non-linear behavior for low received powers, especially for levels below -75 dBm. Compared to the external monopole antennas for USRP2 and WARP boards, TelosB nodes have obviously slightly higher attenuation due to the inverted F microstrip antenna.

As signal sources we used a programmable Agilent E4438C signal generator and TelosB nodes. In this work for the continuous transmissions the E4438C generator was configured to produce QPSK modulated signal with 20 Msps and transmission power of 20 dBm (see Figure D.5a). TelosB nodes were used to generate multi-source and moving signals, as well as different ON-OFF patterns following non-uniform distributions. These signals were produced using the test mode of CC2420 radio transceiver chip of TelosB [362], which provides a continuous 5 MHz wide signal with most of the power concentrated in a bandwidth of 2 MHz (Figure D.5b). The transmit power was set to 0 dBm.

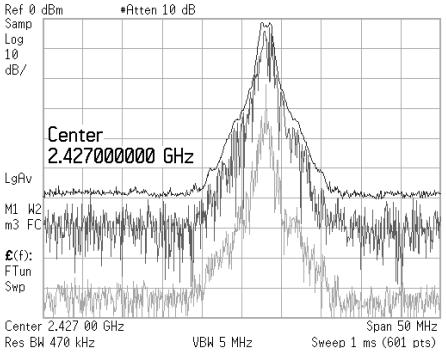
D.2 Examples of Obtained Results

In this section we provide examples of results that can be obtained using this testbed. We have considered two fundamental scenarios for this work. First we have studied the *spatial* aspects with a single transmitter sending a constant signal from the largest room (see Figure D.1). Then we have considered the implications of a moving transmitter.

[1] As most of the processing for USRP2 boards are happening to the host PC, its configuration influences the performance of the spectrum sensor. In our tests we used Core i7 based PC with Ubuntu x64 10.04 OS. The timing values for USRP2 are averaged over 10000 samples.



(a) QPSK Modulated signal at 20Msps



(b) Modulated signal output of a TelosB node

Figure D.5: Spectrum analyzer screenshots of the signals used during the measurements. The figure shows the maximum hold, instantaneous signal and the minimum hold for a duration of 5 minutes.

Finally, we generated different ON-OFF signal patterns illustrating the *temporal* spectral aspects as observed by heterogeneous spectrum sensors, which are primarily reported in Chapter 5, Section 5.4[2].

D.2.1 Influence of Propagation Environment

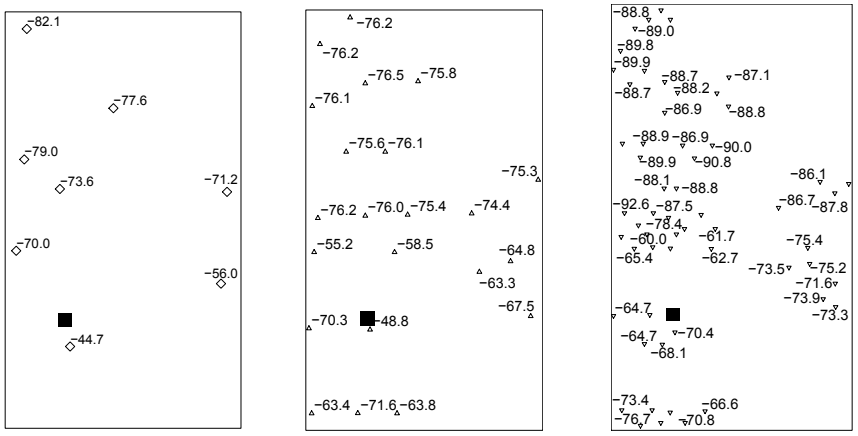
Indoor propagation characteristics play one of the central roles in a number of application scenarios for REMs [457]. On the example of the constant signal transmitted by the E4438C signal generator we illustrate the overall distribution of received power readings as observed by different devices in the experimental indoor environment. The received power levels indicate a high variance in the measurement values obtained by closely located sensors of the same, as well as of the different, types (see Figures D.6–D.7). This is due to the effect of multipath propagation, as one can observe non-linear and even non-monotonically decreasing function of the distance (see Figure D.8). As an illustration we fit the data to the linear equation

$$P_i(\text{dBm}) = K - \alpha \log_{10}(d_i) - \beta C_i, \tag{D.1}$$

where $P_i(\text{dBm})$ is the power received by node i, d_i is the distance between the sender and the node i, and C_i is any further covariate value for node i to be used in the model. We have experimented with linear fitting (a) using only information on the distances d_i to the transmitter, and (b) adding C_i as the number of walls between the sender and the node i. All walls are assumed to be homogenous with β corresponding to the drop in the received power per wall. We obtained the estimates $\hat{K}$, $\hat{\alpha}$ and $\hat{\beta}$ for the propagation model parameters by simple least squares regression. For each fit we also studied the distribution of the residual estimation errors

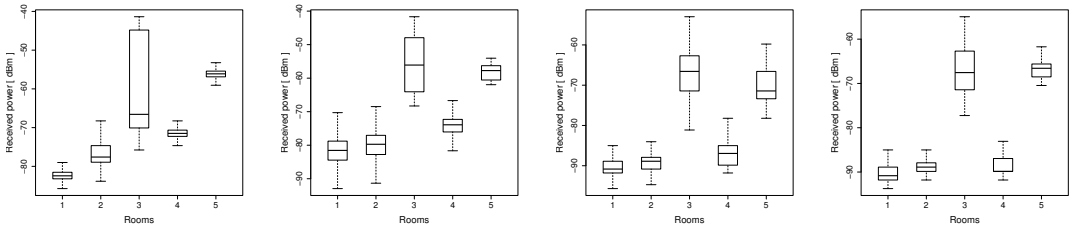
$$P_i(\text{dBm}) - \hat{P}_i(\text{dBm}) = P_i(\text{dB}) - \hat{K} + \hat{\alpha} \log_{10}(d_i) + \hat{\beta} C_i \tag{D.2}$$

[2]We also studied the marginal densities of the results considering each type of sensors and the overall distribution of the received noise levels. We observed close-to-normal distributions of the received power for the noise, as sensed by different types of devices.



(a) WARP boards (b) USRP2 boards (c) TelosB nodes

Figure D.6: Mean received power at different devices in dBm. The filled square represents the position of the transmitter.



(a) WARP (b) USRP2 (c) Aggregate on all channels - TelosB. (d) Only 5 MHz central channel - TelosB

Figure D.7: Distribution of the received power levels [dBm] in rooms for various devices [58].

as well as their dependency on the distances d_i and the chosen covariates. The obtained results are displayed in Figure D.9. They show that the linear fitting with information only on the locations of the receivers resulted in average estimation errors of 5.68 dB, 5.71 dB and 7.19 dB for WARPs, USRP2s and TelosB nodes, respectively. The additional data on the number of walls between the transmitter and the receivers reduced the respective errors to 3.78 dB, 5.3 dB and 6.8 dB. These results indicate that, as expected, the classical propagation models do not fit well the indoor environment and more complex, maybe even non-linear models, have to be dynamically constructed or tuned to be applicable to the particular indoor conditions [450, 452].

D.2.2 Transmitter Mobility

The experiments with the mobile transmitter were carried out using a TelosB node as a signal source, which was placed on top of a model train moving with a constant speed of 5 m/s located as shown in Figure D.1. This resulted in repetitive transmitter mobility patterns with short term variations caused in part by fast fading (see Figure D.10). While fast fading clearly severely affects the results, it is still possible to detect the underlying periodicity, at least given a short correlation time for the channel.

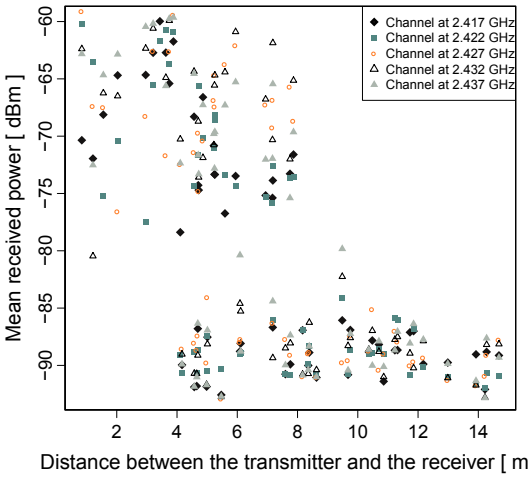


Figure D.8: Illustration of multi-path effects on adjacent channels for TelosB nodes.

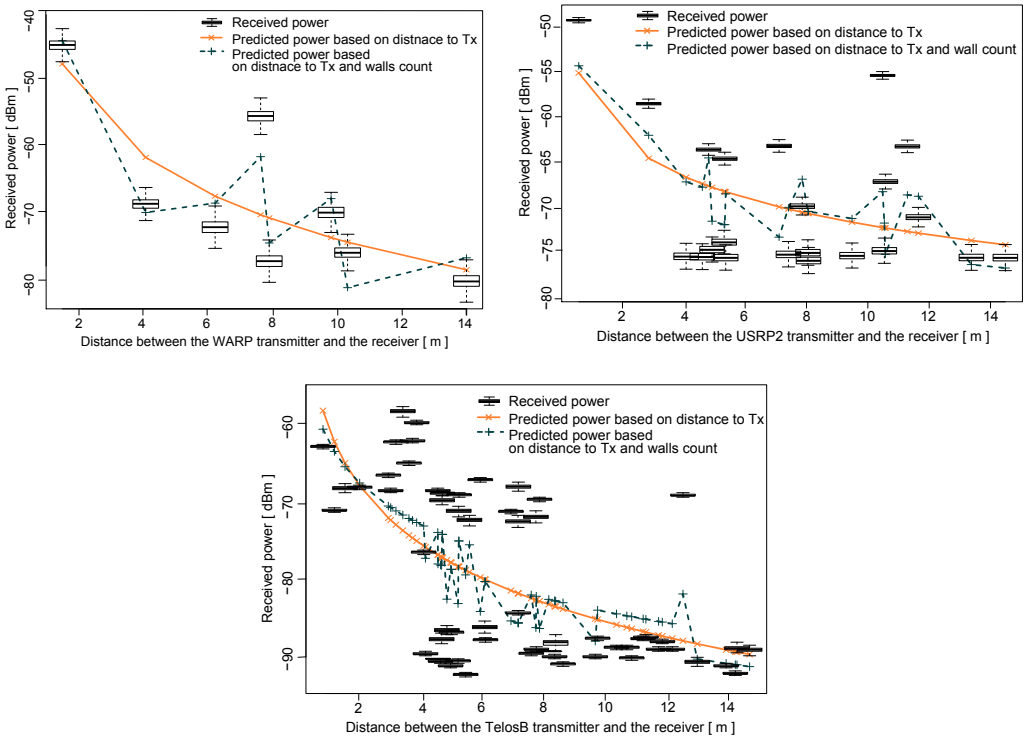
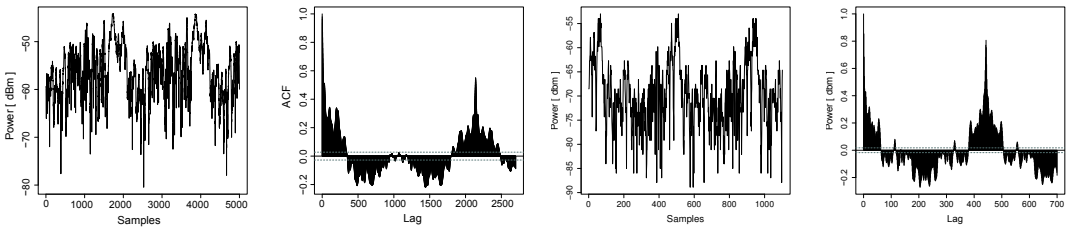


Figure D.9: Boxplots of the received power levels for the spectrum sensors at different distances to the transmitter, and results of linear fitting using the information on distances to the Tx and the walls. The figures show the data for WARP, USRP2 and TelosB devices [58].



(a) The USRP2 board (b) The USRP2 board (c) The TelosB node (d) The TelosB node

Figure D.10: Observed signals by the devices located near the moving transmitter. The figures (a,c) show the time series spectral data, whereas the figures (b,d) potray the autocorrelation function (ACF) of the spectral data [58].

D.2.3 ON-OFF Transmitter Patterns

As discussed in detail in Chapter 5, Section 5.4, we also have used the testbed to monitor the temporal structure of the power levels as observed by spectrum sensors. The basic characterizations of a duty cycled transmitter are the durations of the active (ON) and inactive (OFF) periods. The accurate estimation of the duty cycles, as well as ON-OFF activity periods can significantly enhance the channel sensing and selection processes for DSA networks [345, 346]. The constructed testbed allows detecting signals with fairly high temporal resolution (see Table D.1) that is appropriate to record both packet-level and session-level behaviors of the networks, for example IEEE 802.11g wireless deployments. We have also programmed the TelosB nodes to generate energy levels with timings accurately following various ON-OFF distributions, with the minimal duty cycle duration of 100 ms. These signals are useful to systematically study and evaluate in practice the applicability of various optimization algorithms that utilize temporal spectrum power data.

D.3 Conclusions

In this appendix we have presented the experimental testbed for studying the implications indoor environments pose to cognitive radio technologies relying on power spectrum sensing. Using the testbed that consists of a 80+ heterogenous sensing nodes we gathered a number data traces related to both spatial and temporal aspects indoor power spectrum. Based on this dataset, we showed how the same signals can be differently measured by different types of sensors. We observed the effects of fast-fading and mobility that can drastically affect the received power and lead to misinterpretation of the readings. In the spatial domain we recorded major variations in the data gathered by the nearby nodes. All of this strongly indicates that more measurement-driven theoretical and modeling work is required to understand and efficiently exploit the *network-wide* behavior of the indoor radio environment.

E

HCRM: Sample Codes and Selected Measurements

In this appendix we provide additional materials on the HCRM design, a framework for autonomous radio resource management (self-optimization) for wireless home networks described in Chapter 6. Specifically, we first show the flowchart of the decision tree algorithm to perform RRM based on fast changes in observed utility values and KPIs. We also show the sample pseudocode executed as part of this algorithm. In the second part of the appendix we discuss selected aspects of the flexible parameter state space realization in the meta-optimizer, including the exempts from the corresponding class hierarchy. We provide results from sample measurements that we conducted to test the HCRM and study the indoor operational environment in the third part of the appendix. These results were used to derive the thresholds for the decision tree algorithm, and to decide on the scenarios for testing the multi-arm bandit-based machine learning optimization.

E.1 Flowchart and Samples of Preudocode for the Decision Tree Algorithm

The flowchart of the decision tree algorithm, which is one of the reflex agents for the HCRM (see Chapter 6) is shown in Figure E.1. Sample pseudocode illustrating implementation of parts of this algorithm is given in Figures 2 and 3.

E.2 Details on Flexible Parameter State Space Realization in Meta-optimizer

In this section we discuss the key classes that enable realization of the flexible run-time adjustable parameter state space in the meta-optimizer. These are shown in Figure E.2. The `DETopology` class stores the current knowledge on the network, which includes the current configuration of links, flows, and adapters of both the current and the other relevant nodes. (The knowledge of the other nodes is needed in the case of link configuration and cooperative network optimization.) The knowledge on the allowable state of certain parameters is obtained through the `PossibleConfigurations` class. The actual settings of the parameters are stored in the classes, with names ending with `*ActuatorState`. The permitted, possible, states are stored in the classes with `*PSets` postfix. The configuration state in an `*ActuatorState` class is a pointer to a variable of the `*PSets`, ensuring that only allowed values from the specific `*PSets` class can be given to a parameter. Figure E.2 also shows the main parameters that affect MAC parameter settings, as defined in the classes `CSMAMacActuatorsState` and `CSMAMacPerInterfacePSet`.

The allowable values for each parameter are inherited from the `VirtualValueSet` class, which is basically a Boolean list that specifies if the particular state is allowed or not. The inherited value classes `DoubleValues`, `TrafficClassesPSets`, and `RadioModes` contain a list of double values used, for example, to store permitted central frequencies, traffic classes, and allowed radio modes (e.g. 802.11g or 802.11a).

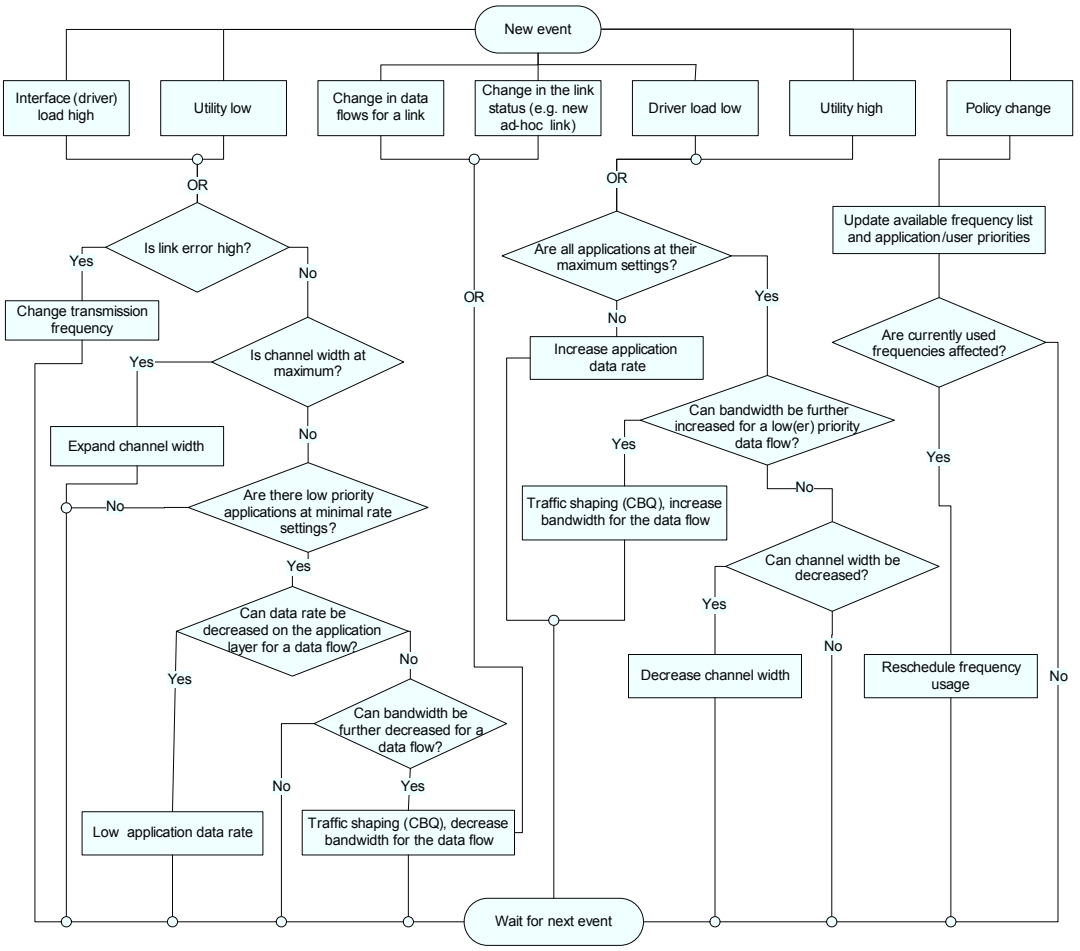


Figure E.1: The overview of the decision tree algorithm.

The data hierarchy is organized as follows. Each adapter might host several links. `ChannelConfiguration` describes a link configuration including radio parameters and MAC parameters. Though not shown on the diagram, the class `RadioConfiguration` enables the current and the permitted channel settings to be stored for each radio mode. Each link also maintains a list of flows routed through it. The queue, transport and applications settings are maintained on per flow basis. There exist wrappers to adjust the settings common for all flows on a certain link or node.

E.3 Measurements with the HCRM for the Indoor Environment

While developing the HCRM we have conducted a number of experiments to both study properties of the indoor wireless environment, and to adjust reactions of the system to it to ensure the desired performance. In particular, our aim was to devise appropriate thresholds and dependencies that lead to robust decision tree optimization algorithms (i.e., reflex agents or standard RRM mechanisms). For the evaluation of the HCRM we use two utility functions. One is the cumulative network throughput as measured on the MAC layer. Another is the *spectrum-aware* utility that is the function of goodput and the

Algorithm 2: Pseudocode showing sample procedures within the decision tree algorithm. Part I: deciding on centralized link scheduling. For simplicity this example is given only for two links always scheduled to operate at maximum bandwidth, though our implementation is done to arbitrary number of links and different channel widths. (The scheduling is done by the node with the highest IP.)

```
if (numLinks == 2) and (isInterfaceLoadHigh) and (isPERHigh)
  (nodeIP == maxIP_network) then
    links = links_in_network;
    moveToPrimary(links);
    spectrum = getSpectrumData(); // scheduling the first link
    min_power_CFreq1 = getChannelRanking(spectrum)[0];
    configureLink(links[0], min_power_CFreq1, max_width);
    moveToSecondary(links[0]);
    // scheduling the second link;
    if there is a free non-interfered channel available move the link
    there, otherwise co-locate it with the first channel.
    spectrum = getSpectrumData();
    min_power_CFreq2 = getChannelRanking(spectrum)[0];
    if spectrum[0] < -90dBm then
        configureLink(links[1], min_power_CFreq2, max_width);
    else
        reconfigureLink(links[1], min_power_CFreq1, max_width);
```

bandwidth network nodes occupy, which is defined as

$$U_N(T,B) \propto \frac{\sum_{i=1}^{F_N} G_i}{\bigcup_{j=0}^{L_N} B_j} \approx \frac{\sum_{i=1}^{L_N} T_i}{\bigcup_{j=0}^{L_N} B_j}, \tag{E.1}$$

where utility of the network U_N is proportional to the sum of the goodputs G_i achieved across all data flows F_N, and inversely propositional to the bandwidth B_i occupied by all the links N_L. The first term of the equation can be further approximated by throughput T_i achieved by all the network links N_L. Throughput T is measured in Mbps, and channel bandwidth B is measured in MHz. In our case studies, we use a network where there are one or two active ad-hoc links. Therefore, the equation (E.1) becomes

$$U_{L1}(T,B) = \begin{cases} \frac{(T_1+T_2)\cdot\log_{10}20}{\log_{10}(B)\cdot\alpha\cdot 20} & \text{if } (T_1+T_2) < \alpha\cdot 20 \\ 1 & \text{otherwise} \end{cases}, \text{ where}$$
$$B = \left| [f_2,f_1] \cup [f_4,f_3] \right|, \tag{E.2}$$

where $f_1,\ldots,f_4$ are frequencies in MHz defining the bandwidth of the first and the second ad-hoc links. Other variables are as in (E.1). If only one ad-hoc link is active then $\alpha = 1$, $[f_4,f_3] = 0$, and $T_2 = 0$, otherwise, for two links, $\alpha = 2$. The utilized bandwidth is logarithmically normalized over the base of 20, which is the maximal channel bandwidth.

We have experimented with the HCRM in various indoor wireless scenarios (see Figure E.3). In particular, we were interested in how the central channel frequency and the transmission bandwidth influence the two utility functions depending on distances between the transmitting pairs of nodes, and the type of interference. (In the experiments described below we always executed the UDP-based iperf traffic [302]. We did not

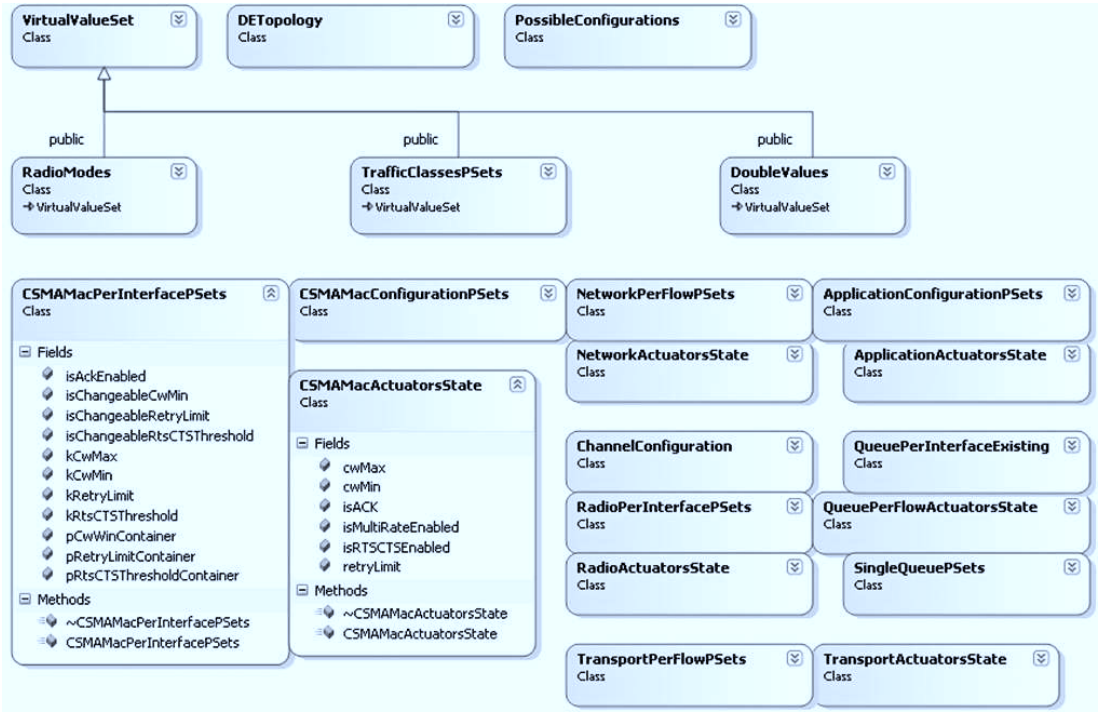
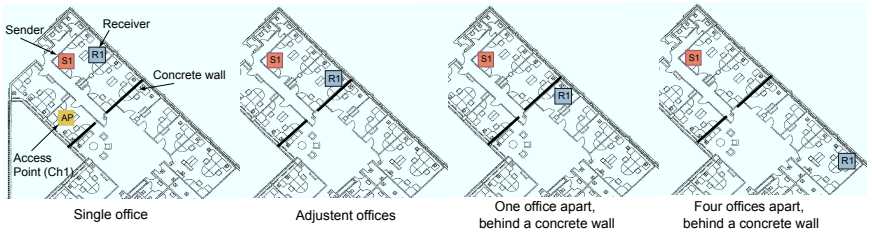
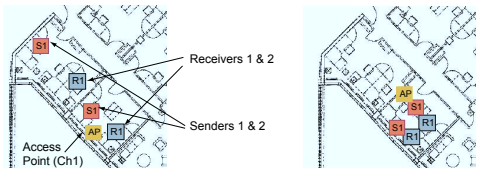


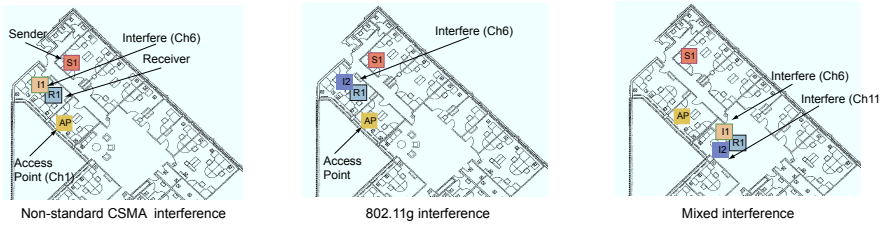
Figure E.2: Sample classes of the meta-optimizer that allow management of the parameter (actuation) state space.



a. The scenario, where the distance between the transmitter and the receiver is changed.



b. The four node scenario.



c. The scenarios with interference.

Figure E.3: Layout of different measurements conducted using the HCRM in this section.

Algorithm 3: Pseudocode showing sample procedures within the decision tree algorithm. Part II: deciding on bandwidth between data flows for CBQ network layer mechanisms. The algorithm is invoked at the stage when no further channel width expansion for a given link is possible.

```
weightOfGOLD = 0.5; weightOfSILVER = 0.3; weightOfBRONZE = 0.2;
safetyLimit = 0.2;
if (onFlowChanges) or (onLinkStateChanges) or (isPERLow and
isChannelWidthMAX and isInterfaceLoadHigh) then
    for flow in currentLink->flows do
        switch flow→trafficClass do
            case GOLD
                numFlowsGoldWeighted + = weightOfGOLD;
            case SILVER
                numFlowsSilverWeighted + = weightOfSILVER;
            case BRONZE
                numFlowsBronzeWeighted + = weightOfBRONZE;
    // percent of bandwidth to be left for fast wireless fluctuations
    maxBW = currentLink → throughput · (1 − safetyLimit);
    totalNumOfFlowsWeighted = numDataGoldWeighted +
            numDataSilverWeighted + numDataBronzeWeighted;
    for flow in currentLink→Flows do
        // higher traffic classes are not bandwidth limited
        switch flow→trafficClass do
            case SILVER
                if count(currentLink→flows→trafficClass == GOLD) > 0 then
                    flow→BWlimit = numFlowsSilverWeighted · maxBW /
                                   totalNumOfFlowsWeighted;
            case BRONZE
                if count(currentLink→flows→trafficClass != BRONZE) > 0
                then
                    flow→BWlimit = numFlowsBronzeWeighted · maxBW /
                                   totalNumOfFlowsWeighted;
```

adjust such MAC settings as the sensitivity threshold, and the retry limit. We kept the datarate at the maximum level of 54 Mbps. Though these parameters may have a crucial effect on the performance as shown in Chapter 5, we did not vary them concentrating on studying the effects of adjustment of channel properties.)

As already discussed in Chapter 6, Chandra et al. [412] have shown that a certain adjustments of the WLAN hardware allows changing the Wi-Fi channel widths, and, as expected, narrower channels, e.g. 5 MHz, lead to higher received powers, thus enabling higher throughput at greater distances between nodes than when the standard 20 MHz bandwidth is used. However, these gains hold only for very specific sets of propagation conditions, distances and interference patterns. In this appendix we provide evidence of such behavior. Figure E.4 illustrates the scenario, where the throughput and the

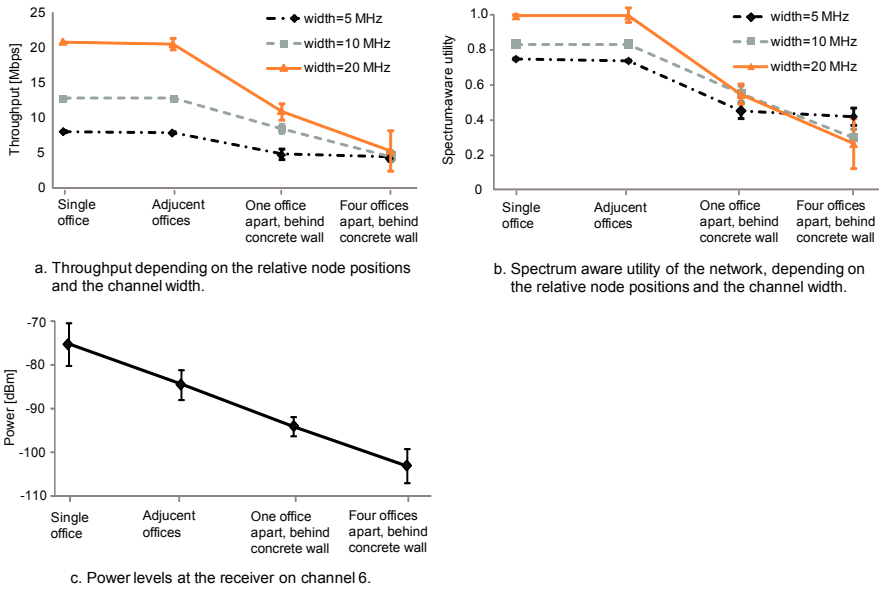


Figure E.4: Dependency of the utilities of the pair of the HCRM nodes on their relative positioning (distance). The nodes transmit on Channel 6. No external interference is generated on the ISM 2.4 GHz band.

spectrum-aware utility is measured for a single network link for the two nodes situated at varying distances from each other. As we observe the channel width adjustment pays off only when nodes are situated four offices apart, and operates of the spectrum-aware utility. In other tested scenarios, including experimentations with longer distances and additional low levels of interference that are not shown in the figure, the narrow transmission bandwidth does not provide considerable gains even for the spectrum-aware utility. For short and medium distances the standard 20 MHz bandwidth provides much higher throughput and overweighs gains from the efficient bandwidth usage[1]. For long distances, basically the "gray" transmission zone, it is difficult to establish ad-hoc links due to loss of beacons and those established provide low throughput. Here the narrow frequency band indeed pays off, but often the alternative solution of using an AP as a relay provides better performance. Our experimentation with the multi-arm bandit algorithm discussed in Chapter 6 confirms this conclusion. Below we show that the combination of the information on the received RSSI levels and the nature of the encountered interference is sufficient to distinguish between these operational contexts. But first we further discuss on the operational contexts, this time considering the cases for node power leakage, and different types of interference.

On an example of a four-node network with two active transmissions, we show how nature of interference influences the two actuation parameters that are the transmission channel and its bandwidth. We also study the feasibility of co-locating the transmissions on the same channel, which could be realized through cooperative decision-making, to avoid co-channel interference. First, in Figure E.5 we show the results obtained on two topologies. The first displays standard positioning, where the nodes are stationed in two

[1]This, of course, only applies for the specific utility functions were are considering. It is possible to design other utility functions that would heavily favor narrower band transmission, but we do not think they are currently realistic for Wi-Fi related wireless home scenarios.

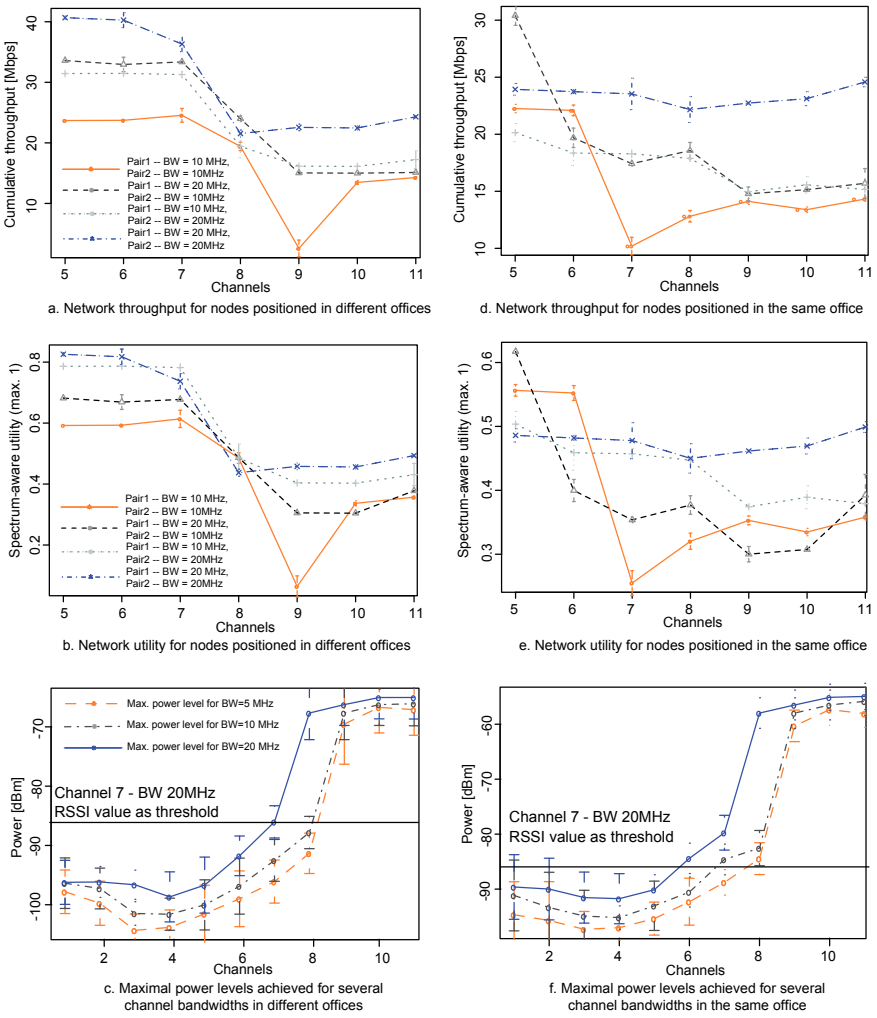


Figure E.5: Performance of a four-node network in the scenarios, where node are positioned either in the nearby offices or very close to each other (the power leakage scenario). All node use IEEE 802.11g standard for transmission. The first pair of nodes transmits on Channels from 6 to 11, while the second pair of nodes always uses Channel 11.

different offices. In the second topology nodes are just two meters apart. The results observed for the second topology (Figures E.5d-f) considerably differ from the expected performance patterns shown by the first scenario (Figures E.5a-c). The difference is induced by the high power leakage to the nearby channels of the WiFi COTS hardware [458]. The second topology, therefore, requires different channel configuration decisions. For the first network setup, in the case of throughput maximization with the maximum bandwidth used, the two links should be at least four or five channels apart. Otherwise, it is better for the links to share the same channel. In the power leakage scenario, this distance should be higher than six channels, which is impossible in our setup, as one of the independent channels (Channel 1) is used as the control link and, therefore, Channels 1 to 3 are forbidden for the ad-hoc transmission. Therefore, for the power leakage scenario it is better to co-locate the links on the same channel, and assign them the maximum transmission bandwidth. In the case of the spectrum aware utility an alternative configuration is pos-

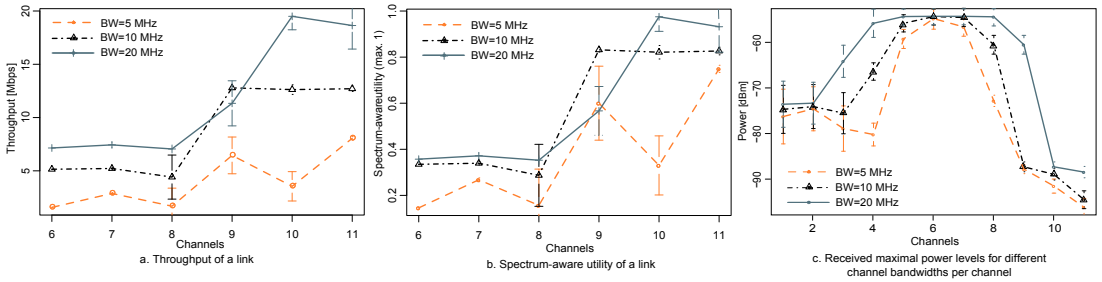


Figure E.6: Performance of a two-node network in the scenario of external interference on Channel 6. The respective sender and receiver are located in the adjacent offices. The interference is in the same office as the receiver.

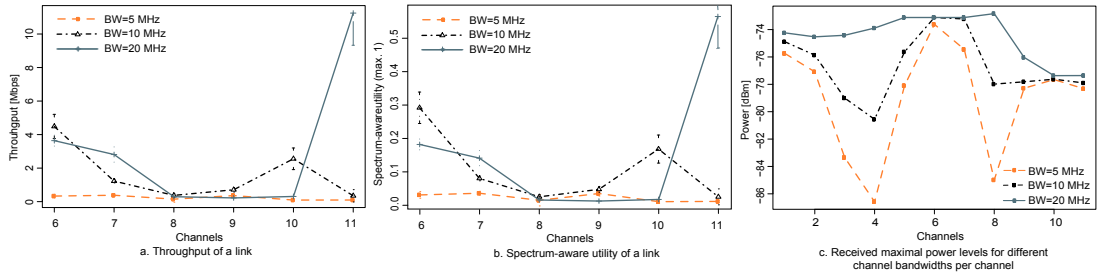


Figure E.7: Performance of a two-node network in the scenario of mixed interference. There is a non-standard CSMA interferer on Channel 6, and on Channel 11 there is a IEEE 802.11g transmission. The respective sender and receiver are located in one office apart across a semi-concrete wall.

sible, where one of the transmitting pairs is assigned a narrow 10 MHz channel that is five channels away from the other transmitting pair, which uses the maximum bandwidth.

The leakage can be identified based on the power spectrograms, which are obtained when only the second pair of nodes is transmitting. In the case of the power leakage it is much wider than in the normal case (see Figures E.5c,f). The decision thresholds, guiding the viable channel allocation, i.e., determining minimum number of channels that should separate the transmissions, can use the RSSI-based metric. However, these are context specific, i.e., they differ, e.g., in the case of the power leakage and the normal operation scenarios. They are also influenced by the MAC datarate settings that are related to the sensitivity thresholds of the protocol, the particular COTS hardware design [459], the nature of the interference, and even the utilized transmission bandwidths. For example, in our measurements for the channel width of 20 MHz good performance was obtainable if a new transmission for placed on a channel with the detected power below -85 dBm. However, for the channel width of 5 MHz this value had be lower than -92 dBm. Dependence of these thresholds on the transmission bandwidth might indicate that the realization of the channel width adaptation has a flaw. For example, it might be that the “sensing mask” of the Wi-Fi is not properly adjusted according to the channel width used.

The optimal behavior for interference avoidance depends on the source of the interference. In the contrast to the previous set of measurements, in the cases of non-standard (though CSMA-based) interference source, it is suboptimal to co-locate the 802.11g WLAN transmissions (from the HCRM) and other equally strong interferes [374]. As shown in Figure E.6, if the link is operating on the same channel as the non-standard

CSMA external interferer, in our case the WifiCopper Packet Injector [414], performance degradation is very high. Though the power spectrum mask of this interferer is different from the standard IEEE 802.11g the RSSI-based decision thresholds determined above still apply (see Figure E.6c). In the case of mixed interference, shown in Figure E.7, we observe, as expected, that it is the best to co-locate the Wi-Fi transmissions (on Channel 11). However, if that is not possible, e.g., due to the policy constraints, then it is better to transmit on Channel 6 with a bandwidth of 10 MHz. The latter option is explainable by the fact that the co-channel interference from two sources harms the transmission more than sharing the same band with the non-standard CSMA interferer. In the case of the last scenario the received RSSI samples and the respective power values are not enough to make the optimization decision and, again, the knowledge on the nature of interference and/or its spectral masks are needed.

Our experiments that were conducted for relatively well-studied small Wi-Fi setups already indicate the crucial need for context awareness for the effective and robust self-optimization. For more complex networks this task becomes much more prominent, and we consider it as one of the major enablers of the Mitola's wireless cognitive networking vision. We strongly believe that probabilistic reasoning and machine-learning methods are required to perform context identification and exploitation in the semi-autonomous manner. This statement is also supported by the results of the application of these techniques obtained in Chapters 4–6. We consider context identification as part of the work on optimization task decomposition to be one of the promising directions for future research.

Bibliography

[1] "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2010-2015", Cisco, http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.pdf [last Visisted 24.09.2011].

[2] "Active BGP entries (FIB)", UCLA, http://bgp.potaroo.net/as2.0/bgp-active.html [last Visisted 24.09.2011].

[3] [Online] Wi-Fi Continues to Dominate One Billion Unit Home Networking Equipment and Networked-Enabled Media Device Market, http://www.abiresearch.com/press/wi-fi-continues-to-dominate-one-billion-unit-home- [Last visited: 25.09.2012].

[4] C. Ebert and C. Jones, "Embedded software: Facts, figures, and future," *IEEE Computer*, vol. 42, no. 4, pp. 42–52, 2009.

[5] A. Saleh, J. Simmons, and M. Architects, "Technology and architecture to enable the explosive growth of the internet," *IEEE Communications Magazine*, vol. 49, no. 1, pp. 126–132, 2011.

[6] J. Crowcroft, "Internet failures: an emergent sea of complex systems and critical design errors?" *The Computer Journal*, vol. 53, no. 10, p. 1752, 2010.

[7] T. Griffin and G. Wilfong, "On the correctness of IBGP configuration," *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 4, pp. 17–29, 2002.

[8] J. Mogul, "Emergent (mis) behavior vs. complex software systems," *ACM SIGOPS Operating Systems Review*, vol. 40, no. 4, pp. 293–304, 2006.

[9] S. Ratnasamy, S. Shenker, and S. McCanne, "Towards an evolvable internet architecture," in *Proc. of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*. ACM, 2005, pp. 313–324.

[10] A. Ghodsi, T. Koponen, B. Raghavan, S. Shenker, A. Singla, and J. Wilcox, "Intelligent design enables architectural evolution," in *Proc. of HotNets 2011*, 2011.

[11] H. Benington, "Production of large computer programs," *Annals of the History of Computing*, vol. 5, no. 4, pp. 350–361, 1983.

[12] B. Blanchard, W. Fabrycky, and W. Fabrycky, *Systems engineering and analysis*. Prentice Hall Upper Saddle River, 1990.

[13] A. Sangiovanni-Vincentelli, "Is a unified methodology for system-level design possible?" *IEEE Design & Test of Computers*, vol. 25, no. 4, pp. 346–357, 2008.

[14] J. Doyle, D. Alderson, L. Li, S. Low, M. Roughan, S. Shalunov, R. Tanaka, and W. Willinger, "The robust yet fragile nature of the internet," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 102, no. 41, p. 14497, 2005.

[15] M. Chiang, S. Low, A. Calderbank, and J. Doyle, "Layering as optimization decomposition: A mathematical theory of network architectures," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 255–312, 2007.

[16] J. He, J. Rexford, and M. Chiang, "Design for optimizability: Traffic management of a future internet," *Algorithms for Next Generation Networks*, pp. 3–18, 2010.

[17] G. Kiczales, J. Lamping, A. Mendhekar, C. Maeda, C. Lopes, J. Loingtier, and J. Irwin, "Aspect-oriented programming," *ECOOP, Object-Oriented Programming*, pp. 220–242, 1997.

[18] M. Harman, "The current state and future of search based software engineering," in *Proc. of Future of Software Engineering*. IEEE Computer Society, 2007, pp. 342–357.

[19] S. Edwards, L. Lavagno, E. Lee, and A. Sangiovanni-Vincentelli, "Design of embedded systems: Formal models, validation, and synthesis," *Proceedings of the IEEE*, vol. 85, no. 3, pp. 366–390, 1997.

[20] Q. Zhu, Y. Yang, M. Natale, E. Scholte, and A. Sangiovanni-Vincentelli, "Optimizing the software architecture for extensibility in hard real-time distributed systems," *IEEE Transactions on Industrial Informatics*, vol. 6, no. 4, pp. 621–636, 2010.

[21] S. Russell, P. Norvig, J. Candy, J. Malik, and D. Edwards, *Artificial intelligence: a modern approach*. Prentice hall, 2010.

[22] F. Kelly, A. Maulloo, and D. Tan, "Rate control for communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research society*, pp. 237–252, 1998.

[23] Ofcom, "Table of base station totals," http://stakeholders.ofcom.org.uk/sitefinder/table-of-totals.

[24] M. Michalopoulou and J. Riihijärvi and A. Achtzehn and P. Mähönen, "Studying the relationships between spatial structures of wireless networks and population densities," in *Proc. of IEEE GlobeCom 2010*, Miami, Florida, December 2010.

[25] A. Mishra, *Fundamentals of Cellular Network Planning and Optimisation: 2G/2.5 G/3G... Evolution to 4G*. John Wiley & Sons, 2004.

[26] "The E-model, a computational model for use in transmission planninod," March 2005.

[27] W. Hale, "Frequency assignment: Theory and applications," *Proceedings of the IEEE*, vol. 68, no. 12, pp. 1497–1514, 1980.

[28] J. Riihijärvi, M. Petrova, and P. Mähönen, "Frequency allocation for wlans using graph colouring techniques," in *Proc. of IEEE WONS 2005*. IEEE, 2005, pp. 216–222.

[29] H. Holma, A. Toskala *et al.*, *Wcdma for Umts*. Wiley Online Library, 2000, vol. 4.

[30] K. Zhou and J. Doyle, *Essentials of robust control*. Prentice Hall Upper Saddle River, NJ, 1998, vol. 104.

[31] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," *ACM Transactions on Programming Languages and Systems*, vol. 4, no. 3, pp. 382–401, 1982.

[32] R. Thomas, D. Friend, L. DaSilva, and A. MacKenzie, "Cognitive networks: adaptation and learning to achieve end-to-end performance objectives," *IEEE Communications Magazine*, vol. 44, no. 12, pp. 51—57, December 2006.

[33] M. Petrova, P. Mähönen, and J. Riihijärvi, "Evolution of radio resource management: A case for cognitive resource manager with VPI," in *Proc. of IEEE ICC 2007*. IEEE, 2007, pp. 6471–6475.

[34] R. Komali, R. Thomas, L. DaSilva, and A. MacKenzie, "The price of ignorance: distributed topology control in cognitive networks," *IEEE Transactions on Wireless Communications*, vol. 9, no. 4, pp. 1434–1445, 2010.

[35] A. Krause and C. Guestrin, "Near-optimal nonmyopic value of information in graphical models," in *Proc. of Uncertainty in AI*, 2005.

[36] R. Howard, "Information value theory," *IEEE Transactions on Systems Science and Cybernetics*, vol. 2, no. 1, pp. 22–26, 1966.

[37] B. Allen, "Information as an economic commodity," *The American Economic Review*, vol. 80, no. 2, pp. 268–273, 1990.

[38] B. Loasby, *Choice, complexity, and ignorance: an enquiry into economic theory and the practice of decision-making*. Cambridge University Press, 1976.

[39] J. Doyle, B. Francis, and A. Tannenbaum, *Feedback control theory*. Macmillan Publishing Company New York, 1992, vol. 134.

[40] V. Srivastava and M. Motani, "Cross-layer design: a survey and the road ahead," *IEEE Communications Magazine*, vol. 43, no. 12, pp. 112–119, 2005.

[41] C. Blum and A. Roli, "Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison," *ACM Computing Surveys*, vol. 35, no. 3, pp. 268–308, 2003.

[42] E. Meshkova, J. Riihijärvi, A. Achtzehn, and P. Mähönen, "On utility-based network management," in *In Proc. of IEEE GLOBECOM Workshops*, December 2010, pp. 600–605.

[43] E. Meshkova, K. Rerkrai, J. Riihijärvi, and P. Mähönen, "Optimizing component-oriented systems: A case study in wireless sensor networks," in *Demonstration paper at ACM SIGCOMM 2008*, August 2008.

[44] J. Ansari, E. Meshkova, W. Masood, A. Muslim, J. Riihijärvi, and P. Mähönen, "CONFab: Ontology and component based optimization of WSN protocol stacks with deployment feedback," *Computer Networks*, vol. 74, pp. 89–108, 2014.

[45] ——, "Confab: component based optimization of wsn protocol stacks using deployment feedback," in *Proceedings of the 10th ACM international symposium on Mobility management and wireless access*. ACM, 2012, pp. 19–28.

[46] E. Meshkova, J. Riihijärvi, A. Achtzehn, and P. Mähönen, "Exploring simulated annealing and graphical models for optimization in cognitive wireless networks," in *Proc. of IEEE GLOBECOM 2009*. IEEE, November 2009, pp. 1–8.

[47] E. Meshkova, J. Riihijärvi, P. Mähönen, and C. Kavadias, "Modeling the home environment using ontology with applications in software configuration management," in *Proc. of ICT 2008*. IEEE, June 2008, pp. 1–6.

[48] S. Grilli, E. Makri, K. Steblovnik, S. Efremidis, N. Mouratidis, P. Mähönen, and E. Meshkova, "Service configuration management for ambient intelligence: the comanche approach," in *Proc. of IEEE AINAW*. IEEE Computer Society, 2007.

[49] E. Meshkova, J. Riihijärvi, J. Ansari, K. Rerkrai, and P. Mähönen, "An extendible metadata specification for component-oriented networks with applications to WSN configuration and optimization," in *Proc. of IEEE PIMRC 2008*. IEEE, September 2008, pp. 1–6.

[50] E. Meshkova, J. Riihijärvi, F. Oldewurtel, C. Jardak, and P. Mähönen, "Service-oriented design methodology for wireless sensor networks: A view through case studies," in *Proc. of IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing*. IEEE, November 2008, pp. 146–153.

[51] A. Achtzehn, E. Meshkova, J. Ansari, and P. Mähönen, "Motemaster: A scalable sensor network testbed for rapid protocol performance evaluation," in *Proc. of IEEE SECON 2009—Demonstrations Track*. IEEE, 2009, pp. 1–3.

[52] E. Meshkova, Z. Wang, J. Nasreddine, D. Denkovski, C. Zhao, K. Rerkrai, T. Farnham, A. Ahmad, A. Gefflaut, L. Gavrilovska *et al.*, "Using cognitive radio principles for wireless resource management in home networking," in *Proc. of IEEE CCNC 2011*. IEEE, January 2011, pp. 669–673.

[53] E. Meshkova, A. Achtzehn, J. Riihijärvi, and P. Mähönen, "Cross-layer optimization of edge networks," in *Demonstration paper at SIGCOMM*, August 2009.

[54] ——, "Towards user-centric network optimization engine," in *Proc. of IEEE SECON 2009, Posters Track*. IEEE, 2009, pp. 1–3.

[55] E. Meshkova, J. Riihijärvi, and P. Mähönen, "Learning in cross-layer wireless network optimization," in *Proc. of the Learning for Networking Workshop, in conjunction with SIGMETRICS/Performance*, 2009.

[56] ——, "Extending graph-based models of wireless network structure with dynamics," in *Proc. of ACM MSWiM 2012*. ACM, 2012, pp. 33–38.

[57] J. Riihijärvi, E. Meshkova, and P. Mähönen, "Graph approximations of spatial wireless network models," in *Proc. of the 6th ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks*. ACM, 2011, pp. 93–100.

[58] E. Meshkova, J. Ansari, D. Denkovski, J. Riihijärvi, J. Nasreddine, M. Pavloski, L. Gavrilovska, and P. Mähönen, "Experimental spectrum sensor testbed for constructing indoor radio environmental maps," in *Proc. of IEEE DySPAN 2010*. IEEE, May 2010, pp. 603–607.

[59] E. Meshkova, J. Ansari, J. Riihijärvi, J. Nasreddine, and P. Mähönen, "Estimating Transmitter Activity Patterns: an Empirical Study in the Indoor Environment," in *Proc. of IEEE PIMRC*, September 2011.

[60] P. Mähönen, M. Petrova, J. Riihijärvi, and M. Wellens, "Cognitive wireless networks: Your network just became a teenager," in *Proc. of IEEE INFOCOM 2006*, 2006.

[61] K. Rerkrai, J. Nasreddine, Z. Wang, J. Riihijärvi, and P. Mähönen, "Design and implementation of utility-based radio resource optimization using capri," in *Wireless Communications and Mobile Computing Conference (IWCMC), 2011 7th International*. IEEE, 2011, pp. 2262–2267.

[62] V. Atanasovski, D. Denkovski, T. Farnham, L. Gavrilovska, A. Gefflaut, V. Kolar, P. Mähönen, E. Meshkova, B. Motz, J. Nasreddine *et al.*, "Cognitive radio for home networking," in *Proc. of IEEE DySPAN 2010*. IEEE, May 2010, pp. 1–2.

[63] Z. Wang, J. Ansari, V. Atanasovski, D. Denkovski, T. Farnham, L. Gavrilovska, A. Gefflaut, R. Manfrin, E. Meshkova, J. Nasreddine *et al.*, "Self-organizing home networking based on cognitive radio technologies," in *Proc. of IEEE DySPAN 2011*. IEEE, May 2011, pp. 666–667.

[64] S. O'Malley and L. Peterson, "A dynamic network architecture," *ACM Transactions on Computer Systems (TOCS)*, vol. 10, no. 2, pp. 110–143, 1992.

[65] D. Batory and S. O'malley, "The design and implementation of hierarchical software systems with reusable components," *ACM Transactions on Software Engineering and Methodology*, vol. 1, no. 4, pp. 355–398, 1992.

[66] M. Lehman, "Programs, life cycles, and laws of software evolution," *Proceedings of the IEEE*, vol. 68, no. 9, pp. 1060 – 1076, September 1980.

[67] A. Sangiovanni-Vincentelli and G. Martin, "Platform-based design and software design methodology for embedded systems," *IEEE Design & Test of Computers*, vol. 18, no. 6, pp. 23–33, 2001.

[68] D. Palomar and M. Chiang, "A tutorial on decomposition methods for network utility maximization," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1439–1451, 2006.

[69] R. Braden, T. Faber, and M. Handley, "From protocol stack to protocol heap: role-based architecture," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 1, pp. 17–22, 2003.

[70] J. Day, *Patterns in Network Architecture: A Return to Fundamentals*. Prentice Hall, 2008.

[71] L. Simic, P. Mähönen, M. Petrova, and J. De Vries, "Illuminating the road from engineering and policy to radio regulation," *Proc. of TPRC 2012*, 2012.

[72] T. G. Griffin and G. Wilfong, "An analysis of BGP convergence properties," *ACM Computer Communications Review*, vol. 29, no. 4, pp. 277–288, Aug. 1999.

[73] W. Lemstra, V. Hayes, and J. Groenewegen, *The innovation journey of Wi-Fi: The road to global success*. Cambridge University Press, 2010.

[74] J. Pelkmans, "The GSM standard: explaining a success story," *Journal of European Public Policy*, vol. 8, no. 3, pp. 432–453, 2001.

[75] M. Jarke, P. Loucopoulos, K. Lyytinen, J. Mylopoulos, and W. Robinson, "The brave new world of design requirements," *Information Systems*, vol. 36, no. 7, pp. 992–1008, 2011.

[76] E. Chikofsky, J. Cross *et al.*, "Reverse engineering and design recovery: A taxonomy," *IEEE Software*, vol. 7, no. 1, pp. 13–17, 1990.

[77] R. Pressman and D. Ince, *Software engineering: a practitioner's approach*. McGraw-hill New York, 1992, vol. 5.

[78] C. Larman and V. Basili, "Iterative and incremental developments. a brief history," *IEEE Computer*, vol. 36, no. 6, pp. 47–56, 2003.

[79] S. Kent, "Model driven engineering," in *Proc. of the 3rd International Conference on Integrated Formal Methods*. Springer, 2002, pp. 286–298.

[80] P. Liggesmeyer and M. Trapp, "Trends in embedded software engineering," *IEEE Software*, vol. 26, no. 3, pp. 19–25, 2009.

[81] F. Balarin, Y. Watanabe, H. Hsieh, L. Lavagno, C. Passerone, and A. Sangiovanni-Vincentelli, "Metropolis: An integrated electronic system design environment," *IEEE Computer*, vol. 36, no. 4, pp. 45–52, 2003.

[82] S. Balsamo, A. Di Marco, P. Inverardi, and M. Simeoni, "Model-based performance prediction in software development: A survey," *IEEE Transactions on Software Engineering*, vol. 30, no. 5, pp. 295–310, 2004.

[83] D. Schmidt, "Guest editor's introduction: Model-driven engineering," *IEEE Computer*, vol. 39, no. 2, pp. 25–31, 2006.

[84] A. Van Lamsweerde, "Goal-oriented requirements engineering: A guided tour," pp. 249–262, 2001.

[85] N. Jennings, "An agent-based approach for building complex software systems," *Communications of the ACM*, vol. 44, no. 4, pp. 35–41, 2001.

[86] B. Cox, *Object oriented programming*. Addison-Wesley, Reading, MA, 1985.

[87] R. France, A. Evans, K. Lano, and B. Rumpe, "The UML as a formal modeling notation," *Computer Standards & Interfaces*, vol. 19, no. 7, pp. 325–334, 1998.

[88] W. Willinger and J. Doyle, "Robustness and the internet: Design and evolution," *Robust design: a repertoire of biological, ecological, and engineering case studies*, 2003.

[89] D. Alderson and J. Doyle, "Contrasting views of complexity and their implications for network-centric infrastructures," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 40, no. 4, pp. 839–852, 2010.

[90] T. Lan, X. Lin, M. Chiang, and R. Lee, "Stability and benefits of suboptimal utility maximization," *IEEE/ACM Transactions on Networking*, vol. 19, no. 4, pp. 1194–1207, 2011.

[91] D. Alderson, J. Doyle, R. Govindan, and W. Willinger, "Toward an optimization-driven framework for designing and generating realistic internet topologies," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 1, pp. 41–46, 2003.

[92] L. Chen, S. Low, and J. Doyle, "Cross-layer design in multihop wireless networks," *Computer Networks*, vol. 55, no. 2, pp. 480–496, 2011.

[93] R. Oliver, *Satisfaction: A behavioral perspective on the consumer*. ME Sharpe Inc, 2009.

[94] A. Parasuraman, V. Zeithaml, and L. Berry, "A conceptual model of service quality and its implications for future research," *Journal of Marketing*, vol. 49, pp. 41–50, 1985.

[95] A. Van Moorsel, "Metrics for the internet age: Quality of experience and quality of business," in *Proc. of the Fifth International Workshop on Performability Modeling of Computer and Communication Systems, Arbeitsberichte des Instituts für Informatik, Universität Erlangen-Nürnberg, Germany*, vol. 34, no. 13, 2001, pp. 26–31.

[96] K. Kilkki, "Quality of experience in communications ecosystem," *Journal of universal computer science*, vol. 14, no. 5, pp. 615–624, 2008.

[97] M. Fiedler, T. Hossfeld, and P. Tran-Gia, "A generic quantitative relationship between quality of experience and quality of service," *IEEE Network Magazine*, vol. 24, no. 2, pp. 36–41, 2010.

[98] A. Odlyzko, "The delusions of net neutrality," in *Proc. of Telecommunications Policy Research Conference*, 2008.

[99] D. Hubbard, *How to measure anything: finding the value of intangibles in business*. Wiley, 2010.

[100] C. Shapiro and H. Varian, *Information rules: a strategic guide to the network economy*. Harvard Business Press, 1999.

[101] J. Bézivin, "In search of a basic principle for model driven engineering," *Novatica Journal, Special Issue*, vol. 5, no. 2, pp. 21–24, 2004.

[102] J. Favre, "Towards a basic theory to model model driven engineering," in *Proc. of the 3rd Workshop in Software Model Engineering*, 2004.

[103] C. Bishop *et al.*, *Pattern recognition and machine learning*. Springer New York, 2006, vol. 4, no. 4.

[104] Y. Ben-Haim, *Info-gap decision theory: decisions under severe uncertainty*. Academic Press, 2006.

[105] Traceroute — Linux man page, http://linux.die.net/man/8/traceroute [Last visited 11.09.2011].

[106] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law relationships of the internet topology," in *ACM SIGCOMM Computer Communication Review*, vol. 29, no. 4. ACM, 1999, pp. 251–262.

[107] W. Willinger, R. Govindan, S. Jamin, V. Paxson, and S. Shenker, "Scaling phenomena in the internet: Critically examining criticality," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 99, no. Suppl 1, p. 2573, 2002.

[108] L. Li, D. Alderson, W. Willinger, and J. Doyle, "A first-principles approach to understanding the internet's router-level topology," in *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4. ACM, 2004, pp. 3–14.

[109] X. Lin, N. Shroff, and R. Srikant, "A tutorial on cross-layer optimization in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1452–1463, 2006.

[110] K. Varadhan, R. Govindan, and D. Estrin, "Persistent route oscillations in inter-domain routing* 1," *Computer networks*, vol. 32, no. 1, pp. 1–16, 2000.

[111] R. Mahajan, D. Wetherall, and T. Anderson, "Understanding BGP misconfiguration," *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 4, pp. 3–16, 2002.

[112] M. Caesar and J. Rexford, "BGP routing policies in ISP networks," *IEEE Network Magazine*, vol. 19, no. 6, pp. 5–11, 2005.

[113] L. Vanbever, B. Quoitin, and O. Bonaventure, "A hierarchical model for BGP routing policies," in *Proceedings of the 2nd ACM SIGCOMM workshop on Programmable routers for extensible services of tomorrow*. ACM, 2009, pp. 61–66.

[114] M. Feridun, M. Leib, M. Nodine, and J. Ong, "ANM: Automated network management system," *IEEE Network Magazine*, vol. 2, no. 2, pp. 13–19, 1988.

[115] G. R., *Mathematical Physics*. The University of Chicago Press, 1985.

[116] M. M. Fokkinga, "A gentle introduction to category theory-the calculational approach," 1992.

[117] H. Herrlich, G. Strecker, and B. Banaschewski, "Abstract and concrete categories. the joy of cats," Tech. Rep., 2004.

[118] R. Rosen, "The representation of biological systems from the standpoint of the theory of categories," *Bulletin of Mathematical Biology*, vol. 20, no. 4, pp. 317–341, 1958.

[119] A. Ehresmann and J. Vanbremeersch, *Memory evolutive systems: hierarchy, emergence, cognition*. Elsevier Science Limited, 2007, vol. 4.

[120] S. Phillips and W. Wilson, "Categorial compositionality: A category theory explanation for the systematicity of human cognition," *PLoS computational biology*, vol. 6, no. 7, p. e1000858, 2010.

[121] J. Baez and M. Stay, "Physics, topology, logic and computation: a rosetta stone," *New structures for physics*, pp. 95–172, 2011.

[122] B. Pierce, *Basic category theory for computer scientists*. MIT press, 1991.

[123] Z. Diskin and T. Maibaum, "Category theory and model-driven engineering: From formal semantics to design patterns and beyond," *arXiv preprint arXiv:1209.1433*, 2012.

[124] K. Williamson, M. Healy, and R. Barker, "Industrial applications of software synthesis via category theory-case studies using specware," *Automated Software Engineering*, vol. 8, no. 1, pp. 7–30, 2001.

[125] D. Kozen, C. Kreitz, and E. Richter, "Automating proofs in category theory," *Automated Reasoning*, pp. 392–407, 2006.

[126] L. dos Santos Leal, D. Claudio, L. Toscani, and P. Menezes, "Approximation problems categories," *Computer Aided Systems Theory–EUROCAST 2005*, pp. 9–14, 2005.

[127] T. Batista, C. Chavez, A. Garcia, U. Kulesza, C. SantAnna, and C. Lucena, "Aspectual connectors: supporting the seamless integration of aspects and adls," *Simpósio Brasileiro de Engenharia de Software-SBES*, pp. 17–32, 2006.

[128] O. de Moor, *Categories, relations and dynamic programming*. Cambridge University Press, 1992.

[129] D. Spivak, "Higher-dimensional models of networks," *Arxiv preprint arXiv:0909.4314*, 2009.

[130] A. Zimmermann, M. Krotzsch, J. Euzenat, and P. Hitzler, "Formalizing ontology alignment and its operations with category theory," *Frontiers in Artificial Intelligence and Applications*, vol. 150, p. 277, 2006.

[131] J. Goguen, "A categorical manifesto," *Mathematical Structures in Computer Science*, vol. 1, no. 01, pp. 49–67, 1991.

[132] M. Kokar, J. Tomasik, and J. Weyman, "Data vs. decision fusion in the category theory framework," *Proc. of FUSION 2001*, 2001.

[133] S. Thorsen and M. Oxley, "What category theory tells us about information fusion," in *Proc. of the 9th International Conference on Information Fusion*. IEEE, 2006, pp. 1–7.

[134] M. Healy and T. Caudell, "Temporal sequencing via supertemplates," Tech. Rep., 2010.

[135] M. Healy, "Category theory as a mathematics for formalizing ontologies," *Theory and Applications of Ontology: Computer Applications*, pp. 487–510, 2010.

[136] S. Abramsky, S. Gay, and R. Nagarajan, "Interaction categories and the foundations of typed concurrent programming," pp. 35–113, 1996.

[137] J. Rutten, "Universal coalgebra: a theory of systems," *Theoretical Computer Science*, vol. 249, no. 1, pp. 3 – 80, 2000.

[138] J. Guo, "Using category theory to model software component dependencies," in *Proc. of the Ninth Annual IEEE International Conference and Workshop on the Engineering of Computer-Based Systems*, 2002, pp. 185 –192.

[139] J. Hou, J. Wan, and S. Wang, "Formalization of architecture-centric model mapping using category theory," in *Proc. of ACIS SNPD 2007*, vol. 1. IEEE, 2007, pp. 670–675.

[140] J. Huang, "Modeling multi-agent systems with category theory," Ph.D. dissertation, Concordia University, 2011.

[141] S. Mardesic and J. Segal, "Shape theory," *North-Holland Math. Library*, vol. 26, 1982.

[142] A. Frank, "Spatial communication with maps: Defining the correctness of maps using a multi-agent simulation," *Spatial Cognition II*, pp. 80–99, 2000.

[143] M. Heather and B. Rossiter, "Information systems and the theory of categories. is every model an anticipatory system," *International Journal Computing Anticipatory Systems*, vol. 16, pp. 219–231, 2004.

[144] H. Balakrishnan, S. Seshan, E. Amir, and R. Katz, "Improving tcp/ip performance over wireless networks," in *Proc. of the 1st annual international conference on Mobile computing and networking*. ACM, 1995, pp. 2–11.

[145] H. Balakrishnan, V. Padmanabhan, S. Seshan, and R. Katz, "A comparison of mechanisms for improving tcp performance over wireless links," *Networking, IEEE/ACM Transactions on*, vol. 5, no. 6, pp. 756–769, 1997.

[146] "Indriya telosb testbed at national university of singapore," http://indriya.comp.nus.edu.sg/motelab/html/index.php [Last visited: 02.08.2011].

[147] J. Mitola and G. Maguire, "Cognitive radio: making software radios more personal," *IEEE Personal Communications*, vol. 6, no. 4, pp. 13–18, 1999.

[148] J. Kephart and D. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, 2003.

[149] T. Gruber *et al.*, "A translation approach to portable ontology specifications," *Knowledge acquisition*, vol. 5, no. 2, pp. 199–220, 1993.

[150] P. Castells, M. Fernandez, and D. Vallet, "An adaptation of the vector-space model for ontology-based information retrieval," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 2, pp. 261–272, 2007.

[151] H. Müller, E. Kenny, and P. Sternberg, "Textpresso: an ontology-based information retrieval and extraction system for biological literature," *PLoS biology*, vol. 2, no. 11, p. e309, 2004.

[152] J. Tejedor, R. García, M. Fernández, F. López-Colino, F. Perdrix, J. Macías, R. Gil, M. Oliva, D. Moya, J. Colás *et al.*, "Ontology-based retrieval of human speech," in *Proc of the 18th International Workshop on Database and Expert Systems Applications*. IEEE, 2007, pp. 485–489.

[153] "Description of cognitive radio ontology (vol. 1.0), winnf-10-s-0007," 2010.

[154] D. Nardi and R. Brachman, "An introduction to description logics," *The description logic handbook: theory, implementation, and applications*, pp. 1–40, 2003.

[155] K. Balasubramanian, A. Gokhale, G. Karsai, J. Sztipanovits, and S. Neema, "Developing applications using model-driven design environments," *IEEE Computer*, vol. 39, no. 2, pp. 33–40, 2006.

[156] A. e. Pinto, "System level design paradigms: Platform-based design and communication synthesis," *ACM Transactions on Design Automation of Electronic Systems*, vol. 11, no. 3, pp. 537–563, 2006.

[157] D. G. Messerschmitt, "Rethinking Components: From Hardware and Software to Systems," *Proceedings of the IEEE*, vol. 95, no. 7, pp. 1473–1496, 2007.

[158] T. Arampatzis, J. Lygeros, and S. Manesis, "A survey of applications of wireless sensors and wireless sensor networks," in *Proc. of the IEEE Int. Symp. on, Mediterrean Conference on Control and Automation*, 2005.

[159] K. Romer and F. Mattern, "The design space of wireless sensor networks," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 54 – 61, December 2004.

[160] Y. Mao, F. Wang, L. Qiu, S. S. Lam, and J. M. Smith, "S4: small state and small stretch routing protocol for large wireless sensor networks," in *Proc. of the 4th USENIX conference on NSDI*, 2007.

[161] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection Tree Protocol," in *Proc. of the ACM Conf. on Embedded Networked Sensor Systems*, 2009.

[162] *TelosB Wireless Sensor Node*, Crossbow Technology Inc., http://memsic.com/products/wireless-sensor-networks/wireless-modules.html, February 2006, Last visited: 11.12.2012.

[163] P. Levis and D. Gay, *TinyOS Programming*. Cambridge University Press, 2009.

[164] C. Jardak, E. Meshkova, J. Riihijärvi, K. Rerkrai, and P. Mähönen, "Implementation and performance evaluation of nanoip protocols: Simplified versions of tcp, udp, http and slp for wireless sensor networks," in *Proc. of IEEE WCNC 2008*, 2008, pp. 2474–2479.

[165] J. Ansari, X. Zhang, O. Salikeen and P. Mähönen, "Enabling Flexible MAC Protocol Design for Wireless Sensor Networks," in *Proc. of the IEEE International Conference on Wireless On-demand Network Systems and Services*, 2011.

[166] B. Y. Alkazemi and E. A. Felemban, "Towards a framework for engineering software development of sensor nodes in wireless sensor networks," in *Proc. of the Workshop on Software Engineering for Sensor Network Applications*, 2010, pp. 72–75.

[167] A. Taherkordi, F. Loiret, R. Rouvoy, and F. Eliassen, "A Generic Component-Based Approach for Programming, Composing and Tuning Sensor Software," *The Computer Journal*, vol. 54, no. 8, pp. 1248–1266, 2011.

[168] A. Köpke, V. Handziski, J. Hauer, and H. Karl, "Structuring the information flow in component-based protocol implementations for wireless sensor nodes," in *Proc. of Work-in-Progress Session of European Workshop on Wireless Sensor Networks*, 2004.

[169] B. Yahya and J. Ben-Othman, "Towards a classification of energy aware MAC protocols for wireless sensor networks," *Wireless Communications and Mobile Computing*, vol. 9, no. 12, pp. 1572–1607, 2009.

[170] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," *Ad Hoc Networks*, vol. 3, pp. 325–349, 2005.

[171] R. Rajagopalan and P. Varshney, "Data-aggregation techniques in sensor networks: a survey," *IEEE Communications Surveys & Tutorials*, vol. 8, no. 4, pp. 48–63, 2006.

[172] C. Wang, K. Sohraby, B. Li, M. Daneshmand, and Y. Hu, "A survey of transport protocols for wireless sensor networks," *IEEE Network Magazine*, vol. 20, no. 3, pp. 34–40, 2006.

[173] S. Madden, M. Franklin, J. Hellerstein, and W. Hong, "TinyDB: An acquisitional query processing system for sensor networks," *ACM Transactions on Database Systems*, vol. 30, no. 1, pp. 122–173, 2005.

[174] J. Il Choi, M. Kazandjieva, M. Jain, and P. Levis, "The case for a network protocol isolation layer," in *Proc. of the 7th ACM Conference on Embedded Networked Sensor Systems*. ACM, 2009, pp. 267–280.

[175] M. Sooriyabandara, T. Farnham, P. Mahonen, M. Petrova, J. Riihijarvi, and Z. Wang, "Generic interface architecture supporting cognitive resource management in future wireless networks," *IEEE Communications Magazine*, vol. 49, no. 9, pp. 103–113, 2011.

[176] L. Mottola and G. Picco, "Programming wireless sensor networks: Fundamental concepts and state of the art," *ACM Computing Surveys (CSUR)*, vol. 43, no. 3, p. 19, 2011.

[177] K. Whitehouse, F. Zhao, and J. Liu, "Semantic streams: A framework for composable semantic interpretation of sensor data," *Wireless Sensor Networks*, pp. 5–20, 2006.

[178] R. Sugihara and R. Gupta, "Programming models for sensor networks: A survey," *ACM Transactions on Sensor Networks*, vol. 4, no. 2, p. 8, 2008.

[179] J. He, J. Rexford, and M. Chiang, "Don't optimize existing protocols, design optimizable protocols," *ACM SIGCOMM CCR*, vol. 37, pp. 53–58, 2007.

[180] X. Lin, N. B. Shroff, and R. Srikant, "A tutorial on cross-layer optimization in wireless networks," *IEEE Journal on Selected Areas in Communication*, vol. 24, pp. 1452–1463, 2006.

[181] K. Klues, G. Hackmann, O. Chipara, and C. Lu, "A Component Based Architecture for Power-Efficient Media Access Control in Wireless Sensor Networks," in *Proceedings of the ACM SenSys 2007*, November 2007.

[182] C. Ivan, V. Dadarlat, and K. Pusztai, "Managing complexity of designing routing protocols using a middleware approach," in *Proc. of the 14th International Conference on Advanced Information Systems Engineering*, ser. CAiSE '02, 2002.

[183] J. S. Baras and H. Huang, "Component based routing: A new methodology for designing routing protocols for MANET," in *25th Army Science Conference*, 2006.

[184] M. Fowler and K. Scott, *UML distilled: applying the standard object modeling language*. Addison-Wesley Professional, 1997.

[185] J. Ellsberger, D. Hogrefe, and A. Sarma, *SDL: formal object-oriented language for communicating systems*. Prentice Hall, 1997.

[186] A. Murphy, G. Picco, and G. Roman, "Lime: A middleware for physical and logical mobility," in *Proc. of the 21st International Conference on Distributed Computing Systems*. IEEE, 2001, pp. 524–533.

[187] P. Eugster, P. Felber, R. Guerraoui, and A. Kermarrec, "The many faces of publish/subscribe," *ACM Computing Surveys*, vol. 35, no. 2, pp. 114–131, 2003.

[188] G. Antoniou and F. van Harmelen, "Web Ontology Language: OWL," *Handbook on Ontologies*, 2004.

[189] P. Baker, V. Catterson, and S. McArthur, "Integrating an agent-based wireless sensor network within an existing multi-agent condition monitoring system," in *Proc. of the Int. Conf. on Intelligent System Applications to Power Systems*, 2009, pp. 1–6.

[190] R. Jurdak, C. Lopes, and P. Baldi, "A framework for modeling sensor networks," in *Proc. of the Building Software for Pervasive Computing*, 2004.

[191] D. Brickley and R. Guha, "RDF Vocabulary Description Language 1.0: RDF Schema," *W3C Working Draft*, vol. 23, 2003.

[192] D. Elenius, G. Denker, M. Stehr, R. Senanayake, C. Talcott, and D. Wilkins, "CoRaL–Policy language and reasoning techniques for spectrum policies," in *Proc. of IEEE Policies for Distributed Systems and Networks*, 2007, pp. 261–265.

[193] O. Lassila and R. Swick, "Resource description framework (RDF) model and syntax," *World Wide Web Consortium, http://www. w3. org/TR/WD-rdf-syntax*, 1999.

[194] H. Knublaich, R. Fergerson, N. Noy, and M. Musen, "The protégé OWL plugin: An open development environment for Semantic Web applications," *Lecture notes in computer science*, pp. 229–243, 2004.

[195] D. Tsarkov and I. Horrocks, "FaCT++ description logic reasoner: System description," in *Proc. of the Int. Joint Conf. on Automated Reasoning*, Seattle, Washington, USA, August 2006.

[196] J. J. Carroll and et al., "Jena: implementing the semantic web recommendations," in *Proc. of WWW2004*, New York, NY, USA, May 2004.

[197] S. Bechhofer, "The DIG Description Logic Interface: DIG/1.1," in *Proceedings of DL2003*, Rome, Italy, September 2003.

[198] E. Sirin, B. Parsia, B. Grau, A. Kalyanpur, and Y. Katz, "Pellet: A practical OWL-DL reasoner," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 5, no. 2, pp. 51–53, 2007.

[199] I. Horrocks, P. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, "SWRL: A Semantic Web Rule Language Combining OWL and RuleML," *W3C Member Submission*, vol. 21, 2004.

[200] J. Kopena and W. Regli, "DAMLJessKB: A Tool for Reasoning with the Semantic Web," *IEEE Intelligent Systems*, vol. 18, no. 3, pp. 74–77, 2003.

[201] J. Polastre, J. Hui, P. Levis, J. Zhao, D. E. Culler, S. Shenker, and I. Stoica, "A unifying link abstraction for wireless sensor networks," in *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, 2005, pp. 76–89.

[202] M. Avvenuti, P. Corsini, P. Masci and A. Vecchio, "Increasing the efficiency of preamble sampling protocols for wireless sensor networks," in *Proc. of the First International Conference on Mobile Computing and Wireless Communication*, 2006.

[203] M. Buettner, G. Yee, E. Anderson, and R. Han, "X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks," in *Proc. of the 4th international conference on Embedded networked sensor systems*. ACM, 2006, pp. 307–320.

[204] J. Ansari, X. Zhang and P. Mähönen, "Traffic Aware Medium Access Control Protocol for Wireless Sensor Networks," in *Proc. of the 7th ACM International Symposium on Mobility Management and Wireless Access*, 2009.

[205] ——, "Practical Experiences and Design Considerations on Medium Access Control Protocols for Wireless Sensor Networks in Handbook of Research on Developments and Trends in Wireless Sensor Networks: From Principle to Practice by H. Jin and W. Jiang (editors)," *IGI Global*, 2010.

[206] E. Sirin and B. Parsia, "Sparql-DL: Sparql query for OWL-DL," in *3rd OWL Experiences and Directions Workshop (OWLED-2007)*, vol. 4, 2007.

[207] K. Rerkrai, J. Riihijarvi, P. Mahonen, and F. Oldewurtel, "UDAE: Universal Data Access Engine for Sensor Networks," in *Proc. of IEEE SECON 2008*. IEEE, 2008, pp. 523–532.

[208] "MATLAB: Global Optimization Toolbox," http://www.mathworks.com/help/toolbox/gads/f6010dfi3.html, [last visited 13.07.2012].

[209] B. McKay, "Nauty user's guide (version 2.4)," *Computer Science Dept., Australian National University*, 2007.

[210] "MultiHopLQI wireless sensor network routing algorithm for TinyOS2.x," http://www.tinyos.net/tinyos-2.x/tos/lib/net/lqi/ [Last Visited 21.03.2012].

[211] K. El Amrani and U. Mansmann, "Description of the graphcomp package: Visual comparison of graphs on the same node set," 2011.

[212] X. Zhang, J. Ansari, and P. Mähönen, "TrawMAC:traffic aware medium access control protocol for wireless sensor networks," in *Proc. of ACM MobiWAC 2009*, 2009.

[213] A. Dunkels, F. Osterlind, N. Tsiftes, and Z. He, "Software-based on-line energy estimation for sensor nodes," in *Proc. of the 4th workshop on Embedded networked sensors*, 2007.

[214] R. Szewczyk, A. Mainwaring, J. Polastre, J. Anderson, and D. Culler, "An analysis of a large scale habitat monitoring application," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, 2004.

[215] R. Haesen, M. Snoeck, W. Lemahieu, and S. Poelmans, "On the definition of service granularity and its architectural impact," in *Advanced Information Systems Engineering*. Springer, 2008, pp. 375–389.

[216] J. Ansari, X. Zhang, A. Achtzehn, M. Petrova, and P. Mähönen, "Decomposable MAC Framework for Highly Flexible and Adaptable MAC Realizations," in *Proceedings of the IEEE Symposium on New Frontiers in Dynamic Spectrum*, 2010.

[217] B. Kauffmann, F. Baccelli, A. Chaintreau, V. Mhatre, K. Papagiannaki, and C. Diot, "Measurement-based self organization of interfering 802.11 wireless access networks," in *Proc. of IEEE INFOCOM 2007*. IEEE, 2007, pp. 1451–1459.

[218] A. Moore and D. Zuev, "Internet traffic classification using bayesian analysis techniques," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 33, no. 1. ACM, 2005, pp. 50–60.

[219] A. Mishra, *Advanced cellular network planning and optimisation*. Wiley Online Library, 2007.

[220] Y. Shi and Y. Hou, "A distributed optimization algorithm for multi-hop cognitive radio networks," in *Proc. of IEEE INFOCOM 2008*. IEEE, 2008, pp. 1292–1300.

[221] D. Kivanc, G. Li, and H. Liu, "Computationally efficient bandwidth allocation and power control for ofdma," *IEEE Transactions on Wireless Communications*, vol. 2, no. 6, pp. 1150–1158, 2003.

[222] J. Riihijärvi and P. Mähönen, "Modeling Spatial Structure of Wireless Communication Networks," in *Proc. of IEEE NetSciCom workshop, in conjunction with IEEE INFOCOM 2010*, 2010, pp. 1–6.

[223] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless communications and mobile computing*, vol. 2, no. 5, pp. 483–502, 2002.

[224] M. Birattari, *Tuning Metaheuristics: A machine learning perspective*. Springer, 2009, vol. 197.

[225] J. Pearl, *Heuristics: intelligent search strategies for computer problem solving*. Addison-Wesley Pub. Co., Inc., Reading, MA, 1984.

[226] A. He, K. K. Bae, T. Newman, J. Gaeddert, K. Kim, R. Menon, L. Morales-Tirado, J. J. Neel, Y. Zhao, J. H. Reed *et al.*, "A survey of artificial intelligence for cognitive radios," *Vehicular Technology, IEEE Transactions on*, vol. 59, no. 4, pp. 1578–1592, 2010.

[227] C. Clancy, J. Hecker, E. Stuntebeck, and T. O'Shea, "Applications of machine learning to cognitive radio networks," *IEEE Wireless Communications Magazine*, vol. 14, no. 4, pp. 47–52, 2007.

[228] R. Kulkarni, A. Forster, and G. Venayagamoorthy, "Computational intelligence in wireless sensor networks: A survey," *IEEE Communications Surveys and Tutorials*, vol. 13, no. 1, pp. 68–96, 2011.

[229] T. Rondeau and C. Bostian, *Artificial intelligence in wireless communications*. Artech House Publishers, 2009.

[230] F. Dressler and O. Akan, "A survey on bio-inspired networking," *Computer Networks*, vol. 54, no. 6, pp. 881–900, 2010.

[231] A. Forster, "Machine learning techniques applied to wireless ad-hoc networks: Guide and survey," in *Proc. of 3rd International Conference on Intelligent Sensors, Sensor Networks and Information*. IEEE, 2007, pp. 365–370.

[232] D. Brélaz, "New methods to color the vertices of a graph," *Communications of the ACM*, vol. 22, no. 4, pp. 251–256, 1979.

[233] P. Mitran, L. Le, C. Rosenberg, and A. Girard, "Resource allocation for downlink spectrum sharing in cognitive radio networks," in *Proc. of IEEE VTC 2008-Fall*. IEEE, 2008, pp. 1–5.

[234] M. Younis and K. Akkaya, "Strategies and techniques for node placement in wireless sensor networks: A survey," *Ad Hoc Networks*, vol. 6, no. 4, pp. 621–655, 2008.

[235] D. Waterman, "Generalization learning techniques for automating the learning of heuristics," *Artificial Intelligence*, vol. 1, no. 1, pp. 121–170, 1970.

[236] M. Shaw, S. PARK, and N. Raman, "Intelligent scheduling with machine learning capabilities: the induction of scheduling knowledge," *IIE transactions*, vol. 24, no. 2, pp. 156–168, 1992.

[237] F. Agakov, E. Bonilla, J. Cavazos, B. Franke, G. Fursin, M. O'Boyle, J. Thomson, M. Toussaint, and C. Williams, "Using machine learning to focus iterative optimization," in *Proceedings of the International Symposium on Code Generation and Optimization*. IEEE Computer Society, 2006, pp. 295–305.

[238] M. Ball and M. Magazine, "Sequencing of insertions in printed circuit board assembly," *Operations Research*, pp. 192–201, 1988.

[239] W. Gutjahr, "Recent trends in metaheuristics for stochastic combinatorial optimization," *Central European Journal of Computer Science*, vol. 1, no. 1, pp. 58–66, 2011.

[240] L. Bianchi, M. Dorigo, L. Gambardella, and W. Gutjahr, "A survey on metaheuristics for stochastic combinatorial optimization," *Natural Computing*, vol. 8, no. 2, pp. 239–287, 2009.

[241] C. Blum, J. Puchinger, G. Raidl, and A. Roli, "Hybrid metaheuristics in combinatorial optimization: A survey," *Applied Soft Computing*, vol. 11, no. 6, pp. 4135–4151, 2011.

[242] E. Talbi, "A taxonomy of hybrid metaheuristics," *Journal of heuristics*, vol. 8, no. 5, pp. 541–564, 2002.

[243] X. Wang, J. Ma, S. Wang, and D. Bi, "Distributed particle swarm optimization and simulated annealing for energy-efficient coverage in wireless sensor networks," *Sensors*, vol. 7, no. 5, pp. 628–648, 2007.

[244] M. Birattari, M. Zlochin, and M. Dorigo, "Towards a theory of practice in metaheuristics design: A machine learning perspective," *RAIRO-Theoretical Informatics and Applications*, vol. 40, no. 02, pp. 353–369, 2006.

[245] W. Wang and Z. Xu, "A heuristic training for support vector regression," *Neurocomputing*, vol. 61, pp. 259–275, 2004.

[246] S. Walczak and N. Cerpa, "Heuristic principles for the design of artificial neural networks," *Information and software technology*, vol. 41, no. 2, pp. 107–117, 1999.

[247] P. Pendharkar, "Genetic algorithm based neural network approaches for predicting churn in cellular wireless network services," *Expert Systems with Applications*, vol. 36, no. 3, pp. 6714–6720, 2009.

[248] C. Potter, G. Venayagamoorthy, and K. Kosbar, "Mimo beam-forming with neural network channel prediction trained by a novel pso-ea-depso algorithm," in *Proc. of IEEE 2008 International Joint Conference on Neural Networks*. IEEE, 2008, pp. 3338–3344.

[249] P. Kampstra, R. van der Mei, and A. Eiben, "Evolutionary computing in telecommunication network design: A survey," VU University Amsterdam, Tech. Rep., 2006, available on: http://www.math.vu.nl/mei/publications.php [Last visited on 12.11.2012].

[250] K. Ferentinos and T. Tsiligiridis, "Adaptive design optimization of wireless sensor networks using genetic algorithms," *Computer Networks*, vol. 51, no. 4, pp. 1031–1051, 2007.

[251] A. Konstantinidis, K. Yang, Q. Zhang, and D. Zeinalipour-Yazti, "A multi-objective evolutionary algorithm for the deployment and power assignment problem in wireless sensor networks," *Computer Networks*, vol. 54, no. 6, pp. 960–976, 2010.

[252] A. Konstantinidis, K. Yang, H.-H. Chen, and Q. Zhang, "Energy-aware topology control for wireless sensor networks using memetic algorithms," *Computer Communications*, vol. 30, no. 1415, pp. 2753 – 2764, 2007.

[253] C.-K. Ting and C.-C. Liao, "A memetic algorithm for extending wireless sensor network lifetime," *Information Sciences*, vol. 180, no. 24, pp. 4818–4833, 2010.

[254] C. Rieser, T. Rondeau, C. Bostian, and T. Gallagher, "Cognitive radio testbed: further details and testing of a distributed genetic algorithm based cognitive engine for programmable radios," in *Proc. of IEEE MILCOM 2004*, vol. 3. IEEE, 2004, pp. 1437–1443.

[255] A. de Baynast, P. Mähönen, and M. Petrova, "ARQ-based cross-layer optimization for wireless multicarrier transmission on cognitive radio networks," *Computer networks*, vol. 52, no. 4, pp. 778–794, 2008.

[256] T. Newman, B. Barker, A. Wyglinski, A. Agah, J. Evans, and G. Minden, "Cognitive engine implementation for wireless multicarrier transceivers," *Wireless communications and mobile computing*, vol. 7, no. 9, pp. 1129–1142, 2007.

[257] D. Huang, C. Leung, and C. Miao, "Memetic algorithm for dynamic resource allocation in multiuser ofdm based cognitive radio systems," in *Proc. of the IEEE Congress on Evolutionary Computation*, 2008, pp. 3860–3865.

[258] Z. Zhao, Z. Peng, S. Zheng, and J. Shang, "Cognitive radio spectrum allocation using evolutionary algorithms," *IEEE Transactions on Wireless Communications*, vol. 8, no. 9, pp. 4421–4425, 2009.

[259] Z. Yangyang, J. Chunlin, Y. Ping, L. Manlin, W. Chaojin, and W. Guangxing, "Particle swarm optimization for base station placement in mobile communication," in *Proc. of the 2004 IEEE International Conference on Networking, Sensing and Control*, vol. 1. IEEE, 2004, pp. 428–432.

[260] X. Wang, S. Wang, and J. J. Ma, "An improved co-evolutionary particle swarm optimization for wireless sensor networks with dynamic deployment," *Sensors*, vol. 7, no. 3, pp. 354–370, 2007.

[261] T. Renk, C. Kloeck, D. Burgkhardt, F. Jondral, D. Grandblaise, S. Gault, and J. Dunat, "Bio-inspired algorithms for dynamic resource allocation in cognitive wireless networks," *Mobile Networks and Applications*, vol. 13, no. 5, pp. 431–441, 2008.

[262] R. Kulkarni and G. Venayagamoorthy, "Particle swarm optimization in wireless-sensor networks: A brief survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 41, no. 2, pp. 262–267, 2011.

[263] I. Kassabalidis, M. El-Sharkawi, R. Marks, P. Arabshahi, A. Gray *et al.*, "Swarm intelligence for routing in communication networks," in *Proc. of IEEE GlobeCom 2001*, vol. 6. IEEE, 2001, pp. 3613–3617.

[264] F. Ducatelle, G. Di Caro, and L. Gambardella, "Principles and applications of swarm intelligence for adaptive routing in telecommunications networks," *Swarm Intelligence*, vol. 4, no. 3, pp. 173–198, 2010.

[265] J. Tillett, R. Rao, and F. Sahin, "Cluster-head identification in ad hoc sensor networks using particle swarm optimization," in *Proc. of IEEE International Conference on Personal Wireless Communications*. IEEE, 2002, pp. 201–205.

[266] N. Latiff, C. Tsimenidis, and B. Sharif, "Performance comparison of optimization algorithms for clustering in wireless sensor networks," in *Proc. of IEEE MASS 2007*. IEEE, 2007, pp. 1–4.

[267] A. Gopakumar and L. Jacob, "Localization in wireless sensor networks using particle swarm optimization," in *Proc. of IET International Conference on Wireless, Mobile and Multimedia Networks*. IET, 2008, pp. 227–230.

[268] Y. Yuan, Z. He, and M. Chen, "Virtual mimo-based cross-layer design for wireless sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 55, no. 3, p. 856, 2006.

[269] J. Zhang, Z. Zhou, W. Gao, Y. Ma, and Y. Ye, "Cognitive radio adaptation decision engine based on binary quantum-behaved particle swarm optimization," in *Proc. of CHINACOM 2011*. IEEE, 2011, pp. 221–225.

[270] E. Güney, İ. Altınel, N. Aras, and C. Ersoy, "A tabu search heuristic for point coverage, sink location, and data routing in wireless sensor networks," *Evolutionary Computation in Combinatorial Optimization*, pp. 83–94, 2010.

[271] M. Kamenetsky and M. Unbehaun, "Coverage planning for outdoor wireless lan systems," in *Proc. of 2002 International Zurich Seminar on Broadband Communications*. IEEE, 2002, pp. 49–1.

[272] I. Demirkol, C. Ersoy, M. Caglayan, and H. Delic, "Location area planning in cellular networks using simulated annealing," in *Proc. of IEEE INFOCOM 2001*, vol. 1. IEEE, 2001, pp. 13–20.

[273] A. Kannan, G. Mao, and B. Vucetic, "Simulated annealing based localization in wireless sensor network," in *Proc. of IEEE LCN 2005*. IEEE, 2005, pp. 2–pp.

[274] J. Nasreddine, J. Perez-Romero, O. Salient, and R. Agusti, "Simulated annealing-based advanced spectrum management methodology for wcdma systems," in *Proc. of IEEE ICC 2008*, May 2008, pp. 2625–2631.

[275] T. Back, *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*. Oxford University Press, USA, 1996.

[276] P. Moscato *et al.*, "On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms," *Caltech concurrent computation program, C3P Report*, vol. 826, p. 1989, 1989.

[277] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. of the IEEE 1995 International Conference on Neural Networks*, vol. 4. IEEE, 1995, pp. 1942–1948.

[278] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 26, no. 1, pp. 29–41, 1996.

[279] S. Kirkpatrick, M. Vecchi *et al.*, "Optimization by simmulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[280] S. Voß, "Meta-heuristics: The state of the art," *Local Search for Planning and Scheduling*, pp. 1–23, 2001.

[281] F. Glover, "Tabu search and adaptive memory programming-advances, applications and challenges," *Interfaces in computer science and operations research*, vol. 1, 1996.

[282] R. Montemanni, L. Gambardella, and A. Das, "The minimum power broadcast problem in wireless networks: a simulated annealing approach," in *Proc. of IEEE WCNC 2005*, vol. 4. IEEE, 2005, pp. 2057–2062.

[283] A. El Rhazi and S. Pierre, "A tabu search algorithm for cluster building in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 8, no. 4, pp. 433–444, 2009.

[284] Z. Han and K. Liu, *Resource allocation for wireless networks: basics, techniques, and applications*. Cambridge University Press, 2008.

[285] W. R. Mebane, Jr. and J. S. Sekhon, "Genetic optimization using derivatives: The rgenoud package for R," *Journal of Statistical Software*, vol. 42, no. 11, pp. 1–26, 2011. [Online]. Available: http://www.jstatsoft.org/v42/i11/

[286] M. Zambrano-Bigiarini, *hydroPSO: Model-Independent Particle Swarm Optimisation for Environmental Models*, 2012, r package version 0.1-57. [Online]. Available: http://CRAN.R-project.org/package=hydroPSO

[287] C. Bergmeir, D. Molina, and J. M. Benítez, *Continuous Optimization using Memetic Algorithms with Local Search Chains (MA-LS-Chains) in R*, 2012.

[288] H. Xia, H. Bertoni, L. Maciel, A. Lindsay-Stewart, and R. Rowe, "Radio propagation characteristics for line-of-sight microcellular and personal communications," *IEEE Transactions on Antennas and Propagation*, vol. 41, no. 10, pp. 1439–1447, 1993.

[289] J. Nasreddine, J. Riihijärvi, and P. Mähönen, "Transmit power control for secondary use in environments with correlated shadowing," in *Proc. of IEEE ICC 2011*. IEEE, 2011, pp. 1–6.

[290] A. El Dor, M. Clerc, and P. Siarry, "Hybridization of differential evolution and particle swarm optimization in a new algorithm: Depso-2s," *Swarm and Evolutionary Computation*, pp. 57–65, 2012.

[291] "QualNet," scalable Network Technologies, http://www.scalable-networks.com [last visited on 23.03.2009].

[292] M. Mitchell, *An introduction to genetic algorithms*. Bradford Books, 1996.

[293] G. Harik *et al.*, "Linkage learning via probabilistic modeling in the ECGA," *Urbana*, vol. 51, no. 61, p. 801, 1999.

[294] D. Newman, "The use of linkage learning in genetic algorithms," 2006.

[295] M. Birattari, T. Stützle, L. Paquete, and K. Varrentrapp, "A racing algorithm for configuring metaheuristics," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2002, pp. 11–18.

[296] M. Pelikan, D. Goldberg, and E. Cantu-Paz, "Linkage problem, distribution estimation, and bayesian networks," *Evolutionary Computation*, vol. 8, no. 3, pp. 311–340, 2000.

[297] I. Ben-Gal, *Bayesian Networks, Encyclopedia of Statistics in Quality and Reliability*. John Wiley and Sons, 2007.

[298] M. Garetto, J. Shi, and E. Knightly, "Modeling media access in embedded two-flow topologies of multi-hop wireless networks," in *Proc. of the 11th annual international conference on Mobile computing and networking*. ACM, 2005, pp. 200–214.

[299] K. Murphy *et al.*, "The Bayes Net Toolbox for Matlab," *Computing Science and Statistics*, vol. 33, no. 2, pp. 1024–1034, 2001.

[300] C. Borgelt and R. Kruse, *Graphical Models: Methods for Data Analysis and Mining*. John Wiley and Sons, 2002.

[301] X. Cheng, C. Dale, and J. Liu, "Understanding the Characteristics of Internet Short Video Sharing: YouTube as a Case Study," in *Proc. of the 7th ACM SIGCOMM Conference on Internet Measurement*, 2007.

[302] "iperf versions 3.x and 2.x," http://code.google.com/p/iperf/,and\url{http://iperf.sourceforge.net/},respectively [Last visited: 06.06.2012].

[303] S. Ganu, K. Ramachandran, M. Gruteser, I. Seskar, and J. Deng, "Methods for restoring mac layer fairness in IEEE 802.11 networks with physical layer capture," in *Proc. of the 2nd international workshop on Multi-hop ad hoc networks: from theory to reality*. ACM, 2006, pp. 7–14.

[304] K. Pahlavan, P. Krishnamurthy, A. Hatami, M. Ylianttila, J. Mäkelä, R. Pichna, and J. Vallstron, "Handoff in hybrid mobile data networks," *IEEE Personal Communications*, vol. 7, no. 2, pp. 34–47, 2000.

[305] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu, "Impact of interference on multi-hop wireless network performance," in *Proc. of ACM MobiCom 2003*, 2003, pp. 66–80.

[306] H. Li and *et. al*, "Multi-dimensional conflict graph based computing for optimal capacity in MR-MC wireless networks," in *Proc. of IEEE ICDCS 2010*, 2010, pp. 774–783.

[307] A. Khattab, J. Camp, C. Hunter, P. Murphy, A. Sabharwal and E. W. Knightly, "WARP: a flexible platform for clean-slate wireless medium access protocol design," *SIGMOBILE Mobile Computing Communication Review*, vol. 12, no. 1, pp. 56–58, 2008.

[308] M. Haenggi, J. G. Andrews, F. Baccelli, O. Dousse, and M. Franceschetti, "Stochastic Geometry and Random Graphs for the Analysis and Design of Wireless Networks," *IEEE Journal on Selected Areas in Communications*, September 2009.

[309] R. Ganti and M. Haenggi, "Interference and outage in clustered wireless ad hoc networks," *IEEE Transactions on Information Theory*, vol. 55, no. 9, pp. 4067–4086, 2009.

[310] R. Albert and A. Barabási, "Statistical mechanics of complex networks," *Reviews of Modern Physics*, vol. 74, no. 1, pp. 47–97, 2002.

[311] M. Newman, "Random graphs as models of networks," *Handbook of graphs and networks*, p. 35.

[312] W. Willinger, D. Alderson, and J. Doyle, "Mathematics and the Internet: A source of enormous confusion and great potential," *Notices of the American Mathematical Society*, vol. 56, no. 5, pp. 586–599, 2009.

[313] R. V. Oliveira, D. Pei, W. Willinger, B. Zhang, and L. Zhang, "In search of the elusive ground truth: the internet's AS-level connectivity structure," *SIGMETRICS Performance Evaluation Review*, vol. 36, no. 1, 2008.

[314] L. Li, D. Alderson, J. Doyle, and W. Willinger, "Towards a theory of scale-free graphs: Definition, properties, and implications," *Internet Mathematics*, vol. 2, no. 4, pp. 431–523, 2005.

[315] M. Garetto, T. Salonidis, and E. Knightly, "Modeling per-flow throughput and capturing starvation in CSMA multi-hop wireless networks," *IEEE/ACM Transactions on Networking*, vol. 16, no. 4, pp. 864–877, 2008.

[316] C. Hua and R. Zheng, "Starvation modeling and identification in dense 802.11 wireless community networks," in *Proc. of IEEE INFOCOM 2008*, 2008.

[317] S. Razak, V. Kolar, and N. Abu-Ghazaleh, "Modeling and analysis of two-flow interactions in wireless networks," *Ad Hoc Networks*, vol. 8, no. 6, pp. 564–581, 2010.

[318] E. Magistretti, O. Gurewitz, and E. Knightly, "Inferring and mitigating a link's hindering transmissions in managed 802.11 wireless networks," in *Proc. of ACM MOBICOM 2010*, 2010, pp. 305–316.

[319] Y. Wu, P. Chou, Q. Zhang, K. Jain, W. Zhu, and S. Kung, "Network planning in wireless ad hoc networks: a cross-layer approach," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 1, pp. 136–150, 2005.

[320] V. Shrivastava *et al.*, "PIE in the sky: online passive interference estimation for enterprise wlans," in *Proc. of ACM NSDI 2011*, 2011.

[321] V. Mhatre, K. Papagiannaki, and F. Baccelli, "Interference mitigation through power control in high density 802.11 WLANs," in *Proc. of IEEE INFOCOM 2007*. IEEE, 2007, pp. 535–543.

[322] N. Ahmed *et al.*, "Online estimation of RF interference," in *Proc. of ACM CoNEXT 2008*, 2008.

[323] S. Rayanchu *et al.*, "FLUID: improving throughputs in enterprise wireless LANs through flexible channelization," in *Proc. of ACM MobiCom 2011*, 2011.

[324] X. Liu *et al.*, "DIRC: increasing indoor wireless capacity using directional antennas," in *ACM SIGCOMM CCR*, vol. 39, no. 4, 2009, pp. 171–182.

[325] U. Alon, "Network motifs: theory and experimental approaches," *Nature Reviews Genetics*, vol. 8, no. 6, pp. 450–461, 2007.

[326] D. Traskov, M. Heindlmaier, M. Medard, R. Koetter, and D. Lun, "Scheduling for network coded multicast: A conflict graph formulation," in *Proc. of IEEE GLOBECOM 2008 Workshops*. IEEE, 2008, pp. 1–5.

[327] Q. Li, G. Kim, and R. Negi, "Maximal scheduling in a hypergraph model for wireless networks," in *Proc. of IEEE ICC 2008*, May 2008.

[328] P. Gupta and P. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 388–404, 2000.

[329] S. Kwon and N. Shroff, "Paradox of shortest path routing for large multi-hop wireless networks," in *Proc. of IEEE INFOCOM 2007*. IEEE, 2007, pp. 1001–1009.

[330] M. Afanasyev, T. Chen, G. Voelker, and A. Snoeren, "Usage patterns in an urban WiFi network," *IEEE/ACM Transactions on Networking*, vol. 18, no. 5, pp. 1359–1372, 2010.

[331] M. Penrose, *Random geometric graphs*. Oxford University Press, 2003.

[332] D. Stoyan, W. S. Kendall, and J. Mecke, *Stochastic geometry and its applications*. Wiley, 1995.

[333] A. F. Karr, *Point Processes and Their Statistical Inference*, 2nd ed. Marcel Dekker, 1991.

[334] J. Riihijärvi, P. Mähönen, and M. Rübsamen, "Characterizing wireless networks by spatial correlations," *Communications Letters, IEEE*, vol. 11, no. 1, pp. 37–39, 2007.

[335] J. Riihijärvi and M. Petrova and P. Mähönen, "Influence of node location distributions on the structure of ad hoc and mesh networks," in *Proc. of IEEE Globecom 2008*, New Orleans, USA, November 2008.

[336] K. Ramachandran, E. Belding, K. Almeroth, and M. Buddhikot, "Interference-aware channel assignment in multi-radio wireless mesh networks," in *Proc. of IEEE INFOCOM 2006*. IEEE, 2006, pp. 1–12.

[337] S. Sengupta, S. Rayanchu, and S. Banerjee, "An analysis of wireless network coding for unicast sessions: The case for coding-aware routing," in *Proc. of IEEE INFOCOM 2007*. IEEE, 2007, pp. 1028–1036.

[338] M. Petrova, L. Wu, M. Wellens, and P. Mähönen, "Hop of no return: Practical limitations of wireless multi-hop networking," *Proc. of RealMAN 2005*, 2005.

[339] P. Owezarski, R. Hasan, G. Kremer, and P. Berthou, "First step in cross-layers measurement in wireless networks how to adapt to resource constraints for optimizing end-to-end services?" in *Proc. of WWIC 2011*, ser. Lecture Notes in Computer Science, X. Masip-Bruin, D. Verchere, V. Tsaoussidis, and M. Yannuzzi, Eds. Springer Berlin / Heidelberg, 2011.

[340] N. Ahmed, U. Ismail, S. Keshav, and K. Papagiannaki, "Poster abstract: measuring multi-parameter conflict graphs for 802.11 networks," *SIGMOBILE MCCR*, vol. 13, no. 3, pp. 54–57, Jan. 2010.

[341] M. Kao, *Encyclopedia of algorithms*. Springer-Verlag New York Inc, 2008.

[342] R. Gentleman, R. Ihaka, D. Bates, J. Chambers *et al.*, "The R project for statistical computing," www.r-project.org [last visited 13.03.2012].

[343] Y. Guédon, "Estimating hidden semi-Markov chains from discrete sequences," *Journal of Computational and Graphical Statistics*, vol. 12, no. 3, pp. 604–639, 2003.

[344] Y. Zhao, J. Gaeddert, K. Bae, and J. Reed, "Radio environment map enabled situation-aware cognitive radio learning algorithms," in *Software Defined Radio Forum (SDRF) technical conference*, 2006.

[345] M. Wellens, J. Riihijärvi, and P. Mähönen, "Evaluation of Adaptive MAC-Layer Sensing in Realistic Spectrum Occupancy Scenarios," in *Proc. of IEEE DySPAN 2010*. IEEE, 2010, pp. 1–12.

[346] S. Geirhofer, L. Tong, and B. Sadler, "Dynamic spectrum access in the time domain: Modeling and exploiting white space," *IEEE Communications Magazine*, vol. 45, no. 5, pp. 66–72, 2007.

[347] H. Kim and K. Shin, "Efficient discovery of spectrum opportunities with MAC-layer sensing in cognitive radio networks," *IEEE Transactions on Mobile Computing*, vol. 7, no. 5, pp. 533–545, 2008.

[348] R. Tandra, M. Mishra, and A. Sahai, "What is a spectrum hole and what does it take to recognize one?" *Proceedings of the IEEE*, vol. 97, no. 5, pp. 824–848, 2009.

[349] V. Chandrasekhar, J. Andrews, and A. Gatherer, "Femtocell networks: A survey," *IEEE Communications Magazine*, vol. 46, no. 9, pp. 59–67, September 2008.

[350] H. Tang, "Some physical layer issues of wide-band cognitive radio systems," in *Proc. of IEEE DySPAN 2005*, 2005, pp. 151–159.

[351] N. Shankar, C. Cordeiro, and K. Challapali, "Spectrum agile radios: Utilization and sensing architectures," in *Proc. of IEEE DySPAN 2005*. IEEE, 2005, pp. 160–169.

[352] T. Yucek and H. Arslan, "A survey of spectrum sensing algorithms for cognitive radio applications," *IEEE Communications Surveys & Tutorials*, vol. 11, no. 1, pp. 116–130, 2009.

[353] D. Cabric, S. Mishra, and R. Brodersen, "Implementation issues in spectrum sensing for cognitive radios," in *Proc. of the 2004 Asilomar Conference on Signals, Systems and Computers*, vol. 1, 2005, pp. 772–776.

[354] I. Akyildiz, W. Lee, M. Vuran, and S. Mohanty, "NeXt generation/dynamic spectrum access/cognitive radio wireless networks: a survey," *Computer Networks*, vol. 50, no. 13, pp. 2127–2159, 2006.

[355] H. Urkowitz, "Energy detection of unknown deterministic signals," *Proceedings of the IEEE*, vol. 55, no. 4, pp. 523–531, 2005.

[356] M. Wellens, J. Riihijärvi, and P. Mähönen, "Empirical time and frequency domain models of spectrum use," *Physical Communication*, vol. 2, no. 1, pp. 10–32, 2009.

[357] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[358] H. X. Nguyen and M. Roughan, "Rigorous statistical analysis of internet loss measurements," in *Proceedings of the ACM SIGMETRICS '10*, 2010, pp. 361–362.

[359] J. O'Connell and S. Højsgaard, *mhsmm: Parameter Estimation and Prediction for Hidden Markov and Semi-Markov models for Data with Multiple Observation Sequences*, R package v. 0.3.6, 2010, http://CRAN.R-project.org/package=mhsmm.

[360] R. Ihaka and R. Gentleman, "R: A language for data analysis and graphics," *Journal of computational and graphical statistics*, vol. 5, no. 3, pp. 299–314, 1996.

[361] *MAX 2829 datasheet*, MAXIM, http://datasheets.maxim-ic.com/en/ds/MAX2828-MAX2829.pdf, last visited: 11.06.2010.

[362] Texas Instruments Inc. CC2420 Radio Transceiver Datasheet, http://focus.ti.com/lit/ds/symlink/cc2420.pdf [Last Visisted 24.09.2012].

[363] J. Ansari, X. Zhang and P. Mähönen, "A Decentralized MAC for Opportunistic Spectrum Access in Cognitive Wireless Networks," in *Proceedings of the ACM CoRoNet 2010*, September 2010.

[364] [Online] ICT ARAGORN homepage, http://www.ict-aragorn.eu/ [Last visited: 25.05.2012].

[365] Y.-C. Liang, K.-C. Chen, G. Li, and P. Mähönen, "Cognitive radio networking and communications: An overview," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 7, pp. 3386–3407, September 2011.

[366] P. Mähönen, "Cognitive trends in making: future of networks," in *Proc. of IEEE PIMRC 2004*, vol. 2. IEEE, 2004, pp. 1449–1454.

[367] M. Petrova and P. Mähönen, "Cognitive resource manager: a cross-layer architecture for implementing cognitive radio networks," *Cognitive Wireless Networks, Springer*, 2007.

[368] M. Petrova, "Cognitive resource manager framework for optimal resource allocation," Ph.D. dissertation, 2011.

[369] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, 2003.

[370] J. Van den Berg, R. Litjens, A. Eisenblätter, M. Amirijoo, O. Linnell, C. Blondia, T. Kürner, N. Scully, J. Oszmianski, and L. Schmelz, "Self-organisation in future mobile communication networks," *Proceedings of ICT Mobile Summit*, vol. 2008, 2008.

[371] J. Zander and S. L. Kim, *Radio Resource Managment for Wireless Networks*. Norwood, MA: Artech House, 2001.

[372] S. Feng and E. Seidel, "Self-organizing networks (son) in 3gpp long term evolution," *Nomor Research GmbH, Munich, Germany*, 2008.

[373] M. Petrova, L. Wu, P. Mähönen, and J. Riihijärvi, "Interference measurements on performance degradation between colocated IEEE 802.11 g/n and IEEE 802.15.4 networks," in *Proc. of Sixth International Conference on Networking*, 2007, pp. 93–93.

[374] S. Fiehe, J. Riihijärvi, and P. Mähönen, "Experimental study on performance of IEEE 802.11n and impact of interferers on the 2.4 GHz ISM band," in *Proc. of the 6th International Wireless Communications and Mobile Computing Conference*, 2010, pp. 47–51.

[375] "Wireless RF Interference Customer Survey Results", Cisco, http://www.cisco.com/en/US/prod/collateral/wireless/ps5678/ps10981/white_paper_c11-609300.html [Last Visisted 24.09.2012].

[376] K. Jones and L. Liu, "What where wi: An analysis of millions of wi-fi access points," in *Proc. of the IEEE International Conference on Portable Information Devices*. IEEE, 2007, pp. 1–4.

[377] J. Mitola, "Cognitive radio architecture evolution: annals of telecommunications," *Annals of Telecommunications*, vol. 64, pp. 419–441, 2009.

[378] P. Pawelczak, K. Nolan, L. Doyle, S. Oh, and D. Cabric, "Cognitive radio: ten years of experimentation and development," *IEEE Communications Magazine*, vol. 49, no. 3, pp. 90–100, 2011.

[379] S. Dobson, S. Denazis, A. Fernández, D. Gaïti, E. Gelenbe, F. Massacci, P. Nixon, F. Saffre, N. Schmidt, and F. Zambonelli, "A survey of autonomic communications," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 1, no. 2, pp. 223–259, 2006.

[380] H. Derbel, N. Agoulmine, and M. Salaün, "Anema: Autonomic network management architecture to support self-configuration and self-optimization in ip networks," *Computer Networks*, vol. 53, no. 3, pp. 418–430, 2009.

[381] W. Berrayana, G. Pujolle, and H. Youssef, "XLEngine: a cross-layer autonomic architecture with network wide knowledge for QoS support in wireless networks," in *Proc. of the 2009 International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly*, 2009, pp. 170–175.

[382] S. Nowak, F. Schaefer, M. Brzozowski, R. Kraemer, and R. Kays, "Towards a convergent digital home network infrastructure," *IEEE Transactions on Consumer Electronics*, vol. 57, no. 4, pp. 1695–1703, 2011.

[383] E. Shihab, L. Cai, F. Wan, A. Gulliver, and N. Tin, "Wireless mesh networks for in-home IPTV distribution," *IEEE Network Magazine*, vol. 22, no. 1, pp. 52–57, 2008.

[384] K. Piamrat and P. Fontaine, "Coordinated architecture for wireless home networks," in *Proceedings of the 2nd ACM SIGCOMM workshop on Home networks*, 2011, pp. 49–54.

[385] J. Son, J. Park, K. Moon, and Y. Lee, "Resource-aware smart home management system by constructing resource relation graph," *IEEE Transactions on Consumer Electronics*, vol. 57, no. 3, pp. 1112–1119, 2011.

[386] R. Budde, N. Langhammer, C. Schilling, and R. Kays, "Dynamic frequency selection for next generation hierarchical wireless home area networks," in *Proc. of IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, 2011, pp. 1–6.

[387] N. Chowdhury and R. Boutaba, "A survey of network virtualization," *Computer Networks*, vol. 54, no. 5, pp. 862–876, 2010.

[388] K. Ram, J. Santos, Y. Turner, A. Cox, and S. Rixner, "Achieving 10 Gb/s using safe and transparent network interface virtualization," in *Proceedings of the ACM SIGPLAN/SIGOPS international conference on virtual execution environments*, 2009, pp. 61–70.

[389] J. Gerhart and M. Kirschner, "The theory of facilitated variation," *Proceedings of the National Academy of Sciences*, vol. 104, no. Suppl 1, p. 8582, 2007.

[390] J. Doyle and M. Csete, "Architecture, constraints, and behavior," *Proceedings of the National Academy of Sciences*, vol. 108, no. Supplement 3, pp. 15 624–15 630, 2011.

[391] J. Riihijärvi, M. Wellens, and P. Mähönen, "Link-layer abstractions for utility-based optimization in cognitive wireless networks," in *Proc. of 1st International Conference on Cognitive Radio Oriented Wireless Networks and Communications*, 2006, pp. 1–6.

[392] J. Nasreddine *et al.*, "An Implementation of Cognitive Resource Management on LTE Platform," in *Proc. of IEEE PIMRC*, September 2010.

[393] D. Clark, C. Partridge, J. Ramming, and J. Wroclawski, "A knowledge plane for the Internet," in *Proc. of ACM SIGCOMM 2003*. ACM, 2003, pp. 3–10.

[394] H. Nwana, "Software agents: An overview," *The Knowledge Engineering Review*, vol. 11, no. 03, pp. 205–244, 1996.

[395] H. Gintis, *Game theory evolving: A problem-centered introduction to modeling strategic behavior*. Princeton University Press, 2000.

[396] M. Wooldridge, *An introduction to multiagent systems*. Wiley, 2002.

[397] H. Tianfield, "Multi-agent based autonomic architecture for network management," in *Proc. of IEEE International Conference on Industrial Informatics*, 2003, pp. 462–469.

[398] C. Shen, D. Pesch, and J. Irvine, "A framework for self-management of hybrid wireless networks using autonomic computing principles," in *Proceedings of the 3rd Annual Communication Networks and Services Research Conference*, 2005, pp. 261–266.

[399] A. MacKenzie, J. Reed, P. Athanas, C. Bostian, R. Buehrer, L. DaSilva, S. Ellingson, Y. Hou, M. Hsiao, J. Park *et al.*, "Cognitive radio and networking research at virginia tech," *Proceedings of the IEEE*, vol. 97, no. 4, pp. 660–688, 2009.

[400] D. Cook, M. Youngblood, E. Heierman III, K. Gopalratnam, S. Rao, A. Litvin, and F. Khawaja, "MavHome: An agent-based smart home," in *Proc. of IEEE PerCom 2003*, 2003, pp. 521–524.

[401] C. Schlenoff, J. Albus, E. Messina, A. Barbera, R. Madhavan, and S. Balakirsky, "Using 4D/RCS to address AI knowledge integration," *AI Magazine*, vol. 27, no. 2, p. 71, 2006.

[402] J. Riihijärvi *et al.*, "Extending Policy Languages with Utility and Prioritization Knowledge: The CAPRI approach," in *Proc. of IEEE DySPAN 2010*, Singapore, April 2010.

[403] R. Braden, D. Clark, and S. Shenker, "RFC 1633: Integrated services in the Internet architecture: an overview," June, 1994.

[404] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "RFC 2475: An architecture for differentiated services," 1998.

[405] [Online] MetaGeek Wi-Spy 2.4 GHz spectrum homepage, http://www.metageek.net/ [Last visited: 6.06.2012].

[406] "The USRP Board," https://radioware.nd.edu/documentation/hardware/the-usrp-board [Last visited: 1st Nov, 2010].

[407] D. Denkovski, V. Pavlovska, V. Atanasovski, and L. Gavrilovska, "Novel Policy Reasoning Architecture for Cognitive Radio Environments," in *Proc. of IEEE GLOBECOM 2010*, Miami, USA, 2010.

[408] E. Meshkova, J. Riihijärvi, M. Petrova, and P. Mähönen, "A survey on resource discovery mechanisms, peer-to-peer and service discovery frameworks," *Computer networks*, vol. 52, no. 11, pp. 2097–2128, 2008.

[409] X. Zhang, J. Ansari, G. Yang, and P. Mahonen, "TRUMP: Supporting efficient realization of protocols for cognitive radio networks," in *Proc. of IEEE DySPAN 2011*, 2011, pp. 476–487.

[410] [Online] WAG511 802.11a/b/g Dual Band PC Card, http://support.netgear.com/product/WAG511 [Last visited: 6.06.2012].

[411] R. Chandra and P. Bahl, "MultiNet: Connecting to multiple IEEE 802.11 networks using a single wireless card," in *Proc. of IEEE INFOCOM 2004*, vol. 2, 2004, pp. 882–893.

[412] R. Chandra, R. Mahajan, T. Moscibroda, R. Raghavendra, and P. Bahl, "A case for adapting channel width in wireless networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 4, pp. 135–146, 2008.

[413] V. VideoLan, "Media player," http://www.videolan.org/vlc/streaming.html [last visited 21.03.2012].

[414] "WiFiCopper 802.11 packet injector," http://www.nutsaboutnets.com/performance-wifi/products/product-wificopper-packet-injector.htm, [last visited 19.02.2012].

[415] M. Shin, A. Mishra, and W. Arbaugh, "Improving the latency of 802.11 hand-offs using neighbor graphs," in *Proc. of the 2nd international conference on Mobile systems, applications, and services*, 2004, pp. 70–83.

[416] S. Grünewälder, J. Audibert, M. Opper, and J. Shawe-Taylor, "Regret bounds for gaussian process bandit problems," in *Proc. of the 14th International Conference on Artificial Intelligence and Statistics*, 2010.

[417] U. Kruse, "Net neutrality regulation of the internet?" *International Journal of Management and Network Economics*, vol. 2, no. 1, pp. 3–23, 2011.

[418] M. Yuksel, K. Ramakrishnan, S. Kalyanaraman, J. Houle, and R. Sadhvani, "Class-of-service in IP backbones: informing the network neutrality debate," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 36, no. 1. ACM, 2008, pp. 435–436.

[419] M. Chiang, P. Hande, H. Kim, S. Ha, and R. Calderbank, "Pricing broadband: Survey and open problems," in *Proc. of IEEE ICUFN 2010*. IEEE, 2010, pp. 303–308.

[420] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: rapid prototyping for software-defined networks," in *Proceedings of the Ninth ACM SIGCOMM Workshop on Hot Topics in Networks*. ACM, 2010, p. 19.

[421] V. Fineberg, "A practical architecture for implementing end-to-end QoS in an IP network," *IEEE Communications Magazine*, p. 122, 2002.

[422] R. Hancock, G. Karagiannis, J. Loughney, and S. Van den Bosch, "Next steps in signaling (NSIS): Framework," IETF, Tech. Rep., 2005.

[423] M. Zukerman *et al.*, "To be fair or efficient or a bit of both," *Computers and Operations Research*, vol. 35, no. 12, pp. 3787–3806, 2008.

[424] P. van Laarhoven and E. Aarts, "Simulated annealing: theory and applications," *Mathematics and Its Applications*, 1987.

[425] J. Lavaei, J. Doyle, and S. Low, "Utility Functionals Associated With Available Congestion Control Algorithms," in *Proc. of IEEE INFOCOM 2010*, San Diego, USA, March 2010.

[426] D. Wei, C. Jin, S. Low, and S. Hegde, "FAST TCP: motivation, architecture, algorithms, performance," *IEEE/ACM Transactions on Networking (ToN)*, vol. 14, no. 6, pp. 1246–1259, 2006.

[427] B. Briscoe, "Flow rate fairness: Dismantling a religion," *ACM SIGCOMM CCR*, vol. 37, no. 2, pp. 63–74, 2007.

[428] M. Mu, A. Mauthe, and F. Garcia, "A Utility-Based QoS Model for Emerging Multimedia Applications," in *in Proc. of NGMAST*, 2008, pp. 521–528.

[429] S. Agarwal and J. Lorch, "Matchmaking for online games and other latency-sensitive P2P systems," *ACM SIGCOMM CCR*, vol. 39, no. 4, pp. 315–326, 2009.

[430] S. Vegesna, *IP Quality of Service:the Complete Resource for Understanding and Deploying IP Quality of Service for Cisco Networks*. Cisco Press, 2005.

[431] J. De Oliveira and W. Pedrycz, *Advances in fuzzy clustering and its applications*. John Wiley and Sons, 2007.

[432] A. Greenberg *et al.*, "The cost of a cloud: research problems in data center networks," *ACM SIGCOMM CCR*, vol. 39, no. 1, pp. 68–73, 2008.

[433] A. Qureshi *et al.*, "Cutting the electric bill for internet-scale systems," *ACM SIGCOMM CCR*, vol. 39, no. 4, pp. 123–134, 2009.

[434] R. Van der Mei, R. Hariharan, and P. Reeser, "Web server performance modeling," *Telecommunication Systems*, vol. 16, no. 3, pp. 361–378, 2001.

[435] T. Briscoe, "DNS support for load balancing," RFC 1794, Tech. Rep., April 1995.

[436] W. Edwards and R. Grinter, "At Home with Ubiquitous Computing: Seven Challenges," in *Proc. of UBICOMM'01*, Atlanta, Georgia, USA, September 2001.

[437] T. Gu, X. Wang, H. Pung, and D. Zhang, "An Ontology-based Context Model in Intelligent Environments," in *Proc. of CNDS'04*, San Diego, California, USA, January 2004.

[438] J. Kalaoja and et al., "The Vocabulary Ontology Engineering for the semantic modelling of home services," in *Proc. of ICEIS'06*, Paphos, Cyprus, May 2006.

[439] H. Chen, F. Perich, T. Finin, and A. Joshi, "SOUPA: standard ontology for ubiquitous and pervasive applications," in *Proc. of MOBIQUITOUS 2004*, Boston, Massachusetts, USA, August 2004.

[440] S. Peters and H. Shrobe, "Using semantic networks for knowledge representation in an intelligent environment," in *Proc. of IEEE PerCom'03*, Ft. Worth, TX, USA, March 2003.

[441] M. Merabti, P. Fergus, O. Abuelma'atti, H. Yu, and C. Judice, "Managing distributed networked appliances in home networks," *Proceedings of the IEEE*, vol. 96, no. 1, pp. 166–185, 2008.

[442] D. Bonino and F. Corno, "Dogont-ontology modeling for intelligent domotic environments," *The Semantic Web-ISWC 2008*, pp. 790–803, 2008.

[443] D. McGuinness, F. Van Harmelen *et al.*, "OWL web ontology language overview," *W3C recommendation*, vol. 10, pp. 2003–04, 2004.

[444] *Protégé Framework*, Stanford University, USA, http://protege.stanford.edu/overview/index.html, last visited: 11.12.2011.

[445] D. Martin and *et al.*, "OWL-S: Semantic markup for web services," *W3C Member Submission*, vol. 22, pp. 2004–07, 2004.

[446] E. Meshkova, J. Riihijärvi, J. Ansari, and P. Mähönen, "Indoor coverage estimation from unreliable measurements using spatial statistics," in *Proceedings of the 16th ACM international conference on Modeling, analysis & simulation of wireless and mobile systems*. ACM, 2013, pp. 211–218.

[447] Y. Zhao, B. Le, and J. H. Reed, *Cognitive Radio Technology*. Elsevier, 2006, ch. Network Support: The Radio Environment Map, pp. 337–363.

[448] R. J. Bultitude, "Measurement, characterization and modeling of indoor 800/900 mhz radio channels for digital communications," *IEEE Communications Magazine*, vol. 25, no. 6, pp. 5 – 12, jun 1987.

[449] H. Hashemi, M. McGuire, T. Vlasschaert, and D. Tholl, "Measurements and modeling of temporal variations of the indoor radio propagation channel," *IEEE Transactions on Vehicular Technology*, vol. 43, no. 3, pp. 733 –737, aug 1994.

[450] I. Kashiwagi, T. Taga, and T. Imai, "Time-varying path-shadowing model for indoor populated environments," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 1, pp. 16 –28, January 2010.

[451] H. Hashemi, "The indoor radio propagation channel," *Proceedings of the IEEE*, vol. 81, no. 7, pp. 943 –968, July 1993.

[452] M. F. Hanif, P. J. Smith, and M. Shafi, "Performance of cognitive radio systems with imperfect radio environment map information," in *Proc. of the Australian Communications Theory Workshop (AusCTW 2009)*, February 2009, pp. 61 –66.

[453] K. Chowdhury and T. Melodia, "Platforms and testbeds for experimental evaluation of cognitive ad hoc networks," *IEEE Communications Magazine*, vol. 48, no. 9, pp. 96–104, 2010.

[454] D. Mills, J. Martin, J. Burbank, and W. Kasch, "RFC 5905: Network time protocol version 4: Protocol and algorithms specification," *IETF*, 2010.

[455] D. Denkovski, V. Atanasovski, and L. Gavrilovska, "Efficient mid-end spectrum sensing implementation for cognitive radio applications based on USRP2 devices," in *Proc. of COCORA 2011*, April 2011.

[456] J. Ansari, T. Ang, and P. Mähönen, "WiSpot - detecting Wi-Fi networks using IEEE 802.15.4 radios," in *Proc. of EWSN 2011*, Feb. 2011.

[457] H. Celebi, I. Guvenc, S. Gezici, and H. Arslan, "Cognitive-Radio Systems for Spectrum, Location, and Environmental Awareness," *IEEE Antennas and Propagation Magazine*, vol. 52, no. 4, pp. 41–61, 2010.

[458] E. Villegas, E. Lopez-Aguilera, R. Vidal, and J. Paradells, "Effect of adjacent-channel interference in ieee 802.11 wlans," in *Proc. of CrownCom 2007*. IEEE, 2007, pp. 118–125.

[459] K. Kosek, S. Szott, M. Natkaniec, and K. Łoziak, "An experimental analysis of IEEE 802.11 a/b/g cards in ad-hoc mode," in *Proc. of IEEE WRENCOM 2007*, 2007.

Curriculum Vitae

Personal Information

Name	Elena Meshkova
Date of birth	23rd February 1981
Place of birth	Moscow, Russia
Nationality	Russian

Education

March 2006 – January 2013	RWTH Aachen University Institute for Networked Systems Ph.D. studies	Aachen, Germany
October 2003 – December 2005	RWTH Aachen University Master of Science (M.Sc.) Major in Communications Engineering	Aachen, Germany
September 1999 – October 2003	Bauman Moscow State Technical University Bachelor of Science (B.Sc.)	Moscow, Russia

Professional Experience

March 2006 – September 2013	RWTH Aachen University Institute for Networked Systems Research assistant	Aachen, Germany
October 2004 – March 2005	Siemens Research trainee	Bocholt, Germany

Selected Publications

Journal papers

E. Meshkova, J. Riihijärvi, M. Petrova and P. Mähönen: *A Survey on Resource Discovery Mechanisms, Peer-to-Peer and Service Discovery Frameworks*, Elsevier Computer Networks Journal, vol. 52, pp. 2097–2128, August 2008.

Conference papers

E. Meshkova, J. Riihijärvi, P. Mähönen, *Extending Graph-based Models of Wireless Network Structure with Dynamics*, Proc. of ACM MSWiM'12, Paphos, Cyprus Island, October 2012.

J. Ansari, **E. Meshkova**, W. Masood, A.Muslim, J. Riihijärvi, P. Mähönen, *CONFab: Component based Optimization of WSN Protocol Stacks using Deployment Feedback*, Proc. of the ACM MobiWac in conjunction with ACM MSWiM'12, Paphos, Cyprus Island, October 2012.

J. Riihijärvi, **E. Meshkova**, P. Mähönen, *Graph Approximations of Spatial Wireless Network Models*, Proc. of the 6th ACM PM2HW2N Workshop, in conjunction with ACM MSWiM'11, Miami Beach, FL, November 2011.

E. Meshkova, J. Ansari, J. Riihijärvi, J. Nasreddine, P. Mähönen, *Estimating Transmitter Activity Patterns: an Empirical Study in the Indoor Environment*, Proc. of IEEE PIMRC 2011, Toronto, Canada, September 2011.

E. Meshkova, J. Ansari, D. Denkovski, J. Riihijärvi, J. Nasreddine, M. Pavloski, L. Gavrilovska, P. Mähönen: *Experimental Spectrum Sensor Testbed for Constructing Indoor Radio Environmental Maps*, Poster paper in IEEE DySPAN 2011, Aachen, Germany, May 2011.

E. Meshkova, Z. Wang, J. Nasreddine, D. Denkovski, C. Zhao, K. Rerkrai, T. Farnham, A. Ahmad, A. Gefflaut, L. Gavrilovska, and P. Mähönen: *Using Cognitive Radio Principles for Wireless Resource Management in Home Networking*, Proc. of IEEE CCNC 2011, Las Vegas, USA, January 2011.

E. Meshkova, J. Riihijärvi, A. Achtzehn, P. Mähönen: *On Utility-Based Network Management*, Proc. of IEEE International Workshop on Management of Emerging Networks and Services (IEEE MENS 2010), in conjunction with IEEE Globecom 2010, Miami, Florida, December 2010.

E. Meshkova, J. Riihijärvi, A. Achtzehn, P. Mähönen: *Exploring Simulated Annealing and Graphical Models for Optimization in Cognitive Wireless Networks*, Proc. of IEEE Globecom 2009, Honolulu, Hawaii, December 2009.

E. Meshkova, A. Achtzehn, J. Riihijärvi and P. Mähönen: *Cross-layer optimization of edge networks*, Demonstration abstract, SIGCOMM 2009, Barcelona, Spain, August 2009.

E. Meshkova, J. Riihijärvi and P. Mähönen: *Learning in Cross-layer Wireless Network Optimization*, Learning for Networking Workshop, in conjunction with SIGMETRICS/Performance 2009, Seattle, USA, June 2009.

E. Meshkova, J. Riihijärvi, J. Ansari, K. Rerkrai and P. Mähönen: *An Extendible Meta-Data Specification for Component-Oriented Networks with Applications to WSN Configuration and Optimization*, Proc. of IEEE PIMRC 2008, Cannes, France, September 2008.

E. Meshkova, K. Rerkrai, J. Riihijärvi and P. Mähönen: *Optimizing Component-Oriented Systems: A Case Study in Wireless Sensor Networks*, Proc. of ACM SIGCOMM 2008, Demo program abstract, Seattle, USA, August 2008.

E. Meshkova, J. Riihijärvi, P. Mähönen and C. Kavadias: *Modeling the Home Environment using Ontology with Applications in Software Configuration Management*, Proc. of 15th International Conference on Telecommunications (ICT), St. Petersburg, Russia, June 2008.

E. Meshkova, J. Riihijärvi, F. Oldewurtel, C. Jardak and P. Mähönen: *Service-Oriented Design Methodology for Wireless Sensor Networks: A View through Case Studies*, Proceedings of IEEE SUTC 2008, Taichung, Taiwan, June 2008.