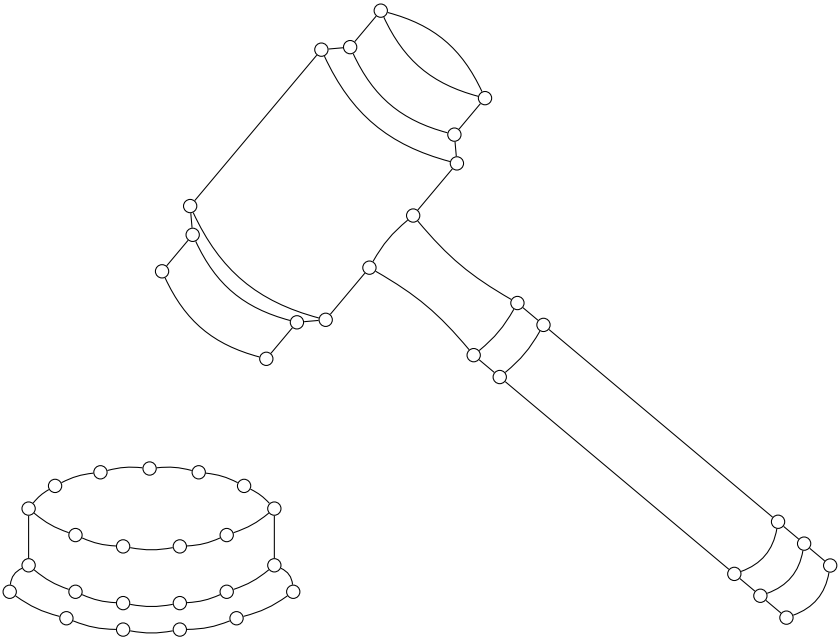
Matroid Optimization in Auction Theory

Niklas Rieken



Matroid Optimization in Auction Theory

Von der Fakultät für Wirtschaftswissenschaften der
Rheinisch-Westfälischen Technischen Hochschule Aachen
zur Erlangung des akademischen Grades eines Doktors der
Wirtschafts- und Sozialwissenschaften genehmigte Dissertation

vorgelegt von

Niklas Rieken, M. Sc.

1. Berichterin	Univ.-Prof. Dr. rer. nat. Britta Peis
2. Berichter	Univ.-Prof. Dr. rer. pol. Thomas Kittsteiner

Tag der mündlichen Prüfung: 11. November 2025

Articles included in this thesis

K. Eickhoff, S.T. McCormick, B. Peis, N. Rieken, L. Vargas Koch (2024). *A flow-based ascending auction to compute buyer-optimal Walrasian prices*. In: *Networks* 84.2, pp. 161–180, eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/net.22218

K. Eickhoff, M. Neuwohner, B. Peis, N. Rieken, L. Vargas Koch, L.A. Végh (2025). *Faster Dynamic Auctions via Polymatroid Sum*. In: *ACM Transactions on Economics and Computation* 13.3, pp. 13.1–13.47, eprint: https://dl.acm.org/doi/pdf/10.1145/3729429

B, Peis, N. Rieken (2025). *A Simplified Analysis of the Ascending Auction to Sell a Matroid Base*. In: *Operations Research Letters* 64, pp. 107381, eprint: https://doi.org/10.1016/j.orl.2025.107381

W. Gálvez, B. Peis, N. Rieken, V. Verdugo, J. Verschae (working paper). *The Greedy+ Algorithm for Submodular Cover*.

Abstract

This thesis is mainly concerned with the theory of dynamic multi-item auctions, i.e., a processes in which over time there is communication between buyers and an auctioneer, who adjusts prices for the items based on that communication. There is a vast number of intriguing questions on those processes, both from an economic and a computational point of view.

A Walrasian equilibrium—characterized by a price vector and an allocation of goods such that all agents maximize utility and the market clears—represents a central concept in general equilibrium theory. It is well-known that Walrasian equilibria are guaranteed to exists if all buyers have gross substitute valuation functions. Finding these equilibria by means of natural dynamic auctions (or tâtonnement) poses an interesting computational problem—finding excess demand sets to increase prices on—that previously involved submodular function minimization, which is a highly non-trivial process. We show that this process can be simplified in cognitive complexity and running time. This can either be achieved by very easy maximum flow computations if the valuation functions are simple enough or by solving a polymatroid sum problem if we want to handle arbitrary gross substitute valuations. In both cases the corresponding dual solution reveals the desired excess demand set. We also prove some structural properties about Walrasian prices (which form a lattice) and their relation to weaker notions of market equilibria. Our results bridge theoretical economics and algorithmic mechanism design, offering both a novel computational perspective and practical auction protocols to find equilibria.

For a related ascending auction, to sell a base of a matroid, we show that its analysis can be simplified by using only a few folklore lemmas from matroid theory. This auction has the neat property that acting truthfully as a buyer is an equilibrium strategy—a feature that is typically not present in dynamic multi-item auctions with buyers having valuations that are not unit-demand—moreover, the auctions also output (under truthful signals) the utilitarian optimum, i.e., a matroid base of maximum weight. We also give additional remarks regarding communication cost and privacy matters of this auction.

Finally, we study a natural extension of a greedy algorithm for submodular covering problems, which allows to remove previously chosen elements if they are recognized to be redundant. We are able to show that a for this greedy algorithm natural subclass of submodular functions exhibits neat properties that can be seen as generalizations

for properties of matroid rank functions. For some of these functions, we can show that the algorithm is guaranteed to compute an optimal solution to the submodular covering problem given any linear cost function.

Deutsche Zusammenfassung

Diese Arbeit befasst sich vor allem mit der Theorie iterativer Auktionen in denen mehrere Güter gleichzeitig zum Verkauf stehen. D.h. wir betrachten Prozesse, bei denen über die Zeit Kommunikation zwischen Käuferinnen und einem Auktionator stattfindet, welcher die Preise der Güter basierend auf dieser Kommunikation anpasst. Es gibt eine Vielzahl spannender Fragen zu diesen Prozessen, sowohl aus ökonomischer als auch aus mathematisch-informatischer Sicht.

Walras-Gleichgewichte – gegeben durch Preisvektoren und Verteilungen der zum Verkauf stehenden Güter mit der Eigenschaft, dass alle Käuferinnen eine nutzenmaximale Menge von Gütern erhalten und dass der Markt geräumt wird – sind ein zentrales Konzept der allgemeinen Gleichgewichtstheorie. Es ist bekannt, dass Walras-Gleichgewichte garantiert existieren, wenn die Bewertungsfunktionen der Käuferinnen die Güter als Bruttosubstitute (*gross substitutes*) behandeln. Die Ermittlung dieser Gleichgewichte mittels natürlicher dynamischer Auktionen (oder Tâtonnement) wirft ein interessantes, kombinatorisches Problem auf, nämlich die Suche nach Gütern mit Nachfrageüberschuss auf denen der Preis erhöht werden soll. Bisher wurde dieses Problem mithilfe von Minimierung submodularer Funktionen gelöst, was ein sehr involvierter Prozess ist. Wir zeigen, dass dieser Problem mit einfacheren Mitteln, aber auch schneller gelöst werden kann. Dies kann entweder durch sehr einfache Maximalflussberechnungen erreicht werden, wenn die Bewertungsfunktionen einfach genug sind, oder durch die Lösung eines Polymatroid-Summenproblems, wenn wir beliebige Bruttosubstitutsbewertungen behandeln wollen. In beiden Fällen ergibt die zugehörige duale Lösung die gewünschte Menge an Gütern mit Nachfrageüberschuss. Wir beweisen außerdem einige strukturelle Eigenschaften von Walras-Preisen (die einen Verband bilden) und deren Beziehung zu schwächeren Gleichgewichtskonzepten. Unsere Ergebnisse schlagen eine Brücke zwischen theoretischen Wirtschaftswissenschaften und algorithmischem Auktionsdesign und bieten sowohl eine neuartige informatische Perspektive als auch praktische Auktionsprotokolle zur Gleichgewichtsfindung.

Für eine verwandte aufsteigende Auktion zum Verkauf einer Basis eines Matroids zeigen wir, dass ihre Analyse durch die Verwendung nur weniger Lemmata aus der Matroidentheorie vereinfacht werden kann. Diese Auktion hat die interessante Eigenschaft, dass wahrheitsgemäße Kommunikation als Käuferin eine Gleichgewichtsstrategie darstellt – typischerweise ist bei dynamischen Auktionen mit mehreren Artikeln dies nicht gegeben, es sei denn, alle Käuferinnen sind an maximal einem Gut interessiert. Darüber

hinaus liefert die Auktion (vermöge wahrheitsgemäßer Gebote) das utilitaristische Optimum, d.h. eine Matroid-Basis mit maximalem Gewicht. Wir geben außerdem zusätzliche Diskussionen zu den Kommunikationskosten und Datenschutzaspekten dieser Auktion.

Abschließend untersuchen wir eine natürliche Erweiterung eines Greedy-Algorithmus für submodulare Überdeckungsprobleme, die es ermöglicht, zuvor ausgewählte Elemente zu entfernen, wenn diese als redundant erkannt werden. Wir können zeigen, dass eine für diesen Greedy-Algorithmus natürliche Unterklasse submodularer Funktionen interessante Eigenschaften aufweist, die als Verallgemeinerungen für Eigenschaften von Matroid-Rangfunktionen angesehen werden können. Für einige dieser Funktionen können wir zeigen, dass der Algorithmus, vermöge einer linearen Kostenfunktion, garantiert eine optimale Lösung für das submodulare Überdeckungsproblem berechnet.

> *If I have seen further it is by standing on the shoulders of Giants.*
>
> — **Isaac Newton**
> (in a letter to Robert Hooke)

Acknowledgements

It has been an honor and a privilege to be a part of the research that made this thesis happen.

I owe a debt of gratitude to the Chair of Management Science at RWTH Aachen University with all its current team members and alumni that I had the pleasure working with. These are Britta, Dina, Marc, Veerle, Vipin, Björn, Laura, Oliver, Kathi, Eran, Komal, Lennart, Andreas, Khai Van, Philipp, Laura, and also Niklas′ from the adjacent chair. I am very happy to say that I also can call them friends and not just colleagues.

Thanks to the members of my PhD commitee, Britta Peis, Thomas Kittsteiner, Marco Lübbecke, and Michael Schneider, who took the time to read, comprehend, and evaluate this thesis.

A big thank you goes of course to my co-authors Britta, José, Victor, Waldo, Andreas, Kathi, Laura, Laci, Meike, Tom, Ike, Constantin, Alexander, and Klaus.

As I was also heavily involved with teaching at the chair, a task that I enjoyed a lot, I would like to thank Kevin, Larissa, Luca, Ivan, Amelie, Alex, Lennart, David, Liam, and Paula. Your dedicated work as student assistants made the job much easier. I got lucky enough to be also awarded the teaching award of the School of Business and Economics at RWTH Aachen University in 2024 for which I am very thankful to the students and in particular the student council of the Faculty 8 at RWTH Aachen University. The teaching task also comes with the supervison of bachelor's and master's theses and the students that I got to advise always made it either easy or interesting for me. So another thank you towards Luisa, Nikhil, Larissa, Max, Ivan, Kevin, Devashish, Tim, Luca, Jakob, and Zhiyuan.

A PhD thesis is a massive body of work and I am very grateful that Kathi, Komal, Laura, and Philipp agreed to proofread it and gave me valuable suggestions. Naturally, the remaining errors are my own.

The quote on top of this page by Sir Isaac Newton refers to all the scientists in history that laid out the solid and strong foundation on which the research that I did with my

colleagues is built upon. However, I clearly had not just the shoulders of giants to stand on but also the shoulders of my friends to lean on. I always had Helen, Ronja, Felix, Janine, Julia, Pauli, Kim, Jule, and Pia to laugh with me or pick me up. Uplifting tunes in good and bad times were always provided by my favorite musicians Frank Turner and Jay "Beans on Toast" McAllister.

I would also like to thank José for multiple invitations to the beautiful country of Chile and Matías for his hospitality there.

To all relatives and friends who extended a hand in any way—whether through your presence, kind words, or a bit of help—thank you!

Finally, another word of thanks to my wonderful partner, Komal. Your confidence in me gives me strength, inspiration, and motivation. It seems that whatever I do and wherever I go, I can always count on your support—and I promise you that you can always count on mine.

Contents

0 Introduction 1

0.1 Publications . . . 4

0.2 Thesis Structure . . . 5

1 Preliminaries 7

1.1 Basic Maths and Notation . . . 7

1.2 Submodular Functions . . . 8

1.3 Matroid Theory . . . 10

1.3.1 Operations on Matroids . . . 13

1.3.2 A Zoo of Matroids . . . 16

1.3.3 Matroid Cryptomorphisms . . . 17

1.3.4 Polymatroids . . . 19

1.4 Game Theory . . . 20

1.4.1 Strategic Games . . . 20

1.4.2 Extensive Form Games . . . 23

1.4.3 Bayesian Games . . . 24

1.5 Auction Theory . . . 26

2 Selling a Matroid Base via an Ascending Auction 33

2.1 The Economic Model . . . 33

2.1.1 Contribution . . . 36

2.1.2 Related Work . . . 36

2.2 Background Theory . . . 37

2.3 The Auction . . . 39

2.4 Analysis . . . 43

2.5 No Dominant Strategy . . . 48

2.6 Truthful Equilibrium . . . 51

2.7 Communication Requirements and Privacy . . . 54
2.7.1 Communication . . . 54
2.7.2 Privacy . . . 56

3 Ascending Auctions via Maximum Flow 59
3.1 The Economic Model . . . 61
3.1.1 Differences to Chapter 2 . . . 63
3.1.2 Contribution . . . 64
3.2 Background Theory . . . 65
3.3 The Auction . . . 69
3.3.1 Demand Correspondences and Oracles . . . 70
3.3.2 Excess Demand and Excess Supply . . . 74
3.3.3 A Flow Network to Check for Excess Demand . . . 75
3.3.4 Implementing the Auction . . . 81
3.4 Analysis . . . 81
3.4.1 Correctness . . . 82
3.4.2 Running Time . . . 87
3.5 Optimality and Incentives . . . 95

4 Faster Dynamic Auctions via Polymatroid Sum 99
4.1 The Economic Model . . . 100
4.1.1 A Landscape of Valuation Functions . . . 101
4.1.2 Differences to Chapter 3 . . . 104
4.1.3 Contribution . . . 104
4.1.4 Related Work . . . 106
4.2 Background Theory . . . 110
4.3 Demand Correspondences and Oracles . . . 115
4.4 Solving Matroid Union and Polymatroid Sum . . . 120
4.4.1 Knuth's Algorithm for Matroid Union . . . 121
4.4.2 A Generic Push-Relabel Algorithm for M-convex Set Packing 124
4.4.3 An Efficient Implementation for Matroid Union . . . 129
4.4.4 An Efficient Implementation for Polymatroid Sum . . . 130
4.5 Excess Demand and Supply with Discrete Convexity . . . 136
4.6 The Auctions . . . 144
4.7 Extremal Equilibrium Prices . . . 147
4.7.1 Minimal Packing Prices are Walrasian . . . 150

4.7.2 Maximal Covering Prices are Walrasian . . . 151
4.8 Monotone Comparative Statics . . . 152

5 The Greedy+ Algorithm for Submodular Cover 155
5.1 Introduction . . . 156
5.1.1 Contribution . . . 158
5.1.2 Related Work . . . 158
5.2 Background Theory . . . 160
5.3 Generalized Ranks . . . 162
5.3.1 Examples . . . 162
5.3.2 Basic Properties . . . 165
5.4 One-to-Many Exchange Properties for z-Bases . . . 169
5.5 The Greedy+ Algorithm . . . 173
5.5.1 A Sufficient Optimality Condition . . . 177
5.5.2 Nice Submodular Cover Problems . . . 181
5.6 Towards a Duality Theory for Generalized Ranks . . . 183
5.6.1 The Dual of a Generalized Rank . . . 184
5.6.2 Reflexivity . . . 185
5.7 Efficient Implementations of the Removal Oracle . . . 189
5.7.1 Laminar Set Cover . . . 189
5.7.2 Power-of-Two Knapsack Cover . . . 190

6 Conclusion 197
6.1 Review . . . 197
6.2 Open Problems . . . 198

References 201

CHAPTER 0

Introduction

Mathematical optimization and computer science is omnipresent in virtually every aspect of modern life. The study of economics is no exception to this statement and in fact, there is a variety of applications, theorems, and names linking these fields. Among various branches of mathematical optimization, *combinatorial optimization* is especially relevant in economic scenarios where decision variables are *discrete*—often involving choices such as "yes or no", or the selection from a finite (but very large) set of alternatives. Typical problems include scheduling, network design, and resource allocation. In economics, these types of problems emerge in areas such as market design, logistics, auction theory, and production planning. For instance, determining how to allocate a set of workers to tasks, or how to distribute goods through a supply chain efficiently, are economic problems that can be modeled into a combinatorial optimization problem.

This thesis is concerned with *combinatorial optimization* (in particular results from *matroid theory*) and its use in economics and especially *dynamic auctions*. We consider an economic setting in which multiple agents are interested in acquiring a discrete set of goods. This is referred to as a *pure exchange economy* as there is a fixed set of items to distribute among agents and no additional production. There are different aspects that are of interest:

The fact that there are multipe agents that want to divide goods among themselves immediately yields what is known as a *partitioning problem*, i.e., we need to divide the

set of goods into subsets that 1. do not overlap, 2. distribute all goods, and 3. give every agent a subset that they consider acceptable. Very often those kind of problems are computationally hard to solve but if the notion of *acceptable subsets* is somewhat good-natured, then we will see that we can indeed develop beautiful (and fast) algorithms. Concretely, in our economic setting we assume agents to have *gross substitute preferences*, which in simple terms means that if some agent already owns some good, then she has less demand for other goods. We can then show that an *equilibrium*, a distribtion of goods along with prices, always exists and it can be found via an elegant auction algorithm.

Another factor is that every agent wants to maximize her own utility but is not concerned with the utility of the others (or the system as a whole). Once agents agreed to a procedure (for instance, an auction) to distribute the goods according to their preferences, we can not a priori exclude the possibility an agent *games* the system, i.e., instead of reporting her preferences truthfully, she does so strategically to achieve a better outcome for herself. There is a huge body of work on *mechanism design* that shows how to design auctions that have truthful reporting as dominant strategy or at least as an equilibrium strategy. However, these auctions (typically *sealed-bid auctions*) are often not practicable in real life; they also work quite differently from what people think of when they refer to an auction (on eBay, at Christie's, or the bicycle auctions conducted by the police in Aachen). In this thesis, we explore auctions that are closer to those real life auctions in the sense that they are *dynamic*, meaning that there are multiple rounds of communication such that preferences are only revealed incrementally. While incentive guarantees get partially lost (inevitably), they would be quite easy to implement in practice.

So far the discussion was mainly concerned with the mechanism's (or auctioneer's) side of our exchange economy, i.e., finding allocation, prices, and making sure that agents cannot act strategically. However, the agents also have an optimization problem to solve. In order to participate in an auction, they have to figure out what to report to the mechanism. Their preferences, while implicitly existent, might not always be explicitly available to them and instead being somewhat hidden in a combinatorial optimization problem that they first have to solve themselves. More precisely, when the auctioneer asks an agent what set of goods she would want at some given prices, then the agent might first have to compute this preferred set. This step is usually omitted in the analysis of auctions since it does not reveal anything about the auction itself. Yet, the problem itself is interesting on its own and we consider a similar problem at the

end of this thesis in which we explore a *covering problem* that under some conditions can be solved (optimally and efficiently) by an algorithm that is essentially a greedy algorithm that allows for hindsight improvements.

As already mentioned, this thesis is, to a large extent, concerned with *dynamic auctions*. *Sealed-bid auctions* (or *direct revelation mechanisms*) have the advantage to have only a single round of communication between auctioneer and buyers, which makes them very easy to reason about from a game-theoretic persepective. Every buyer has only one decision point at which she might act strategically, which allows a mechanism designer use very simple (but also incredibly elegant) procedure to conduct the auction. Standing out as fundamental theorems are the VCG Theorem due to Vickrey [Vic61], Clarke [Cla71], and Groves [Gro73], and Myerson's Lemma from [Mye81] that describe sealed-bid auctions that have truthful bidding as dominant strategy. On the other hand, these direct mechanisms require all agents to reveal the complete information about their preferences to the auctioneer. This is concerning from multiple points of view. By design, the auctioneer is the only agent who has full information. In some setting it might also be possible that the auctioneer himself act strategically, for instance, by introducing shill bids to increase the seller's revenue. With less information available in a dynamic auction (for instance, only the identity of the buyer with the highest bid but not the amount of the highest bid), the auctioneer's ability to introduce a shill (and profit from doing so) is impaired [AL20; KKR24]. Of course, not revealing too much information may also be a concern of a buyer for purely privacy-related matters. Ascending auctions at least protect the winners' information as only her identity but not necessarily the exact amount of her bid is sufficient to determine a winner [MS20]. Moreover, revealing less information naturally comes with the automatic advantage that less bits have to be sent between buyers and auctioneer. Finally, while a sealed-bid auction with a (truthful) dominant strategy makes it very easy for a buyer to determine what to communicate to the auctioneer, i.e., her true preference, it might not always be clear to a cognitively limited buyer *why* she should do so. If a buyer is only asked to reveal information incrementally over time, it might be easier for her to determine that truth-telling is indeed optimal [Li17; KK24]. This is in particular true if the buyer also does not really have access to her full information because it might not be explicitly given to her but only implicitly, for instance in some combinatorial structure on which she has to solve an optimization problem. For an extended discussion of advantages of ascending auctions over sealed-bid auctions see [Aus04] and [Cra98].

We are also looking at greedy algorithms, which in every step make a locally optimal decision (for example, selecting an element from a set of available elements). With the notable exception of linear optimization over matroids, these kind of algorithms most of the time fail to find an optimal solution as it is easy to trick a greedy algorithm into making a step that does not pay off later. We propose a greedy algorithm that is slightly smarter by adding a bit of hindsight which allows the greedy algorithm to delete previously selected elements if it turns out that they are no longer necessary. In some cases this algorithm then can salvage the optimal solution and this thesis presents a natural class of functions for which this is guaranteed.

0.1 Publications

This thesis captures the content of four research papers that I contributed to. Three of the articles have been published in peer-reviewed journals. Additionally, a previous version of one of the published papers was also accepted at a peer-reviewed conference and an abstract was published in its proceedings. This thesis also includes preliminary results of unfinished work on an algorithm for the Submodular Cover problem and a class of set functions for which this algorithm maintains a partial solution with a nice structure.

List of Papers The following are the papers whose results are also in this thesis. In all of these articles I contributed in all phases of the scientific process, in particular conceptualization, analysis, and writing. As it is standard practice in theoretical computer science, discrete mathematics, and mathematical economics, authors on these papers are listed in alphabetical order.

- *A Simplified Analysis of the Ascending Auction to Sell a Matroid Base.*
  Britta Peis and Niklas Rieken
  published in *Operations Research Letters* (ORL) [PR26] (preprint: [PR24])
- *A flow-based ascending auction to compute buyer-optimal Walrasian prices.*
  Katharina Eickhoff, S. Thomas McCormick, Britta Peis, Niklas Rieken, and Laura Vargas Koch
  published in *Networks* [Eic+24a] (preprint: [Eic+23a])
- *Faster Dynamic Auctions via Polymatroid Sum.*
  Katharina Eickhoff, Meike Neuwohner, Britta Peis, Niklas Rieken, Laura Vargas

Koch, and László A. Végh
published in *Transactions on Economics and Computation* (TEAC) [Eic+25] (preprint: [Eic+24b])

> The previous version of the paper
> *Faster Ascending Auctions via Polymatroid Sum*.
> Katharina Eickhoff, Britta Peis, Niklas Rieken, Laura Vargas Koch, and László A. Végh
> has been accepted at *The 19th Conference on Web and InterNet Economics* (WINE 2023) (abstract in proceedings: [Eic+23b])

- *The Greedy+ Algorithm for Submodular Cover*.
  Waldo Gálvez, Britta Peis, Niklas Rieken, Victor Verdugo, and José Verschae
  (working paper)

Not part of this thesis are the following two papers that have been accepted at small conferences (workshops).

- *A Primal-Dual and Primal-Greedy Approximation Framework for Weighted Covering Problems*.
  Britta Peis, Niklas Rieken, José Verschae, and Andreas Wierz
  accepted at *Workshop on Models and Algorithms for Planning and Scheduling Problems* (MAPSP 2022) (paper in proceedings: [Pei+22])
- *Using Explicit (Host-to-Network) Flow Measurements for Network Tomography*.
  Ike Kunze, Constantin Sander, Alexander Ruhrmann, Niklas Rieken, and Klaus Wehrle
  accepted at *Applied Networking Research Workshop* (ANRW 2025) (paper in proceedings: [Kun+25])

0.2 Thesis Structure

Chapter 1 contains an overview over a few basic mathematical concepts that are used in this thesis and fixes some notation. In particular, we give a short introduction to graph theory, game theory, auction theory, and matroid theory. These sections are also meant to be a primer into these topics.

Chapter 2 considers an ascending auction to sell a base of a matroid. This auction was originally introduced by Bikhchandani, de Vries, Schummer and Vohra [Bik+11], where they show that if an auctioneer is constrained to sell a subset of items that is an independent set of a given matroid, then one can do so by using an ascending

auction (to be more precise, a *clock auction*). This auction also has three very desirable properties: 1. truthful behavior of the buyers is an equilibrium strategy, 2. given truthful buyers, the auction yields a welfare-maximizing allocation (i.e., a max-weight base of the given matroid), and 3. its running time scales polynomially in the number of items and buyers. While the auction itself is quite elegant, its analysis in the aforementioned paper is quite tedious and technical. We show that one can obtain the theorems for properties 1 – 3 using simpler proofs that do not require a microscopic view of the auction as in [Bik+11].

In **Chapter 3** and **Chapter 4** we again consider ascending auctions for multiple items (and also descending auctions and hybrids of both). However, there is no underlying matroid constraint for the auctioneer; the goal is to sell as many items as possible but in a way such that the allocation is stable in the sense that no buyer is willing to purchase a different set of items under their current prices. Such an allocation along with a price vector that supports this allocation is called a *Walrasian equilibrium*. These Walrasian equilbria have been studied extensively and also the auctions that can compute them. In particular, Walrasian equilibria are guaranteed to exist if all valuation functions of all buyers are *gross substitute*. We present ascending and descending auctions (and hybrids of these) using simple combinatorial structures to implement the celebrated general auction framework by Gul and Stacchetti [GS99; GS00]. Indeed, our methods improve the running time of these auctions—more precisely, to find excess demand and supply sets—compared to the previous state of the art via submodular function minimization (first identified by Ausubel [Aus06]) and improved to the more delicate $L^\natural$-convex function minimization (by Murota, Shioura, and Yang [MSY13]). This part of the thesis is shared with my great colleague Katharina Eickhoff.

Chapter 5 introduces a subclass of submodular functions that we call *generalized ranks* along with a for this class natural greedy algorithm that can solve minimum cost covering problems of certain generalized ranks to optimality. Generalized ranks seem to offer a variety of promising research directions, including a notion of duality.

Finally, in **Chapter 6** we recap the results and give a few ideas for future research directions.

CHAPTER 1

Preliminaries

1.1 Basic Maths and Notation

In this first subsection, we fix the notation that is in some sense not standardized across mathematical literature, in particular graph theory. That being said, readers familiar with the topics in this thesis will have no issue with skipping this subsection.

We use the symbols $\mathbb{R}$ and $\mathbb{Z}$ to denote the *real numbers* and *integers*, respectively, and $\mathbb{R}_+$ and $\mathbb{Z}_+$ to denote their restriction to non-negative (i.e., including 0) values. The notation $[k]$ is shorthand for the set $\{1, \dots, k\}$. All other sets in this thesis will be finite and we denote the *cardinality* of a set A by $|A|$. Given a set A and an element a, we may write $A + a := A \cup \{a\}$ and $A - a := A \setminus \{a\}$ as shorthands. We regularly consider functions $z \colon A \to \mathbb{R}$ that express how much an element $a \in A$ covers or costs, or how much it is valued. If we have some function $z \colon A \to \mathbb{R}$, we may also define its linear expansion that maps each subset of A to the sum of function values for each element in that set, i.e. we write $z \colon 2^A \to \mathbb{R}$ (a *set function*) with $z(S) := \sum_{a \in S} z(a)$, where $2^A := \{B : B \subseteq A\}$ denotes the *powerset* of A.. Observe that if A is finite, we can also view z as a vector $z = (z(a))_{a \in A}$, so we may treat a function on finite domain as a vector (or vice versa) if it is convenient.

A *graph* is a structure $G = (V, A)$, where V is the set of *vertices* and $A \subseteq \binom{V}{2} \cup \binom{V}{1}$ the (multi-)set of *undirected edges* (or $A \subseteq V \times V$ the (multi-)set of *directed edges*). Note

that this definition explicitly allows edges to be *parallel* (multiple edges connecting the same vertices (in the directed case also in the same direction)) or *loops* (edges connecting a vertex with itself). A *(v-w-)path* in a graph $G = (V, A)$ with $v, w \in V$ is a sequence of vertices $(u_0, u_1, \dots, u_k)$ with $u_0 = v$, $u_k = w$, and $\{u_{i-1}, u_i\} \in A$ (or $(u_{i-1}, u_i) \in A$ in the directed case) for all $i \in [k]$. Two vertices $v, w \in V$ are called *adjacent* if $\{v, w\} \in A$ ($(v, w) \in A$, respectively). An edge $\{v, w\} \in A$ ($(v, w) \in A$) is *incident* to both v and w. It is also called *incident* to another edge if they share a common vertex. We denote the *neighborhood* of a vertex v by $\Gamma(v) := \{w \in V \mid \{v, w\} \in A\}$ and use $\Gamma[v] := \Gamma(v) \cup \{v\}$. In the directed case, we write $\Gamma^+(v) := \{w \in V \mid (v, w) \in A\}$ and $\Gamma^-(v) := \{w \in V \mid (w, v) \in A\}$ and $\Gamma^+[v], \Gamma^-[v]$ if we want to include v. Given a set $S \subseteq V$, we may also write $\Gamma(S) := \bigcup_{v \in S} \Gamma(v)$ and $\delta(S) := \{\{v, w\} \in A \mid v \in S, w \notin S\}$ (or $\delta^+(S) := \{(v, w) \in A \mid v \in S, w \notin S\}$ (outgoing edges) and $\delta^-(S) := \{(v, w) \in A \mid v \notin S, w \in S\}$ (incoming edges) for directed graphs). A *matching* in an undirected graph is a subset of edges $\mu \subseteq A$ such that for all pairwise different $a, a' \in \mu$ it holds that $a \cap a' = \emptyset$. Vertices $v \in V$ for which there exists an edge $a \in \mu$ with $v \in a$ are called *covered* by μ, otherwise *exposed*. A matching that covers all vertices of the graph is called *perfect*.

1.2 Submodular Functions

In this section, we discuss submodular functions, which is a special class of set functions that have a *diminishing returns* property. Informally speaking this means that the addition of an element to a smaller set increases the function value more (or decreases it less) than the same element added to a bigger set. This property make submodular functions very useful in economic applications but most recently machine learning (in particular, active learning algorithms) emerged as a killer application (for an overview, we refer to the survey by Bilmes [Bil22]).

We now define the notion of a submodular function.

Definition 1.1. A set function $z\colon 2^E \to \mathbb{R}$ is called *submodular* if for all $S, T \subseteq E$, it holds that

$$z(S) + z(T) \geq z(S \cup T) + z(S \cap T)$$

There are two more ways to define submodularity.

Lemma 1.2. *Let $z\colon 2^E \to \mathbb{R}$ be a set function on E. Then the following are equivalent:*

(i) z is submodular,

(ii) for all $S \subseteq T \subseteq E$ and all $e \notin T$ it holds that

$$z(S+e) - z(S) \geq z(T+e) - z(T),$$

(iii) for all $S \subseteq E$ and all $e, f \notin S$ it holds that

$$z(S+e) + z(S+f) \geq z(S+e+f) + z(S).$$

A set function z is called *supermodular* if $-z$ is submodular and it is called *modular* if it is submodular and supermodular. A modular function can always be represented with weights $w: E \to \mathbb{R}$ such that $z(S) = \sum_{e \in S} w(e)$.

We mainly look at submodular functions that are also *normalized*, i.e., $z(\varnothing) = 0$ and *monotone*, i.e., for all $S \subseteq T \subseteq E$ it holds that $z(S) \leq z(T)$.

Given a submodular function $z: 2^E \to \mathbb{R}$, there are of course two essential problems that are of interest.

Submodular Function Minimization

Given:	a submodular function $z: 2^E \to \mathbb{R}$ on finite ground set E.
Find:	a set $S \subseteq E$ with $f(S)$ minimum.

Submodular Function Maximization

Given:	a submodular function $z: 2^E \to \mathbb{R}$ on finite ground set E.
Find:	a set $S \subseteq E$ with $f(S)$ maximum.

Submodular functions can be seen as discrete analogues of both, convex and concave functions: At first glance, the diminishing returns property of submodular functions appears to be a discrete analogue of concavity; however, for concave functions the maximization problem is easy (as there are no local maxima), while it is the minimization problem that is easy for submodular functions.

There are indeed a variety of efficient Submodular Function Minimization algorithms, e.g., by Cunningham [Cun85], Schrijver [Sch00], Iwata, Fleischer, and Fujishige [IFF01], Iwata [Iwa02; Iwa03], Iwata and Orlin [IO09], Orlin [Orl09], Lee, Sidford, and Wong [LSW15], and Chakrabarty, Lee, Sidford, and Wong [Cha+17]. Those are all far from trivial, however.

Proposition 1.3 ([Cun85; Sch00; IFF01; Iwa02; Iwa03; IO09; Orl09; LSW15; Cha+17])**.** Submodular Function Minimization *is solvable in (strongly) polynomial time.*

Practically (and successfully) tested is the Fujishige-Wolfe Algorithm [Wol76; Fuj80], which was considered a heuristic due to lack of subexponential running time guarantees until Chakrabarty, Jain, and Kothari [CJK14] showed that its running time is pseudopolynomial (with a quadratic dependency on the largest singleton value of the submodular function).

On the other hand, Submodular Function Maximization is NP-hard as the NP-complete Maximum Cut problem reduces to Submodular Function Maximization.

Proposition 1.4 ([Pap94, Theorem 9.5] for Maximum Cut)**.** Submodular Function Maximization *is* NP*-hard.*

However, there is also a catalogue of good approximation algorithms for Submodular Function Maximization, even for constrained Submodular Function Maximization. We refer to the survey by Krause and Golovin [KG14].

Our thesis is indeed concerned with both Submodular Function Minimization and Submodular Function Maximization, even though it is somewhat hidden. The auctions in Chapter 3 and 4 solve a special Submodular Function Minimization problem. The main part of our contribution is however, that we do not need the aforementioned algorithms, which are quite involved and not easy to implement;[1] instead, we provide much simpler algorithms to perform these steps that previously have been carried out with Submodular Function Minimization algorithms. On the other hand, Chapter 5 is concerned with the Submodular Cover problem, which needs to maximize a submodular coverage function at minimum cost.

1.3 Matroid Theory

The study of matroids started in the 1930s with the seminal paper by Hassler Whitney [Whi35]. He considered matroids as an abstract generalization of matrices and the notion of linear independence. However, to some extent matroids are also motivated by his earlier work in graph theory. The overlap is omnipresent in virtually every aspect of matroid theory as its nomenclature borrows a lot of words from algebra and graph theory terminology. Our notation is mainly based on Chapter 39 in the book by Schrijver [Sch03]. A modern monograph on matroids is by Oxley [Oxl06] but the author of this thesis personally prefers the classic book by Welsh [Wel76].

[1] In fact, we are not aware of any wide-spread implementation of those.

Many combinatorial optimization problems ask to determine a feasible subset of a given set of items that minimize or maximize an objective function. Often in those problems, "taking nothing" is a feasible choice as well as "taking less" of something feasible is again feasible. In some sense, matroids are a special combinatorial structure that describe a structure of feasible sets, i.e., sets of items that are not in some way constricted. Additionally to the properties above ($\varnothing$ is feasible and if I is feasible then also every subset of I is feasible), matroids have an *exchange* or *augmentation* property. There are multiple ways to define what a matroid is and we mention a few in Subsection 1.3.3. We start with a definition via the *independence axioms*.

Definition 1.5. A pair $M = (E, \mathcal{I})$ on a finite *ground set* E with a collection of *independent sets* $\mathcal{I} \subseteq 2^E$ is a *matroid* if the following three properties hold

(I1) $\varnothing \in \mathcal{I}$,

(I2) for all $I_1 \subseteq I_2 \subseteq E$ with $I_2 \in \mathcal{I}$, it holds that $I_1 \in \mathcal{I}$, and

(I3) for all $I_1, I_2 \in \mathcal{I}$ with $|I_1| < |I_2|$, there exists $e \in I_2 \setminus I_1$ such that $I_1 + e \in \mathcal{I}$.

We call a system $(E, \mathcal{I})$ that satisfies axioms (I1) and (I2) but not necessarily (I3) an *independence system*.

An inclusion-wise maximal independent set is called a *base* and we denote the set of bases of a matroid by $\mathcal{B}$. It follows from (I3) that all bases have the same cardinality. A set $X \notin \mathcal{I}$ is called *dependent*. An inclusion-wise minimal dependent set we call a *circuit* and denote the set of all circuits by $\mathcal{C}$. A circuit of cardinality 1 is also called a *loop* and for circuits with two elements $\{e, f\} \in \mathcal{C}$, we call e and f *parallel*.

To illustrate these terms above, we use a *graphic matroid*, i.e., an undirected multi-graph $G = (V, E)$ in which the edge set forms a ground set of a matroid. The independent sets of a graphic matroid are those sets of edges that induce a forest. A base in the graphic matroid is a spanning tree. Whenever a set $X \subseteq E$ contains a cycle, then X is dependent in the matroid sense and the cycles of a graph are precisely the circuits of the graphic matroid. An edge is a loop in the matroid sense if and only if it is a loop in the graph theoretic sense. Parallel edges in a graph a parallel elements of the graphic matroid.

We define the *rank function* $\rho\colon 2^E \to \mathbb{Z}_+$ of a matroid $M = (E, \mathcal{I})$ by $\rho(X) := \max\{|I| : X \supseteq I \in \mathcal{I}\}$. We may also write $\rho(M) := \rho(E)$, which equals the rank of any base of M. A matroid $M = (E, \mathcal{I})$ is called *trivial* if $\rho(M) = 0$ and *free* if $\rho(M) = |E|$.

We call a set $F \subseteq E$ a *flat* (or *closed*) if the addition of any $e \notin F$ to F increases the rank, i.e., if $\rho(F + e) = \rho(F) + 1$ for all $e \notin F$. A flat of rank $\rho(M) - 1$ is also called a *hyperplane*. We define the *span* (or *closure*) of a set X as the maximal set of elements that can be

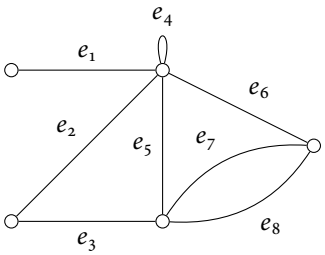


Figure 1.1: A graphic matroid. The edges $\{e_1, e_2, e_3, e_6\}$ are a spanning tree (base), $\{e_2, e_3, e_5\}$ is a cycle (circuit), $\{e_7, e_8\}$ are parallel, e_4 is a loop.

added to X without increasing the rank, i.e., $\operatorname{span}(X) := \{e \in E \mid \rho(X + e) = \rho(X)\}$. Note that $\rho(\operatorname{span}(X)) = \rho(X)$ holds for all $X \subseteq E$ and $\operatorname{span}(F) = F$ if F is a flat.

Matroids' most-known feature is that a greedy algorithm solves the problem of finding a minimum or maximum weight base for any linear weight function.

Minimum/Maximum Weight Matroid Base

Given: matroid $M = (E, \mathcal{I})$, weights $w\colon E \to \mathbb{Z}_+$.
Find: base $B \in \mathcal{B}$ of minimum/maximum weight $w(B)$.

Indeed, the Matroid Greedy Algorithm below resembles Kruskal's Algorithm [Kru56] for the Minimum Spanning Tree problem which is just this algorithm applied to a graphic matroid. For details we refer to Theorem 40.1 in [Sch03].

Matroid Greedy Algorithm:

Input: Matroid $M = (E, \mathcal{I})$, weight function w
Output: Minimum/Maximum weight base of M

```
1 I := ∅
2 for e ∈ E in ascending/descending order of w(e) do
3     if I + e ∈ 𝓘 then
4         I := I + e
5 return I
```

1.3.1 Operations on Matroids

Here we introduce some basic operations on matroids. We can generate a *minor* of a matroid by removing one or more elements from its ground set. There are two different removal operations, *deletion* and *contraction*. Given a matroid $M = (E, \mathcal{I})$ and some $X \subseteq E$, we obtain a matroid by deleting X from M, i.e., $M \setminus X := (E \setminus X, \mathcal{I} \setminus X)$, where $\mathcal{I} \setminus X := \{I \in E \setminus X \mid I \in \mathcal{I}\}$.[2] We may also write $M \,|\, X := M \setminus (E \setminus X)$ for the *restriction* of M to elements in X. We also obtain a matroid by contracting X in M, i.e., $M / X := (E \setminus X, \mathcal{I} / X)$, where $\mathcal{I} / X := \{I \in E \setminus X \mid I \cup X^{\mathcal{B}} \in \mathcal{I}\}$ and $X^{\mathcal{B}}$ denoting an arbitrary base of $M \,|\, X$. As a shorthand, we also write $M \,.\, X := M / (E \setminus X)$. It is well-known that each of the two *minors* $M \setminus X$ and M / X is indeed again a matroid (see Section 39.3 in [Scho3]). For the sake of convenience, we may also write $\mathcal{S} \setminus X$ and $\mathcal{S} / X$ for $\mathcal{S} \in \{\mathcal{B}, \mathcal{C}\}$ for the set of bases and circuits of $M \setminus X$ and M / X, respectively. If X is a singleton $\{e\}$, we may also write $M \setminus e$ and M / e instead of $M \setminus \{e\}$ and $M / \{e\}$, respectively. Also note that a series of deletions and contractions can be performed in any order of the elements, the resulting minor will always be the same (Proposition 3.1.25 in [Oxlo6]). Hence, it is valid to write $M \setminus X / Y$ to denote the minor of M after deleting X and contracting Y in any particular order, provided that these sets are disjoint.

A fundamental result by Whitney [Whi35] is that matroids allow a notion of duality.

Definition 1.6. Let $M = (E, \mathcal{I})$ be a matroid. Then $M^* := (E, \mathcal{I}^*)$ with $\mathcal{I}^* := \{I^* \subseteq E \mid$ there exists $B \in \mathcal{B}$ with $B \subseteq E \setminus I^*\}$ is the *dual matroid* of M.

It is fairly straight-forward to show that M^* is indeed a matroid.

Given a matroid M and its dual M^*, we call the independent sets $I^* \in \mathcal{I}^*$ of M^*, as well as its bases $B^* \in \mathcal{B}^*$, circuits $C^* \in \mathcal{C}^*$ the *coindependent sets*, *cobases*, and *cocircuits* of M, respectively. Similarly, we define *coloops* as cociruits of size 1.

The *corank function* of M, $\rho^* \colon 2^E \to \mathbb{Z}_+$ is the rank function of M^*. The rank and corank function of some matroid M are related via the following theorem.

Lemma 1.7 ([Oxlo6, Proposition 2.1.9])**.** *Let $M = (E, \mathcal{I})$ be a matroid with rank and corank functions ρ, ρ^*. Then*

$$\rho^*(E \setminus X) = |E \setminus X| - \rho(E) + \rho(X).$$

[2] Please note a minor detail concerning the typesetting: we use $\setminus$ to denote a set difference, which is different from \ to denote a matroid deletion.

There are some more useful connections between a matroid and its dual. For instance, we have that the complement of a hyperplane yields a cocircuit [Oxl06, Proposition 2.1.6 (iii)] and that $M / X = (M^* \setminus X)^*$ [Oxl06, (3.1.1)].

Given two matroids $M_1 = (E_1, \mathcal{I}_1)$ and $M_2 = (E_2, \mathcal{I}_2)$ with $E_1 \cap E_2 = \emptyset$, we define the *direct sum* of M_1 and M_2 by

$$M_1 \oplus M_2 := (E_1 \cup E_2, \mathcal{I}_1 \oplus \mathcal{I}_2)$$

where $\mathcal{I}_1 \oplus \mathcal{I}_2 := \{I \subseteq E_1 \cup E_2 \mid I \cap E_1 \in \mathcal{I}_1, I \cap E_2 \in \mathcal{I}_2\}$. It is again easy to verify that a direct sum of finitely many matroids yields again a matroid.

More interesting are the operations of *matroid union* and *matroid intersection*. Given matroids $M_1 = (E, \mathcal{I}_1)$ and $M_2 = (E, \mathcal{I}_2)$ on the same ground set E,[3] we can define a new structure via *matroid union*—which indeed will be a matroid—by

$$M_1 \vee M_2 := (E, \mathcal{I}_1 \vee \mathcal{I}_2)$$

where $\mathcal{I}_1 \vee \mathcal{I}_2 := \{I_1 \cup I_2 : I_1 \in \mathcal{I}_1, I_2 \in \mathcal{I}_2\}$.

Matroid Union Theorem (1.8) ([Edm70, (88–91)]). *The union of finitely many matroids $M_1 = (E, \mathcal{I}_1), \ldots, M_n = (E, \mathcal{I}_n)$ with rank functions $\rho_1, \ldots, \rho_n$ is a matroid*

$$M_1 \vee \ldots \vee M_n = (E, \mathcal{I}),$$

with rank function $\rho(S) = \min_{T \subseteq S} \left(|S \setminus T| + \sum_{i=1}^{n} \rho_i(T)\right)$ and $\mathcal{I} = \{I \subseteq E \mid \rho(I) = |I|\}$.

This prompts the question whether the Matroid Greedy Algorithm alone can solve the Matroid Union problem, i.e., given n matroids $(E, \mathcal{I}_1), \ldots, (E, \mathcal{I}_n)$, the task is to compute maximum set I that can be partitioned into n sets $I_1 \uplus \ldots \uplus I_n$ with $I_i \in \mathcal{I}_n$ for all $i \in [n]$.[4] The difficulty comes in the oracle step of the Matroid Greedy Algorithm; one cannot just ask the oracles of the individual matroids whether an element can be added to its current independent set.

Example 1.1. Consider the two graphic matroids M_1, M_2 in Figure 1.2 on $E = \{e_1, e_2, e_3\}$. If the current independent sets are $I_1 = \{e_1, e_2\}$ and $I_2 = \emptyset$, then the remaining element

[3] This is without loss of generality; if there is a non-empty symmetric difference, we can just append these elements to the matroids in which they are missing by direct sum of the trivial matroid.

[4] See Section 4.2 for a complete description.

(a) M_1 (b) M_2

Figure 1.2: The two graphic matroids from Example 1.1.

e_3 cannot be added to either set. However, $\{e_1, e_2, e_3\}$ clearly can be partitioned into independent sets, e.g., $I_1 = \{e_1, e_3\}$ and $I_2 = \{e_2\}$, i.e., $M_1 \vee M_2 = (E, 2^E)$.

An algorithm for Matroid Union has to be able to perform those kind of exchanges to reach an optimal solution. This requires a more sophisticated idea than naïve oracle calls to each individual independence oracle. We will discuss algorithms for Matroid Union in Section 4.4.

On the other hand, given two matroids $M_1 = (E, \mathcal{I}_1)$ and $M_2 = (E, \mathcal{I}_2)$ on the same ground set E, the *matroid intersection*

$$M_1 \wedge M_2 := (E, \mathcal{I}_1 \wedge \mathcal{I}_2)$$

with $\mathcal{I}_1 \wedge \mathcal{I}_2 := \{I \subseteq E, I \in \mathcal{I}_1 \cap \mathcal{I}_2\}$ is not necessarily a matroid as the following example shows.

Example 1.2. Consider the two matroids

$$M_1 = (\{e_1, e_2, e_3\}, \{\varnothing, \{e_1\}, \{e_2\}, \{e_3\}, \{e_1, e_2\}, \{e_1, e_3\}\}),$$
$$M_2 = (\{e_1, e_2, e_3\}, \{\varnothing, \{e_1\}, \{e_2\}, \{e_3\}, \{e_1, e_3\}, \{e_2, e_3\}\}).$$

Both matroids are graphic and are depicted in Figure 1.3. The matroid intersection yields

$$M_1 \wedge M_2 = (E, \{\varnothing, \{\{e_1\}, \{e_2\}, \{e_3\}, \{e_1, e_3\}\})$$

which violates (I3) for $I_1 = \{e_2\}$ and $I_2 = \{e_1, e_3\}$.

The problem of finding a maximum set that is independent in both M_1 and M_2 can be solved however, by means of the Matroid Union problem due to the following Theorem by Edmonds [Edm70].

(a) matroid M_1 (b) matroid M_2

Figure 1.3: The two graphic matroids from Example 1.2.

Matroid Intersection Theorem (1.9) ([Edm70, (104)]). *Let $M_1 = (E, \mathcal{I}_1)$ and $M_2 = (E, \mathcal{I}_2)$ be two matroids on a common ground set E with rank functions ρ_1 and ρ_2. The maximum size of a set in $\mathcal{I}_1 \cap \mathcal{I}_2$ is*

$$\min_{S \subseteq E} (\rho_1(S) + \rho_2(E \setminus S)).$$

A notable special case is where one of the two matroids is a uniform matroid (which just sets a capacity on the number of elements, see below for details): in that case the resulting matroid intersection yields another matroid. The operation of intersecting a matroid M with a uniform matroid $\mathcal{U}_n^r$ is called a *truncation* (at $r \in \mathbb{Z}_+$, the rank of the uniform matroid).

1.3.2 A Zoo of Matroids

Graphic matroids introduced above are probably the most used examples of matroids as they are easy to illustrate and the matroid terminology often has a direct twin in graph theory. However, not every matroid is graphic and there are a many more examples that are interesting and we want to list a few in this subsection. We assume $E = \{e_1, \dots, e_n\}$ for all matroids listed here.

Uniform Matroid A uniform matroid just puts a capacity $r \in \mathbb{Z}_+$ on the number of elements that can be in a set to be independent. That is, $\mathcal{U}_n^r = (E, \{X \subseteq E \mid |X| \leq r\})$.

Partition Matroid Given a partition $\mathcal{P} = \{P_1, \dots, P_k\}$ of E and k capacities $r_1, \dots, r_k \in \mathbb{Z}_+$ a set is independent if the capacity constraint for all k parts is satisfied, i.e., $M = (E, \{X \subseteq E \mid |X \cap P_i| \leq r_i, i \in [k]\})$. Note that every partition matroid is a direct sum of uniform matroids.

Laminar Matroid We say a family of sets $\mathcal{S}$ is *laminar* if for any two $S_1, S_2 \in \mathcal{S}$ with $S_1 \cap S_2 \neq \varnothing$ it holds that $S_1 \subseteq S_2$ or $S_1 \supseteq S_2$. Given a laminar family $\mathcal{S} = \{S_1, \ldots, S_k\}$ on E and k capacities $r_1, \ldots, r_k \in \mathbb{Z}_+$ a set is independent if the capacity constraints are satisfied for all sets in $\mathcal{S}$, i.e., $M = (E, \{X \subseteq E \mid |X \cap S_i| \leq r_i, i \in [k]\})$. Clearly, partitions are laminar set families, hence, this is a more general class of matroids.

Linear Matroid Given a matrix over some field $\mathbb{F}$, i.e., $A \in \mathbb{F}^{m \times n}$, a set X of columns of A is independent in the matroid sense if the column vectors in X are linearly independent over $\mathbb{F}$. Linear matroids are also called *representable* (over the field $\mathbb{F}$) and the corresponding matrix X its *representation*. A matroid that is representable over all fields is called *regular*.

It is a very interesting (yet out of the scope of this thesis) problem to determine which matroids are representable over which fields.

Transversal Matroid Given a set system $\mathcal{S} = (S_1, \ldots, S_k)$ over a finite universe $\mathcal{U}$ a *partial transversal* of $\mathcal{S}$ is a subset $T \subseteq \mathcal{U}$ such that there is an injection $\varphi : \mathcal{U} \to [k]$ with $t \in S_{\varphi(t)}$ for all $t \in T$. If $|T| = k$, then T is called a *transversal*. Edmonds and Fulkerson [EF65] showed that the set of partial transversals of any set system $\mathcal{S}$ are the set of independent sets of a matroid.

Matching Matroid Given a graph $G = (V, E)$, we can call a set of *vertices* $S \subseteq V$ *matchable* if there exists a matching $\mu \subseteq E$ covering S. The set of all matchable sets is again the set of independent sets of a matroid. Note that the set of matchings does not yield a matroid, only an independence system.

Gammoid Given a directed graph $G = (V, A)$ and $X, Y \subseteq V$, we say X is *connected* to Y, if there are $|Y|$ vertex-disjoint paths from X to Y. Let $S, T \subseteq V$, then a gammoid over G, S, and T is given by its independent sets $\mathcal{I} = \{X \subseteq T \mid S \text{ is connected to } X\}$. A gammoid is *strict* if $T = V$.

1.3.3 Matroid Cryptomorphisms

We already defined notions of bases, circuits, ranks, spans, cobases, cocircuits etc. It is quite clear that a set of bases $\mathcal{B}$ uniquely defines a matroid as we can just take $\mathcal{I} = \{I \subseteq E \mid \text{there exists a } B \in \mathcal{B} \text{ with } B \supseteq I\}$ yields a set of independent sets of a

matroid. Conveniently, we can indeed uniquely characterize matroids by any of those terms above, meaning that there are not just the axioms of independence (I1)–(I3) but we can define matroids also with base axioms, circuit axioms, rank axioms, closure axioms, hyperplane axioms, and also via their dual interpretation. Whenever two structures (e.g., a set of bases and a set of circuits) are equivalent in some semantic sense but not *obviously equivalent*, they are called *cryptomorphic*.[5] We give a list of these axioms here and want to emphasize that these are all equivalent to Definition 1.5. In the rest of this thesis, we may abuse notation and write $M = (E, \mathcal{I}) = (E, \mathcal{I}^*) = (E, \mathcal{B}) = (E, \mathcal{B}^*) = (E, \mathcal{C}) = (E, \mathcal{C}^*) = (E, \rho) = (E, \rho^*) = (E, \operatorname{span})$, to use whatever description of the matroid is convenient.

Proposition 1.10. *A pair $M = (E, \mathcal{B})$ on a finite ground set E with a collection of bases $\mathcal{B} \subseteq 2^E$ is a matroid if and only if the following two properties hold*

(B1) *$\mathcal{B} \neq \varnothing$ and*

(B2) *for all $B_1, B_2 \in \mathcal{B}$ and $e \in B_1 \setminus B_2$, there exists some $f \in B_2 \setminus B_1$ such that $B_1 - e + f \in \mathcal{B}$.*

Proposition 1.11. *A pair $M = (E, \mathcal{C})$ on a finite ground set E with a collection of circuits $\mathcal{C} \subseteq 2^E$ is a matroid if and only if the following three properties hold*

(C1) *$\varnothing \notin \mathcal{C}$,*

(C2) *for all $C_1, C_2 \in \mathcal{C}$ with $C_1 \subseteq C_2$ it holds that $C_1 = C_2$, and*

(C3) *for all $C_1, C_2 \in \mathcal{C}$ with $e \in C_1 \cap C_2$, there exists some $C_3 \subseteq (C_1 \cup C_2) - e$ with $C_3 \in \mathcal{C}$.*

Proposition 1.12. *A pair $M = (E, \rho)$ on a finite ground set E with a rank function $\rho \colon 2^E \to \mathbb{Z}_+$ is a matroid if and only if the following three properties hold*

(R1) *for all $S \subseteq E$ it holds that $0 \leq \rho(S) \leq |S|$,*

(R2) *for all $S \subseteq T \subseteq E$ it holds that $\rho(S) \leq \rho(T)$, and*

(R3) *ρ is submodular, i.e., for all $S, T \subseteq E$ it holds that $\rho(S \cup T) + \rho(S \cap T) \leq \rho(S) + \rho(T)$.*

Proposition 1.13. *A pair $M = (E, \operatorname{span})$ on a finite ground set E with a span function $\operatorname{span} \colon 2^E \to 2^E$ is a matroid if and only if the following four properties hold*

(S1) *for all $S \subseteq E$ it holds that $S \subseteq \operatorname{span}(S)$,*

(S2) *for all $S \subseteq T \subseteq E$ it holds that $\operatorname{span}(S) \subseteq \operatorname{span}(T)$,*

[5] However, while there are cryptomorphic concepts outside of matroid theory, the term seems to be used mainly among researchers in matroid theory and only occasionally outside of this community.

(S3) *for all* $S \subseteq E$ *it holds that* $\operatorname{span}(\operatorname{span}(S)) = \operatorname{span}(S)$, *and*
(S4) *for all* $S \subseteq E$ *and* $e, f \in E$ *such that* $e \in \operatorname{span}(S + f) \setminus \operatorname{span}(S)$ *it holds that* $f \in \operatorname{span}(S + e)$.

Note that additionally to those presented axiom systems, which we may call *base axioms*, *circuit axioms*, *rank axioms*, and *span axioms*, respectively, we have corresponding axiom systems in the dual, which we will not state explicitly again here (as they are the same). Proofs that those axiom systems also completely determine a matroid in the sense of Definition 1.5 can be found in virtually every textbook that touches matroid theory, in particular of course in [Wel76] and [Oxl06].

Typically, matroids are not given explicitly as inputs for a computational problem as the list of bases, independent sets, circuits, etc. can become too long. Knuth [Knu74] showed that for the number g_m of non-isomorphic loopless matroids on ground set of size m, there is a constant $c > 0$ such that for sufficiently large values of m,

$$\log\log g_m \geq m - \tfrac{3}{2}\log m + c\log\log m.$$

As a consequence, there cannot exists an encoding scheme (over a finite alphabet) such that every matroid's encoding has a length that is polynomially bounded by m. Therefore, when using matroids as part of input of a computational problem, we just use the ground set E as input and assume that other information (whether a set is independent, a base, or a circuit etc.) is queried from an oracle (*"Is* $X \in \mathcal{I}$*?"*, *"Is* $X \in \mathcal{B}$*?"*, *"Is* $X \in \mathcal{C}$*?"*, etc.). If we want to specify the running time of an algorithm on a matroid, we then describe it in terms of just the size of the ground set and the number of oracle calls. Mayhew [May08] studies the impact if one would not do that and how the hardness of a problem can be affected if an oracle (say for bases) is replaced by a list in the input (e.g., a list of bases).

1.3.4 Polymatroids

There is a natural extension of matroid theory to define a notion of independence to vectors instead of subsets (which are Boolean vectors). It is most convenient to generalize the rank axioms (see Proposition 1.12) of matroids, which just requires us to omit the restriction that the rank cannot be larger than the cardinality of the set (except the empty set) and the integrality constraint.

Definition 1.14. A pair $P = (E, \rho)$ on a finite *ground set* E and real-valued *rank function* $\rho\colon 2^E \to \mathbb{R}$ is a *polymatroid* if

- ρ is normalized, i.e., $\rho(\varnothing) = 0$,
- ρ is non-decreasing, i.e if $S \subseteq T \subseteq E$, then $\rho(S) \leq \rho(T)$, and
- ρ is submodular, i.e., for all $S, T \subseteq E$, $\rho(S \cup T) + \rho(S \cap T) \leq \rho(S) + \rho(T)$.

In a polymatroid, a point x is *independent* if for all $S \subseteq E$ it holds that $x(S) \leq \rho(S)$. If additionally $x(E) = \rho(E)$ holds, then x is also called a *base*. If $\rho(e) \leq r$ for some $r \in \mathbb{R}$ for all $e \in E$, we also call the polymatroid a *r-polymatroid*. A polymatroid is called *integer* (or *discrete*) if ρ maps each set into $\mathbb{Z}_+$. It is easy to see that matroids are just integer 1-polymatroids.

Every integer polymatroid can be reduced to a matroid (on a potentially exponentially larger ground set) using Helgason's reduction [Hel72].

Theorem 1.15 ([Hel72, Theorem 3.3])**.** *A polymatroid $P = (E, \rho)$ is an integer polymatroid if and only if there exists a matroid $M = (E', \rho_M)$ and a function $\varphi\colon E \to 2^{E'}$ such that $\rho(S) = \rho_M\left(\bigcup_{e \in S} \varphi(e)\right)$ for all $S \subseteq E$.*

1.4 Game Theory

Game theory is the formal study of conflict and cooperation that occurs everywhere where multiple agents with different interests interact. It is concerned with the mathematical modeling and solving problems where players with their own objectives, that typically do not align with the objectives of the others, interact and influence each other. The development of game theory started with mathematicians, most notably John von Neumann, who is also well-known for contributions in logic, computer science, economics, and theoretical physics. He and Oskar Morgenstern published the first monograph on game theory that laid out the foundation of this fascinating subject [NM47]. In the 1950s, a group of young researchers at Princeton University developed game theory further, among them Harold Kuhn, Lloyd Shapley, and of course, John Nash.

1.4.1 Strategic Games

In game theory, one distinguishes between *cooperative* and *non-cooperative* games. This thesis is only concerned with the latter. In this subsection, we give definitions for

strategic form games and their solution concepts. In the subsequent section, we also consider *extensive form games* which allow players to act sequentially instead of all at once, which we need to analyze the auctions in the subsequent chapters of this thesis.

Definition 1.16. A *strategic game* (in standard form) is a triple $\Gamma = (N, (\Sigma_i)_{i\in N}, (u_i)_{i\in N})$, where

- $N = \{1, \dots, n\}$ is a set of *players* (or *agents*),[6]
- for all $i \in N$, there is a set of *strategies* Σ_i (or *actions*), and
- for all $i \in N$, there is a *utility function* $u_i \colon \Sigma_1 \times \dots \Sigma_n \to \mathbb{R}$.

The idea is that all players $i \in N$ simultaneously choose a strategy $\sigma_i \in \Sigma_i$, which yields a *strategy profile* (or *outcome*) $\sigma = (\sigma_1, \dots, \sigma_n) \in \Sigma := \bigtimes_{i\in N} \Sigma_i$.

As the name suggests, a player i prefers a profile σ over a profile τ if $u_i(\sigma) > u_i(\tau)$. That is, players want to maximize their utility functions.[7]

If the game is small enough (in both, the number of players and the number of strategies), a game can be represented completely using a payoff matrix.

Given a game $\Gamma = (N, (\Sigma_i)_{i\in N}, (u_i)_{i\in N})$, we can define *social scoring function* $U \colon \Sigma \to \mathbb{R}$ that maps every profile to a value that signifies how desirable this outcome is from a social point of view. There are a lot of possibilities for social scoring functions but the most common ones are the *utilitarian welfare function* ($U(\sigma) = \sum_{i\in N} u_i(\sigma)$) and the *egalitarian welfare function* ($U(\sigma) = \min_{i\in N} u_i(\sigma)$). While a single optimizer usually aims to find a profile that is optimal w.r.t. some social scoring function, players in a game have individual goals in mind (their own utility). Hence, it is interesting to understand in what profile a game might end up. Naturally, given a strategy profile σ, we can ask for any player $i \in N$, whether σ_i is the best strategy she can use against the strategies that the other players play. To make notation a little bit more convenient, we write (σ_i, σ_{-i}) when we want to talk about i's strategy against the strategies of the other players $\sigma_{-i} := (\sigma_1, \dots, \sigma_{i-1}, \sigma_{i+1}, \dots, \sigma_n)$ Similarly, we may write $\Sigma_{-i} := \bigtimes_{j\in N\setminus\{i\}} \Sigma_j$ for the set of strategy profiles without i.

[6] All games considered in this thesis will be *auction games* and only have a finite number of players.

[7] However, it is also possible to instead use cost functions that the players want to minimize. Indeed, both definitions are equivalent in the sense that one can transform a cost minimization game into a utility maximization game and vice versa by multiplying either type of function by -1.

Definition 1.17. Given a strategic game $\Gamma = (N, (\Sigma_i)_{i\in N}, (u_i)_{i\in N})$, a strategy profile $\sigma \in \Sigma$, and a player $i \in N$, we say i plays a *best response* against σ_{-i} if σ_i solves

$$\begin{aligned} &\text{maximize } u_i(\sigma_i, \sigma_{-i}) \\ &\text{subject to } \sigma_i \in \Sigma_i. \end{aligned}$$

Typically, in games that are represented in a payoff matrix, searching for a best response of a player given the strategies of the other players is just an exhaustive search in the matrix. However, as the formulation in Definition 1.17 suggests, this search might actually be an arbitrary complex optimization problem.

Using the definition of a best response, we can now also define the most common solution concept for strategic games. A profile σ is stable if no player can unilaterally change her strategy to improve her own utility.

Definition 1.18. Let $\Gamma = (N, (\Sigma_i)_{i\in N}, (u_i)_{i\in N})$ be a strategic game. The strategy profile $\sigma \in \Sigma$ is a *Nash equilibrium* (in pure strategies) if for all $i \in N$, $u_i(\sigma) \geq u_i(\tau_i, \sigma_{-i})$ for all $\tau_i \in \Sigma_i$.

A Nash equilibrium in pure strategies might not always exist. However, if we allow players to use a probability distributions over their strategy set, then we can also define Nash equilibria in mixed strategies which are guaranteed to exist for every finite (in N and Σ) game.

Definition 1.19. Given a strategic game $\Gamma = (N, (\Sigma_i)_{i\in N}, (u_i)_{i\in N})$, a *mixed strategy* for player $i \in N$ is a probability distribution over Σ_i, i.e., a function $\mu_i \colon \Sigma_i \to [0, 1]$ with the property that $\sum_{\sigma_i\in\Sigma_i} \mu_i(\sigma_i) = 1$.[8]

We denote the *set of mixed strategies* of a player $i \in N$ by $\Delta_i = \{\mu_i \in [0,1]^{\Sigma_i} \mid \sum_{\sigma_i\in\Sigma_i} \mu_i(\sigma_i) = 1\}$ and the *set of mixed strategy profiles* by $\Delta := \times_{i\in N} \Delta_i$. We also may use the shorthand notation (μ_i, μ_{-i}) for strategy profile $\mu \in \Delta$.

For all our purposes, we assume that players are *risk-neutral*, i.e., they are only interested in maximizing their *expected utility*

$$\mathbb{E}u_i(\mu) = \sum_{\sigma\in\Sigma} u_i(\sigma) \prod_{j\in N} \mu_j(\sigma_j).$$

This yields the following definition of a mixed Nash equilibrium.

[8] If we consider games with an infinite number of strategies, then we have $\int_{\Sigma_i} \mu_i(\sigma_i) \mathrm{d}\sigma_i = 1$.

Definition 1.20. Let $\Gamma = (N, (\Sigma_i)_{i\in N}, (u_i)_{i\in N})$ be a strategic game. The mixed strategy profile $\mu \in \Delta$ is a *Nash equilibrium* (in mixed strategies) if for all $i \in N$, $\mathbb{E}u_i(\mu) \geq \mathbb{E}u_i(\nu_i, \mu_{-i})$ for all $\nu_i \in \Delta_i$.

Nash's Theorem (1.21) ([Nas50, Theorem 1])**.** *Every finite game has a Nash equilibrium in mixed strategies.*

Multiple equilibria might exist in a game, which makes it still difficult to predict an outcome of a game. A stronger solution concept is that of *dominant strategies* which—if they exist—makes reasoning about the game very obvious. Excluding irrational play, it is very natural to assume that if a player has a dominant strategy, then she will also play it.[9]

Definition 1.22. Given a strategic game $\Gamma = (N, (\Sigma_i)_{i\in N}, (u_i)_{i\in N})$, a strategy $\sigma_i \in \Sigma_i$ *dominates* $\tau_i \in \Sigma_i$ if $u_i(\sigma_i, \sigma_{-i}) \geq u_i(\tau_i, \sigma_{-i})$ for all $\sigma_{-i} \in \Sigma_{-i}$ and $u_i(\sigma_i, \sigma_{-i}) > u_i(\tau_i, \sigma_{-i})$ for some $\sigma_{-i} \in \Sigma_{-i}$. We say σ_i *strictly dominates* τ_i if the inequality is strict for all $\sigma_{-i} \in \Sigma_{-i}$. A strategy $\sigma_i \in \Sigma_i$ is (*strictly*) *dominant* if it (strictly) dominates all $\tau_i \in \Sigma_i \setminus \{\sigma_i\}$.

1.4.2 Extensive Form Games

In this thesis, we are mainly interested in games in which the strategies of the players are chosen not simultaneously but sequentially. Games like this are called *extensive form games* and they can be modeled by a game tree.

Definition 1.23. An *extensive form game* is given by the quadruple $\Gamma = (N, G, (\Sigma_i)_{i\in N}, (u_i)_{i\in N})$, where

- $N = \{1, \dots, n\}$ is a set of *players* (or *agents*),
- $G = (V, A)$ is the *game tree* on vertices V (*states*), partitioned into $\{V_i\}_{i\in N}$, and edges A corresponding to *actions* that are available to a player: if $a = (v, w) \in A$ and $v \in V_i$, then i can take action a if the game reached the state v. The game tree has a *root vertex* v_0 and *terminals* $T \subseteq V$ (corresponding to *outcomes*).
- for all $i \in N$, there is a set of *strategies* $\Sigma_i = A^{\mathfrak{I}_i}$, where $\mathfrak{I}_i$ partitions V_i into *information sets*, i.e., states that i cannot distinguish (because, e.g., a state was reached via unobservable actions of other players).[10]

[9] In practice, irrational play might occur, however. If a game is too complicated such that a player might not understand that she has a dominant strategy, she might play a different one.

[10] In a game of perfect information, the information sets are all singletons, i.e., $\mathfrak{I}_i = \{\{v\} : v \in V_i\}$.

- for all $i \in N$, there is a *utility function* $u_i \colon T \to \mathbb{R}$.

The information that is available to a player i at vertex v is the history of information sets and actions she took until v is reached. We denote the *history* for player i at vertex v by $H_i(v) := \{\mathfrak{I}_i(w) : w \text{ on } v_0\text{-}v\text{-path and } w \in V_i\} \cup \{a = (w, w') \in A \mid a \text{ on } v_0\text{-}v\text{-path and } w \in V_i\}$.
A *subgame* of an extensive form game $\Gamma = (N, G, (\Sigma_i)_{i\in N}, (u_i)_{i\in N})$ is the game Γ restricted to a subtree $G' \sqsubseteq G$.

Definition 1.24. A strategy profile σ in of an extensive form game $\Gamma = (N, G, (\Sigma_i)_{i\in N}, (u_i)_{i\in N})$ is a *subgame perfect Nash equilibrium* if for each subgame Γ' of Γ, σ restricted to Γ' is a Nash equilibrium of Γ'.

Given an extensive form game with *perfect information*, we can compute a *subgame perfect Nash equilibrium* with Zermelo's Algorithm.[11]

1.4.3 Bayesian Games

In some games, players might not know about the complete utility functions of the other players. This is, for instance, very much the case in auctions, which are a main theme of this thesis. Valuation functions for goods and items are usually considered *private information* and only known to the agent holding the valuation function. To account for this, we also briefly introduce *Bayesian games* (or *incomplete information games*). The private information is captured by the concept of an *epistemic type*, describing the knowledge of a player about the game. We make however, two assumptions to simplify the concept of Bayesian games. Both assumptions are not really restrictive in the context of auctions.

- Only utility functions can be private information. The number of players and their strategy spaces are always the same and independent of the type of the players.
- At all states of the game, players have a belief about the other players utilities, i.e., a probability distribution over types. All information that is known before receiving any private information is common knowledge. In Bayesian terms, this is referred to as the *common prior*.

As it is natural in the Bayesian setting, beliefs are always updated using Bayes' rule whenever a player receives new information.

[11] *backwards induction*

Definition 1.25. A *Bayesian game* is a quintuple $\Gamma = (N, (\Sigma_i)_{i\in N}, (\Theta_i)_{i\in N}, F, (u_i)_{i\in N})$, where

- $N = \{1, \dots, n\}$ is a set of *players* (or *agents*),
- for all $i \in N$, there is a set of *strategies* Σ_i (or *actions*),
- there is a *type space* Θ_i for each player $i \in N$ ($\Theta := \bigtimes_{i\in N} \Theta_i$),
- there is a *common prior* $F\colon \Theta \to [0, 1]$, a probability distribution over the players' types, and
- for all $i \in N$, the *utility function* does not only depend on the strategy profile but also on the players' types $u_i\colon \Sigma \times \Theta \to \mathbb{R}$.

We assume that over the course of a Bayesian game, all players learn their own type eventually.

Bayesian games are divided into three stages:

ex ante No player knows about the type of any player (including herself). When we reason about strategies at this stage, we use the expected utility

$$\mathbb{E}[u_i(\sigma_i, \sigma_{-i}; \vartheta_i, \vartheta_{-i}) \mid \vartheta \sim F].$$

interim Players know their own type but not the type of the other players. Consequently, at this stage we consider the expected utility

$$\mathbb{E}[u_i(\sigma_i, \sigma_{-i}; \vartheta_i, \vartheta_{-i}) \mid \vartheta_{-i} \sim F_{-i|\vartheta_i}]$$

ex post Players know the types of every player. In this final stage, we do not need expected utilities:

$$u_i(\sigma_i, \sigma_{-i}, \vartheta_i, \vartheta_{-i}).$$

This allows us also to define three stages of equilibria.

Definition 1.26. Given a Bayesian game $\Gamma = (N, (\Sigma_i)_{i\in N}, (\Theta_i)_{i\in N}, F, (u_i)_{i\in N})$, a strategy profile σ is an *ex ante Bayes-Nash equilibrium* if for all $i \in N$ and all $\tau_i \in \Sigma_i$ it holds that

$$\mathbb{E}[u_i(\sigma_i, \sigma_{-i}; \vartheta_i, \vartheta_{-i}) \mid \vartheta \sim F] \geq \mathbb{E}[u_i(\tau_i, \sigma_{-i}; \vartheta_i, \vartheta_{-i}) \mid \vartheta \sim F].$$

It is an *interim Bayes-Nash equilibrium* if for all $i \in N$ and all $\tau_i \in \Sigma_i$ it holds that

$$\mathbb{E}[u_i(\sigma_i, \sigma_{-i}; \vartheta_i, \vartheta_{-i}) \mid \vartheta_{-i} \sim F_{-i|\vartheta_i}] \geq \mathbb{E}[u_i(\tau_i, \sigma_{-i}; \vartheta_i, \vartheta_{-i}) \mid \vartheta_{-i} \sim F_{-i|\vartheta_i}].$$

The profile σ is an *ex post Bayes-Nash equilibrium* (or just *ex post Nash equilibrium*, as all Bayesian uncertainty is removed at this stage) if for all $i \in N$, all possible types $\vartheta \in \Theta$, and all $\tau_i \in \Sigma_i$ it holds that

$$u_i(\sigma_i, \sigma_{-i}, \vartheta_i, \vartheta_{-i}) \geq u_i(\tau_i, \sigma_{-i}, \vartheta_i, \vartheta_{-i}).$$

1.5 Auction Theory

The first three main chapters of this thesis are on auctions. Auctions are omnipresent in our society and a common mean of trade of goods or services. The earliest auctions (end of the Babylonian age) were conducted to trade women for marriage [HerBC], during the Renaissance and early modern era auctions were used more frequently in England to trade goods and leaseholds [Cas23; Pat70]. Today's auctions are used to trade anything from antiques, real estates, livestock, paintings and other rare collectibles to radio spectrum licenses, carbon emissions, all kinds of financial assets, and online advertising [Kle99].

Auction Theory is a branch of economics that got popular when William Vickrey [Vic61] first presented the advantages of a second-price sealed-bid auction. In particular, the quests of designing auctions that are *efficient* (maximize social welfare), *optimal* (maximize revenue), and *strategy-proof* (incentivize truthful bidding) are interesting.

In this thesis, we are mainly concerned with the objectives of *efficiency* (in the economic and computational sense) and *incentive compatibility* (i.e., incentivize truthful bids). We introduce a few concepts from *mechanism design*, the science of rule making. Before we introduce the somewhat abstract definitions, we first give an intuition of the task and an example. In *algorithm design*, the task is to design an efficient process from a problem description via reformulations to mathematical problems that are (hopefully) well understood. An underlying assumption is (most of the time) that the whole input is a priori accessible to the process to obtain an optimal solution and that every step of the algorithm is carried out as intended. On the other hand, in *mechanism design*, this is not the case. In some sense, the task is still to design an algorithm but this time, parts of the input are not necessarily revealed to the algorithm as they are private information by strategic agents who—within some rules—perform certain steps as they might think is best for them (not for the overall system).

Example 1.3. There are n buyers $N = \{1, \dots, n\}$ who are interested in a single item. All buyers $i \in N$ have a valuation for the item $v_i \in \mathbb{Z}_+$ that is only known to them. Suppose a central authority wants to give the item to a buyer i^* with maximum valuation, i.e., $i^* \in \arg\max\{v_i : i \in N\}$ (to maximize (utilitarian) social welfare). If the valuations are known to a central authority, then this is a trivial linear search to find the maximum. However, as these valuations are a priori private, the task becomes more difficult: if the buyers know that the central authority will choose a buyer with maximum valuation and just gives the item to her, then upon asking for the valuations, a strategic agent would just answer with a exceptionally high number b_i for the item in order to win it and obtain the utility v_i for it.

The goal in mechanism design is to define a mechanism (i.e., an algorithm, typically in style of an auction) that solves the incentive problem while still being efficient. A mechanism induces a game (in the sense of Section 1.4) played by the buyers which we can analyze with respect to various solution concepts. Concretely, the mechanism should achieve the following three goals:

(G1) bidding truthfully is a dominant strategy and never leads to negative utility,
(G2) the resulting allocation should be economically efficient, and
(G3) the auction runs in (strongly) polynomial time.

Now let us define what a mechanism environment and a mechanism formally is.

Definition 1.27. A *mechanism design environment* is a triple $(N, \Omega, (u_i)_{i \in N})$, where

- $N = \{1, \dots, n\}$ is a set of *agents*,
- Ω is a set of *outcomes*, and
- for all $i \in N$, there is a *utility function* $u_i \colon \Omega \to \mathbb{R}$.

The set of outcomes Ω can be anything. In social choice theory it is typically a ranking or a winner among a set of alternatives that the agents can vote on and a mechanism therefore would be a ballot format and a way to evaluate the collection of all ballots. We consider rather *markets* in which a fixed set of *items* E is to be distributed to the agents (which we call *buyers* or *bidders* in the sequel) in exchange for *prices*. Hence, the set of outcomes Ω is described by the set of functions $(2^E)^N$ (for allocations) and $\mathbb{Z}_+^N$ (for prices). However, note that the utility functions u_i are typically constant on the components that are not describing the items or prices of i. We also restrict our attention to utility functions that are quasi-linear in price and items, i.e., we can write the utility function always as $u_i(x_i, p_i) = v_i(x_i) - p_i$ for some *valuation function* $v_i \colon 2^E \to \mathbb{Z}_+$.

A mechanism is a process that has to acquire some information about the agents' utility functions and return an outcome. We denote the set of information that a mechanism might be able to receive (i.e., bids, truthful or not) by $\mathfrak{U}$.

Definition 1.28. A *mechanism* (or in the following an *auction*) for a mechanism design environment $(N, \Omega, (u_i)_{i \in N})$ is a triple $\mathfrak{M} = (C, x, p)$, where

- C is an abstract process to collect some information $b \in \mathfrak{U}$ about the buyers' utilities,
- $x: \mathfrak{U} \to \Omega$ is the *allocation rule*, and
- $p: \mathfrak{U} \to \mathbb{Z}_+^N$ is the *pricing rule*.

We call a mechanism that asks all agents for their complete utility functions a *direct mechanism* (or *sealed-bid mechanism*) and reference a direct mechanism by only (x, p). Otherwise, C can be an arbitrarily complex communication protocol that specifies precisely the back-and-forth exchange of information between the mechanism (the auctioneer) and the agents (the buyers).

We saw in Example 1.3 how the naïve approach of just letting buyers bid an arbitrary high number as their valuation without punishing overbidding backfires immediately: bidding truthfully is not a dominant strategy; there actually does not even exist a Nash equilibrium (as Nash's Theorem does not apply here). With a lack of incentive guarantees, economical efficiency also becomes unachievable as choosing the maximum number would be just as good as picking a buyer at random.

A much more sensible approach is to let the winning buyer i^* pay her bid b_{i^*}, this is called a *first-price auction*. Clearly this solves the problem of agents overbidding: if a buyer bids higher than her valuation then she is guaranteed to receive non-positive utility. However, truthful bidding is still not incentivized (as truthful bid guarantees zero utility). There is also no dominant strategy in first-price auctions as the best response of the buyer with the highest valuation clearly depends on the highest bid of the other buyers. It is a simple exercise to see that an ex ante Bayes-Nash equilibrium of a first-price auction is where every buyer bids the expected value of the highest valuation except their own if it is smaller than their own valuation and zero otherwise, this is commonly referred to as *bid shading*.

The key to obtain an incentive compatible mechanism is the observation above: if a buyer does not have to worry to overpay by not shading her valuation enough and instead is just guaranteed to pay whatever would have been optimal to bid in a first-price auction, then truthful bidding is a dominant strategy. We call the auction that assigns the item to the agent with the highest bid but only charges the amount of the

second-highest bid (plus some small increment) *second-price* or *Vickrey auction* [Vic61]. Clearly, economic and computational efficiency are also obtained for the second-price auction.

Theorem 1.29 ([Vic61, Section III]). *The single-item second-price auction is dominant strategy incentive compatible, achieves maximum utilitarian social welfare, and is implementable in polynomial time.*

Theorem 1.29 is also the special case of Myerson's Lemma below. A mechanism design environment $(N, \Omega, (v_i)_{i\in N})$ is *single-parameter* if $\Omega \subseteq \mathbb{R}_+^N$ (how much each buyer gets allocated) and each buyers valuation can be captured in a single value v_i (the value she receives per unit of good). That is, there is a single type of good that is to be distributed among buyers and the utility of buyer i for an allocation $x \in \Omega$ for prices $p = (p_1, \dots, p_n)$ is completely determined by x_i, v_i, and p_i as $u_i = v_i x_i - p_i$.

In single-parameter environments it is very easy to build incentive compatible mechanisms as there is essentially a unique way to do it as shown by Myerson [Mye81]. We call an allocation rule $x\colon \mathbb{R}_+^N \to \mathbb{R}_+^N$ *monotone* if for all $b \in \mathbb{R}_+^N$ and all $i \in N$ it holds that $x_i(b) \le x_i(b_i', b_{-i})$ whenever $b_i' \ge b_i$.

Myerson's Lemma (1.30) ([Mye81, Lemma 3]). *For a single-parameter environment and an allocation rule x it holds that x can be extended to a incentive-compatible sealed-bid mechanism if and only if x is monotone. Moreover, in that case the payment rule is unique: Given bids b we set the payment of buyer i to*

$$p_i(b) = b_i x_i(b) - \int_0^{b_i} x_i(\xi, b_{-i})\,\mathrm{d}\xi.$$

Next we consider multi-parameter environments, i.e., general mechanism design environments. It is fairly clear that goal (G3) is not really achievable in the general case as the determination of a social welfare maximizing allocation alone can already be an NP-hard combinatorial optimization problem.

Remarkably, there is a very simple sealed-bid mechanism, the VCG Mechanism [Vic61; Cla71; Gro73], that is dominant strategy incentive compatible and it can be viewed as a direct extension of the second-price rule for single-item auctions we have seen above. In fact, there are infinitely many sealed-bid mechanisms that are VCG mechanisms, we introduce the one with the *Clarke pivot rule* [Cla71] which yields non-negative payments (i.e., no subsidies for losing buyers). Its correctness proof is easy but instructive

and hence, restated in this thesis. It can also be found in any set of lecture notes on mechanism design.

VCG Theorem (1.31) ([Vic61; Cla71; Gro73]). *For every mechanism design environment* $(N, \Omega, (v_i)_{i\in N})$*, the (direct)* VCG Mechanism *given by an allocation rule* $x\colon \mathfrak{U} \to \Omega$ *that satisfies*

$$x(b) \in \arg\max_{x\in\Omega} \sum_{i\in N} b_i(x)$$

together with payment rule that sets for all $i \in N$

$$p_i(b) = \max_{x\in\Omega} \sum_{j\in N\setminus\{i\}} b_j(x) - \sum_{j\in N\setminus\{i\}} b_j(x(b)),$$

is dominant strategy incentive compatible and maximizes social utilitarian welfare.

Proof. Assuming truthful bids, the welfare-maximization part is clear by the choice of x.

To show that this is a reasonable assumption, we need to show that (x, p) is dominant strategy incentive compatible, i.e., for all buyers $i \in N$ and every bid vector b_{-i} of the other agents, i maximizes her utility $v_i(x(b)) - p_i(b)$ by using the bid $b_i = v_i$.

So let $i \in N$ and b_{-i} be arbitrary and let x^* be the outcome chosen by the allocation rule $x(b)$ (for some bid b_i). Then i's utility is

$$v_i(x(b)) - p_i(b) = \underbrace{\left(v_i(x(b)) + \sum_{j\in N\setminus\{i\}} b_j(x(b))\right)}_{(\heartsuit)} - \underbrace{\left(\max_{\omega\in\Omega} \sum_{j\in N\setminus\{i\}} b_j(\omega)\right)}_{(\spadesuit)}.$$

The first term $(\heartsuit)$ involves $x(b)$, which depends on b_i but the second term $(\spadesuit)$ is a constant w.r.t. b_i. Thus, maximizing i's utility (i.e., finding i's best response against b_{-i}) boils down to maximizing $(\heartsuit)$. Now recall that $x(b)$ is chosen as the outcome that maximizes $\sum_{j\in N} b_j(x)$ $(\clubsuit)$ among all $x \in \Omega$. Hence, if i bids $b_i = v_i$ then $(\clubsuit)$ becomes identical to $(\heartsuit)$, which is the term that i wants to have maximized. Thus, bidding truthfully results in the mechanism choosing an outcome that maximizes i's utility; there cannot be a better bid.

We still need to show that submitting a truthful bid yields non-negative utility, i.e., the truthful bid is at least as good as not participating. Assume contradiction that $p_i(b) > b_i(x(b))$ for some $i \in N$, i.e.

$$\begin{aligned} & \max_{x \in \Omega} \sum_{j \neq i} b_j(x) - \sum_{j \neq i} b_j(x(b)) > b_i(x(b)) \\ \iff \quad & \max_{x \in \Omega} \sum_{j \neq i} b_j(x) > \sum_{j \in N} b_j(x(b)). \end{aligned}$$

But with $\max_{x \in \Omega} \sum_{j \in N} b_j(x) \geq \max_{x \in \Omega} \sum_{j \neq i} b_j(x)$, we obtain a contradiction as $x(b)$ maximizes the left hand side. □

To close this final section of this chapter, we introduce one more neat theorem that states that, when designing a mechanism, it is not too important to aim for goal (G1) as a whole. Instead it suffices to design a mechanism in which every buyer has a dominant strategy. From there we get truthfulness automatically in a direct mechanism that can be constructed from the original mechanism. Due to its relevance for the ascending auction in Chapter 2, we also give a proof to the theorem.

Theorem 1.32 (Revelation Principle, [Mye81, Lemma 1])**.** *For every mechanism $\mathfrak{M} = (C, x, p)$ in which every buyer $i \in N$ has a dominant strategy, there exists an equivalent direct mechanism $\mathfrak{M}'$ that is dominant strategy incentive compatible.*

Proof. The idea is to simulate $\mathfrak{M}$ within $\mathfrak{M}'$. Let us call $s_i(v_i)$ the dominant strategy of buyer i w.r.t. her valuation v_i in the mechanism $\mathfrak{M}$.

We now construct the direct mechanism $\mathfrak{M}' = (x', p')$ as follows: The buyers are asked to submit sealed-bids b_i to $\mathfrak{M}'$. Then $\mathfrak{M}'$ determines the dominant strategy w.r.t. these b_is, i.e., $\sigma_i(b_i)$ for every $i \in N$ and communicates according to this strategy with $\mathfrak{M}$ which applies its allocation rule $x((\sigma_i(b_i))_{i \in N})$ and payment rule $p((\sigma_i(b_i))_{i \in N})$. Then $\mathfrak{M}'$ just uses the same output as $\mathfrak{M}$. Formally, $x'(b) = x((\sigma_i(b_i))_{i \in N})$ and $p_i'(b) = p_i((\sigma_i(b_i))_{i \in N})$ for all $i \in N$.

Then $\mathfrak{M}'$ is dominant strategy incentive compatible. If a buyer i has valuation v_i, then submitting $b_i \neq v_i$ can only result in $\mathfrak{M}'$ playing a strategy that is different from $\sigma_i(v_i)$ which can only decrease i's utility as $\sigma_i(v_i)$ is dominant. Formally, for all $i \in N$, the right hand side of

$$v_i(x'(b_i, b_{-i})) - p_i'(b_i, b_{-i}) = v_i(x((\sigma_j(b_j))_{j \in N})) - p_i((\sigma_j(b_j))_{j \in N})$$

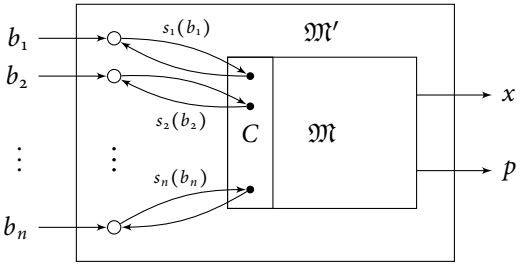


Figure 1.4: Sketch of the proof of Theorem 1.32.

is maximized by setting $b_i = v_i$, independent of b_{-i} as $\sigma_i(v_i)$ is a dominant strategy for the mechanism $\mathfrak{M} = (C, x, p)$. □

In Figure 1.4 you find a sketch of the mechanism $\mathfrak{M}'$ from the proof, which essentially adds a proxy-agent for all buyers to play according to their dominant strategy. Note that $\mathfrak{M}'$ is a direct mechanism, every buyer has to submit her whole private valuation (or a fake one, which she should not), while $\mathfrak{M}$ can be a dynamic mechanism, i.e., a mechanism that involves more than one round of communication.

CHAPTER 2

Selling a Matroid Base via an Ascending Auction

In this chapter, we consider an ascending multi-item auction, where the auctioneer is constrained to sell an independent set of a given matroid. The auction was introduced by Bikhchandani, de Vries, Schummer, and Vohra in the paper *"An Ascending Vickrey Auction for Selling Bases of Matroids"* at SODA 2008 [Bik+08] and published in the journal OR in 2011 [Bik+11] (in the sequel, we only reference the journal version). While the auction itself is very elegant, its analysis in the original papers is quite laborious and cumbersome. We show that this does not need to be the case with a simpler analysis.

2.1 The Economic Model

We consider a market consisting of a set of m indivisible distinguishable *items* E and a set of n *buyers* N. Each buyer $i \in N$ is interested in a subset of items $E_i \subseteq E$ and has a *valuation* $v_i \colon E_i \to \mathbb{Z}_+$. We may also write $v_i(X) = \sum_{e \in X} v_i(e)$ for any subset $X \subseteq E_i$, i.e., the valuation of each buyer is additive. The auctioneer is constrained to sell an independent set of a matroid $M = (E, \mathcal{I})$.

Recall from Section 1.5 the three desirable properties for an auction:

(G1) bidding truthfully is a dominant strategy and never leads to negative utility,
(G2) the resulting allocation should be economically efficient, and
(G3) the auction runs in (strongly) polynomial time.

However, in this chapter we aim to design an *ascending auction*, i.e., instead of a sealed-bid auction, where every buyer submits her complete valuation function as her bid, the buyers reveal information about their valuations incrementally over time via back-and-forth communication with the auctioneer, which yields more decision points where a buyer might act strategically. We can show that goal (G1) is too much to ask for, as a dominant strategy—let alone a truthful one—does not need to exist. Hence, we relax (G1) to

(G1*) bidding truthfully is an *equilibrium* strategy and never leads to negative utility.

However, to obtain any incentive guarantees it is clear that the auctioneer has to determine prices that buyers have to pay in order to disincentivize overbidding for items. Then we obtain an extensive form game $\Gamma = (N, (\Sigma_i)_{i \in N}, (u_i)_{i \in N})$, where the Σ_i will essentially be communication from buyer i to the auctioneer, whenever i is queried and $u_i(X, p) = v_i(X) - p$ is i's utility upon receiving X and paying p.

The aforementioned paper by Bikhchandani, de Vries, Schummer, and Vohra [Bik+11] introduces a *clock auction* (a special kind of ascending auction) that sells a base of the given matroid at Vickrey prices.

The main building block for showing that the ascending auction (described in Section 2.3) fulfills properties (G1*), (G2), and (G3) is Theorem 13 in [Bik+11] (or Theorem 2.8 in this thesis). It states that the auction computes a welfare-maximizing base (which yields (G2) immediately) and charges Vickrey prices (which is the key for proving (G1*)). This work provides a simpler proof of the theorem using only some matroid folklore lemmas.

It might not seem too obvious why the model described above is interesting, however, one can make up countless applications. Indeed, every combinatorial optimization problem that can be reformulated such that the objective is to just find a maximum weight base can be viewed as a market in which different agents assign weights to elements that are just interpreted as their maximum willingness to pay for that element. A typical setting where matroid constraints can arise is public procurement,[1] i.e., when a governing body purchases a contract over the provision of goods or service from some organization or company, which then becomes a representative for that governing body to provide those goods or services. Usually, those kind of contracts come with

[1] *Procurement* refers to the process by which an organization sources, negotiates, and purchases goods, works, or services from an external source. In that sense, the roles of buyer and seller are reversed. Our setting allows for such a process as well, by simply interpreting the price as a discount that is deducted from a previously known baseline price.

the guarantee that the the organization that won the contract will become an exclusive supplier of the particular goods or services as an exchange for lower prices (but also a guaranteed revenue as a sole provider of essential goods or services). To give a concrete but simple example, say a government wants to rebuild its highway network and offers private companies to purchase a contract that allows them to build, maintain, and operate the roads in exchange for the permission to charge tolls to drivers who want to use a road. In a small enough country, the highway network should roughly resemble a spanning tree to ensure that every road becomes essential for some journeys and there is no possible detour to avoid a road (and hence, giving the maintaining company some revenue via tolls).[2] Other examples include:

- scheduling jobs to machines: bidding that an agent's job gets processed on a fast machine (a partition matroid, [DGS86]) or before its due date (a classic example from the maybe best-known textbook on analysis of algorithms [Cor+22][3]),
- allocating homogeneous goods (a uniform matroid, [Aus04]),
- pair-wise kidney exchange: bidding that a patient-donor pair is part of a maximum matching to perform as many transplantations as possible (a matching matroid or a transversal matroid, [RSÜ05]),
- spatially distributed markets: bidding for identical (or very similar goods) in different locations such that transportation costs, tariffs, or embargoes have to be considered (a partition matroid, [BNP09]),
- bandwidth-markets: bidding for channels that are less noisy and more reliable against any fading issues (polymatroid, [TH02]),
- multi-class queueing systems: bidding for a higher priority in a queueing system (polymatroid, [SY92]).

Note that the latter two applications use polymatroids and hence, are a little more involved if one wants to apply the auction we present in this chapter. A naïve way to apply the auction is by applying Helgason's reduction [Hel72] mentioned at the end of Section 1.3. More recently, Raach and de Vries [RV25] describe a more direct way of applying the auction directly on the polymatroid instead of the associated matroid obtained by Helgason's reduction and thus, avoiding the pseudo-polynomial blow-up caused by the reduction.

[2] Privatization of roads and highways is widely unpopular in Germany but very much present in other countries of the world. For instance, in France roughly 80% of highway kilometers are under private concession.

[3] One could call it *scheduling matroid* but this term is not widely used. However, if we want to assign a type of matroid here that can model this scheduling problem, then a transversal matroid will do.

Finally, it is also possible to model markets that do not have any matroidal restrictions and in which buyers are competing for the same sets of items by simply creating for each item $e \in E$ and each buyer $i \in N$ a copy (e, i). Then we can find a market-clearing allocation (or a Walrasian equilibrium[4]) by simply building a partition matroid with partition $\{\{(e, i)\}_{i \in N}\}_{e \in E}$ and capacities 1 for each part. If buyers have more delicate valuation functions than additive ones considered in this chapter, we need more involved auctions, which will be clarified in Section 3.1.1.

2.1.1 Contribution

We provide a simplified proof the main theorem of Bikhchandani, de Vries, Schummer and Vohra [Bik+11] that states that an ascending auction computes Vickrey prices and returns a welfare-maximizing allocation. The original proof requires a very microscopic analysis of the auction: In every step, one has to keep track of deleted items and cocircuits that act as a witness for price-setting items, building an auction history, which is called *VCG sequence*. In contrast, our proof does not require this step-by-step view of the auction and instead only uses three matroid folklore lemmas which reduces the length of the proof significantly, is less technical and less index-heavy. The chapter as a whole, however, is still self-contained and includes a few more insights that are worth noting. We also give some more details about the issues of communication costs and privacy which are after all two crucial selling points for ascending auctions over sealed-bid auctions.

This chapter is based on the paper *A Simplified Analysis of the Ascending Auction to Sell a Matroid Base* which has been accepted for publication in the January 2026 volume of *Operations Research Letters* [PR26] and is also available as a preprint [PR24].

2.1.2 Related Work

There is a substantial body of literature on ascending auctions (or *dynamic* or *iterative* auctions in general), and over the course of this thesis, we touch different branches of this exciting part of mechanism design. In this subsection, we restrict ourselves to references that are closest to the present interest and our model. For a broad overview on dynamic auctions, we refer to Chapter 11 in [Nis+07] and [Par06b; BN05].

[4] See Chapters 3 and 4 for a very detailed treatment.

The VCG mechanism [Vic61; Cla71; Gro73] (cf. Theorem 1.31 in Section 1.5) is a cornerstone of mechanism design as it lays out a blueprint for incentive compatible mechanism in any conceivable environment with the objective of maximizing social welfare. It does not come without weaknesses though. While the potentially bad running time is not a result of the mechanism itself but just a result of the nominal welfare maximization problem, the fact that the VCG mechanism is a *direct* mechanism makes the costs of communication very high as buyers have to report their whole valuation function. Thus, mathematical economists and computer scientist were in search of *dynamic auctions* that implement the VCG mechanism in which buyers only have to reveal relevant parts of their valuation functions—which is also desirable with respect to privacy requirements. Multiple auctions that implement the VCG mechanism in a dynamic setting exist [Aus04; Aus06; COP15; VSV07; Bik+01; MP07; SY14]. Typically though, these auction have multiple price trajectories or involve a proxy auctions that are a bit strange to follow. This is in contrast to the paper by Bikhchandani, de Vries, Schummer, and Vohra [Bik+11] whose auction we consider in this chapter.

The VCG mechanism additionally has some drawbacks in terms of revenue generation, which on one hand can be very low and is also non-monotone in the buyers' bids. Moreover, while the mechanism itself is strategy-proof (truthful bidding is a dominant strategy), it is vulnerable to collusion. Finally, even though the payment rule of the VCG mechanism sounds intuitively fair (*players pay the utility loss of the other buyers incurred by their participation*), it can be considered unfair as buyers might have to pay two different prices for the same item [AM06].

2.2 Background Theory

In Section 1.3 we gave a primer on matroid theory. There are a few statements that we did not mention there as they are quite tailored for the present interest in this chapter. An integral part of the auction that is introduced in the next section is that the auctioneer has to keep track of the cocircuits of the given matroid and its minors that appear during the auction. In Section 1.3 we introduced cocircuits of a matroid as the *circuits of its dual*, however, while it remains helpful to think of cocircuits this way, a neat part of this analysis is that matroid duality is not really relevant to follow the analysis of the auction (or at least, can be hidden without leaving black boxes).

There is an equivalent *primal* definition of what a cocircuit is, which allows us to omit the concept of the dual matroid in the sequel.

Proposition 2.1. *Let $M = (E, \mathcal{B})$ be a matroid. Then C^* is a cocircuit of M if and only if it is an inclusion-wise minimal set with the property that $C^* \cap B \neq \emptyset$ for all $B \in \mathcal{B}$. Moreover, for all $e \in C^*$ there exists a base $B \in \mathcal{B}$ such that $C^* \cap B = \{e\}$.*

Proof. Let $C^* \in \mathcal{C}^*$ and $B \in \mathcal{B}$. As C^* is a circuit of M^*, there is no proper subset of C^* which is a cocircuit. If $C^* \cap B = \emptyset$, then $E \setminus C^* \supseteq B$ and hence $C^* \in \mathcal{I}^*$, contradicting the fact that C^* is a circuit of M^*.

Now let $e \in C^* \in \mathcal{C}^*$. Assume for all $B \in \mathcal{B}$, $B \cap C^* \supset \{e\}$, then $(C^* - e) \cap B \neq \emptyset$. As $C^* \in \mathcal{C}^*$, $C^* - e \notin \mathcal{C}^*$, i.e., $C^* - e \in \mathcal{I}^*$. Thus, $E \setminus (C^* - e) = (E \setminus C^*) + e$, contains a base $B' \in \mathcal{B}$ and hence, $(C^* - e) \cap B' = \emptyset$. □

Conveniently, this description of cocircuits gives us a very natural interpretation of why the auctioneer has to keep track of them during the auction. We will follow-up on this in the next section.

Another useful result is due to Brualdi [Bru69], who showed that the base exchange property of matroids (cf. Axiom (B2) from Proposition 1.10) can be strengthened to allow for a simultaneous exchange between two bases.

Proposition 2.2 (Strong Base Exchange [Bru69, Theorem 2])**.** *Let $M = (E, \mathcal{B})$ be a matroid and $B, B' \in \mathcal{B}$. Then for all $e \in B \setminus B'$, there exists some $f \in B' \setminus B$ such that $B - e + f \in \mathcal{B}$ and $B' - f + e \in \mathcal{B}$.*

Before we consider the ascending auction, we remark that there exists a *sealed-bid* auction which runs in strongly polynomial time for this model, namely the direct *VCG mechanism* (cf. Theorem 1.31): the auctioneer starts by collecting the bids $b(e) \in \mathbb{Z}_+$ from every buyer $i \in N$ for every item $e \in E_i$, which may or may not be equal to her true valuation $v_i(e)$. Based on the resulting bid vector $b = (b(e))_{e \in E}$, the auctioneer first computes a base $\hat{B} \in \mathcal{B}$ of maximum b-weight. To compute $\hat{B}$, the auctioneer might use the Matroid Greedy Algorithm as described in Section 1.3. Afterwards, the auctioneer computes for every buyer $i \in N$ the Vickrey price, which incentivizes truthful bidding (cf. [Vic61; Cla71; Gro73] and Section 1.5). In our setting the Vickrey price of buyer $i \in N$ is

$$\hat{p}_{\hat{B}}(i) = \max_{B \in \mathcal{B} \setminus E_i} b(B) - b(\hat{B} \setminus E_i). \tag{2.1}$$

Since for every $i \in N$ a base of maximum b-weight in $M \setminus E_i$ can be computed with the Matroid Greedy Algorithm, the auction runs in strongly polynomial time. As already argued in the introduction, we prefer an ascending auction over a sealed-bid auction, since an ascending auction is more transparent and requires less information from the buyers.

2.3 The Auction

Recall that we consider an auction in which an auctioneer is selling a finite set of distinguishable, indivisible items E to a finite set of buyers N. We assume that each buyer $i \in N$ is interested in buying only the items in a subset $E_i \subseteq E$, and that each buyer $i \in N$ has an additive valuation function, i.e., they have a non-negative integral valuation $v_i(e) \in \mathbb{Z}_+$ for each item $e \in E_i$ and for each subset $X \subseteq E_i$, we write $v_i(X) = \sum_{e \in X} v_i(e)$. In the auction we consider, the auctioneer is constrained to sell a base of a given matroid $M = (E, \mathcal{I})$ on E, where $\mathcal{I}$ denotes the collection of independent sets. We may assume that E only contains those items for which at least one buyer is interested in buying them. Notice that we can assume without loss of generality that no two players are interested in buying the same item. If not, say if $e \in E_i \cap E_j$ for some item $e \in E$ and two buyers $i \neq j$, we can replace e by two parallel copies e^i and e^j, and replace $M = (E, \mathcal{I})$ by $M' = (E', \mathcal{I}')$ with ground set $E' = E - e \cup \{e^i, e^j\}$ (and $E'_i = E_i - e + e^i, E'_j = E_j - e + e^j$) and independence system $\mathcal{I}'$, where $\mathcal{I}' = \{I \in \mathcal{I} \mid e \notin I\} \cup \{I - e + e^i, I - e + e^j : I \in \mathcal{I}, e \in I\}$. This construction extends naturally to the case when even more than two buyers are interested in the same item.

In other words, we can assume that $\{E_i\}_{i \in N}$ is a partition of E. Thus, an instance of our auction is defined by a matroid $M = (E, \mathcal{I})$ on a finite set E partitioned into $\{E_i\}_{i \in N}$, and a valuation function $v: E \to \mathbb{R}_+$ with $v(e) = v_i(e)$ for the (unique) buyer $i \in N$ with $e \in E_i$. We furthermore assume that none of the buyers initially is a monopsonist in the sense that no cocircuit of M is contained in any E_i. Otherwise, we could just resolve the monopsonies by selling an item from the monopsony (buyer's choice) for price 0 in the same way we resolve monopsonies later in the auction (see below).

The paper [Bik+11] describes and analyses an ascending auction that meets the three mechanism design goals to (i) incentivize truthful signals, (ii) compute a welfare-maximizing base, and (iii) run in strongly polynomial time. In this section, we give a slightly

modified but equivalent presentation of the auction introduced in [Bik+11]. An analysis of the auction, which is significantly simpler than the one in [Bik+11], is presented in Section 2.4.

Sketch of the ascending auction At any point of the auction, we can partition the items in E into three parts: the set of deleted items D, the set of contracted items I, and all remaining items $E \setminus (D \cup I)$. Intuitively, items in D will remain unsold, the items in I are already sold, and for all remaining items are still active in the auction, which may or may not be sold later. A key ingredient of the auction is that the auctioneer has to keep track of the cocircuits of the currently observed matroid $M \setminus D / I$ (a minor of the original matroid M). As cocircuits have a non-empty intersection with every base, the auctioneer has to make sure that he sells at least one item of each cocircuit in order to sell a base. The idea now is that the auctioneer increases the price for all items simultaneously. At any time, a buyer j has the option to announce that there is an item $f \in E_j$ that she would not buy if the price (currently p) would increase further. She may also announce this for multiple items of her set and other buyers may do so at the same time. In the sequel, we refer to those items as *critical* at price p. Then, the auctioneer removes critical items from the matroid one after another. This removal will be carried out as a *deletion* in the matroid. Now as every item occurs in some cocircuit (unless it is a loop) and the cocircuits of the restricted matroid are the same up to removal of the deleted item (see Lemma 2.6 below), some cocircuits decrease in size. At some point there will be a cocircuit C^* that is completely contained in some buyer's item set E_i; in that case, we call C^* a *monopsony*. Note that in [Bik+11], they refer to such a cocircuit as a *monopoly*, which is just as accurate in case of a reverse auction or procurement. Whenever a monopsony $C^* \subseteq E_i$ occurs, the auctioneer asks the buyer i (the monopsonist) to name her most valuable item e from C^*. This announced item e will then immediately be sold to buyer i at price p (in the words of Ausubel [Aus04], buyer i has *clinched* an item). After item e is sold, the auctioneer removes e from the matroid. This removal, however, will be performed as a *contraction*. Note that a deletion of an item f might create multiple monopsonies (up to $n - 1$) which will all be resolved with the aforementioned procedure. After that the auctioneer resumes to deleting critical items until he either recognizes another monopsony (he then proceeds as above) or he runs out of critical items (he then increases the price of all remaining items by 1). Note that it might very well happen that an item that is announced as critical at price p still gets sold at price p if there are multiple items of the same cocircuit

announced as critical at the same time. In that case this item will just be skipped by the auctioneer in the deletion loop.

As a nice gimmick of this process, we also have a price for every individual item that is sold during the auction. This is in contrast to the direct VCG mechanism which only returns a total for every buyer.

The pseudocode Ascending Matroid Auction below gives a precise description of the auction. We amended the original pseudocode of [Bik+11] only in the sense that we left out the tie-breaking (which is arbitrary anyway) and perform the deletion steps right away instead of after resolving all monoposonies that would have occurred.

Ascending Matroid Auction: $M = (E, \mathcal{I})$ monopsony-free, partition $\{E_i\}_{i \in N}$ of E

```
1  I := ∅, p := 0, p̂(e) := 0 for all e ∈ E
2  while I ∉ B do
3      p := p + 1
4      R(p) := {f ∈ E | f announced as critical at price p}
5      for f ∈ R(p) ∖ I do
6          M := M \ f
7          while M contains a monopsony C* ⊆ E_i of some buyer i do
8              e := item from C* that buyer i chooses to buy now at price p
9              I := I + e
10             p̂(e) := p
11             M := M / e
12 return B̂ := I and p̂
```

Initially, the auctioneer only knows the matroid $M = (E, \mathcal{I})$ (that he can access via an independence oracle) and the partition $\{E_i\}_{i \in N}$. For all further information, the auctioneer has to receive *signals* from the buyers, i.e., answers to queries about their valuation functions (see paragraph on truthful signaling below). As usual in matroid optimization, we assume that the auctioneer has access to an independence oracle. That is, he might query for any set $X \subseteq E$, whether $X \in \mathcal{I}$ in time $\mathcal{O}(1)$. Note that, using an independence oracle, the auctioneer can efficiently check whether the conditions in the two **while** loops are satisfied.

Lemma 2.3. *A query to a base oracle can be simulated with $\Theta(m)$ independence oracle queries.*

Proof. Given a set X, we want to determine whether $X \in \mathcal{B}$ but only may query *"Is $Y \in \mathcal{I}$?"*. We first do a query *"Is $X \in \mathcal{I}$?"*. If the answer is `false`, then we can already terminate with `false`.[5] Otherwise, we iterate through all $e \notin X$ and query *"Is $X+e \in \mathcal{I}$?"*. If the answer is `true` for any of those, then we again can return `false` as X was not maximal independent. Otherwise, if all those queries were answered with `false`, then we answer with `true`. In total, there are $1+|E \setminus X| \in \Theta(m)$ queries to the independence oracle to answer the base oracle query. □

Lemma 2.4. *The auctioneer can identify a monopsony of single buyer with $\Theta(m)$ independence oracle queries.*

Proof. We describe how the auctioneer can determine a monopsony of buyer i (or verify that she does not have one) at a stage where items in D are deleted and items in I are contracted. By Proposition 2.1, to check whether E_i contains a cocircuit of $M \setminus D / I$, it suffices to check whether E_i contains items that are in every base of $M \setminus D / I$.

The auctioneer runs the Matroid Greedy Algorithm on $M \setminus D / I \setminus E_i$ to determine a base B of the remaining items except the ones of buyer i.[6] This requires $|E \setminus (D \cup I \cup E_i)|$ queries. After that the auctioneer queries the independence oracle for every $e \in E_i \setminus (I \cup D)$, *"Is $B + e \in \mathcal{I}$?"*. Every time the oracle answers `true`, the item e is part of the monopsony, otherwise it is not. If all those queries are answered with `false`, then i is not a monopsonist. These are $|E_i \setminus (D \cup I)|$ additional queries, i.e., a total of $|E \setminus (D \cup I)| \in \Theta(m)$. □

It is worth pointing out that if we somehow have access to the independence oracle of the dual matroid M^*, i.e., the coindependence oracle of M, then instead of querying all items of the current minor to find one monopsony within one buyer's set E_i, it suffices to check the items in E_i. So in total $\Theta(m)$ queries to the coindependence oracle suffice to find *all* monopsonies of $M \setminus D / I$. If the matroid has a compact description, i.e., an encoding that has polynomial size in $|E|$ (which indeed is the case for all the matroids we introduced in Section 1.3.2), then it is also conceivable to determine the dual description of the matroid and only work with this (handling contractions and deletions in reverse). In that case the initial effort to convert the matroid description to its dual is amortized over the course of the auction and more efficient. Note that this

[5] Note that this step is not necessary in our case as the Ascending Matroid Auction maintains an independent set.

[6] Mind that D and I both can contain items in E_i. In that case we assume the operations to be performed from left to right, i.e., only the remaining items in E_i get deleted.

only holds if the class of the dual matroid is also closed under deletion and contraction (which is, for instance, not the case for gammoids and transversal matroids). The details can be found in the bachelor's thesis of Kempkes [Kem25, Chapter 4].

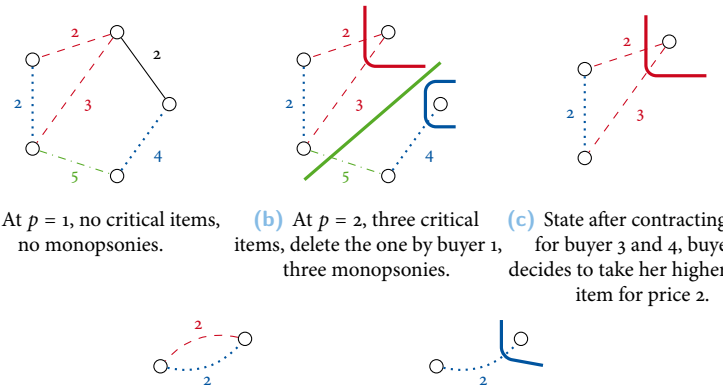
Truthful signaling We emphasize that the communication between auctioneer and buyers takes place only in lines 4 and 8. We say that buyer i is *signaling truthfully* if her signals to the auctioneer are consistent with v_i in the following sense: If buyer i is asked to report on her critical items in line 4, she truly announces those items e as critical for which $v_i(e) = p$. Furthermore, whenever the auctioneer asks buyer i for her most-valued item in her monopsony $C^* \subseteq E_i$ in line 8, she truly selects the item of largest v-value in her monopsony C^*.

Example 2.1. Consider the graphic matroid in Figure 2.1 with edges labeled by its buyer's valuation. At price $p = 1$, there are no critical items (Figure 2.1a). Hence, the auctioneer raises the price of all items to $p = 2$ at which multiple items become critical. After deleting one of those critical items, the auctioneer observes that three buyers now hold a monopsony (Figure 2.1b). While two of those monopsonies are singletons, one buyer still has to signal to the auctioneer which item in her cocircuit she values the most (Figure 2.1c). Note that in this example untruthful signaling would be a strategy that is inconsistent with every possible valuation function since one of the two items was announced as critical while the other one was not). After contracting all best elements from each monopsony, the auctioneer is left with two more parallel elements, both of which are critical, so he performs another deletion (Figure 2.1d) resulting in another singleton monopsony (Figure 2.1e) which can be contracted.

2.4 Analysis

Recall the three mechanism design goals (G1*), (G2), and (G3) described in the introduction. Computational efficiency, i.e., property (G3), is clearly fulfilled: The running time of the auction scales polynomially in the number of items and buyers. It even runs in strongly polynomial time by using a long-step variant that only requires a reasonable consistency check on the signals (see [Bik+11] and [RV25] for the polymatroid case). That is, by replacing Line 3 in Ascending Matroid Auction with

3' | $p := \min_{i \in N} \{ q_i \mid \text{smallest } q_i > p \text{ for which } i \text{ has a critical item} \}$



(a) At $p = 1$, no critical items, no monopsonies.

(b) At $p = 2$, three critical items, delete the one by buyer 1, three monopsonies.

(c) State after contracting items for buyer 3 and 4, buyer 2 decides to take her higher value item for price 2.

(d) Delete critical item of buyer 2.

(e) Buyer 3 has now a monopsony, sell her the item.

Figure 2.1: Example iteration (see Example 2.1) on a graphic matroid with four buyers 1 (black, solid), 2 (red, dashed), 3 (blue, dotted), 4 (green, dotdashed): For $p = 0$ and $p = 1$ there are no critical items. When p increases to 2 multiple items get critical and trigger some sales, including some of critical items.

and the understanding that in Line 4 every buyer who minimized the term in Line 3' has to announce at least one item as critical, we obtain a long-step auction that results in the same outcome.

In this section, we give a simplified analysis of the auction that shows that the auction returns a welfare-maximizing base (property (G2)) and charges Vickrey prices. As it turns out, this (together with an observation on inconsistent signals) is sufficient to show that truthful signaling is an ex post equilibrium (see Section 2.6 below), i.e., the auction also fulfills (G1*).

The keys to show that the Ascending Matroid Auction yields a welfare maximizing allocation (under truthful signals) and assigns Vickrey prices are a few statements about cocircuits and how they evolve by taking minors of the matroid.

In essence, we will need the following three matroid folklore lemmas. Since the proofs of these lemmas are not always easy to find (including Dawson's Lemma, which we do not state verbatim below), we give their proofs here.

Lemma 2.5 (Cocircuits after deletion)**.** *Let C^* be a cocircuit of M and $e \in E$. Then $C^* - e$ is a union of cocircuits of $M \setminus e$.*

Proof. We prove the following stronger statement (see [Oxl06], Exercise 2 in Section 3.1):

Let $C \in \mathcal{C}$ and $e \in E$.

(a) If $e \in C$, then $\{e\} \in \mathcal{C}$ or $C - e \in \mathcal{C}(M / e)$.
(b) If $e \notin C$, then C is a union of circuits of $\mathcal{C}(M / e)$.

Our statement follows as the dual statement by taking deletions in the cocircuit set instead of contractions in the circuit set and the understanding that the empty set is an empty union of cocircuits if e was a coloop and a single cocircuit is the union over a singleton.

(a) Let $e \in C \in \mathcal{C}$. If $C = \{e\}$, there is nothing to show. Otherwise, consider M / e. Assume $C - e \notin \mathcal{C} / e$, i.e., $C - e \in \mathcal{I} / e$ or $C - e$ properly contains a circuit of M / e. As $\mathcal{I} / e = \{I \subseteq E - e \mid I + e \in \mathcal{I}\}$, the former cannot be true, so assume the latter case. Let $C' \subset C - e$ with $C' \in \mathcal{C} / e$. But then $C' + e \subset C$ and hence, $C' + e \in \mathcal{I}$ thus, $C' \in \mathcal{I} / e$.

(b) Let $e \notin C \in \mathcal{C}$. If $C \in \mathcal{C} / e$, we are done. Otherwise, note that $C \notin \mathcal{I} / e$ as the contrary would imply $C + e \in \mathcal{I}$ and hence, $C \in \mathcal{I}$. To show that C does not contain any f outside of a circuit of M / e with other items from C, we show that

$$\rho_{M/e}(C) = \rho_{M/e}(C - f)$$

for all $f \in C$. Suppose not, i.e., $\rho_{M/e}(C) = \rho_{M/e}(C - f) + 1$ for some $f \in C$. Then using the identity from Proposition 3.1.6 in [Oxl06]

$$\rho_{M/Z}(X) = \rho(X \cup Z) - \rho(Z)$$

we also have that

$$\begin{aligned} & \rho(C + e) - \rho(e) = \rho(C - f + e) - \rho(e) + 1 \\ \iff \quad & \rho(C + e) = \rho(C - f + e) + 1, \end{aligned}$$

which implies with submodularity and $|\{f\}| = 1$ that $\rho(C) = \rho(C - f) + 1$ and hence, $C \notin \mathcal{C}$. □

Lemma 2.6 (Cocircuits before deletion)**.** *Let C^* be a cocircuit of $M \setminus e$. Then C^* or $C^* + e$ is a cocircuit of M.*

Proof. Clearly, at most one can be true. As $C^* \in \mathcal{C}^*(M \setminus e)$, we know that $C^* \cap B \neq \varnothing$ by definition for all $B \in \mathcal{B}(M \setminus e)$. Thus, $E \setminus C^*$ does not contain a base of $M \setminus e$. Also $E \setminus (C^* + e)$ does not contain a base of M: Suppose it would, i.e., $B \subseteq E \setminus (C^* + e)$ with $B \in \mathcal{B}$, then $B \in \mathcal{I}$ and hence, $B \in \mathcal{I}(M \setminus e)$ as $e \notin B$. Now assume $E \setminus (C^* \setminus T)$ does not contain a base B for some maximal $\varnothing \subset T \subseteq C^*$, making $\tilde{C}^* := C^* \setminus T \subset C^*$ a cocircuit in $M \setminus e$. But then $\tilde{C}^* \in \mathcal{C}^*(M \setminus e)$, which contradicts the fact that $C^* \in \mathcal{C}^*(M \setminus e)$. □

Lemma 2.7 (Augmentation by cocircuit, [Daw80, Theorem 1])**.** *Let I be an independent set of a matroid M that can be extended to a maximum weight base. Let C^* be any cocircuit of M / I and $e \in C^*$ its maximum weight element. Then $I + e$ is independent and can be extended to a maximum weight base.*

Proof. Let $\hat{B}$ be a base of maximum weight with $I \subseteq \hat{B}$ with respect to weight function w. If $e \in \hat{B}$, we are already done. Otherwise, choose $B_0 \in \mathcal{B} / I$ with $B_0 \cap C^* = \{e\}$. Such a B_0 always exists as C^* is by definition an inclusion-wise minimal set in $E \setminus I$ that

intersects every base of M / I. Now construct a base of M by taking $B = B_0 \cup I$ and observe that $B \cap C^* = \{e\}$ holds as well (as $C^* \subseteq E \setminus I$), which means that C^* is also a cocircuit of M. Since $e \in B \setminus \hat{B}$, by the strong base exchange property (Proposition 2.2), there exists some $f \in \hat{B} \setminus B$ such that both, $\hat{B} - f + e \in \mathcal{B}$, and $B - e + f \in \mathcal{B}$. Note that $f \notin I$, since $f \in \hat{B} \setminus B \subseteq \hat{B} \setminus I$. Then it also follows that $f \in C^*$, as otherwise $(B_0 - e + f) \cap C^* \subseteq (B - e + f) \cap C^* = \varnothing$, contradicting the fact that C^* is a cocircuit of M and M / I. Since $\hat{B}$ is a base of maximum weight, the weight of base $\hat{B} - f + e$ cannot be larger; thus, $w(e) \leq w(f)$. However, since $e, f \in C^*$ and e is max-weight element of C^*, we have $w(e) = w(f)$, implying that $\hat{B} - f + e$ is a max-weight base containing $I + e$. □

Now let us state the main theorem of [Bik+11] and prove it in a new and shorter way.

Theorem 2.8 ([Bik+11, Theorem 13])**.** *The Ascending Matroid Auction returns a welfare-maximizing allocation and assigns Vickrey prices.*

Proof. For the proof, we assume truthful signals from the buyers. The discussion in Section 2.6 shows why this is justified.

First, we show that the resulting allocation $\hat{B}$ is welfare-maximizing. For that it suffices to show that I, the set of sold items in the Ascending Matroid Auction, is max-weight base extendable throughout the whole auction. Since we start with $I = \varnothing$, the statement is obviously true in the beginning. Given some stage of the auction with sold items I and deleted items D, we consider the minor $M \setminus D / I$. Say $e \in E \setminus (D \cup I)$ is the next item to be added to I. This means e is an item of maximum value in some monopsony C^*, i.e., a cocircuit of $M \setminus D / I$. Thus, by applying Lemma 2.6 repeatedly, we have that $\tilde{C}^* = C^* \cup D'$ is a cocircuit of M / I for some $D' \subseteq D$. As $v(e) \geq v(f)$ for all $f \in D$ (and in particular D'), e has also maximum value in $\tilde{C}^*$. Hence, by Lemma 2.7, $I + e$ is max-weight base extendable.

Now let us show that the auction also charges Vickrey prices to the buyers. For each $e \in \hat{B}$, let f_e denote the item which is deleted before e was added to I, i.e., f_e is the price-setting item for e and we have $p(e) = v(f_e)$. We need to show that for all $i \in N$

$$\hat{B}' = \hat{B} \setminus E_i \cup \{f_e : e \in \hat{B} \cap E_i\}$$

is a max-weight base of $M \setminus E_i$. Note that a deletion of an item f can create at most one monopsony per buyer: If C_1^*, C_2^* are both monopsonies of buyer j, then $C_1^* + f$

and $C_2^* + f$ are both cocircuits in the matroid before deleting f and hence, by the cocircuit exchange property, there exists a cocircuit $C_3^* \subseteq (C_1^* \cup C_2^*) - f$ which would be a monopsony of j before deleting f. That is, $\hat{B}'$ has the correct cardinality to be a base of $M \setminus E_i$ (recall that initially there is no monopsony in M). Thus, we only need to show that $\hat{B}'$ is indeed welfare-maximizing and independent $M \setminus E_i$. We again use Lemma 2.7 to show that all elements $e \in \hat{B} \setminus E_i$ are still suitable for the new base and that for all $e \in \hat{B} \cap E_i$, its corresponding f_e is a suitable replacement. Consider some $e \in \hat{B}$ and the set I before e was added. Then e was an item of maximum value in some monopsony C^* of $M \setminus D / I$. We also have $v(e) \geq v(f)$ for all $f \in D$. Thus, e also has maximum value in $\tilde{C}^* = C^* \cup D'$, a cocircuit of M / I with $D' \subseteq D$ by applying Lemma 2.6. Moreover $f_e \in D'$ as otherwise C^* would have been a monopsony before f_e was deleted. From here, we make a case distinction:

Case 1: $e \notin E_i$. By applying Lemma 2.5 on M / I w.r.t. all the items in E_i, we also have that e has maximum value in some cocircuit of $M / I \setminus E_i = M \setminus E_i / I$ and hence, by Lemma 2.7, $I + e$ is max-weight base extendable for $M \setminus E_i$.

Case 2: $e \in E_i$. Deleting E_i (including e) yields the minor $M / I \setminus E_i = M \setminus E_i / I$. Note that $v(f_e) \geq v(f)$ for all $f \in D$ as f_e was deleted last. Applying Lemma 2.5 on M / I w.r.t. all items in E_i yields that there is a cocircuit $\check{C}^* \subseteq \tilde{C}^*$ of $M \setminus E_i / I$ with f_e as item of maximum value. So by Lemma 2.7, we have that $I + f_e$ is max-weight base extendable in $M \setminus E_i$. □

2.5 No Dominant Strategy

In the beginning of this chapter, we mentioned that we need to relax our auction design goal (G1) to (G1*) as we cannot expect, in an ascending auction, to obtain an auction that has truthful signaling as a dominant strategy. In fact, there might not even exist a dominant strategy at all. This is already pointed out in [Bik+11] but without a concrete example. Below we provide a small example how buyers can react differently on events within the auction such that truthful signaling might not be a best response (and hence, not a dominant strategy).

A strategy in the Ascending Matroid Auction is a mapping that returns an action for each possible information set a buyer might have at any of her vertices in the corresponding extensive form game which we sketched in Figure 2.2.

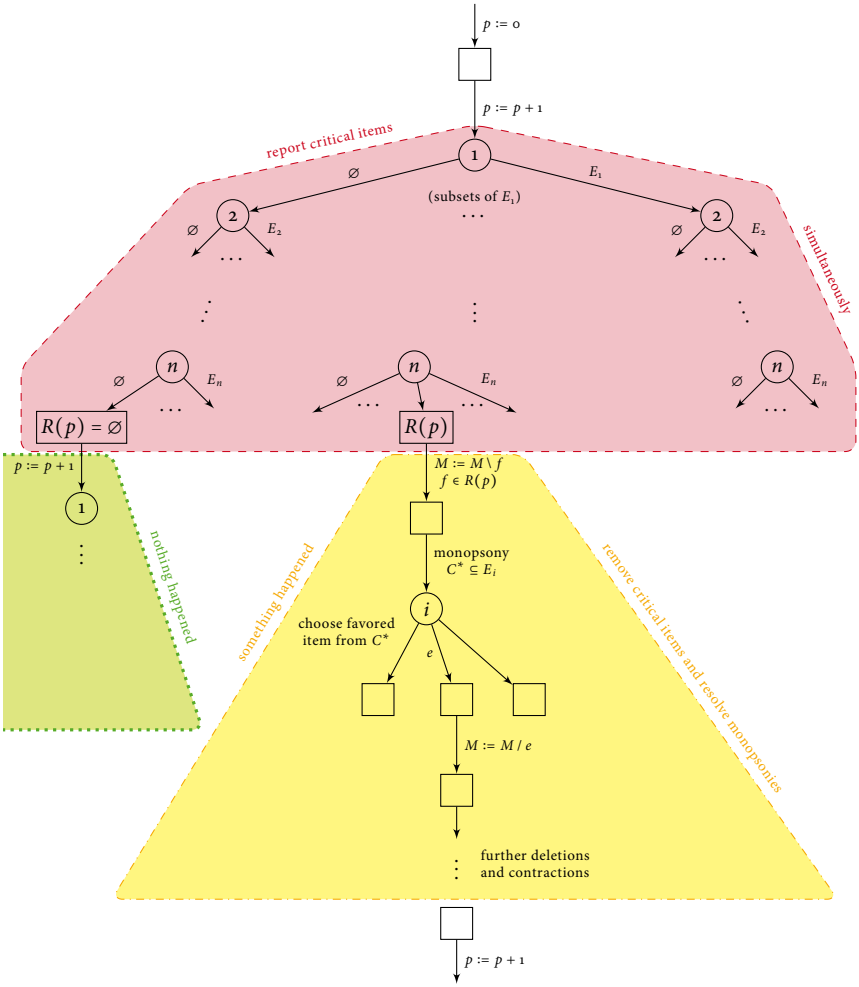


Figure 2.2: The Ascending Matroid Auction as game in extensive form. The auctioneer's vertices are square and his actions are completely determined by the pseudocode of Ascending Matroid Auction (with arbitrary tie-breaking). The round vertices are labeled with the buyer that gets to do an action. The reporting of critical items happens simultaneously, i.e., all vertex labeled i in such a phase (red, dashed) have the same information set. If all buyers report $\varnothing$ as critical, the auctioneer increases the price and we start over (green, dotted—this phase is just as the red one again). If $R(p)$ is not empty (yellow, dashdotted), the auctioneer starts deleting items until he spots a monopsony which trigger further communications with the respective monopsonist and contractions according to her answer.

Example 2.2. Consider the market with two buyers $N = \{1, 2\}$ and four items $E = \{e^1, e^2, f^2, f^2\}$. We have the valuations $v_i = (v_i(e^i), v_i(f^i))$ are $v_1 = (2, 4)$ and $v_2 = (3, 3)$. In Figure 2.3 you find the graphic matroid on E (which actually also can be viewed as a partition matroid). Let us denote by τ_1 the *truthful* strategy of buyer 1, i.e., announcing e_1 as critical at price 2 and f_1 at price 4, independent of whatever buyer 2 does. Also note that there will never be a monopsony containing both e_1 and f_1, so if buyer 1 ever becomes a monopsonist, her choice of item is unique. If buyer 2 also signals truthfully (call this strategy τ_2), then by the VCG Theorem τ_1 is a best response. However, there is a second strategy that yields the same outcome against τ_2: Buyer 1 can use an *altruistic* strategy against τ_2 (call this strategy α_1) by instead of announcing e_1 as critical at price 2, she already does so at price 1 (but signaling truthfully from there onward), which yields a lower price for e_2 for buyer 2. Note that α_1 and τ_1 are the only two best responses against τ_2 (except for the ones that involve overbidding on f_1, which are clearly not dominant either), so if there is a dominant strategy, it has to be one of those two.

Buyer 2 on the other hand can also incentivize the altruistic signal of buyer 1 by in turn also signal altruistically by her strategy α_2, which works as follows:

- If buyer 1 announces e_1 as critical at price 1, she announces f_2 as critical at price 2 (giving buyer 1 a better price on f_1).
- If buyer 1 announces e_1 as critical only at price 2 (truthfully), she announces f_2 as critical at price 3 (truthfully).

In this case α_1 is a better response against α_2, which means that τ_1 is not dominant.

However, buyer 2 might also react *spiteful* against α_1 (call this strategy σ_2), which works as follows:

- If buyer 1 announces e_1 as critical at price 1, she announces f_2 as critical at price 4 (giving buyer 1 a worse price on f_1).
- If buyer 1 announces e_1 as critical only at price 2 (truthfully), she announces f_2 as critical at price 3 (truthfully).

Here we have that τ_1 is a better response against σ_2 than α_1, so α_1 is not dominant either and hence, there cannot be a dominant strategy at all.

In essence, the problem for buyer 1 is that she does not know for her first decision point whether buyer 2's reaction will be altruistic or spiteful towards her first signal, which makes neither of her possible strategies dominant.

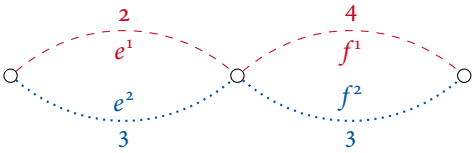


Figure 2.3: Graphic matroid from Example 2.2 with two buyers 1 (red, dashed) and 2 (blue, dotted) where signaling truthfully is not a dominant strategy for buyer 1.

2.6 Truthful Equilibrium

The previous section shows that—unlike the direct VCG mechanism—the Ascending Matroid Auction truthful signaling generally is not a dominant strategy (as there might be no dominant strategy at all). However, we can show that truthful signaling is an equilibrium strategy. More precisely, the state in which every buyer signals truthfully is an ex post Nash equilibrium, which is still a very strong incentive guarantee (albeit not as strong as an equilibrium in dominant strategies) as probabilistic beliefs that buyers might have or not have about the valuation functions of others are irrelevant. Even under the assumption that a buyer might have full knowledge about the other buyers' valuations, truthful signaling is still without regret if those buyers also signal truthfully. In other words, if we fix a buyer i and assume that all buyers but i signal truthfully, then i's best response is to signal truthfully as well. The proof follows the *Revelation Principle* (cf. [Mye81] and Theorem 1.32 in Section 1.5).

To illustrate the proof, let us first look at a sealed-bid proxy auction: at the beginning of the auction, all buyers submit a bid $b_i \colon E_i \to \mathbb{R}_+$ to a proxy buyer (e.g., the auctioneer himself), who does not know whether b_i is truthful or not. Then the proxy buyers answer all the requests of the auctioneer in Lines 4 and 8 of the Ascending Matroid Auction consistently with b_i. As we have shown, the Ascending Matroid Auction will now return an optimal allocation (w.r.t. the b_is) and Vickrey prices. By the VCG Theorem, the dominant strategy for each buyer is to submit her true valuations to her proxy buyer. The proxy auction is sketched in Figure 2.4.

The key difference between the Ascending Matroid Auction and the described proxy auction is that in the latter the buyers are submitting a valuation (truthful or not) and stick with it for the entire auction, while in the former they can in principle answer requests of the auctioneer in a way that is inconsistent in the sense that the signals

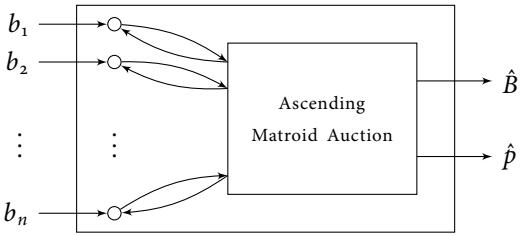


Figure 2.4: Sketch of the proxy auction.

cannot be represented by any valuation function. This kind of strategic behavior could be beneficial to a buyer because she can react to events during the auction.

A signaling strategy σ_i is called *consistent* if there is a valuation function v_i such that truthful buyer i with valuation v_i would signal according to σ_i. Given a valuation function v_i, we denote by $\gamma_i(v_i)$ the (unique) signaling strategy that is consistent with v_i.

First we reprove the following lemma from [Bik+11] that states that among consistent strategies, truthful ones are best provided that everyone else is truthful.

Lemma 2.9 ([Bik+11, Lemma 15]). *Let $\tau = (\tau_1, \dots, \tau_n)$ be the state in which every buyer signals truthfully (i.e., w.r.t. valuations $v_i \colon E_i \to \mathbb{Z}_+$ for all $i \in \mathbb{N}$) in every iteration of the Ascending Matroid Auction. Let σ_i be the signals of some $i \in N$ consistent with some false valuation $v_i' \colon E_i \to \mathbb{Z}_+$ in response to τ_{-i}. Then $u_i(\tau) \geq u_i(\sigma_i, \tau_{-i})$.*

Proof. If all buyers $i \in N$ play consistently according to their true valuation function v_i, then by Theorem 2.8, the Ascending Matroid Auction computes an allocation, i.e., a base $\hat{B}$, that maximizes welfare, i.e., $\sum_{i \in N} v_i(\hat{B} \cap E_i)$ and charges Vickrey prices $p_{\hat{B}}(i)$ w.r.t. these valuations. By Theorem 1.31, we have that a deviation to a valuation function v_i' of one buyer $i \in N$ and consistent signals w.r.t. this false valuation yield to an allocation and Vickrey prices corresponding to this false valuation. As the direct VCG mechanism is incentive compatible, buyer i cannot increase her utility by this deviation. □

However, this lemma alone is not sufficient to conclude that truthful signaling is an equilibrium in the Ascending Matroid Auction since we only considered deviations to signaling schemes that are consistent w.r.t. some valuation. We still need to show that signals that are inconsistent with any possible valuation function also cannot improve a buyer's utility. For an example of such inconsistent signaling, see Figure 2.5.

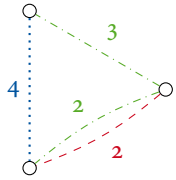


Figure 2.5: At price $p = 2$ two items are (truthfully announced) as critical: one by buyer 1 (green, dotdashed), one by buyer 2 (red, dashed). If buyer 2's item gets deleted first by the auctioneer, player 1 has a monopsony and could now announce her critical item as best, which would be inconsistent with the information that her other item is not critical.

The following lemma shows that every inconsistent strategy can be replaced by a consistent strategy that yields the same outcome in the Ascending Matroid Auction. The proof is also due to [Bik+11] and just restated here to give a complete and instructive presentation.

Lemma 2.10 ([Bik+11, Lemma 16]). *Let $\sigma = (\iota_i, \gamma_{-i})$ be a state in which buyer i plays an inconsistent signaling strategy ι_i and all other buyers $j \in N \setminus \{i\}$ signal consistently via γ_j for a valuation v_j (truthful or not). Then there exists a valuation function v_i^σ such that $\sigma' = (\gamma_i(v_i^\sigma), \gamma_{-i})$ achieves the same allocation and prices in the Ascending Matroid Auction.*

Proof. Suppose i is awarded the items $B_i = \{e_1, \ldots, e_q\} \subseteq E_i$ at prices $p_{e_1} \leq \ldots \leq p_{e_q}$. Moreover, let p_f be the price for each $f \in E_i \setminus B_i$ in which i reported f as critical. We define a valuation function v_i^σ for buyer i that yields the same outcome if i signals consistently according to this valuation.

Let P be the highest price computed by the Ascending Matroid Auction w.r.t. to the state σ. For all $e_k \in B_i$, we set $v_i^\sigma(e_k) = (P+1) + (q-k)$ and for all $f \in E_i \setminus B_i$, we set $v_i^\sigma(f) = p_f$. Then we have that all $e \in B_i$ are still awarded to buyer i in state $\sigma' = (\gamma_i(v_i^\sigma), \gamma_{-i})$ and she will also receive them in the same order as the $(q-k)$ terms in her valuation yield an ordering according to which i will select an item e_k before any item $e_\ell \in B_i$ with $\ell > k$ if they ever occur simultaneously in a monopsony controlled by buyer i. Moreover, for all $e \in B_i$ and $f \in E_i \setminus B_i$, we have $v_i^\sigma(e) \geq P + 1 > p_f = v_i^\sigma(f)$. Buyer i will also report all $f \in E_i \setminus B_i$ as critical at price p_f in Line 4. However, note that in state σ she might also report additional items that got awarded to her anyway as the tie-breaking removed other items first such that she controlled a monopsony. There is no difference in the outcome, however, as in σ' the same monopsony arises but i is

reporting an item as her favorite of this monopsony, which under the signaling σ' will not be an item that has previously been reported as critical. □

As a corollary, we obtain the following theorem.

Theorem 2.11 ([Bik+11, Theorem 17])**.** *For any profile of true valuations v_i, truthful signaling is an ex post equilibrium of the Ascending Matroid Auction.*

Proof. Suppose all buyers $j \in N \setminus \{i\}$ are signaling truthfully. Then by Lemma 2.10, there is a consistent signaling strategy that can be used as a best response. Among consistent strategies, by the Revelation Principle (cf. Theorem 1.32) and the VCG Theorem, signaling truthfully is a best response (Lemma 2.9). □

2.7 Communication Requirements and Privacy

As discussed in the introduction, ascending auctions might be preferred over sealed-bid auctions as communication costs might be lower or because a buyer does not have to give up her entire private information to the auctioneer, even though the total running time of the auction might be worse. In this section, we elaborate a bit more on these two arguments and compare how the Ascending Matroid Auction compares against its counterpart, the direct revelation VCG mechanism.

2.7.1 Communication

The sealed-bid VCG auction requires that all buyers submit their valuation $v_i \colon E_i \to \mathbb{Z}_+$ to the auctioneer and after the computation of the optimal allocation $\hat{B}$ along with prices $\hat{p}$, the auctioneer has to report those back to the buyers. Note that the first stage (the bidding phase) intuitively takes place in private communication between an individual buyer and the auctioneer, while the second stage (the report of the outcome) could be done via a public channel or in private with each individual buyer. If we assume that each buyer only needs to know which items she has won and at which price, then the amount of bits in each model would be the same (as the auctioneer only sends the information about an item to the corresponding buyer). On the other hand, if we want all buyers to know the complete outcome of the auction but do not allow existence of a public channel (for whatever reason), the auctioneer has to send the complete outcome to each buyer via a separate channel. We assume each buyer to use the same indexing

scheme for the items, so a word $\alpha \in \{0,1\}^m$ (or $\{0,1\}^{m_i}$ for an individual buyer i, where $m_i = |E_i|$) uniquely determines a subset of E (or E_i, respectively). We take a look at a minimal protocol (in terms of number of bits to be send) using the public channel to announce the results. We also assume the communication to be noise-free and without any errors.

A-priori, the valuations are unbounded and hence, so are the number of bits required to conduct the auction. Let us assume that $v_{\max} := \max\{v_i(e) : i \in N, e \in E_i\}$. Then the first stage requires $\Theta(m(\log v_{\max} + 1))$ bits of communication from the buyers to the auctioneer. Then, the auctioneer sends m bits to the public channel announcing the resulting allocation $\hat{B}$ and $\Theta(\rho(M)(\log v_{\max} + 1))$ bits for the prices (where the k-th number refers to the price of the k-th item that has been announced with a 1 in the initial string. In total this yields $\Theta((m + \rho(M))(\log v_{\max} + 1) + m)$ bits of communication. Note that there are not many degrees of freedom to change the protocol to conduct the sealed-bid auction. Clearly, the auctioneer needs every buyer's complete valuation function (assuming that the matroid is loop-free), to determine $\hat{B}$ and $\hat{p}$ correctly. For the auctioneer it can sometimes be cheaper to just enumerate the sold items using their index if $\rho(M)$ is small compared to m to announce $\hat{B}$ to the buyers (then this would correspond to $\rho(M)(\log m + 1)$ many bits). For the prices, a way to save communication costs is by announcing a price per buyer instead of item. These are not arguments from communication complexity (they are not necessary here since the protocol consists of a sequence of one-way protocols), the auctioneer essentially needs to find the highest and second-highest value of a cocircuit and to determine those he indeed has to see *all* of them. There is also nothing that we can do about sending those values with at least logarithmically many bits.

Now let us consider the protocol for the Ascending Matroid Auction. The auctioneer can use a single bit for each price increment step, which will amount to up to V bits. If a buyer stays silent after a price increment, it means that she does not have any critical item. Otherwise she can just announce her critical items using $\log m + 1$ bits per critical item. Note that it might very well happen, that all m items might be announced as critical during the auction and even at different price steps, which would make an encoding of a subset every time significantly less efficient (m^2 vs $m(\log m + 1)$ many bits). To tell a buyer that she is currently a monopsonist, the auctioneer can communicate the buyer's monopsony. This has to happen $\rho(M)$ many times and needs up to m bits every time. The monopsonist then has to name a single item from this cocircuit using $\log m + 1$ many bits. Note that the auctioneer does not have to announce the price to the buyer

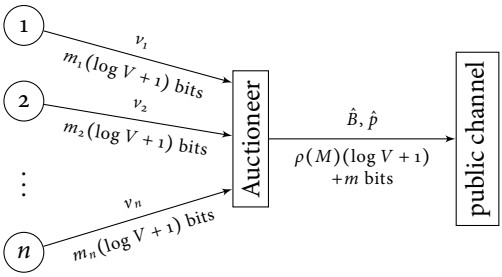
since it is already known to her using the price increment bits of the auctioneer. In total this protocol uses $\Theta(v_{\max} + m(\log m + 1) + \rho(M)(m + \log m + 1))$ many bits. Indeed these are more bits than the sealed-bid auction uses, in particular because the auctioneer has to send a bit for every price increment step (up to V many), whereas the sealed-bid auction, the number V only has to be sent occasionally (as bid or as price) using $\log v_{\max} + 1$ many bits, while nothing has to be sent V times.

However, one can envision a way to avoid the explicit announcement of a price increment. After all, we introduced the Ascending Matroid Auction as a *clock auction*, so we might just use a clock as a proxy to determine the current price. This model is also studied in communication complexity (see e.g., Impagliazzo and Williams [IW10]). We already used silence of a buyer after a price increment step as information that that buyer does not have any critical item at the current price. The goal is now to use a similar idea to avoid sending a single bit for a price increment step up to V times. If we use a discrete synchronized clock, then we can use elapsed time as information for the buyers that the price increased with enough time in between to perform necessary communication of critical items, monopsonies, favored items, and deletion and contraction operations. That is, we are trading additional elapsing time (measured by a synchronized clock accessible by all agents) for bits that do not have to be sent, which would reduce the total number of bits to be sent during the Ascending Matroid Auction to $\Theta(m(\log m+1)+\rho(M)(m+\log m+1))$, which are less bits than what the sealed-bid auction needs if V is large enough.

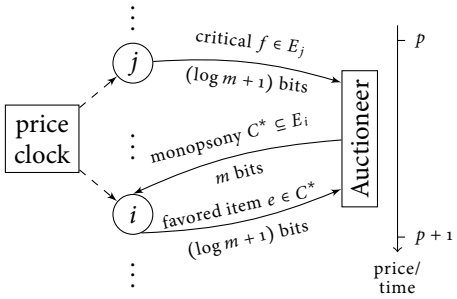
This comparison does not need to be considered fair; after all, using a synchronized clock, auctioneers and buyers in the sealed-bid auction could also just submit their information encoded in unary (using elapsing time) and then just sending a single bit as a separation symbol. However, using a synchronized clock in a clock auction clearly seems to be less absurd than its use in a sealed-bid auction. For practical purposes the comparison between those two protocols using different models of communication seems appropriate.

2.7.2 Privacy

Finally we also discuss how the Ascending Matroid Auction preserves buyers' privacy. An auction is a *preference elicitation process*, i.e., a method to learn sufficiently much information about the agents to make an optimal decision (here, finding and allocating a maximum weight matroid base). In computational social choice theory, Brandt and



(a) Sealed-bid auction protocol with public channel.



(b) Ascending auction protocol using a discrete synchronized clock.

Figure 2.6: Sketch of the two protocols.

Sandholm [BS05] say that a social welfare function preserves *unconditional privacy* if no coalition of agents can uncover any information except the information that can be inferred from the outcome and the coalition's preferences. This definition has been altered by Milgrom and Segal [MS20] to work with clock auctions.[7]

Definition 2.12 ([MS20])**.** A communication protocol for an auction has the *unconditional winners' privacy* property if for any buyer $i \in N$ and any possible private information v_i, v_i' and possible private information of other agents v_{-i} such that the protocol yields the same items for i (i.e., $\alpha(v_i, v_{-i}) \cap E_i = \alpha(v_i', v_{-i}) \cap E_i$, where α is the allocation determined by the protocol), the protocol finishes in terminals with the same outcome.

In sealed-bid auctions, the auctioneer would just examine all bids to determine allocation and prices (in our case, determine the highest and second-highest bid per cocircuit). However, this is not really necessary; the auctioneer needs to know the second highest bid in a cocircuit to determine the price but not really the amount of the highest bid. The information that there is a higher bid (and that bidder's identity) is enough to determine a winner. This is what the Ascending Matroid Auction exploits: assuming no ties (for the sake of this argument), the buyer with the highest valuation in a cocircuit will not reveal to the auctioneer what exactly her bid is. By not reporting any critical item, she just signals to the auctioneer that her bid is still higher than the current price and hence, any potential second-highest bid with that amount.

However, if we instead use the long-step variant of the Ascending Matroid Auction, a winning buyer reveals more information (at least the value of her least-desired item) but still less than in the sealed-bid auction.

[7] Note that an auction is indeed just a protocol to implement a social welfare function.

CHAPTER 3

Ascending Auctions via Maximum Flow

For this and the following chapter, we change the market setting a bit. The auctioneer is no longer constrained to sell an independent set of a matroid but in fact, wants to sell all items, while every buyer should receive a subset of items such that none of them feels that they would rather have more, less, or other items at current prices. That is, we aim for prices that support a *market-clearing* allocation or more explicitly, there is neither *excess supply* nor *excess demand*. We refer to such an equilibrium state as *Walrasian equilibrium*, named after French economist Léon Walras [Wal74].

The theory of Walrasian equilibria has been very well studied. In particular, there is a very well understanding about the existence and computation of Walrasian equilibria. Kelso and Crawford [KC82] showed that Walrasian equilibria are guaranteed to exist if the valuation functions of the buyers have the *gross substitute* property, and Gul and Stacchetti [GS99] proved in their Maximum Domain Theorem that gross substitute valuations are essentially the biggest class of valuation functions that guarantee existence of Walrasian equilibria.[1] In their follow-up work [GS00], they provided the framework for auction algorithms that compute Walrasian prices with the very simple idea that also follows the spirit of a Walrasian tâtonnement process: Starting from all-zero prices, the auctioneer increases prices on items that have excess demand until a market-clearing allocation is possible. For the setting in which workers have to be assigned to jobs

[1] We will go into details about the gross substitute property in Chapter 4.

(the unit demand and unit supply setting), this auction has already been studied by Demange, Gale, and Sotomayor [DGS86]. This can be expressed by the pseudocode Generic Ascending Auction.

Generic Ascending Auction: [GS00]

```
1 p := 0
2 while excess demand set F ⊆ E exists do
3     for e ∈ F do
4         p(e) := p(e) + 1
5 Compute a Walrasian allocation x
6 return (x, p)
```

We will define the term *excess demand set* precisely in Definition 3.8 in Section 3.3.2 but for now, we go with an informal description: Given valuation functions of buyers and prices, we can describe for every buyer i and every subset F of items, how many items i needs at least from F to receive a bundle she is happy with, this number is called the *requirement* of i on F. Then, a set F is in excess demand if the total requirement (the sum of all requirements on F) is larger than the cardinality of F.

Gul and Stacchetti left one crucial detail open: How can items in excess demand be computed? This gap was initially filled by Ausubel [Aus06; Aus05] using submodular function minimization. Murota, Shioura and Yang [MSY13] and Shioura [Shi17] showed that the function that has to be minimized is not just submodular but $L^\natural$-convex, which admits using a faster version of submodular function minimization. Moreover, these algorithms also work when items are available in multiplicities and that without having a preprocessing that creates a single item for every copy of some good. However, one might argue that this is still an overpowered sledgehammer for the problem of finding an excess demand set.

In this chapter, we show how one can use a far more basic algorithm, i.e., maximum flow algorithm, to compute the excess demand sets if the valuation functions are easy enough, namely *item-capped additive valuations* introduced in the following section. The more general case will be handled with a bit more involved, but nonetheless elegant algorithm in Chapter 4. Moreover, we also discuss how a buyer submits her demand correspondence to the auctioneer, which is non-trivial to do in a non-exhaustive way in most settings.

3.1 The Economic Model

We consider a market consisting of a set of m indivisible distinguishable *item types* E with *quantities* $b(e) \in \mathbb{Z}_+$ for all $e \in E$ and a set of n *buyers* N. We denote the set of *bundles* that a buyer can receive by $[0, b]_{\mathbb{Z}} := \{x \in \mathbb{Z}^E \mid 0 \leq x(e) \leq b(e), e \in E\}$. Each buyer $i \in N$ has a *valuation* for each item type, i.e., a function $\tilde{v}_i : E \to \mathbb{Z}_+$, and a *capacity* $d_i \in \mathbb{Z}_+ \cup \{\infty\}$ that specifies the maximum number of units a buyer can buy in total. From $\tilde{v}_i$ and d_i, we then derive the *item-capped additive valuation function* $v_i : [0, b]_{\mathbb{Z}} \to \mathbb{Z}_+$, as follows

$$v_i(x) = \max \left\{ \sum_{e \in E} y(e)\tilde{v}_i(e) : y \leq x, \|y\|_1 \leq d_i, y \in \mathbb{Z}_+^E \right\}. \tag{3.1}$$

Note that a buyer i's valuation function is fully determined by the pair $((\tilde{v}_i(e))_{e \in E}, d_i)$.

We design an ascending auction that finds an allocation $X = (x_i)_{i \in N}$ and prices $p = (p(e))_{e \in E}$ that satisfy the following properties:

(P1) For all $i \in N$, x_i maximizes buyer i's utility among all $x \in [0, b]_{\mathbb{Z}}$, i.e.,

$$x_i \in \arg\max\{v_i(x) - \langle p, x \rangle : x \in [0, b]_{\mathbb{Z}}\}.$$

(P2) For all $e \in E$ we satisfy the capacity constraint induced by b, i.e.,

$$\sum_{i \in N} x_i(e) \leq b(e).$$

(P3) For all $e \in E$ with $p(e) > 0$, all copies of e are sold, i.e.,

$$\sum_{i \in N} x_i(e) \geq b(e).$$

(P4) The allocation X maximizes total utility, i.e.,

$$X \in \arg\max \left\{ \sum_{i \in N} v_i(y_i) : Y \in [0, b]_{\mathbb{Z}}^N \right\}.$$

(P5) The prices p are component-wise minimal among all prices that satisfy (P1)-(P4).

Given a price vector p, we can define buyer i's *demand correspondence*, the set of all of her preferred bundles under the given prices, by

$$\mathcal{D}_i(p) := \arg\max\{v_i(x) - \langle p, x\rangle : x \in [0, b]_{\mathbb{Z}}\}$$

and her *indirect utility function*, i.e., the utility upon receiving such a preferred bundle and paying its price, by

$$V_i(p) := \max\{v_i(x) - \langle p, x\rangle : x \in [0, b]_{\mathbb{Z}}\}.$$

We say a price vector p *supports* an allocation X if for all $i \in N$, $x_i \in \mathcal{D}_i(p)$.
Now Property (P2), which in the economical sense states that there is no excess demand, can be read as a *packing* constraint. That is, we need to find one bundle x_i from each $\mathcal{D}_i(p)$ such that the there is enough supply to pack all the demanded bundles. An allocation $x = (x_i)_{i\in N}$ satisfying all packing constraints is called *packing*. On the other hand, Property (P3), stating that there should be no excess supply, can be viewed as a *covering* constraint. That is, we need to find one bundle x_i from each $\mathcal{D}_i(p)$ such that the total demand covers the total supply. An allocation $x = (x_i)_{i\in N}$ satisfying all covering constraints is called *covering*. Both of these conditions (P2)+(P3), or *no excess demand and no excess supply*, or *packing and covering*[2] together with the demand constraint (P1) define a *Walrasian equilibrium*.

Definition 3.1. We call a price vector p *packing* if it supports a packing allocation (P2). It is called *covering* if it supports a covering allocation (P3). If it supports an allocation X that is both packing and covering, it is called *Walrasian* and then (X, p) is called *Walrasian equilibrium*.

Note that we assume our valuation functions to be non-decreasing since the per-item valuations $\tilde{v}_i(e)$ are non-negative for all $i \in N$ and all $e \in E$. This monotonicity of the valuation functions v_i is also called *free disposal*; items that do not contribute positively to a buyers valuation (not utility) can be disposed by her at zero additional cost. With the free disposal property, we also have that an allocation can be considered Walrasian if all items with a positive price are sold. Items with price zero can then be allocated arbitrarily.
For the rest of the chapter, we will see how to obtain a Walrasian equilibrium via an ascending auction that only uses simple algorithmic ideas to determine excess demand

[2] One might also say *partitioning*.

sets, which we formally define in Subsection 3.3.2. For the moment, one can think of an excess demand set as a subset $S \subseteq E$ from which the buyers need more units of items than are available. In some sense, the here presented auction can be viewed as an efficient implementation of the auctions of Gul and Stacchetti [GS00], and Ausubel [Aus06] for the special case where all buyers have a valuation function of the type described in (3.1).

In the sequel, we use the convention that functions, sets etc. that are associated with a buyer i, carry the buyer's index as subscript, e.g., $\mathcal{D}_i(p)$ and x_i, whereas items e are referred to using parentheses, e.g., $b(e)$ or $x_i(e)$.

3.1.1 Differences to Chapter 2

Before we discuss the auction, we briefly discuss in which way the auction here differs from the matroid auction in Chapter 2. To illustrate, let us restrict the model above such that $b(e) = 1$ for all $e \in E$ and $d_i = \infty$. We also refer to this restricted model as the *unit supply* case. With the Ascending Matroid Auction we can in fact find a market-clearing allocation for the unit supply model from this chapter by transforming the market given by E, N and $(v_i)_{i \in N}$ to an instance of the market in Chapter 2 with sets E', $(E'_i)_{i \in N}$, valuations $(v'_i)_{i \in N}$ and a matroid M' as follows: Introduce one copy of each item per buyer, i.e., we build the set $E' = E \times N$ and the demand sets $E'_i = \{(e, i) : e \in E\}$. The valuation functions $v'_i \colon E'_i \to \mathbb{Z}_+$ are then just given by $v'_i((e, i)) = \tilde{v}_i(e)$. Then we define the partition matroid M' on the partition $\{\{(e, i) : i \in N\}\}_{e \in E}$ and capacity 1 for each part. A market-clearing allocation of E then corresponds to a base of the partition matroid defined above, where an element (e, i) in the base means that item e is allocated to buyer i.

This shows, that the Ascending Matroid Auction can find a market-clearing allocation in unit supply markets when all buyers have additive valuation functions (without a cap on the number of items). Using either Helgason's reduction [Hel72] or the extension by Raach and de Vries [RV25] it is also straight-forward to do the same by adding multiplicities, i.e., lifting the restrictions on $b(e)$ for all $e \in E$ again. However, there does not seem to be any immediate way to also use the Ascending Matroid Auction when we also want to use a limit on the number of items that buyers can buy.

On the other hand, one should point out that the setting in Chapter 2 allows a generalization of the term *market-clearing*. As seen above, the notion of market-clearing

can be modeled via a very easy partition matroid on the items but Chapter 2 allows for arbitrary matroids.

3.1.2 Contribution

In this chapter we describe how to find an excess demand set (specifically the inclusion-wise minimal one with maximum shortage) that is required in the Generic Ascending Auction only using a simple algorithmic idea if the valuation functions are item-capped additive valuations. The economic setting generalizes the one of Demange, Gale, and Sotomayor [DGS86] for matchings but it is a special case of markets with strong gross substitute valuations considered by e.g., Ausubel [Aus06] and Murota, Shioura, and Yang [MSY13]. However, the algorithms for this more general setting rely on submodular function minimization, which is highly non-trivial and slower.

This chapter is based on the paper *A flow-based ascending auction to compute buyer-optimal Walrasian prices* which was published in the Wiley journal *Networks* in 2024 [Eic+24a] but also available as a preprint [Eic+23a]. There are different ways to show that the in the paper presented auction returns buyer-optimal Walrasian prices. In the paper, these correctness results are shown directly on the auction by using properties of the auxiliary flow network. Here, we give an altered presentation (compared to the paper version) and provide different proofs using auxiliary theorems and lemmas by Ausubel [Aus06]. In particular, this means that Lemmas 3.6, 3.7,and 3.12 are new, and Lemma 3.11, Corollary 3.15, and Theorem 3.18 have new proofs. An advantage of presenting the results in this way is that it offers a smooth transition into Chapter 4, where the proof strategies are similar.

Since the paper [Eic+24a] was produced in close collaboration with all coauthors, in particular Katharina Eickhoff, we decided to use it in both PhD theses. While the description of the auction and its correctness are included in both theses, the foci are different. This thesis puts a special emphasis on running times and communication models whereas in the upcoming thesis of Katharina Eickhoff the focus is set to structural results on equilibria and price monotonicity. For that reason, the latter results are just stated without proof in this thesis.

3.2 Background Theory

We already introduced directed graphs in Section 1.1. There is a rich theory on flow networks, originally by Ford and Fulkerson [FF62] which has a vast spectrum of applications.

Definition 3.2. A *flow network* is a capacitated directed graph $G = (V, A, u)$ with two dedicated vertices $s, t \in V$ (*source* and *sink*) and capacities $u\colon A \to \mathbb{R}_+$.

Given a flow network $G = (V, A, u)$ with source s and sink t, a *flow* (*s-t-flow*) is a mapping $f\colon A \to \mathbb{R}_+$ that obeys the following constraints

capacity no edge is overcongested, i.e., $f(a) \leq u(a)$ for all $a \in A$ and
flow conservation for all $v \in V \setminus \{s, t\}$ the flow entering v is the same as the flow exiting v, i.e., $\sum_{a \in \delta^-(v)} f(a) = \sum_{a \in \delta^+(v)} f(a)$.

The *value* of the flow is $\operatorname{val}(f) := \sum_{a \in \delta^+(s)} f(a)$.

This definition prompts the following problem.

Maximum Flow

Given: flow network $G = (V, A, u)$, source $s \in V$, sink $t \in V$.
Find: s-t-flow $f\colon A \to \mathbb{R}_+$ such that $\operatorname{val}(f)$ is maximized.

The Maximum Flow problem is a prototype combinatorial optimization problem with multiple elegant ways to solve it. One particular feature—that is also exploited by most algorithms—is that there is very intuitive upper bound on the optimal objective function value, which is the minimum cut (cf. Figure 3.1).

Definition 3.3. Given a flow network $G = (V, A, u)$ with source s and sink t, a *cut* (or more explixitly, an *s-t-cut*) is a set $S \subseteq V$ with $s \in S$ and $t \notin S$. The *capacity* of the cut is $\operatorname{cap}(S) := \sum_{a \in \delta^+(S)} u(a)$.

Clearly, by the capacity constraint, the value of the maximum flow cannot exceed the capacity of the minimum cut, the solution to the following problem.

Minimum Cut

Given: flow network $G = (V, A, u)$, source $s \in V$, sink $t \in V$.
Find: s-t-cut $S \subseteq V$ such that $\operatorname{cap}(S)$ is minimized.

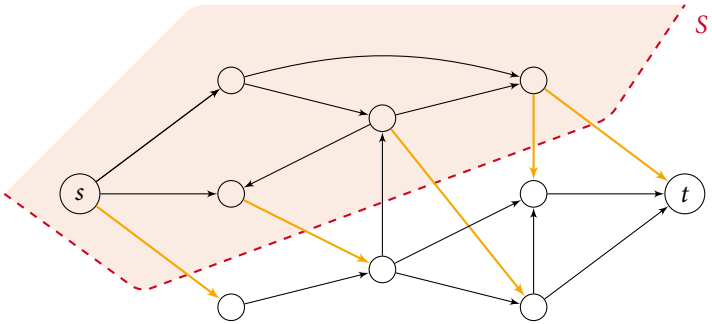


Figure 3.1: A visual proof of the fact that the minimum cut upper bounds the maximum flow. Given any s-t-cut S, every unit of flow that leaves s has to eventually cross the cut, i.e., travel from a vertex in S to a vertex outside of S (orange edges). There can be at most $\mathrm{cap}(S)$ units that can do this due to the capacity constraints. Since this bound has to hold for all s-t-cuts, it also has to hold for the minimum cut.

Remarkably, the two problems Maximum Flow and Minimum Cut are dual to each other and hence, their optimal objective function values are equal. This yields the following famous theorem by Ford and Fulkerson [FF56].

Max-Flow-Min-Cut Theorem (3.4) ([FF56, Theorem 1]). *Let $G = (V, A, u)$ be a flow network with source s and sink t. Let f^* be a maximum s-t-flow and S^* a minimum s-t-cut. Then* $\mathrm{val}(f^*) = \mathrm{cap}(S^*)$.

Moreover, if the capacities are all integral, then there exists a maximum flow that is integral and it can be found efficiently by various algorithms using different paradigms (and hence, also a minimum cut). In particular popular because of their simplicity are *augmenting path algorithms*, which are usually presented in everyone's first course in algorithms; the first one being by Ford and Fulkerson [FF56], however their presentation lacked details to make the algorithm polynomial in the size of the network, which were provided later by Dinic [Din70] and independently by Edmonds and Karp [EK72]. More recently, so called *push-relabel algorithms* became more popular, the first such algorithm is due to Goldberg and Tarjan [GT88]. Push-relabel algorithms are often outperforming augmenting path algorithms in theory and practice; not just for the Maximum Flow problem but for various combinatorial optimization problems. Very recently, Chen, Kyng, Liu, Peng, Probst Gutenberg, and Sachdeva presented how to compute a maximum flow in almost linear time [Che+23].

For our purposes, we will use the maximum flow in some auxiliary network to determine an allocation of the items. However, more interestingly, we use a slightly different auxiliary network and use a minimum cut to determine the items on which we increase the prices to reach a Walrasian price vector. Given a solution of the Maximum Flow problem f^*, we can determine a minimum cut by performing a breadth-first search in the *residual network* $G_{f^*} = (V, A_{f^*}, u_{f^*})$, where

$$A_{f^*} = \underbrace{\{a \in A \mid u(a) > f^*(a)\}}_{\overrightarrow{A}_{f^*}} \cup \underbrace{\{(v,u) : (u,v) \in A \text{ with } f(u,v) > 0\}}_{\overleftarrow{A}_{f^*}}$$

and residual capacities

$$u_{f^*}(a) = \begin{cases} u_{f^*}(a) - f^*(a) & \text{if } a \in \overrightarrow{A}_{f^*}, \\ f^*(a) & \text{if } a \in \overleftarrow{A}_{f^*}. \end{cases}$$

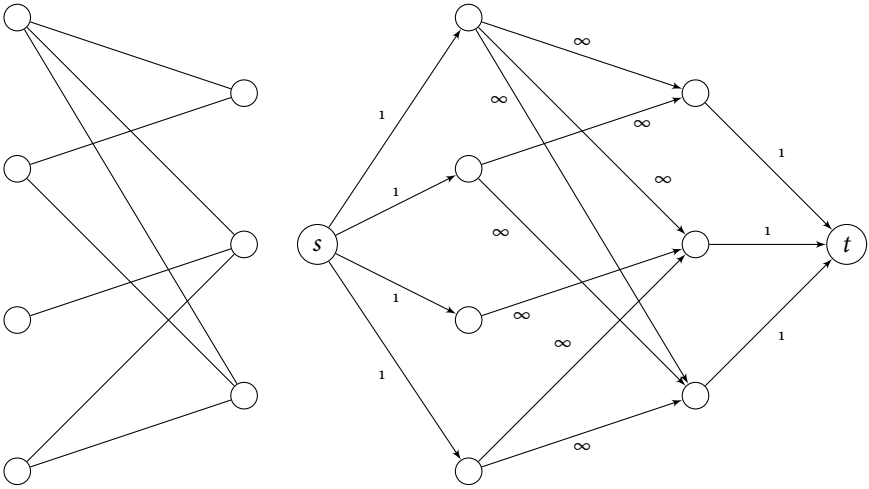
Clearly, the minimum cut in a flow network does not need to be unique. However, the set of all minimum cuts is a lattice with union and intersection as join and meet. The BFS in the residual network will always find the bottom element of this lattice, i.e., the inclusion-wise minimal minimum cut, which we will also refer to as the *left-most min cut*.

Finally, we shortly demonstrate how we can use the Maximum Flow problem to determine an allocation. An allocation of items to buyers is a generalized matching problem: every item type e can be allocated up to $b(e)$ times and a buyer i might be interested in buying multiple items, say up to d_i. For the sake of simplicity, let us restrict to the unit supply and unit demand setting for the moment, i.e., an allocation determines at most one buyer for every item and vice versa. This can be formulated as a matching problem in a bipartite graph: Given prices p and demand correspondences $\mathcal{D}_i(p)$ for each buyer, we arrange buyers $i \in N$ and items $e \in E$ in the bipartite graph $G(p) = (N \cup E, A(p))$ with

$$A(p) = \{(i,e) : e \in \mathcal{D}_i(p)\}.$$

Then the problem of finding a maximum allocation is just solving the matching problem in $G(p)$.

And here is how we can solve the problem:



(a) G (b) $\overrightarrow{G}$

Figure 3.2: From bipartite graph G to flow network $\overrightarrow{G}$.

Maximum Cardinality Bipartite Matching

Given: bipartite graph $G = (L \cup R, A)$.
Find: matching $\mu \subseteq A$ of maximum cardinality.

From G, we construct the flow network $\overrightarrow{G} = (\{s\} \cup L \cup R \cup \{t\}, \overrightarrow{A}, u)$ with

$$\overrightarrow{A} = \{(s,\ell) : \ell \in L\} \cup \underbrace{\{(\ell, r) : \{\ell, r\} \in A, \ell \in L, r \in R\}}_{\overrightarrow{A}_o} \cup \{(r,t) : r \in R\}$$

and capacities

$$u(a) = \begin{cases} 1 & \text{if } a = (s,\ell) \text{ with } \ell \in L, \\ \infty & \text{if } a = (\ell, r) \in \overrightarrow{A}_o, \\ 1 & \text{if } a = (r,t) \text{ with } r \in R. \end{cases}$$

An integral maximum flow f^* now corresponds to a matching $\mu = \{\{\ell, r\} : f^*(\ell, r) = 1\}$. A neat feature that comes in handy to find sets in excess demand (which are preventing a packing allocation) is the following theorem by Hall [Hal35].

Hall's Marriage Theorem (3.5) [Hal35, Theorem 1]. *Let $G = (L \cup R, A)$ be a bipartite graph. Then G has a matching covering L if and only if $|\Gamma(S)| \geq |S|$ for all $S \subseteq L$.*

A set $S \subseteq L$ violating the constraint of the theorem is called a *Hall violator*. A Hall violator can be computed (if it exists) using the residual network of a maximum flow in the auxiliary network by simply computing a minimum cut S^* and its intersection with L, i.e., $S^* \cap L$.

The case for maximum generalized matching with capacities greater than 1 on either side of the bipartite can now also easily be handled by just adjusting the capacity of the incoming edges for vertices $\ell \in L$ and outgoing edges for vertices $r \in R$. However, for the allocation problem in our market setting a few more steps are required, which we present in this chapter.

3.3 The Auction

Recall that we are given item types $e \in E$ with quantities $b(e)$ and buyers $i \in N$ with item-capped additive valuations v_i (determined by per-item valuations $\tilde{v}_i(e)$ and a capacity d_i). We are looking for a price vector p such that there exists an allocation X with the property that $v_i(x_i) - \langle p, x_i \rangle$ is maximized for all $i \in N$, subject to the constraints for all $e \in E$ that $\sum_{i \in N} x_i(e) = b(e)$, i.e., the allocation should be both packing and covering. Consequently, we also need to find Walrasian prices that support such an allocation and we want to do so via an *ascending auction* which works as follows: Starting at prices $p = \mathbf{0}$, the auctioneer asks all buyers which bundle they would like the most at the given prices (i.e., their demand correspondence). If it is not possible to fulfill all the demands (i.e., there is no packing allocation), then there is a subset of item types that are in excess demand and the auctioneer increases prices on these item types. On the other hand, if the auctioneer can give every buyer a set that she prefers without overselling an item type, then the auction stops. We will give a precise description of the auction in pseudocode Flow Auction at the end of the section. As an illustrative example, we show how the price raising steps (i.e., Gul and Stacchetti's Generic Ascending Auction) work in the unit demand setting.

Example 3.1. Consider the following market with four buyers $N = \{1, 2, 3, 4\}$ and four items $E = \{e_1, e_2, e_3, e_4\}$. The valuations for items ($\tilde{v}_i = (\tilde{v}_i(e_1), \tilde{v}_i(e_2), \tilde{v}_i(e_3), \tilde{v}_i(e_4))$ for $i \in N$) are as follows: $\tilde{v}_1 = (1, 0, 0, 1)$, $\tilde{v}_2 = (0, 1, 2, 0)$, $\tilde{v}_3 = (1, 1, 3, 1)$, and $\tilde{v}_4 =$

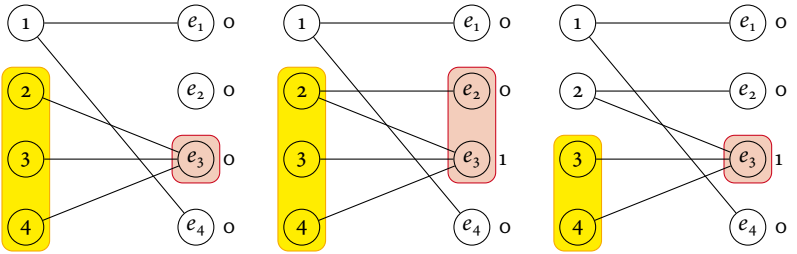
$(1, 0, 5, 0)$. This yields the unit demand valuation functions $v_i \colon 2^E \to \mathbb{Z}_+$ with $v_i(S) = \max_{e \in S} \tilde{v}_i(e)$ for all $i \in N$. Given any price vector $p = (p(e))_{e \in E}$, to determine whether it is Walrasian (i.e., a packing and covering allocation exists), it suffices to find a perfect matching in a bipartite graph: $G(p) = (N \cup E, \{\{i, e\} : \tilde{v}_i(e) - p(e) \text{ maximum and non-negative}\})$. In our example, the corresponding demand graph for all-zero prices ($G(\mathbf{0})$ is depicted in Figure 3.3a. We can see that there is no perfect matching in $G(\mathbf{0})$. Hence, by Hall's Marriage Theorem, there has to exists a Hall violator $S = \{2, 3, 4\}$ and its neighborhood $\Gamma(S) = \{e_3\}$ is an excess demand set on which we increase the price by 1. Intuitively, the three buyers 2, 3, and 4 demand the item e_3 but it is available only once. In the next demand graph $G(0, 0, 1, 0)$ the Hall violator of the previous iteration is again a Hall violator but this time e_2 is also in its neighborhood, so one could increase the price on e_2 and e_3 this time (cf. Figure 3.3b. However, the item e_2 is not really in excess demand; only buyer 2 is interested in it, so we do not really want to increase its price (our objective is not revenue maximization, in fact, we would like to obtain minimal Walrasian prices). A more careful look at the graph reveals that the set $\{3, 4\}$ is also already a Hall violator and its neighborhood only contains $\{e_3\}$ again (see Figure 3.3c). Depending on the chosen Hall violator (and the corresponding excess demand set), we end up with the demand graph in Figure 3.3d or Figure 3.3e, corresponding to the price vectors $p = (0, 1, 2, 0)$ or $p = (0, 0, 2, 0)$ (both Walrasian), respectively.

Remark. The unit demand setting can always be modeled such that the number of buyers and items are the same: if there are less buyers than items, we can add dummy buyers that have a valuation of 0 for all items. On the other hand, if there are less items than buyers, then we can treat "getting nothing" as something that is captured within the market by adding dummy items that every buyer values at 0.

3.3.1 Demand Correspondences and Oracles

In the auction briefly sketched above, the auctioneer asks every buyer $i \in N$ in every iteration with price p for her *demand correspondence* $\mathcal{D}_i(p)$, i.e., the set of bundles that maximize her utility under the given prices. We assume in this chapter (and also Chapter 4) that buyers will report their demand correspondence truthfully.[3] When

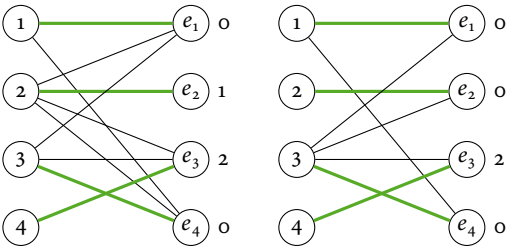
[3] Unlike for the auction in Chapter 2, this is not a mild assumption, however. We will show in Section 3.5 that the auction we present here is not incentive compatible. That is, truthful reporting is not an ex post Nash equilibrium of the auction.



(a) $G(0)$ with Hall violator and excess demand set.

(b) $G(0,0,1,0)$ with Hall violator and excess demand set.

(c) $G(0,0,1,0)$ with smaller Hall violator and excess demand set.



(d) $G(0,1,2,0)$ with perfect matching.

(e) $G(0,0,2,0)$ with same perfect matching but lower prices (and less edges).

Figure 3.3: Illustrations of the demand graphs for Example 3.1. Hall violators are highlighted in yellow boxes on the left, their neighborhood (excess demand set) in red on the right. A perfect matching is highlighted with a thick green line.

given explicitly, this might require a lot of communication from buyer to auctioneer as the following example illustrates.

Example 3.2. Consider the case with m item types each in quantity $b(e) = 1$ and a buyer 1 with $d_1 = \frac{m}{2}$. Assume the prices to be such that $\tilde{v}_1(e) - p(e) = u$ for some $u > 0$ and all $e \in E$, i.e., she values every item the same. Then every subset of E of size $\frac{m}{2}$ is a preferred bundle, i.e., 1's demand correspondence has size $|\mathcal{D}_1(p)| = \binom{m}{\frac{m}{2}} \in \Omega(2^m m^{-\frac{1}{2}})$.[4]

However, in the the special case where buyers have item-capped additive valuations one can argue that specifying all of those sets in the demand correspondence explicitly is not really necessary. There is a very simple greedy algorithm to construct a preferred bundle. Its correctness essentially follows from the fact that the set of (inclusion-wise minimal) preferred bundles form a matroid base set. We prove this fact below for the unit supply setting, where $b(e) = 1$ for all $e \in E$. The actual statement then follows either by Helgason's reduction (cf. Theorem 1.15) or by our own reduction for the more general case of gross substitute valuations whose reduction we will demonstrate in Section 4.7.

Lemma 3.6. *Let E be our finite set of items and $\breve{\mathcal{D}}_i(p)$ denote the demand correspondence of a buyer $i \in N$ restricted to inclusion-wise minimal sets, i.e.,*

$$\breve{\mathcal{D}}_i(p) = \{X \subseteq E \mid v_i(X) - p(X) \text{ maximum and } X \text{ minimal}\},$$

where v_i is an item-capped additive valuation. Then $(E, \breve{\mathcal{D}}_i(p))$ is a matroid.

Proof. We prove the base axioms (cf. Proposition 1.10). Clearly (B1) holds as the utility maximization problem is over a non-empty finite domain. For (B2), consider any $X, Y \in \breve{\mathcal{D}}_i(p)$ and $e \in X \setminus Y$. Suppose for all $f \in Y \setminus X$ we have $X - e + f \notin \breve{\mathcal{D}}_i(p)$, i.e., $v_i(X) - p(X) > v_i(X-e+f) - p(X-e+f) = v_i(X) - \tilde{v}_i(e) + \tilde{v}_i(f) - p(X) + p(e) - p(f)$. But then $\tilde{v}_i(e) - p(e) > \tilde{v}_i(f) - p(f)$ and hence, $v_i(Y - f + e) - p(Y - f + e) > v_i(Y)$ in contradiction to $Y \in \breve{\mathcal{D}}_i(p)$. □

Lemma 3.6 is a special case of a more general statement (Lemma 4.5) about gross substitute valuations, which we will use in Chapter 4.

We can classify the item types into four tiers:

[4] By Stirling's approximation, $\binom{m}{\frac{m}{2}} \sim \sqrt{\frac{2}{\pi m}} 2^m$.

Construct Preferred Bundle:

Input: Valuations $\tilde{v}_i$, demand d_i, prices p

Output: Preferred bundle x_i

Determine order $e_1 \succ_i \ldots \succ_i e_m$ such that $e \succ_i f$ only if $\tilde{v}_i(e) - p(e) \geq \tilde{v}_i(f) - p(f)$

while $d_i > 0$ **and** $k \leq m$ **and** $\tilde{v}_i(e_k) > p(e_k)$ **do**

  $x_i(e_k) := \min\{b(e), d_i\}$

  $d_i := d_i - x_i(e_k)$

  $k := k + 1$

return x_i

crucial items $E_i^1(p) := \{e \in E \mid \text{for all } x \in \mathcal{D}_i(p), x(e) = b(e)\}$. The buyer wants to buy the whole supply of those item types.

replaceable items $E_i^2(p) := \{e \in E \mid \text{there exists } x, y \in \breve{\mathcal{D}}_i(p), x(e) > 0, y(e) < b(e)\}$. The buyer wants to fill up her bundle with any of those items. Each item in this set is replaceable by any other item in this set.

omittable items $E_i^3(p) := \{e \in E \mid \text{there exists } x \in \mathcal{D}_i(p) \setminus \breve{\mathcal{D}}_i(p), x(e) > 0\}$. The buyer is indifferent between receiving these items to fill up her bundle and not receiving these items. In particular, since we assume all valuations to be non-negative, $E_i^3(p)$ contains all item types e with $p(e) = 0$ that are not already contained in $E_i^1(p) \cup E_i^2(p)$.

unwanted items $E_i^4(p) := \{e \in E \mid \text{for all } x \in \mathcal{D}_i(p), x(e) = 0\}$. Items that would be replaced in every preferred bundle by an item with better payoff, in particular those that would actually yield negative utility under current prices.

Given valuations $\tilde{v}_i$ and demands d_i for each $i \in N$ it would suffice to just communicate these $m + 1$ numbers to auctioneer once but this would defeat the purpose of having an ascending auction and would just resemble a sealed-bid auction. An ascending auction (following the framework of [GS00]) should only ask for the demand correspondence of a buyer given a price vector p. Using our definitions above, the auctioneer can ask for an implicit description of $\mathcal{D}_i(p)$ by simply asking for the sets $E_i^1(p)$, $E_i^2(p)$, and $E_i^3(p)$. In summary, this yields a very simple communication protocol between auctioneer and buyers (requiring to send just $2|E|$ bits per buyer and price vector) which reveals indeed the complete demand correspondence of every buyer given the posted prices to the auctioneer.

In the sequel, we refer to such a request of the auctioneer to a buyer as a call to a *tier oracle* and we analyze the auction under algorithmic aspects with respect to the number of tier oracle calls the auctioneer has to make to determine an excess demand set.

The following lemma is useful for item-capped additive valuations.

Lemma 3.7. *For all price vectors p and $x \notin \mathcal{D}_i(p)$, there exists some $y \in [0, b]_{\mathbb{Z}}$ such that $y = x - \chi_e + \chi_f$ for some $e, f \in E \cup \{\varnothing\}$ and $v_i(x) - \langle p, x\rangle < v_i(y) - \langle p, y\rangle$.*

Proof. Let $x \notin \mathcal{D}_i(p)$.

Case 1: $\|x\|_1 < d_i$. If there exists some $f \in E$ with $\tilde{v}_i(f) - p(f) > 0$ and $x(f) < b(f)$, then $y = x + \chi_f$ ($e = \varnothing$) is the desired vector. In this case f can be chose from $E_i^1(p) \cup E_i^2(p)$. Otherwise, x must contain some $e \in E_i^4(p)$ because $x \in \widetilde{\mathcal{D}}_i(p) \subseteq \mathcal{D}_i(p)$ if x contains all items from E_i^1 and E_i^2 without exceeding the demand d_i. Then $y = x - \chi_e$ ($f = \varnothing$) is as desired.

Case 2: $\|x\|_1 = d_i$. Then x cannot contain the full supply of items in $E_i^1(p)$ and otherwise only items from $E_i^2(p)$ as it would be a set from i's demand correspondence. If there is an $f \in E_i^1(p)$ with $x(f) < b(f)$, then we can exchange this with any $e \in \operatorname{supp} x \setminus E_i^1(p)$ and obtain $y = x - \chi_e + \chi_f$ as desired. Otherwise, there is an item $e \in E_i^3(p) \cup E_i^4(p)$ which can be exchanged against some $f \in E_i^2(p)$.

Case 3: $\|x\|_1 > d_i$. Again, if there is an item for $E_i^1(p)$ available, we can just add it, to improve utility. The same is true if there are items in $E_i^2(p)$ available but $x(E_i^1(p) \cup E_i^2(p)) < d_i$. Otherwise, if x contains a bundle from $\mathcal{D}_i(p)$, then removing any $e \in E_i^3(p) \cup E_i^4(p)$ with $p(e) > 0$ yields again more utility. If no such item exists, then there must be too many items of $E_i^2(p)$ and at least one of them mus have a positive price. □

Note that Lemma 3.7 actually holds for much more general valuation functions, the gross substitutes, for which it is not only a necessary but also sufficient condition as shown in [GS99].

3.3.2 Excess Demand and Excess Supply

We still need a a formal notion of excess demand and supply in our model. In principle the auctioneer needs to know for any set $F \subseteq E$, there are enough units of item types in F available to assign to buyers according to their demand correspondences. This motivates the following definition.

Definition 3.8. Let p be a price vector, $F \subseteq E$ and $i \in N$. We denote *i's requirement* on F w.r.t. p by

$$r_i^p(F) := \min_{x \in \mathcal{D}_i(p)} \langle x, \chi_F \rangle.$$

The *total requirement* on F w.r.t. p is then given by

$$r^p(F) := \sum_{i \in N} r_i^p(F)$$

A set $F \subseteq E$ is in *excess demand*[5] if $r^p(F) > b(F)$. We call the difference $r^p(F) - b(F)$ the *shortage* of F.

Intuitively, $r_i^p(F)$ is composed by the number of crucial items in F and the minimum number of replaceable items in F that has to be in a preferred bundle (i.e., the difference of $d_i^2(p)$ and the number of items outside of F that can substitute items in F).

Remark. Let $v_i \colon [0, b]_{\mathbb{Z}} \to \mathbb{Z}_+$ be an item-capped additive valuation and $F \subseteq E$. Then it holds that,

$$r_i^p(F) = b(F \cap E_i^1(p)) + \left(d_i^2(p) - b(E_i^2(p) \setminus F)\right)^+.$$

3.3.3 A Flow Network to Check for Excess Demand

The auctioneer has to be able to determine an excess demand set or an allocation that is both packing and covering, given the buyers' responses (as a tier oracle) to a price vector p. We have seen (in Example 3.1) that the case with unit demand and unit supply can essentially be handled using an algorithm for perfect bipartite matchings, which boils down to solving an instance of Maximum Flow.

The duplication method A natural idea is it to transform every instance of the multi-supply and multi-demand setting to a unit supply and unit demand setting by treating an item type e with quantity $b(e)$ just as $b(e)$ individual items and a buyer i with demand d_i as d_i individual buyers. Certainly, an ascending auction of the unit supply and -demand instance will return Walrasian prices, but these prices are in general not buyer-optimal, as the following example demonstrates.

Example 3.3. Consider one buyer $N = \{1\}$ with a demand of $d_1 = 2$, and two different items $E = \{e_1, e_2\}$ with a supply of $b(e_1) = b(e_2) = 1$ which are valued differently

[5] overdemanded

by the sole buyer, say $\tilde{v}_1 = (5,1)$. If we copy the buyer, there is no stable allocation if $p(e_1) < 4$ as both copies compete for object e_1. If $p(e_1) = 4$, both copies of the sole buyer are indifferent between the objects and thus, $p = (4,0)$ and $x = ((1,0),(0,1))$ is a Walrasian equilibrium. However, considering the original situation, since the buyer is alone $p = (0,0)$ and $x = (1,1)$ is the buyer-optimal Walrasian equilibrium. The intuitive explanation is that the duplication method is oblivious to the fact that the two copies represent the same buyer and hence, the computed Hall sets are wrongly interpreted as competition between two buyers. Thus, the prices computed by the duplication method are not buyer-optimal.

Another weakness of such an attempt is that the resulting matching problem (solved by a maximum flow algorithm) blows up the graph exponentially.

Now we make a more sophisticated attempt to model the problem to find an excess demand set (or to determine that non exists and the prices are Warasian). Using the tier oracle, which yields the four sets $E_i^1(p)$ to $E_i^4(p)$ for each buyer i and her total demand d_i, we construct the following auxiliary flow network $G(p) = (V_G, A_G(p))$ with vertices $V_G = \{s,t\} \cup (N \times [2]) \cup E$ and edge set

$$\begin{aligned} A_G(p) = \{(s,(i,k)) : i \in N, k \in [2]\} \cup \{((i,1),e) : i \in N, e \in E_i^1(p)\} \\ \cup \{((i,2),e) : i \in N, e \in E_i^2(p)\} \cup \{(e,t) : e \in E\}. \end{aligned}$$

The main idea is that a flow which completely saturates the incoming edges of a buyer i, i.e., $(i,1)$ and $(i,2)$, corresponds to a vector of her demand correspondence. To this end, we split for each $i \in N$ her demand into $d_i^1(p) = b(E_i^1(p))$ and $d_i^2(p) = \min\{d_i - d_i^1(p), b(E_i^2(p))\}$. Then we can use the idea from Section 3.2 to assign capacities $u_G^p : A_G(p) \to \mathbb{Z}_+$ on the edges:

$$\begin{aligned} u_G^p(s,(i,1)) &= d_i^1(p) && \text{for all } i \in N, \\ u_G^p(s,(i,2)) &= d_i^2(p) && \text{for all } i \in N, \\ u_G^p((i,1),e) &= b(e) && \text{for all } i \in N \text{ and all } e \in E_i^1(p), \\ u_G^p((i,2),e) &= \min\{b(e), d_i^2(p)\} && \text{for all } i \in N \text{ and all } e \in E_i^2(p), \\ u_G^p(e,t) &= b(e) && \text{for all } e \in E. \end{aligned}$$

Note that we did not set the capacities of the $((i,k),e)$-edges to ∞ (just as in described in Section 3.2 for the Maximum Cardinality Bipartite Matching problem). By

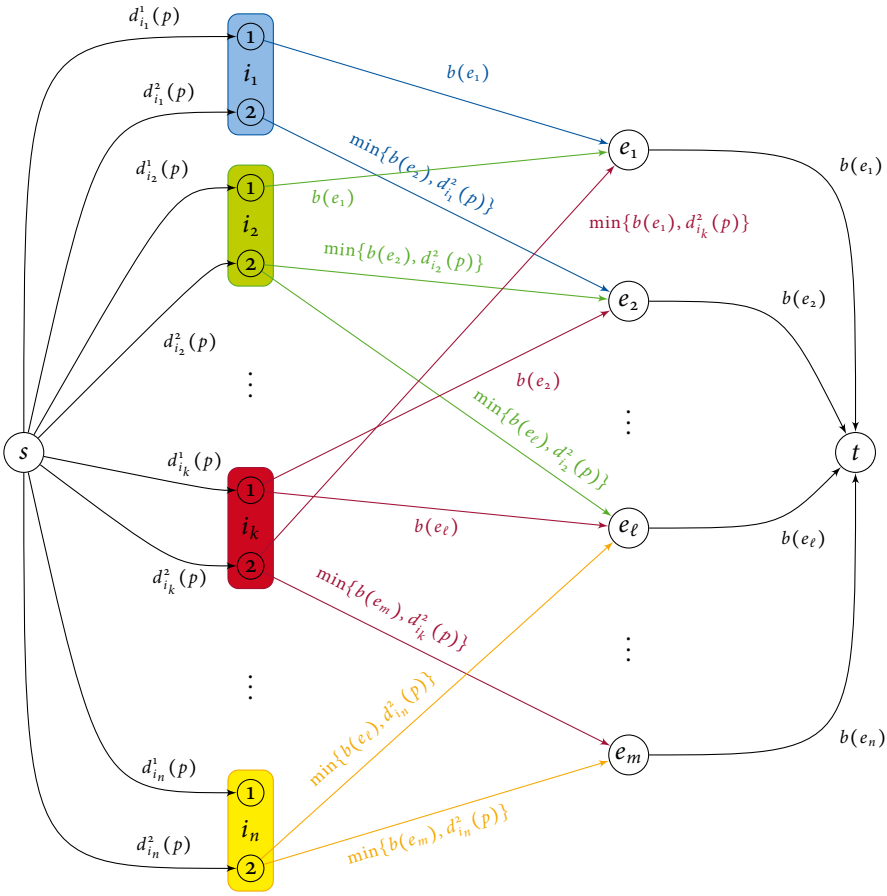


Figure 3.4: Sketch of an auxiliary flow network. The blue buyer i_1 has only item type e_1 as crucial and e_2 as replaceable, i.e., she wants to buy the full supply of e_1 but only a fraction of the supply of e_2. The green buyer i_2 competes with i_1 for e_1, as it is also a crucial item type for her but she is indifferent between filling up her bundle with e_2 or e_ℓ. The red buyer i_k has multiple item types that are crucial and replaceable. Finally the yellow buyer i_n only has replaceable items.

choosing these values a bit more carefully, the capacity of an s-t-cut has an economic interpretation. If $S = \{s\}$ or $S = V_G \setminus \{t\}$, then $\mathrm{cap}(S)$ is just the total requirement $r^p(E)$ or the total supply $b(E)$, respectively. More general, given an s-t-cut S, we can compute its capacity as follows (see Figure 3.5):

$$\begin{aligned}\mathrm{cap}(S) &= \sum_{(i,k)\notin S} d_i^k(p) + \sum_{\substack{(i,1)\in S\\ e\in E_i^1(p)\setminus S}} b(e) + \sum_{\substack{(i,2)\in S\\ e\in E_i^2(p)\setminus S}} \min\{b(e), d_i^2(p)\} + \sum_{e\in S} b(e)\\ &= \underbrace{r^p(E)}_{\text{const.}} + \underbrace{\sum_{e\in S} b(e) - \sum_{(i,k)\in S} d_i^k(p)}_{\text{supply} - \text{demand in } S}\\ &\quad + \underbrace{\sum_{\substack{(i,1)\in S\\ e\in E_i^1(p)\setminus S}} b(e) + \sum_{\substack{(i,2)\in S\\ e\in E_i^2(p)\setminus S}} \min\{b(e), d_i^2(p)\}}_{\substack{\text{correcting term}\\ \text{part of the demand that can be covered outside of } S}}.\end{aligned}$$

That is, the capacity of an s-t-cut S (with at least one buyer vertex and one item vertex) is just the difference between the supply of items within S and the demand of the buyers within S for those items (plus a constant). Hence, a minimum cut in this network corresponds to a set of buyers and items for which this difference is minimum or, in other words, for which there is a maximum shortage of items. An inclusion-wise minimal such cut (i.e., the left-most min cut) necessarily includes a minimal set (again, w.r.t. inclusion) of items that are in such a maximum shortage.

Lemma 3.9. *A price vector p is packing if and only if $G(p)$ admits a flow of value* $\mathrm{cap}(\{s\})$.

Proof. From an integral s-t-flow $f\colon A_G(p) \to \mathbb{Z}_+$ we can construct a packing allocation X of bundles: One unit of flow via vertices (i,k) and e for some $i \in N$, $k \in [2]$, and $e \in E$ corresponds to one unit of $x_i(e)$. More precisely $x_i(e) = f((i,1),e) + f((i,2),e)$ for all $i \in N$, $e \in E$.[6,7] The assignment is feasible, since a feasible flow will not have more than $b(e)$ units of flow routed through e.

Since $G(p)$ only has integral capacities u_G^p, there exists an integral maximum flow f^*. It has value $\mathrm{val}(f^*) = \mathrm{cap}(\{s\})$ if and only if all s-leaving arcs are saturated by f^*, which

[6] By construction, we have $E_i^1(p) \cap E_i^2(p) = \varnothing$ however, which means at least one of those summands is identical zero.

[7] Under abuse of notation, we treat $f(a) = 0$ if $a \notin A_G(p)$.

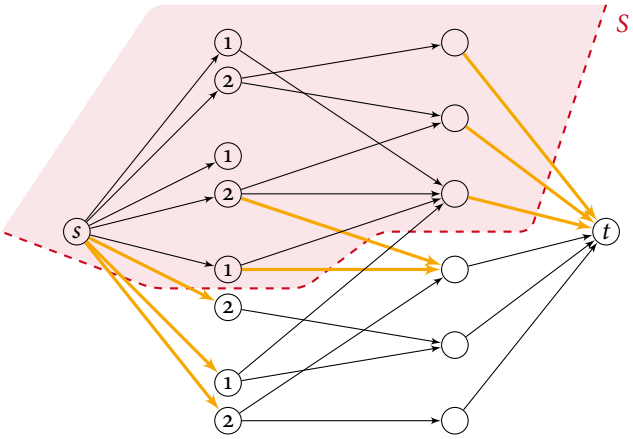


Figure 3.5: The three type of edges that occur in an s-t-cut of $G(p)$.

means that every buyer's total demand under prices p (i.e., $d_i^1(p) + d_i^2(p)$) is saturated with items from there respective tiers of crucial ($E_i^1(p)$) and replaceable items ($E_i^2(p)$), respectively. □

However, we cannot be sure whether such an allocation—additionally to being a packing one—is also covering and hence, Walrasian. Item types that yield zero payoff to a buyer i are never allocated to i using the flow network construction for G. To allocate the remaining items (if any), we construct another auxiliary network $H(p) = (V_H, A_H(p))$ based on $G(p)$ with vertex set $V_H = V_G \cup \{(i,3) : i \in N\} \cup \{(i_o,3)\}$ and edge set

$$A_H(p) = A_G(p) \cup \{(s,(i,3)) : i \in N\} \cup \{((i,3),e) : i \in N, e \in E_i^3(p)\}.$$

The capacities $u_H^p : A_H(p) \to \mathbb{Z}_+$ are given by $u_H^p|_{A_G(p)} \equiv u_G^p$ and

$$\begin{aligned}
u_H^p(s,(i,3)) &= d_i - (d_i^1(p) + d_i^2(p)) && \text{for all } i \in N,\\
u_H^p((i,3),e) &= d_i - (d_i^1(p) + d_i^2(p)) && \text{for all } e \in E_i^3(p) \text{ with } p(e) > 0,\\
u_H^p(s,(i_o,3)) &= b(E),\\
u_H^p((i_o,3),e) &= b(e) && \text{for all } e \in E \text{ with } p(e) = 0.
\end{aligned}$$

We introduce a dummy buyer i_o that will be allocated all items that have price 0 and are not demanded by anyone else. Recall that this is not in violation of the covering

constraint for Walrasian equilibria as we can just assume that these items may stay unallocated or—equivalently—are just allocated to any buyer who may dispose them for free.

The following lemma is obvious.

Lemma 3.10. *A maximum flow f^* of $G(p)$ is feasible in $H(p)$.*

Lemma 3.11. *A maximum flow f^* in $H(p)$ corresponds to a Walrasian equilibrium if and only if p is packing and* $\mathrm{val}(f^*) = \mathrm{cap}(V_H \setminus \{t\})$.

Proof. As p is packing, there is a packing allocation and it is feasible to assign a flow according to this allocation in $H(p)$ due to Lemma 3.10. Starting with this flow, every buyer stays within her demand correspondence if we augment the flow in $H(p)$ using paths via their respective $(i,3)$-vertices as she only receives items that cannot lower her utility (as the flow will respect her total capacity d_i). The dummy buyer i_0 is capable of receiving *all* items with price 0, so if the maximum flow indeed saturates all (e,t)-edges, then the resulting allocation x is indeed covering and hence, (x,p) is a Walrasian equilibrium. □

Note the following subtlety in the proof above. We started with the initial flow that we can receive from $G(p)$ and augment it in $H(p)$ until all edges entering t are saturated. Maximum flow algorithms might not necessarily maintain the initial flow; however, if we assume that the used algorithm is an augmenting path algorithm that augments along shortest paths (i.e., the Edmonds-Karp Algorithm) then there will never be a rerouting of flow away from $(i,1)$- or $(i,2)$-vertices towards a $(j,3)$-vertex for some $j \neq i$. Moreover, the leftover items with price 0 will all be allocated towards the dummy buyer i_0 first. If after that there is still some slack on some edge entering t, then its corresponding item type e must have positive price but cannot be in anyone's tier 1 set and every augmentation (if possible) with just replace an allocation with an equivalently good one for the buyers involved. On the other hand, if there is no augmentation possible such that all edges entering t are saturated, then the price vector is not covering, i.e., the price vector already exceeded the minimal Walrasian price vector. However, by Ausubel [Aus06] and [BLN13] this will not happen given that our price increment steps are always performed on the minimal excess demand set of maximum shortage.

Note that the final flow f in $H(p)$ with p Walrasian and f such that edges $(s,(i,1))$ and $(s,(i,2))$ are saturated yields a suitable allocation that assigns every buyer a set from

her demand correspondence and allocates all items to (non-dummy) buyers except for those that are not demanded even at price 0.

For the sake of convenience, from now on we will omit the price vector p when it is clear from the context, i.e., we will shorten $E_i^1(p), \dots, E_i^4(p), d_i^1(p), d_i^2(p), G(p)$ etc. to $E_i^1, \dots, E_i^4, d_i^1, d_i^2, G$, respectively.

3.3.4 Implementing the Auction

We can now implement the Generic Ascending Auction using the aforementioned flow networks.

Flow Auction:

```
1 p := 0
2 Construct G(p) and compute integral maximum flow f*
3 while val(f*) < cap({s}) do
4     Compute left-most minimum cut S ⊆ V_G
5     for e ∈ S ∩ E do
6         p(e) := p(e) + 1
7     Construct new G(p) and compute integral maximum flow f*
8 Construct H(p) and compute integral maximum flow f* that corresponds to
    complete allocation X
9 return (X, p)
```

In Line 2, we can choose from a variety of algorithms for the Maximum Flow problem as mentioned in Section 3.2. From any integral solution, we can also easily construct the required (left-most) minimum cut for Line 4 by running a breadth-first search from s in the residual network.

After analyzing correctness and running time of the Flow Auction, we also present two improved implementations in the next section.

3.4 Analysis

In this section, we prove that the Flow Auction computes indeed a Walrasian equilibrium and that it is the (unique) buyer-optimal one, i.e., the prices are component-wise minimal. Moreover, we show that the running time to find the necessary excess demand

sets is polynomial in $|N|$ and $|E|$. We cannot guarantee that the total running time is polynomial as it also depends on $\max_{\substack{i\in N\\ e\in E}} \tilde{v}_i(e)$ but the two suggested variants with warm starts and long steps yield an improvement of the total running time (while still being only pseudo-polynomial).

3.4.1 Correctness

The analysis involves a series of technical lemmas. We first show that if we increase the price of the items in a set $F \subseteq E$ a buyers indirect utility decreases by exactly her requirement for F.

Lemma 3.12. *Let p be a price vector, $F \subseteq E$, and $i \in N$. Then $V_i(p+\chi_F) = V_i(p) - r_i^p(F)$.*

Proof. Let $x \in \mathcal{D}_i(p)$ that uses as few items from F as possible, i.e., exactly $r_i^p(F)$ and $y \in \mathcal{D}_i(p+\chi_F)$. Then

$$\begin{aligned} V_i(p) &= v_i(x) - \langle p, x\rangle \\ &= v_i(x) - \langle p, x\rangle + \langle \chi_F, x\rangle - \langle \chi_F, x\rangle \\ &= v_i(x) - \langle p+\chi_F, x\rangle + \langle \chi_F, x\rangle \\ &\leq v_i(y) - \langle p+\chi_F, y\rangle + \langle \chi_F, x\rangle \\ &= V_i(p+\chi_F) + \langle \chi_F, x\rangle \\ &= V_i(p+\chi_F) + r_i^p(F). \end{aligned}$$

Next we show that for all $x \in [0, b]_{\mathbb{Z}}$, $V_i(p) \geq v_i(x) - \langle p+\chi_F, x\rangle + r_i^p(F)$. Then the other direction follows from taking $x \in \mathcal{D}_i(p+\chi_F)$. For $x \in [0,b]_{\mathbb{Z}}$ let $\kappa = V_i(p) - (v_i(x) - \langle p, x\rangle)$. We prove the inequality by induction on κ. If $\kappa = 0$, then $x \in \mathcal{D}_i(p)$ and hence, $\langle \chi_S, x\rangle \geq r_i^p(S)$ for all $S \subseteq E$. Thus,

$$\begin{aligned} v_i(x) - \langle p+\chi_F, x\rangle &= v_i(x) - \langle p, x\rangle - \langle \chi_f, x\rangle \\ &= V_i(p) - \langle \chi_F, x\rangle \\ &\leq V_i(p) - r_i^p(F) \end{aligned}$$

as desired. Now let $\kappa > 0$. Then, by Lemma 3.7, there exists $y \in [0, b]_{\mathbb{Z}}$ with $y = x - \chi_e + \chi_f$ for some $e, f \in E$ and $v_i(y) - \langle p, y\rangle > v_i(x) - \langle p, x\rangle$ and hence, the κ corresponding to y is smaller. We apply the induction hypothesis with y and obtain

$$\begin{aligned} V_i(p) &\geq v_i(y) - \langle p + \chi_F, y\rangle + r_i^p(F) \\ &= v_i(y) - \langle p, y\rangle - \langle \chi_F, y\rangle + r_i^p(F) \\ &\geq v_i(x) - \langle p, x\rangle + 1 - \langle \chi_F, y\rangle + r_i^p(F) \\ &\geq v_i(x) - \langle p, x\rangle - \langle \chi_F, x\rangle + r_i^p(F), \\ &= v_i(x) - \langle p + \chi_F, x\rangle + r_i^p(F), \end{aligned}$$

where the second inequality follows from integrality of v_i, x, y, and p and the last inequality from the fact that $y = x - \chi_e + \chi_f$ and hence, their inner products with χ_F can differ by at most 1. □

The next lemma essentially justifies Definition 3.8. In particular, we show that a price vector p is packing if and only if there is no set in excess demand.

Lemma 3.13. *A price vector p is packing if and only if there is no set $F \subseteq E$ in excess demand. Moreover, if p is not packing and $S^* \subseteq V_G(p)$ is a left-most minimum cut, then $F = S^* \cap E$ is in excess demand.*

Proof. By Lemma 3.9, there is a packing allocation if and only if there exists a flow f in $G(p)$ with $\mathrm{val}(f) = \mathrm{cap}(\{s\}) = \sum_{i \in N} d_i^1 + d_i^2$. Thus, by the Max-Flow-Min-Cut Theorem, p is packing if and only if $\{s\}$ is a minimum cut (which then happens to be the left-most minimum cut) of $G(p)$ So consider f with $\mathrm{val}(f) = \sum_{i \in N} d_i^1 + d_i^2$, implying that a packing allocation exists, and let $F \subseteq E$. Then for all $(i, k) \in N \times [2]$ we have by flow conservation,

$$\begin{aligned} \sum_{e \in E_i^k \cap F} f((i,k), e) = \overbrace{f(s, (i,k))}^{= d_i^k} - \sum_{e \in E_i^k \setminus F} \overbrace{f((i,k), e)}^{\leq b(e)} \\ \geq \max\{0, d_i^k - b(E_i^k \setminus F)\}. \end{aligned} \tag{3.2}$$

Now we use the inequality above to show that the total demand on F does not exceed the supply of F:

$$\begin{aligned}
r^p(F) &= \sum_{i \in N} r_i^p(F) \\
&= \sum_{i \in N} \left(b(E_i^1 \cap F) + (d_i^2 - b(E_i^2 \setminus F))^+ \right) \\
&= \sum_{i \in N} \left(d_i^1 - b(E_i^1 \setminus F) + (d_i^2 - b(E_i^2 \setminus F))^+ \right) \\
&= \sum_{i \in N} \left(\max\{0, d_i^1 - b(E_i^1 \setminus F)\} + \max\{0, d_i^2 - b(E_i^2 \setminus F)\} \right) \\
&\leq \sum_{i \in N} \left(\sum_{e \in E_i^1 \cap F} f((i,1),e) + \sum_{e \in E_i^2 \cap F} f((i,2),e) \right) \\
&= \sum_{e \in F} \sum_{i \in N} (f((i,1),e) + f((i,2),e)) \\
&= \sum_{e \in F} f(e,t) \\
&\leq b(F),
\end{aligned}$$

where we used (3.2) at the first inequality. Thus, F is not in excess demand.

Now we show that the condition is also sufficient. Suppose p is not packing and let S^* be the left-most minimum cut in $G(p)$. Since p is not packing, we have

$$\operatorname{cap}(S^*) < \operatorname{cap}(\{s\}) = \sum_{i \in N} d_i^1 + d_i^2 \tag{3.3}$$

Let $F := S^* \cap E$ and $P := S^* \cap (N \times [2])$. The capacity of S^* is

$$\operatorname{cap}(S^*) = b(F) + \sum_{(i,k) \in P} \sum_{e \in E_i^k \setminus F} b(e) + \sum_{(i,k) \in N \setminus P} d_i^k. \tag{3.4}$$

By combining and rearranging (3.3) and (3.4) we get the following chain of inequalities:

$$\begin{aligned} b(F) &< \sum_{(i,k)\in P} d_i^k - \sum_{(i,k)\in P} b(E_i^k \setminus F) \\ &= \sum_{(i,k)\in P} \left(d_i^k - b(E_i^k \setminus F)\right) \\ &\leq \sum_{(i,k)\in P} \max\left\{0, b(E_i^k \setminus F)\right\} \\ &= r^p(F). \end{aligned}$$

Thus, F is in excess demand. □

So far we essentially showed that if the auction terminates, the resulting prices and allocation are packing. We still need to show that the auction indeed is guaranteed to terminate and that the prices are not only packing but also covering. To show termination, we utilize a potential function, the *Lyapunov* $L\colon \mathbb{Z}_+^E \to \mathbb{Z}_+$ with

$$L(p) := \sum_{i\in N} V_i(p) + \langle p, b\rangle,$$

which is lower bounded by 0 (as both terms are clearly non-negative) but decreases in every iteration of the auction by a positive integer (we show this in the proof of Theorem 3.18) and hence, has to reach its minimum. The Lyapunov in its continuous version was introduced by Varian [Var81]. Its discrete version, which we use here, has been used by Ausubel [Aus06]. Indeed, our flow auction is an efficient implementation of Ausubel's auction restricted to item-capped additive valuation functions.

Lemma 3.14 ([Aus06, Proposition 1])**.** *A price vector p is Walrasian if and only if $L(p) \leq L(q)$ for all price vectors q.*

From Lemma 3.12, we immediately get the following corollary.

Corollary 3.15. *Let p be a price vector and $F \subseteq E$, then the difference of the Lyapunov is exactly the shortage of F, i.e., $L(p) - L(p + \chi_F) = r^p(F) - b(F)$.*

Now recall that a minimum cut corresponds to an excess demand set of maximum shortage (see the discussion after the flow network construction in Subsection 3.3.3). The following corollary will imply together with a result of Ausubel [Aus06] that the Flow Auction will not accidentally pass by a Walrasian price vector in the sense that

a price of an item type is increased such that an excess supply set is generated. Thus, assuming that there was no excess supply set at the initial price vector (say $\mathbf{o}$), there will never be an excess supply set during the auction, which implies that the first packing price vector must be Walrasian.

Corollary 3.16. *Let S^* be the left-most min cut of $G(p)$. Then $S^* \cap E$ is the minimal minimizer of $L(p + \chi_X)$ over all $X \subseteq E$.*

To prove the main theorem, we only need one more lemma by Ausubel [Auso6]. Its proof can be found in [Auso5].

Lemma 3.17 ([Auso6, Proposition 4])**.** *Let p_* be the minimal Walrasian price vector and $p \leq p_*$. Let $S \subseteq E$ be a minimal excess demand set of maximum shortage at p. Then $p + \chi_S \leq p_*$.*

Theorem 3.18. *The prices $\hat{p}$ returned by Flow Auction are Walrasian. Moreover, they are minimal Walrasian.*

Proof. Let p_* be minimal Walrasian. We show that $\hat{p} = p_*$. From the loop variant $\mathrm{val}(f^*) < \mathrm{cap}(\{s\})$, we know that if the Flow Auction terminates, the returned prices must be packing due to Lemma 3.9. To show that the auction indeed terminates, we use the Lyapunov

$$L(p) = \sum_{i \in N} V_i(p) + \langle p, b \rangle.$$

Note that $L(p) \geq 0$ for all $p \in \mathbb{Z}_+$ since $V_i(p) = \max\{v_i(x) - \langle p, x \rangle : x \in [0, b]_{\mathbb{Z}}\} \geq v_i(\mathbf{o}) - \langle p, \mathbf{o} \rangle = 0$ and $\langle p, b \rangle \geq 0$ as $p, b \geq \mathbf{o}$. Then let $K := L(\mathbf{o}) = \sum_{i \in N} V_i(\mathbf{o}) \geq 0$. We now show that in each iteration of the **while** loop of the Flow Auction, the Lyapunov has to decrease by at least 1. Since $K = L(\mathbf{o})$ is finite and L is lower bounded by 0, this can only happen finitely often and hence, the auction has to terminate eventually. Let $F \subseteq E$ be an excess demand set in some iteration of the Flow Auction, i.e., we increase the price for all $e \in F$ and obtain $p' = p + \chi_F$. Thus, we have $\langle p', b \rangle - \langle p, b \rangle = b(F)$. However, $\sum_{i \in N} V_i(p) - \sum_{i \in N} V_i(p') = r^p(F) > b(F)$ by Lemmas 3.12 and 3.13. Hence, in total L decreases by at least 1 when going from p to p'. Thus, the Lyapunov can decrease at most K times and the auction has to terminate after at most K iterations. However, the auction terminates only if there is no excess demand set and hence, the resulting price vector has to be packing by Lemma 3.13. Using Lemma 3.17, we also have that using a minimal minimizer $F \subseteq E$ of $L(p + \chi_F)$ (which we do by selecting

the left-most min cut S of $G(p)$ and using $F = S \cap E$, see Corollary 3.16), then for the resulting price vector it holds that $p + \chi_F \leq p_*$. □

3.4.2 Running Time

It is clear that the total running time of the Flow Auction cannot be bounded in $n = |N|$ and $m = |E|$ alone as the number of price increments also depend on the valuation functions, which can take arbitrary high numbers. However, Murota, Shioura and Yang [MSY13] show that the number of price increment steps—in our case the number of maximum flow and minimum cut computations—is just $p_*^{\max} = \max_{e \in E} p_*(e)$, where p_* is the (minimal) Walrasian price vector, which is found by the Flow Auction.

Theorem 3.19 ([MSY13, Theorem 1.2])**.** *The number of price increments in the Generic Ascending Auction, using a minimal excess demand set of maximum shortage in every iteration, is $\|p_* - p_0\|_\infty$, where p_0 is the initial price vector.*

The more interesting question is, how fast a single computation of an excess demand set is. In the Flow Auction this depends of course on the used algorithm for Maximum Flow, whose running time typically depends on the size of the graph (the number of edges) and sometimes the maximum capacity. In our case, the size of the flow network is bounded by $\mathcal{O}(nm)$ as we have $2n$ edges from s to the buyer vertices, at most nm edges between buyer and item vertices, and m edges from item vertices to t. The maximum capacity is $B := \max_{e \in E} b(e)$. We use the shorthand TO as a placeholder for the time that a tier oracle call required and MF for the time of a maximum flow computation. To give an intuition, as we have seen in Construct Preferred Bundle a tier oracle essentially just requires sorting the items according to their payoff, which can be done in time $\mathcal{O}(m \log m)$. Note however, that this task is carried out by the buyers and not part of the auction, which just has to receive two bits of information per item and buyer that encode to which tier an item belongs to. We briefly mentioned a few algorithms for the Maximum Flow problem in Section 3.2. The fastest known algorithm by Chen, Kyng, Liu, Peng, Probst Gutenberg, and Sachdeva achieves a running time of $\mathcal{O}((nm)^{1+o(1)} \log B)$. However, we also need to take into account the initial construction of the flow network (for which we need the tier oracles) and the minimum cut computation. Using the tier oracles, we can construct all $\mathcal{O}(nm)$ edges of the graph. No matter whether, the maximum flow algorithm uses a residual network or not, we can construct it from an optimal integral flow after its termination to build the residual

network, also with $\mathcal{O}(nm)$ edges and run a breadth-first search to find the left-most minimum cut, which again is linear in the number of edges, i.e., $\mathcal{O}(nm)$. In summary, the running time is dominated by the maximum flow computation.

Theorem 3.20. *A single price increment step of the Flow Auction requires* $\mathcal{O}(n\mathsf{TO} + \mathsf{MF})$ *time.*

Note that this running time is faster than the previous method by Murota, Shioura, and Yang (for more general valuation functions, however), which has a running time of $\mathcal{O}(n\mathsf{DO} + nm^4 \log(nmB)\mathsf{ExO})$, where DO is an oracle call that returns one set from a demand correspondence, and ExO an oracle call that returns for a given set D and two item types e, f how many units of e can be exchanged for the same number of units of f such that the resulting set is still in the demand correspondence.[8] As we use different oracles, the running times stated above are not directly comparable but note that the running time of a tier oracle can simply be bounded by $\mathcal{O}(m \log m)$ as it suffices to sort the item types once in decreasing order of $\tilde{v}_i(e)$ to assign the item types to their respective tier. In the following chapter, we also see that the running time bound by Murota, Shioura, and Yang above is not the best possible as we improve it to $\mathcal{O}(n\mathsf{DO} + nm^3\mathsf{ExO})$ (for arbitrary strong gross substitute valuations; cf. Corollary 4.32). The running time to determine an excess demand set achieved by Flow Auction in Theorem 3.20 can still be better (using the running time for MF of $\mathcal{O}((nm)^{1+o(1)})$ by Chen, Kyng, Liu, Peng, Probst Gutenberg, and Sachdeva [Che+23]) even if we assume that DO and ExO require $\mathcal{O}(1)$ time if $n^{o(1)} < m^{2-o(1)}$, i.e., when the number of buyers is not drastically larger than the number of items.

In the following we present two ways to improve the overall running time of this auction.

warm-start auction use the optimal flow from the previous iteration to obtain an optimal flow for the current iteration by observing how the buyers' demand correspondences change (i.e., which items go from crucial to replaceable or vice versa).

long-step auction decrease the number of rounds the auction has to perform by increasing prices by a value greater than 1 in case the auxiliary network G does not change too much (in the sense that the left-most min cut stays the same).

One easy observation that we can make is that the auxiliary flow network for prices $p+\chi_F$ for some excess demand set F clearly cannot change too much from the network for

[8] We discuss these oracles more in the next chapter.

prices p. The idea is now to amend the flow network according to the updated demand correspondences but keep as much flow as possible from the previous assignment. The procedure can be found in the pseudocode Flow Update Algorithm. Let $D_p := \sum_{i \in N} d_i^1(p) + d_i^2(p)$, i.e., the minimum number of items that have to be assigned to have a packing allocation. We can show that the obtained flow—while not necessarily optimal for $G(p + \chi_F)$—will have a slack that is at most the shortage under p.

Flow Update Algorithm:

Input: Network $G(p)$ with maximum flow f and network $G(p')$, where $p' = p + \chi_F$ for some $F \subseteq E$
Output: A feasible flow f' in $G(p')$ with $D_p - \mathrm{val}(f) \geq D_{p'} - \mathrm{val}(f')$

```
f' := 0
for (i, e) ∈ N × E with f((i,1), e) > 0 or f((i,2), e) > 0 do
    if ((i,1), e) ∈ A_G(p') then
        Add f((i,1)e) + f((i,2), e) units of flow to f' on path (s, (i,1), e, t)
    else if ((i,2), e) ∈ A_G(p') then
        Add f((i,1)e) + f((i,2), e) units of flow to f' on path (s, (i,2), e, t)
return f
```

We show first that the flow that we obtain from Flow Update Algorithm is indeed feasible.

Lemma 3.21. *The flow f computed in Flow Update Algorithm is a feasible flow in $G(p')$.*

Proof. Note that flow conservation is fulfilled in every vertex of $G(p')$ by construction. It remains to show that the capacity constraints for $G(p')$ are satisfied. Let S be the left-most minimum cut in iteration $G(p)$ and $F = S \cap E$ the set of items on which the prices are increased.

- For $(s, (i,1))$ we assign $\sum_{e \in E_i^1(p')} (f((i,1), e) + f((i,2), e))$ units of flow and the capacity is given by $\sum_{i \in E_i^1(p')} b(e)$. Note that $f((i,1), e) + f((i,2), e) \leq b(e)$ since the only edges leaving e has capacity $b(e)$ already in $G(p)$.
- For $(s, (i,2))$ we analyze the change from $d_i^2(p)$ to $d_i^2(p')$, i.e., when items in F increase in price.
  - If there are item types in $E_i^2(p) \cap E_i^1(p')$, let $Z := E_i^2(p) \cap E_i^1(p')$ be the types which move from E_i^2 to E_i^1 and $Y := E_i^2(p) \cap E_i^2(p')$ be the types that stay in E_i^2.

We know that the objects in $E_i^2(p) \cap F \subseteq E_i^2(p')$. Moreover, the objects in $E_i^1(p)$ stay in $E_i^1(p')$, i.e., $E_i^1(p) \subseteq E_i^1(p')$. Hence, the demand d_i^1 increases and the demand d_i^2 decreases by the number of items in Z, i.e., $\sum_{e \in Z} b(e)$.
Recall that left-most minimum cut S is defined by the vertices reachable from s in the residual network corresponding to f. By definition $Z \cap F = \varnothing$ and $Y \subseteq F$. If $(i, 2)$ is in the left-most minimum cut, the edges in the cut which are leaving this vertex are saturated in any maximum flow. Thus, $f((i,2), e) = u_G^p((i,2), e) = b(e)$ for $e \in Z$. It follows that $u_G^p((i,2), e) = \min\{b(e), d_i^2\} = b(e)$ since otherwise with this item type the complete demand could be saturated. But this is a contradiction since buyer i wants to buy items with less payoff at prices p', in other words $e \in E_i^1(p')$ and not $e \in E_i^2(p')$.
Since this flow is shifted to $(i, 1)$ after the price update, the flow is reduced by the number of items in Z as well, thus the capacity constraints are still satisfied.
If $(i, 2)$ is not in the left-most minimum cut, we cannot reach $(i, 2)$ from items in $Y \subseteq F$. Hence, the flow on the edges $((i, 2), e)$ with $e \in Y$ is zero (otherwise the backwards edge exists in the residual network and $(i, 2)$ is reachable). Thus, in the Flow Update Algorithm we do not assign any flow to an edge through the vertex $(i, 2)$. Hence, the capacity of $(s, (i, 2))$ is still not exceeded.

– Consider the case where $E_i^2(p) \cap E_i^1(p') = \varnothing$.
If E_i^2 does not lose any objects by the price update, we know that

$$d_i^2(p') = d_i^2(p) + \sum_{e \in E_i^1(p) \cap E_i^2(p')} b(e).$$

Hence, the capacity is not exceeded on $(s, (i, 2))$.
It remains to show that the capacity is not exceeded if E_i^2 loses some objects (it may get new types from E_i^1, however). The only situation which can cause problems is if the demand of objects in E_i^2 decreases. The demand d_i^2 only decreases if there are objects moving from E_i^2 to E_i^1 (which does not happen by assumption) or if there are not enough objects with a positive payoff. The latter case implies that all items available in $E_i^2(p')$ are demanded and thus, the capacity constraint on $(s, (i, 2))$ is fulfilled.

- For edges (e, t) with $e \in E$ the capacities do not change, so the constraints in $G(p')$ are trivially fulfilled.

- For $((i,1),e)$ and $((i,2),e)$ it follows directly by the definition of the capacity for $G(p')$ and since the capacities on the s-leaving and t-entering edges are not exceeded. □

Lemma 3.22. *The adaption of the network and the check whether the set of objects F in the left-most minimum cut changes runs in time $\mathcal{O}(n\mathsf{TO} + nm)$.*

Proof. After asking every buyer to report their demand correspondences for the new prices, every edge is checked and then in constant time the flow is adapted (see Flow Update Algorithm). Afterwards, the left-most min cut can be computed by a breadth-first search in the residual network. As there are $\mathcal{O}(nm)$ many edges, the statement follows. □

We can use the structure of the left-most minimum cut to show that if the flow decreases in an update step, then the demand decreases by at least the same amount. This will help us to show that the prices returned by the auction are covering.

Lemma 3.23. *Given a maximum flow in $G(p)$ with corresponding left-most minimum cut S. Let $F = S \cap E$ be the excess demand set and $p' = p + \chi_F$ be the price vector after the price update. Then, the demand $d_i^1 + d_i^2$ of buyer i will decrease at least by*

$$\sum_{\substack{e \in E_i^2(p): \\ e \notin E_i^1(p') \cup E_i^2(p')}} f((i,2),e). \tag{3.5}$$

Proof. If (3.5) is zero, the statement is obvious. Thus, from now on, we consider the cases when flow is removed. This can only occur if some item types get a payoff of zero after the price update, i.e., if

$$F' := E_i^2(p) \setminus (E_i^1(p') \cup E_i^2(p')) \neq \emptyset.$$

Note that F' describes the set of item types which move from E_i^2 to E_i^3. Since E_i^3 just contains types with that yield no additional utility and E_i^2 only those with a positive payoff, all item types in F' are contained in F.
By definition, the demand $d_i^1 + d_i^2$ will decrease by

$$\max\left\{0, d_i^2(p) - \sum_{e \in E_i^2(p) \setminus F} b(e)\right\}.$$

If $(i,2)$ is contained in the left-most minimum cut, all edges from $(i,2)$ to $E_i^2(p) \setminus F$ are saturated (since they are not reachable from $(i,2)$). Hence, there can be at most $d_i^2(p) - \sum_{e \in E_i^2(p) \setminus F} b(e) \geq 0$ units of flow going through $(i,2)$ to vertices in F'. Thus, for buyer i we reduce the demand at least by the removed flow units traveling through a vertex of i.

If $(i,1)$ is not contained in the left-most minimum cut, there is no flow on the edges from $(i,2)$ to F. Thus, no flow through buyer i is removed and we are done in this case as well. □

Corollary 3.24. *The flow f computed in the Flow Update Algorithm satisfies*

$$D_p - \mathrm{val}(f) \geq D_{p'} - \mathrm{val}(f').$$

While the Flow Update Algorithm in the worst case does not really speed up the auction (or even one iteration of it), it is clear that in some cases a maximum flow computation can be skipped in iterations where the auxiliary network did not change at all, for instance. However, the Flow Update Algorithm becomes really useful when we combine it with the following approach to make the whole auction faster by decreasing the number of iterations.

One natural approach to speed up the auction is to increase the price by more than 1 on items of an excess demand set. Clearly, this has to be done carefully as otherwise we might increase the price of an item by too much such that it was not in excess demand for some price vector between the original price and the new one. This could even lead to the undesired outcome that we overshoot the minimal Walrasian price vector. We want to increase the prices only by that much such that the left-most minimum cut does not change, i.e., find the correct increment q such that one price-raising step is performed instead of q increments of 1. A natural idea is to use a binary search for q (in contrast, the unit price increments are essentially a linear search for q). Clearly, one drawback of this is that buyers have to reveal more information: instead of only revealing their demand correspondences for price vectors that are reached during the auction (between $p_0 = \mathbf{0}$ and p_*) they also have to reveal their demand correspondences for *virtual prices* that will never be reached by the auction. In particular, the prices that are presented to the buyers are not monotone. What is nice about the binary search is that it essentially leaves the communication protocol between auctioneer and buyers as it was for the unit-step auction: the auctioneer announces prices (virtual or not) and the buyers respond with their partition into tiers for these prices.

However, it is not a priori clear that a binary search would work correctly. After all, between a unit increment and the increment q found by the binary search, there might be a different increment $q' < q$ for which the left-most minimum cut in the particular auxiliary network $G(p + q')$ changes. The following lemma by Shioura [Shi17] implies that this is not the case.

Lemma 3.25 ([Shi17, Proposition 4.16])**.** *Let $S \subseteq E$ be a steepest descent direction at p of the Lyapunov, i.e., $L(p + \chi_S)$ is minimum and let $S' \subseteq E$ be a steepest descent direction at $p + \chi_S$. Then, $0 \leq L(p + \chi_S) - L(p + \chi_S + \chi_{S'}) \leq L(p) - L(p + \chi_S)$ holds. Moreover, assuming that S and S' are chosen as minimal steepest descent directions, at least one of the following holds:*

(i) $L(p + \chi_S) - L(p + \chi_S + \chi_{S'}) < L(p) - L(p + \chi_S)$ *or*
(ii) $S' \supseteq S$.

In our case, this means whenever the left-most minimum cut changes, then either the shortage decreases (the slope of the Lyapunov, cf. Corollary 3.15) or the number of item types that are in excess demand increases. Due to the implied monotonicity of the left-most minimum cut, we hence can apply the binary search to find the optimal price increment. To perform the binary search, we use an upper bound on the valuations $\tilde{v}_{\max} := \max_{i \in N, e \in E} \tilde{v}_i(e)$ that the auctioneer is ought to know, which in most economic settings is a reasonable assumption. Note that Lemma 3.25 also applies to the more general market settings in which buyers can have arbitrary gross substitute valuations (see Chapter 4) and is not restricted to the Flow Auction which only works for item-capped additive valuations.

Lemma 3.26. *Every update step (Line 12) takes time at most $\mathcal{O}(n\mathsf{TO} + nm)$.*

Proof. To increase the flow by one unit (or more), we use a breadth first search (BFS) in the residual network. The same is true for the computation of a min cut. The BFS runs in time $\mathcal{O}(nm)$.
By Lemma 3.22 the adaption of the network runs in time $\mathcal{O}(n\mathsf{TO} + nm)$. □

Lemma 3.27. *The number of update steps is bounded by $\mathcal{O}(\log(\tilde{v}_{\max})mD_{\mathbf{0}})$.*

Proof. At the end of the auction we obtain some prices p_* and some flow f^* such that $\mathrm{val}(f^*) = \mathrm{cap}(\{s\}) = D_{p_*}$. It starts with the all 0-flow and $D_{\mathbf{0}}$. During the algorithm $\mathrm{val}(f) - D_p$ is non-increasing (see Corollary 3.24). If it decreases in an update step it decreases by at least by one.

Long-Step Auction:

$p := \mathbf{0}$
Construct $G(p)$ and compute integral maximum flow f^* and left-most minimum cut S^*
while $\text{val}(f^*) < \text{cap}(\{s\})$ **do**
    $\alpha_{\text{up}} := \tilde{v}_{\max}$
    $\alpha_{\text{lo}} := 1$
    $\alpha := \left\lfloor \frac{\alpha_{\text{up}} + \alpha_{\text{lo}}}{2} \right\rfloor$
    while $\alpha_{up} > \alpha_{lo}$ **do**
        Construct $G' := G(p + \alpha\chi_{S^* \cap E})$ and compute left most minimum cut S'
        if $S' = S^*$ **then** $\alpha_{\text{lo}} := \alpha$ **else** $\alpha_{\text{up}} := \alpha$
        $\alpha := \left\lfloor \frac{\alpha_{\text{up}} + \alpha_{\text{lo}}}{2} \right\rfloor$
    $p := p + \alpha\chi_{S^* \cap E}$
    Construct new $G(p)$ (using the Flow Update Algorithm) and augment to new integral maximum flow f^* and compute left-most minimum cut S^*
Construct $H(p)$ and compute integral maximum flow f^* that corresponds to complete allocation X
return (X, p)

Using Lemma 3.25 we can show that this value decreases all $\mathcal{O}(\log(\tilde{v}_{\max})m)$ update steps.

Given some network and some flow, if we can augment the flow, clearly the value $\text{val}(f) - D_p$ decreases. So assume it is not possible to augment the flow. With one update step we can recognize this situation and compute a minimum cut giving an minimal excess demand set of maximum shortage. It is possible that we amend the network in the next update step, but in the resulting graph, the adapted flow is already maximum, i.e., the minimum cut does not change. Using a binary search, we can find in $\mathcal{O}(\log(\tilde{v}_{\max}))$ time a point where the minimum cut changes.

It is still possible that the left-most minimum cut changes, but the capacity of the minimum cut and thus, the value of the maximum flow remains the same. Using Lemma 3.25, we know that the left-most minimum cut changes monotonically. By the structure of the network, the left-most minimum cut is determined by the item types in the cut. Thus, a left-most minimum cut with the same capacity can be seen at most m times. Hence, after $\mathcal{O}(\log(\tilde{v}_{\max})m)$ update steps, the flow needs to be augmentable. □

As a corollary of Lemma 3.26 and Lemma 3.27 we obtain a running time bound of Long-Step Auction.

Theorem 3.28. *The running time of the Long-Step Auction is bounded by* $\mathcal{O}(\log(\tilde{v}_{\max})mnD_{\mathbf{o}}(m+\mathsf{TO}))$, *where* $\tilde{v}_{\max} = \max_{i\in N, e\in E} \tilde{v}_i(e)$ *and* $D_{\mathbf{o}} = \sum_{i\in N} d_i$.

3.5 Optimality and Incentives

In this last section of the chapter, we show a few properties of Walrasian equilibria. A fundamental result is that any Walrasian equilibrium is Pareto-efficient, i.e., no agent (including the seller or auctioneer) can improve their own utility without some other agent losing utility. This is generally referred to as *First Welfare Theorem* which is rooted in the works of Pareto [Par06a] and Walras [Wal74] and formally proven by Arrow and Debreu [AD54; Arr51; Deb51] and is a long-standing result which is included in every textbook on microeconomics. As its proof is easy but elegant, we restate it here.

First Welfare Theorem (3.29) ([Arr51, Theorem 2], [Deb51])**.** *Every Walrasian equilibrium* (x, p) *is Pareto-efficient.*

Proof. Suppose (x, p) is a Walrasian equilibrium but not Pareto-efficient. Then there exists a feasible allocation x^* such that $v_i(x_i^*) \geq v_i(x_i)$ and there is some buyer $j \in N$

for which this inequality is strict. However, we then must also have $\langle p, x_j^* \rangle > \langle p, x_j \rangle$ as otherwise x cannot have been Walrasian since for all $i \in N$, we also have $v_i(x_i) - \langle p, x_i \rangle \geq v_i(x_i^*) - \langle p, x_i^* \rangle$. In total, by summing over all $i \in N$, we obtain

$$\sum_{i \in N} (v_i(x_i) - \langle p, x_i \rangle) \geq \sum_{i \in N} (v_i(x_i^*) - \langle p, x_i^* \rangle).$$

Now observe that $\sum_{i \in N} \langle p, x_i \rangle = \langle p, b \rangle$ as x is Walrasian (in particular packing, i.e., for all $e \in E$, $\sum_{i \in N} x_i(e) \leq b(e)$ and for every e for which this inequality is strict, we must have $p(e) = 0$). Moreover, $\sum_{i \in N} \langle p, x_i^* \rangle \leq \langle p, b \rangle$ as x^* is also feasible (i.e., $\sum_{i \in N} x_i^*(e) \leq b(e)$ for all $e \in E$). This implies

$$\sum_{i \in N} v_i(x_i) \geq \sum_{i \in N} v_i(x_i^*)$$

in contradiction to our assumption that there exists a buyer j with $v_j(x_j^*) > v_j(x_j)$ and for all other buyers $i \in N \setminus \{j\}$ it holds that $v_i(x_i^*) \geq v_i(x_i)$. □

Corollary 3.30. *Every Walrasian equilibrium (x, p) maximizes utilitarian welfare.*

Corollary 3.30 also implies that we can use the VCG mechanism (see Theorem 1.31) to show that there is a direct revelation mechanism that is dominant strategy incentive compatible.

However, the following example shows that we in general do not obtain Vickrey prices when running the Flow Auction.

Example 3.4. Let $N = \{1, 2, 3\}$ and $E = \{e_1, e_2\}$. Say the demands of the buyers are $d_1 = d_2 = 2$ and $d_3 = 1$, and the supplies $b(e_1) = 3$ and $b(e_2) = 2$. The valuations per item are given by the following vectors: $\tilde{v}_1 = (3, 1)$, $\tilde{v}_2 = (2, 0)$, and $\tilde{v}_3 = (0, 1)$ It is easy to see that there are two (symmetric) feasible allocations ($x = ((2, 0), (1, 1), (0, 1))$ and $y = (1, 1), (2, 0), (0, 1))$) and they are both socially optimal, so let us consider x.

Then the net effect of buyer 1 on buyer 2 (i.e., the loss of utility that buyer 1 imposes on buyer 2 by her presence) is 2, so the only possible price for e_1 is 1 if prices should add up to Vickrey prices. Since buyer 2 also has to purchase an item of e_1 once, object e_2 needs to be subsidized (i.e., assigned a price of -1) to give her a total cost of 0. However, this would give buyer 3 total cost of -1, which is not consistent with the VCG mechanism.

As an immediate observation from the last example, we can also see that it would have been better for buyer 2 to submit her preferred bundles according to a false valuation

function $\tilde{v}_2' = (0, 0)$, which would have resulted in prices $p = (0, 0)$ and the same allocation. Hence, she would have gotten a payoff of 2. The prices we would have obtained by the auction with truthful reporting would have been $(2, 0)$ which yields a payoff of 0 for buyer 2. Thus, the Flow Auction does not incentivize truthful reporting of demand correspondences.

CHAPTER 4

Faster Dynamic Auctions via Polymatroid Sum

In this final chapter on auctions in this thesis, we allow the buyers to have even more general valuation functions. In fact, we only restrict the valuation functions to be *(strong) gross substitutes*, which in some sense is the least restrictive class that we can go to without losing the guarantee that Walrasian equilibria exist (and that they can be efficiently computed). Conveniently, the auction itself that we are going to use is essentially the same as before. We start at all-zero prices and iteratively increase the prices of items that are in excess demand until we reach an equilibrium. The only difficulty that arises when going from item-capped additive valuations to (strong) gross substitute valuations is that the excess demand sets cannot be found via a minimum cut in a flow network. In essence, this is due to the fact that with gross substitutes valuations, we do not have the possibility to rank item types into the four tiers and then allocate them in that order. As the name *gross substitutes* suggests, two (or more) items can be substitutes of each other which means that them combined are not as valuable as the sum of the individual valuations.

In this chapter, we present how we can determine excess demand and excess supply sets for ascending and descending auctions (and hybrids) if all buyers have (strong) gross substitute valuations. We use a very efficient and yet quite simple algorithm to solve a *polymatroid sum problem*, which in the case of unit supplies is just the matroid union problem. From the solution to those auxiliary problems, we can easily construct the excess demand set or excess supply set, respectively. After running the

ascending auction, we obtain the unique minimal Walrasian prices and conversely after running a descending auction—which works very similarly—we obtain the unique maximal Walrasian prices. The running time to solve the auxiliary problems and the construction of the required sets is also faster than the methods by Ausubel [Aus06], Murota, Shioura, and Yang [MSY13], and Shioura [Shi17]. We also discuss how the communication protocol has to be set up, which is more involved than for the auctions in Chapter 2 and 3.

4.1 The Economic Model

The market setting does not differ a lot from the one in Chapter 3. We still consider a market with m indivisible distinguishable *item types*, collected in a set E and *quantities* $b(e) \in Z_+$. This yields the set of possible *bundles* $[0, b]_{\mathbb{Z}} := \{x \in \mathbb{Z}_+^E \mid 0 \leq x(e) \leq b(e), e \in E\}$. We also still have n *buyers* from a set N but instead of a valuation per item type ($\tilde{v}_i(e)$) and a demand d_i, that define only item-capped additive valuations, we consider arbitrary (for the moment) *valuations* $v_i \colon [0, b]_{\mathbb{Z}} \to \mathbb{Z}_+$.

Recall from the previous chapter that given a price vector p, a player i's *indirect utility function*, i.e., the maximum utility she can achieve under those prices, is given by

$$V_i(p) := \max\{v_i(x) - \langle p, x\rangle : x \in [0, b]_{\mathbb{Z}}\}$$

and her *demand correspondence*, the set of bundles that achieve this maximum utility (*preferred bundles*), is defined by

$$\mathcal{D}_i(p) := \arg\max\{v_i(x) - \langle p, x\rangle : x \in [0, b]_{\mathbb{Z}}\}.$$

For the auctions that we discuss in this chapter (to obtain minimal and maximal equilibrium prices), we are particularly interested in inclusion-wise minimal and maximal bundles in her demand correspondence i.e.

$$\overset{\smile}{\mathcal{D}}_i(p) := \{x \in \mathcal{D}_i(p) \mid x - \chi_e \notin \mathcal{D}_i(p) \text{ for all } e \in E\}$$

and

$$\overset{\frown}{\mathcal{D}}_i(p) := \{x \in \mathcal{D}_i(p) \mid x + \chi_e \notin \mathcal{D}_i(p) \text{ for all } e \in E\}.$$

Definition 4.1. A valuation function $v_i\colon [0,b]_{\mathbb{Z}} \to \mathbb{Z}_+$ is *gross substitute* if for any two price vectors p and q with $p \leq q$, and any bundle $x \in \mathcal{D}_i(p)$, there exists some bundle $y \in \mathcal{D}_i(q)$ such that $x(e) \leq y(e)$ for all $e \in E$ with $p(e) = q(e)$. It is called *strong gross substitute* if additionally v_i is concave-extensible[1] and $x(E) \geq y(E)$ holds for those bundles.

There are multiple equivalent ways to define gross substitutes. From an economic perspective, the ones by Gul and Stacchetti [GS99] and Ben-Zwi [Ben17] are suitable. However, in particular relevant for our work is the equivalence to discrete convex functions (more precisely $M^\natural$-concave functions) which has been shown by Fujishige and Yang for the unit supply case [FY03] and Murota, Shioura and Yang for the general case of strong gross substitutes [MSY13] (see also Lemma 4.3). The class of gross substitute valuations is very important since Walrasian equilibria are guaranteed to exist for markets in which every buyer holds a gross substitute valuation as shown by Kelso and Crawford [KC82]. Moreover, Gul and Stacchetti [GS99] showed that gross substitutes are a largest class (containing unit demand valuations) that have this guarantee. For a broad overview, we also refer to the survey by Paes Leme [Pae17].

In the following we give a few example and non-examples of gross substitute valuations. For the sake of convenience, we may also use the set notation (instead of vectors) in examples, in the unit supply setting, i.e., if $b(e) = 1$ for all $e \in E$. That is, instead of $v_i(x_i)$ with a 0-1-vector x_i, we may write $v_i(X_i)$, where $X_i = \{e \in E \mid x_i(e) = 1\}$.

4.1.1 A Landscape of Valuation Functions

Gross substitutes are a very general class of valuation functions that essentially capture (1) the concept of decreasing marginal returns and (2) substitutability of similar goods. To illustrate these concepts, for (1) one can think of pieces of cake; The first piece of cake makes the one consuming it usually happy, maybe this is also still true for a second piece. However, eating too many pieces will inevitably lead to a stomach ache, so the value of later pieces of the cake will not be as high as the value of the first pieces. For (2), consider again two pieces of cake, one chocolate cake and one apple pie. One can have a preference for either one of the cakes but they still are substitutes in the sense that if one already owns a chocolate cake, the demand for apple cake decreases.

[1] A function $v_i\colon [0,b]_{\mathbb{Z}} \to \mathbb{Z}_+$ is *concave-extensible* if there exists a concave function $\hat{v}_i\colon \{x \in \mathbb{R}^n \mid 0 \leq x \leq b\} \to \mathbb{R}$ such that $v_i(x) = \hat{v}_i(x)$ for $x \in [0,b]_{\mathbb{Z}}$.

While this economic interpretation of gross substitutes (Definition 4.1) seems very general, it is clearly not the only sensible way how a valuation might look like. As a counterpart, one can also define *gross complements* for different goods that have (1) increasing marginal returns or (2) complement each other. To also illustrate those, for (1) think of a collection of coins (stamps, trading cards, etc.). A complete collection, say all State quarters of the United States, might be considered worth more than its nominal value of 12.5 USD as a collector would not have to search for missing coins. On the other hand, an incomplete collection would have a value that is just the nominal value of the coins in the set. For (2), one can also think of two goods as pencil and eraser. The pencil might just have a value on its own but the eraser is mostly useless if its owner does not also own a pencil.

Indeed among Gul and Stacchetti's alternative definitions of gross substitute valuations [GS99], we can also find the *no complementaries* condition which outrules the existence of complementing goods.

Now we give a few examples of gross substitute valuations. It is easy to recognize that *unit demand* valuations ($v_i(S) = \max_{e \in S} \tilde{v}_i(e)$) and *additive* valuations ($v_i(S) = \sum_{e \in S} \tilde{v}_i(e)$) (also with item caps, cf. Chapter 3) are all gross substitute. Another easy examples are *symmetric concave* valuations which have the form $v_i(S) = h(|S|)$ for some monotone concave function $h: \mathbb{Z}_+ \to \mathbb{Z}_+$.

A less obvious examples are *weighted matroid rank* functions that are determined by a matroid $M_i = (E, \mathcal{I}_i)$ and weights $\tilde{v}_i: E \to \mathbb{Z}_+$. A valuation of a subset $S \subseteq E$ is then $v_i(S) = \max_{I \subseteq \mathcal{I}_i} \sum_{e \in S \cap I} \tilde{v}_i(e)$. As a last example we mention OXS valuations, introduced by Lehmann, Lehmann, and Nisan [LLN06], which can be described by a bipartite graph $G_i = (P \cup E, A)$ and edge weights $w: A \to \mathbb{Z}_+$. The valuation of S is then given by $v_i(S) = \max \left\{ \sum_{\{k,e\} \in \mu} w(k,e) : \mu \text{ matching in } G_i \right\}$.

Figure 4.1 indicates how the different examples of valuation functions relate to each other. These inclusions all follow from either Gul and Stacchetti [GS99] or Lehmann, Lehmann, and Nisan [LLN06].

The following example also demonstrate why we restrict ourselves to (strong) gross substitute valuations and do not make any further statements to complementaries as Walrasian equilibria are not guaranteed to exist if there is a buyer with a valuation function that is not gross substitute.

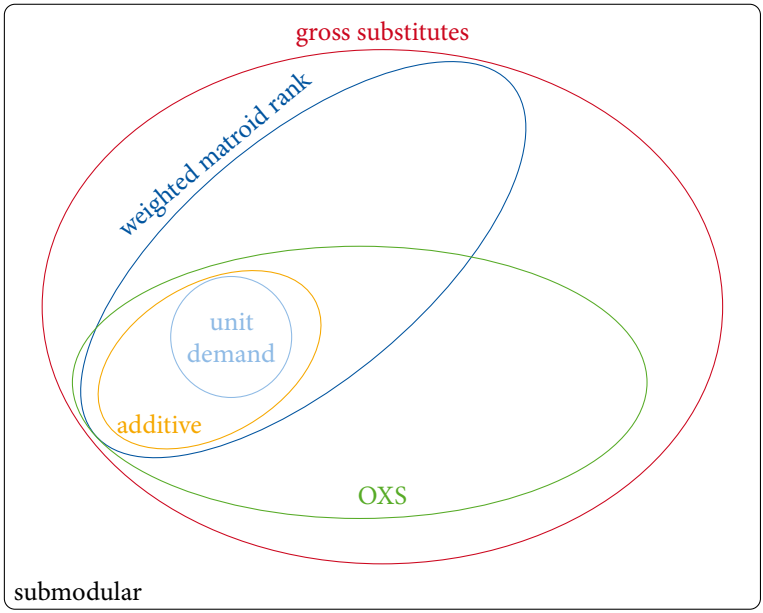


Figure 4.1: Landscape of valuation functions. All inclusions are strict.

Example 4.1. Consider an example with two players $N = \{1, 2\}$ and three items $E = \{e_1, e_2, e_3\}$ with unit supply. The valuation functions of the players are as follows (note that they are not gross substitutes):

$$v_1(S) = \begin{cases} 2 & \text{if } \{e_1, e_2\} \subseteq S, \\ 1 & \text{if } \{e_1, e_2\} \nsubseteq S, \{e_3\} \subseteq S, \\ 0 & \text{otherwise,} \end{cases} \qquad v_2(S) = \begin{cases} 2 & \text{if } \{e_2, e_3\} \subseteq S, \\ 1 & \text{if } \{e_2, e_3\} \nsubseteq S, \{e_1\} \subseteq S, \\ 0 & \text{otherwise.} \end{cases}$$

Here $p = (0, 0, 0)$ is not packing since the unique preferred bundle of buyer 1 is $\{e_1, e_2\}$, while the unique preferred bundle of buyer 2 is $\{e_2, e_3\}$. The vector $p = (0, 1, 0)$ is packing but it is not covering, since item e_2 is not sold although it has a positive price. All other prices are clearly also not covering.

4.1.2 Differences to Chapter 3

In the previous chapter, we restricted ourselves to item-capped additive valuation functions, which indeed are a subclass of gross substitutes. In this chapter, we tackle the whole class of gross substitutes with different dynamic auctions (ascending, descending, and hybrids of both). While the overall framework stays the same, in particular the Generic Ascending Auction by Gul and Stacchetti as a foundation for our auctions, we have to adapt our method to find the excess demand sets (and excess supply sets for price decrements, e.g., in the descending auction). The main reason for that item types cannot be ordered into tiers such that we can pick the items from top of the order to construct a preferred bundle for a buyer. In contrast, with general gross substitutes, we can also have the following: Let $e, f \in E$ be two different item types that a buyer i values very highly on their own; however, if i already owns e, she might not have any demand for f as e and f are substitutes (e.g., two cars of similar size but different brands). While there is still a greedy algorithm that constructs a preferred bundle, it has to be more dynamic regarding the order items are picked. This also has the consequence that a flow algorithm cannot be used anymore to determine excess demand sets: figuratively speaking, to keep a flow-based algorithm, we would need some gadget that allows us to adjust capacity on one edge because there is flow on another edge, which is not something that conventional maximum flow algorithms are able to handle. Instead, we introduce another algorithm that can handle the full (poly)matroidal structure that is contained in gross substitute valuations.

4.1.3 Contribution

The contribution of this chapter are twofold: we provide faster auction algorithms, and we present structural results on price monotonicity.

In the same vein as Murota, Shioura, and Yang [MSY13], we exploit the special structure of the *Lyapunov* to obtain even faster running times. We view the submodular minimization problem from a dual viewpoint as a *matroid union problem* (for the unit supply case) or a *polymatroid sum problem* (for the multi-supply case), and use a fast (and also simple) push-relabel algorithm for this problem. Let us denote by DO the time needed for each *demand oracle query* and by ExO the time of each *exchange oracle query* (see Section 4.3 for definitions and discussion). Then, our algorithm runs in time $\mathcal{O}(n \cdot \mathsf{DO} + nm^3 \cdot \mathsf{ExO})$ in the multi-supply case. In comparison, the algorithm

in [MSY13] runs in time $\mathcal{O}(n \cdot \mathsf{DO} + nm^4 \log(nmB) \cdot \mathsf{ExO})$, where B is the maximum number in the supply vector b. For the unit supply case, the problem becomes the classical *matroid union problem*, and we obtain an even better running time of $\mathcal{O}(n \cdot \mathsf{DO} + (m^3 + nm^2) \cdot \mathsf{ExO})$. Note that our algorithm is significantly simpler than the one in [MSY13].

An auxiliary *exchange graph* with respect to some optimal solution will then reveal a minimal excess demand set of maximum shortage (or a minimal excess supply set of maximum abundance) which we can use for our auction setting.

Our push-relabel algorithm is a special implementation of the more general submodular flow feasibility algorithm by Frank and Miklós [FM12]. In this context, our contribution is giving an efficient implementation in terms of the number of queries to an oracle. The description in [FM12] is generic and gives bounds in terms of *basic operations*. Implementing a single such operation may take $\mathcal{O}(nm \cdot \mathsf{ExO})$; however, we show that this can be amortized over a sequence of basic operations. See Section 4.1.4 on further discussion of push-relabel algorithms.

Our second main contribution is on price monotonicity. We make a clear conceptual distinction between packing, covering and Walrasian prices. For example, it is not clear a priori whether there may exist a packing price vector q with $q(e) < p_*(e)$ for some items $e \in E$, where p_* denotes the buyer-optimal Walrasian price vector, or whether there may exist a covering price vector q' with $q'(e) > p^*(e)$ for some items $e \in E$, where p^* is the maximal Walrasian price vector. Theorem 4.37 states that in fact $p_* \leq q$ for all packing price vectors q and Theorem 4.42 states that $p^* \geq q$ for all covering price vectors q.

Building on this result, we can also prove in Section 4.8 that the minimal and maximal Walrasian prices, respectively, react naturally to changes in supply and demand, i.e., if total supply of items decreases or the total demand of buyers increases, the minimal and maximal Walrasian prices can only increase. Independently, Raach [Raa24] proved the same monotonicity results.

This chapter is based on the paper *Faster Dynamic Auctions via Polymatroid Sum* which was published in the ACM journal *Transactions on Economics and Computation* in 2025 [Eic+25] but it is also available as a preprint [Eic+24b]. A previous version of the paper [Eic+23b] was also accepted at *The 19th Conference on Web and InterNet Economics* in 2023. The difference between the two versions is that Meike Neuwohner joined as a co-author and we extened together the results on price monotonicity for maximal Walrasian prices (see above). We give a complete description of the algorithms and

auctions and prove their correctness and running time guarantees, also with a special focus on communication costs (oracles) and the different implementation details for unit-supply and multi-supply settings. We only briefly sketch the structural results for weaker notions of equilibria and price monotonicity (which follow from the former). These parts will be subject of the PhD thesis of Katharina Eickhoff but can of course be already found in the published paper [Eic+25].

4.1.4 Related Work

In this section, we give a literature review over the topics related to our economic model and equilibrium concept. For an excellent overview over (strong) gross substitutes, Walrasian equilibria, and ascending auctions we refer to the survey by Paes Leme [Pae17].

Gross substitutes and Walrasian equilibria Gross substitute valuations are the frontier in mechanism design both from an economic and computational perspective. Kelso and Crawford [KC82] showed in their seminal paper that a Walrasian equilibrium is guaranteed to exist if all valuation functions satisfy the gross substitutes condition that in layman's terms can be stated as, *"the demand for an item does not not decrease, if only the prices of other items are increased"*.[2] Roughly speaking, the study of economic models from an algorithmic and complexity theoretical point of view really started off in the 1990s when also the foundations of algorithmic game theory were laid out. Gul and Stacchetti [GS99] showed with their Maximum Domain Theorem that indeed gross substitutes are a maximal class of valuation functions (containing unit demand valuations) that guarantee the existence of Walrasian equilibria. Additionally, they provided equivalent definitions for the gross substitutes condition and showed that Walrasian prices form a complete lattice, which implies that there exist unique component-wise minimal and maximal Walrasian price vectors. There are additional equivalent characterization of gross substitutes by Ben-Zwi [BLN13]. However, the characterization by Fujishige and Yang [FY03] turns out to be most useful from a mathematical point of view. It states that a valuation function has the gross substitutes property if and only if it is $M^\natural$-concave, which allows for the usage of powerful tools from discrete convexity. This result also has been extended to strong gross substitutes valuation functions by

[2] A formal definition is provided in Section 1.

Murota, Shioura, and Yang [MSY13]. The differences between strong gross substitutes and "weak gross substitutes" is discussed by Milgrom and Strulovici [MS09].

There are other classes than (strong) gross substitutes for which Walrasian prices are guaranteed to exist [SY09; BLN13] but these classes do not contain the natural unit demand valuations. On the other hand, for certain subclasses of gross substitute valuations such as unit demand valuations [DGS86; MT10] and item-capped additive valuations (see the previous chapter and [Eic+24a]) for which the structure of Walrasian prices has been described more explicitly.

Dynamic Auctions or Walrasian tâtonnement Léon Walras, the namesake of our equilibrium concept, already proposed how equilibria may be found, namely by a 'tâtonnement'[3] process. This procedure basically describes a dynamic auction: an auctioneer posts prices and unless these prices are at equilibrium, the auctioneer makes an adjustment. The first modern study of such a process was done by Demange, Gale, and Sotomayor [DGS86] for unit demand valuations. In Gul and Stacchetti's follow-up work [GS00], they gave the framework for ascending auctions for gross substitutes valuations which we already used in Chapter 3; that is, start at all-zero prices and increase prices on an inclusion-wise minimal excess demand set of maximum shortage until there is no excess set anymore. They proved that such an auction always terminates with minimal Walrasian prices. However, while they showed that an excess demand set of items has to exist whenever the prices are not yet at equilibrium, they left it open how to compute those sets. This gap was initially closed by Ausubel [Aus06; Aus05] using submodular function minimization on a discrete version of the Lyapunov function (a submodular potential function, introduced by Varian [Var81] for divisible goods) which is minimized at Walrasian prices. If all valuation functions are gross substitutes, then this function is submodular and its minimum (and every steepest descent direction, which corresponds to an excess demand set of maximum shortage) can be found efficiently using submodular function minimization, which can be done efficiently (see [GLS81; IFF01; Iwa03; IO09; LSW15; Cha+17]). Murota, Shioura, and Yang [MSY13] showed that the Lyapunov function is not just submodular but $L^\natural$-convex, which allows for even faster methods than plain submodular function minimization. Note that these methods in [Aus06; MSY13] allow for descending auctions or other kinds of dynamic auctions, where prices maybe adjusted in a non-monotone fashion. Similar auctions and

[3] French: *trial-and-error*

guarantees for termination with minimal and maximal Walrasian prices are discussed in [AAT13] and [BLN13].

We should mention that the literature mentioned above follows the definition of a dynamic auction as proposed by Gul and Stacchetti [GS00]. However, the literature also explores other auction designs, e.g., in [BO02; Aus04; PU00; AM02] that consider slightly different market settings or different ways for buyers to report their demands to the auctioneer.

A weak point of dynamic auctions would be their running time if we assume full information for the auctioneer; e.g., in an ascending auction $\|p_*\|_\infty$ price increase steps are needed, where p_* is the (minimal) Walrasian price vector computed by the auction [MSY13]. This process can be speeded up by increasing prices on an excess demand set not only by one but by the maximum possible amount before the steepest descent direction of the Lyapunov function changes. This results in at most nmB price adjustment steps (c.f. [Shi17, Proposition 4.17]). Further speed ups of the ascending auction as a rounding scheme are discussed in [Pae17, Section 10.1.]. There are other algorithms which can compute Walrasian prices efficiently, for instance by Paes Leme and Wong [PW20]. However, in contrast to dynamic auctions, these algorithms need direct information of the valuation functions, i.e., by a value oracle.

Oracles and communication complexity To compute Walrasian prices, it is crucial to get some information on buyers' valuation functions. Usually, the necessary communication between auctioneer and buyers is modeled via oracle calls. Depending on which oracle type is used, one can get different information. A comparison of the computational power of some oracle types can be found, e.g., in [BN10]. In literature, typically, value oracles and demand oracles are used for dynamic auctions. In case of direct mechanisms, there is also the concept of *bidding languages* as introduced by Nisan [Nis00]. However, those are necessarily very exhaustive in their space requirements. A essentially optimal bidding language for gross substitute valuations is due to Klemperer [Kle09] who developed the Product-Mix Auction for the Bank of England which uses the auction to trade collaterals such as UK soverign debt or mortag-backed securties. More recently, in collaboration with Baldwin, Bichler, Fichtl [Bal+24], Baldwin and Lock [BKL24] the efficiency of this auction is shown for strong substitute valuation functions. Lock, Goldberg, and Marmolejo-Cossío [LGM22] also show how to learn the efficient encoding of strong substitutes using the Product-Mix Auction bidding language via demand and valuation oracles. For our dynamic auctions, we require a

demand oracle and an exchange oracle as those work naturally in our proposed algorithm to find the required excess demand or excess supply sets. The exchange oracle is related to the dynamic rank oracle [Bli+23], as both provide information on sets close to already requested ones. A detailed overview which oracles we use and how they can be compared is given in Section 4.3.

Incentives As a corner stone of auction theory and mechanism design, we have the VCG Theorem [Vic61; Cla71; Gro73] that states that there is a sealed-bid mechanism that has truthful bidding as a dominant strategy. However, as briefly discussed in the introduction, ascending auctions are often preferred as they yield a more natural and transparent price-finding process. As a consequence, mathematical economists went to great lengths to implement the VCG mechanism in an ascending auction [Aus04; Aus06; VSV07; Bik+01; Bik+11; MP07; SY14; COP15]. Kern, Manthey, and Uetz [KMU16] showed that for unit demand valuations, the auction by Demange, Gale, and Sotomayor [DGS86] mentioned above indeed returns Vickrey prices. If we go beyond unit demand valuations, Vickrey prices cannot be achieved using a single price trajectory in a dynamic auction, as shown by Gul and Stacchetti [GS00] since the auction does not gain enough information during the iterations to determine Vickrey prices. However, there exist variants of dynamic auctions that circumvent this and can obtain Vickrey prices [Aus06; SY14; COP15], e.g., by the use of proxy auctions, which compute multiple price trajectories.

In a multi-supply setting, we already noted in Chapter 3 ([Eic+24a]) that Vickrey prices that are Walrasian and in which every copy of an item has the same cost, do not necessarily exist.

Discrete convexity: matroid union and polymatroid sum We refer the interested reader to the monographs by Murota [Mur03] and Fujishige [Fuj05]. There is also a survey on the connections between discrete convexity and auction theory by Shioura [ST15].

As pointed out above, all buyers having (strong) gross substitutes valuation functions means that we can use methods from discrete convex optimization to solve the computational problem that arises during a dynamic auction. In particular, the whole auction can be viewed as an iterative process to minimize a submodular (and even $L^\natural$-convex) function [Aus06; MSY13; MSY16; Shi17]. As we will see in the sequel, these methods can be quite complicated. For the unit supply case and gross substitutes valuations,

the problem of finding a minimal excess demand set of maximum shortage becomes equivalent to the matroid union problem on n matroids, which is efficiently solvable as shown by Edmonds [Edm68]. Over time, more efficient algorithms were developed by Knuth [Knu73], Greene and Magnanti [GM75], Cunningham [Cun86], Chakrabarty, Lee, Sidford, Singla, and Wong [Cha+19], and most recently by Terao [Ter23] and Blikstad, Mukhopadhyay, Nanongkai, and Tu [Bli+23]. Note that some of these methods are hard to compare due to the use of different oracles.

In the more general setting with multi-supply and strong gross substitutes valuations, we need to consider the more general polymatroid sum problem.

Push-relabel algorithms The algorithms mentioned above to solve the matroid union and the polymatroid sum problem are so called *augmenting path algorithms*. They start by taking a trivially feasible solution (e.g., the empty set or the all-zero vector, respectively) and augment it by finding additional elements that can be added to the current solution or $(k+1)$-to-k-swaps to increase the size of the solution until it is optimal. However, more recently, so called *push-relabel algorithms* have been studied for various combinatorial optimization problems and they can outperform augmenting path algorithms both in theory as well as in practice. The push-relabel paradigm was introduced for the maximum flow problem by Goldberg and Tarjan [GT88]. The idea is in some sense dual to augmenting path algorithms: One starts with a possibly infeasible solution which otherwise satisfies some optimality condition. The push and relabel operations then maintain this optimality condition while making the solution "less infeasible" over time. We use a push-relabel algorithm by Frank and Miklós [FM12] for matroid union. Our algorithm in Section 4.4 gives an efficient implementation in the auction setting and also generalizes to polymatroid sum and M-convex sets in general. It is interesting to note that the level functions and local exchanges of push-relabel algorithms have some resemblance to auction algorithms, as already pointed out by Bertsekas [Ber92]. In our push-relabel algorithm in Section 4.4, one could interpret the level $\ell(e)$ of an item as a small marginal price discount used for tie-breaking.

4.2 Background Theory

In this section, we give important definitions and theorems from discrete convex analysis which we will use in four different dynamic auctions in order to find excess

demand or excess supply sets. We refer to the monographs by Murota [Muro03] and Fujishige [Fujo5] for more details. Remarkably, these definitions and derived theoretical and practical results have a very natural direct economical interpretation, in particular in the context of dynamic auctions, which we will discuss in the sequel.

Definition 4.2. A valuation function $v_i\colon [0,b]_{\mathbb{Z}} \to \mathbb{Z}_+$ is $M^\natural$-*concave* if for any $x, y \in [0,b]_{\mathbb{Z}}$ and for any $e \in \operatorname{supp}^+(x-y)$, there exists some $f \in \operatorname{supp}^-(x-y) \cup \{\varnothing\}$ it holds that

$$v_i(x) + v_i(y) \leq v_i(x - \chi_e + \chi_f) + v_i(y - \chi_f + \chi_e).$$

Lemma 4.3 ([MSY13, Theorem 1.5])**.** *A valuation function is strong gross substitute if and only if it is* $M^\natural$*-concave.*

Definition 4.4. A set of bundles $\mathcal{B} \subseteq [0,b]_{\mathbb{Z}}$ is M-*convex* if for any $x, y \in \mathcal{B}$ and for any $e \in \operatorname{supp}^+(x-y)$, there exists $f \in \operatorname{supp}^-(x-y)$ such that $x - \chi_e + \chi_f \in \mathcal{B}$ and $y - \chi_f + \chi_x \in \mathcal{B}$.

Lemma 4.5. *If* $v_i\colon [0,b]_{\mathbb{Z}} \to \mathbb{Z}_+$ *is strong gross substitute, then* $\overline{\mathcal{D}}_i(p)$*, and* $\widehat{\mathcal{D}}_i(p)$ *are* M*-convex sets for any price vector* $p \in \mathbb{Z}_+^E$*.*

Proof. We only show the statement for $\overline{\mathcal{D}}_i(p)$, the argument for $\widehat{\mathcal{D}}_i(p)$ is analogous. By Lemma 4.3, the valuation function v_i is $M^\natural$-concave. Thus, the utility function $u_i(x,p) = v_i(x) - \langle p, x\rangle$ is also $M^\natural$-concave in x as the sum of an $M^\natural$-concave and a modular function. Let $p \in \mathbb{Z}_+^E$ and $x, y \in \overline{\mathcal{D}}_i(p)$. Let $e \in \operatorname{supp}^+(x-y)$. Then there exists $f \in \operatorname{supp}^-(x-y) \cup \{\varnothing\}$ such that $u_i(x,p) + u_i(y,p) \leq u_i(x - \chi_e + \chi_f, p) + u_i(y - \chi_f + \chi_e, p)$. Thus, $2V_i(p) \leq u_i(x - \chi_e + \chi_f, p) + u_i(y - \chi_f + \chi_e, p)$. If the inequality was strict, then at least one of the terms must be strictly larger that $V_i(p)$, which is not possible. Hence, the inequality is an equality and both terms have to be equal to $V_i(p)$. Thus, $x - \chi_e + \chi_f, y - \chi_f + \chi_e \in \mathcal{D}_i(p)$.

We still need to show that no other bundle strictly contains x' and y'. So let $x, y \in \overline{\mathcal{D}}_i(p)$ and $x' = x - \chi_e + \chi_f$ for e and f as above. We need to show that x' is not covering and is not covered by any $z \in \overline{\mathcal{D}}_i(p)$. Assume towards a contradiction that there exists some $z \in \overline{\mathcal{D}}_i(p)$ with $z \leq x'$ but $z \neq x'$. Then, there exists some $g \in E$ with $z(g) < x'(g)$. Since $x \in \overline{\mathcal{D}}_i(p)$, we must have $z(f) = x(f) + 1$ as otherwise $z \leq x$ and $z \neq x$ as well, a contradiction.

Moreover, $z(e) \leq x(e) - 1$ as $z \leq x'$. Thus, $f \in \operatorname{supp}^+(z - x) = \{f\}$ and hence, by $M^\natural$-concavity of u_i, we have

$$u_i(z, p) + u_i(x, p) \leq u_i(z - \chi_f + \chi_g, p) + u_i(x - \chi_g + \chi_f, p)$$

for some $g \in \operatorname{supp}^-(z - x) \cup \{\varnothing\}$. Note that $z - \chi_f + \chi_g \in \mathcal{D}_i(p)$ by the same argument as above. By minimality of x, we must have $g \neq \varnothing$, however. By definition of z and g, we have $z - \chi_f + \chi_g \leq x$. As x is minimal in $\mathcal{D}_i(p)$, we must then also have $x = z - \chi_f + \chi_g$. As $z(e) < x(e)$, this implies $g = e$. However, this yields the contradiction $z(e) + 1 \leq x'(e) < x(e)$. □

Lemma 4.6. *If v_i is a gross substitute valuation function, then*

(i) $\check{\mathcal{D}}_i(p) = \arg\min\{\|x\|_1 : x \in \mathcal{D}_i(p)\}$ *and*
(ii) $\hat{\mathcal{D}}_i(p) = \arg\max\{\|x\|_1 : x \in \mathcal{D}_i(p)\}$.

Proof. We only prove the first statement, the other proof is analogous.

The $\supseteq$ inclusion is trivial. For the $\subseteq$ part choose any $y \in \arg\min\{\|x\|_1 : x \in \mathcal{D}_i(p)\}$ with $\|y\|_1 < \|x\|_1$ for some $x \in \check{\mathcal{D}}_i(p)$. In particular, choose x such that its Hamming distance is closest to y. Then there exists some $e \in \operatorname{supp}^+(x - y)$ and hence by $M^\natural$-convexity of $\check{\mathcal{D}}_i(p)$ (see Lemma 4.5) some $f \in \operatorname{supp}^-(x-y)\cup\{\varnothing\}$ such that $x' = x-\chi_e+\chi_f \in \check{\mathcal{D}}_i(p)$. Independent of the choice of f, the Hamming distance between x' and y is shorter than the distance between x and y, a contradiction. □

Lemma 4.7 ([Muro3, Theorem 4.15])**.** *Let $\mathcal{B} \subseteq [0, b]_{\mathbb{Z}}$ be an* M*-convex set. Let $\rho\colon 2^E \to \mathbb{Z}_+$ be a mapping*

$$\rho(S) = \max\{x(S) : x \in \mathcal{B}\}. \tag{4.1}$$

Then ρ is an integer valued submodular set function and S is the set of integer points in the corresponding base polytope. That is,

$$\mathcal{B} = \{x \in \mathbb{Z}_+^E \mid x(S) \leq \rho(S) \text{ for all } S \subseteq E, x(E) = \rho(E)\}.$$

Let us summarize the above. If v_i is gross substitute, then it is also $M^\natural$-concave which leaves us in a mathematically good-natured place. In particular, the demand correspondences $\mathcal{D}_i(p), \check{\mathcal{D}}_i(p), \hat{\mathcal{D}}_i(p)$ are M-convex sets, which means that after removing an item from a demand set, we can just augment with a different item (or none) to get back to a demand set. One might argue that this is a more direct definition for the

term "gross substitutes". Moreover, $\check{\mathcal{D}}_i(p)$ and $\hat{\mathcal{D}}_i(p)$ are sets of integer points in a polymatroid base polytope.

Definition 4.8. For an M-convex set $\mathcal{B} \subseteq [0, b]_{\mathbb{Z}}$ with associated polymatroid rank function ρ as defined in (4.1), and a vector $x \in \mathcal{B}$, we say that a set $S \subseteq E$ is *tight* if $x(S) = \rho(S)$. We denote the *collection of tight sets* w.r.t. $\mathcal{B}$ and x by $\mathcal{T}_{\mathcal{B}}(x) := \{S \subseteq E \mid x(S) = \rho(S)\}$.

In the context of a demand correspondence, an set of item types is tight w.r.t. x if we cannot add further items of those types to x without leaving the demand correspondence.

Lemma 4.9. *Given an* M*-convex set $\mathcal{B}$ with associated rank ρ and $x \in \mathcal{B}$, the collection of tight sets $\mathcal{T}_{\mathcal{B}}(x)$ is closed under union and intersection. Consequently, there also exists a unique minimal and maximal tight set (in particular for every $e \in E$ there is one that contains e).*

Proof. Let $S, T \in \mathcal{T}_{\mathcal{B}}(x)$. Then, by submodularity of ρ and the fact that $x \in \mathcal{B}$,

$$\begin{aligned} x(S) + x(T) &= \rho(S) + \rho(T) \\ &\geq \rho(S \cup T) + \rho(S \cap T) \\ &\geq x(S \cup T) + x(S \cap T) \\ &= x(S) + x(T). \end{aligned}$$

Hence, $x(S \cup T) = \rho(S \cup T)$ and $x(S \cap T) = \rho(S \cap T)$, i.e., the union and intersection of S and T are tight. □

For an M-convex set $\mathcal{B}$, a bundle $x \in \mathcal{B}$, and an item type $e \in E$, we denote the unique minimal tight set containing e by

$$\mathcal{T}_{\mathcal{B}}(x, e) := \bigcap_{\substack{S \in \mathcal{T}_{\mathcal{B}}(x) \\ e \in S}} S.$$

We will show in Lemma 4.12 that $\mathcal{T}_{\check{\mathcal{D}}_i(p)}(x, e)$ and $\mathcal{T}_{\hat{\mathcal{D}}_i(p)}(x, e)$ contain exactly those item types $f \in E$, that admit an exchange with e w.r.t. a minimal or maximal preferred bundle x, respectively. That is, $x - \chi_e + \chi_f \in \check{\mathcal{D}}_i(p)$ or $x - \chi_e + \chi_f \in \hat{\mathcal{D}}_i(p)$, respectively. Since we consider gross substitute valuations, we know that the demand correspondence $\mathcal{D}_i(p)$ for any $p \in \mathbb{Z}_+^E$ is M-convex. Hence, to check whether a price vector p is

packing or covering, we only need to check whether there exists a packing or covering, respectively, that uses one demand set per buyer. More precisely, we are interested in a *partitioning* (i.e., an allocation that is both packing and covering) of $[0, b]_{\mathbb{Z}}$ into vectors $x_1, \dots, x_n$ from $\mathcal{D}_1(p), \dots, \mathcal{D}_n(p)$. So it is not surprising that an auction algorithm will also involve an algorithm to compute a partitioning of $[0, b]_{\mathbb{Z}}$ from M-convex set. However, we also use the same algorithm in a less obvious way: in our approach the algorithm for M-convex partitioning does not only compute an allocation, if one exists, but also computes a witness (or dual solution) in case it does not exists. We show that this witness happens to be an excess demand or excess supply set in which we want to adjust the prices in our dynamic auctions.

In particular, we consider the following general problem that has to be solved in every iteration of any of the dynamic auctions (which still follow the same paradigm as the Generic Ascending Auction).

M-convex Set Packing

Given: M-convex sets $\mathcal{B}_1, \dots, \mathcal{B}_n \subseteq [0, b]_{\mathbb{Z}}$.
Find: vectors $x_1 \in \mathcal{B}_1, \dots, x_n \in \mathcal{B}_n$ such that $\sum_{i=1}^{n} x_i \leq b$.

This problem generalizes the following two:

Matroid Union

Given: matroids $M_1 = (E, \mathcal{I}_1), \dots, M_n = (E, \mathcal{I}_n)$.
Find: partition of independent sets $I_1 \in \mathcal{I}_1, \dots, I_n \in \mathcal{I}_n$.

Polymatroid Sum

Given: polymatroids $\mathcal{M}_1 = (E, \rho_1), \dots, \mathcal{M}_n = (E, \rho_n)$, capacities $b(e)$ for all $e \in E$.
Find: vectors $x_1 \in \mathcal{P}(\rho_1), \dots, x_n \in \mathcal{P}(\rho_n)$ such that $\sum_{i=1}^{n} x_i \leq b$.

Min-Max Theorem for Polymatroid Sum (4.10). *Let E be a finite set, $N = \{1, \dots, n\}$, and $\mathfrak{B} = \times_{i \in N} \mathcal{B}_i$, where for all $i \in N$, $\mathcal{B}_i \in [0, b]_{\mathbb{Z}}$ is an M-convex sets with associated rank function ρ_i. Then*

$$\max_{x \in \mathfrak{B}} \left\{ \sum_{e \in E} \min \left\{ \sum_{i \in N} x_i(e), b(e) \right\} \right\} = \min_{F \subseteq E} \left\{ \sum_{i \in N} \rho_i(E \setminus F) + b(F) \right\}. \tag{4.2}$$

Proof. We first show that

$$\sum_{e \in E} \min \left\{ \sum_{i \in N} x_i(e), b(e) \right\} \leq \sum_{i \in N} \rho_i(E \setminus F) + b(F)$$

holds for all $x \in \mathfrak{B}$ and all $F \subseteq E$ and hence, also for the maximum over all $x \in \mathfrak{B}$ and minimum over all $F \subseteq E$. Let $x \in \mathfrak{B}$ be an arbitrary vector and $F \subseteq E$ be an arbitrary set. Then we have

$$\begin{aligned} \sum_{i \in N} \rho_i(E \setminus F) + b(F) &\geq \sum_{i \in N} x_i(E \setminus F) + b(F) \\ &= \sum_{e \in E \setminus F} \sum_{i \in N} x_i(e) + \sum_{e \in F} b(e) \\ &\geq \sum_{e \in E} \min \left\{ \sum_{i \in N} x_i(e), b(e) \right\}. \end{aligned} \tag{4.3}$$

The first inequality holds with equality if and only if $x_i(E \setminus F) = \rho_i(E \setminus F)$ for all $i \in N$. The second inequality holds with equality if and only if for all $e \in F$, $\sum_{i \in N} x_i(e) \geq b(e)$ and for all $e \notin F$, $\sum_{i \in N} x_i(e) \leq b(e)$.

It remains to show that such a pair (x, F) that satisfies the inequalities in (4.3) with equality always exist. We postpone this part of the proof and show it as part of Lemma 4.17, which explicit constructs x and F as desired with our Push-Relabel Algorithm. □

Observe that (4.3) from the proof expresses some complementary slackness conditions (cf. duality in linear programming), which we summarizes in the following corollary.

Corollary 4.11. *Given a collection of* M*-convex sets* $\mathfrak{B} = \times_{i \in N} \mathcal{B}_i$ *with associated rank functions* ρ_i *for all* $i \in N$*, a vector* $x \in \mathfrak{B}$ *is optimal for* M-convex Set Packing *if and only if there exists a set* $F \subseteq E$ *with*

(1) *for all* $e \in F$, $\sum_{i \in N} x_i(e) \geq b(e)$,
(2) *for all* $e \notin F$, $\sum_{i \in N} x_i(e) \leq b(e)$, *and*
(3) *for all* $i \in N$, $x_i(E \setminus F) = \rho_i(E \setminus F)$.

4.3 Demand Correspondences and Oracles

Typically, the running times of algorithms is measured in terms of the input size. However, this becomes impractical when the input is too large. For instance, to determine

the maximum flow in a capacitated network (as we do for the Flow Auction), we only need a graph and the capacities on each edge as input. This can be encoded by a single matrix and recall from Chapter 3 that the size of the graph used in the Flow Auction was just bilinear in the number of items and buyers which exploited the simple structure of the demand correspondence that arises with item-capped additive valuation functions. For the more general gross substitute valuation functions that we consider in this chapter, this simple structure is lost and as a corollary of a theorem by Knuth [Knu74], it is generally impossible to encode the demand correspondence—or more general, an M-convex set—in space that is polynomial in the number of items. Hence, evaluating the running time in the input size is not the most forthright thing to do: even the brute-force running time would be bilinear then (assuming arithmetic operations and comparisons are done in constant time), while it is exponential in n and m.

Thus, it is more appropriate to measure the running time of the algorithms and auctions rather directly in n, m, and a number of *oracle queries*, i.e., queries to get information about the solution space that are not part of the input. With some justification, one can call an algorithm with access to an oracle efficient if its running time is polynomial in n, m, and the number of oracle queries. From a computational point of view, this seems like a fair measure as such an oracle query essentially resolves a computational step that is understood to be efficient (like finding *some* base of a polymatroid). However, the treatment of those computations as oracle query also especially makes sense in our setting of a dynamic auction, where the M-convex sets (the demand correspondences of the buyers) are supposed to be private information that the central computing entity (the auctioneer) should not have full access to. Moreover, the computation of those sets are not really part of the auction itself and rather something that is supposed to be done by the buyers on their own.

This section introduces two types of oracles that our algorithms will use to solve the aforementioned problem M-convex Set Packing and its two special cases Matroid Union and Polymatroid Sum. In the auction setting it is also opportune to view a query to an oracle as communication with a specific buyer i about her demand correspondence. For the dynamic auctions that we consider later, we are particularly interested in $\widetilde{\mathcal{D}}_i(p)$ and $\widehat{\mathcal{D}}_i(p)$ so it is convenient to think of those sets when we talk about the M-convex sets $\mathcal{B}_i$ and minimal or maximal preferred bundles x_i from those sets in the sequel.

The first type of oracle we call *demand oracle* which returns one (arbitrary) $x_i \in \mathcal{B}_i$ for some $i \in N$. When discussing running times, we will use DO as a placeholder for the

time that a single demand oracle queries takes. In terms of a dynamic auction, a demand query presents prices $p(e)$ for all $e \in E$ to the buyer who then responds with a set in her demand correspondence, i.e., a vector x_i that maximizes $v_i(x_i) - \langle p, x_i \rangle$ among all $x_i \in [0, b]_{\mathbb{Z}}$. It is important to stress that a demand oracle indeed just returns a single vector instead of the whole set $\mathcal{B}_i$, so the answer of the oracle is guaranteed to be polynomial in time and space with respect to the number of item types m. Demand oracles are common in auction theory they provide a good balance between expressiveness and efficiency in terms of the amount of work to actually compute a response to a query and the number of bits that have to be sent (see [NS06; BN10]).

However, it is clear that a single vector $x_i \in \mathcal{B}_i$ for every $i \in N$ does not suffice to determine an optimal solution deterministically. The algorithms we present here will just use $x = (x_i)_{i \in N}$ as an initial solution but build an optimal solution using the second type of oracle which we call *exchange oracle* which we describe below. Given some $x_i \in \mathcal{B}_i$ and $e, f \in E$, let

$$w_i(e, f) := \max\{\alpha \in \mathbb{Z} \mid x_i - \alpha\chi_e + \alpha\chi_f \in \mathcal{B}_i\},$$

i.e., the maximum number of possible exchanges of e for f such that the resulting vector is still in $\mathcal{B}_i$.

The following lemma is enlightening.

Lemma 4.12. *For an* M*-convex set $\mathcal{B}_i$, $i \in N$ with associated rank function ρ_i, $x_i \in \mathcal{B}_i$, and $e, f \in E$, we have*

$$w_i(e, f) = \min\{\rho_i(S) - x_i(S) \mid e \notin S, f \in S\}. \tag{4.4}$$

Thus, $w_i(e, f) > 0$ if and only if $e \in \mathcal{T}_{\mathcal{B}_i}(x_i, f)$. Further, a set S is tight w.r.t $\mathcal{B}_i$ and x_i if and only if $\mathcal{T}_{\mathcal{B}_i}(x_i, f) \subseteq S$ for all $f \in S$.

Proof. Let $\alpha := w_i(e, f)$ and let β denote the minimum value on the right-hand side of (4.4), and S a minimizer. By definition, $x_i' = x_i - \alpha\chi_e + \alpha\chi_f \in \mathcal{B}_i$. We first show $\alpha \leq \beta$. This follows since $x_i'(S) \leq \rho_i(S)$, and $x_i'(S) = x_i(S) + \alpha$, and therefore $\beta = \rho_i(S) - x_i(S) \geq x_i'(S) - x_i(S) = x_i'(S) - (x_i'(S) - \alpha) = \alpha$.

Let us now turn to $\alpha \geq \beta$. Towards a contradiction assume that $x_i'' = x_i - \beta\chi_e + \beta\chi_f \notin \mathcal{B}_i$, that is, $x_i''(T) > \rho_i(T)$ for some $T \subseteq E$. Thus, $x_i''(T) > x_i(T)$; by definition, this means that $f \in T$, $e \notin T$. Hence, T is in the feasible domain of the minimization problem

in (4.4), giving $\rho_i(T) - x_i(T) \geq \beta$. But this means that $x_i''(T) = x_i(T) + \beta \leq \rho_i(T)$, leading to a contradiction. This completes the proof of (4.4).

From here, the second statement is immediate. For the final statement, since $\mathcal{T}_{\mathcal{B}_i}(x_i, f)$ is the inclusion-wise minimal tight set containing f, it is clearly contained in every tight set $S \subseteq E$ with $f \in S$. Conversely, if $\mathcal{T}_{\mathcal{B}_i}(x_i, f) \subseteq S$ is true for every $f \in S$, then $\bigcup_{f \in S} \mathcal{T}_{\mathcal{B}_i}(x_i, f) = S$, which is tight as it is the union of tight sets according to Lemma 4.9. □

The exchange oracle is queried with some vector x_i (obtained either by the demand oracle directly or feasible exchanges that have been queried before) and two item types e and f. The oracle answers with the maximum number of exchanges of e and f that are allowed starting from x_i, i.e., the number $w_i(e, f)$. Note that in some cases this number might be 0 but cannot be negative as this would imply that $x_i \notin \mathcal{B}_i$. Regarding running times, we use the placeholder ExO to denote the time that is needed to answer an exchange query.

Note that both DO and ExO depend on how the underlying set $\mathcal{B}_i$ is represented to the algorithm that computes it and on the algorithm itself.

We shall also discuss which other oracles would be alternatives to ours. A common model to represent the valuation functions v_i is by a *value oracle* that upon receiving a bundle $x_i \in [0, b]_{\mathbb{Z}}$ returns $v_i(x_i)$. Given a value oracle, a bundle $x_i \in \mathcal{B}_i(p)$ for some prices p and $\mathcal{B}_i \in \{\mathcal{D}_i(p), \widetilde{\mathcal{D}}_i(p), \widehat{\mathcal{D}}_i(p)\}$ can be computed using the greedy algorithm by m^2 value oracle calls, provided that v_i is (strong) gross substitute, where any of these choices of $\mathcal{B}_i$ is a polymatroid base set. Given $x_i \in \mathcal{B}_i$ and $\alpha \in \mathbb{Z}$, checking if $x_i - \alpha\chi_e + \alpha\chi_f \in \mathcal{B}_i$ takes a single value oracle call. Thus, one can compute $w_i(e, f)$ in $\mathcal{O}(\log(B))$ calls to the value oracle where $B := \max_{e \in E} b(e)$. We can also formulate the computation of $w_i(e, f)$ as submodular function minimization problem in (4.4). Hence, one can use a (strongly) polynomial submodular function minimization algorithm (see Section 1.2 for references). Nevertheless, this requires access to $\rho_i(S)$, which may lead to running a greedy algorithm for each query. The following example shows that the particular running times DO and ExO heavily depend on the representation of the valuation function, using the example of OXS functions.

Example 4.2. Let us consider the following OXS valuation function (see also Subsection 4.1.1) on a market with unit supply which is given by a weighted bipartite graph

$G = (E \cup T_i, A_i, \tilde{v}_i)$. A vector $x_i \in \{0,1\}^E$ corresponds to the set $X_i = \operatorname{supp}(x_i)$ and then the valuation is given by

$$v_i(x_i) = \max\{\tilde{v}_i(S) : S \subseteq X_i \text{ such that there exists a matching covering } S\}.$$

In this case, the time needed to evaluate a demand oracle query (i.e., DO, the time to find a set $x_i \in \widetilde{\mathcal{D}}_i(p)$ for some p) equals the time needed to find a maximum weight matching in a bipartite graph. The Hungarian method [Kuh55] (see also e.g., [Cor+22] for a more contemporary description) is a classical algorithm to solve this problem but better algorithms exists in particular if the graph is sparse ([DK69; GK97; DS12]).

To compute an answer to an exchange oracle query (i.e., EXO, the time to find the number $w_i(e,f)$ given some $X_i \in \widetilde{\mathcal{D}}_i(p)$), we only need to decide whether an alternating path from e to f with respect to a matching supporting the current set X_i.

For the unit supply setting, the sets $\widetilde{\mathcal{D}}_i(p)$ and $\widehat{\mathcal{D}}_i(p)$ are matroid base sets and our task is to solve MATROID UNION. For matroid optimization, the usual oracle settings are via independence or rank oracle queries. Such oracles may be expensive if our primary oracle is for the valuation function v_i: each independence or rank oracle query may require running a greedy algorithm.

To take the actual computation time to answer the oracle query in some underlying structure into account, Blikstad, Mukhopadhyay, Nanongkai, and Tu [Bli+23] propose to use the *dynamic rank oracle*. The idea is that given an independent set S, it is easy to find an independent set close to S, but much harder to find an independent set which is farther away (with respect to the symmetric difference of the sets). To be precise, a dynamic rank oracle starts with the empty set $S = \emptyset$, then three different operations are denoted as oracle calls (1) adding an element to S, (2) deleting an element from S, and (3) obtaining the rank of S (see also [Bli+23, Definition 1.2]).

Our exchange oracle EXO is efficient from this point of view, since given an independent set, we always query sets with symmetric difference of size two. We can perform an exchange oracle query by using three dynamic rank oracles calls.[4] Note that we assume that we do not have to build the first independent set X_i with the dynamic rank oracle but that we can perform the queries starting from the set that we obtained from the demand oracle. A demand oracle query can be answered by $\mathcal{O}(m)$ dynamic oracle queries. The other way around, the dynamic rank oracle is stronger than the exchange

[4] If we cannot perform the queried exchange, one can also argue that we need two dynamic oracle calls more to return to the original set, i.e., five dynamic oracle calls for one exchange oracle query.

oracle, in the sense that the dynamic rank oracle can compute the rank of a given set S in $\mathcal{O}(|S|)$ computations, while the exchange oracle cannot answer this query. Note that in the algorithm for the unit supply setting, we will use exchange operations that do not change the rank of the sets, so in fact we know the ranks as well. It can therefore be said that the two models behave very similarly in the matroid case and our exchange oracle model can be generalized to polymatroids.

If we analyze our Matroid Union algorithm (see Section 4.4.3) in the unit supply setting with the dynamic rank oracle, we get a running time of $\mathcal{O}((m^3 + nm^2) \cdot \mathsf{DRO})$. Hence, the running time of our Matroid Union algorithm in the dynamic rank oracle model is not comparable to the running time $\mathcal{O}(\mathrm{poly}(n \log(m))m^{\frac{3}{2}} \cdot \mathsf{DRO})$ of [Bli+23] as we are linear in n, while [Bli+23] achieve a better dependency on m.

4.4 Solving Matroid Union and Polymatroid Sum

We now show how the aforementioned problems Matroid Union and Polymatroid Sum can be solved. This is on the one hand important in order to compute an allocation for a suitable, i.e., Walrasian, price vector p but as we teased earlier, we can also use the dual problem to compute an excess demand set or excess supply set to adjust the prices in a dynamic auction, which we will show in Theorem 4.25 and Theorem 4.30. Conveniently, the dual solution to Matroid Union and Polymatroid Sum, respectively, will just fall off from an auxiliary graph that we use in the algorithm to solve the "primal" problem. This is indeed very similar to the our observations from Chapter 3, where an excess demand set also could be found (if it exists) using a minimum cut, which is the dual solution to the underlying Maximum Flow problem. Moreover, there are multiple ways—or algorithmic paradigms—to solve these problems. As for Maximum Flow, we can find an optimal solution using augmenting paths algorithms (Ford-Fulkerson Algorithm and Edmonds-Karp Algorithm, cf. Section 3.2) or push-relabel algorithms (Goldberg-Tarjan Algorithm, Preflow-Push). While the Ford-Fulkerson algorithm (and its efficient derivate by Edmonds and Karp) and the Goldberg-Tarjan algorithm for Maximum Flow are pretty much standard in maths and computer science curricula, it is less common to encounter algorithms to solve Matroid Union and Polymatroid Sum even in advanced courses. In this section we present a push-relabel algorithm for Polymatroid Sum that we developed based on the one by Frank and Miklós [FM12] for Matroid Union.

4.4.1 Knuth's Algorithm for Matroid Union

As a gentle introduction, we start by presenting an augmenting path algorithm to solve Matroid Union. The first algorithm for Matroid Union is due to Edmonds [Edm68]. The algorithm we present here was developed by Knuth [Knu73] and independently by Greene and Magnanti [GM75].

Recall from Theorem 1.8 in Section 1.3 that a union of matroids again yields a matroid.

As introduced above, for Matroid Union we are given n matroids $M_1, \dots, M_n$ with $M_i = (E, \mathcal{I}_i)$ for all $i \in N = \{1, \dots, n\}$ all defined on the same ground set E. The task is to find a set $I = I_1 \uplus \dots \uplus I_n$ of maximum cardinality such that $I_i \in \mathcal{I}_i$ for all $i \in N$. For any $i \in N$ with matroid $M_i = (E, \mathcal{I}_i)$ and any $I \in \mathcal{I}_i$ we can construct an *exchange graph* $G_i(I_i) = (E, A_i, F_i)$, where

$$A_i = \{(e, f) \in E \times E \mid e \in I_i, f \notin I_i \text{ and } I_i - e + f \in \mathcal{I}_i\},$$
$$F_i = \{e \notin I_i \mid I_i + e \in \mathcal{I}_i\}.$$

In some sense, $G_i(I_i)$ depicts the set I_i and its independent neighborhood, where the neighborhood of I_i consists of all independent sets of M_i that can be obtained from I_i by either exchanging two elements along an edge in A_i or adding one element from F_i without dropping any from I_i. Intuitively, $G_i(I_i)$ shows which elements a buyer i that is currently awarded the set I_i would like to add to her set or which one she is willing to exchange without losing utility. Recall that in the unit supply setting $\breve{\mathcal{D}}_i(p)$ is a matroid base set.

Given a feasible solution $I = I_1 \uplus \dots \uplus I_n$, we can merge the exchange graphs of $G_1(I_1), \dots, G_n(I_n)$ into an exchange graph $G(I) = (E, (A_i)_{i \in N}, (F_i)_{i \in N})$, which will be the base of Knuth's Algorithm for Matroid Union. Visually, one can think of edges and vertices in this exchange graph being colored by the set N.

The overall procedure of constructing $G(I)$ is given in the pseudocode below. Note that additionally to storing the current independent set I_i for each matroid M_i, we also store for each element e to whose independent set it currently belongs (`owner` mapping). Maintaining both, the partition and the `owner` mapping is straight-forward and we assume the changes in the latter are made implicitly when updating the former in the following steps.

Construct Exchange Graph:

Input: Matroids $(M_i)_{i \in N}$ and partition of independent sets $(I_i)_{i \in N}$
Output: Exchange graph $G(I)$

```
1  F_i := ∅, A_i := ∅ for all i ∈ N
2  for f ∈ E do
3      j := owner(f)
4      for i ∈ N ∖ {j} do
5          for e ∈ I_i do
6              if I_i − e + f ∈ 𝓘_i then
7                  A_i := A_i + (e, f)
8          if I_i + f ∈ 𝓘_i then
9              F_i := F_i + f
10 return (E, (A_i)_{i∈N}, (F_i)_{i∈N})
```

From an exchange graph $G(I) = (E, (A_i)_{i \in N}, (F_i)_{i \in N})$ we can essentially read off whether the current set I is maximum by conducting a breadth-first search from $F := \bigcup_{i \in N} F_i$. Say the resulting set is

$$R := \{e \in E \mid e \text{ is reachable in } G(I) \text{ from some } f \in F\}.$$

Then we can augment I if $R \cap (E \setminus I) \neq \emptyset$ which is formalized in the following lemma and pseudocode. The idea is to take a shortcut-free path from some $f \in F$ to some $e \in R \setminus I$ and exchange the elements along this path according to the color of the arcs of that path. Let us call the resulting set I', which is feasible as we made only feasible exchanges along the exchange graph. This results in the starting element f being free, while the resulting set I' has the same size as I. Now we only have to add f to some I_i with $f \in F_i$.

Lemma 4.13 ([Sch03, Theorem 42.4]). *The set I is an independent set of maximum cardinality in the union of matroids if and only if $R \cap (E \setminus I) = \emptyset$.*

Example 4.3. Consider the following three matroids on the common ground set $E = \{e_1, \ldots, e_6\}$

- M_1 (blue) is a partition matroid with $E_1 = \{e_1, e_2, e_3, e_4\}$ and $E_2 = \{e_5\}, E_3 = \{e_6\}$ and capacities $c_1 = 3$, $c_2 = 0$, and $c_3 = 1$, also depicted in Figure 4.2a.

Knuth's Algorithm for Matroid Union:

Input: Matroids $((M_i)_{i \in N})$
Output: Maximum Partition of independent sets

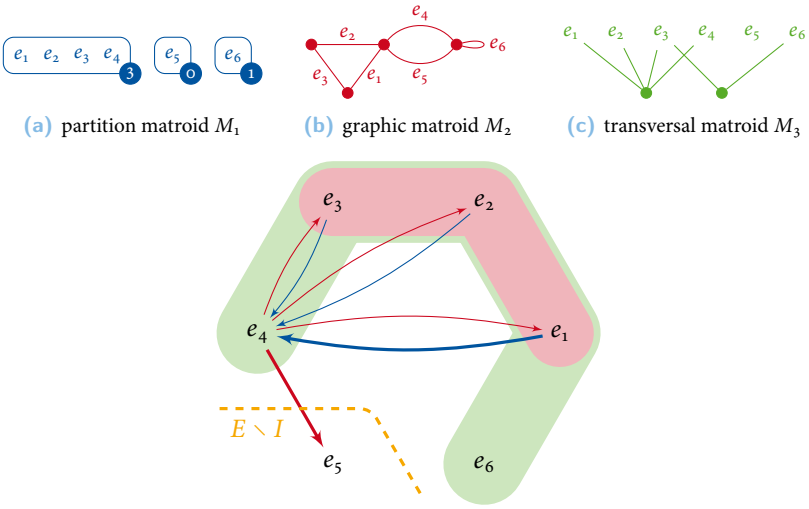
```
I_i := ∅ for all i ∈ N, I := ⋃_{i∈N} I_i
G(I) = (E, (A_i)_{i∈N}, (F_i)_{i∈N}) := Construct Exchange Graph
while P := shortcut-free F-(E ∖ I)-path in G(I) exists do
    for (e, f) ∈ P do
        i := owner(e)
        I_i := I_i − e + f
    f := first vertex on P
    I_i := I_i + f for some i ∈ N with f ∈ F_i
    G(I) := Construct Exchange Graph
return (I_i)_{i∈N}
```

- M_2 (red) is a graphic matroid where $\{e_1, e_2, e_3\}$ are a circuit, e_4, e_5 are parallel edges, and e_6 is a loop as depicted in Figure 4.2b.
- M_3 (green) is a transversal matroid with two sets $\{e_1, e_2, e_3, e_4\}$ and $\{e_3, e_6\}$, also as depicted in Figure 4.2c.

While we can run Knuth's Algorithm for Matroid Union using the empty set as initial allocation, this becomes a bit tedious and these steps are not that insightful. Instead let us fast-forward to the allocation $I_1 = \{e_1, e_2, e_3, e_6\}$, $I_2 = \{e_4\}$, and $I_3 = \emptyset$. In particular the item e_5 is not allocated. Then we get the following possible exchanges:

- In M_1 we can exchange e_4 for any item out of $\{e_1, e_2, e_3\}$, therefore, we have $A_1 = \{(e_1, e_4), (e_2, e_4), (e_3, e_4)\}$. As I_1 is a base of M_1, we have $F_1 = \emptyset$.
- In M_2, any item of $F_2 = \{e_1, e_2, e_3\}$ can be added to I_2 (in fact, any two of those items even). Moreover, we can exchange e_4 for e_5, i.e., $A_2 = \{(e_4, e_5)\} \cup \{(e_4, f) : f \in F_2\}$.
- As $I_3 = \emptyset$, there is nothing to exchange, i.e., $A_1 = \emptyset$. However, any item except e_5 can be added to I_3, i.e., $F_3 = E \setminus \{e_5\}$.

Note that the unallocated item e_5 cannot be directly allocated to anyone without removing an item. The exchange graph is depicted in Figure 4.2d. We now need to find a shortcut-free path from $F = F_1 \cup F_2 \cup F_3$ to $E \setminus (I_1 \cup I_2 \cup I_3)$. A breadth-first search from F can yield to different paths (depending on the start vertex). For the sake of exposure, let us choose a path from e_1 (which is colored by M_2 (red) and M_3 (green)). The path is $\{e_1, e_4, e_5\}$ via a (blue) M_1-edge and a (red) M_2-edge. The colors indicate how the allocation changes:



(a) partition matroid M_1 (b) graphic matroid M_2 (c) transversal matroid M_3

(d) exchange graph $G(I)$

Figure 4.2: Matroids and exchange graph for Example 4.3.

- The item e_1 gets allocated to I_2 or I_3 as it is colored by these two matroids, say I_3.
- The blue edge (e_1, e_4) indicates that e_1 gets replaced by e_4 in I_1.
- The red edge (e_4, e_5) indicates that e_4 gets replaced by e_5 in I_2.

This yields the new allocation $I_1 = \{e_2, e_3, e_4, e_6\}$, $I_2 = \{e_5\}$, and $I_3 = \{e_1\}$, which clearly is maximum. The corresponding exchange graph cannot have a path from F into $E \setminus I$ as the latter set is empty.

Theorem 4.14 ([Knu73]). *Knuth's Algorithm for Matroid Union computes an optimal solution to an instance of* Matroid Union *with n matroids on a ground set of size m in time $\mathcal{O}(m^3 + m^2 n\mathsf{IO})$, where IO represents the time for a query to an independence oracle.*

4.4.2 A Generic Push-Relabel Algorithm for M-convex Set Packing

We already briefly explained the push-relabel paradigm to solve combinatorial optimization problems (in particular submodular function minimization problems and subclasses of those) at the beginning of this chapter. Recall that the push-relabel paradigm

states to take an arbitrary solution—not necessarily feasible—that satisfies an optimality condition and maintain this optimality condition while making the solution "less infeasible" over time. In this subsection we present a push-relabel framework to solve the M-convex Set Packing problem and apply it to Polymatroid Sum. We again will make use of an exchange graph as described in the previous subsection, however, additionally to the color of an edge (that indicates which of the buyers would make the exchange) we also use a weight for each edge (that indicates with how often that exchange is possible as there are multiple copies of an item type). Our push-relabel algorithm is based on the one by Frank and Miklós [FM12] who presented one for Matroid Union and a more general one for the feasibility problem Submodular Flow. Our algorithm generalizes the former (in the sense that it works not just for matroids but also polymatroids) but is a special case of the latter. In [FM12], the running time of the algorithm is only measured in terms of basic operations, i.e., pushes and relabels that get performed, while our analysis also involves the number of necessary oracle queries. Here we also give a self-contained presentation of the algorithm.

In the following, we consider M-convex sets $\mathcal{B}_1, \dots, \mathcal{B}_n \subseteq [0, b]_{\mathbb{Z}}$ and for each $i \in N = \{1, \dots, n\}$ a fixed vector $x_i \in \mathcal{B}_i$. Of course, in our auction setting the M-convex sets $\mathcal{B}_i$ will be demand correspondences and the $x_i \in \mathcal{B}_i$ to preliminary allocations of items (*bundles*). This economic inspiration for the problem at hand also gives us some intuitive natural language to use, which we will do in the following Since we consider a push-relabel algorithm, we do not restrict ourselves to vectors of bundles $x \in \mathfrak{B} = \mathcal{B}_1 \times \dots \times \mathcal{B}_n$ that satisfy the capacity constraints induced by b. For $x \in \mathfrak{B}$, let us call an item type e *scarce*[5] if $\sum_{i \in N} x_i(e) > b(e)$ and *ample*[6] if $\sum_{i \in N} x_i(e) < b(e)$.

For the algorithm we define a *label function* (or *level function*) $\ell \colon E \to \{0, 1, \dots, m\}$ and denote $\ell_{\min}(S) := \min\{\ell(e) : e \in S\}$ for any $S \subseteq E$. We use the shorthand $\mathcal{T}_i(e)$ for $\mathcal{T}_{\mathcal{B}_i}(e, x_i)$, i.e., the collection of tight sets w.r.t. $\mathcal{B}_i$ and x_i that contain e (cf. Definition 4.8 and Lemma 4.9).

Definition 4.15. Given $x_i \in \mathcal{B}_i$ for all $i \in N$, a label function $\ell \colon E \to \{0, 1, \dots, m\}$ is *valid* if

(L1) $\ell(e) = 0$ if e is scarce,
(L2) $\ell_{\min}(\mathcal{T}_i(e)) \geq \ell(e) - 1$ for all $i \in N$ and $e \in E$.

[5] oversold
[6] undersold

Recall the definition of the edge weights: given a vector $x_i \in \mathcal{B}_i$ for some M-convex set B_i, we have $w_i(e,f) = \max\{\alpha \in \mathbb{Z} \mid x_i - \alpha\chi_e + \alpha\chi_f \in \mathcal{B}_i\}$. The general idea of the push-relabel algorithm is as follows. Initially, all $e \in E$ have the label 0, i.e., $\ell(e) = 0$. As long as there are ample items left that have a label smaller than m, we take such an item f and try to **push** at this item type. If this is not possible, we **relabel** f. These two operations work as follows

push (at f w.r.t. e and i) If we find an item e with $e \in \mathcal{T}_i(f)$ for some ample item f and buyer $i \in N$ with $\ell(f) = \ell(e) - 1$, then we update $x_i := x_i - \alpha\chi_e + \alpha\chi_f$ with $\alpha := \min\{b(e) - \sum_{j\in N} x_j(e), w_i(e,f)\}$.

relabel (f) If no such item exists, we update $\ell(f) := \ell(f) + 1$.

For the analysis later, we make the following distinction between pushes: If in a push, the minimum $\alpha = \min\{b(e) - \sum_{j\in N} x_j(e), w_i(e,f)\}$ is attained at the $w_i(e,f)$ term, we call it a *saturating push*; otherwise, if the minimum is only assumed by the first term $b(e) - \sum_{j\in N} x_j(e)$, then it is called a *non-saturating push*.

Push-Relabel Algorithm:

Input: M-convex sets $\mathfrak{B} = (\mathcal{B}_i)_{i\in N}$
Output: Maximum independent vector

Arbitrary vector $x \in \mathfrak{B}$
$\ell(e) := 0$ for all $e \in E$
while *ample item f with $\ell(f) < m$ exists* **do**
  Select any such f
  if *e and i with $e \in \mathcal{T}_i(f)$ and $\ell(e) = \ell(f) - 1$ exists* **then**
    // **push** at f w.r.t. e and i
    $\alpha = \min\{b(e) - \sum_{j\in N} x_j(e), w_i(e,f)\}$
    $x := x - \alpha\chi_e + \alpha\chi_f$
  else
    // **relabel** f
    $\ell(f) := \ell(f) + 1$
return x

We now show with two lemmas that the Push-Relabel Algorithm computes an solves the Polymatroid Sum problem. For that, we use the conditions (L1) and (L2) from Definition 4.15 as invariants that are maintained throughout the execution of the algorithm.

Lemma 4.16. *The invariants (L1) and (L2) hold throughout the Push-Relabel Algorithm.*

Proof. Clearly, both invariants hold at initialization. Invariant (L1) holds also in all subsequent iterations as labels are only changed for ample items, which will never become scarce by choice of α in any iteration.

For invariant (L2), consider any iteration τ of the Push-Relabel Algorithm that satisfies (L2). We show that then it will also hold in the next iteration $\tau + 1$. If this iteration $\tau + 1$ performs a relabel for some item f, then in iteration τ we have that for all $e \in \mathcal{T}_i(f)$ it holds that $\ell(e) > \ell(f) - 1$; otherwise a push would have been performed at f w.r.t. e (if equality) or (L2) would have been already violated in iteration τ (if inequality was reversed). Thus, $\ell_{\min}(\mathcal{T}_i(f)) \geq \ell(f)$, and hence, (L2) also holds in iteration $\tau + 1$ where f's label increased by one.

Now suppose, iteration $\tau + 1$ performs a push at f w.r.t. some $e \in E$ and $i \in N$ instead. That is, we make an update $x_i' := x_i - \alpha\chi_e + \alpha\chi_f$ for some $\alpha > 0$. Let $\mathcal{T}_j'(g)$ denote the tight sets for each $g \in E$ and $j \in N$ after the push in iteration $\tau + 1$ and $\mathcal{T}_j(g)$ refers to the tight sets before the push in iteration τ, where we assume (L2) holds. We need to show that

$$\ell_{\min}(\mathcal{T}_j(g)) \geq \ell(g) - 1 \tag{4.5}$$

holds for all $j \in N$ and all $g \in E$. For all $j \neq i$ and all $g \in E$ this is immediate as the vectors and the labels do not change, i.e., $x_j' = x_j$ and $\mathcal{T}_j'(g) = \mathcal{T}_j(g)$. It also is clearly true for $j = i$ and $g \in E$ where $e \notin \mathcal{T}_i(g)$, since in this case $\mathcal{T}_i(g)$ is also tight for x_i' and hence, $\mathcal{T}_i'(g) \subseteq \mathcal{T}_i(g)$. Hence, it is left to show that (4.5) hold for $j = i$ and $g \in E$ with $e \in \mathcal{T}_i(g)$. We have $\mathcal{T}_i'(g) \subseteq \mathcal{T}_i(g) \cup \mathcal{T}_i(f)$, since the union of two tight sets (before the push) is tight again by Lemma 4.9. Thus,

$$\begin{aligned}
\ell_{\min}(\mathcal{T}_i'(g)) &\geq \min\{\ell_{\min}(\mathcal{T}_i(g)), \ell_{\min}(\mathcal{T}_i(f))\} \\
&= \min\{\ell_{\min}(\mathcal{T}_i(g)), \ell(e)\} \\
&= \ell_{\min}(\mathcal{T}_i(g)),
\end{aligned}$$

where the penultimate equality follows from the facts that $\ell(e) = \ell_{\min}(\mathcal{T}_i(f))$ if f is selected for the push and that (L2) is satisfied before the push. The last equality follows by $e \in \mathcal{T}_i(g)$. Thus, (4.5) holds true also after the push in iteration $\tau + 1$. □

Lemma 4.17. *The Push-Relabel Algorithm returns an optimal solution to the* Polymatroid Sum *problem* (4.8).

Proof. First note that when the algorithm stops, there is some unused label: by pigeonhole principle, with $m+1$ labels but only m item types, there exists some $k \in \{0, 1, \dots, m\}$ with $\ell^{-1}(k) = \varnothing$.

By the first half of the proof of the Min-Max Theorem for Polymatroid Sum (4.10), we already know that

$$\sum_{e \in E} \min \left\{ \sum_{i \in N} x_i(e), b(e) \right\} \leq \sum_{i \in N} \rho_i(E \setminus F) + b(F)$$

holds for all $x \in \mathfrak{B}$ and all $F \subseteq E$ and if it holds with equality, then x is optimal by Corollary 4.11 (note that without the second half of the proof of Theorem 4.10, the corollary yields still a sufficient condition).

We now show that the Push-Relabel Algorithm constructs x and a witness set F as desired, completing the proof of the Min-Max Theorem for Polymatroid Sum. Let $F = \{e \in E \mid \ell(e) < k\}$, where k is the aforementioned unused label. Note that it is entirely possible that $F \in \{\varnothing, E\}$.

Since there is no ample item e with $\ell(e) < m$ by the stopping criterion of the Push-Relabel Algorithm, there is also no ample item in F. Thus,

$$\sum_{i \in N} x_i(e) \geq b(e)$$

for all $e \in F$. Also note that

$$\sum_{i \in N} x_i(e) \leq b(e)$$

for all $e \notin F$ as for those $\ell(e) > 0$ holds while scarce items have the label 0 by (L1). Finally, since $\mathcal{T}_i(e) \subseteq E \setminus F$ for all $e \in E \setminus F$ by (L2) and the choice of k, we have

$$x_i(E \setminus F) = \rho_i(E \setminus F)$$

for all $i \in N$. That is, x and F satisfy the complementary slackness conditions of Corollary 4.11. Hence, $E \setminus F = \bigcup_{e \in E \setminus F} \mathcal{T}_i(e)$ is tight for all $i \in N$ as a union of tight sets. □

4.4.3 An Efficient Implementation for Matroid Union

Let us now consider the unit supply setting first, i.e., where we have $b(e) = 1$ for all $e \in E$. In that case, for all $i \in N$ we have $\mathcal{B}_i \subseteq [0,1]_{\mathbb{Z}}$ which just corresponds to a matroid base set where every $x_i \in \mathcal{B}_i$ is the indicator vector of a base B_i of the underlying matroid. Then it is also very easy to describe the tight sets and weights of the exchange graph. The tight sets w.r.t. $\mathcal{B}_i$ and e (technically, χ_e) are

$$\mathcal{T}_i(e) = \begin{cases} \{e\} & \text{if } e \in B_i, \\ C(B_i, e) & \text{if } e \notin B_i, \end{cases}$$

where $C(B_i, e)$ denotes the fundamental circuit of B_i and e. All the weights in the exchange graph are binary, i.e., $w_i(e,f) \in \{0,1\}$ and these values are just redundant information as they are already contained in the exchange graph as $w_i(e,f) = 1$ if and only if $(e,f) \in A_i$.

In the body of the **while** loop, given an ample item f, the algorithm checks if it can perform a push at f. For this, the algorithm might need to go through all $i \in N$, and check whether there exists an item e in the fundamental circuit $C(B_i, f)$ with label $\ell(e) = \ell(f) - 1$. This may cost nm queries, but only if f has label $\ell(f) = 1$. If f has the label $\ell(f) = 0$, we can immediately relabel. Whenever f has the label $\ell(f) > 1$, then it suffices to go through each item e at most once, since each item e of with label $\ell(e) = \ell(f) - 1$ is allocated exactly once (by (L1) and since there is only one unit per item). In particular, this means that all pushes are saturating.

The proof of the following theorem makes use of the described procedure to bound the running time.

Theorem 4.18. *The Push-Relabel Algorithm can be implemented in running time* $\mathcal{O}(n \cdot \mathsf{DO} + (m^3 + m^2 n) \cdot \mathsf{ExO})$ *for the unit supply case.*

Proof. The algorithm starts with finding an initial allocation x in $n \cdot \mathsf{DO}$ time. Given an allocation x, we can compute how often an item e is allocated in time $\mathcal{O}(nm)$. These values can be updated in time $\mathcal{O}(1)$ at every push operation. Moreover, these values allow us to maintain a list of all ample items, which needs $\mathcal{O}(m)$ time for initialization and can be maintained in $\mathcal{O}(1)$ at each push. Note that at a relabel operation, nothing changes for these values and the list.

We find an ample (i.e., unsold) item e and check its label $\ell(e)$ in $\mathcal{O}(1)$. If $\ell(e) = 0$, we can relabel immediately. If $\ell(e) = 1$, we will go through all buyer-item pairs of items with label 0 to determine whether a push is possible, otherwise we relabel. This takes $\mathcal{O}(mn)$ exchange oracle queries. For every other label $k \geq 2$, any item on label $k-1$ is allocated to at most one buyer (by (L1), and since there is only one unit per item). Thus, we just need to go through the items once to determine a push or a relabel operation. This can be done in $\mathcal{O}(m)$ exchange oracle queries.

The number of relabel operations per label $k \geq 1$ is bounded by $\mathcal{O}(m)$. The number of pushes per label is also bounded by $\mathcal{O}(m)$. To see this, we show that for each label k the value $\Phi(k) := \sum_{i \in N} |B_i^{\geq k}|$ is non-decreasing in every relabel or push operation, where $|B_i^{\geq k}|$ is the number of items in B_i that were added to the bundle when their label was at least k. Relabel operations are only used for ample items, and hence, they do not affect the $B_i^{\geq k}$ sets; thus, $\Phi(k)$ does not change. At any push at f with respect to e and i, $|B_j^{\geq k}|$ does not change for $j \neq i$. For buyer i with bundle B_i, the item e which leaves B_i has a label exactly one lower than the the label of the item f which enters the bundle. Hence, $|B_i^{\geq k}|$ does not decrease. When pushing at f with respect to i and e with $\ell(e) = k$, we strictly increase $|B_i^{\geq k}|$ and thus, $\Phi(k)$.

Since $\Phi(k) \leq m$ for each $k \geq 1$ (as each item with label $k \geq 1$ is currently allocated to exactly one buyer), we get a bound of m on the number of pushes per level.

This gives a total running time of $\mathcal{O}((m^2 n + m^3) \cdot \mathsf{ExO})$ after finding the initial allocation, where the first term is generated by the pushes and relabel operations on items labeled 1 and the second term captures all remaining pushes. □

We will later show that from this solution of the Polymatroid Sum problem, we can construct an excess demand set or excess supply set within the same running time. Essentially, there is just one breadth-first search in an exchange graph (similar to the one used by Knuth's Algorithm for Matroid Union) missing, which only requires time linear in the size of the exchange graph ($\mathcal{O}(nm^2)$).

4.4.4 An Efficient Implementation for Polymatroid Sum

In this subsection, we give a fast implementation of the Push-Relabel Algorithm for Polymatroid Sum which we will later use to find excess demand and excess supply sets in the setting where item types are available in multiple copies. The crucial difference to the setting in the previous section is that a non-ample item type e can be allocated

to more than one buyer. That means if we want to perform a push at some ample item type f with respect to e, we might have multiple buyers i to choose from and indeed, might also have to select multiple of them to allocate all units of f. It is crucial to search for the suitable buyer-item pairs (i, e) in a structured way such that we do not repeat unnecessary work—we already made this observation in the unit supply case if $\ell(f) > 1$; in that case we also only needed to look at every e at most once to determine whether a push is feasible. For the sake of convenience, we assume both, the buyers and item types to be indexed by naturals, i.e., $N = \{1, \dots, n\}$ and $E = \{1, \dots, m\}$. Then we define that a buyer-item pair (i, e) is *lexicographically smaller* than the item pair (j, f) if $i < j$, or $i = j$ and $e < f$. In this case we write $(i, e) \prec (j, f)$. When we search for a buyer-item pair to perform a push, we use the lexicographical ordering to scan through these pairs. This idea is not novel in combinatorial optimization. The approach has already been used in [Sch80] for Polymatroid Intersection and has also been used by Fujishige and Zhang [FZ92; FZ96] in push-relabel algorithms for problems involving submodular functions. Note that the Push-Relabel Algorithm (or the original version by Frank and Miklós [FM12]) does not rely on a specific selection rule to perform the push correctly. The goal of this rule is simply to limit the number of queries to the exchange oracle and in particular, avoid unnecessary queries.

Algorithm We implement the Push-Relabel Algorithm for Polymatroid Sum with two subtleties: (1) we always select an ample item f with highest label $\ell(f)$ among those that have a label strictly smaller than m, and (2) for some given item e, when selecting the buyer-item pair (i, e) for the next push, we always select the pair which is lexicographically smallest among all buyer-item pairs that fulfill the necessary requirements for a push at item f. In Lemma 4.19 we show that between two relabels of some item f the algorithm does not have to go back in the lexicographic order to find suitable buyer-item pairs (i, e) to perform a push, which limits the number of pairs that we have to search through between two relabels of some item to just $\mathcal{O}(mn)$.

The pseudocode Push-Relabel Algorithm for Polymatroid Sum makes this description more precise.

Correctness We show a lemma from which we can follow that the Push-Relabel Algorithm for Polymatroid Sum computes an optimal solution to Polymatroid Sum.

Push-Relabel Algorithm for Polymatroid Sum:

Input: Polymatroids $(P_i)_{i \in N}$
Output: Maximum independent vector

Arbitrary vector $x \in \mathfrak{B}$
for $e \in E$ **do** `// initialize labels and exchange pointers`
  $\ell(e) := 0$
  $(j_e, g_e) := (1,1)$
while *there is an ample item* f *with* $\ell(f) < m$ **do**
  push := `false`
  Select $f \in \arg\max_{g \in E}\{\ell(g) \mid g \text{ is ample and } \ell(g) < m\}$
  for *buyer-item pairs* (i,e) *in lexicographically increasing order starting with the pair* (j_f, g_f) **do**
    if $w_i(e,f) > 0$ **and** $\ell(f) = \ell(e) + 1$ **then** `// buyer-item pair to perform a push`
      push := `true`
      $\alpha := \min\left\{b(f) - \sum_{j \in N} x_j(f), w_i(e,f)\right\}$
      $x_i := x_i - \alpha\chi_e + \alpha\chi_f$
      if $b(f) - \sum_{j \in N} x_j(f) < w_i(e,f)$ **then** `// non-saturating push`
        $(j_f, g_f) := (i,e)$ `// set pointer to current pair`
        continue with next ample item f (by aborting the for loop) `// f is not ample anymore`
      else `// saturating push`
        $(j_f, g_f) := \begin{cases} (i, e+1) & \text{if } e+1 \text{ exists,} \\ (i+1, 1) & \text{otherwise} \end{cases}$ `// set pointer to next pair in lex. order`
  if not *push* **then** `// relabel if no push was found for` f
    $\ell(f) := \ell(f) + 1$
    $(j_f, g_f) := (1,1)$ `// reset pointer to first pair in lex. order`
return x *and* $\ell(e)$ *for all* $e \in E$

Lemma 4.19. *Let $f \in E$ with $\ell(f) = k$ and let $i \in N$. If all pushes in the Push-Relabel Algorithm are performed with respect to some minimal suitable item, then*

$$\min\{e \in \mathcal{T}_i(f) \mid \ell(e) = k-1\}, \tag{4.6}$$

is non-decreasing and strictly increasing at every saturating push at f with respect to e and i until f is relabeled.

Proof. We show that the statement is true after each push and after each relabel.

If a push operation is performed with respect to some buyer $j \neq i$, then neither $\mathcal{T}_i(f)$ nor the labels change and hence, the minimum is the same as before for fixed f and i. From some bundle x_i, the push yields the bundle $x_i' = x_i - \alpha\chi_e + \alpha\chi_f$ for some $\alpha \in \mathbb{Z}_+$ and we denote the corresponding tight sets by $\mathcal{T}_i'(\cdot)$. We now consider a push with respect to i and some item e and distinguish between the cases whether the push is performed at f or some other item $g \neq f$.

- push at f (w.r.t. i and e): Then $e = \min\{e \in \mathcal{T}_i(f) \mid \ell(e) = k-1\}$ by the assumption of the lemma that the Push-Relabel Algorithm performs every push on a minimal suitable item.
  If the push is saturating, then we have $\alpha = w_i(e,f)$ and any further exchange between those items is not feasible, i.e., $x_i' - \chi_e + \chi_f \notin \mathcal{B}_i$, and hence, $e \notin \mathcal{T}_i'(f)$ by Lemma 4.12. Moreover, $\rho_i(\mathcal{T}_i(f)) = x_i(\mathcal{T}_i(f)) = x_i'(\mathcal{T}_i(f))$ since $e, f \in \mathcal{T}_i(f)$. Thus, $\mathcal{T}_i(f)$ is a tight set with respect to x_i' and hence, $\mathcal{T}_i'(f) \subseteq \mathcal{T}_i(f)$. We get that $\mathcal{T}_i'(f) \subseteq \mathcal{T}_i(f) \setminus \{e\}$, from which it follows that the minimum in (4.6) is strictly increasing.
  Now suppose the push is non-saturating, i.e., we have $\alpha < w_i(e,f)$. Then further exchanges of e and f are accepted by i, more precisely, $x_i' - (w_i(e,f) - \alpha)\chi_e + (w_i(e,f) - \alpha)\chi_f \in \mathcal{B}_i$, implying $e \in \mathcal{T}_i'(f)$ again by Lemma 4.12. Moreover, $\mathcal{T}_i(f)$ is tight with respect to x_i' and thus $\mathcal{T}_i'(f) \subseteq \mathcal{T}_i(f)$, which implies that the minimum of (4.6) is still e.
- push at $g \neq f$ (w.r.t. i and e): If $g, e \notin \mathcal{T}_i(f)$, then $\mathcal{T}_i'(f) = \mathcal{T}_i(f)$ and hence, (4.6) remains the same.
  Now suppose that $g \in \mathcal{T}_i(f)$, then it holds that $\mathcal{T}_i(g) \subseteq \mathcal{T}_i(f)$ by Lemma 4.12. Thus, we also have that $e \in \mathcal{T}_i(f)$ which implies that $x_i'(\mathcal{T}_i(f)) = x_i(\mathcal{T}_i(f))$. Since $\mathcal{T}_i(f)$ is tight, we have $x_i(\mathcal{T}_i(f)) = \rho_i(\mathcal{T}_i(f))$. In other words, $\mathcal{T}_i(f)$ remains tight after the push at g. This yields $\mathcal{T}_i'(f) \subseteq \mathcal{T}_i(f)$ as $\mathcal{T}_i'(f)$ is the minimal tight set containing

f with respect to x_i' and the fact that $\mathcal{T}_i(f)$ is already a valid candidate set. Thus, we must also have that (4.6) can only increase.

Finally, we consider the case where $g \notin \mathcal{T}_i(f)$ but $e \in \mathcal{T}_i(f)$. Then we have $\mathcal{T}_i'(f) \subseteq \mathcal{T}_i(f) \cup \mathcal{T}_i(g)$ since $\mathcal{T}_i(f) \cup \mathcal{T}_i(g)$ is tight as the union of tight sets (Lemma 4.9) and remains tight after the push as it contains g and e. If the $\ell(g) > k$, then $\ell_{\min}(\mathcal{T}_i(g)) \geq k$ by (L2) and hence, $\{e \in E \mid \ell(e) = k-1\} \cap \mathcal{T}_i'(f) \subseteq \mathcal{T}_i(f)$. In other words, every item that is a candidate for the minimum after the push is also in $\mathcal{T}_i(f)$, i.e., a candidate already before the push as all items in $\mathcal{T}_i(g)$ have a label which is too high. If $\ell(g) = k$, then

$$\underbrace{\min\{e' \in \mathcal{T}_i(g) \mid \ell(e') = k-1\}}_{=e} \geq \min\{e' \in \mathcal{T}_i(f) \mid \ell(e') = k-1\},$$

since we assume that $e \in \mathcal{T}_i(f)$ and $\ell(e) = k-1$ and thus, e is a suitable item for the minimum on the right-hand side. Hence, we get that $\mathcal{T}_i'(f) \subseteq \mathcal{T}_i(f) \cup \mathcal{T}_i(g)$ from which it follows that (4.6) cannot decrease after the push.

The last case where $\ell(g) < k$ cannot occur since $\ell(e) = \ell(g) - 1 < k - 1$ as we push at g with respect to e and $e \in \mathcal{T}_i(f)$, which implies $\ell(e) \geq \ell(f) - 1 = k - 1$ which is a contradiction.

The argument for a relabel is straight-forward: A relabel does not change $\mathcal{T}_i(f)$ as the set of buyer i stays the same. Moreover, due to (L2), there does not exist an item $e \in \mathcal{T}_i(f)$ with label $k-2 = \ell(e) < \ell(f) - 1 = k - 1$. Thus, there is no further candidate item in $e \in \mathcal{T}_i(f) \mid \ell(e) = k-1\}$ over which we take the minimum. If however, the relabel is performed on the current minimum of (4.6) for f, then this minimum increases. □

In the Push-Relabel Algorithm for Polymatroid Sum, we perform pushes with respect to the lexicographically minimal buyer-item pairs, which automatically satisfies the condition of Lemma 4.19. Thus, we can apply the lemma to obtain monotonicity of (4.6) for every buyer i. This implies that, when fixing i, the potential items for a push at f are non-decreasing, and we ignore items that have been already considered until f is relabeled. Moreover, at a saturating push (4.6) strictly increases and thus, also the buyer-item pair, i.e., we move down the lexicographic order. Hence, the Push-Relabel Algorithm for Polymatroid Sum is indeed a valid implementation of the general push-relabel framework.

Corollary 4.20. *The Push-Relabel Algorithm for Polymatroid Sum returns an optimal solution to* Polymatroid Sum.

Running Time As we already discussed in Section 4.3, our running time analysis is not in terms of the input size but instead in terms of n (the number of polymatroids), m (the number of item types), and DO and EXO (the time for an oracle query). This is due to the fact that a compact encoding of the inputs is not possible in general and would yield to misleading statements about the performance of the algorithm.[7] As we already showed in Lemma 4.19, a saturating push makes the candidate set to find the minimum (4.6) strictly smaller and hence, the minimum strictly increases. Hence, the number of saturating pushes between relabels is bounded by mn. But what about the non-saturating pushes? Since the minimal labels do not necessarily increase, we do not get such a trivial upper bound as for saturating pushes. The main idea to get a bound after all is that in the Push-Relabel Algorithm for Polymatroid Sum, we always choose an ample item type with the maximum label below m for a push and the fact that an initially ample item type will not be ample after a non-saturating push and can only get ample again if there is a push on an item type with a higher label.

Lemma 4.21. *The number of non-saturating pushes in the Push-Relabel Algorithm for Polymatroid Sum is at most m^3.*

Proof. We call the set of iterations between two relabel operations a *phase*. As soon as every item type $e \in E$ has reached $\ell(e) = m$, the algorithm stops since then there is no item type left we can choose for a push. Since an item type of label m is never selected for either operation, there can be at most m^2 phases (m item types that are relabeled at most m times).

We will show that for each $e \in E$ there can be at most one non-saturating push in each phase. Here we use the algorithm's choice of an ample item e for which $\ell(e) = k$ is maximal. After a non-saturating push at e, e is not ample anymore. Hence, it cannot be picked for a push until it becomes ample again. A push on an item with label k' can make an item type ample but only on level $k' - 1$. Thus, e can only become ample again due to a push at f with $\ell(f) = k + 1$. However, no such item type can exist until a relabel operation (initiating the next phase) by choice of e as item type with maximum label. Hence, there can be at most one non-saturating push per item type per phase. As

[7] An illustrative example is the case for 3-MATROID INTERSECTION if the matroids are given explicitly by a list of their bases (of length that might be superpolynomial in $|E|$). Then the problem becomes easy by just comparing the intersection of every triple of bases. However, the problem is complete for NP if we assume the matroids are given by more succinct encodings (list of circuits or hyperplanes but also in particular assuming only an independence oracle). A detailed discussion is due to Mayhew [Mayo8].

there are at most m^2 phases and at most m non-saturating pushes per phase, we have that the total number of non-saturating pushes is bounded by m^3. □

Theorem 4.22. *The Push-Relabel Algorithm for Polymatroid Sum runs in* $\mathcal{O}(n\mathsf{DO} + nm^3\mathsf{ExO})$ *time.*

Proof. We can find an initial allocation in time $\mathcal{O}(n\mathsf{DO})$ time by asking every buyer $i \in N$ for a vector $x_i \in \mathcal{B}_i$.

According to Lemma 4.21, the number of non-saturating pushes is bounded by $\mathcal{O}(m^3)$. We can estimate the running time of all relabel operations and saturating pushes induced by a fixed item e while it is on level $\ell(e) = k$. Given e, we go through all buyer-item pairs in a structured way such that we consider every buyer-item pair at most once unless there is a non-saturating push. So, without counting the non-saturating pushes, an item e with fixed label k induces a running time of mn arithmetic operations and edge weight queries (in particular one call to $\mathsf{ExO}$for each buyer-item pair). Since there are m item types, that can happen for at most m different labels with a total addition of m^3 for non-saturating pushes.

In the analysis so far, we ignored the time needed to pick an ample item type e with the maximal possible label below m. We show now that this is not a bottleneck operation. We do so by using a standard network flow push-relabel argument (see e.g., [AMO88, Section 7.8]). In the beginning we initialize a list per label giving all ample item types that have this label. Initially, all lists are empty except for the list for label 0, which contains all ample item types. This construction takes $\mathcal{O}(m)$ time but is only needed once. Moreover, we keep track of the maximal label smaller than m that has a non-empty list. The list is easily maintainable, since only one item type changes its label and the status of being ample or not per iteration. The total increase of the maximal relevant label is bounded by $\mathcal{O}(m^2)$ (m item types with up to m labels). Thus, the total decrease is at most $\mathcal{O}(m^2)$. Hence, scanning the lists to find the first non-empty list is not a bottleneck operation. Having such a data structure, we can access any item type from the list for the maximum label in $\mathcal{O}(1)$ time. □

4.5 Excess Demand and Supply with Discrete Convexity

Recall the idea to determine an excess demand set in the Flow Auction. We first put every item type into a tier for every buyer that essentially determined how useful the

items of this type are for that buyer (E_i^1: crucial to get all, E_i^2: substitutable by others, E_i^3: fine to omit, E_i^4: crucial to omit. From this classification, we could determine how large the demand on a specific subset $F \subseteq E$ is and compare it against the supply.

As mentioned in the introduction of this chapter, we cannot use the same approach here since we cannot rank item types individually on how useful they are to a buyer; instead an item's usefulness to a buyer also depends on what other items a buyer receives with it.

Hence, we need a more sophisticated approach to determine the total requirement on a set $F \subseteq E$ if the valuation functions are not just additive functions but more general gross substitutes.

Recall from Definition 3.8 that i's requirement on F is defined by $r_i^p(F) = \min_{x \in \mathcal{D}_i(p)} x(F)$ and the total requirement on F is $r^p(F) = \sum_{i \in N} r_i^p(F)$. Note that the requirement can be defined only using $\breve{\mathcal{D}}_i(p)$, i.e., $r_i^p(F) = \min_{x \in \breve{\mathcal{D}}_i(p)} x(F)$ as all v_i are strong gross substitute. In particular, we can express the requirement in terms of the rank function of $\breve{\mathcal{D}}_i(p)$, i.e., $r_i^p(F) = \breve{\rho}_i^p(E) - \breve{\rho}_i^p(E \setminus F)$. For the dynamic auctions, that we present in the sequel, we are particularly interested in sets that maximize the shortage of an item set (or maximize the abundance) in order to adjust prices on items where the gap between supply and demand is bigger and hence, closing this gap faster. In the following, we use the shorthand $\text{short}^p(F) := r^p(F) - b(F)$ for the shortage of F at price p.

Lemma 4.23. *If all buyers have (strong) gross substitute valuations, then prices p are packing if and only if* $\text{short}^p(F) \leq 0$ *for all* $F \subseteq E$.

Proof. Let p be packing and $x = (x_i)_{i \in N}$ be a corresponding packing allocation. Without loss of generality, $x_i \in \breve{\mathcal{D}}_i(p)$. Then, using the polymatroid rank function associated with $\breve{\mathcal{D}}_i(p)$, i.e., $\breve{\rho}_i^p(F) = \max\{x_i(F) : x_i \in \breve{\mathcal{D}}_i(p)\}$, we have $\breve{\rho}_i^p(E) = x_i(E)$ for all $i \in N$. Thus,

$$\begin{aligned}\sum_{i \in N} \breve{\rho}_i^p(E) &= \sum_{i \in N} x_i(E) \\ &= \sum_{e \in E} \sum_{i \in N} x_i(e) \\ &= \sum_{e \in E} \min \left\{ \sum_{i \in N} x_i(e), b(e) \right\},\end{aligned}$$

where the last equality holds as x is packing. By Theorem 4.10, we have then

$$\begin{aligned}\sum_{i\in N}\check{\rho}_i^p(E) &= \sum_{e\in E}\min\left\{\sum_{i\in N}x_i(e), b(e)\right\}\\ &\le \max_{y\in\widetilde{\mathcal{D}}}\sum_{e\in E}\min\left\{\sum_{i\in N}y_i(e), b(e)\right\}\\ &= \min_{F\subseteq E}\left\{\sum_{i\in N}\check{\rho}_i^p(E\setminus F)+b(F)\right\}.\end{aligned}$$

Thus, we have for all $F\subseteq E$ that

$$\underbrace{\sum_{i\in N}\left(\check{\rho}_i^p(E)-\check{\rho}_i^p(E\setminus F)\right)}_{=r^p(F)}\le b(F). \tag{4.7}$$

That is, no F is in excess demand.

To show necessity, assume that there is no excess demand set at price p and use (4.7) to obtain

$$\begin{aligned}\sum_{i\in N}\check{\rho}_i^p(E) &\le \min_{F\subseteq E}\left\{\sum_{i\in N}\check{\rho}_i^p(E\setminus F)+b(F)\right\}\\ &= \max_{y\in\widetilde{\mathcal{D}}}\left\{\sum_{e\in E}\min\left\{\sum_{i\in N}y_i(e), b(e)\right\}\right\}\\ &\le \max_{y\in\widetilde{\mathcal{D}}}\left\{\sum_{e\in E}\sum_{i\in N}y_i(e)\right\}\\ &= \max_{y\in\widetilde{\mathcal{D}}}\left\{\sum_{i\in N}y_i(E)\right\}\\ &= \sum_{i\in N}\check{\rho}_i^p(E).\end{aligned}$$

Thus, equality must hold throughout the chain of inequalities and equalities. In particular, this means that there exists an allocation $y\in\widetilde{\mathcal{D}}$ with $\sum_{i\in N}y_i(e)\le b(e)$ for all $e\in E$, i.e., y is packing and so is p. □

The next step is to link a solution x of the Polymatroid Sum problem

$$\max_{x\in\widetilde{\mathcal{D}}}\sum_{e\in E}\min\left\{\sum_{i\in N}x_i(e), b(e)\right\} \tag{4.8}$$

to the problem of finding a set of maximum shortage.

Lemma 4.24. *Assume $x = (x_1, \dots, x_n)$ an optimal solution to* (4.8)*. Then, a set $S \subseteq E$ has maximum shortage if and only if the following three properties hold:*

(1) *$E \setminus S$ is tight (w.r.t. $\breve{\mathcal{D}}_i(p)$) for all $i \in N$,*
(2) *S does not contain an ample item type, and*
(3) *S contains all scarce item types.*

Moreover, there is a unique inclusion-wise minimal set fulfilling (1)*,* (2)*, and* (3)*.*

Proof. Let $S \subseteq E$ and p be any price vector. We have

$$\operatorname{short}^p(S) = \sum_{i \in N} \check{\rho}_i^p(E) - \left(\sum_{i \in N} \check{\rho}_i^p(E \setminus S) + b(S) \right).$$

Hence, S has maximum shortage if and only if it is a minimizer of the right hand side of (4.2). Since x is optimal, we get by the Min-Max Theorem for Polymatroid Sum,

$$\sum_{e \in E} \min \left\{ \sum_{i \in N} x_i(e), b(e) \right\} = \sum_{i \in N} \check{\rho}_i^p(E \setminus S) + b(S).$$

By Corollary 4.11, this is the case if and only if S satisfies (1), (2), and (3).
Moreover, if S and S' satisfy these three properties then $S \cap S'$ and $S \cup S'$ do so, too. Hence, there exists a unique inclusion-wise minimal set that satisfies those properties. □

We already defined the concept of an *exchange graph* for Knuth's Algorithm for Matroid Union that indicates for every buyer i and a set $I_i \in \mathcal{I}_i$ that is currently allocated to her which items can be added or exchanged such that the resulting set stays independent in buyer i's matroid. Recall that for the Polymatroid Sum problem, we presented a push-relabel algorithm to solve the problem and the concept of an exchange graph was not necessary. However, in this section the exchange graph becomes relevant again to perform the search for the inclusion-wise minimal excess demand set with maximum shortage. While $G(I)$ for the Matroid Union problem was defined for feasible solution I (i.e., a partition of E), our Push-Relabel Algorithm for Polymatroid Sum algorithm instead might oversell item types (i.e., they become scarce) but we can still define an exchange graph in a very similar way.
Given a solution x to (4.8) at price p, define $\check{G}_i(x_i) = (E, \check{A}_i)$ for $i \in N$ by

$$\check{A}_i = \{(e, f) \in E \times E \mid x_i - \chi_e + \chi_f \in \breve{\mathcal{D}}_i(p)\}.$$

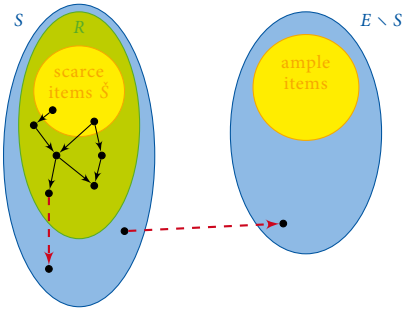


Figure 4.3: Proof sketch that an arbitrary set S satisfying Lemma 4.24 must contain R, which consequently also must satisfy Lemma 4.24. Dashed red edges cannot exist by construction of R (left) or due to Lemma 4.12 (right).

Note that the construction of this graph can be realized by querying the exchange oracle (ExO, see Section 4.3) about any (e, f) pair with $x_i(e) > 0$ and $x_i(f) < b(f)$. We let $\check{G}(x) = (E, \check{A}, \check{S})$, where $\check{A} = \bigcup_{i \in N} \check{A}_i$ and $\check{S} = \{e \in E \mid e \text{ is scarce}\}$.

Theorem 4.25. *Let x be an optimal solution to* (4.8)*. Then the minimal excess demand set of maximum shortage is the set $R = \{f \in E \mid f$ is reachable from some scarce item $e\}$ of items that is reachable from $\check{S}$ in $\check{G}(x)$.*

Proof. Let S be an excess demand set with maximum shortage. By Lemma 4.24, S contains all scarce item types and no ample item type (with respect to x). Moreover, $E \setminus S$ is tight for all $i \in N$ and hence, by Lemma 4.12, there is no edge $(e, f) \in \check{A}$ with $e \in S$ and $f \notin S$. Now consider R. By definition, R contains all scarce items. As all ample items are in $E \setminus S$ and there is no path from S to $E \setminus S$, we also cannot have any ample item in R. Also by definition, there is no edge from R to $E \setminus R$ and hence, by again invoking Lemma 4.12, $E \setminus R$ is tight for all $i \in N$. Thus, again using Lemma 4.24, R has maximum shortage. Finally $R \subseteq S$ since otherwise there is some item $f \in R \setminus S \subseteq E \setminus S$ that has a path from some scarce item e in $\check{G}$ and in particular an edge from S to $E \setminus S$, contradicting Lemma 4.12. □

Figure 4.3 depicts a visual aid to the proof of Theorem 4.25.

Additionally to a purely ascending auction, we also introduce descending auctions and hybrids of both in the next section. Intuitively, a price decrement is necessary if the demand on a subset of item types is too low, i.e., if there is excess supply on that set. Consequently, we also need to identify those excess supply sets and a search in the

exchange graph (this time starting at ample item types) seems like a good first idea. A subset S of item types is in excess supply if not all items from S are sold even if every buyer receives a preferred bundle that uses the maximum number of items in S. We formalize this in the following definition, where we use the rank functions of the polymatroids corresponding to $\widehat{\mathcal{D}}_i(p)$ for buyers $i \in N$i, i.e.

$$\hat{\rho}_i^p(F) := \max\{x(F) : x \in \widehat{\mathcal{D}}_i(p)\}.$$

Definition 4.26. Let p be a price vector. A set $F \subseteq E$ is in *excess supply*[8] if $\sum_{i\in N} \hat{\rho}_i^p(F) > b(F)$ and we call the difference $\sum_{i\in N} \hat{\rho}_i^p(F) - b(F)$ the *abundance* of F.

We denote the abundance of a set F by

$$\operatorname{abun}^p(F) := b(F) - \sum_{i\in N} \hat{\rho}_i^p(F).$$

Note that the abundance can be directly be defined with the rank functions $\hat{\rho}_i^p$ of the polymatroid corresponding to $\widehat{\mathcal{D}}_i(p)$, while the shortage required a detour via the requirement function. The intuition behind this is that the rank function is indeed an upper bound (or a capacity), whereas the requirement has to be understood as a lower bound (cf. Definition 3.8). In particular, we want to stress that abundance and shortage are in some sense complementary to describe the demand on a market but it is clearly not true that $\operatorname{short}^p(F) = -\operatorname{abun}^p(F)$ or even that a negative shortage implies a positive abundance. In economic literature, the function $\hat{\rho}_i^p$ is also called the *redundant*.

Using the notion of abundance and excess supply, we can formulate a lemma that is a counterpart to Lemma 4.23.

Lemma 4.27. *If all buyers have (strong) gross substitute valuations, then prices p are covering if and only if* $\operatorname{abun}^p(F) \leq 0$ *for all* $F \subseteq E$.

Proof. Observe that for any price vector p, we have

$$\max_{x\in\widehat{\mathcal{D}}(p)} \left\{ \sum_{e\in E} \min\left\{ \sum_{i\in N} x_i(e), b(e) \right\} \right\} \leq b(E). \tag{4.9}$$

[8] underdemanded

Moreover, we have equality in (4.9) if and only if there is a covering allocation $x \in \widehat{\mathcal{D}}(p)$ and then p is covering. Now assume that p is covering. Using the Min-Max Theorem for Polymatroid Sum (Theorem 4.10), we get

$$\begin{aligned}\min_{F\subseteq E}\left\{b(F)+\sum_{i\in N}\hat{\rho}(E\setminus F)\right\} &= \max_{x\in\widehat{\mathcal{D}}(p)}\left\{\sum_{e\in E}\min\left\{\sum_{i\in N}x_i(e),b(e)\right\}\right\}\\ &= b(E).\end{aligned}$$

We obtain

$$\begin{aligned} & b(F)+\sum_{i\in N}\hat{\rho}(E\setminus F)\geq b(E) && \text{for all } F\subseteq E\\ \iff\ & b(E\setminus F)+\sum_{i\in N}\hat{\rho}(F)\geq b(E) && \text{for all } F\subseteq E\\ \iff\ & \operatorname{abun}^p(F)\leq 0 && \text{for all } F\subseteq E.\end{aligned}$$

That is, there is no excess supply set F.

Next we show that there is a covering allocation if there is no excess supply set. By the same argument as above, we get

$$\begin{aligned} b(E) &\leq \min_{F\subseteq E}\left\{b(F)+\sum_{i\in N}\hat{\rho}(E\setminus F)\right\}\\ &= \max_{x\in\widehat{\mathcal{D}}(p)}\left\{\sum_{e\in E}\min\left\{\sum_{i\in N}x_i(e),b(e)\right\}\right\}\\ &\leq b(E),\end{aligned} \tag{4.10}$$

where the first inequality holds as there is no excess supply set and the equality holds by the Min-Max Theorem for Polymatroid Sum. Thus, (4.10) holds with equality everywhere, in particular for the last inequality. However, this is only possible if there is a covering allocation $x \in \widehat{\mathcal{D}}(p)$. □

Rearranging terms in the abundance formula yields

$$\begin{aligned}\operatorname{abun}^p(F) &= b(F)-\sum_{i\in N}\hat{\rho}_i^p(F)\\ &= b(E)-\left(b(E\setminus F)+\sum_{i\in N}\hat{\rho}_i^p(E\setminus(E\setminus F))\right)\end{aligned}$$

and from there we obtain the following corollary.

Corollary 4.28. *A set $F \subseteq E$ is maximal abundant under prices p if and only if*

$$E \setminus F \in \arg\min \left\{ b(S) + \sum_{i \in N} \hat{\rho}_i^p(E \setminus T) : T \subseteq E \right\}.$$

Moreover, F is a minimal excess demand set of maximum abundance if and only if $E \setminus F$ is a maximal minimizer of $S \mapsto b(S) + \sum_{i \in N} \hat{\rho}_i^p(E \setminus S)$.

This allows us to again use the Min-Max Theorem for Polymatroid Sum and the Push-Relabel Algorithm for Polymatroid Sum to find such a set. Recall the Polymatroid Sum problem, this time over $\widehat{\mathcal{D}}(p)$:

$$\max_{x \in \widehat{\mathcal{D}}} \sum_{e \in E} \min \left\{ \sum_{i \in N} x_i(e), b(e) \right\}. \tag{4.11}$$

We obtain an analogue of Lemma 4.24.

Lemma 4.29. *Assume $x = (x_1, \ldots, x_n)$ is an optimal solution to* (4.11)*. Then, a set $S \subseteq E$ has maximum abundance if and only if the following three properties hold:*

(1) *S is tight (w.r.t. $\widehat{\mathcal{D}}_i(p)$) for all $i \in N$,*
(2) *S includes all ample item types, and*
(3) *S does not contain a scarce item type.*

Moreover, there is a unique inclusion-wise minimal set fulfilling (1)*,* (2)*, and* (3)*.*

Analogously to the exchange graph to find a minimal excess demand set of maximum shortage, we now define an exchange graph to find a minimal excess supply set of maximum abundance. Let x be a solution to (4.11) and define $\hat{G}_i(x_i) = (E, \hat{A}_i)$ with

$$\hat{A}_i = \{(e, f) \in E \times E : x - \chi_e + \chi_f \in \widehat{\mathcal{D}}_i(p)\}.$$

We define further $\hat{G}(x) = (E, \hat{A}, \hat{S})$ with $\hat{A} = \bigcup_{i \in N} \hat{A}_i$ and $\hat{S} = \{e \in E \mid e \text{ is ample}\}$.

Theorem 4.30. *Let x be an optimal solution to* (4.11)*. Then the minimal excess supply set of maximum abundance is the set $R = \{f \in E \mid f$ is reachable from some ample item $e\}$ of items that is reachable from $\hat{S}$ in $\hat{G}(x)$.*

Note that the set R in Theorems 4.25 and 4.30 can be obtained by simply using a breadth-first search, and hence, in time linear in the number of edges of the respective exchange

graph. Thus, given a solution to (4.8) and (4.11), respectively, the main workload is to construct the exchange graph.

Lemma 4.31. *The minimal excess demand set of maximum shortage and the minimal excess supply set of maximum abundance can be computed in time* $\mathcal{O}(nm^2 \cdot \mathsf{ExO})$*, given an optimal solution* x *to* (4.8) *and* (4.11)*, respectively.*

Lemma 4.31 together with Theorem 4.22 (or Theorem 4.18) yields the total running time to find excess demand and supply sets.

Corollary 4.32. *In the multi-supply setting, given prices* p*, we can compute a minimal excess demand set of maximum shortage and a minimal excess supply set of maximum abundance in time* $\mathcal{O}(n\mathsf{DO} + nm^3\mathsf{ExO})$*. In the unit supply setting, the running time reduces to* $\mathcal{O}(n\mathsf{DO} + (m^3 + nm^2)\mathsf{ExO})$*.*

Compared to the previous state of the art by Murota, Shioura, and Yang [MSY13] which has a running time of $\mathcal{O}(n\mathsf{DO} + nm^4 \log(nmB)\mathsf{ExO})$, where $B = \max_{e \in E} b(e)$, we gain a factor of $m \log(nmB)$; in particular, we lose the dependence on B.

4.6 The Auctions

We already established that a dynamic auction essentially simulates a (discrete) gradient descent method on the Lyapunov. More precisely, given any price vector p and a set $F \subseteq E$, if a dynamic auction increases (or decreases) the price on F the Lyapunov decreases if and only if F is an excess demand set (or an excess supply set, respectively) and the decrement is proportional to the shortage (or abundance, respectively) of F under price p.

As the Lyapunov is submodular and the minimizers of any submodular function form a lattice, by Lemma 3.14, this is also true for the Walrasian price vectors. In particular, there exists a unique buyer-optimal, i.e., minimal Walrasian price vector p_* and a unique seller-optimal, i.e., maximal Walrasian price vector p^*.[9]

Ausubel [Aus06] has shown that p_* and p^* can be obtained via the following Ascending Auction and Descending Auction, respectively. Note that the former is just an imple-

[9] Note however, that *seller-optimal Walrasian* does not need to mean *revenue-maximizing*: A single buyer on a market with two items $\{e_1, e_2\}$ and a unit demand valuation of $\tilde{v}_1 = (3, 1)$ yields a maximal Walrasian price vector of $(2, 0)$ with allocation $\{e_1\}$. However, the price vector $(3, 2)$ leads to the same allocation but with revenue 3 instead of 2.

mentation of the Generic Ascending Auction by Gul and Stacchetti [GSoo] and in fact the Flow Auction is also just an implementation of the Ascending Auction.

Ascending Auction:

Sufficiently low price vector p
while *an excess demand set* $F \subseteq E$ *exists* **do**
    $F := \arg\min\{|F'| : F' \subseteq E \text{ with } \operatorname{short}^p(F') \text{ maximum}\}$
    for $e \in F$ **do**
        $p(e) := p(e) + 1$
return p

Descending Auction:

Sufficiently high price vector p
while *an excess supply set* $F \subseteq E$ *exists* **do**
    $F := \arg\min\{|F'| : F' \subseteq E \text{ with } \operatorname{abun}^p(F') \text{ maximum}\}$
    for $e \in F$ **do**
        $p(e) := p(e) - 1$
return p

A minor defect of these two auctions might be the fact that the starting prices of item types cannot be arbitrary: if the Ascending Auction has a starting price that is higher than the minimal Walrasian price vector, then it cannot find this price vector at all. Even more crucially, if its starting price is higher than the maximal Walrasian price vector, then the Ascending Auction does not find a Walarasian equilibrium at all, since it cannot resolve excess supply sets and hence, will not obtain a covering allocation. It is natural to also consider hybrids of the previous two auction and these allow for an arbitrary starting price vector. However, those auction will not necessarily find one of the extremal Walrasian price vectors but just an arbitrary one. The first auction has two phases; in the first phase we eliminate all excess demand sets by increasing prices and in the second we eliminate the excess supply sets by decreasing prices. Importantly, since we choose the excess demand and excess supply sets carefully (by not increasing prices of items with a shortage and not decreasing prices on items with abundance), we will never create additional excess supply sets and excess demand sets, respectively during the process.

Two-Phase Auction:

```
Arbitrary price vector p
while excess demand set F ⊆ E exists do
    F := arg min{|F'| : F' ⊆ E with short^p(F') maximum}
    for e ∈ F do
        p(e) := p(e) + 1
while excess supply set F ⊆ E exists do
    F := arg min{|F'| : F' ⊆ E with abun^p(F') maximum}
    for e ∈ F do
        p(e) := p(e) − 1
return p
```

Murota, Shioura, and Yang [MSY13] shows that one can reduce the number of iterations by interleaving both phases and always performing an increasing or decreasing step depending on which of the two provides the greater reduction of the Lyapunov. This results in the following Greedy Auction.

Greedy Auction:

```
Arbitrary price vector p
while an excess demand set F_1 ⊆ E or excess supply set F_2 ⊆ E exists do
    F_1 := arg min{|F'| : F' ⊆ E with short^p(F') maximum}
    F_2 := arg min{|F'| : F' ⊆ E with abun^p(F') maximum}
    if short^p(F_1) ≥ abun^p(F_2) then
        for e ∈ F_1 do
            p(e) := p(e) + 1
    else
        for e ∈ F_2 do
            p(e) := p(e) − 1
return p
```

The latter two auctions of course loose the property that we are guaranteed to find a minimal or maximal Walrasian price vector, provided that the starting price vector is not component-wise lower (or higher, respectively) than the minimal (or maximal) Walrasian price vector.

4.7 Extremal Equilibrium Prices

We already mentioned that the set of Walrasian prices forms a lattice (with component-wise minimum and maximum as meet and join operation, respectively) as they are the set of minimizers of a submodular function, namely the Lyapunov (cf. Lemma 3.14). As a consequence, there exists a unique minimal and a unique maximal Walrasian price vector, p_* and p^*.

Packing and covering prices are already interesting on their own as they both constitute weaker notions of equilibria (*packing*: every buyer gets a preferred bundle, *covering*: all items are sold). Moreover, packing and covering prices are guaranteed to exist, also if the valuation functions are not gross substitutes, e.g., by setting the prices very high (such that no buyer is interested in any good) or to zero, respectively. It is a natural question whether these relaxations of Walrasian equilibria inherit the lattice properties and whether minimal packing prices or maximal covering prices (if they exist) are guaranteed to be Walrasian. Particularly, the question whether minimal packing and maximal covering prices exist is interesting.

We first demonstrate that packing and covering do not have a unique minimum and maximum, respectively, if the valuations are not gross substitutes and hence, cannot form a lattice.

Example 4.4. Given a market with four item types $E = \{e_1, e_2, e_3, e_4\}$ where $b(e) = 1$ for each $e \in E$ and two buyers $N = \{1, 2\}$. The valuations for the first buyer are given by the item-wise valuations $\tilde{v}_1 = (6, 6, 6, 10)$ and the (non-matroidal) independence system $\mathcal{I}_1 = \{I \subseteq F \mid F \in \{\{e_1, e_2, e_3\}, \{e_4\}\}\}$. The valuation of a bundle S is given by $v_1(S) = \max_{I \in \mathcal{I}_1} \sum_{e \in I \cap S} \tilde{v}_1(e)$. The valuation function for the second buyer is defined similarly using item-wise valuations $\tilde{v}_2 = (10, 6, 6, 6)$ and the independent sets $\mathcal{I}_2 = \{I \subseteq F \mid F \in \{\{e_1\}, \{e_2, e_3, e_4\}\}\}$.

First we show that there is no packing price vector p with $\sum_{e \in E} p(e) < 8$. Consider such a vector p and let E' be all elements in $e \in E$ with $p(e) < 6$. Clearly, $|E \setminus E'| \leq 1$. The unique preferred bundle for buyer 1 is $\{e_1, e_2, e_3\} \cap E'$ and for buyer 2, it is $\{e_2, e_3, e_4\} \cap E'$. Since $|\{e_1, e_2, e_3\} \cap \{e_2, e_3, e_4\}| = 2$ the intersection of the preferred bundles is non-empty, so p cannot be a packing price vector.

Moreover, there are packing price vectors with $\sum_{e \in E} p(e) = 8$, namely $p = (0, 2 + \alpha, 6 - \alpha, 0)$ with $\alpha \in [0, 4]$ are packing price vectors, since the first buyer is indifferent between $\{e_1, e_2, e_3\}$ and $\{e_4\}$ and the second one is indifferent between $\{e_1\}$ and $\{e_2, e_3, e_4\}$.

Thus, the allocation $\{e_4\}$ to the first buyer and $\{e_1\}$ to the second buyer is packing. This family of packing prices gives a component-wise minimal vector $(0, 2, 2, 0)$ which itself is not packing. However, since $\sum_{e \in E} p(e) = 8$ for all price vectors in the defined family, they are all buyer-optimal packing prices. Hence, clearly, the buyer-optimal packing prices are not unique and moreover these buyer-optimal packing price vectors are all not covering.

Note that the valuation functions in this example are not strong gross substitutes. Consider the valuation function of the first buyer. It holds that $\{e_1, e_2, e_3\} \in \mathcal{D}_1(\mathbf{0})$. But if we increase the prices of items e_1 and e_2 by 6, i.e., $p = (6, 6, 0, 0)$, there is no demand for e_3 anymore since $\mathcal{D}_1(p) = \{\{e_4\}\}$. Similarly, the valuation function of the second buyer is also not strong gross substitutes.

Example 4.5. Given a market with three item types $E = \{e_1, e_2, e_3\}$ and unit supply, i.e, $b(e) = 1$ for each $e \in E$, and two buyers $N = \{1, 2\}$. Both buyers have the same valuation function, given by the following table:

S	$\varnothing$	$\{e_1\}$	$\{e_2\}$	$\{e_3\}$	$\{e_1, e_2\}$	$\{e_1, e_3\}$	$\{e_2, e_3\}$	E
$v_i(S)$	0	7	7	8	14	13	13	18

Then $p = (6, 7, 7)$ is covering, since $\mathcal{D}_i(p) = \{\{e_1\}, \{e_1, e_2\}, \{e_3\}\}$ for both $i \in N$. Moreover, $p' = (7, 7, 6)$ is covering with $\mathcal{D}_i(p) = \{\{e_2\}, \{e_1, e_2\}, \{e_3\}\}$ for both $i \in N$. However, $q = p \vee p' = (7, 7, 7)$ is not covering, as $\mathcal{D}_i(q) = \{\{e_3\}\}$.

Note however, that there is a Walrasian price vector above all these price vectors, which is $p^* = (7, 7, 8)$.

The following example shows that even with gross substitute valuations (even unit demand), we do not have the property that packing prices form a lattice as the join of two packing price vectors is not necessarily packing.

Example 4.6. Consider a market with three item types $E = \{e_1, e_2, e_3\}$ and unit supply, i.e., $b(e) = 1$ for each $e \in E$, and two buyers $N = \{1, 2\}$. Both $i \in N$ buyers have the same unit demand valuation function (i.e., gross substitute) with item-wise valuation $\tilde{v}_i = (2, 2, 2)$. Now consider the price vectors $p = (1, 1, 3)$ and $p' = (3, 1, 1)$. Then the demand correspondences are $\mathcal{D}_i(p) = \{\{e_1\}, \{e_2\}\}$ and $\mathcal{D}_i(p') = \{\{e_2\}, \{e_3\}\}$ for each $i \in N$ and hence, both p and p' are packing as we can assign one item each under those prices. However, the price vector $q = p \vee p' = (3, 1, 3)$ is not packing as $\mathcal{D}_i(q) = \{\{e_2\}\}$ for both $i \in N$.

On the other hand, we can show that there is a unique minimal packing price vector and a unique maximal covering price vector. Moreover, we can also prove that these vectors are equal to the minimal, respectively maximal, Walrasian price vector. The necessary lemmas and the theorems that follow are just stated here; their proofs can be found in the published research paper [Eic+25] and will also get a special treatment in the PhD thesis of Katharina Eickhoff.

As we have seen in the previous sections, it is beneficial in an algorithmic context, to distinguish between the unit supply case (where $b(e) = 1$ for all $e \in E$) and the case where items are available in higher quantities since we can achieve better running times.

However, from a purely structural view, there is not really a difference between those two cases: every market with multiple quantities of item types can be transferred into a unit supply market by simply treating every unit of an item type as an individual object. Formally, let E be a set of item types and $b \in \mathbb{Z}_+^E$ the supply vector. Then we can define $\tilde{E} := \{\tilde{e}_1, \dots, \tilde{e}_{b(e)} : e \in E\}$ as the set of all copies of all item types $e \in E$. In that case a set $S \subseteq \tilde{E}$ corresponds to some allocation $x \in [0, b]_{\mathbb{Z}}$ if for all $e \in E$ it holds that $x(e) = |\{\tilde{e} \in S \mid \tilde{e} \text{ is a copy of } e\}|$. Under abuse of notation, we may write $v(S)$ instead of $v(x)$ in that case. The remaining results of this chapter are all rather structural and not very algorithmic, so for the sake of convenience, we restrict ourselves to the unit supply setting.

Note that the reduction to the unit supply case is not completely exact in the following sense: If we consider a multi-supply market on (E, b), we force the auctioneer to charge the same price for each unit of an item type, whereas in the unit supply market on $\tilde{E}$ corresponding to (E, b) different copies of the same item might have different prices. It seems counter-intuitive but those anomalies can also happen at equilibrium as the following very simple example demonstrates.

Example 4.7. Consider a market with one item type $E = \{e\}$, $b(e) = 2$, and one buyer $N = \{1\}$ with valuation $v(0) = 0, v(1) = 2, v(2) = 3$. On $\tilde{E} = \{\tilde{e}_1, \tilde{e}_2\}$ the discriminating price vector $\tilde{p} = (0, 1)$ is Walrasian with allocation $(\{\tilde{e}_1, \tilde{e}_2\})$.

However, when we consider extremal Walrasian prices, i.e., minimal and maximal ones, then this subtlety does not matter as the minimal and maximal Walrasian prices on $\tilde{E}$ coincide with the minimal and maximal Walrasian prices on (E, b), respectively, as the following lemma expresses.

Lemma 4.33. *A vector p is the minimal (maximal) Walrasian price vector in a market on (E, b) if and only if $\tilde{p} = ((p(e))_{k=1}^{b(e)})_{e \in E}$ is the minimal (maximal) Walrasian price vector in a market on $\tilde{E}$.*

The following lemma states that if all buyers have strong gross substitute valuations on E, then the described reduction to the unit supply setting results in valuation functions are gross substitutes.

Lemma 4.34 ([MSY13, Proposition A.1])**.** *Given a strong gross substitutes valuation $v_i : [0, b]_{\mathbb{Z}} \to \mathbb{Z}_+$ the corresponding valuation after reduction to the unit supply setting $v_i : 2^{\tilde{E}} \to \mathbb{Z}_+$ is gross substitute.*

4.7.1 Minimal Packing Prices are Walrasian

Ben-Zwi, Lavi, and Newman [BLN13] already showed that a packing price vector p on a unit supply market is necessarily Walrasian if there is a Walrasian price vector p' such that $p \leq p'$. Their argument can be carried forward fairly easily to the multi-supply setting.

Proposition 4.35 (generalized from [BLN13, Proposition 4.13])**.** *Let p be Walrasian and let $q \leq p$ be a packing price vector. Then q is Walrasian.*

However, this does not imply that there exists a unique minimal packing price vector as it is not a priori clear whether there can exists a packing price vector p for which $p(e) < p_*(e)$ but $p(f) > p_*(f)$ for some $e, f \in E$. We can show that this is indeed not the case.

Lemma 4.36. *Let p_* be the minimal Walrasian price vector and q be any packing price vector. Moreover, let x be a packing allocation w.r.t. q. Then $p_*(e) \leq q(e)$ for all $e \in E$ with $\sum_{i \in N} x_i(e) > 0$.*

Using this lemma, we can show that indeed any packing price vector is necessarily component-wise not smaller than the minimal Walrasian price vector.

Theorem 4.37. *Let p_* be the minimal Walrasian price vector. Then $p_* \leq q$ for all packing price vectors q.*

As a direct consequence of Theorem 4.37, we obtain the following corollary.

Corollary 4.38. *There exists a component-wise minimal packing price vector and its equal to the minimal Walrasian price vector.*

4.7.2 Maximal Covering Prices are Walrasian

We now establish a analogous theorem for maximal covering prices and show that those coincide with maximal Walrasian prices. By Lemma 4.34, it suffices to do so for the unit supply setting. The roadmap towards such a theorem is the following: given a unit supply market on E with buyers $i \in N$ and gross substitute valuations $v_i \colon 2^E \to \mathbb{Z}_+$, let p^* be the maximal Walrasian price vector and q be a covering price vector. We fix a covering allocation $(S_i)_{i \in N}$ w.r.t. q. As $(S_i)_{i \in N}$ is covering, we have $\bigcup_{i \in N} S_i = E$ but it does not necessarily hold that $S_i \cap S_j = \varnothing$ for pairwise different $i, j \in N$. The latter is the main difficulty for a complementary result to Theorem 4.37 as a covering allocation does not need to be feasible in the first place. We denote by $k_e = |\{i \in N \mid e \in S_i\}|$ how often an item e is allocated under allocation $(S_i)_{i \in N}$. Then we construct an market instance in which k_e copies of e are available for each $e \in E$, i.e., $E' = \{e'_1, \dots, e'_{k_e} : e \in E\}$, and show that q (the covering price vector for E) is Walrasian for E'. To this end, let $\psi \colon E' \to E, e'_i \mapsto e$ be the projection of E' onto E and we define the new valuation functions $v'_i \colon E' \to \mathbb{Z}_+$ for each $i \in N$ by $v'_i(S') = v_i(\psi(S'))$.

For a price vectors $p \in \mathbb{Z}_+^E$ and $p' \in \mathbb{Z}_+^{E'}$ we denote the demand correspondences in the respective markets by $\mathcal{D}_i(p)$ and $\mathcal{D}'_i(p')$, respectively. As a shorthand, we also define for some price vector $p' \in \mathbb{Z}_+^{E'}$ for the market on E' a corresponding price vector for the market on E that uses the minimal price among all copies, i.e., $p^\downarrow \in \mathbb{Z}_+^E$ with

$$p^\downarrow(e) = \min\{p'(e'_\ell) : 1 \le \ell \le k_e\}$$

for all $e \in E$.

Lemma 4.39. *Let $p' \in \mathbb{Z}_+^{E'}$ and $S' \subseteq E'$ be an arbitrary set. Then $S' \in \mathcal{D}'_i(p')$ for some $i \in N$ if and only if the following conditions hold:*

1. *$\psi(S') \in \mathcal{D}_i(p^\downarrow)$,*
2. *S' only contains cheapest copies of each item, and*
3. *whenever S' contains multiple copies of an item, all of them have price 0.*

Lemma 4.40. *If $v_i \colon 2^E \to \mathbb{Z}_+$ is gross substitute, then $v'_i \colon 2^{E'} \to \mathbb{Z}_+$ is gross substitute.*

Lemma 4.41. *Let $p \in \mathbb{Z}_+^E$ be a covering price vector on a market on E. Then $p' \in \mathbb{Z}_+^{E'}$ with $p(e') = p(\psi(e'))$ for all $e' \in E'$ is a Walrasian price vector on the market for E'.*

Using these lemmas, we can prove a complementary theorem of Theorem 4.37 for covering prices.

Theorem 4.42. *Let p^* be the maximal Walrasian price vector. Then $p^* \geq q$ for all covering price vectors q.*

Corollary 4.43. *There exists a component-wise maximal covering price vector and its equal to the maximal Walrasian price vector.*

4.8 Monotone Comparative Statics

In this final section, we present additional results on monotonicity in supply and demand for the (unique) buyer-optimal, as well as for the (again unique) seller-optimal Walrasian prices, provided that all buyers have (strong) gross substitute valuation functions. This shows that Walrasian equilibrium prices react *naturally* to changes in supply or demand.[10] Decrease in supply of an item type $e \in E$ is straight-forward: We just decrease the supply vector b at position e. The decrease in demand, we model as an *item truncation*, i.e., instead of following the valuation function v_i as it is, we consider again a demand d_i (cf. the item-capped additive valuations from Chapter 3).

Definition 4.44. Let $v_i \colon [0, b]_{\mathbb{Z}} \to \mathbb{Z}_+$ be a valuation function and $d_i \in \mathbb{Z}_+$. The *item truncation* of v_i w.r.t. d_i is given by

$$v_i^{d_i}(x) := \max\{v_i(y) \mid y \leq x, \|y\|_1 \leq d_i\}.$$

Note that if we choose d_i large enough, say $d_i = \|b\|_1$, then $v_i^{d_i} \equiv v_i$. Under abuse of notation, we may also write v_i^d for a demand vector $d = (d_i)_{i \in N}$ to refer to the truncated valuation function $v_i^{d_i}$.

Using a non-trivial Lemma by Collina and Weinberg [CW20], we have that (strong[11]) gross substitutes are closed under item truncation, i.e., if v_i is (strong) gross substitute, then so is $v_i^{d_i}$ for any $d_i \in \mathbb{Z}_+$.

We can then show two monotonicity results:

Theorem 4.45. *(i) Let p_*, p^* be the buyer- and seller-optimal Walrasian price vector w.r.t. some supply vector $b = (b(e))_{e \in E}$ and $p'_*, p^{*\prime}$ the buyer- and seller-optimal Walrasian price vectors w.r.t. some supply vector $b' \leq b$. Then for all $e \in E' = \{f \in E \mid b'(f) > 0\}$ it holds that $p'_*(e) \geq p_*(e)$ and $p^{*\prime}(e) \geq p^*(e)$.*

[10] While our paper [Eic+25] was under revision at TEAC, Raach published equivalent results [Raa24].
[11] if we apply the reduction from Lemma 4.34

(ii) Let p_*, p^* be the buyer- and seller-optimal Walrasian price vector w.r.t. some demand vector $d = (d_i)_{i \in N}$ and $p'_*, p^{*\prime}$ the buyer- and seller-optimal Walrasian price vectors w.r.t. some demand vector $d' \leq d$. Then for all $e \in E$ it holds that $p'_*(e) \leq p_*(e)$ and $p^{*\prime}(e) \leq p^*(e)$.

The proof of this theorem can be found in detail in our article [Eic+25] and in the upcoming PhD thesis of Katharina Eickhoff. Theorem 4.45 *(i)* can be viewed as a monotonicity result w.r.t supply wheres as *(ii)* captures price monotonicity in demand.

Note that there is no direct dependency which prices increase if the supply of say only one item decreases. As item types are substitutes, the shortage that is created by an item type which has now a lower supply can cause a price increase of any other item.

There is not a direct way to generalize the result of Theorem 4.45 *(ii)* to other changes in the valuation functions (which would include, for instance, per-item demands) as the following example shows.

Example 4.8. Consider a unit supply market with three items $E = \{e_1, e_2, e_3\}$ and three buyers with unit demand valuations $\tilde{v}_1 = (2, 3, 0)$, $\tilde{v}_2 = (0, 1, 1)$ and $\tilde{v}_3 = (0, 1, 1)$, i.e., $v_i(S) = \max_{e \in S} \tilde{v}_i(e)$ for all $S \subseteq E$. The buyer-optimal Walrasian prices are given by $p_* = (0, 1, 1)$. Now, assume that the valuation of the first buyer for the second item e_2 is decreased by one, i.e., $\tilde{v}'_1 = (2, 2, 0)$. In this setting, the prices $p'_* = (0, 0, 0)$ is the buyer-optimal Walrasian price vector. Hence, the price of the items whose valuation was not changed are the same or reduced by one unit.

Consider the same setting again and assume that the first buyer decreases her valuation for item e_1 as well, i.e., $\tilde{v}''_1 = (1, 2, 0)$. The minimal Walrasian price vector is $p''_* = (0, 1, 1)$. Thus, the price of the items whose valuation was not changed are increased by one unit.

The example shows that the minimal Walrasian prices do not change in a monotone way when the valuations change. This seems to be intuitive since if the valuation of one item is decreased, it could increase the demand for other items which then have a shortage and the price would need to increase. On the other hand, the item itself is less attractive, such that the demand can reduce and thus also potential competition for that item vanishes.

Also note that the question regarding price monotonicity w.r.t. changes in valuations is a somewhat very restricted problem, as the strong gross substitutes property might get lost if one is not careful with the changes in the valuation function.

CHAPTER 5

The Greedy+ Algorithm for Submodular Cover

> *And now for something completely different.*
>
> — **Monty Python**
> (British comedy group)

The greedy paradigm is probably the most used first approach to tackle all sorts of combinatorial optimization problems algorithmically. On one hand, the idea to always make a step that provides the most bang per buck is very natural and resembles the steepest descent method from continuous optimization, which guarantees to find at least a local optimum. On the other hand, due to its simplicity, the analysis of a greedy algorithm does not require too many case distinctions or other pitfalls as the selection rule for the next step is simple and fast and the structure of the solution has some sort of monotonicity property that can be exploited in the analysis of the quality of the solution. However, it is also known that for most combinatorial optimization problems, a greedy algorithm might return a solution that can be arbitrarily bad compared to the optimum; linear optimization problems on matroids being the notable exception. In this chapter, we consider the Submodular Cover problem,[1] which generalizes a lot of combinatorial optimization problems and suggest an improved greedy algorithm which finds an optimal solution to a Submodular Cover problem if its structure is simple enough.

[1] In the literature, one may also find the name Submodular Set Cover.

5.1 Introduction

A set function $z\colon 2^E \to \mathbb{Z}_+$ is called *normalized* if $z(\varnothing) = 0$ and *monotone* if for all $S \subseteq T$ it holds that $z(S) \leq z(T)$. The Submodular Cover problem on a finite *ground set* E can be formally described using just two functions, a linear *cost function* $c\colon E \to \mathbb{Z}_+$ and a monotone submodular *coverage function* $z\colon 2^E \to \mathbb{Z}_+$. The task is to find a *cover* of z, i.e., a set $S \subseteq E$ with $z(S) = z(E)$ of minimum cost $c(S) = \sum_{e \in S} c(e)$.

Submodular Cover

Given: finite ground set E, monotone submodular coverage function $z\colon 2^E \to \mathbb{Z}_+$, linear costs $c\colon E \to \mathbb{Z}_+$.

Find: cover $S \subseteq E$ ($z(S) = z(E)$) with $c(S)$ minimum.

As a gentle start, we consider the Set Cover problem, whose decision version is among Karp's initial list of NP-complete problems [Kar72] and hence, it is unlikely that it can be solved in polynomial time by a deterministic algorithm.

Set Cover

Given: universe $\mathcal{U}$ of points, a set of subsets $E = \{e_1, \ldots, e_n\}$ ($e_i \subseteq \mathcal{U}$) and a cost function $c\colon E \to \mathbb{R}$.

Find: $S \subseteq E$ of minimum cost such that $\bigcup_{e \in S} e = \mathcal{U}$.

For such a Set Cover instance $(\mathcal{U}, E, c)$, we can define a *coverage* function $z\colon 2^E \to \mathbb{Z}_+$ by $z(S) = |\bigcup_{e \in S} e|$. Note that this function is indeed monotone and submodular. We can use in the following natural greedy algorithm which iteratively selects the subset that offers the cheapest cost-per-unit ratio.

Wolsey's Greedy:

Input: A monotone submodular function $z\colon 2^E \to \mathbb{Z}_+$ and a linear cost function $c\colon E \to \mathbb{R}_+$
Output: A cover S

```
1 S := ∅
2 while z(S) < z(E) do
3     ē := arg min { c(e) / (z(S+e)−z(S)) : e ∈ E with z(S + e) > z(S) }
4     S := S + ē
5 return S
```

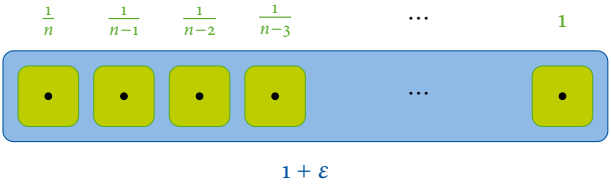


Figure 5.1: Set Cover instance from Example 5.1. The costs of the singleton sets are given above the set in green, while the cost of the large set containing all elements are labeled below the set in blue.

Wolsey [Wol82] showed that Wolsey's Greedy is a $H(\max_{e \in E} z(e))$-approximation, where $H(k)$ is the k-th harmonic number, i.e., $H(k) = \sum_{i=1}^{k} \frac{1}{i} \leq 1 + \ln k$. This does not hold only if z is a coverage function of a Set Cover instance but also for the more general Submodular Cover problem.

Proposition 5.1 ([Wol82, Proposition 3]). *Wolsey's Greedy is an $H(\max_{e \in E} z(e))$-approximation algorithm for* Submodular Cover.

Interestingly, this bound is tight already for a very simple instance presented in the following example.

Example 5.1. Consider the Set Cover instance with universe $\mathcal{U} = \{1, \ldots, n\}$ and the family of sets $E = \{e_0, e_1, e_2, \ldots, e_n\}$ with $e_0 = \mathcal{U}$, $e_i = \{i\}$ for all $0 < i \leq n$, and costs $c(e_0) = 1 + \varepsilon$, with some $\varepsilon > 0$ and $c(e_i) = \frac{1}{i}$ for all $0 < i \leq n$. The instance is also depicted in Figure 5.1. Then Wolsey's Greedy returns the solution $\{e_1, \ldots, e_n\}$ with cost $H(n)$, whereas the optimal solution $\{e_0\}$ has cost $1 + \varepsilon$.

Another theorem by Moshkovitz [Mos12] (building upon Feige's slightly weaker result [Fei98]) shows that it is impossible that there can exist a better algorithm for Submodular Cover (and even Set Cover) than Wolsey's Greedy unless P = NP.

Proposition 5.2 ([Mos12, Theorem 1.2]). *For any $\varepsilon > 0$, approximating* Set Cover *on inputs of size n better than $(1 - \varepsilon) \ln n$ is* NP*-hard.*

However, there is a very simple adaption to Wolsey's Greedy that would have found the optimal solution to the instance in Example 5.1: By taking into account that the addition of the set e_0 would have allowed to remove all other sets added so far, the algorithm could have saved costs for redundant elements. That is, instead of looking at

the ratio $\frac{c(e)}{z(S+e)-z(S)}$, one could instead consider the ratio $\frac{c(e)-c(R(S,e))}{z(S+e)-z(S)}$, where $R(S,e)$ is the most expensive set of elements $f \in S$ that can be removed from S after adding e without losing coverage. Note that determining the set $R(S,e)$ is not necessarily easy but in this chapter we will learn about a few functions where it is.

Summarizing, the greedy approach often fails as it is easy to trick the algorithm into making an initial decision that looks good but does not pay off later. In this chapter, we consider a subclass of Submodular Cover problems, which are forgiving for those kind of mistakes in the sense that an optimal solution can still be recovered by simply reverting some steps that previously have been made which turned out to be useless. Accompanying with a sufficient condition for those kind of problems, we give a greedy algorithm that solves those problems optimally.

5.1.1 Contribution

These here presented results are unpublished at the time of writing this thesis.

We define a subclass of submodular functions that we call *generalized ranks*, which have a one-to-many exchange property that is stronger than the (one-to-one) base exchange property of matroids (cf. Section 1.3.3). A natural greedy algorithm for the Submodular Cover problem on these functions selects an element in each iteration that has the cheapest cost-to-coverage ratio but accounting for cost saved via deleting redundant elements. If the generalized rank has some nice property in an auxiliary exchange graph, we can show that this algorithm guarantees an optimal solution and its running time is polynomial if and only if the identification of redundant elements can be done in polynomial time. Further, we developed the foundations of some duality theory on generalized ranks.

5.1.2 Related Work

Matroid theory originated in the 1930s by the work of Whitney [Whi35]. He considered them as an abstract generalization of matrices and the notion of linear independence. Our notation is mainly based on the modern monograph by Oxley [Oxl06]. As described above, optimizing a linear function of a matroid is easy using he matroid greedy algorithm and its correctness stems from the base exchange property of matroids. There have been various attempts to generalize or adapt this exchange property to achieve

even more general abstractions. Notable examples are *greedoids* by Korte and Lovász [KL84] and *exchange systems* by Brualdi and Scrimger [BS68].

Finding a minimum cost base (or spanning set) of a matroid (or any independence system) can easily also be recognized as a *covering problem*, which are among the most studied combinatorial optimization problems. Any minimizing integer program with non-negative data and $\geq$-constraints can essentially be viewed as a covering problem. Among them are also various hard problems such as Set Cover or Vertex Cover whose decision variants are on Karp's original list of NP-complete problems [Kar72]. However, while solving those problems optimally in polynomial time seems unlikely, there are elegant methods to achieve approximate solutions, sometimes with a greedy algorithm. In particular, Chvátal's work [Chv79] for the Set Cover problem and Dobson's paper on greedy heuristics for integer programming [Dob82] make a good starting point in the literature. Wolsey [Wol82] gives an analysis of the greedy algorithm for the very general Submodular Cover problem, which is in the current interest. In particular, Wolsey shows that Submodular Cover can be approximated up to a factor of $1+\ln(\max_{e\in E} z(e))$ and a result by Moshkovitz [Mos12] building upon a slightly weaker result by Feige [Fei98] shows that this is best possible unless P = NP. It is also natural to use greedy algorithms that are based on linear programming duality. Hochbaum [Hoc82] showed that a dual greedy algorithm (with a clean-up phase at the end) achieves an approximation factor for Set Cover of $\max_{p\in\mathcal{U}} |\{e \in E \mid p \in e\}|$. Extending this idea [Fuj00] yields an approximation algorithm for Submodular Cover with an approximation guarantee of $\max_{S\subseteq E} \left(\sum_{e\in X} z(S+e) - z(S)\right) / \left(z(E) - z(E \setminus S)\right)$, where X is a cover of $z' \colon 2^{E\setminus S} \to \mathbb{Z}_+$ with $z'(T) = z(S \cup T) - z(S)$. A different special case of Submodular Cover is the Knapsack Cover problem for which linear programming duality leads to an approximation factor of 2 as shown by Carnes and Shmoys [CS08] using strengthened formulations using flow cover inequalities by Carr, Fleischer, Leung, and Phillips [Car+99]. Note that the Knapsack Cover (while equivalent w.r.t. complexity to its packing variant) has an unbounded integrality gap, whereas Knapsack (Packing) has an integrality gap of 2 [Car+99]. We also refer the reader to the survey on Submodular Cover by Fujito [Fuj00] and the monograph by Fujishige [Fuj05].

Submodular functions and related optimization problems occur in several real-world applications, for instance, economics (as they have a diminishing returns property, which makes them natural valuation functions over sets of goods, see e.g., [MS09]) and machine learning (e.g., for multi-document summarization, as information coverage can be modeled using graph cuts, see e.g., the survey by Bilmes [Bil22]). While Sub-

MODULAR FUNCTION MINIMIZATION has a strongly polynomial algorithm (the most popular one is due to Iwata, Fujishige, and Fleischer [IFF01] but the newer ones in [Iwa02; Iwa03; IO09; Orl09; LSW15; Cha+17] have stronger running time guarantees), SUBMODULAR FUNCTION MAXIMIZATION is hard (for an overview, we refer to [KG14]). The SUBMODULAR COVER problem can be formulated as a SUBMODULAR FUNCTION MAXIMIZATION problem as we maximize coverage subject to minimum cost.

5.2 Background Theory

Exchanges in Matroids Recall from Section 1.3 that a matroid $M = (E, \mathcal{I})$ is completely determined by its rank function $\rho\colon 2^E \to \mathbb{Z}_+$ with $\rho(S) = \max\{|I| : I \in \mathcal{I}, I \subseteq S\}$, which is monotone and submodular and for all $S \subseteq E$ it holds that $0 \leq \rho(S) \leq |S|$ (cf. Proposition 1.12). By the base axiom (B2) (cf. Proposition 1.10), it is always possible to make a one-to-one exchange between two bases. It is not hard to see that this also holds for arbitrary independent sets by just restricting to a truncated matroid. However, there is an intriguing stronger property for those exchanges which we can observe in an *exchange graph*.

Definition 5.3. Let $M = (E, \mathcal{I})$ be a matroid and $I \in \mathcal{I}$. The bipartite graph $D_M(I) = (E, A)$ with bipartition $(I, E \setminus I)$ and

$$A = \{\{e, f\} : e \in I, f \notin I, I - e + f \in \mathcal{I}\}$$

is called *exchange graph*.

Lemma 5.4 ([Sch03, Corollary 39.12a])**.** *Let $M = (E, \mathcal{I})$ be a matroid and $I, J \in \mathcal{I}$ with $|I| = |J|$. Then there exists a perfect matching in $D_M(I)$ between $I \setminus J$ and $J \setminus I$.*

Duality in Optimization Covering problems are prototypical examples of combinatorial optimization problems, in particular, linear ones. We can usually write them in form of an *integer program* that looks like

$$\begin{aligned} &\text{minimize} \sum_{e \in E} c(e) x_e \\ &\text{subject to } Ax \geq b \\ &\qquad x \in \{0,1\}^E \end{aligned} \tag{5.1}$$

where $c \in \mathbb{R}_+^E$, $A \in \mathbb{R}_+^{[k]\times E}$, and $b \in \mathbb{R}_+^{[k]}$ contain non-negative data. These kind of programs can be solved by using, for instance, branch-and-bound techniques on its *linear relaxation*

$$\begin{aligned} &\text{minimize} \sum_{e\in E} c(e)x_e \\ &\text{subject to } Ax \geq b \\ &x \geq \mathbf{0} \end{aligned} \tag{5.2}$$

which can be solved in polynomial time using the ellipsoid method [Kha80] or interior point methods [Kar84]. In practice, *linear programs* are also solved using the simplex method [Dan63] although there are (so far) no polynomial running time guarantees. There is a neat theory in linear optimization around *duality*. For every linear program there is a corresponding *dual program*. The one for (5.2) is

$$\begin{aligned} &\text{maximize} \sum_{j\in[k]} b(j)\lambda_j \\ &\text{subject to } A^T\lambda \leq c \\ &\lambda \geq \mathbf{0} \end{aligned} \tag{5.3}$$

which easily can be recognized as a linear relaxation of a *packing problem*.

The following theorem is fundamental. It was first proven by von Neumann to prove that two-player zero-sum games have a value, i.e., a minimum profit (or maximum loss) for either player that can be guaranteed (in expectation) by playing a minimax strategy [Neu28]. Duality theory was rigorously formalized in the 1950s by Dantzig [Dan63] and Gale, Kuhn, and Tucker [GKT51].

Strong Duality Theorem (5.5) ([Dan63; GKT51])**.** *Let P be a linear program and D its dual. Let x^* be an optimal solution to P and λ^* an optimal solution to D. Then, $c(x^*) = b(\lambda^*)$.*

It is important to note that duality theory in matroids (say with a linear objective function) is not the same thing as duality in linear programming. Yet, we can make an obvious statement that has a similar flavor as the Strong Duality Theorem.

Proposition 5.6. *Let $M = (E, \mathcal{I})$ be a matroid and $w\colon E \to \mathbb{R}_+$ a weight function. If $B \in \mathcal{B}$ is a base of M with minimum weight $w(B)$, then $B^* = E \setminus B \in \mathcal{B}^*$ is a cobase of M with maximum weight $w(B^*)$.*

In both cases, one can observe that the dual solution acts as a kind of *witness* for optimality of the optimal primal solution.

5.3 Generalized Ranks

We denote by E the *ground set* of our covering problems. A *cover* of a set function $z\colon 2^E \to \mathbb{Z}_+$ is a set $S \subseteq E$ with $z(S) = z(E)$.[2] In particular, we are interested in covers that are optimal with respect to some *cost function* $c\colon E \to \mathbb{R}_+$, i.e., they minimize $c(S) = \sum_{e\in S} c(e)$ among all covers $S \subseteq E$.

As mentioned in the introduction, we want to generalize the *one-to-one exchange* property (base exchange property (B2)) of matroids to a *one-to-many exchange* property. We borrow some words from matroid terminology. Given a monotone submodular coverage function $z\colon 2^E \to \mathbb{Z}_+$ we say that

(1) $S \subseteq E$ is *z-independent* if $z(S - e) < z(S)$ for all $e \in S$.
(2) $S \subseteq E$ is *z-dependent* if it is not z-independent.
(3) $S \subseteq E$ is *z-spanning* if $z(S) = z(E)$ (i.e., if S is a cover).
(4) For $S \subseteq E$, its *z-span* corresponds to the set $\langle S \rangle_z := \{e \in E \mid z(S + e) = z(S)\}$.
(5) $S \subseteq E$ is a *z-base* if it is z-independent and z-spanning.

The following definition captures the main object we study in this chapter.

Definition 5.7. A function $z\colon 2^E \to \mathbb{Z}_+$ is a generalized rank if

(1) z is normalized, monotone, and submodular, and
(2) for every z-independent set $S \subseteq E$, we have $z(S) = \sum_{e\in S} z(e)$.

For ease of notation, we may write $z^+(S) := \sum_{e\in S} z(e)$ for any $S \subseteq E$.

Note that the rank function of a matroid satisfies the conditions in Definition 5.7. We will use two running examples of generalized ranks which are the coverage functions of Laminar Set Cover problems and Power-of-Two Knapsack Cover problems introduced below.

5.3.1 Examples

Weighted Matroid Ranks Given a matroid $M = (E, \mathcal{I})$, we immediately have that its rank function ρ is a generalized rank. However, we can also consider weighted matroid

[2] Usually for covering problems one would aim to find a set S with $z(S) \geq z(E)$. However, since we restrict ourselves to monotone coverage functions, the definition of a cover with an equality constraint is equivalent.

ranks, i.e., given item weights $w: E \to \mathbb{Z}_+$, we define $z: 2^E \to \mathbb{Z}_+$ by $z(S) = \max\{w(I) : I \in \mathcal{I}, I \subseteq S\}$.

Proposition 5.8. *A weighted matroid rank is a generalized rank.*

Proof. Let $M = (E, \mathcal{I})$ be a matroid with rank function ρ and weights $w: E \to \mathbb{R}_+$. Let z be as described above and denote for some $S \subseteq E$ a maximum weight independent set within S by I_S. The fact that z is normalized, monotone, and submodular is well known. Let $S \subseteq E$ be z-independent. Then $z(S - e) < z(S)$ for all $e \in S$ which implies $S \in \mathcal{I}$ and hence, $z(S) = w(S)$. □

Set Cover on a Laminar Family Recall the Set Cover instance from Example 5.1 (Figure 5.1). What makes the instance somewhat easy is that the available sets are all *nested*, i.e., if there are multiple options to cover the same points of the universe, it is always feasible to select exactly one set to cover a point. In other words, we never have to use two sets that overlap anywhere. This motivates the following definition.

Definition 5.9. Let $\mathcal{U}$ be a universe of points. A family of subsets $E \subseteq 2^{\mathcal{U}}$ is *laminar* if for all $e, f \in E$ with $e \cap f \neq \emptyset$ we have $e \subseteq f$ or $f \subseteq e$.

Laminar Set Cover

Given:	Laminar family $E = \{e_1, \ldots, e_n\}$ of subsets of a universe $\mathcal{U}$, costs $c: E \to \mathbb{R}_+$.
Find:	Subset $S \subseteq E$ such that $\bigcup_{e \in S} e = \mathcal{U}$ with $c(S)$ minimum.

We can define a coverage function (not just for Laminar Set Cover but also the general Set Cover) as described in the introduction by $z(S) = |\bigcup_{e \in S} e|$.

Proposition 5.10. *The coverage function of a* Laminar Set Cover *instance is a generalized rank.*

Proof. Normalization, monotonicity and submodularity are clear even in the case for non-laminar families. By submodularity, we immediately have for any set $z(S) \leq \sum_{e \in S} z(e)$, in particular for z-independent ones. Suppose there is a z-independent set S with $z(S) < \sum_{e \in S} z(e)$. Then there must exist $e, f \in S$ with $e \cap f \neq \emptyset$. By linearity of E, we have without loss of generality $e \subseteq f$. But then $z(S - e) = z(S)$, a contradiction to the assumption that S was z-independent. □

Knapsack Cover with only Powers of Two The Knapsack problem is one of the prototypical NP-complete problems. Given a set of items with weights and profits and a knapsack with some capacity, one is tasked to find a subset of items that does not exceed the weight capacity but maximize the total profit. This is conceptually a packing problem but in this chapter we consider covering problems. As sketched in Section 5.2, packing and covering problems are sort of dual to each other. We can define the Knapsack Cover problem as follows:

Knapsack Cover

Given: Items $E = \{e_1, \dots, e_n\}$ with weights $w(e)$, costs $c: E \to \mathbb{R}_+$, and a target weight W.

Find: Subset $S \subseteq E$ such that the total weight $w(S) \geq W$ with $c(S)$ minimum.

As its packing counter part Knapsack Cover, it is also an NP-hard optimization problem which however, can be formulated as a Submodular Cover problem. However, by restricting the item weights and target weights to powers of 2, we obtain a generalized rank as a coverage function.

Power-of-Two Knapsack Cover

Given: Items $E = \{e_1, \dots, e_n\}$ with weights $w(e)$ that are powers of 2 and a target weight W, i.e., $w(e), W \in \{2^k : k \in \mathbb{Z}_+\}$ for all $e \in E$, costs $c: E \to \mathbb{R}_+$.

Find: Subset $S \subseteq E$ such that the total weight $w(S) \geq W$ with $c(S)$ minimum.

Given an instance to Power-of-Two Knapsack Cover, we set the coverage $z: 2^E \to \mathbb{Z}_+$ to $z(S) = \min\{w(S), W\}$. Before we show that this indeed yields a generalized rank, we prove the following lemma.

Lemma 5.11. *For any $W = 2^m$ and $S \subseteq E$ with $w(S) \geq W$ such that $w(e) \leq W$ for all $e \in S$, we have some $T \subseteq S$ with $w(T) = W$.*

Proof. We show this by induction on m:

The claim is clearly true for $W = 2^0 = 1$ by the assumption that $W \geq w(e)$ for all $e \in E$.

Suppose $z(e) = w(e) \leq \frac{W}{2}$ for all $e \in S$. Then by induction hypothesis we must, there exists $T_1 \subseteq S$ such that $w(T_1) = \frac{W}{2}$. Moreover, $w(S \setminus T_1) \geq W - \frac{W}{2} = \frac{W}{2}$. Applying the induction hypothesis again, we must also have $T_2 \subseteq S \setminus T_1$ with $w(T_2) = \frac{W}{2}$. Then $T = T_1 \cup T_2$ is the desired set.

Otherwise, if there exists some f with $w(f) > \frac{W}{2}$, then we must have by our assumption that $W \geq w(e)$ for all $e \in E$ that actually $w(f) = W$ and hence, we can simply choose $T = \{f\}$. □

Proposition 5.12. *The coverage function for* Power-of-Two Knapsack Cover *is a generalized rank.*

Proof. Normalization and monotonicity are clear. Submodularity holds as z is the lower envelope of a modular function and a constant. Condition (2) follows now by observing that every $S \subseteq E$ with $w(S) < W$ is trivially z-independent and for all $w(S) \geq W$, we can always find a subset $T \subseteq S$ with $w(T) = W = z(T)$ by Lemma 5.11. □

Power-of-Two Truncated Generalized Ranks We can generalize the last example by setting a capacity restriction to an arbitrary generalized rank.

Let z be a generalized rank such that $z(e) \in \{2^k : k \in \mathbb{Z}_+\}$ for all $e \in E$. Let also $W \in \{2^k : k \in \mathbb{Z}_+\}$. Then we can consider the *truncation* $z' : 2^E \to \mathbb{Z}_+$ given by $z'(S) = \min\{z(S), W\}$. In this case, we say that the function z' is a *Power-of-Two Truncated Generalized Rank*.

Proposition 5.13. *Any Power-of-Two Truncated Generalized Rank is a generalized rank.*

Proof. Let $z'(S) = \min\{z(S), W\}$ be a Power-of-Two Truncated Generalized Rank. It is clear that z' is monotone and normalized. Also, z' is submodular as it is the minimum of a submodular function and a constant.

The fact that it is a generalized rank follows from Lemma 5.11 that we already used for Proposition 5.12. Indeed, assume that S is z'-independent. Then S is also z-independent. Hence, $z(S) = \sum_{e \in S} z(e)$, and thus, $z'(S) = \min\{z^+(S), W\}$. As the z values are powers of two, if $z'(S) = W$ there exists a subset $T \subseteq S$ such that $z(T) = W$. This implies that $T = S$, as otherwise $z'(S - e) = z'(S)$ for any $e \in S \setminus T$. Hence, $z'(S) = z^+(S)$. Similarly, if $z'(S) = z(S) = z^+(S)$. Hence, z' is a generalized rank. □

5.3.2 Basic Properties

In what follows, we introduce basic properties of the structure of z-independent sets and generalized ranks. In the following proposition, we show that under submodularity, the family of z-independent sets forms an independence system.

Lemma 5.14. *Let $z: 2^E \to \mathbb{Z}_+$ be a submodular and monotone function. Then, if T is z-independent, any $S \subseteq T$ is also z-independent.*

Proof. For any $e \in S$, submodularity implies $z(T) - z(T - e) \leq z(S) - z(S - e)$. By monotonicity, $z(T - e) < z(T)$ which implies then $z(S - e) < z(S)$, i.e., S is z-independent. □

Lemma 5.15. *Let $z: 2^E \to \mathbb{Z}_+$ be a submodular and monotone function. Let $S, T \subseteq E$ be two sets such that $z(T) > z(S)$. Then, there exists $e \in T \setminus S$ such that $z(S + e) > z(S)$.*

Proof. Monotonicity implies that $z(S \cup T) \geq z(T) > z(S)$. Since z is a submodular function, we have $z(S \cup T) - z(S) \leq \sum_{e \in T \setminus S} (z(S + e) - z(S))$. Therefore, we have

$$\sum_{e \in T \setminus S} (z(S + e) - z(S)) \geq z(S \cup T) - z(S) > 0.$$

As $z(S + e) - z(S) \geq 0$ for all $e \in T \setminus S$, there must exist one $e \in T \setminus S$ such that $z(S + e) - z(S) > 0$. □

Lemma 5.16. *Let $z: 2^E \to \mathbb{Z}_+$ be a generalized rank and $S \subseteq E$. For any $e \in E \setminus S$, we have*

$$z(S + e) - z(S) = z(e) - z(R)$$

for some z-independent set $R \subseteq S + e$. Moreover, if S is z-independent and $z(S + e) > z(S)$, then R can be taken as any inclusion-wise maximal set such that $z((S + e) \setminus R) = z(S + e)$.

Proof. First let us assume that S is z-independent. If $z(S + e) - z(S) = 0$ the result holds by taking $R = \{e\}$. So let us assume that $z(S + e) - z(S) > 0$. Let R be any maximal set such that $z((S + e) \setminus R) = z(S + e)$. Hence, $(S + e) \setminus R$ is z-independent by definition. Notice that $e \notin R$ as otherwise $z(S + e) = z(S)$. Then we have that

$$z(S + e) = \sum_{f \in S \setminus R} z(f) + z(e).$$

Thus, as S is z-independent,

$$z(S + e) - z(S) = \sum_{f \in S \setminus R} z(f) + z(e) - \sum_{f \in S} z(f) = z(e) - \sum_{f \in R} z(f).$$

As S is z-independent and $R \subseteq S$, we have that R is also z-independent and hence, $z(R) = \sum_{f \in R} z(f)$. This implies the case where S is z-independent and the second part of the lemma.

Let us now show the lemma when S is not z-independent. Let $S' \subseteq S$ be a minimal set such that $z(S) = z(S')$. For the proof to follow it suffices to show that $z(S+e) = z(S'+e)$. Clearly $z(S+e) \geq z(S'+e)$ by the monotonicity of z. By submodularity it holds that $z(S+e) - z(S) \leq z(S'+e) - z(S')$. As $z(S) = z(S')$ we obtain that $z(S'+e) - z(S') = z(S+e) - z(S)$, and hence, we can apply the result for the z-independent set S'. □

Remark. In general, unlike matroid ranks, for a generalized rank z, an inclusion-wise maximal z-independent set is not necessarily a z-base. This can easily be seen by a Laminar Set Cover instance with $\mathcal{U} = \{1, 2\}$ and $E = \{\{1\}, \{1, 2\}\}$. Then $\{\{1\}\}$ is an inclusion-wise maximal z-independent set but not a z-base as it does not cover 2.

Remark. It is clear that any z-base is necessarily an inclusion-wise maximal z-independent set but it does not need to be cardinality-wise maximal z-independent set. The Laminar Set Cover instance with $\mathcal{U} = \{1, 2\}$ and $E = \{\{1\}, \{2\}, \{1, 2\}\}$ has two z-bases $\{\{1\}, \{2\}\}$ and $\{\{1, 2\}\}$ which have different cardinality.

Note that for Lemma 5.16 if z is the rank function of a matroid, then $|R| \leq 1$. On the other hand, for the coverage function of a Laminar Set Cover instance, the set R can have larger cardinality: take for example the laminar family $E = \{\{1, 2, 3\}, \{1\}, \{2\}, \{3\}\}$, then take $S = \{\{1\}, \{2\}\}$ and $e = \{1, 2, 3\}$, then $R = \{\{1\}, \{2\}\}$. Although for laminar set cover functions the maximal set R is unique, in general this is not the case. For example, take z to be the (unweighted) rank of a graphic matroid. Then adding e to a forest S might close a cycle and then R can be any singleton of an element on the cycle.

In the following lemma, we summarize the main properties of the span operator.

Lemma 5.17. *Let $z\colon 2^E \to \mathbb{Z}_+$ be a generalized rank. The following hold:*

(i) $\langle \cdot \rangle_z$ *is inflationary, i.e.,* $S \subseteq \langle S \rangle_z$ *for all* $S \subseteq E$.
(ii) $\langle \cdot \rangle_z$ *is monotone, i.e.,* $\langle S \rangle_z \subseteq \langle T \rangle_z$ *for all* $S \subseteq T \subseteq E$.
(iii) $\langle \cdot \rangle_z$ *is idempotent, i.e.,* $\left\langle \langle S \rangle_z \right\rangle_z = \langle S \rangle_z$ *for all* $S \subseteq E$.
(iv) $z(S) = z(\langle S \rangle_z)$ *for all* $S \subseteq E$.
(v) If $\langle S \rangle_z \subseteq \langle T \rangle_z$ *then for any* $A \subseteq E$, $\langle S \cup A \rangle_z \subseteq \langle T \cup A \rangle_z$.
(vi) Let $S \subseteq E$, $A \subseteq \langle S \rangle_z$, *and* $T \subseteq E$ *with* $z(S \cup T) = z(S) + z(T)$. *Then* $z(A \cup T) = z(A) + z(T)$.

Proof. For *(i)*, observe that for any $e \in S$ we have $z(S + e) = z(S)$ and hence, $e \in \langle S \rangle_z$.

For *(ii)*, let $e \in \langle S \rangle_z$, i.e., $z(S + e) = z(S)$. By submodularity, for any $T \supseteq S$ that $z(T + e) - z(T) \leq z(S + e) - z(S) = 0$ and by monotonicity it follows that $z(T + e) - z(T) = 0$ and hence, $e \in \langle T \rangle_z$.

For *(iii)*, the inclusion $\supseteq$ follows directly from *(i)* and *(ii)* as $S \subseteq \langle S \rangle_z$. For the other inclusion, let $e \in \langle \langle S \rangle_z \rangle_z$. Then $z(\langle S \rangle_z + e) = z(\langle S \rangle_z) = z(S)$, where the second equality follows from submodularity of z. Now suppose $z(S + e) > z(S)$, then by monotonicity of z, $z(\langle S \rangle_z + e) \geq z(S + e) > z(S)$ in contradiction to $z(\langle S \rangle_z + e) = z(S)$.

For *(iv)*, we have $z(S) = z(S + e)$ for all $e \in \langle S \rangle_z$ by definition. Then by submodularity, $z(\langle S \rangle_z) = z(S \cup (\langle S \rangle_z \setminus S)) \leq z(S) + \sum_{e \in \langle S \rangle_z \setminus S} z(S + e) = z(S)$. By monotonicity, equality must hold throughout.

For *(v)*, we notice that $\langle S \cup A \rangle_z = \langle \langle S \rangle_z \cup A \rangle_z$. Indeed, the inclusion $\subseteq$ follows from monotonicity and the fact that $S \subseteq \langle S \rangle_z$. For the other inclusion, we notice that

$$\langle S \rangle_z \cup A \subseteq \langle S \cup A \rangle_z ,$$

and property *(ii)* implies

$$\langle \langle S \rangle_z \cup A \rangle_z \subseteq \langle S \cup A \rangle_z .$$

We conclude analogously that $\langle T \cup A \rangle_z = \langle \langle T \rangle_z \cup A \rangle_z$. Thus,

$$\begin{aligned} \langle S \cup A \rangle_z &= \langle \langle S \rangle_z \cup A \rangle_z \\ &\subseteq \langle \langle T \rangle_z \cup A \rangle_z \\ &= \langle \langle T \rangle_z \cup A \rangle_z \\ &= \langle T \cup A \rangle_z . \end{aligned}$$

Finally, for *(vi)*, since $A \subseteq \langle S \rangle_z$, we have by *(iv)* and monotonicity of z that $z(S) = z(S \cup A) \geq z(A)$. Note that by the assumption that $z(S \cup T) = z(S) + z(T)$ we have $T \cap \langle S \rangle_z = \varnothing$. Hence,

$$\begin{aligned} z(S) + z(T) &= z(S \cup A) + z(T) \\ &\geq z(S \cup A \cup T) + z(\varnothing) \\ &= z(S \cup A \cup T) \\ &\geq z(S \cup T) \\ &= z(S) + z(T) \end{aligned}$$

where the first inequality follows from submodularity and the second from monotonicity. In total, we must have equality throughout the whole chain which implies $z(S \cup A \cup T) = z(S \cup A) + z(T)$. Again using submodularity twice, we have

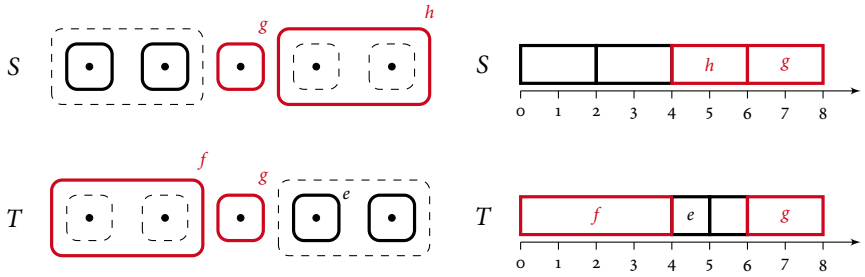
$$z(T) = z(S \cup A \cup T) - z(S \cup A) \leq z(A \cup T) - z(A) \leq z(T)$$

which again enforces equality throughout and hence, $z(A \cup T) = z(A) + z(T)$ as desired. □

Remark. Note that the requirement for z to be a generalized rank is not necessary for items *(i)–(v)* of Lemma 5.17. It suffices that z is normalized, monotone, and submodular.

5.4 One-to-Many Exchange Properties for z-Bases

Arguably, the base exchange property of matroids ((B2) in Proposition 1.10) is the key feature that makes its theory so rich. Simply put, the base exchange property allows one to move from a non-optimal base to an optimal base via a finite sequence of *one-to-one exchanges* and each of them is a strictly improving step. This is a property that other structures, in particular independence systems (without the exchange property) are lacking. In some sense, our goal is to find a structure that allows us to go from a non-optimal base to an optimal base via improving *one-to-many exchanges* and give an optimal algorithm for this structure. While matroids are most often defined using set systems $(E, \mathcal{I})$, $(E, \mathcal{B})$, $(E, \mathcal{C})$ etc. (independent sets, bases, circuits, …), they can also be defined via a *rank function* (E, ρ), which for us is the most convenient starting point for the generalization.



(a) Two z-bases, S and T, of a Laminar Set Cover instance. The bold sets are selected, the dashed ones are not.

(b) Two z-bases, S and T, of a Power-of-Two Knapsack Cover instance.

Figure 5.2: Examples of (S, T)- and (T, S)-exchangeable and non-exchangeable elements. In both cases, for element $e \in T \setminus S$ there is no $R \subseteq S \setminus T$ such that $(S \setminus R) + e$ is a z-base. The red elements in $(S \setminus T) \cup (T \setminus S)$, i.e., f and h, are either (S, T)- or (T, S)-exchangeable.

We already presented Lemma 5.16 which gives already an idea how a one-to-many exchange is feasible for generalized ranks. However, we clearly cannot expect that there is a direct counterpart of every property that can be derived from the base exchange property of matroids. Intuitively, this is due to the lack of symmetry in the exchanges, which (already in its name) is present in one-to-one exchanges. For example, given two z-bases $S, T \subseteq E$ and some $e \in T \setminus S$, there is not necessarily a set $R \subseteq S \setminus T$ such that $(S + e) \setminus R$ is again a z-base. We give an example of this in Lemma 5.2 for Laminar Set Cover and Power-of-Two Knapsack Cover.

Figuratively speaking, an exchange of $e \in T \setminus S$ and $R \subseteq S \setminus T$ is only possible if $z(e)$ is large enough such that it covers elements in T. This prompts the following definition.

Definition 5.18. Let $z\colon 2^E \to \mathbb{Z}_+$ be a generalized rank and S, T be two z-bases. We say that $e \in T \setminus S$ is (S, T)-*exchangeable* if there exists $R \subseteq S \setminus T$ such that $(S + e) \setminus R$ is a z-base.

This section is devoted to identifying a large collection of (S, T)-exchangeable and (T, S)-exchangeable elements. As we will see, these elements are deeply connected to the greedy algorithm below.

Given some z-spanning set $S \subseteq E$, we can always construct a z-base, i.e., some set $B \subseteq S$ with $z(B) = z(E)$ by using the algorithm Greedy z-Base, which uses a total order $\succeq$ of the elements $e_1 \succeq \ldots \succeq e_m$ with the property that $z(e_j) \geq z(e_{j+1})$ for all $j \in [m-1]$. We call an order with this property *weight-respecting*.

Greedy z-Base:

Input: Generalized rank $z\colon 2^E \to \mathbb{Z}_+$, a subset $S \subseteq E$, and weight-respecting order $\geq$

Output: A z-base

```
B ← ∅
for e ∈ S in order of ≥ do
    if z(B + e) > z(B) then
        B ← B + e
return B
```

We refer to the z-base B that we obtain from Greedy z-Base as the *greedy z-base relative to* $(S, \geq)$. For the analysis, we enumerate the iterations of Greedy z-Base by $\tau \in \{0, \dots, m\}$, where $B_0 = \emptyset$ and $B_{\tau+1} = B_\tau + e_{\tau+1}$ if $z(B_\tau + e_\tau) > z(B_\tau)$ and otherwise $B_{\tau+1} = B_\tau$. Then $B_m = B$ is the output of the algorithm.

Lemma 5.19. *Let $z\colon 2^E \to \mathbb{Z}_+$ be a generalized rank. The set B returned by Greedy z-Base is indeed a z-base.*

Proof. We need to show that B is z-spanning and z-independent.

Assume towards a contradiction that B is not z-spanning, that is, $z(B) < z(S) = z(E)$. Then, by Lemma 5.15, there exists $e_\tau \in S \setminus B$ such that $z(B+e_\tau) > z(B)$. By submodularity of z, we then have that $z(B_{\tau-1} + e_\tau) > z(B_{\tau-1})$. Hence, Greedy z-Base would have selected e_τ, a contradiction. Thus, B is z-spanning.

To show that B is z-independent we proceed by induction. Clearly, $B_0 = \emptyset$ is z-independent. Consider the set B_τ before entering iteration $\tau + 1$ of the **for** loop. Assume by induction that B_τ is z-independent and that $z(B_\tau + e_{\tau+1}) > z(B_\tau)$, so $e_{\tau+1}$ is added. (The other case where $e_{\tau+1}$ is not added is trivial as we keep the set B_τ in that case, which by induction hypothesis is z-independent.) Notice that $z(e_{\tau+1}) \leq z(e)$ for all $e \in B_\tau$ by our assumption on the ordering of S. As $z(B_\tau + e_{\tau+1}) > z(B)$, by Lemma 5.16 there exists a z-independent set $R \subseteq B_\tau + e_{\tau+1}$ such that $z(B_\tau + e_{\tau+1}) - z(B_\tau) = z(e_{\tau+1}) - z(R) > 0$. Since $z(e_{\tau+1}) \leq z(e)$ for all $e \in R$, we must have that $R = \emptyset$. Thus, $z(B_\tau + e_{\tau+1}) = z(B_\tau) + z(e_{\tau+1}) = \sum_{e \in B_\tau} z(e) + z(e_{\tau+1})$, which means that $B_\tau + e_{\tau+1}$ is z-independent. □

The next theorem uses the last lemma to show that a z-base B within $S \cup T$ exists such that all elements in $B \cap (T \Delta S)$ are either (S, T)- or (T, S)-exchangeable; see Figure 5.2 for examples, where the set B is depicted in red.

Theorem 5.20. *Let $z\colon 2^E \to \mathbb{Z}_+$ be a generalized rank and S and T be two z-bases. There exists a z-base $B \subseteq S \cup T$ satisfying the following. Let $e \in B \cap (S \setminus T)$ (respectively $e \in B \cap (T \setminus S)$). Then there exists a set $R \subseteq T \setminus B$ (respectively $R \subseteq S \setminus B$) such that $(T + e) \setminus R$ (respectively $(S + e) \setminus R$) is a z-base.*

Proof. Fix any weight-respecting ordering on $S \cup T = \{e_1, \ldots, e_m\}$, i.e., an order $(S \cup T, \geq)$ with $e \geq f$ only if $z(e) \geq z(f)$ for all $e, f \in S \cup T$. Let B be the corresponding greedy z-base. Take $e \in B$ and without loss of generality assume that $e \in S$. Consider the greedy z-base relative to $T + e$ with the same ordering used for $S \cup T = \{e_1, \ldots, e_m\}$ but restricted to $T + e$ (which is z-spanning). Call this new z-base B'.

The main idea of the proof is to show that $B \cap (T + e) \subseteq B' \subseteq T + e$. This implies, in particular, that $e \in B'$, and hence, the z-base B' equals $(T + e) \setminus R$ for some $R \subseteq T \setminus B$, which implies the theorem. Observe that showing that $B \cap (T + e) \subseteq B'$ is equivalent to the following: when running Greedy z-Base on $T + e$, the algorithm selects all elements from B that belong to $T + e$. In the following, we assume that $e = e_\tau$ for some τ.

For any $\tau \in \{1, \ldots, m\}$ and set $A \subseteq S \cup T$, let us call $A_{<\tau} := A \cap \{e_1, \ldots, e_{\tau-1}\}$, the set of elements in A considered before e_τ in the greedy algorithm.

Claim 1. *For each $\tau \in \{1, \ldots, m\}$ it holds that $\langle (S \cup T)_{<\tau} \rangle_z = \langle B_{<\tau} \rangle_z$. Analogously, for each $\tau \in \{1, \ldots, m\}$ it holds that $\langle (T + e)_{<\tau} \rangle_z = \langle B'_{<\tau} \rangle_z$.*

Proof. The two statements in the claim are analogous, so it suffices to show the first one. We first show that $\langle (S \cup T)_{<\tau} \rangle_z \subseteq \langle B_{<\tau} \rangle_z$. Let z' be the restriction of z to elements in $\{e_1, \ldots, e_{\tau-1}\}$. Applying Lemma 5.19 to z', as we have that the set $\{e_1, \ldots, e_{\tau-1}\}$ is z'-spanning by definition, we obtain that the partial solution given by the algorithm, $B_{<\tau}$, is a z'-base. In other words, for all $e \in (S \cup T)_{<\tau}$ we have that $z(B_{<\tau} + e) = z(B_{<\tau})$. Thus, $(S \cup T)_{<\tau} \subseteq \langle B_{<\tau} \rangle_z$. By Lemma 5.17.*(ii)* and Lemma 5.17.*(iii)*, we obtain that $\langle (S \cup T)_{<\tau} \rangle_z \subseteq \langle B_{<\tau} \rangle_z$. Finally, as $B_{<\tau} \subseteq (S \cup T)_{<\tau}$, we also get by Lemma 5.17.*(ii)* that $\langle B_{<\tau} \rangle_z \subseteq \langle (S \cup T)_{<\tau} \rangle_z$. The claim follows. ∎

Claim 2. *For each $\tau \in \{1, \ldots, m\}$ it holds that $\langle B'_{<\tau} \rangle_z \subseteq \langle B_{<\tau} \rangle_z$.*

Proof. This follows quite directly from Claim 1, as $(T+e)_{<\tau} \subseteq (S \cup T)_{<\tau}$ and thus, $\langle (T+e)_{<\tau} \rangle_z \subseteq \langle (S \cup T)_{<\tau} \rangle_z$ by the monotonicity of the $\langle \cdot \rangle_z$ function. Therefore, the claim holds as $\langle B'_{<\tau} \rangle_z = \langle (T+e)_{<\tau} \rangle_z \subseteq \langle (S \cup T_{<\tau}) \rangle_z = \langle B_{<\tau} \rangle_z$. ∎

Claim 3. *All elements in $B \cap (T+e)$ belong to B'.*

Proof. Take any element $e_\tau \in B \cap (T+e)$. To show that e_τ belongs to B' is suffices to notice that $e_\tau \notin \langle B'_{<\tau} \rangle_z$. But as $e_\tau \in B$, we have that $e_\tau \notin \langle B_{<\tau} \rangle_z$. Claim 2 implies then that $e_\tau \notin \langle B'_{<\tau} \rangle_z$. ∎

With the last claim we conclude that $e = e_\tau \in B \cap (T+e) \subseteq B' \subseteq T+e$. It then follows that $B' = (T+e) \setminus R$ for some $R \subseteq T \setminus B$. The theorem follows as B' is a z-base by Lemma 5.19. □

It is natural to ask whether generalized ranks also have a *many-to-one exchange* property (instead of one-to-many). That is, given two z-spanning sets S, T and a z-base $B \subseteq S \cup T$, can we find for each $e \in B \cap (S \setminus T)$ some $R \subseteq T \setminus B$ such that $(S \cup R) - e$ is a z-base. The following example shows that this is not the case.

Example 5.2. Consider a ground set $E = \{e_0, e_1, e_2, e_3, e_4, f_0, f_1, f_2\}$, the 4-uniform matroid M_1 over the ground set $\{e_1, e_2, e_3, e_4, f_1, f_2\}$, and the free matroid (i.e., all subsets are independent) M_2 over the ground set $\{e_0, f_0\}$. We consider the matroid given by the direct sum of M_1 and M_2, i.e., $M = M_1 \oplus M_2 = (E, \mathcal{I})$. Also, define $w(e_i) = w(f_j) = 1$ for $i, j \geq 1$, $w(e_0) = 2$ and $w(f_0) = 4$. We consider the generalized rank

$$z(S) = \max \left\{ \sum_{e \in J} w(e) : J \subseteq S, J \in \mathcal{I}, w(J) \leq 6 \right\}.$$

The proof that this is a generalized rank follows along the same lines as the proof of Proposition 5.13.

Consider the z-base $S = \{f_0, f_1, f_2\}$ and $T = \{e_0, \ldots, e_4\}$. Observe that the unique greedy z-base computed by Greedy z-Base in $S \cup T$ is $B = \{e_0, f_0\}$. However, $T - e_0$ cannot be extended to a z-base (without removing elements).

5.5 The Greedy+ Algorithm

In this section, we propose an algorithm that solves certain Submodular Cover problems optimally. As a motivation, recall that the default greedy algorithm for Sub-

MODULAR COVER as described in Section 5.1 fails to find an optimal solution on a very simple SET COVER instance (namely, a LAMINAR SET COVER instance, see Example 5.1). The key observation from this example is that the optimal solution—at any point—could have been salvaged if the algorithm did not choose the set with the best cost-to-coverage ratio but instead also accounted for saved cost, i.e., the cost of sets that could have been removed from the solution as they become redundant.

We use the notation $R(S, e) := \arg\max\{c(R) : z((S \setminus R) + e) = z(S + e), R \subseteq S\}$. Later we show that for some special cases $R(S, e)$ can also be computed efficiently.

Greedy+:

Input: A monotone submodular function $z\colon 2^E \to \mathbb{Z}_+$ and a linear cost function $c\colon E \to \mathbb{R}_+$

Output: A z-spanning set

1 $S \leftarrow \varnothing$

2 **while** $z(S) < z(E)$ **do**

3 $\quad \bar{e} \leftarrow \arg\min \left\{ \frac{c(e)-c(R(S,e))}{z(S+e)-z(S)} : e \in E \text{ with } z(S+e) > z(S) \right\}$

4 $\quad S \leftarrow (S \setminus R(S, \bar{e})) + \bar{e}$

5 **return** S

As usual for algorithms that operate on set functions, we assume that the values $z(S)$ for each $S \subseteq E$ are given by an oracle in time $\mathcal{O}(1)$. If $R(S, e)$ can be computed efficiently (i.e., in polynomial time), then finding the arg min in Line 3 requires just a linear search of elements $e \in E \setminus S$, which makes every iteration efficient.

In the sequel, we denote the iterations of Greedy+ by numbers $\tau \in \{1, 2, \ldots, r\}$ and the currently allocated sets, chosen elements, and set of removed elements by S_τ, e_τ, and R_τ, respectively. That is, for all $\tau \in \{1, 2, \ldots, r\}$, we have $S_\tau := (S_{\tau-1} \setminus R_\tau) + e_\tau$, where $S_0 := \varnothing$.

Remark. The subproblem to find the set $R(S, e)$ in each iteration is a SUBMODULAR COVER problem itself: Essentially, we need to find a cost-minimum set $T \subseteq S + e$, such that $z(T) = z(S + e)$. All those covering sets T yield a feasible removal set R via $R = (S + e) \setminus T$ and we have $R(S, e)$ constructed from a cover T of minimum cost. Note that by monotonicity of z and choice of e, we must have $e \in T$ and hence, $e \notin R(S, e)$.

First note that Lemma 5.15 implies that Greedy+ always terminates if the coverage function z is monotone and submodular as it always finds a suitable element to augment the current set.

Before we go into the analysis regarding the optimality of Greedy+ for Submodular Cover problems, we make a few observations about the algorithm that will come in handy.

The next lemma shows that if an element can be added to the current set such that it stays z-independent without removing any previously selected elements, then this element had the same property in all previous iterations already, i.e., the removal steps of Greedy+ were not relevant for that element.

Lemma 5.21. *Let $z\colon 2^E \to \mathbb{Z}_+$ be a generalized rank function. Then for every iteration τ of Greedy+ and all $e \in E \setminus S_\tau$, if $S_\tau + e$ is z-independent, then $S_{\tau-1} + e$ is z-independent.*

Proof. It holds that $z((S_{\tau-1} \setminus R_\tau) + e_\tau) = z(S_{\tau-1} + e_\tau)$. By submodularity and monotonicity of z it follows that $z(S_{\tau-1} + e_\tau + e) = z((S_{\tau-1} \setminus R_\tau) + e_\tau + e)$. Subtracting the first equality from the second yields

$$\begin{aligned} z(S_{\tau-1} + e_\tau + e) - z(S_\tau + e_\tau) &= z((S_{\tau-1} \setminus R_\tau) + e_\tau + e) - z((S_{\tau-1} \setminus R_\tau) + e_\tau) \\ &= z(S_\tau + e) - z(S_\tau) = z(e) \end{aligned}$$

where the second equality follows from the fact that $S_\tau + e$ is z-independent and z is a generalized rank. Again using submodularity, we have

$$\begin{aligned} z(e) &\geq z(S_{\tau-1} + e) - z(S_{\tau-1}) \\ &\geq z(S_{\tau-1} + e_\tau + e) - z(S_{\tau-1} + e_\tau) = z(e), \end{aligned}$$

hence, we have equality throughout and in particular $z(S_{\tau-1} + e) - z(S_{\tau-1}) = z(e)$. □

Also consider the contrapositive of Lemma 5.21. It implies that, given a generalized rank z, if $e \in R_\tau$ (and hence, $z(S_\tau + e) = z(S_\tau)$), then we must have $z(S_{\tau'} + e) = z(S_{\tau'})$ for all $\tau' > \tau$. Thus, once an element is removed from a partial solution of Greedy+, it never has to be considered again in later iterations as its marginal coverage will remain 0. That means, there are at most $|E|$ iterations of the **while** loop of Greedy+. Thus, we obtain the following corollary:

Corollary 5.22. *Let $z\colon 2^E \to \mathbb{Z}_+$ be a generalized rank and consider any* Submodular Cover *problem on z. Then the total running time of Greedy+ is in $\mathcal{O}(|E|^2 \mathsf{RO})$, where* RO *is the running time to determine the set $R(S, e)$ in each iteration.*

With structure, the running time can be better than that; the aforementioned bound assumes a naïve linear search of the arg min in Line 3. In Section 5.7, we give some implementation of the *removal oracle* for Laminar Set Cover and Power-of-Two Knapsack Cover and show that there running time RO is indeed polynomial in $|E|$. The following lemma states that whenever Greedy+ selects an element e_τ which does not prompt any removal of previously selected elements (i.e., $R_\tau = \varnothing$), then all $f \in S_{\tau-1}$ have a better cost-to-weight ratio than e_τ.

Lemma 5.23. *Let $z\colon 2^E \to \mathbb{Z}_+$ be a generalized rank function. Then for every iteration τ of Greedy+ in which $z(S_\tau) = z(S_{\tau-1} + e_\tau) = z(S_{\tau-1}) + z(e_\tau)$, it holds for all $f \in S_{\tau-1}$ that*

$$\frac{c(f)}{z(f)} \leq \frac{c(e_\tau)}{z(e_\tau)}.$$

Proof. This is trivially true for S_1. Assume as induction hypothesis that the statement holds for $S_\tau = (S_{\tau-1} \setminus R_\tau) + e_\tau$ and now consider $S_{\tau+1} = (S_\tau \setminus R_{\tau+1}) + e_{\tau+1}$. If $R_{\tau+1} \neq \varnothing$, then there is nothing to be shown, so assume $R_{\tau+1} = \varnothing$.

Case 1 ($f \in S_{\tau-1} \setminus R_\tau$): By Lemma 5.21, we know $z(S_{\tau-1} + e_{\tau+1}) = z(S_{\tau-1}) + z(e_{\tau+1})$. Then by the induction hypothesis we have $\frac{c(f)}{z(f)} \leq \frac{c(e_{\tau+1})}{z(e_{\tau+1})}$.

Case 2 ($f \notin S_{\tau-1} \setminus R_\tau$, i.e., $f = e_\tau$): Suppose $\frac{c(e_\tau)}{z(e_\tau)} > \frac{c(e_{\tau+1})}{z(e_{\tau+1})}$. Then we have

$$\frac{c(e_\tau) - c(R_\tau)}{z(e_\tau) - z(R_\tau)} = \frac{c(e_\tau) - c(R_\tau)}{z(S_{\tau-1} + e_\tau) - z(S_{\tau-1})} \leq \frac{c(e_{\tau+1})}{z(e_{\tau+1})} < \frac{c(e_\tau)}{z(e_\tau)},$$

where the equality holds as z is a generalized rank function and the first inequality follows from the Greedy+ selection rule and Lemma 5.21. Since $z(e_\tau) > z(R_\tau)$, we have that this is equivalent to:

$$\frac{c(R_\tau)}{z(R_\tau)} > \frac{c(e_\tau)}{z(e_\tau)}.$$

Then by the *mediant inequality*,[3] we then must have some $g \in R_\tau \subseteq S_{\tau-1}$ with

$$\frac{c(g)}{z(g)} > \frac{c(e_\tau)}{z(e_\tau)} > \frac{c(e_{\tau+1})}{z(e_{\tau+1})}$$

[3] For integers $p_1, \ldots, p_k, q_1, \ldots, q_k$, it holds that

$$\min_{i \in [k]} \frac{p_i}{q_i} \leq \frac{\sum_{i=1}^k p_i}{\sum_{i=1}^k q_i} \leq \max_{i \in [k]} \frac{p_i}{q_i}.$$

which contradicts the induction hypothesis as $e_{\tau+1}$ was feasible to add to $S_{\tau-1}$ by Lemma 5.21. □

In the following, we call a set $S \subseteq E$ *regular* w.r.t. z and c if the condition of Lemma 5.23 holds, i.e.:

$$\text{For all } e \in E \setminus S: \text{ If } S + e \text{ is } z\text{-independent, then } \frac{c(f)}{z(f)} \leq \frac{c(e)}{z(e)} \text{ for all } f \in S. \qquad (\spadesuit)$$

5.5.1 A Sufficient Optimality Condition

We want to describe a subclass of Submodular Cover problems on which Greedy+ is guaranteed to compute an optimal solution. To this end, we first define for a coverage function z and two z-independent sets an exchange hypergraph. The idea is that these hypergraphs give us a similar property as the bipartite exchange graphs of matroids (see Section 5.2).

Definition 5.24. Let $z: 2^E \to \mathbb{Z}_+$ be a monotone submodular function and $S, T \subseteq E$ two z-independent sets with $S \subseteq \langle T \rangle_z$. We call $G_{S,T} = (S \setminus T, T \setminus S, H)$ with

$$\begin{aligned} H = &\{(R, e) : R \subseteq S \setminus T, e \in T \setminus S, z(S + e) = z(S) - z(R) + z(e) > z(S)\} && (H_1)\\ \cup &\{(R, e) : R \subseteq S \setminus T, e \in T \setminus S, z(S + e) = z(S) + z(e) \text{ and } z(R) < z(e)\} && (H_2)\\ \cup &\{(R, Q) : R \subseteq S \setminus T, Q \subseteq T \setminus S, Q \subseteq \langle S \rangle_z, z((S \setminus R) \cup Q) = z(S)\} && (H_3) \end{aligned}$$

the *exchange hypergraph* of z with respect to S and T.

Example 5.3. Consider some covering problem (e.g., Power-of-Two Knapsack Cover) with $z(E) = 16$ and suppose we have a partial solution S (with $z(S) < z(E)$) and a solution T (with $z(T) = z(E)$). We depicted parts of the exchange hypergraph in Figure 5.3, where the first three elements in both sets are the same ($S \cap T$ in light red) and all others differ (S in shades of green, T in shades of blue). The exchange hypergraph $G_{S,T}$ only uses elements in the symmetric difference of S and T. Elements $e \in T \setminus S$ that can be added to S such that $S + e$ is z-independent have a hyperedge $(\varnothing, e)$, which is of type (H_1). Those elements e that make $S + e$ z-dependent have a set R to build hyperedge (R, e), where $R \subseteq S \setminus T$ is a minimal set such that $(S \setminus R) + e$ is z-independent (this R is not necessarily unique, in particular not for Power-of-Two Knapsack Cover).

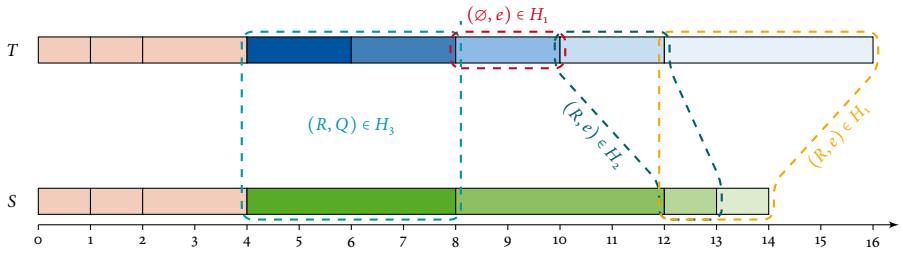


Figure 5.3: Example of an exchange hypergraph for a Power-of-Two Knapsack Cover instance. The red elements are in the intersection of S and T, while the blue elements are taken from $T \setminus S$ and the green ones from $S \setminus T$.

Elements e that can be added to S such that it remains z-independent can also contain hyperedges of type (H_2) if there are subsets $R \subseteq S \setminus T$ that can be removed while still increasing the overall coverage.

Finally, subsets $R \subseteq S \setminus T$ and $Q \subseteq T \setminus S$ that can just be swapped without changing the coverage build hyperedges of type (H_3).

Note that Figure 5.3 only shows four hyperedges of the corresponding exchange hypergraph for demonstration; the actual number of hyperedges here is much higher. We denote the hyperedges of type (H_1), (H_2), and (H_3) by H_1, H_2, and H_3, respectively.

Note that in the special case, where z is the rank function of a matroid, the associated exchange hypergraph is just a bipartite graph, which always admits a perfect matching (cf. e.g., [Sch03, Corollary 39.12a]). While Definition 5.24 is a bit obscure, we can use it to show that Greedy+ is guaranteed to compute an optimal solution to Submodular Cover if the underlying coverage function z has only exchange graphs that admit a perfect hypermatching for regular sets S. We now define a subclass of generalized rank functions.

Definition 5.25. A generalized rank $z\colon 2^E \to \mathbb{Z}_+$ is *nice* if for any cost function $c\colon E \to \mathbb{R}_+$ and any two z-independent sets S, T with $S \subseteq \langle T \rangle_z$ and S satisfying the regularity condition (♠) w.r.t. z and c, $G_{S,T}$ admits a perfect hypermatching.

In Section 5.5.2 we show that the coverage functions of Laminar Set Cover and Power-of-Two Knapsack Cover, and the rank function of a matroid are all nice.

The following lemma shows that whenever a z-independent set satisfying (♠) can be augmented using some new element such that the resulting set remains z-independent, then removing any other element is not a descent direction.

Lemma 5.26. *Let $z\colon 2^E \to \mathbb{Z}_+$ be a generalized rank and S be a z-independent set satisfying property (♠), and $e \in E \setminus S$ with $z(S+e) = z(S) + z(e)$. Then for all $R \subseteq S$ with $z(R) < z(e)$, it holds that $\frac{c(e)}{z(e)} \leq \frac{c(e)-c(R)}{z(e)-z(R)}$.*

Proof. Consider any $e \in E \setminus S$ with $z(S+e) = z(S) + z(e)$, and any $R \subseteq S$ with $z(R) < z(e)$. By property (♠), we know that $\frac{c(f)}{z(f)} \leq \frac{c(e)}{z(e)}$ for all $f \in R$. Let $R = \{f_1, \ldots, f_k\}$. Note that R is z-independent as S is z-independent. Then, by the mediant inequality,

$$\frac{c(e)}{z(e)} \geq \frac{z(f_1)}{z(R)} \cdot \frac{c(f_1)}{z(f_1)} + \ldots + \frac{z(f_k)}{z(R)} \cdot \frac{c(f_k)}{z(f_k)} = \frac{c(R)}{z(R)}.$$

Observe that $\frac{c(e)}{z(e)} \geq \frac{c(R)}{z(R)}$ is equivalent to

$$\frac{c(e)}{z(e)} \leq \frac{c(e)-c(R)}{z(e)-z(R)}.$$

□

We are now ready to prove some sufficient optimality conditions for Greedy+.

Theorem 5.27. *If $z\colon 2^E \to \mathbb{Z}_+$ is a generalized rank function that is nice, then Greedy+ returns an optimal solution to the* Submodular Cover *problem for every linear cost function $c\colon E \to \mathbb{R}_+$.*

Proof. We show that in each iteration τ of Greedy+, the current z-independent set S_τ is an optimal solution to

$$\min\{c(S) : S \subseteq E, S \text{ is a } z\text{-base on } \langle S_\tau \rangle_z\}. \tag{SC$_\tau$}$$

Towards a contradiction assume there is a nice generalized rank function $z\colon 2^E \to \mathbb{Z}_+$ and a cost function $c\colon E \to \mathbb{R}_+$, where there exists a first iteration τ for which the current solution $S_\tau = (S_{\tau-1} \setminus R_\tau) + e_\tau$ is not optimal for (SC$_\tau$). Let S^* be an optimal solution for (SC$_\tau$) and consider the exchange hypergraph $G_{S_{\tau-1},S^*} = (V, W, H)$, where $V = S^* \setminus S_{\tau-1}$, $W = S_{\tau-1} \setminus S^*$ and H as given by Definition 5.24. Observe that $S_{\tau-1} \subseteq \langle S_\tau \rangle_z = \langle S^* \rangle_z$. By Lemma 5.23, we have that any intermediary solution of Greedy+ satisfies (♠) w.r.t. any z and c and hence, per Definition 5.25, $G_{S_{\tau-1},S^*}$ admits a perfect hypermatching μ.

We partition μ into μ_1, μ_2, μ_3 by distinguishing hyperedges in μ according to the lines (H_1), (H_2), and (H_3) in Definition 5.24. We also partition V and W into V_1, V_2, V_3 and W_1, W_2, W_3 such that for all $e \in V_i \cup W_i$ it holds that e is covered by a hyperedge in μ_i for all $i \in [3]$.

Since we assume by the induction hypothesis that $S_{\tau-1}$ is optimal for $(\mathrm{SC}_{\tau-1})$, each exchange along μ_3-hyperedges cannot lower the cost, thus, $c(W_3) \le c(V_3)$. Note that $c(S_\tau) > c(S^*)$ is equivalent to $c(e_\tau) - c(R_\tau) > c(V) - c(W)$ and then we can derive

$$\begin{aligned} c(e_\tau) - c(R_\tau) &> c(V) - c(W) \\ &\ge c(V_1) + c(V_2) - c(W_1) - c(W_2). \end{aligned} \tag{$\heartsuit$}$$

For each hyperedge $(R, e) \in \mu_1$, we have

$$\begin{aligned} \frac{c(e_\tau) - c(R_\tau)}{z(S_{\tau-1} + e_\tau) - z(S_{\tau-1})} &\le \frac{c(e) - c(R)}{z(S_{\tau-1} + e) - z(S_{\tau-1})} \\ &= \frac{c(e) - c(R)}{z(e) - z(R)}, \end{aligned}$$

where the inequality follows by the Greedy+ selection rule and the equality holds as (R, e) is a μ_1-hyperedge (i.e., as defined in (H_1)). Suppose $V = \{e_1, \dots, e_\ell\}$ and $W = R_{e_1} \uplus \dots \uplus R_{e_\ell}$, where $(R_{e_i}, e_i) \in \mu_1$ are the hyperedges of type (H_1). Define $T^{(i)} = \{e_1, \dots, e_i\}$ for $i \in [\ell]$ and $T^{(0)} = \varnothing$. Using that V_1 and W_1 are both z-independent, we can use the mediant inequality to derive

$$\begin{aligned} \frac{c(e_\tau) - c(R_\tau)}{z(S_{\tau-1} + e_\tau) - z(S_{\tau-1})} &\le \min_{(R,e)\in\mu_1} \frac{c(e) - c(R)}{z(S_{\tau-1} + e) - z(S_{\tau-1})} \\ &\le \min\left\{ \frac{c(e_1) - c(R_{e_1})}{z(S_{\tau-1} \cup T^{(1)}) - z(S_{\tau-1} \cup T^{(0)})}, \dots, \right. \\ &\qquad \left. \frac{c(e_\ell) - c(R_{e_\ell})}{z(S_{\tau-1} \cup T^{(\ell)}) - z(S_{\tau-1} \cup T^{(\ell-1)})} \right\} \\ &\le \frac{c(V_1) - c(W_1)}{z(S_{\tau-1} \cup V_1) - z(S_{\tau-1})} \\ &= \frac{c(V_1) - c(W_1)}{z(V_1) - z(W_1)}, \end{aligned}$$

where the second inequality follows by submodularity of z and the equality by $z(S_{\tau-1} \cup V_1) - z(S_{\tau-1}) = z(S_{\tau-1}) + z(V_1) - z(W_1) - z(S_{\tau-1})$.

Now consider the hyperedges $(R, e) \in \mu_2$. For those, we have

$$\begin{aligned}\frac{c(e_\tau) - c(R_\tau)}{z(S_{\tau-1} + e_\tau) - z(S_{\tau-1})} &\leq \min_{(R,e)\in\mu_2} \frac{c(e)}{z(e)} \\ &\leq \min_{(R,e)\in\mu_2} \frac{c(e) - c(R)}{z(e) - z(R)} \\ &\leq \frac{c(V_2) - c(W_2)}{z(V_2) - z(W_2)},\end{aligned}$$

where the first inequality is due to the Greedy+ selection rule, the second one follows by Lemma 5.26 (and the fact that $S_{\tau-1}$ satisfies (♠)), and the last one again by the mediant inequality applied to the whole set μ_2.

Using the latter again for both μ_1- and μ_2-hyperedges, we obtain

$$\frac{c(e_\tau) - c(R_\tau)}{z(S_{\tau-1} + e_\tau) - z(S_{\tau-1})} \leq \frac{c(V_1) + c(V_2) - c(W_1) - c(W_2)}{z(V_1) + z(V_2) - z(W_1) - z(W_2)}.$$

Since

$$\begin{aligned}z(S_{\tau-1} + e_\tau) - z(S_{\tau-1}) &= z(S_{\tau-1} \cup V_1 \cup V_2) - z(S_{\tau-1}) \\ &= z(V_1) + z(V_2) - z(W_1) - z(W_2)\end{aligned}$$

this implies $c(e_\tau) - c(R_\tau) \leq c(V_1) + c(V_2) - c(W_1) - c(W_2)$ in contradiction to (♡). □

5.5.2 Nice Submodular Cover Problems

We show that the coverage functions of Laminar Set Cover and Power-of-Two Knapsack Cover are nice and therefore, by Theorem 5.27, Greedy+ is guaranteed to compute an optimal solution to these problems for any linear cost function.

Proposition 5.28. *The coverage function of a* Laminar Set Cover *instance is a nice generalized rank.*

Proof. Let $S, T \subseteq E$ be two z-independent sets with $S \subseteq \langle T \rangle_z$ and S satisfying (♠). For the sake of convenience, assume $S \cap T = \varnothing$; otherwise the sets in the intersection are omitted in the exchange hypergraph anyway. We can partition S into two sets $(S^<, S^\geq)$,

where $S^{<}$ of sets that are covered by some $e \in T$ and $S^{\geq}$ of sets that cover multiple items in T. Formally,

$$S^{<} = \{f \in S \mid f \subset e \text{ for some } e \in T\},$$
$$S^{\geq} = \left\{f \in S \mid f = \bigcup_{e \in Q} e \text{ for some } Q \subseteq T\right\}.$$

Note that the latter set indeed has an equality in its condition as otherwise $f \notin \langle T \rangle_z$. We can also find a similar partition $(T^{>}, T^{\leq})$ for T with

$$T^{\leq} = \{e \in T \mid e \subseteq f \text{ for some } f \in S\},$$
$$T^{>} = T \setminus T^{\leq}.$$

Further observe that due to z-independence of S and laminarity of E, we must have that $S^{<}$ can again be partitioned into $(S_e^{<})_{e \in T^{>}}$ such that $S_e^{<} = \{f_1, \ldots, f_{k_e}\}$ with $f_i \subset e$ for all $i \in [k_e]$ and some cardinality k_e. Then, we can find a perfect hypermatching in $G_{S,T}$ using only edges of type (H_1) and (H_3): We have one hyperedge $(S^{\geq}, T^{\leq}) \in H_3$. For the remaining sets, we can again split $T^{>}$ into two sets $(T_{>}^{>}, T_{=}^{>})$, where $T_{>}^{>}$ contains those $e \in T^{>}$ for which we have that the inclusion-wise maximal set $R \subseteq S$ with $e \supseteq \bigcup_{f \in R} f$ the inclusion is strict. In that case, $(R, e) \in H_1$ for this largest R (which may be empty). All remaining sets $e \in T_{=}^{>}$ are covered using hyperedges $(R, \{e\}) \in H_3$. This covers all $e \in T$ by enumeration. Note that all $f \in S$ are also covered as otherwise some $f \in S$ would either be excluded from $\langle T \rangle_z$ or, we would not have chosen a maximal set R above, when considering the hyperedges $(R, e) \in H_1$. □

Essentially, the proof just makes a tedious case distinction on how the sets in S and T can relate to each other; By the conditions of Definition 5.25, we must have that for all $f \in S$ covering some $e \in T$, we must have that T actually contains a complete tiling of f, otherwise $f \notin \langle T \rangle_z$. For the remaining sets $f \in S$, we must have either a complete tiling of some $e \in T$ (which we can exchange via (H_3)) or it is strictly covered by some $e \in T$, which yields hyperedges of type (H_1). This is also depicted in Figure 5.4.

Proposition 5.29. *The coverage function of a* Power-of-Two Knapsack Cover *instance is a nice generalized rank.*

Proof. Let $S, T \subseteq E$ be two z-independent sets with $S \subseteq \langle T \rangle_z$. If T is not z-spanning, then we must have $S \subset T$ and hence, if the inclusion is strict, we have some only

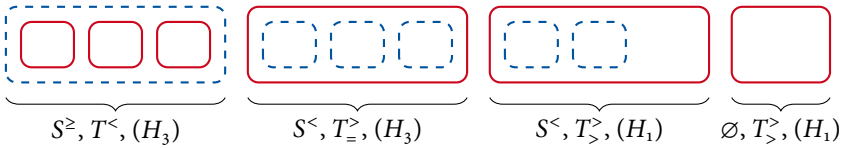


Figure 5.4: Illustration of the proof of Proposition 5.28. Sets $f \in S$ are blue and dashed, sets $e \in T$ are red and solid. Underneath the nested sets, one can find the corresponding hyperedges in the exchange hypergraph that are used in the perfect hypermatching.

hyperedges $(\varnothing, e) \in H_1$ to cover all $e \in T \setminus S$. If both S and T are z-bases, then $(S, T) \in H_3$ and that is the whole matching. So let us assume that only T is a z-base but $z(S) < W$. If there is some a set $Q \subseteq T \setminus S$ such that $z(S) + \sum_{e \in Q} w(e) = W$, then we have $(\varnothing, e) \in H_1$ for all $e \in Q$ as a matching hyperedge and $(S, T \setminus Q) \in H_3$ for the remaining elements. Otherwise choose any inclusion-wise minimal $Q \subseteq T$ with $z(S) + \sum_e w(e) > W$ and some $e_o \in Q$. By choice of Q, we have $z(S) + \sum_{e \in Q \setminus \{e_o\}} w(e) < W$; in particular $w(e_o) > 1$ (and Q does not contain any element with weight 1). Then by Lemma 5.11, there exists some $R \subseteq S$ such that $w((S \cup Q) \setminus R) = W$. Now we can construct a perfect hypermatching as follows:

$$\mu = \underbrace{\{(\varnothing, e) : e \in Q \setminus \{e_o\}\}}_{\subseteq H_1} \cup \{\underbrace{(R, e_o)}_{\in H_1}, \underbrace{((S \setminus T) \setminus R, (T \setminus S) \setminus Q)}_{\in H_3}\}. \qquad \square$$

5.6 Towards a Duality Theory for Generalized Ranks

In this section, our goal is to generalize the concept of duality in matroids to generalized ranks, which could potentially increase the number of tools to approach optimization problems involving these objects. As it will be discussed, it is not clear whether the natural definition of duality leads to a generalized rank function or not, but we provide sufficient conditions for that to happen in terms of the *dual of the dual* of the generalized rank function, revealing a rich and interesting structure along that line. Furthermore, we prove that in some relevant cases, the dual of a generalized rank function is indeed a generalized rank function. Prospectively, duality theory for generalized ranks could reveal more natural optimality conditions for Submodular Cover problems that are closer to know duality results from linear programming (Strong Duality Theorem) or matroid theory (Proposition 5.6).

5.6.1 The Dual of a Generalized Rank

We start by defining the dual of a generalized rank, which naturally generalizes the notion of duality from matroid theory. Recall that we write $z^+(S) = \sum_{e \in S} z(e)$ for any $S \subseteq E$.

Definition 5.30. Given a function $z\colon 2^E \to \mathbb{Z}_+$, we define its dual $z^*\colon 2^E \to \mathbb{Z}_+$ as

$$z^*(S) := \max\{z^+(J) : z(E \setminus J) = z(E), J \subseteq S\}.$$

We say that if z is a loopless generalized rank, that is, if $z(e) > 0$ for all $e \in E$, then S is z^*-independent if and only if $z(E \setminus S) = z(E)$, that is, S is z^*-independent if and only if $E \setminus S$ is z-spanning. Moreover, S is a z^*-base, if and only if $E \setminus S$ is a minimal z-spanning set, which is equivalent to saying that $E \setminus S$ is a z-base. In other words, the collection of z-bases is in one-to-one correspondence with the collection of z^*-bases by taking complements.

Observe that, if z is the rank function of a loopless matroid $M = (E, \mathcal{I})$, then z^* is the rank of the dual matroid M^*. Indeed, let $S \subseteq E$. Recall that $J \subseteq E$ is independent in M^* if and only if $E \setminus J$ contains a z-base, that is, if and only if $z(E \setminus J) = z(E)$. By definition of z^*, we have that

$$\begin{aligned} z^*(S) &= \max\{z^+(J) : z(E \setminus J) = z(E), J \subseteq S\} \\ &= \max\{z^+(J) : J \text{ is independent in } M^*, J \subseteq S\}. \end{aligned}$$

Moreover, as M is loopless, it holds that $z(e) = 1$ for all $e \in E$, and hence, $z^+(J) = |J|$ for any set J.

As mentioned before, it is not known whether the dual of a generalized rank function is a generalized rank function itself; however, as the following proposition shows, submodularity of the dual is the only missing link.

Proposition 5.31. *Assume that $z\colon 2^E \to \mathbb{Z}_+$ is a loopless generalized rank. Then, z^* is normalized, monotone, and satisfies that, for every z^*-independent set S, $z^*(S) = (z^*)^+(S)$.*

Proof. Normalization is trivial. For monotonicity, let $S \subseteq T \subseteq E$. If $z^*(S) = z^+(J)$ and $z(E \setminus J) = z(E)$ with $J \subseteq S \subseteq T$, then clearly J is feasible for the maximization problem defining $z^*(T)$, and hence, $z^*(T) \geq z^+(J) = z^*(S)$.

For the third property, let S be z^*-independent, that is, $z^*(S-e) < z^*(S)$ for all $e \in S$. Let $J \subseteq S$, with $z(E \setminus J) = z(E)$, be such that $z^*(S) = z^+(J)$. Then, notice that if $S \neq J$, we can take $e \in S \setminus J$ and obtain that $z^*(S-e) = z^+(J) = z^*(S)$, which yields a contradiction. Hence, $J = S$. Thus, $z^*(S) = z^+(S) = \sum_{e\in S} z(e)$. Finally, we observe that $z^*(e) = z(e)$ for all $e \in S$. Indeed, $z^*(e)$ can be either $z(e)$ or 0. However, as z is loopless $z(e) \neq 0$. Hence, $z(e) = z^*(e)$ for all e. □

5.6.2 Reflexivity

As we will see, the dual of the dual of a generalized rank function z, namely $z^{**} := (z^*)^*$, plays a fundamental role as sufficient condition for z^* to be a generalized rank function. We will start by providing a simple characterization of z^{**}-independent sets. The following simple proposition will be helpful for our purposes.

Proposition 5.32. *Let $z\colon 2^E \to \mathbb{Z}_+$ be a generalized rank and $S \subseteq E$ be any set. Let R be an inclusion-wise maximal subset of S such that $z(S \setminus R) = z(S)$. Then $\sum_{e\in R} z(e) = \sum_{e\in S} z(e) - z(S)$, in particular $\sum_{e\in R} z(e)$ is independent from the particular choice of R.*

Proof. By the maximality of R we have that $S \setminus R$ is z-independent. This implies that

$$z(S) = \sum_{e\in S\setminus R} z(e) = \sum_{e\in S} z(e) - \sum_{e\in R} z(e).$$ □

Lemma 5.33. *Let $z\colon 2^E \to \mathbb{Z}_+$ be a normalized and monotone function satisfying that, for every z-independent set $S \subseteq E$, $z(S) = z^+(S)$. Assume that z^* is loopless (that is, $z^*(e) > 0$ for all $e \in E$). A set S is z^{**}-independent if and only if there exists a z-base $B \subseteq E$ that contains S. In particular, if S is z^{**}-independent then it is also z-independent. Moreover, we have that for all S,*

$$z^{**}(S) := \max\{z^+(J) : J \subseteq S \text{ s.t. } J \subseteq B \text{ for some } z\text{-base } B\}.$$

Proof. Let S be a z^{**}-independent set. By definition,

$$z^{**}(S) = \max\{(z^*)^+(J) : z^*(E \setminus J) = z^*(E), J \subseteq S\}.$$

Let J be a maximizer of this optimization problem. If J is a strict subset of S, then $z^{**}(J) = z^{**}(S)$, and hence, S is not z^{**}-independent. We conclude that $J = S$, which implies in particular that $z^*(E \setminus S) = z^*(E)$. Similarly, if $z^*(E \setminus S) = z^*(E)$, then

$z^{**}(S) = (z^*)^+(S) < (z^*)^+(S-e) = z^{**}(S-e)$ for all $e \in S$, where the inequality follows as z^* is loopless. We conclude that S is z^{**}-independent if and only if $z^*(E) = z^*(E \setminus S)$.

Consider now the two optimization problems defining $z^*(E \setminus S)$ and $z^*(E)$, which are,

$$z^*(E) := \max\{z^+(I) : z(E \setminus I) = z(E), I \subseteq E\},$$
$$z^*(E \setminus S) := \max\{z^+(I) : z(E \setminus I) = z(E), I \subseteq E \setminus S\}.$$

Notice that the family of feasible sets for the optimization problem defining $z^*(E \setminus S)$ is included in the family of feasible sets for the optimization problem defining $z^*(E)$. Then, if $z^*(E \setminus S) = z^*(E)$, there must exist a set $I \subseteq E \setminus S$ that is also optimal for $z^*(E)$. In particular, as z^* is loopless, $I \subseteq E$ is maximal, that is, for all $e \notin I$ it must hold that $z(E \setminus (I+e)) = z((E \setminus I) - e) < z(E)$. This implies that $E \setminus I$ is z-independent. As also $z(E \setminus I) = z(E)$, we conclude that $E \setminus I$ is a z-base. This implies that $S \subseteq (E \setminus I)$ is contained in a z-base.

Similarly, assume that there exists a z-base B such that $S \subseteq B$. Let $I = E \setminus B$. Then clearly $z(E \setminus I) = z(B) = z(E)$. Also I is inclusion-wise maximal, as B is z-independent. By Proposition 5.32, I must be optimal, and hence, $z^*(E) = z^+(I)$. Since $I \subseteq E \setminus S$, then I is also optimal for the optimization problem defining $z^*(E \setminus S)$. We conclude that $z^*(E) = z^*(E \setminus S)$.

With all this we conclude that S is z^{**}-independent if and only if there exists a z-base $B \subseteq E$ that contains S. It clearly follows that, if S is z^{**}-independent, then S is z-independent, as S is contained in a z-base. □

With the last lemma, we already observe that is not true that $z^{**} = z$ for any generalized rank function. This is in contrast to the case where z is the rank function of a matroid, where $z^{**} = z$ holds.

Definition 5.34. We say that a generalized rank function $z: 2^E \to \mathbb{Z}_+$ is *reflexive* if $z = z^{**}$.

By the previous lemma, we obtain that z is reflexive if and only if every maximal z-independent set is also a z-base. Equivalently, z is reflexive if and only if every z-independent set is contained in a z-base.

Proposition 5.35. *Consider a generalized rank function $z\colon 2^E \to \mathbb{Z}_+$ without coloops. Then, z is reflexive if and only if every inclusion-wise maximal z-independent set is a z-base.*

Proof. Suppose first that z is reflexive, and let S be an inclusion-wise maximal z-independent set. By Lemma 5.33, S is contained in some z-base, and due to its maximality must be a z-base as well.

Suppose now that every inclusion-wise maximal z-independent set is a z-base, and consider an arbitrary set S. Let $S' \subseteq S$ be a maximal z-independent set; by construction, $z(S) = z(S')$. Let S'' be an inclusion-wise maximal z-independent set containing S', which by hypothesis is a z-base. By Lemma 5.33, this implies that $z^{**}(S) \geq z(S') = z(S)$, and hence, $z^{**}(S) = z(S)$. As S was an arbitrary set, this means that z is reflexive. □

It is easy to construct examples of generalized ranks that are not reflexive. For example, consider the coverage function of the laminar family given by the collection of sets $\{\{a, b\}, \{a\}\}$. However, we notice that for a generalized rank arising from a Laminar Set Cover instance, we can remove all sets that are not contained in a z-base (equivalently, we remove all elements that are loops of z^{**}, which we call *cocoloops*). This does not change the collection of z-bases. Notice that for solving a Submodular Cover instance, we only need to consider the collection of z-bases, so removing the cocoloops do not change the optimal solution of the submodular cover instance. Moreover, in this case is not hard to see that removing the cocoloops yield a new function which is reflexive.

Lemma 5.36. *Let $z\colon 2^E \to \mathbb{Z}_+$ be a loopless generalized rank function. Then, z^* is reflexive.*

Proof. Let S be a inclusion-wise maximal z^*-independent set. We need to show that S is a z^*-base. We know that

$$z^*(S) = \max\{z^+(J) : z(E \setminus J) = z(E), J \subseteq S\}.$$

As S is z^* independent, then the maximizer J of this optimization function must be $J = S$, as otherwise $z^*(J) = z^*(S)$. Therefore, $z(E \setminus S) = z(E)$. As S is a maximal z-independent set, then S is inclusion-wise maximal such that $z(E) = z(E \setminus S)$. By Proposition 5.32, $z^+(S)$ has the same value as any other maximal set with $z(E) = z(E \setminus S)$. Therefore, if B is a z^*-base, then $z^+(B) = z^+(S)$, and therefore $z^*(S) = z^*(B)$. This implies that S is also a z^*-base. We conclude that z^* is reflexive. □

Given the last lemma, we obtain that $z^* \equiv z^{***} = (z^{**})^*$. As z^{**} is reflexive, a possible way to show that z^* is submodular is to show that z^{**} is submodular. This would imply that the dual of a generalized rank is always a generalized rank.

The next lemma gives an identity satisfied by z^* and z^{**} for any generalized rank function z.

Lemma 5.37. *Let $z: 2^E \to \mathbb{Z}_+$ be a generalized rank function without coloops. Then, for any $S \subseteq E$, it holds that*

$$z^*(S) = z^+(S) - z(E) + z^{**}(E \setminus S).$$

Proof. By definition we have that

$$z^*(S) = \max\{z^+(J) : z(E \setminus J) = z(E), J \subseteq S\}.$$

Taking $I = E \setminus J$ then

$$z^*(S) = \max\{z^+(E \setminus I) : z(I) = z(E), E \setminus S \subseteq I\}.$$

As z^+ is modular, it holds that $z^+(E \setminus I) = z^+(E) - z^+(I)$, and thus

$$z^*(S) = z^+(E) - \min\{z^+(I) : z(I) = z(E), (E \setminus S) \subseteq I\}.$$

Also, $z(I) = z(E)$ is equivalent to the fact that I contains a z-base B. Moreover, as $z^+(e) \geq 0$ for all $e \in E$ and we are minimizing $z^+(I)$, in an optimal I it holds that $I = (E \setminus S) \cup B$ for some z-base B. Therefore,

$$z^*(S) = z^+(E) - \min\{z^+(B) + z^+((E \setminus S) \setminus B)) : B \text{ is a } z\text{-base}\}.$$

Since $z^+(B) = z(E)$ is a constant, and $z^+((E \setminus S) \setminus B)) = z^+(E \setminus S) - z^+(B \cap (E \setminus S))$ we obtain that

$$z^*(S) = z^+(E) - z(E) - z^+(E \setminus S) + \max\{z^+((E \setminus S) \cap B)) : B \text{ is a } z\text{-base}\}.$$

Finally, we observe that $\max\{z^+((E \setminus S) \cap B)) : B \text{ is a } z\text{-base}\}$ is $z^{**}(E \setminus S)$ by Lemma 5.33. □

Corollary 5.38. *If $z\colon 2^E \to \mathbb{Z}_+$ is a reflexive generalized rank without coloops, then z^* is also a generalized rank.*

Proof. We just need to argue that z^* is submodular as the remaining conditions are already subject of Proposition 5.31. If z is reflexive then $z = z^{**}$. Thus, by Lemma 5.37, $z^*(S) = z^+(S) - z(E) + z(E \setminus S)$. As z^+ is modular, $z(E)$ is constant, and $z(E \setminus S)$ is submodular, we conclude that z^* is submodular. □

The main open problem of this section is given by the following conjecture.

Conjecture 5.39. *If $z\colon 2^E \to \mathbb{Z}_+$ is a generalized rank, then z^* is a generalized rank.*

5.7 Efficient Implementations of the Removal Oracle

In Section 5.5.1 we gave some sufficient conditions to when Greedy+ returns an optimal solution. However, to make Greedy+ efficient, we still need to show that the set $R(S, e)$ for any $S \subseteq E$ and $e \in E \setminus S$ can also be efficiently computed. Previously, we just queried a *removal oracle* and represented its running time by RO. If it is the case that RO is polynomial in $|E|$, then the arg min computation in Line 3 of Greedy+ just requires a linear number of computations of those sets $R(S, e)$ (a linear search) and hence, we have a total running time that is polynomial in $|E|$.

5.7.1 Laminar Set Cover

As a warm-up, we first show that if z is the coverage function of a Laminar Set Cover instance, then $R(S, e)$ can be computed greedily as $R(S, e)$ will be a z-base (actually a unique one) of maximum cost in a simple partition matroid.

Lemma 5.40. *Let $z\colon 2^E \to \mathbb{Z}_+$ be a coverage function for a* Laminar Set Cover *instance, $S \subseteq E$ z-independent and $e \in E \setminus S$ such that $z(S + e) > z(S)$. Then $(S, \mathcal{R}(S, e))$ with $\mathcal{R}(S, e) = \{R \subseteq S \mid z((S \setminus R) + e) = z(S + e)\}$ is a matroid where $\mathcal{R}(S, e)$ is the collection of independent sets.*

Proof. Clearly, $\emptyset \in \mathcal{R}(S, e)$ and by monotonicity $R \subseteq R' \in \mathcal{R}(S, e)$ implies $R \in \mathcal{R}(S, e)$. For the augmentation property, let $R, R' \in \mathcal{R}(S, e)$ with $|R| < |R'|$. Note that for all

$f \in R \in \mathcal{R}(S, e)$, we have $f \subseteq e$. By pigeon hole principle, there exists $f \in R' \setminus R$. Any of these f are suitable to augment R:

$$\begin{aligned} z(S + e \setminus (R + f)) &= \left| \bigcup_{g \in S + e \setminus (R+f)} g \right| \\ &= \left| \bigcup_{g \in S \setminus (R+f)} g \cup \underbrace{e}_{\substack{e \supseteq g \\ \text{for all } g \in R+f}} \right| \\ &= \left| \bigcup_{g \in S} g \cup e \right| \\ &= z(S + e). \end{aligned}$$

Thus, $R + f \in \mathcal{R}(S, e)$. □

More explicitly, $(S, \mathcal{R}(S, E))$ is a partition matroid with partition $\{S_1, S_2\}$, where $S_1 = \{f \in S \mid f \cap e \neq \varnothing\}$, $S_2 = S \setminus S_1$, and capacities $k_1 = |S_1|$, $k_2 = 0$.

5.7.2 Power-of-Two Knapsack Cover

To compute an optimal set $R(S, e)$ that can be removed upon adding e to S is itself indeed a Power-of-Two Knapsack Cover problem. We present a strategy to compute it efficiently.

Let S_τ be the set of elements that Greedy+ selected up until iteration τ. We maintain a binary tree $\mathcal{T} := \mathcal{T}_\tau = (V, A)$ with root r to store the elements in S_τ. Before we describe the tree in detail, we introduce some notation. For a vertex $v \in V$, we denote the subtree rooted at v by $\mathcal{T}(v)$ and the leaves of $\mathcal{T}(v)$ by $L(v)$. The elements $e \in S_\tau$ will be the leaves of $\mathcal{T}$.

We denote the weight and cost of a vertex $v \in V$ by the sum of weights and costs, respectively, of $L(v)$, i.e.,

$$w(v) = \sum_{e \in L(v)} w(e) \qquad \text{and} \qquad c(v) = \sum_{e \in L(v)} c(e).$$

The *level* (or *height*) of a vertex $v \in V$ is denoted by $h(v)$ and is defined as the number of edges on the unique path from r to v (hence, $h(r) = 0$). Each vertex $v \in V$ will represent

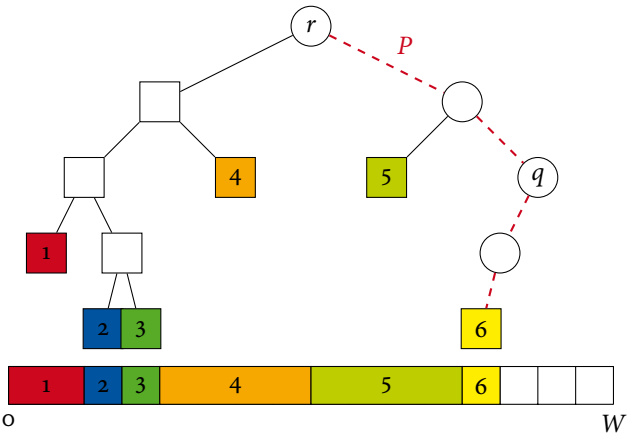


Figure 5.5: An example of the tree data structure for Power-of-Two Knapsack Cover. The active vertices are round, tight vertices are square. Only the root r, the last right child q of P and leaves are carrying a label (the latter are elements from the current set S_τ). We highlighted the path P with a red dashed line.

a *block* (of the knapsack) of size $2^{m-h(v)}$. We call a vertex $v \in V$ with $w(v) = 2^{m-h(v)}$ *tight*, otherwise *active*. The knapsack is full if $w(r) = 2^m = W$.

We maintain the following invariants on $\mathcal{T}$ during the execution of Greedy+:

(I1) $w(v) \leq 2^{m-h(v)}$ for all $v \in V$

(I2) Every $e \in S_\tau$ is placed as a leaf at level $h = m - \log_2 w(e)$, (in other words $w(e) = 2^{m-h}$, i.e., every leaf is tight).

(I3) For every $v \in V$, if v has a right child, then v has a tight left child.

(I4) For each inner vertex $v \in V$ it holds that it is tight if and only if both of its children are tight.

(I5) On each level h, the tight vertices are sorted from left to right in non-decreasing cost.

From (I3) and (I4) it follows that on each level, there is at most one vertex that is not tight and this vertex is necessarily the right-most one on this level. Moreover, as long as the knapsack is not full, i.e., $w(r) < W$, there is a unique path P from r to a tight vertex such that all vertices on this path are active except for the last one (it might consist only of r). We denote the last vertex on this path that is a right child by q (which has to be an inner vertex if the knapsack is not full yet; we treat r as a right child if necessary). Figure 5.5 illustrates how knapsack blocks and subtrees relate to each other.

Consider now again the vertex q and denote its level $\ell = h(q)$. If the next element $e_{\tau+1}$ that Greedy+ selects has weight 2^{k-j} with $j \leq \ell$, then we perform the sole removal step $S_{\tau+1} := S_\tau + e_{\tau+1} \setminus R(S_\tau, e_{\tau+1})$, where $R(S_\tau, e_{\tau+1}) = L(p)$ with p being the unique active vertex on p at level j (between r and q). In this case Greedy+ terminates.

Now suppose $w(e_{\tau+1}) = 2^{m-j}$ for $j > \ell$. Then $e_{\tau+1}$ will just be added to S_τ (i.e., $R(S_\tau, e_{\tau+1}) = \varnothing$) and we need to update $\mathcal{T}$.

Either way, note that the tree $\mathcal{T}$ gives us an efficient way to compute the arg min in Line 3 of Greedy+. Once $e_{\tau+1}$ is determined, we can perform an update step on $\mathcal{T}$ (provided that indeed $w(e_{\tau+1}) < 2^{k-\ell}$).

Let h be the required level of $e_{\tau+1}$ (as per invariant (I2)). Note that the level h contains at most one active vertex we call it α if it exists.

Update Tree:

Input: previous tree $\mathcal{T}_\tau$ for set S_τ, new element $e_{\tau+1}$ for level h
Output: optimal removal set $R(S_\tau, e_{\tau+1})$ or updated tree $\mathcal{T}_{\tau+1}$

1 **if** $w(S_\tau) + w(e_{\tau+1}) \geq W$ **then**
2 Determine unique vertex α on P at level h
3 **return** $L(\alpha)$

`// Otherwise, construct` $\mathcal{T}_{\tau+1}$`:`

4 **if** *level h contains an active vertex α* **then**
5 **if** *α is a left child* **then**
6 Replace $\mathcal{T}(\alpha)$ with $e_{\tau+1}$ and add $\mathcal{T}(\alpha)$ as right child of α's parent
7 **else**
8 Let β be the first right child on the path from α to r (other than α) and γ be β's left child
9 Replace $\mathcal{T}(\alpha)$ with $e_{\tau+1}$ and add $\mathcal{T}(\alpha)$ to the subtree on β (adjust for height)
10 **for** $\ell := h - 1$ **downto** $h(\gamma) - 1$ **do**
11 Reorder Subtrees on ℓ and γ
12 **else**
13 Let β be the active vertex with the highest level $h(\beta)$
14 Add a right child to β and then as many left children as necessary to add $e_{\tau+1}$ as leaf
15 **return** $\mathcal{T}_{\tau+1}$

Note that by design, most of the invariants are clearly fulfilled after calling Update Tree. The only non-obvious one is (I5) if we are in the second case as the block γ which

just became tight might be cheaper than other blocks on its level to the left. This gets resolved using the following subroutine and has to be iterated as rearranging a level can also upset the order on a lower level (closer to the root).

Reorder Subtrees:

Input: current tree $\mathcal{T}$, level ℓ, vertex γ
Output: Reordered tree such that invariant (I5) is recovered on ℓ

Let $a_1 \prec a_2 \prec \ldots \prec a_p$ be an ordering of tight vertices other than γ on level ℓ
if $c(\gamma) < c(a_p)$ **then**
    Let i be the maximum index with $c(a_i) \leq c(\gamma)$
    Replace $T(a_{i+1})$ with $T(\gamma)$ and move all further $T(a_j)$ to the right

Intuitively, Reorder Subtrees moves a subtree rooted at γ to the left by exchanging it with its siblings and cousins until the order at level ℓ is restored. The Update Tree method then calls the same method on γ's new parent to reorder the subtrees on its level.

We depicted in Figure 5.6 how the tree $\mathcal{T}$ changes after insertion of some new element.

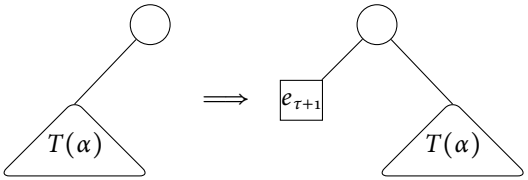
Lemma 5.41. *The invariants (I1)-(I5) are maintained throughout the algorithm.*

Proof. At the beginning with $S_0 = \varnothing$ and $\mathcal{T}_0$ only consisting of the root r the invariants are all clearly fulfilled.

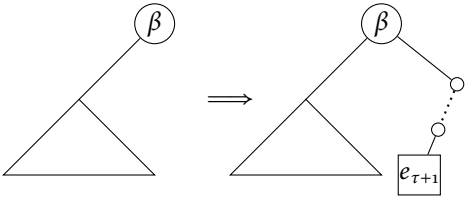
Now let us assume $\mathcal{T}_\tau$ satisfies all invariants and we add an item $e_{\tau+1}$ that does not trigger the end of the algorithm, i.e., we need to update the tree at level h. We check each invariant for each case:

Case 1: Level h does not contain an active vertex. (**else** case in Line 14). In that case, $\mathcal{T}_{\tau+1}$ mostly stays the same as $\mathcal{T}_\tau$, we just added a right child to the last active vertex on the path P and sufficiently many left children on that. By construction, (I1)-(I4) are all satisfied in $\mathcal{T}_{\tau+1}$. By Lemma 5.23, the cost-to-weight ratio of $e_{\tau+1}$, i.e., $\frac{c(e_{\tau+1})}{w(e_{\tau+1})}$, is larger or equal than of all other elements in S_τ and hence, also for every tight vertex of $\mathcal{T}$. Thus, (I5) is also still satisfied.

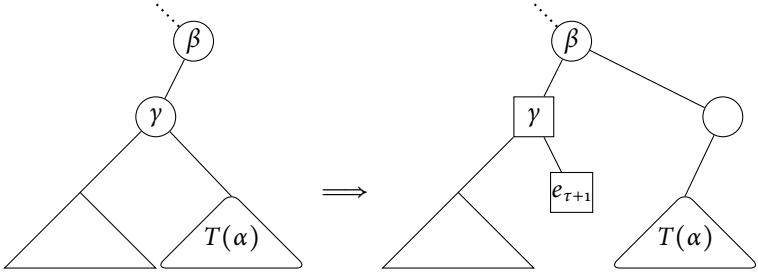
Case 2: Level h contains an active vertex α and it is a left child (Line 6). The invariants (I1)-(I4) again hold by construction and (I5) follows again by Lemma 5.23. Note that α is at level h but now as a right child. It has a better cost-to-weight ratio than $e_{\tau+1}$ (also as per Lemma 5.23 but it is still an active vertex and hence, not a tight one, so we do not violate (I5).



(a) Level h contains an active vertex α which is a left child.



(b) Level h does not contain an active vertex.



(c) Level h contains an active vertex α which is a right child.

Figure 5.6: Illustration how Update Tree changes the tree $\mathcal{T}$ when inserting a new element $e_{\tau+1}$. Circles and *rounded* triangles denote active vertices and subtrees with active root, respectively, and squares and *sharp* triangles tight vertices and subtrees that only have tight vertices, respectively.

Case 3: Level h contains an active vertex α and it is a right child (Line 9). Again, (I1)-(I4) are satisfied by construction. In particular for (I4), we have that all vertices in $T(\gamma)$ are tight now. Also note that such a γ exists as otherwise $e_{\tau+1}$ would have exceeded the knapsack's capacity.

At this point, the parent of $e_{\tau+1}$ (and all vertices up to γ) might violate (I5) as it turned from active to tight and hence, have to be sorted into its correct position which we do by calling Reorder Subtrees. We call this subroutine recursively to also reorder the vertices at a lower level (closer to the root) as they get affected by the reordering on the higher level. □

Running Time Performing Line 3 in Greedy+ requires to iterate over all $e \in E \setminus S_\tau$ and compute $R(S_\tau, e)$ which requires constant time if e is small enough such that the knapsack is not full after its addition. Otherwise, we need to find the right vertex (q) where e would trigger a removal step. Naïvely, we then could find $L(q)$ (if we save the tree explicitly, then a BFS or DFS does the trick, which needs time linear in S_τ). However, we can do this step more efficiently as the leaves $L(q)$ are not actually needed to compute the arg min, just their cost. For every internal vertex v we can store its costs $c(v)$ in the vertex, so finding q is sufficient. For that, we do not even need a BFS as we just have to follow the path P (which is always taking the right child if it exists and otherwise taking the left child unless it is tight).

However, after determining $\bar{e}$, we have to do some work to update the tree. The insertion requires essentially just a constant number of operations (moving pointers around, creating some new ones). The reordering of the subtrees should also be linear as every level has at most $|S_\tau|$ many vertices that have to be reordered. Whenever we swap subtrees, we also need to compute the costs of the ancestor vertices, which again are just linearly many.

CHAPTER 6

Conclusion

In this final chapter, we look back at the results of this thesis but also point out the open problems and future research directions.

6.1 Review

We saw that matroid theory has proven itself to be useful in the context of dynamic auctions. Matroids (or let us say matroidish structures) occur naturally in form of demand correspondences if the valuation functions of buyers are (strong) gross substitutes—a very general class of valuation functions that have the natural decreasing marginal returns property. This allows us to use methods to determine excess demand sets (or excess supply sets) with rather simple complexity. The achievement is twofold: Algorithms for Matroid Union and Polymatroid Sum (and of course Maximum Flow) are rather simple to implement on a computer provided, we can represent valuation functions. But more importantly, these methods are also fast and this came with a little bit of a surprise. Algorithms for minimizing a submodular function (or $L^\natural$-convex function) are sledgehammer methods for all kinds combinatorial optimization problems on the one hand. However, one the other hand, they are also quite tailored for finding excess demand sets as strong gross substitute valuations come with their equivalence to $M^\natural$-concave functions that give us exactly $L^\natural$-convex functions to minimize. It was

quite astonishing that in the end, we were able to not just propose a simple method to solve the problem but also to do it faster.

We also gave a simplified analysis of an ascending auction that sells at equilibrium a maximum weight base of a matroid. In this auction, matroids were not just appearing as a hidden part of the mechanism, they were rather put in there in the economic model. Yet, these market are quite natural and allowed for a more general notion of "clearing the market", i.e., by imposing a matroid constraint instead of a partitioning constraint.

Finally, we attempted to give optimality conditions for a greedy algorithm that is not as stubborn in the sense that it will take back some choices if there is an easy improvement step using a one-to-many exchange instead of just additions. This gave rise to a class of submodular functions we call generalize ranks, which inherit some useful properties from matroid rank functions.

6.2 Open Problems

As researchers, we never run out of problems to explore—each answer uncovers new questions, and every advance reveals deeper layers of complexity. For our market settings with Walrasian equilibria, recall that we only looked at a *pure exchange economy*—a market in which there is a fixed set of items and buyers who want to buy them. However, there is a whole second side of these *Arrow-Debreu markets* that is omitted from any considerations of this thesis, that is the *production* side, whereas we only looked at the *consumption* side. Also for these two-sided markets, it is known that Walrasian equilibria exist under just mild conditions. It is now very natural to ask, whether we can extend our methods of finding those equilibria in the two-sided model.

We already looked at monotonicity of extremal Walrasian equilibria, when we change the supply and demand of a market (for a suitable mathematical model of those terms, but not without alternatives). There are more general techniques in *comparative statics* that might be adaptable to find stronger results, e.g., how non-extremal Walrasian equilibria react to changes in supply and demand (or other parameters). A promising direction would be, for instance, to develop or explore discrete envelope theorems and apply them to the market.

These are interesting further research directions that are worth exploring. However, the previous chapter was meant to open up a new theory of a subclass of submodular functions and therefore offers probably the most open problems, which are also more

foundational. Above all stands Conjecture 5.39 which would make duality theory of generalized ranks much stronger. As we tried to draw parallels to matroid theory, there is the question, whether we can also describe set systems of z-independent sets axiomatically, i.e. find cryptomorphisms for generalized ranks. We also saw the Greedy+ does not find an optimal solution to problem of finding a minimum cost maximum weight base of a matroid. However, there is a simple fix to this by allowing the algorithm to also make one-to-many exchanges that do not increase the coverage (and just lower costs). It takes further research to check if we can get optimality guarantees for all generalized ranks, ideally necessary and sufficient conditions. Another interesting questions is whether we can make also more general statements about the removal oracle that is used by Greedy+, in particular, under which conditions it can be computed efficiently. Finally, note that Greedy+ is an algorithm that is natural to use on all kinds of covering problems. It would be interesting to determine termination guarantees, running time bounds, and approximation factors for those.

Bibliography

[AAT13] Tommy Andersson, Christer Andersson, and A.J.J. Talman. "Sets in excess demand in simple ascending auctions with unit-demand bidders". In: *Annals of Operations Research* 211 (2013), pp. 27–36.

[AD54] Kenneth J. Arrow and Gerard Debreu. "Existence of an equilibrium for a competitive economy". In: *Econometrica: Journal of the Econometric Society* 22(3) (1954), pp. 265–290.

[AL20] Mohammad Akbarpour and Shengwu Li. "Credible auctions: A trilemma". In: *Econometrica: Journal of the Econometric Society* 88.2 (2020), pp. 425–467.

[AM02] Lawrence M. Ausubel and Paul R. Milgrom. "Ascending auctions with package bidding". In: *The BE Journal of Theoretical Economics* 1.1 (2002), p. 20011001.

[AM06] Lawrence M. Ausubel and Paul R. Milgrom. "The lovely but lonely Vickrey auction". In: *Combinatorial auctions* 17.3 (2006), pp. 22–26.

[AMO88] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network flows*. Cambridge, Mass.: Alfred P. Sloan School of Management, Massachusetts, 1988.

[Arr51] Kenneth J. Arrow. "An Extension of the Basic Theorems on Classical Welfare Economics". In: *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*. University of California Press, 1951, pp. 507–532.

[Aus04] Lawrence M. Ausubel. "An efficient ascending-bid auction for multiple objects". In: *American Economic Review* 94.5 (2004), pp. 1452–1475.

[Aus05] Lawrence M. Ausubel. "Walrasian tatonnement for discrete goods". In: *Unpublished paper* (2005).

[Aus06] Lawrence M. Ausubel. "An efficient dynamic auction for heterogeneous commodities". In: *American Economic Review* 96.3 (2006), pp. 602–629.

[Bal+24] Elizabeth Baldwin, Martin Bichler, Maximilian Fichtl, and Paul Klemperer. "Strong substitutes: structural properties, and a new algorithm for competitive equilibrium prices". In: *Mathematical Programming* 203.1 (2024), pp. 611–643.

[Ben17] Oren Ben-Zwi. "Walrasian's characterization and a universal ascending auction". In: *Games and Economic Behavior* 104 (2017), pp. 456–467.

[Ber92] Dimitri P. Bertsekas. "Auction algorithms for network flow problems: A tutorial introduction". In: *Computational optimization and applications* 1 (1992), pp. 7–66.

[Bik+01] Sushil Bikhchandani, Sven de Vries, James Schummer, and Rakesh V. Vohra. "Linear programming and Vickrey auctions". In: *IMA Volumes in Mathematics and its Applications* 127 (2001), pp. 75–116.

[Bik+08] Sushil Bikhchandani, Sven de Vries, James Schummer, and Rakesh V. Vohra. "Ascending auctions for integral (poly)matroids with concave nondecreasing separable values". In: *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008, San Francisco, California, USA, January 20-22, 2008*. Ed. by Shang-Hua Teng. SIAM, 2008, pp. 864–873.

[Bik+11] Sushil Bikhchandani, Sven de Vries, James Schummer, and Rakesh V. Vohra. "An ascending vickrey auction for selling bases of a matroid". In: *Operations research* 59.2 (2011), pp. 400–413.

[Bil22] Jeff Bilmes. "Submodularity in machine learning and artificial intelligence". In: *arXiv preprint arXiv:2202.00132* (2022).

[BKL24] Elizabeth Baldwin, Paul Klemperer, and Edwin Lock. "Implementing Walrasian Equilibrium: the Languages of Product-mix Auctions". In: *Available at SSRN 4931623* (2024).

[Bli+23] Joakim Blikstad, Sagnik Mukhopadhyay, Danupon Nanongkai, and Ta-Wei Tu. "Fast Algorithms via Dynamic-Oracle Matroids". In: *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*. 2023, pp. 1229–1242.

[BLN13] Oren Ben-Zwi, Ron Lavi, and Ilan Newman. "Ascending auctions and walrasian equilibrium". In: *arXiv preprint arXiv:1301.1153* (2013).

[BN05] Liad Blumrosen and Noam Nisan. "On the computational power of iterative auctions". In: *Proceedings of the 6th ACM Conference on Electronic Commerce*. 2005, pp. 29–43.

[BN10] Liad Blumrosen and Noam Nisan. "On the computational power of demand queries". In: *SIAM Journal on Computing* 39.4 (2010), pp. 1372–1391.

[BNP09] Moshe Babaioff, Noam Nisan, and Elan Pavlov. "Mechanisms for a spatially distributed market". In: *Games and Economic Behavior* 66.2 (2009), pp. 660–684.

[BO02] Sushil Bikhchandani and Joseph M Ostroy. "The package assignment model". In: *Journal of Economic theory* 107.2 (2002), pp. 377–406.

[Bru69] Richard A. Brualdi. "Comments on bases in dependence structures". In: *Bulletin of the Australian Mathematical Society* 1.2 (1969), pp. 161–167.

[BS05] Felix Brandt and Tuomas Sandholm. "Unconditional privacy in social choice". In: *TARK*. 2005, pp. 207–218.

[BS68] Richard A. Brualdi and Edward B. Scrimger. "Exchange systems, matchings, and transversals". In: *Journal of Combinatorial Theory* 5.3 (1968), pp. 244–257.

[Car+99] Robert D. Carr, Lisa K. Fleischer, Vitus J. Leung, and Cynthia A. Phillips. *Strengthening integrality gaps for capacitated network design and covering problems*. Tech. rep. Sandia National Lab.(SNL-NM), Albuquerque, NM (United States); Sandia …, 1999.

[Cas23] Ralph Cassady Jr. *Auctions and auctioneering*. Univ of California Press, 2023.

[Cha+17] Deeparnab Chakrabarty, Yin Tat Lee, Aaron Sidford, and Sam Chiu-wai Wong. "Subquadratic submodular function minimization". In: *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*. 2017, pp. 1220–1231.

[Cha+19] Deeparnab Chakrabarty, Yin Tat Lee, Aaron Sidford, Sahil Singla, and Sam Chiu-Wai Wong. "Faster matroid intersection". In: *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2019, pp. 1146–1168.

[Che+23] Li Chen, Rasmus Kyng, Yang P. Liu, Richard Peng, Maximilian Probst Gutenberg, and Sushant Sachdeva. "Almost-linear-time algorithms for maximum flow and minimum-cost flow". In: *Communications of the ACM* 66.12 (2023), pp. 85–92.

[Chv79] Vašek Chvátal. "A greedy heuristic for the set-covering problem". In: *Mathematics of operations research* 4.3 (1979), pp. 233–235.

[CJK14] Deeparnab Chakrabarty, Prateek Jain, and Pravesh Kothari. "Provable submodular minimization via Fujishige-Wolfe algorithm". In: *Adv. in Neu. Inf. Proc. Sys.(NIPS)* (2014).

[Cla71] Edward H. Clarke. "Multipart pricing of public goods". In: *Public choice* 11.1 (1971), pp. 17–33.

[COP15] Ozan Candogan, Asuman Ozdaglar, and Pablo A. Parrilo. "Iterative auction design for tree valuations". In: *Operations Research* 63.4 (2015), pp. 751–771.

[Cor+22] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to algorithms*. Vol. 4. MIT press, 2022.

[Cra98] Peter Cramton. "Ascending auctions". In: *European Economic Review* 42.3-5 (1998), pp. 745–756.

[CS08] Tim Carnes and David Shmoys. "Primal-dual schema for capacitated covering problems". In: *International Conference on Integer Programming and Combinatorial Optimization*. Springer. 2008, pp. 288–302.

[Cun85] William H. Cunningham. "On submodular function minimization". In: *Combinatorica* 5.3 (1985), pp. 185–192.

[Cun86] William H. Cunningham. "Improved bounds for matroid partition and intersection algorithms". In: *SIAM Journal on Computing* 15.4 (1986), pp. 948–957.

[CW20] Natalie Collina and S. Matthew Weinberg. "On the (in-) approximability of Bayesian Revenue Maximization for a Combinatorial Buyer". In: *Proceedings of the 21st ACM Conference on Economics and Computation*. 2020, pp. 477–497.

[Dan63] George B. Dantzig. "Linear programming and extensions". In: *Linear programming and extensions*. Princeton University Press, 1963.

[Daw80] Jeremy E. Dawson. "Optimal matroid bases: An algorithm based on cocircuits". In: *The Quarterly Journal of Mathematics* 31.1 (1980), pp. 65–69.

[Deb51] Gerard Debreu. "The coefficient of resource utilization". In: *Econometrica: Journal of the Econometric Society* (1951), pp. 273–292.

[DGS86] Gabrielle Demange, David Gale, and Marilda Sotomayor. "Multi-item auctions". In: *Journal of political economy* 94.4 (1986), pp. 863–872.

[Din70] Efim A. Dinic. "Algorithm for solution of a problem of maximum flow in networks with power estimation". In: *Soviet Math. Doklady*. Vol. 11. 1970, pp. 1277–1280.

[DK69] EA Dinic and MA Kronrod. "An algorithm for the solution of the assignment problem". In: *Soviet Math. Dokl.* Vol. 10. 6. 1969, pp. 1324–1326.

[Dob82] Gregory Dobson. "Worst-case analysis of greedy heuristics for integer programming with nonnegative data". In: *Mathematics of Operations Research* 7.4 (1982), pp. 515–531.

[DS12] Ran Duan and Hsin-Hao Su. "A scaling algorithm for maximum weight matching in bipartite graphs". In: *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*. SIAM. 2012, pp. 1413–1424.

[Edm68] Jack Edmonds. "Matroid partition". In: *Mathematics of the Decision Sciences* (1968), pp. 335–345.

[Edm70] Jack Edmonds. "Submodular functions, matroids, and certain polyhedra". In: *Combinatorial Structures and their Applications*. Ed. by Richard Guy, Haim Hanani, Norbert Sauer, and Johanan Schonheim. University of Calgary, Alberta, Canada: Gordon and Breach, New York, 1970, pp. 69–87.

[EF65] Jack Edmonds and Delbert Ray Fulkerson. "Transversals and matroid partition". In: *J. Res. Nat. Bur. Standards Sect. B* 69 (1965), pp. 147–153.

[Eic+23a] Katharina Eickhoff, S. Thomas McCormick, Britta Peis, Niklas Rieken, and Laura Vargas Koch. "A flow-based ascending auction to compute buyer-optimal Walrasian prices". In: *arXiv preprint arXiv:2304.14262* (2023).

[Eic+23b] Katharina Eickhoff, Britta Peis, Niklas Rieken, Laura Vargas Koch, and László A. Végh. "Faster Ascending Auctions via Polymatroid Sum". In: *Web and Internet Economics (WINE)* (2023), p. 682.

[Eic+24a] Katharina Eickhoff, S. Thomas McCormick, Britta Peis, Niklas Rieken, and Laura Vargas Koch. "A flow-based ascending auction to compute buyer-optimal Walrasian prices". In: *Networks* 84.2 (2024), pp. 161–180.

[Eic+24b] Katharina Eickhoff, Meike Neuwohner, Britta Peis, Niklas Rieken, Laura Vargas Koch, and László A. V'egh. *Faster Dynamic Auctions via Polymatroid Sum*. 2024.

[Eic+25] Katharina Eickhoff, Meike Neuwohner, Britta Peis, Niklas Rieken, Laura Vargas Koch, and László A. Végh. "Faster Dynamic Auctions via Polymatroid Sum". In: *ACM Trans. Econ. Comput.* 13.3 (June 2025).

[EK72] Jack Edmonds and Richard M. Karp. "Theoretical improvements in algorithmic efficiency for network flow problems". In: *Journal of the ACM (JACM)* 19.2 (1972), pp. 248–264.

[Fei98] Uriel Feige. "A threshold of ln n for approximating set cover". In: *Journal of the ACM (JACM)* 45.4 (1998), pp. 634–652.

[FF56] Lester R. Ford Jr. and Delbert R. Fulkerson. "Maximal flow through a network". In: *Canadian journal of Mathematics* 8 (1956), pp. 399–404.

[FF62] Lester R. Ford Jr. and Delbert R. Fulkerson. "Flows in Networks". In: (1962).

[FM12] András Frank and Zoltán Miklós. "Simple push-relabel algorithms for matroids and submodular flows". In: *Japan Journal of Industrial and Applied Mathematics* 29 (2012), pp. 419–439.

[Fuj00] Toshihiro Fujito. "Approximation algorithms for submodular set cover with applications". In: *IEICE Transactions on Information and Systems* 83.3 (2000), pp. 480–487.

[Fuj05] Satoru Fujishige. *Submodular Functions and Optimization*. Elsevier, 2005.

[Fuj80] Satoru Fujishige. "Lexicographically Optimal Base of a Polymatroid with Respect to a Weight Vector". In: *Mathematics of Operations Research* 5.2 (1980), pp. 186–196.

[FY03] Satoru Fujishige and Zaifu Yang. "A note on Kelso and Crawford's gross substitutes condition". In: *Mathematics of Operations Research* 28.3 (2003), pp. 463–469.

[FZ92] Satoru Fujishige and Xiaodong Zhang. "New algorithms for the intersection problem of submodular systems". In: *Japan Journal of Industrial and Applied Mathematics* 9 (1992), pp. 369–382.

[FZ96] Satoru Fujishige and Xiaodong Zhang. "A Push/Relabel framework for submodular flows and its definement for 0-1 submodular flows". In: *Optimization* 38.2 (1996), pp. 133–154.

[GK97] Andrew V. Goldberg and Robert Kennedy. "Global price updates help". In: *SIAM Journal on Discrete Mathematics* 10.4 (1997), pp. 551–572.

[GKT51] David Gale, Harold W. Kuhn, and Albert W. Tucker. "Linear programming and the theory of games". In: *Activity analysis of production and allocation* 13 (1951), pp. 317–335.

[GLS81] Martin Grötschel, László Lovász, and Alexander Schrijver. "The ellipsoid method and its consequences in combinatorial optimization". In: *Combinatorica* 1 (1981), pp. 169–197.

[GM75] Curtis Greene and Thomas L. Magnanti. "Some abstract pivot algorithms". In: *SIAM Journal on Applied Mathematics* 29.3 (1975), pp. 530–539.

[Gro73] Theodore Groves. "Incentives in teams". In: *Econometrica: Journal of the Econometric Society* (1973), pp. 617–631.

[GS00] Faruk Gul and Ennio Stacchetti. "The English auction with differentiated commodities". In: *Journal of Economic theory* 92.1 (2000), pp. 66–95.

[GS99] Faruk Gul and Ennio Stacchetti. "Walrasian equilibrium with gross substitutes". In: *Journal of Economic theory* 87.1 (1999), pp. 95–124.

[GT88] Andrew V. Goldberg and Robert E. Tarjan. "A new approach to the maximum-flow problem". In: *Journal of the ACM (JACM)* 35.4 (1988), pp. 921–940.

[Hal35] Philip Hall. "On Representatives of Subsets". In: *Journal of the London Mathematical Society* s1-10.1 (1935), pp. 26–30.

[Hel72] Thorkell Helgason. "Aspects of the theory of hypermatroids". In: *Hypergraph Seminar: Ohio State University 1972*. Springer. 1972, pp. 191–213.

[HerBC] Herodotus. *Histories*. Vol. 1. c. 440 BC.

[Hoc82] Dorit S. Hochbaum. "Approximation algorithms for the set covering and vertex cover problems". In: *SIAM Journal on computing* 11.3 (1982), pp. 555–556.

[IFF01] Satoru Iwata, Lisa Fleischer, and Satoru Fujishige. "A combinatorial strongly polynomial algorithm for minimizing submodular functions". In: *Journal of the ACM (JACM)* 48.4 (2001), pp. 761–777.

[IO09] Satoru Iwata and James B Orlin. "A simple combinatorial algorithm for submodular function minimization". In: *Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms*. SIAM. 2009, pp. 1230–1237.

[IW10] Russell Impagliazzo and Ryan Williams. "Communication complexity with synchronized clocks". In: *2010 IEEE 25th Annual Conference on Computational Complexity*. IEEE. 2010, pp. 259–269.

[Iwa02] Satoru Iwata. "A fully combinatorial algorithm for submodular function minimization". In: *J. Comb. Theory, Ser. B* 84.2 (2002), pp. 203–212.

[Iwa03] Satoru Iwata. "A faster scaling algorithm for minimizing submodular functions". In: *SIAM Journal on Computing* 32.4 (2003), pp. 833–840.

[Kar72] Richard M. Karp. "Reducibility Among Combinatorial Problems". In: *Complexity of Computer Computations* (1972).

[Kar84] Narendra K. Karmarkar. "Some comments on the significance of the new polynomial-time algorithm for linear programming". In: *AT&T Bell Laboratories, Murray Hill, New Jersey* (1984).

[KC82] Alexander S. Kelso Jr. and Vincent P. Crawford. "Job matching, coalition formation, and gross substitutes". In: *Econometrica: Journal of the Econometric Society* (1982), pp. 1483–1504.

[Kem25] Luca Kempkes. "Tailoring a Matroid-Constrained Ascending Vickrey Auction to Representable Matroids". Bachelor's Thesis. Rheinisch-Westfälische Technische Hochschule Aachen, 2025.

[KG14] Andreas Krause and Daniel Golovin. "Submodular function maximization." In: *Tractability* 3.71-104 (2014), p. 3.

[Kha80] Leonid G. Khachiyan. "Polynomial algorithms in linear programming". In: *USSR Computational Mathematics and Mathematical Physics* 20.1 (1980), pp. 53–72.

[KK24] Peter Katuščák and Thomas Kittsteiner. "Strategy-Proofness Made Simpler". In: *Management Science* (2024).

[KKR24] Andrew Komo, Scott Duke Kominers, and Tim Roughgarden. "Shill-proof auctions". In: *arXiv preprint arXiv:2404.00475* (2024).

[KL84] Bernhard Korte and László Lovász. "Greedoids-A structural framework for the greedy algorithm". In: *Progress in combinatorial optimization*. Elsevier, 1984, pp. 221–243.

[Kle09] Paul Klemperer. "A new auction for substitutes: Central-Bank liquidity auctions,'toxic asset'auctions, and variable product-mix auctions". In: *Journal of the European Economic Association, Forthcoming* (2009).

[Kle99] Paul Klemperer. "Auction theory: A guide to the literature". In: *Journal of economic surveys* 13.3 (1999), pp. 227–286.

[KMU16] Walter Kern, Bodo Manthey, and Marc Uetz. "Note on VCG vs. Price Raising for Matching Markets". In: *arXiv preprint* (2016).

[Knu73] Donald E. Knuth. *Matroid partitioning*. Computer Science Department, Stanford University, 1973.

[Knu74] Donald E. Knuth. "The asymptotic number of geometries". In: *Journal of Combinatorial Theory, Series A* 16.3 (1974), pp. 398–400.

[Kru56] Joseph B. Kruskal. "On the shortest spanning subtree of a graph and the traveling salesman problem". In: *Proceedings of the American Mathematical society* 7.1 (1956), pp. 48–50.

[Kuh55] Harold W. Kuhn. "The Hungarian method for the assignment problem". In: *Naval research logistics quarterly* 2.1-2 (1955), pp. 83–97.

[Kun+25] Ike Kunze, Constantin Sander, Alexander Ruhrmann, Niklas Rieken, and Klaus Wehrle. "Using Explicit (Host-to-Network) Flow Measurements for Network Tomography". In: *Proceedings of the 2025 Applied Networking Research Workshop*. ANRW '25. Madrid, Spain: Association for Computing Machinery, 2025, pp. 54–61.

[LGM22] Edwin Lock, Paul W. Goldberg, and Francisco Marmolejo-Cossío. "Learning strong substitutes demand via queries". In: *ACM Transactions on Economics and Computation* 10.2 (2022), pp. 1–22.

[Li17] Shengwu Li. "Obviously strategy-proof mechanisms". In: *American Economic Review* 107.11 (2017), pp. 3257–3287.

[LLN06] Benny Lehmann, Daniel Lehmann, and Noam Nisan. "Combinatorial auctions with decreasing marginal utilities". In: *Games and Economic Behavior* 55.2 (2006). Mini Special Issue: Electronic Market Design, pp. 270–296.

[LSW15] Yin Tat Lee, Aaron Sidford, and Sam Chiu-wai Wong. "A faster cutting plane method and its implications for combinatorial and convex optimization". In: *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*. IEEE. 2015, pp. 1049–1065.

[May08] Dillon Mayhew. "Matroid complexity and nonsuccinct descriptions". In: *SIAM Journal on Discrete Mathematics* 22.2 (2008), pp. 455–466.

[Mos12] Dana Moshkovitz. "The projection games conjecture and the NP-hardness of ln n-approximating set-cover". In: *International Workshop on Approximation Algorithms for Combinatorial Optimization*. Springer. 2012, pp. 276–287.

[MP07] Debasis Mishra and David C. Parkes. "Ascending price Vickrey auctions for general valuations". In: *Journal of Economic Theory* 132.1 (2007), pp. 335–366.

[MS09] Paul R. Milgrom and Bruno Strulovici. "Substitute goods, auctions, and equilibrium". In: *Journal of Economic theory* 144.1 (2009), pp. 212–247.

[MS20] Paul R. Milgrom and Ilya Segal. "Clock auctions and radio spectrum reallocation". In: *Journal of Political Economy* 128.1 (2020), pp. 1–31.

[MSY13] Kazuo Murota, Akiyoshi Shioura, and Zaifu Yang. "Computing a Walrasian equilibrium in iterative auctions with multiple differentiated items". In: *International Symposium on Algorithms and Computation*. Springer. 2013, pp. 468–478.

[MSY16] Kazuo Murota, Akiyoshi Shioura, and Zaifu Yang. "Time bounds for iterative auctions: A unified approach by discrete convex analysis". In: *Discrete Optimization* 19 (2016), pp. 36–62.

[MT10] Debasis Mishra and Dolf Talman. "Characterization of the Walrasian equilibria of the assignment model". In: *Journal of Mathematical Economics* 46.1 (2010), pp. 6–20.

[Mur03] Kazuo Murota. *Discrete Convex Analysis*. Society for Industrial and Applied Mathematics (SIAM), 2003.

[Mye81] Roger B. Myerson. "Optimal auction design". In: *Mathematics of operations research* 6.1 (1981), pp. 58–73.

[Nas50] John F. Nash Jr. "Non-cooperative Games". PhD thesis. Princeton University, 1950.

[Neu28] John von Neumann. "Zur Theorie der Gesellschaftsspiele". In: *Mathematische Annalen* 100.1 (1928), pp. 295–320.

[Nis+07] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V. Vazirani. *Algorithmic game theory*. Cambridge university press, 2007.

[Nis00] Noam Nisan. "Bidding and allocation in combinatorial auctions". In: *Proceedings of the 2nd ACM Conference on Electronic Commerce*. 2000, pp. 1–12.

[NM47] John von Neumann and Oskar Morgenstern. "Theory of games and economic behavior, 2nd rev". In: (1947).

[NS06] Noam Nisan and Ilya Segal. "The communication requirements of efficient allocations and supporting prices". In: *Journal of Economic Theory* 129.1 (2006), pp. 192–224.

[Orl09] James B Orlin. "A faster strongly polynomial time algorithm for submodular function minimization". In: *Mathematical Programming* 118.2 (2009), pp. 237–251.

[Oxl06] James G. Oxley. *Matroid Theory*. Vol. 3. Oxford University Press, USA, 2006.

[Pae17] Renato Paes Leme. "Gross substitutability: An algorithmic survey". In: *Games and Economic Behavior* 106 (2017), pp. 294–316.

[Pap94] Christos H. Papadimitriou. *Computational Complexity*. Ed. by Tom Stone. Addison-Wesley, 1994.

[Par06a] Vilfredo Pareto. *Manuale di economia politica con una introduzione alla scienza sociale*. Vol. 13. Società editrice libraria, 1906.

[Par06b] David C. Parkes. *Iterative combinatorial auctions*. MIT press, 2006.

[Pat70] R.W. Patten. "Tatworth candle auction". In: *Folklore* 81.2 (1970), pp. 132–135.

[Pei+22] Britta Peis, Niklas Rieken, José Verschae, and Andreas Wierz. "A Primal-Dual and Primal-Greedy Approximation Framework for Weighted Covering Problems". In: *15th Workshop on Models and Algorithms for Planning and Scheduling Problems, MAPSP 2022*. Biella, Italy, 2022, pp. 77–80.

[PR24] Britta Peis and Niklas Rieken. "A Simplified Analysis of the Ascending Auction to Sell a Matroid Base". In: *arXiv preprint arXiv:2404.12121* (2024).

[PR26] Britta Peis and Niklas Rieken. "A simplified analysis of the ascending auction to sell a matroid base". In: *Operations Research Letters* 64 (2026), p. 107381.

[PU00] David C. Parkes and Lyle H. Ungar. "Iterative combinatorial auctions: Theory and practice". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Washington, DC 20004, USA: Association for the Advancement of Artificial Intelligence, 2000, pp. 74–82.

[PW20] Renato Paes Leme and Sam Chiu-wai Wong. "Computing Walrasian equilibria: Fast algorithms and structural properties". In: *Mathematical Programming* 179.1 (2020), pp. 343–384.

[Raa24] Stephen Raach. "Monotonicity of Extremal Walrasian Prices in Gross Substitutes Markets". In: *Operations Research Letters* (2024), p. 107149.

[RSÜ05] Alvin E. Roth, Tayfun Sönmez, and M. Utku Ünver. "Pairwise kidney exchange". In: *Journal of Economic theory* 125.2 (2005), pp. 151–188.

[RV25] Stephen Raach and Sven de Vries. "An Ascending Polynomial Running Time Vickrey Auction for Selling Bases of an Integer Polymatroid". In: *ACM Trans. Econ. Comput.* (Apr. 2025).

[Sch00] Alexander Schrijver. "A combinatorial algorithm minimizing submodular functions in strongly polynomial time". In: *Journal of Combinatorial Theory, Series B* 80.2 (2000), pp. 346–355.

[Sch03] Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*. Vol. 24. Springer, 2003.

[Sch80] Paul Schönsleben. "Ganzzahlige Polymatroid-Intersektions-Algorithmen". PhD thesis. ETH Zürich, 1980.

[Shi17] Akiyoshi Shioura. "Algorithms for L-convex function minimization: Connection between discrete convex analysis and other research fields". In: *Journal of the Operations Research Society of Japan* 60.3 (2017), pp. 216–243.

[ST15] Akiyoshi Shioura and Akihisa Tamura. "Gross substitutes condition and discrete concavity for multi-unit valuations: a survey". In: *Journal of the Operations Research Society of Japan* 58.1 (2015), pp. 61–103.

[SY09] Ning Sun and Zaifu Yang. "A double-track adjustment process for discrete markets with substitutes and complements". In: *Econometrica* 77.3 (2009), pp. 933–952.

[SY14] Ning Sun and Zaifu Yang. "An efficient and incentive compatible dynamic auction for multiple complements". In: *Journal of Political Economy* 122.2 (2014), pp. 422–466.

[SY92] J. George Shanthikumar and David D. Yao. "Multiclass queueing systems: Polymatroidal structure and optimal scheduling control". In: *Operations Research* 40.3-supplement-2 (1992), S293–S299.

[Ter23] Tatsuya Terao. "Faster Matroid Partition Algorithms". In: *arXiv preprint* (2023).

[TH02] David N. C. Tse and Stephen V. Hanly. "Multiaccess fading channels. I. Polymatroid structure, optimal resource allocation and throughput capacities". In: *IEEE Transactions on Information theory* 44.7 (2002), pp. 2796–2815.

[Var81] Hal R. Varian. "Dynamical systems with applications to economics". In: *Handbook of mathematical economics* 1 (1981), pp. 93–110.

[Vic61] William Vickrey. "Counterspeculation, auctions, and competitive sealed tenders". In: *The Journal of finance* 16.1 (1961), pp. 8–37.

[VSV07] Sven de Vries, James Schummer, and Rakesh V. Vohra. "On ascending Vickrey auctions for heterogeneous objects". In: *Journal of Economic Theory* 132.1 (2007), pp. 95–118.

[Wal74] Léon Walras. *Éléments d'économie politique pure ou théorie de la richesse sociale*. F. Rouge, 1874.

[Wel76] Dominic J.A. Welsh. *Matroid Theory*. London, New York, San Francisco: Academic Press, 1976, pp. xi+433.

[Whi35] Hassler Whitney. "On the Abstract Properties of Linear Dependence". In: *American Journal of Mathematics* 57.3 (1935), pp. 509–533.

[Wol76] Philip Wolfe. "Finding the nearest point in a polytope". In: *Mathematical Programming* 11 (1976), pp. 128–149.

[Wol82] Laurence A. Wolsey. "An analysis of the greedy algorithm for the submodular set covering problem". In: *Combinatorica* 2.4 (1982), pp. 385–393.

Colophon

This thesis was typeset with LaTeX 2ε. It uses the *OMS Thesis* style developed by Niklas Rieken. The development of the *OMS Thesis* style followed a roadmap that was heavily inspired by the Clean Thesis style by Ricardo Langner.