European Journal of Operational Research 325 (2025) 20–37

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/eor

Discrete optimization

Patient-to-room assignment with single-rooms entitlements: Combinatorial insights and integer programming formulations

Tabea Brandt *,[1], Christina Büsing *,[1,2], Felix Engelhardt *,[1]

Combinatorial Optimization, RWTH Aachen University, Ahornstraße 55, 52074 Aachen, Germany

ARTICLE INFO

Keywords:
Combinatorial optimisation
Hospital bed management
Patient-to-room assignment
Binary integer programming
Patient admission scheduling
Dynamic planning

ABSTRACT

Patient-to-room assignment (PRA) is a scheduling problem in decision support for hospitals. It consists of assigning patients to rooms during their stay at a hospital according to certain conditions and objectives, e.g., ensuring gender separated rooms, avoiding transfers and respecting single-room requests. This work presents combinatorial insights about the feasibility of PRA and about how (many) single-room requests can be respected. We further compare different integer programming (IP) formulations for PRA as well as the influence of different objectives on the runtime using real-world data. Based on these results, we develop a fast IP-based solution approach, which obtains high quality solutions. In contrast to previous IP-formulations, the results of our computational study indicate that large, real-world instances can be solved to a high degree of optimality within (fractions of) seconds. We support this result by a computational study using a large set of realistic but randomly generated instances with 50% to 95% capacity utilisation.

1. Introduction

Beds and rooms for patients are important resources in hospitals and the decision which bed and room a patient occupies impacts not only the staff's workload (Blay, Roche, Duffield and Robyn, 2017), but also patient satisfaction (He et al., 2018), and the provision of surcharges (Hendrich & Lee, 2005). The assignment of patients to beds and rooms is usually either performed by so-called case managers or by experienced nurses. In literature, the terms patient-to-room assignment problem (PRA), patient-to-bed assignment problem (PBA), and patient-admission scheduling (PAS) have been used to describe this task. In their original problem definition, Demeester et al. use the term patient-admission scheduling for the decision to which bed a patient is assigned (Demeester, Souffriau, Causmaecker, & Berghe, 2010). This term, however, can easily be confused with the task of scheduling the admission dates for planned inpatient treatment, which is a very different challenge (Schäfer, Walther, Hübner, & Kuhn, 2019). Overall, the term bed is used synonymously for bed space. In general, there are different bed types, e.g., for small children or heavy weight patients which are provided as rolling stock. A room's bed spaces, however, can be considered as equal. The task of finding a physical bed of appropriate bed type for a patient is independent of assigning the patient to a room/bed space and is not further considered in this paper. We therefore use the term PRA to avoid confusion.

Typically, there are two types of case management systems in hospitals: centralised and decentralised systems. In a centralised system, all patient-to-room assignments are decided by the same person or work group. Whereas in a decentralised system, the patient-to-room assignments are decided on ward or speciality level (Schmidt, Geisler, & Spreckelsen, 2013). In both cases, PRA is based on a previously fixed admission scheduling decision. In the first formal definition of PRA proposed by Demeester et al. in 2010 (Demeester et al., 2010), they considered a centralised system with multiple wards and specialities. Patients then need a room in a ward with appropriate speciality. This definition is still often used in literature. However, we found that in our local hospital a decentralised system is used. In this work, we present combinatorial insights and a solution approach for a decentralised system where patient-to-room assignments are decided on ward level. Combinatorially, the decentralised system is a special case of the centralised system.

Another characteristic of the definition proposed by Demeester et al. is that some patients may only be assigned to specific rooms to account

* Corresponding authors.
E-mail addresses: brandt@combi.rwth-aachen.de (T. Brandt), buesing@combi.rwth-aachen.de (C. Büsing), engelhardt@combi.rwth-aachen.de (F. Engelhardt).

[1] This work was supported by the Freigeist-Fellowship of the Volkswagen Stiftung and by the German research council (DFG) Research Training Group 2236 UnRAVeL.

[2] This work was partially supported by the German Federal Ministry of Education and Research (grant no. 05M16PAA) within the project "HealthFaCT - Health: Facility Location, Covering and Transport".

https://doi.org/10.1016/j.ejor.2025.02.018
Received 12 March 2024; Accepted 14 February 2025
Available online 3 March 2025

for, e.g., equipment requirements (Demeester et al., 2010). However, we found that in our local hospital all rooms have the same default equipment and the special equipment is rolling stock. Therefore, we assume that a ward's rooms are all equal and that every patient can be assigned to every room. We experience this to be a common setting in German hospitals.

Real-life optimisation problems often have to balance the, potentially conflicting, interests of multiple stakeholders. For PRA, Schäfer et al. identified patients, doctors and nurses as the main stakeholders (Schäfer et al., 2019). Based on literature and interviews with members of each of the groups, they concluded that patients primarily desire a pleasant stay, i.e., a bed in a suitable room, without unnecessary transfers or waiting in an overflow area, and suitable roommates. Doctors primarily look for visiting rounds that minimise walking distances. In comparison to that, nursing staff emphasise the relevance of a balanced workload (Schäfer et al., 2019). A common approach, also used by Schäfer et al. is to combine the objectives of all stakeholders into one objective function as a weighted sum. However, the appropriate choice of weights is not obvious and depends strongly on the hospital management's values. On the contrary, we consider only two objective functions and attempt a thorough investigation of their combinatorial structure, their performance in binary integer programs (BIPs) and their interoperability. For this, we consider the objectives both separately and in different hierarchical orders that are motivated by the different stakeholders' points of view.

Our first objective is to avoid that patients have to change rooms during their stay, so-called *patient transfers*. Patient transfers increase the staff's workload and reduce patient satisfaction while providing no immediate health benefits for patients (Storfjell, Ohlson, Omoike, Fitzpatrick, & Wetasin, 2009). A case study by Blay et al. reports that transfers require on average between11min (intra-ward transfer) and 25min (inter-ward transfer) of direct nursing time (Blay, Roche, Duffield, Robyn, 2017). Additionally, there are several ways (Fekieta et al., 2020) in which these transfers can put patients health at risk, e.g., by leading to delays in care (Johnson et al., 2013), interruptions in treatment (Papson, Russell, & Taylor, 2007) and increased infections (Blay, Roche, Duffield and Xu, 2017).

Our second objective is the assignment of single rooms to patients who need isolation for medical reasons or who are entitled to one because of a private health insurance (*private patients*). In practice, medical reasons have priority, but the latter case is also of high interest for the hospital management as such additional services provide income opportunities, with, e.g., a single room surcharge numbering 175€ per day (UKA Aachen, 2021). Due to current German laws, hospitals rely on these surcharges for income. In our computational study, we only model the assignment of single rooms to private patients because we lack data on which patients require isolation for medical reasons. Therefore, we only consider single-room requests depending on the patients' insurance types for our computational evaluations. However, all our results can easily be extended to incorporate medically necessary isolation either as an additional objective (analogously as for private patients but with higher priority), or as an additional feasibility condition.

In this work, we present combinatorial insights about checking an instance's feasibility and about computing the maximum number of fulfillable single-room requests. Additionally, we show that, in practice, the minimum total number of transfer can be computed efficiently using integer programming, although minimising the number transfers is $\mathcal{NP}$-hard in theory (Brandt, Büsing, & Knust, 2024). We further investigate the runtime of different BIP-formulations for solving PRA as well as the influence of the two objectives on the runtime using real-world data. We combine our insights into an efficient, IP-based heuristic. Here, our combinatorial insights not only improve the heuristic's runtime but also allow an assessment of the solution's quality. An extensive evaluation of our heuristic using both real-world data as well as a large set of artificial instances shows that the heuristic obtains high quality solution in most cases. The code of the implementation that we used for the computational study and the artificial instances are publicly available for further research (Brandt & Engelhardt, 2024).

This paper is organised as follows. In Section 2, we present a formal definition of PRA. In Section 3, we give an overview of existing research on integer programming in the context of PRA. In Section 4, we present combinatorial insights into both feasibility and the maximum number of private patients that can be assigned a single bed each day. Then, in Section 5 we propose and compare multiple IP formulations for PRA. The computational evaluation shows that in most our (real-life) instances, no transfers are necessary. Building on that, we propose and compare a second set of IP formulations that contain no transfers in Section 5.3. In Section 6, we combine the best performing IP formulations with our combinatorial insights from Section 4 to solve a dynamic version of PRA with a rolling-time-horizon approach. Although PRA is known to be $\mathcal{NP}$-hard (Brandt et al., 2024), we find solutions that are optimal or close to optimality for both heterogeneous real world data and artificial instances. Furthermore, on average, our algorithm requires less than a second per day to find high quality solutions for realistically sized artificial instances. Finally Section 7, we point out multiple directions for further research.

2. Problem definition

Formally, we consider a ward with rooms $\mathcal{R}$ and $c_r \in \mathbb{N}$ beds in room $r \in \mathcal{R}$, as well as a discrete planning horizon $\mathcal{T} = \{1, \dots, T\}$. In our computational study, we use 24h as length of one time period so that T refers to the number of days in the planning horizon. However, all concepts in this paper are easily transferable to half-day or even smaller planning intervals. Further, let $\mathcal{P}$ denote the set of all patients. For every patient $p \in \mathcal{P}$, we know their registration period and their arrival period $a_p \in \mathcal{T}_0 := \mathcal{T} \cup \{0\}$, where patients with $a_p = 0$ are patients who arrived in an earlier time period. Therefore, those patients already have pre-assigned rooms which are given in the set $\mathcal{F} \subset \{p \in \mathcal{P} \mid a_p = 0\} \times \mathcal{R}$. Commonly, patients whose arrival and registration periods are identical are called *emergency patients* and otherwise *elective patients* (Noonan, O'Brien, Broderick, Richardson, & Walsh, 2019). We further have every patient's discharge period $d_p \in \mathcal{T}$ satisfying $a_p < d_p$, their gender, and whether they are entitled to a single room. We denote the set of female patients with $\mathcal{P}^{\mathrm{f}} \subseteq \mathcal{P}$, the set of male patients with $\mathcal{P}^{\mathrm{m}} \subseteq \mathcal{P}$, and the set of patients entitled to a single room with $\mathcal{P}^{*} \subseteq \mathcal{P}$. Note that we assume $\mathcal{P}^{\mathrm{f}} \cap \mathcal{P}^{\mathrm{m}} = \emptyset$ and $\mathcal{P} = \mathcal{P}^{\mathrm{f}} \cup \mathcal{P}^{\mathrm{m}}$ based on the data provided by our local hospital.

The main task in PRA is to assign every patient $p \in \mathcal{P}$ to a room $z(p,t) \in \mathcal{R}$ for every time period $a_p \leq t < d_p$ of their stay. We assume that all patients stay in hospital on consecutive periods from admission to discharge period and that they are discharged at the beginning of a time period. Thus, patients do not need a room in their discharge period, which is a common assumption in literature, cf. Vancroonenburg, De Causmaecker, and Vanden Berghe (2016). We define the set of all patients who need a room in time period $t \in \mathcal{T}$ as $\mathcal{P}(t) = \{p \in \mathcal{P} \mid a_p \leq t < d_p\}$. Further, we denote for any subset of patients $S \subset \mathcal{P}$ the subset of patients in need for a bed in time period $t \in \mathcal{T}$ by $S(t) := S \cap \mathcal{P}(t)$.

The assignment z of patients to rooms has to fulfil two conditions for every room $r \in \mathcal{R}$ and every time period $t \in \mathcal{T}$ in order to be feasible:

(C) room capacities c_r are respected, i.e., $|\{p \in \mathcal{P} \mid z(p,t) = r\}| \leq c_r$,

(S) female and male patients are assigned to separate rooms, i.e.,

$$\{z(p,t) \mid p \in \mathcal{P}^{\mathrm{f}}(t)\} \cap \{z(p,t) \mid p \in \mathcal{P}^{\mathrm{m}}(t)\} = \emptyset.$$

Theoretically, these constraints may lead to infeasibility, which is unacceptable in practical application. However, we assume based on practitioners demands that the case manager ensures respect of the

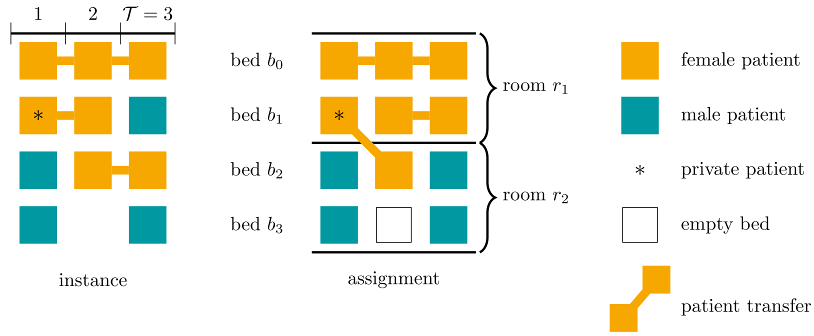


Fig. 1. Example for $\mathcal{T} = 3$ with one private patient where a patient transfer is necessary for feasibility.

ward's capacity under consideration of the gender-separation condition. Therefore, all considered instances in this paper are feasible under both conditions and . We discuss how feasibility under these constraints can be checked combinatorially in Section 4.1.

As first objective function, we minimise the total number of patient transfers

$$f^{\text{trans}} := \sum_{p\in\mathcal{P}} \left(\sum_{t=a_p}^{d_p-2} |\{z(p,t), z(p,t+1)\}| - 1 \right).$$

Another possibility of addressing the topic of patient transfers is to minimise the maximum number of transfers per patient. According to Brandt et al. those two interpretations of avoiding transfers are not conflicting but can be optimised simultaneously for the case of double rooms (Brandt et al., 2024). More precisely, they showed that there always exists an optimal solution with regard to f^{trans} where each patient is transferred at most once. Therefore, we choose the minimisation of the total number of transfers as objective function. Our computational experiments with real life data showed that in an optimal solution with respect to f^{trans} no patient is transferred twice regardless of an upper bound on the number of transfers per patient. Moreover, the enforcement of an upper bound on the number of transfers per patient did not affect the runtime. Therefore, we exclusively consider f^{trans} as objective function for transfers.

As second objective, we maximise the total number of time periods that private patients spend alone in a room, as, in Germany, the fees for a single room are paid by the insurance companies for every day individually. Thus, we maximise

$$f^{\text{priv}} := \sum_{t\in\mathcal{T}} \left(\sum_{p\in\mathcal{P}^*(t)} 1 - \min\{1, \; |\{q \in \mathcal{P}(t) \setminus \{p\} \mid z(p,t) = z(q,t)\}|\} \right).$$

Fig. 1 illustrates the role of both objectives. Note that, in general, transfers are necessary for feasibility in PRA. In the example, two rooms and a time horizon of three time steps are given. In step one, both rooms are assigned two male/female patients each. After the first step, the two male patients leave and a third female patient arrives. Now, as two more male patients arrive in the third step, a transfer is necessary to ensure feasibility. The example also includes a private patient which is marked by a $*$. Here, the private patient can be charged a single room surcharge only for the second time step. Hence, for the assignment depicted in the example we have $f^{\text{priv}} = f^{\text{trans}} = 1$ and this is optimal for both objectives.

3. Literature review

In 2010, Demeester et al. provided the first formal definition of a PRA problem and proposed a tabu-search algorithm for what they called the "problem of automatically and dynamically assigning patients to beds in a general hospital setting" (Demeester et al., 2010). According to their problem definition, patients have to be assigned to suitable rooms respecting numerous equipment, specialism, and age constraints. However, limiting a patient's room choices immediately renders the task of assigning patients to beds/rooms $\mathcal{NP}$-complete (Brandt et al., 2024). Ceschia and Schaerf extended the definition by Demeester et al. to include dynamic admission, operating room constraints, time horizons, and patient delays (Ceschia & Schaerf, 2009, 2011, 2012, 2014). The dynamic setting they consider is similar to our own, but the problem definition differs slightly, as their formulation is directly based on Demeester et al. (2010).

Next, we give an overview of published integer linear programming approaches in PRA. For a general literature overview of PRA we refer to the recent work by Brandt et al. on integrated planning (Br et al., 2023), and the review by Abdalkareem, Amir, Ehkan, and Al-Betar (2018). A frequent pattern in literature on PRA is to use a integer linear program (IP) to formalise the problem definition, but not to use integer programming as a solution method. This may be due to the fact that, in 2010, Demeester et al. considered integer programming as solution approach. However, the authors dismissed this, as the given formulation did not result in a feasible solution within an hour and even after a week of computation, no optimal solution was obtained using standard solver software (Demeester et al., 2010). Ceschia and Schaerf also used an exact solver based on IPs as a reference for small instances, while noting its inability to solve larger instances (Ceschia & Schaerf, 2012).

Nonetheless there are several publications that specifically make use of mixed-integer programming (MIP) based solution approaches: Schmidt et al. define a binary integer program (BIP) based on patients' length-of-stay (los) and use it to compare an exact approach, using the MIP solver SCIP, with three heuristic strategies (Schmidt et al., 2013). Range et al. reformulate Demeester et al.'s patient admission scheduling problem via Dantzig–Wolfe decomposition and apply a heuristic based on column generation to solve it (Range, Lusby, & Larsen, 2014). Turhan and Bilgen propose two MIP based heuristics which achieve high quality solutions in fast runtimes compared to respective state of the art studies (Turhan & Bilgen, 2016). Vancroonenburg et al. extend the patient assignment problem formulation and develop two corresponding online IP formulations. The first formulation focuses on newly arrived patients, whereas the second also considers planned future patients. They then study the effect of uncertainty in the patients' los, as well as the effect of the percentage of emergency patients. In all of the cases mentioned above, integer programming is used either as a basis for the development of heuristic solutions or as a reference for small instances, but no exact solving of larger real-world instances is attempted.

Some recent publications also employ integer programming to model both PRA and operating-room usage exactly (Conforti, Guido, Mirabelli, & Solina, 2018a, 2018b). However, the models include significant simplifications: fixed room-gender assignment, no transfers and a limited time-horizon.

Guido, Groccia, and Conforti (2018) compare different IP formulations for PRA. The authors propose iteratively extending the (PRA) search space, starting with small formulation and extending it if optimality is not reached. The same idea underlies our sequential approach for the dynamic PRA, which we improve using combinatorial insights. Bastos, Marchesi, Hamacher, and Fleck (2019) present a MIP approach to patient admission scheduling problem, which involves assigning patients to beds over a given time horizon so as to maximise treatment efficiency, patient comfort and hospital utilisation, while satisfying all necessary medical constraints and taking into consideration patient preferences as much as possible.

In a recent paper, Liu, Wang, and Hao (2024) revisited the original MIP formulation by Ceschia and Schaerf (2011). The authors then decompose room and bed assignment, and use constraint aggregation to reduce the size of the IP formulation. This leads to improvements on a range of reference instances. Their aggregation of patient transfers is similar to the model used in this work, without a dynamic model and/or same-day transfers. The gender aggregation they perform is similar to the aggregations we perform, although we test multiple variants thereof. Additionally, they extensively reviewed (heuristic) literature on PRA/PBA.

There also exist solution approaches that are not based on Demeester's problem definition but are inspired by the setting in a specific hospital. Thomas et al. developed a MIP based decision support system that balances 13 objectives (T et al., 2013). Schäfer et al. disallow (non-medically induced) patient transfers but include overflow and patient preferences (Schäfer et al., 2019). They also model doctor preferences, i.e., homogeneous routes, and then solve the model via a greedy look-ahead heuristic. In a follow-up publication, they focus on emergency patients and integrate them into the model (Schäfer, Walther, Grimm, & Hübner, 2023). Brandt et al. propose a MIP based heuristic for integrated planning of patient-to-room and nurse-to-patient assignment (Br et al., 2023). More generally, Rachuba et al. introduce a taxonomy for evaluating integration consisting of three stages: linkage by constraints/restrictions, sequential and completely integrated planning (Rachuba, Reuter-Oppermann, & Thielen, 2023). Here, our work can contribute to multiple levels, with the combinatorial insights facilitating easy linkage by constraints/restrictions and the IP based approach being suitable for fully integrated planning.

Combinatorial insights about patient-to-room assignment and its underlying structure are still rare. For the definition proposed by Demeester et al. it was proven that it is $\mathcal{NP}$-complete to decide whether a feasible solution exists even if all rooms have capacity 2 (Ficker, Spieksma, & Woeginger, 2021; Vancroonenburg, Croce, Goossens, & Spieksma, 2014). However, for this result it is important that not every patient can be assigned to every room. Therefore, in our case (without patient-room restrictions), this result is not applicable. In fact, in our problem setting, we can efficiently decide an instance's feasibility if all rooms have the same capacity while the decision remains $\mathcal{NP}$-complete for arbitrary room capacities (Brandt et al., 2024). We complement these results by providing an efficient way to decide feasibility for instances with single and double rooms, and for computing the maximum number of fulfillable single-room requests. Remark that deciding whether a solution is feasible without transfers, however, $\mathcal{NP}$-complete even for instances with only double rooms if there are at least three rooms (Brandt et al., 2024).

4. Combinatorial insights

In this section, we first present new combinatorial insights regarding the feasibility of instances with single and double rooms which extend the known results on feasibility from Brandt et al. (2024). Second, we present a combinatorial formula to compute the maximum number of private patients who can be feasibly assigned to single rooms. Both questions can be decided independently for every single time period, since we allow arbitrary many transfers. Therefore, in this section we consider an arbitrary but fixed time period $t \in \mathcal{T}$ and abbreviate the number of female patients who are in hospital in time period t with $F_t := |\mathcal{P}^{\mathrm{f}}(t)|$, and respectively the number of male patients, female private patients, and male private patients needing a bed in time period t with $M_t := |\mathcal{P}^{\mathrm{m}}(t)|$, $F_t^* := |\mathcal{P}^{\mathrm{f}}(t) \cap \mathcal{P}^*(t)|$, and $M_t^* := |\mathcal{P}^{\mathrm{m}}(t) \cap \mathcal{P}^*(t)|$. We further denote with $R_c := |\{r \in \mathcal{R} \mid c_r = c\}|$ the number of rooms with a specific capacity $c \in \mathbb{N}$.

4.1. Feasibility

Brandt et al. define the feasibility problem for an arbitrary but fixed time period $t \in \mathcal{T}$ as follows Brandt et al. (2024).

Definition 1 (*Feasibility Problem*)**.** Given the number of female and male patients $F_t, M_t \in \mathbb{N}_0$, and room capacities $c_r \in \mathbb{N}$ for $r \in \mathcal{R}$, does there exist a subset $S \subseteq \mathcal{R}$ of rooms such that it can host all female patients while the male patients fit into the remaining rooms, i.e.,

$$\sum_{r \in S} c_r \geq F_t \quad \text{and} \sum_{r \in \mathcal{R} \setminus S} c_r \geq M_t? \tag{1}$$

Brandt et al. prove that the feasibility problem is $\mathcal{NP}$-complete in general and solvable in polynomial time for constant room capacities $c_r = c \in \mathbb{N}$ (Brandt et al., 2024). Clearly, in the common case of rooms with only double rooms an instance is feasible if and only if

$$\left\lceil \frac{F_t}{2} \right\rceil + \left\lceil \frac{M_t}{2} \right\rceil \leq |\mathcal{R}| \tag{2}$$

holds true for every time period $t \in \mathcal{T}$ (Brandt et al., 2024). However, this is no longer accurate for wards that have at least one single room in addition to double rooms otherwise. For those, it suffices to check whether enough beds are available in total.

Lemma 1. *Consider a ward with room capacities $c_r \in \{1, c\}$ with $c \in \mathbb{N}$ for all rooms $r \in \mathcal{R}$. Let the number of female and male patients $F_t, M_t \in \mathbb{N}_0$, be given. If $R_1 \geq c - 1$, then the instance is feasible if and only if the number of patients does not exceed the ward's total capacity, i.e., if and only if*

$$F_t + M_t \leq \sum_{r \in \mathcal{R}} c_r \tag{3}$$

holds true for every time period $t \in \mathcal{T}$.

Proof. For $c = 1$, the instance is obviously feasible if and only if Eq. (3) holds true. Therefore, let $c \geq 2$. If the number of patients exceeds ward's capacity, i.e., Eq. (3) is violated, then the instance is infeasible as at least one patient cannot be assigned to a room without violating the capacity constraint. Hence, we assume Eq. (3) to hold true and show that the instance is then feasible by constructing a set $S \subseteq \mathcal{R}$ which satisfies Eq. (1).

We compute the maximum number k of rooms with capacity c we could completely fill with female patients

$$k := \min \left\{ \left\lfloor \frac{F_t}{c} \right\rfloor, R_c \right\},$$

and, respectively, we compute the maximum the number ℓ of rooms with capacity c that we could completely fill with male patients

$$\ell := \min \left\{ \left\lfloor \frac{M_t}{c} \right\rfloor, R_c - k \right\}.$$

Remark that $R_c \geq k+\ell$, $F_t - ck \geq 0$, and $M_t - c\ell \geq 0$ by construction. On the one hand, if $R_1 \geq F_t - ck + M_t - c\ell$, i.e., all remaining patients can be assigned to single rooms, then we define $S := S' \cup S''$ with arbitrary sets S' and S'' that fulfil the following condition

$$S' \subseteq \{r \in \mathcal{R} \mid c_r = c\} \quad \text{with } |S'| = k,$$
$$S'' \subseteq \{r \in \mathcal{R} \mid c_r = 1\} \quad \text{with } |S''| = F_t - ck.$$

Then,

$$\sum_{r \in S} c_r = \sum_{r \in S'} c_r + \sum_{r \in S''} c_r = ck + F_t - ck = F_t,$$

$$\sum_{r\in\mathcal{R}\setminus S} c_r = \sum_{r\in\mathcal{R}} c_r - \sum_{r\in S} c_r \overset{\text{Eq. (3)}}{\geq} F_t + M_t - F_t = M_t,$$

i.e., the feasibility condition Eq. (1) is satisfied and the instance is feasible.

On the other hand, if $R_1 < F_t - ck + M_t - c\ell$, then

$$\begin{aligned} R_c &\overset{(1)}{\geq} \frac{1}{c}(F_t + M_t - R_1) \\ &> \frac{1}{c}(F_t + M_t - F_t + ck - M_t + c\ell) = k + \ell. \end{aligned}$$

Therefore, we also have $F_t - ck < c$ and $M_t - c\ell < c$ and we define $S \subseteq \{r \in \mathcal{R} \mid c_r = c\}$ arbitrary with $|S| = k+1$. Then

$$\begin{aligned} \sum_{r\in S} c_r &= ck + c > ck + F_t - ck = F_t, \\ \sum_{r\in\mathcal{R}\setminus S} c_r &= R_1 + c(R_c - k - 1) \\ &\overset{R_c \geq k+\ell+1}{\geq} R_1 + c(k+\ell+1-k-1) = R_1 + c\ell \\ &\overset{R_1\geq c-1}{\geq} c - 1 + c\ell \overset{c-1\geq M_t - c\ell}{\geq} M_t - c\ell + c\ell \geq M_t, \end{aligned}$$

i.e., the feasibility condition Eq. (1) is satisfied and the instance is feasible. □

Remark that the condition $R_1 \geq c-1$ in Lemma 1 is tight: let $c = 3$ and $R_1 < c-1$, i.e., let $R_1 = R_3 = 1$. Then an instance with $F_t = M_t = 2$ satisfies Eq. (3), however, there exists no feasible solution.

Lemma 1 covers especially the case of wards with single and double rooms only. Additionally, we can use it to derive a similar result for wards with rooms of even capacity.

Lemma 2. *Consider a ward with room capacities $c_r \in \{2, 2c\}$ with $c \in \mathbb{N}_{\geq 2}$ for all rooms $r \in \mathcal{R}$. Let the number of female and male patients $F_t, M_t \in \mathbb{N}_0$, be given. If $R_2 \geq c-1$, then the instance is feasible if and only if for every time period $t \in \mathcal{T}$ one of the two following conditions holds true.*

1. *F_t and M_t are both even and the number of patients does not exceed the ward's total capacity, i.e., $F_t + M_t \leq \sum_{r\in\mathcal{R}} c_r$*
2. *the number of patients is strictly smaller than the ward's total capacity, i.e., $F_t + M_t < \sum_{r\in\mathcal{R}} c_r$*

Proof. First, assume both conditions are violated, then we have $F_t + M_t = \sum_{r\in\mathcal{R}} c_r$, and both F_t and M_t are odd. Then, no feasible assignment of patients to rooms exists. Second, assume Condition 1 holds true. Then, we construct an equivalent instance by dividing all c_r, F_t, and M_t by 2. This instance is feasible according to Lemma 1 and hence also the original one.

Third, assume Condition 2 holds true and Condition 1 does not. Thus, we have $F_t + M_t < \sum_{r\in\mathcal{R}} c_r$ and either both F_t and M_t are odd or exactly one of them. If both F_t and M_t are odd, it directly follows that $F_t + M_t \leq \sum_{r\in\mathcal{R}} c_r - 2$. We then construct an equivalent instance by increasing the number of female and male patients each by one, i.e., $F_t' := F_t + 1$ and $M_t' := M_t + 1$. Then F_t' and M_t' are both even with $F_t' + M_t' \leq \sum_{r\in\mathcal{R}} c_r$. This new instance is feasible according to Condition 1 and hence also the original one. We proceed analogously if either F_t or M_t is odd. □

4.2. Maximum number of private patients in single rooms

We define the problem of computing the maximum number $s_t^{\max}$ of private patients who can get a room for themselves in time period t as follows.

Definition 2 (*Private Patient Problem (PPP)*). Let the total number of female and male patients $F_t, M_t \in \mathbb{N}_0$, the number of female and male private patients $F_t^*, M_t^* \in \mathbb{N}_0$, and room capacities $c_r \in \mathbb{N}$ for $r \in \mathcal{R}$ be given. Do there exist four pairwise disjoint subsets $S_F \cup S_F^* \cup S_M \cup S_M^* \subseteq \mathcal{R}$ such that

1. all female patients are assigned to rooms $S_F \cup S_F^*$, and all patients assigned to rooms in S_F^* are private patients and alone in their rooms, i.e.,
$$\sum_{r\in S_F} c_r + |S_F^*| \geq F_t \quad \text{and} \quad |S_F^*| \leq F_t^*, \tag{4}$$
2. all male patients are assigned to rooms $S_M \cup S_M^*$, and all patients assigned to rooms in S_M^* are private patients and alone in their rooms, i.e.,
$$\sum_{r\in S_M} c_r + |S_M^*| \geq M_t \quad \text{and} \quad |S_M^*| \leq M_t^*, \tag{5}$$
3. the number of private patients who have a room to themselves is maximal, i.e.,
$$s_t^{\max} := |S_F^*| + |S_M^*| \quad \text{is maximal.} \tag{6}$$

We first take a look at the complexity of PPP.

Lemma 3. *PPP is $\mathcal{NP}$-hard and not approximable.*

Proof. For $F_t^* = M_t^* = 0$, PPP is equivalent to the feasibility problem. Hence, also PPP is $\mathcal{NP}$-hard. Since the objective value $s_t^{\max}$ is 0 in this case, PPP is not approximable. □

However, PPP can be solved in polynomial time if the ward has only single and double rooms.

Lemma 4. *For feasible instances with $c_r \in \{1, 2\}$, PPP can be solved in polynomial time. Moreover, we can compute $s_t^{\max}$ as follows. Let*

$$\begin{aligned} \alpha_t &:= |\mathcal{R}| - \left\lceil \frac{F_t - F_t^*}{2} \right\rceil - \left\lceil \frac{M_t - M_t^*}{2} \right\rceil, \\ \beta_t^{\mathrm{f}} &:= \min\left\{ \left(F_t - F_t^*\right) \mod 2,\ F_t^* \right\} \in \{0,1\}, \text{ and} \\ \beta_t^{\mathrm{m}} &:= \min\left\{ \left(M_t - M_t^*\right) \mod 2,\ M_t^* \right\} \in \{0,1\}. \end{aligned}$$

Then

$$s_t^{\max} = \begin{cases} |\mathcal{P}^*(t)| & \text{if } \alpha_t \geq |\mathcal{P}^*(t)|, \\ |\mathcal{P}^*(t)| - 1 & \text{if } \alpha_t = |\mathcal{P}^*(t)| - 1 \text{ and } \beta_t^{\mathrm{f}} = \beta_t^{\mathrm{m}} = 1, \\ 2\alpha_t + \beta_t^{\mathrm{f}} + \beta_t^{\mathrm{m}} - |\mathcal{P}^*(t)| & \text{otherwise.} \end{cases}$$

Proof. For feasible instances with single and double rooms, we can treat all single rooms as double rooms since this does not affect the number of private patients who can get a room for themselves. Therefore, let us consider a feasible instance of PPP with only double rooms, i.e., Eq. (2) holds true. We now have to assign at least

$$\left\lceil \frac{F_t - F_t^*}{2} \right\rceil + \left\lceil \frac{M_t - M_t^*}{2} \right\rceil$$

rooms to non-private patients. Since we aim to maximise the number of private patients who are alone in a room, we assign exactly that many rooms to non-private patients. If the number of free remaining rooms α_t is greater or equal to the number of unassigned private patients, i.e., $\alpha_t \geq |\mathcal{P}^*(t)|$, then every private patient can get a room for themselves, i.e.,

$$s_t^{\max} = |\mathcal{P}^*|.$$

Otherwise, after assigning all non-private patients, we have α_t empty double rooms as well as potentially one (β_t^{f}) free bed in a double-bed room where a non-private female patient is present which we can assign to a private female patient (if at least one is present), respectively for male patients (β_t^{m}). This results in a total of

$$\gamma_t := 2\alpha_t + \beta_t^{\mathrm{f}} + \beta_t^{\mathrm{m}}$$

available beds for private patients. If $\beta_t^{\mathrm{f}} = 0$ or $\beta_t^{\mathrm{m}} = 0$ or $\alpha_t \leq |\mathcal{P}^*(t)| - 2$, then the difference of β_t and the total number of private patients gives us the of number of empty beds. Then, all private patients who can get a free room will do so, i.e.,

$$s_t^{\max} = \gamma_t - |\mathcal{P}^*(t)|.$$

However, if both $\beta_t^{\mathrm{f}} = 1$ and $\beta_t^{\mathrm{m}} = 1$ and exactly $|\mathcal{P}^*(t)| = \alpha_t + 1$ private patients need a room, then exactly one private patient will be placed in a room together with a non-private patient, i.e.,

$$s_t^{\max} = |\mathcal{P}^*(t)| - 1.$$

Overall, we achieve the stated formula for computing $s_t^{\max}$. □

Knowing the maximum number $s_t^{\max}$ of private patients who can get a single room in time period t allows us to assess the trade-off between f^{priv} and f^{trans} or other objectives that occur in practice, e.g., hosting all patients who need immediate care. Using the exact computation of $s_t^{\max}$, we know that their sum over all time periods $t \in \mathcal{T}$ is a tight upper bound on the total objective value for f^{priv}, i.e.,

$$f^{\mathrm{priv}} \leq s^{\max} := \sum_{t\in\mathcal{T}} s_t^{\max}. \tag{7}$$

This bound can always be achieved as long as arbitrary many transfers may be used.

5. Comparison of different IP-formulations

In this section, we propose and compare different IP-formulations for PRA. The most performant IP-formulations then constitute the basis for the IP-based heuristic for PRA that we propose in Section 6. To reduce the total number of IP variants that we compare, we first evaluate different formulations for minimising the total number of transfers only in Section 5.1. Second, we use the best performing LP formulation for minimising transfers and then compare different extensions for incorporating single-room requests of private patients in Section 5.2. Third, we compare different IP-formulations for maximising f^{priv} without using transfers in Section 5.3. Since we present multiple formulations for some of the conditions, we explain every constraint individually and then state for every IP which of the constraints are used.

Independently of our work, Liu et al. also worked on improvements of IP-formulations for PRA as defined by Demeester et al. (under the term patient admission scheduling) (Liu et al., 2024). However, there are major differences between their work and ours. On the one hand, their objective function differs as they use the one proposed by Ceschia and Schaerf (2011), which balances 9 criteria for a good patient-room assignment. In contrast, we focus on 2 objective functions and optimise them both exclusively and in different hierarchical orders. This allows us to visualise the objective's influence on the runtime and make justified decisions regarding the trade-off between runtime and the objective's real-world priority in the design of our algorithm, cf. Section 6.

On the other hand, Liu et al. also use a slightly different set of constraints as gender separation may be violated in their setting and single-room requests only count as fulfilled if the patient is assigned to a room that contains exactly one bed. In contrast, gender separation is mandatory in our setting and we count a single-room request as fulfilled if the patient is alone in a room regardless of the room's capacity. Hence, the results of Liu et al. do not directly translate to our setting. Nevertheless, some of the constraint aggregations used by Liu et al. are similar to the ones we use in Section 5.1. We therefore report the similarities there in detail.

5.1. Minimise transfers only

As a first step, we propose and compare different IP-formulations for PRA minimising transfers only. In the next section, we then extend the best performing IP-formulation to incorporate single-room requests. To model the assignment of patients to rooms as well as the minimisation of transfers, we use the following binary variables:

$$x_{prt} = \begin{cases} 1, & \text{if patient } p \text{ is assigned to room } r \text{ in time period } t, \\ 0, & \text{otherwise,} \end{cases} \tag{8}$$

$$\delta_{prt} = \begin{cases} 1, & \text{if patient } p \text{ is transferred from room } r \text{ to another room} \\ & \text{after time period } t \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

We then model the total number of transfers as the sum of all variables δ together with all altered pre-fixed assignments

$$f^{\mathrm{trans}} = \sum_{t\in\mathcal{T}} \sum_{p\in\mathcal{P}(t)} \sum_{r\in\mathcal{R}} \delta_{prt} + |\mathcal{F}| - \sum_{(r,p)\in\mathcal{F}} x_{pr1}. \tag{10}$$

Regarding the constraints, we first ensure that all patients are assigned to rooms for every time period of their stay:

$$\sum_{r\in\mathcal{R}} x_{prt} = 1 \quad \forall t \in \mathcal{T}, p \in \mathcal{P}(t). \tag{11}$$

Second, we ensure that the room capacity is respected via

$$\sum_{p\in\mathcal{P}(t)} x_{prt} \leq c_r \quad \forall t \in \mathcal{T}, r \in \mathcal{R}. \tag{12}$$

Third, to model gender separation, we introduce two additional sets of binary variables

$$g_{rt} = \begin{cases} 1, & \text{if there is a female patient assigned to room } r \text{ in time} \\ & \text{period } t, \\ 0, & \text{otherwise,} \end{cases} \tag{13}$$

$$m_{rt} = \begin{cases} 1, & \text{if there is a male patient assigned to room } r \text{ in time} \\ & \text{period } t, \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

We then ensure gender separation via

$$x_{prt} \leq g_{rt} \qquad \forall t \in \mathcal{T},\ p \in \mathcal{P}^{\mathrm{f}}(t),\ r \in \mathcal{R}, \tag{15}$$

$$x_{prt} \leq m_{rt} \qquad \forall t \in \mathcal{T},\ p \in \mathcal{P}^{\mathrm{m}}(t),\ r \in \mathcal{R}, \tag{16}$$

$$g_{rt} + m_{rt} \leq 1 \qquad \forall t \in \mathcal{T},\ r \in \mathcal{R}. \tag{17}$$

Using $m_{rt} \leq 1 - g_{rt}$ we can remove variable m_{rt} and replace constraints Eqs. (16) and (17) with

$$x_{prt} \leq 1 - g_{rt} \quad \forall t \in \mathcal{T},\ p \in \mathcal{P}^{\mathrm{m}}(t),\ r \in \mathcal{R}. \tag{18}$$

Instead of modelling capacity and gender separation constraints separately, we can also combine them and use

$$\sum_{p\in\mathcal{P}^{\mathrm{f}}(t)} x_{prt} \leq c_r g_{rt} \qquad \forall t \in \mathcal{T},\ r \in \mathcal{R}, \tag{19}$$

$$\sum_{p\in\mathcal{P}^{\mathrm{m}}(t)} x_{prt} \leq c_r m_{rt} \qquad \forall t \in \mathcal{T},\ r \in \mathcal{R}, \tag{20}$$

instead of Eqs. (12), (15) and (16). Or, if we omit variable m_{rt}, we use

$$\sum_{p\in\mathcal{P}^{\mathrm{m}}(t)} x_{prt} \leq c_r(1 - g_{rt}) \quad \forall t \in \mathcal{T},\ r \in \mathcal{R}, \tag{21}$$

instead of Eqs. (17) and (20). Fourth, we count the transfers via

$$x_{prt} - x_{pr(t+1)} \leq \delta_{prt} \quad \forall r \in \mathcal{R},\ p \in \mathcal{P}, a_p \leq t < d_p - 1. \tag{22}$$

We compare the performance of the following four IP-formulations to investigate the usage of variables m_{rt}, as well as the integration of capacity and gender-separation constraints.

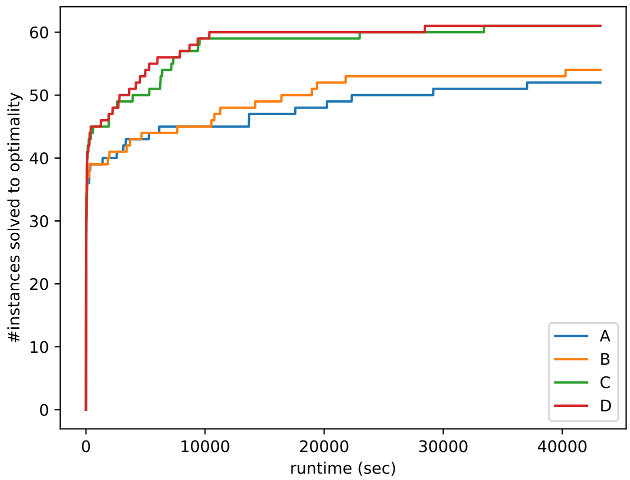


Fig. 2. Comparison of IPs (A)–(D) using 62 real-life instances, 61 instances were solved to optimality after 12 h by IPs (C) and (D) with objective value 0.

(A) min f^{trans} s.t. Eqs. (11) and (12), (15) to (17) and (22)
(B) min f^{trans} s.t. Eqs. (11), (12), (15), (18) and (22)
(C) min f^{trans} s.t. Eqs. (11), (17), (19), (20) and (22)
(D) min f^{trans} s.t. Eqs. (11), (19), (21) and (22)

Liu et al. use similar constraint aggregations and the constraint set of our IP (A) resembles the constraint set of their IP $\text{APRA}_{\text{GC}_0\&\text{TC}}$, respectively for our IP (B) and their IP $\text{APRA}_{\text{GC}_1\&\text{TC}}$, our IP (C) and their IP $\text{APRA}_{\text{AGC}_0\&\text{TC}}$, as well as for our IP (D) and their IP $\text{APRA}_{\text{AGC}_1\&\text{TC}}$ (Liu et al., 2024). In their setting, the most aggregated IP $\text{APRA}_{\text{AGC}_1\&\text{TC}}$ performs better than the others.

All our IPs were implemented in *python* 3.10.4 and solved using *Gurobi 10.0.0*. All simulations were done on the RWTH High Performance Computing Cluster using CLAIX-2018-MPI with Intel Xeon Platinum 8160 Processors "SkyLake" (2.1 GHz, 32 CPUs per task, 3.9 GB per CPU). The code is publicly available on GitHub (Brandt & Engelhardt, 2024).

For testing, we used 62 real-world instances provided by the *RWTH Aachen University Hospital (UKA)*, each spanning a whole year, and a time limit of 12 h. We performed consistency checks on the patient data ensuring valid input data: patients with missing information on arrival or discharge and patients with $a_p = d_p$ were dropped from the data and for patients whose registration was noted after their arrival, we set the registration date to the arrival date. All instances together still contain more than 53.000 patient stays. For every instance, the number of rooms and their capacities are given as well as the patients' arrival, departure, and registration dates, their gender, unique Patient-ID and information on the insurance status. Note that the real-world data is subject to non-disclosure and as such is not provided together with the code.

The results of comparing "transfers only" formulations are depicted in Fig. 2. They show that the integration of capacity and gender-separation constraints decreases computation time. Similarly, removing the variable m_{rt} also decreases computation time, which is in line with the results obtained by Liu et al. (2024). In general, instances were either solved to optimality with objective value 0 or resulted in a MIPGap of 100% after 12 h.

5.2. Integration of single-room constraints

In literature, single-room requests are often modelled as part of the objective function counting the patients (with such requests) who are assigned to rooms with capacity $c_r = 1$, see for example Demeester et al. (2010) and Liu et al. (2024). Schäfer et al. choose a different approach by counting single-room requests also as fulfilled if patients are alone in a room with higher capacity (Schäfer et al., 2019). This is similar to our problem definition. In their setting, however, those constraints are mandatory whereas, in our setting, we treat them as an objective function. We are not aware of other published papers that report IP-formulations fitting our setting.

In the last section, we identified IP (D) as the best performing IP formulation for PRA with the only objective function of minimising transfers. In this section, we extend IP (D) by integrating single-room requests. Instead of combining both objectives into one objective via a weighted sum approach, we optimise them hierarchically. This means that first one objective is (exclusively) optimised and under all solutions that are optimal w.r.t. this first objective, we search the solution that obtains the best value for the second objective. With a total of two objectives, this approach leads to two possible hierarchical orders. In our computational study, we compare two different levels of constraint aggregation and both hierarchical orders.

To incorporate single-room request into our IP (D), we define binary variables encoding whether a private patient gets a single room via

$$s_{prt} = \begin{cases} 1, & \text{if } p \text{ is alone in room } r \text{ in time period } t, \\ 0, & \text{otherwise.} \end{cases} \tag{23}$$

Thus, the total number of time periods that private patients are assigned to single rooms is given by

$$f^{\text{priv}} = \sum_{t\in\mathcal{T}} \sum_{p\in\mathcal{P}^*(t)} \sum_{r\in\mathcal{R}} s_{prt}. \tag{24}$$

Then, we can model the single-room constraints via

$$s_{prt} \le x_{prt} \qquad \forall t \in \mathcal{T},\ p \in \mathcal{P}^*(t),\ r \in \mathcal{R}, \tag{25}$$

$$c_r s_{prt} + \sum_{q\in\mathcal{P}(t)\setminus\{p\}} x_{qrt} \le c_r \qquad \forall t \in \mathcal{T},\ p \in \mathcal{P}^*(t),\ r \in \mathcal{R}. \tag{26}$$

Alternatively to Eq. (26), we can also integrate the single-room constraints with the gender-separation and capacity constraints Eqs. (19) and (20) via

$$\sum_{p\in\mathcal{P}^{\text{f}}(t)} x_{prt} + \sum_{p\in\mathcal{P}^{\text{f}}\cap\mathcal{P}^*(t)} (c_r - 1)s_{prt} \le c_r g_{rt} \qquad \forall t \in \mathcal{T},\ r \in \mathcal{R} \tag{27}$$

$$\sum_{p\in\mathcal{P}^{\text{m}}(t)} x_{prt} + \sum_{p\in\mathcal{P}^{\text{m}}\cap\mathcal{P}^*(t)} (c_r - 1)s_{prt} \le c_r(1 - g_{rt}) \qquad \forall t \in \mathcal{T},\ r \in \mathcal{R}. \tag{28}$$

We compare the performance of the following LP-formulations that integrate single-room requests based on the previous results

(E) max $(-f^{\text{trans}}, f^{\text{priv}})$ s.t. constraints of (D), Eqs. (25) and (26)
(F) max $(f^{\text{priv}}, -f^{\text{trans}})$ s.t. constraints of (D), Eqs. (25) and (26)
(H) max $(-f^{\text{trans}}, f^{\text{priv}})$ s.t. Eqs. (11), (22), (25), (27) and (28)
(I) max $(f^{\text{priv}}, -f^{\text{trans}})$ s.t. Eqs. (11), (22), (25), (27) and (28).

The objectives' order determines their priority in optimisation, i.e., max $(-f^{\text{trans}}, f^{\text{priv}})$ means that first f^{trans} is minimised and then f^{priv} is maximised.

The formulations for IPs (E) to (I) were evaluated on the same computational setup as in . The results are given in Fig. 3. We see that the decisive factor is not the set of constraints but the objective function. Minimising the number of transfers first, i.e., $\max(-f^{\text{trans}}, f^{\text{priv}})$ performs significantly better than maximising the fulfilled single-room requests first. However, it is noticeable that the second set of constraints performs overall better than the first set. We further observed that, when optimising f^{priv}, the solver frequently finds an optimal solution quickly but then requires extended time to prove optimality. Therefore, we use our combinatorial insights to help the solver prove optimality in this case.

If maximising f^{priv} has highest priority, we can use the combinatorial insights from Section 4 and fix the number of private patients in single rooms for time period t to $s_t^{\max}$, i.e.,

$$\sum_{p\in\mathcal{P}^*(t)} \sum_{r\in\mathcal{R}} s_{prt} \ge s_t^{\max} \quad \forall t \in \mathcal{T} \tag{29}$$

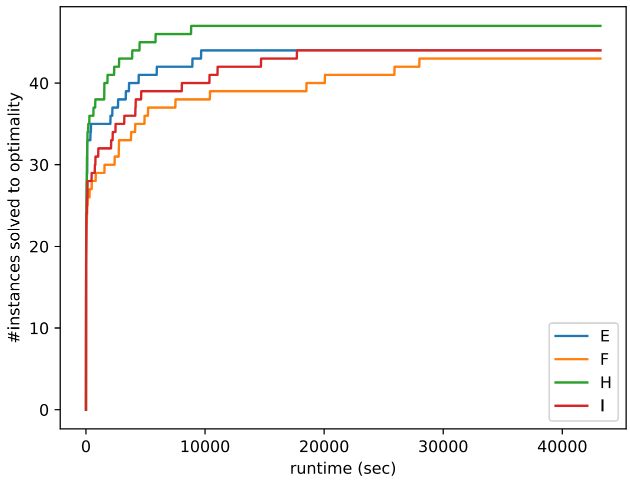


Fig. 3. Comparison of IPs E - H using 62 real-life instances, maximum runtime 12 h.

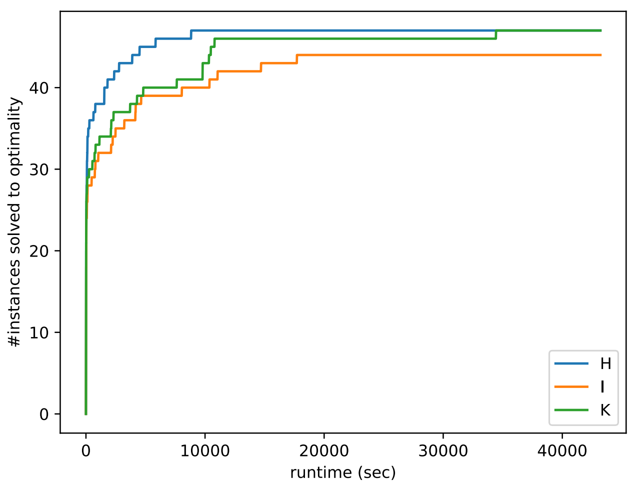


Fig. 4. Performance of IP K using 62 real-life instances, maximum runtime 12 h.

instead of using the bi-objective approach. Hence, we also evaluate the resulting IP

(K) $\min f^{\text{trans}}$ s.t. constraints of (H). Eq. (29)

We compare the respective IP's performance to the one of (H) and (I). Fig. 4 shows that IP (K) clearly outperforms IP (I), however, its performance is not as good as the one of IP (H). The implementation is available on GitHub (Brandt & Engelhardt, 2024).

5.3. IP-formulation without transfers

The objective values computed in our computational experiments in Sections 5.1 and 5.2 showed that in many instances no transfers are necessary throughout the entire planning period of one year. Therefore, we propose and compare in this section IP formulations where transfers are prohibited by construction while f^{priv} is maximised.

Liu et al. also report on an IP-formulation without transfers and argue it speeds up the computations by reducing the search scope (Liu et al., 2024). In this setting, they evaluate the use of variable m_{rt} and conclude that it is better to use the substitution with $(1 - g_{rt})$. Based on our observation of the significant performance advantage of IP-formulations without variable m_{rt} that we observed in Section 5.1, we omit the evaluation of the use of variable m_{rt} in this section which is backed up by the result of Liu et al. Instead, we focus on three different levels of constraint aggregation of capacity, gender-separation, and single-room constraints under the objective of maximising fulfilment of single-room requests. Furthermore, we evaluate whether it is faster to solve the optimisation problem with objective function f^{priv} or the feasibility problem where $f^{\text{priv}} = s^{\max}$ is fixed.

We use binary variables

$$x_{pr} = \begin{cases} 1, & \text{if patient } p \text{ is assigned to room } r \text{ for their stay,} \\ 0, & \text{otherwise,} \end{cases} \tag{30}$$

to model the assignment of patients to rooms together with the previously introduced variables s_{prt} as in (23), and variables g_{rt} as in (13).

Regarding the constraints, we first ensure that all patients are assigned to rooms in every time period of their stay:

$$\sum_{r \in R} x_{pr} = 1 \quad \forall p \in \mathcal{P}. \tag{31}$$

Second, we ensure that the room capacity is respected via

$$\sum_{p \in \mathcal{P}(t)} x_{pr} \le c_r \quad \forall t \in \mathcal{T}, r \in \mathcal{R}. \tag{32}$$

Third, we ensure gender separation via

$$x_{pr} \le g_{rt} \qquad \forall t \in \mathcal{T},\ p \in \mathcal{P}^{\mathrm{f}}(t),\ r \in \mathcal{R}, \tag{33}$$

$$x_{pr} \le (1 - g_{rt}) \qquad \forall t \in \mathcal{T},\ p \in \mathcal{P}^{\mathrm{m}}(t),\ r \in \mathcal{R}. \tag{34}$$

Instead of modelling capacity and gender-separation constraints separately, we can also combine them and use

$$\sum_{p \in \mathcal{P}^{\mathrm{f}}(t)} x_{pr} \le c_r g_{rt} \qquad \forall t \in \mathcal{T},\ r \in \mathcal{R}, \tag{35}$$

$$\sum_{p \in \mathcal{P}^{\mathrm{m}}(t)} x_{pr} \le c_r(1 - g_{rt}) \qquad \forall t \in \mathcal{T},\ r \in \mathcal{R}, \tag{36}$$

instead of Eqs. (32) to (34). Fourth, we model the single room constraints via

$$s_{prt} \le x_{pr} \qquad \forall t \in \mathcal{T},\ p \in \mathcal{P}^*(t),\ r \in \mathcal{R}, \tag{37}$$

$$c_r s_{prt} + \sum_{q \in \mathcal{P}(t) \setminus \{p\}} x_{qr} \le c_r \qquad \forall t \in \mathcal{T},\ p \in \mathcal{P}^*(t),\ r \in \mathcal{R}. \tag{38}$$

Alternatively to Eq. (38), we can also integrate the single room constraints with the gender-separation and capacity constraints Eqs. (35) and (36) via

$$\sum_{p \in \mathcal{P}^{\mathrm{f}}(t)} x_{pr} + \sum_{p \in \mathcal{P}^{\mathrm{f}} \cap \mathcal{P}^*(t)} (c_r - 1) s_{prt} \le c_r g_{rt} \qquad \forall t \in \mathcal{T},\ r \in \mathcal{R}, \tag{39}$$

$$\sum_{p \in \mathcal{P}^{\mathrm{m}}(t)} x_{pr} + \sum_{p \in \mathcal{P}^{\mathrm{m}} \cap \mathcal{P}^*(t)} (c_r - 1) s_{prt} \le c_r(1 - g_{rt}) \qquad \forall t \in \mathcal{T},\ r \in \mathcal{R}. \tag{40}$$

Last, we ensure that any pre-fixed assignments are respected:

$$x_{pr} = 1 \quad \forall (p, r) \in \mathcal{F}. \tag{41}$$

We then compare the following IP-formulations to find the best performing constraint set.

(M) $\max f^{\text{priv}}$ s.t. Eqs. (31) to (34), (37), (38) and (41)
(N) $\max f^{\text{priv}}$ s.t. Eqs. (31), (35) to (38) and (41)
(O) $\max f^{\text{priv}}$ s.t. Eqs. (31), (37) and (39) to (41)
(P) $\max 0$ s.t. constraints of (O), Eq. (29)

Apart from the additional constraints modelling the single-room assignment resembles the constraint set of our IP (M) the constraint set of Liu et al.'s IP $\mathrm{APRA}^{\mathrm{WT}}_{\mathrm{GC}_1\&\mathrm{TC}}$, and the constraints of our IP (N) resembles the ones of their IP $\mathrm{APRA}^{\mathrm{WT}}_{\mathrm{AGC}_1\&\mathrm{TC}}$. Liu et al. (2024). In their setting, the more aggregated IP $\mathrm{APRA}^{\mathrm{WT}}_{\mathrm{AGC}_1\&\mathrm{TC}}$ performs better.

The formulations for IPs (M) to (P) were evaluated on the same computational setup as in and the implementation is available on Brandt

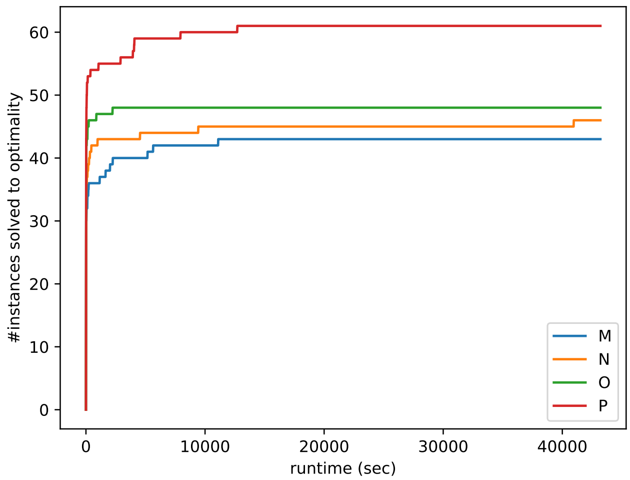


Fig. 5. Comparison of IPs (M)–(P) using 62 real-life instances.

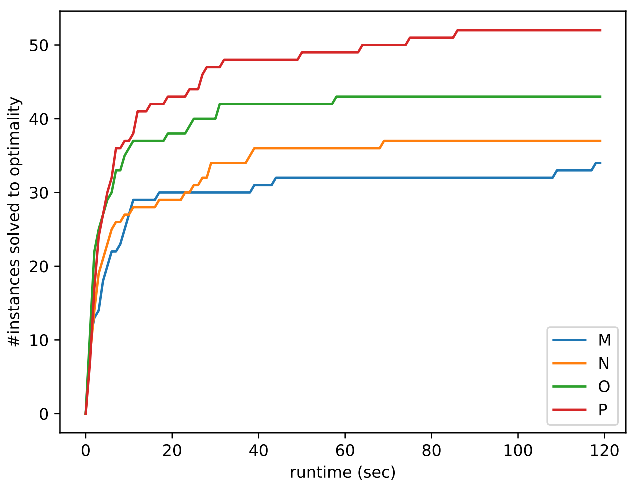


Fig. 6. IP (P) solves 52 instances in <100 s.

and Engelhardt (2024). The results show the dominance of IP (P) over the other IPs, cf. Fig. 5. However, it strongly depends on the use case whether IP (P) is the best one to use as, naturally, it is feasible in fewer instances than IP (O). With our real-life instances, (P) was feasible in 72.5% whereas (O) was feasible in 97.75%. However, due to the fast runtime of IP (P), cf. Fig. 6, it may be worthwhile to check first whether IP (P) is feasible before switching to (O).

6. Dynamic PRA

As Dynamic PRA, we understand PRA with a rolling time horizon similar to the definition in Ouelhadj and Petrovic (2009). Here, for every patient we are also given a registration time period so that the set $\mathcal{P}$ of all (known) patients is updated each time period. For every time period $t \in \mathcal{T}$, all known patients, i.e., patients whose registration dates are before or equal to t, have to be assigned to rooms. All room assignments of the current time period are then stored in the set $\mathcal{F}$. We assume that $\mathcal{F}$ does not contain irrelevant data, i.e., discharged patients are deleted immediately to ensure the correct computation of f^{trans}. Hence, $\mathcal{F}$ is updated after every iteration just like the patient set $\mathcal{P}$.

As rescheduling is frequently done in practice, this approach relates more closely to the real-life problem than the static version, where we assume total information regarding patient arrivals and departures. In this section, we describe how we combine four IP models and our combinatorial insights to efficiently solve the dynamic PRA by exploiting the IPs' different runtimes.

The key question is how to link $\mathcal{F}$ to the new model, since past assignments may be arbitrary bad in the here-and-now. Using heuristics, this can be addressed by using neighbourhoods that allow for transfers/reassignments, see Ceschia and Schaerf (2012). In comparison, we solve every single iteration to optimality.[3]

Thus, we require a mechanism that allows for transfers of some kind. Here, the iterative nature of the dynamic PRA allows us to introduce a variant of IPs (P) and (O) where transfers are not entirely forbidden, but only changes to the current room assignment are allowed. We call this concept *same-day transfers* and formulate it as

(O*) $\max\ (f^{\text{priv}}, \sum_{(r,p)\in\mathcal{F}} x_{pr})$ s.t. Eqs. (31), (37), (39) and (40)
(P*) $\max\ \sum_{(r,p)\in\mathcal{F}} x_{pr}$ s.t. constraints of (O*), Eq. (29).

For our algorithm, we combine the IPs (H), (O*), (P), and (P*) and our combinatorial insights as follows. First, we check combinatorially whether the instance is feasible since we observed that the combinatorial feasibility check is significantly faster than building a respective IP (using gurobipy), not to mention solving it. Second, we use the no-transfers formulation IP (P). Note that we here make use of our second combinatorial insight, i.e., the computation of $s^{\max}$. If IP (P) is infeasible, we solve the instance again using the same-day transfer formulation IP (P*). If IP (P*) is also infeasible, we use IP (O*) maximising the number of private patients who get their own room while minimising the number of transfers in the first time period. If again, no feasible solution for (O*) is found within 20 s, we solve the instance using IP (H), which allows arbitrary many transfers and is therefore always feasible.

After successful computation, we fix all patient-room assignments for patients that are in hospital in the current time period by adding them to set $\mathcal{F}$ while removing outdated ones. We then update the patient set and continue analogously with the next time period. A visualisation of this algorithm is provided in Fig. 7.

Real-world instances. We evaluate our algorithm again on 62 real-world instances spanning a whole year. As a result we get that all instances use 365 iterations of the algorithm and all are solved within less than 600 s per year, cf. Fig. 8. For application purposes however, the runtime per iteration is more interesting than the total runtime of 365 iterations. Therefore, we report in Fig. 9 the runtime of all $62 \cdot 365 = 22630$ iterations individually. The results show that all but four iterations are solved within less than 15 s, cf. Fig. 9(a), and more than 95% of all iterations are solved within less a second, cf. Fig. 9(b).

Although our dynamic algorithm is a heuristic and, thus, does not guarantee a certain solution quality, we can assess a solution's quality using our combinatorial insights: For 44 instances the optimal value $f^{\text{priv}} = s^{\max}$ was achieved. For 12 instances, we achieve $f^{\text{priv}} \geq 0.9885 s^{\max}$ and for one $f^{\text{priv}} = 0.946 s^{\max}$. The remaining 5 instances have no private patients. The high quality of our solutions w.r.t. f^{priv} is especially remarkable since in 26 of them no transfers are needed, 28 use between 1 and 27 transfers, and 8 between 28 and 80 transfers. The complete report of the total computation time of all instances and the objective values found by our algorithm can be found in the Appendix A.

Artificial instances. To allow independent variation and further benchmarking on our results, we also report results on artificial instances, which we generated based on randomised recreation of the structure of our real-world data. The instances are publicly available alongside the implementation of our dynamic algorithm at Brandt and Engelhardt

[3] This is not equivalent to global optimality, since past decisions may be suboptimal for a changed patient set, and changing those might incur additional transfers.

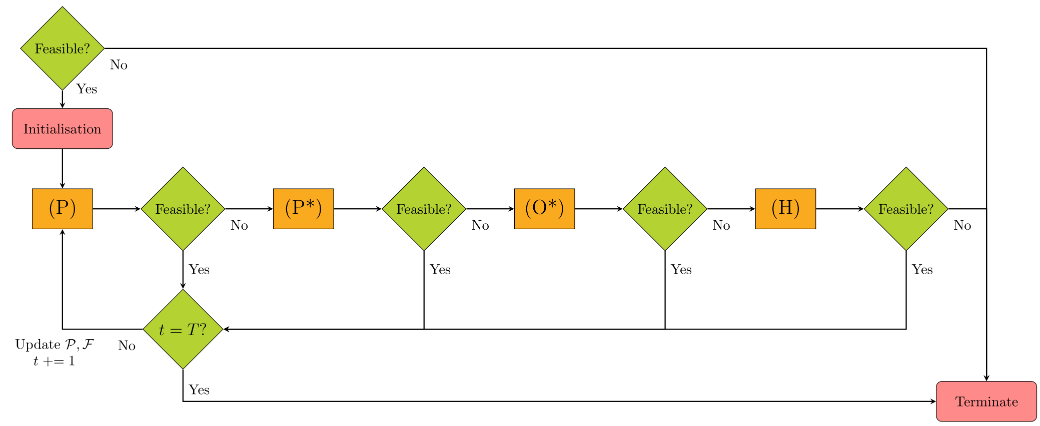


Fig. 7. Algorithm for dynamic PRA.

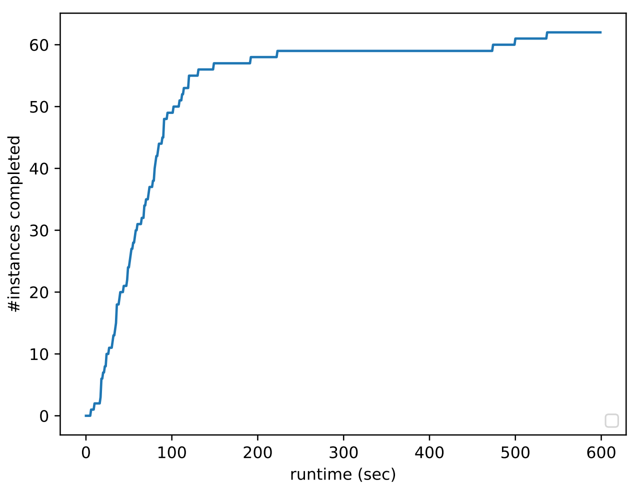


Fig. 8. Runtime of algorithm for dynamic PRA with $T = 365$.

(2024). We considered four different occupancy levels (0.5, 0.65, 0.8, 0.95), and generated 80 instances for each level. The proposed algorithm solves all instance within less than 1800 s per year, cf. Fig. 10. Fig. 10 further suggests that our artificial instances are a similar mix of easy and hard instances as our real-world instances.

Notably, with higher occupancy the algorithm's performance degrades. However, even in the worst case, we require no more than half an hour to simulate a full year. For application purposes, we again report the runtime for each of the $80 \cdot 365 = 29200$ iterations in Fig. 9. The results show a single outlier of 91.3 s. Apart from that, >75% of instances are solved in less than a second, and ≈95% of instances in less than 10 s (see Fig. 11). Although our dynamic algorithm is a heuristic and, thus, does not guarantee a certain solution quality, we can assess a solution's quality using our combinatorial insights: First consider f^{priv}. For low occupancy (50%), the algorithm reaches global optimality, i.e, $f^{\text{priv}} = s^{\max}$ for 67/80 instances. For the remaining instances, we achieve $f^{\text{priv}} \geq 0.97 s^{\max}$ on average and $f^{\text{priv}} \geq 0.81 s^{\max}$ in the worst case. As we increase occupancy, this pattern shifts. For high occupancy (95%), 16/80 instances are solved to global optimality. For the remaining instances, we achieve $f^{\text{priv}} \geq 0.96 s^{\max}$ on average and $f^{\text{priv}} \geq 0.82 s^{\max}$ in the worst case. For transfers, we also see a directly link between occupancy and number of transfers. In the 50% occupancy setting, 60/80 solutions require no transfers, whereas 10/80 require between 60 and 186 transfers. The overall mean of f^{trans} in this setting is 21. As we increase occupancy, this distribution shifts. For the 95% occupancy setting, 10/80 solutions require fewer than 60 transfers, and 70/80 require between 60 and 376. The overall mean of f^{trans} in this high occupancy setting is 180, which is still less than one transfer every other day. We report detailed tables with statistical information about the artificial instances together with the computation time and objective values found by our algorithm in Appendices A and B.

In summary, our results showcase that the given artificial dataset captures important features of the real-world problem. This includes both easier and harder instances with different patient set configurations. Compared to real-world data it allows for more insights from benchmarking, e.g., via assessing the effect of feature variations such as occupancy levels on algorithm performance.

7. Future work

We close this paper by pointing out multiple not yet fully explored aspects of PRA to inspire future research. We give an overview over possible modelling extensions for our definition of PRA. Where possible, we provide first experimental computational results and point out promising areas for further research.

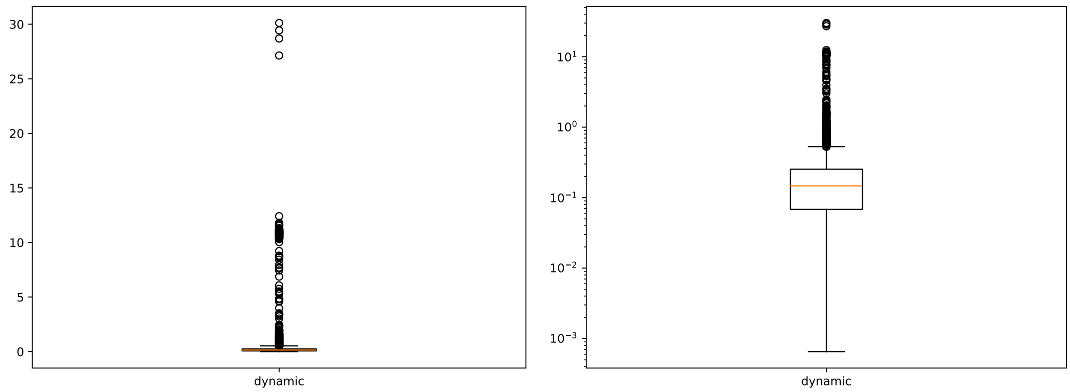
7.1. Criteria for easy and hard instances

Both our real-world instances and the artificial instances contain instances for which our algorithm needs significantly more time to solve them than for others. We are currently not aware of any criteria to characterise such instances other than size. However, our computational results show that size is not always the decisive factor. Our algorithm solves for example instance `load_50_27` with 1538 patients in 123 s whereas it needs for `load_50_54` with only 330 patients 571 s. Hence, further research is needed to identify criteria that lead to instances that are hard (easy) to solve for IP solvers.

7.2. Scaling to multiple wards

Similar wards within the same speciality can be planned jointly. Initial computational testing showed that, in this case, the runtime scales linearly up to a 150 rooms over a planning horizon of 365 time periods.

The proposed IP modelling approach can also be extended to manage multiple dissimilar wards. For this, new constraints must be added to model which patient can be assigned to which ward. We evaluated this both for single specialities with up to 9 normal and 2 intensive care wards, and for the full RWTH Aachen University Hospital with 800 rooms and 53.000 patients (again over a planning horizon of 365 time periods). The runtime for the full hospital averages to about



(a) normal axis   (b) logscaled axis

Fig. 9. Runtime per iteration of the algorithm for dynamic PRA on real-world instances.

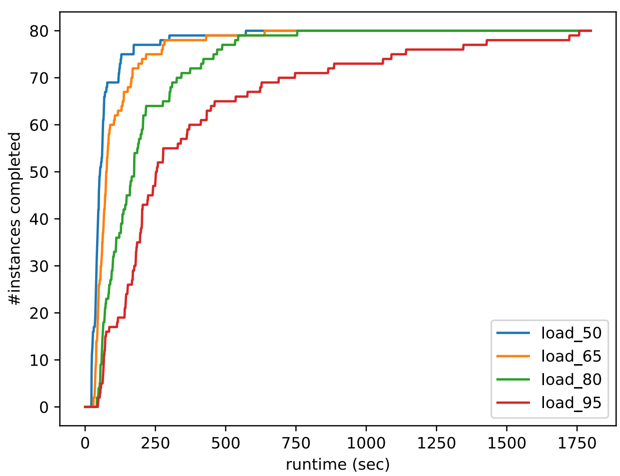


Fig. 10. Runtime of algorithm for dynamic PRA with $T = 365$ for multiple occupancy levels.

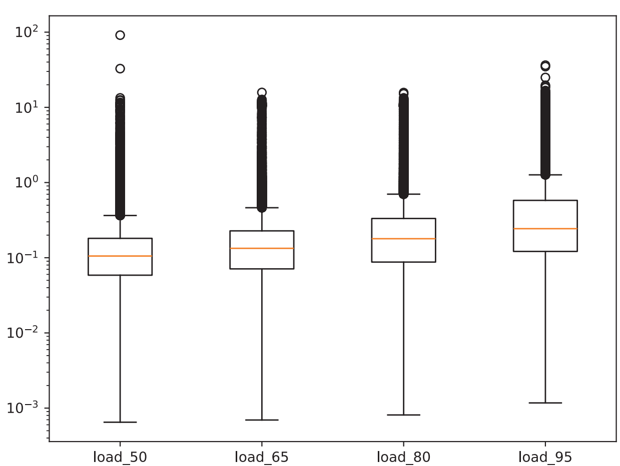


Fig. 11. Runtime per iteration for artificial instances, logscaled axis.

2s per iteration, with a larger variation than for single wards, i.e, some days requiring more than 10 s for an initial feasible solution. However, in our modelling approach patient-assignment feasibility was only based on past patient-stay data, without consulting with medical professionals. Thus, further research first needs to identify suitable metrics for patient-ward suitability and then include these as an objective component.

7.3. Patient conflicts

Due to medical or social reasons there may be pairs of patients who cannot share a room, e.g., two patients with asthma or one woman who just gave birth and one who lost the child. Such so-called patient conflicts can easily be integrated into all our proposed IP-formulations by adding conflict constraints. Since we do not have any real data about patient conflicts, we experimented with a small number of randomly generated conflicts. In our setting, this had neither an effect on the runtime nor the objective value. However, in theory, a large number of conflicts may render an instance infeasible. In the future, we will further investigate what conflicts occur in reality and constitutes their effect on runtime and solution quality.

7.4. Patient preferences

If more than one patient is assigned to a room, assigning suitable room-mates also constitutes a further goal (Hantel & Benkenstein, 2019). Specifically, patient combinations exist that may be beneficial both for patients and staff. For example, it is known that patients recover faster if they feel comfortable, therefore, a room-mate to whom they can relate may be beneficial (Chaudhury, Mahmood, & Valente, 2005; Hantel & Benkenstein, 2019). Or, if an international patient is not fluent in the local language, then it is beneficial for both patient and staff if the roommate can translate. First computational experiments with IP-formulations showed that incorporating inter-patient preferences into the IP models leads to a significant increase in runtime. Developing an efficient way to integrate the choice of suitable room-mates remains ongoing research.

7.5. Accompanying person

Some patients are entitled to bring an accompanying person with them to the hospital. If the accompanying person occupies a normal patient bed, this can easily be integrated into all our proposed IP-formulations by adding weights to patients and/or not implementing assignment variables for single rooms for the respective patients. If the accompanying person sleeps on an additional roll-in bed and does not occupy a patient bed, it depends on the hospital's policy whether it is, e.g., desirable to avoid assigning multiple patients with an accompanying person to the same room or whether gender separation also needs to be respected for the accompanying person. It is still ongoing research to determine the decisive criteria currently in use for this task.

Table A.1
Overview of artificial data. For lor and los the given values are the medians. For rooms, the number of single/double rooms is given.

Instance	Patients	Female	Male	Private	Emergency	lor	los	Beds	Rooms
load_50_1	898	367	531	142	285	3	6	28	0/14
load_50_2	791	323	468	111	253	3	6	28	0/14
load_50_3	851	365	486	140	268	3	6	28	0/14
load_50_4	877	387	490	139	299	3	6	28	0/14
load_50_5	782	336	446	115	248	3	5	28	0/14
load_50_6	837	359	478	142	295	3	5	28	0/14
load_50_7	839	368	471	150	259	3	6	28	0/14
load_50_8	877	392	485	126	292	3	5	28	0/14
load_50_9	1124	463	661	155	391	3	6	34	6/14
load_50_10	1096	491	605	164	343	3	6	34	6/14
load_50_11	1069	456	613	151	362	3	5	34	6/14
load_50_12	1007	422	585	154	353	3	5	34	6/14
load_50_13	1045	443	602	169	361	3	5	32	8/12
load_50_14	1057	444	613	160	368	3	6	32	8/12
load_50_15	953	407	546	144	305	3	5	32	8/12
load_50_16	1054	450	604	174	325	3	6	32	8/12
load_50_17	1014	431	583	162	319	3	6	32	8/12
load_50_18	1045	444	601	167	379	2	5	32	8/12
load_50_19	898	367	531	142	310	3	5	28	4/12
load_50_20	791	323	468	111	253	3	6	28	4/12
load_50_21	412	176	236	70	126	3	6	16	0/8
load_50_22	589	256	333	92	185	2	6	16	0/8
load_50_23	503	229	274	83	178	3	5	18	14/2
load_50_24	568	260	308	94	206	3	5	18	14/2
load_50_25	581	241	340	85	183	3	5	18	14/2
load_50_26	545	239	306	78	169	3	6	18	14/2
load_50_27	1538	666	872	261	457	3	6	48	0/24
load_50_28	1414	591	823	250	444	3	6	48	0/24
load_50_29	1021	408	613	187	322	3	6	32	8/12
load_50_30	1000	423	577	179	306	3	6	32	8/12
load_50_31	896	360	536	151	290	3	5	32	8/12
load_50_32	935	420	515	172	292	3	6	32	8/12
load_50_33	943	391	552	152	314	3	6	32	8/12
load_50_34	928	426	502	152	303	3	6	32	8/12
load_50_35	922	374	548	192	251	3	6	32	8/12
load_50_36	979	404	575	147	313	3	5	32	8/12
load_50_37	1038	419	619	180	332	3	5	32	8/12
load_50_38	941	415	526	174	284	3	6	32	8/12
load_50_39	1183	529	654	164	383	2	6	24	2/11
load_50_40	1170	491	679	170	381	2	6	24	2/11
load_50_41	1352	559	793	198	446	2	6	28	0/14
load_50_42	1423	601	822	201	429	2	7	28	0/14
load_50_43	1355	622	733	199	449	2	6	28	0/14
load_50_44	1382	606	776	187	444	2	6	28	0/14
load_50_45	1355	622	733	199	449	2	6	28	4/12
load_50_46	1382	606	776	187	444	2	6	28	4/12
load_50_47	1352	559	793	198	446	2	6	28	4/12
load_50_48	1423	601	822	201	429	2	7	28	4/12
load_50_49	791	361	430	118	270	2	5	16	0/8
load_50_50	723	320	403	97	268	2	4	16	0/8
load_50_51	783	345	438	140	267	2	5	16	0/8
load_50_52	753	332	421	106	256	2	5	16	0/8
load_50_53	316	136	180	47	109	7	6	30	2/14
load_50_54	330	147	183	51	122	7	4	30	2/14
load_50_55	853	378	475	137	299	2	5	16	0/8
load_50_56	791	341	450	150	241	2	6	16	0/8
load_50_57	747	322	425	130	236	2	7	16	0/8
load_50_58	831	322	509	110	271	2	6	16	0/8
load_50_59	803	359	444	125	236	2	6	16	0/8
load_50_60	832	360	472	140	275	2	6	16	0/8
load_50_61	836	355	481	131	273	2	6	16	0/8
load_50_62	878	364	514	132	273	2	6	16	0/8
load_50_63	863	391	472	112	292	2	5	16	0/8
load_50_64	812	361	451	154	267	2	6	16	0/8
load_50_65	1791	769	1022	273	547	2	6	34	6/14
load_50_66	1653	666	987	243	529	2	6	34	6/14
load_50_67	1032	428	604	165	352	2	6	20	8/6
load_50_68	995	437	558	169	304	2	7	20	8/6
load_50_69	984	387	597	182	299	2	6	20	8/6
load_50_70	910	376	534	149	272	2	7	20	8/6

(continued on next page)

Table A.1 (*continued*).

Instance	Patients	Female	Male	Private	Emergency	lor	los	Beds	Rooms
load_50_71	988	396	592	155	299	2	6	20	8/6
load_50_72	1065	443	622	181	344	2	6	20	8/6
load_50_73	1095	460	635	179	373	2	5	20	8/6
load_50_74	1046	454	592	171	315	2	7	20	8/6
load_50_75	164	71	93	27	54	7	5	12	4/4
load_50_76	115	45	70	14	38	6	5	12	4/4
load_50_77	205	96	109	28	72	7	5	16	8/4
load_50_78	157	76	81	21	46	8	6	16	8/4
load_50_79	205	96	109	28	64	7	8	16	0/8
load_50_80	157	76	81	21	46	8	7	16	0/8
load_65_1	1042	446	596	149	321	3	6	28	0/14
load_65_2	1098	477	621	150	367	2	6	28	0/14
load_65_3	1258	565	693	198	408	3	6	28	0/14
load_65_4	1083	472	611	166	358	3	5	28	0/14
load_65_5	1094	478	616	158	363	3	5	28	0/14
load_65_6	998	418	580	150	332	3	5	28	0/14
load_65_7	1007	418	589	153	332	3	6	28	0/14
load_65_8	1242	554	688	164	400	3	6	28	0/14
load_65_9	1491	638	853	224	453	3	6	34	6/14
load_65_10	1234	538	696	193	396	3	5	34	6/14
load_65_11	1371	618	753	220	452	3	6	34	6/14
load_65_12	1359	576	783	216	447	3	5	34	6/14
load_65_13	566	247	319	86	189	3	5	16	0/8
load_65_14	690	318	372	115	216	3	5	16	0/8
load_65_15	633	280	353	86	200	3	6	18	14/2
load_65_16	793	337	456	122	249	2	6	18	14/2
load_65_17	646	280	366	89	196	3	6	18	14/2
load_65_18	789	324	465	116	257	3	5	18	14/2
load_65_19	1408	596	812	213	491	3	5	32	8/12
load_65_20	1262	568	694	196	412	3	6	32	8/12
load_65_21	1183	520	663	203	418	3	5	32	8/12
load_65_22	1196	512	684	174	388	3	5	32	8/12
load_65_23	1313	596	717	193	444	3	5	32	8/12
load_65_24	1401	604	797	215	440	3	5	32	8/12
load_65_25	1453	622	831	214	499	3	5	33	3/15
load_65_26	1268	542	726	178	392	3	6	33	3/15
load_65_27	1258	565	693	198	408	3	6	28	4/12
load_65_28	1042	446	596	149	321	3	6	28	4/12
load_65_29	1366	583	783	228	441	3	5	32	8/12
load_65_30	1213	552	661	211	391	3	6	32	8/12
load_65_31	1158	512	646	179	380	3	6	32	8/12
load_65_32	1240	523	717	207	381	3	6	32	8/12
load_65_33	1212	499	713	219	380	3	5	32	8/12
load_65_34	1131	474	657	185	363	3	5	32	8/12
load_65_35	1161	517	644	177	388	3	6	32	8/12
load_65_36	1243	536	707	195	393	3	5	32	8/12
load_65_37	1213	514	699	216	396	3	6	32	8/12
load_65_38	1226	552	674	208	422	3	5	32	8/12
load_65_39	1820	797	1023	320	581	3	7	48	0/24
load_65_40	2229	1021	1208	384	721	3	6	48	0/24
load_65_41	1509	675	834	216	520	2	5	24	2/11
load_65_42	1480	648	832	209	492	2	6	24	2/11
load_65_43	1749	822	927	238	596	2	6	28	0/14
load_65_44	1781	765	1016	254	638	2	5	28	0/14
load_65_45	1805	787	1018	271	603	2	5	28	0/14
load_65_46	1798	778	1020	247	593	2	6	28	0/14
load_65_47	1749	822	927	238	596	2	6	28	4/12
load_65_48	1781	765	1016	254	638	2	5	28	4/12
load_65_49	1805	787	1018	271	622	2	5	28	4/12
load_65_50	1798	778	1020	247	593	2	6	28	4/12
load_65_51	1053	481	572	170	360	2	6	16	0/8
load_65_52	1039	427	612	143	355	2	5	16	0/8
load_65_53	972	406	566	141	340	2	4	16	0/8
load_65_54	1041	459	582	151	328	2	7	16	0/8
load_65_55	1098	446	652	174	361	2	6	16	0/8
load_65_56	954	389	565	175	306	2	6	16	0/8
load_65_57	1047	491	556	153	319	2	7	16	0/8
load_65_58	961	417	544	125	292	2	7	16	0/8
load_65_59	1099	482	617	153	346	2	6	16	0/8
load_65_60	1097	481	616	174	325	2	7	16	0/8
load_65_61	1018	423	595	160	332	2	6	16	0/8
load_65_62	1035	464	571	164	304	2	6	16	0/8
load_65_63	927	374	553	175	306	2	5	16	0/8
load_65_64	1021	434	587	162	327	2	6	16	0/8

(continued on next page)

Table A.1 (*continued*).

load_65_65	2336	1005	1331	386	745	2	6	34	6/14
load_65_66	2306	1031	1275	371	718	2	7	34	6/14
load_65_67	1337	559	778	223	438	2	6	20	8/6
load_65_68	1377	580	797	212	450	2	6	20	8/6
load_65_69	1488	681	807	250	513	2	5	20	8/6
load_65_70	1394	586	808	240	445	2	6	20	8/6
load_65_71	1334	567	767	218	445	2	6	20	8/6
load_65_72	1433	620	813	232	452	2	6	20	8/6
load_65_73	1359	574	785	203	424	2	6	20	8/6
load_65_74	1208	514	694	195	375	2	7	20	8/6
load_65_75	206	87	119	33	69	7	5	12	4/4
load_65_76	187	84	103	28	64	6	5	12	4/4
load_65_77	245	102	143	36	96	7	4	16	8/4
load_65_78	258	103	155	28	72	8	6	16	8/4
load_65_79	245	102	143	36	86	7	5	16	0/8
load_65_80	258	103	155	28	72	8	6	16	0/8
load_80_1	1523	645	878	225	455	3	6	28	0/14
load_80_2	1251	545	706	185	398	3	5	28	0/14
load_80_3	1502	656	846	236	494	3	6	28	0/14
load_80_4	1323	569	754	194	399	3	7	28	0/14
load_80_5	1340	582	758	213	449	3	5	28	0/14
load_80_6	1475	669	806	242	471	3	5	28	0/14
load_80_7	1349	575	774	191	464	3	6	28	0/14
load_80_8	1435	596	839	225	475	3	6	28	0/14
load_80_9	1713	712	1001	268	538	3	6	34	6/14
load_80_10	1694	724	970	245	571	3	5	34	6/14
load_80_11	1618	712	906	247	544	2	5	34	6/14
load_80_12	1697	747	950	228	591	3	5	34	6/14
load_80_13	769	330	439	122	256	3	6	16	0/8
load_80_14	704	303	401	92	233	3	6	16	0/8
load_80_15	951	371	580	160	330	3	5	18	14/2
load_80_16	896	392	504	146	274	3	6	18	14/2
load_80_17	938	393	545	149	298	3	5	18	14/2
load_80_18	900	410	490	128	302	3	5	18	14/2
load_80_19	1652	713	939	261	526	3	6	32	8/12
load_80_20	1438	658	780	210	469	3	6	32	8/12
load_80_21	1524	668	856	220	486	3	6	32	8/12
load_80_22	1540	693	847	224	485	3	6	32	8/12
load_80_23	1557	687	870	238	513	3	6	32	8/12
load_80_24	1751	739	1012	256	575	2	5	32	8/12
load_80_25	1652	713	939	256	520	3	7	33	3/15
load_80_26	1477	648	829	223	465	3	6	33	3/15
load_80_27	1523	645	878	225	477	3	6	28	4/12
load_80_28	1251	545	706	185	416	3	5	28	4/12
load_80_29	2167	961	1206	345	685	3	6	48	0/24
load_80_30	1735	744	991	266	579	3	6	48	0/24
load_80_31	1615	711	904	279	482	3	7	32	8/12
load_80_32	1403	582	821	227	434	3	5	32	8/12
load_80_33	1436	599	837	211	459	3	5	32	8/12
load_80_34	1561	676	885	262	515	3	6	32	8/12
load_80_35	1555	652	903	270	494	3	6	32	8/12
load_80_36	1632	727	905	251	513	3	6	32	8/12
load_80_37	1477	632	845	238	478	3	6	32	8/12
load_80_38	1373	575	798	218	450	3	6	32	8/12
load_80_39	1589	658	931	260	495	3	7	32	8/12
load_80_40	1560	649	911	267	501	3	6	32	8/12
load_80_41	1858	829	1029	288	583	2	7	24	2/11
load_80_42	1862	844	1018	267	634	2	5	24	2/11
load_80_43	2137	947	1190	294	730	2	5	28	0/14
load_80_44	2108	954	1154	287	696	2	5	28	0/14
load_80_45	2125	938	1187	332	689	2	6	28	0/14
load_80_46	2210	962	1248	313	748	2	6	28	0/14
load_80_47	2137	947	1190	294	730	2	5	28	4/12
load_80_48	2108	954	1154	287	696	2	5	28	4/12
load_80_49	2130	941	1189	332	692	2	6	28	4/12
load_80_50	2210	962	1248	313	765	2	5	28	4/12
load_80_51	1269	603	666	206	418	2	6	16	0/8
load_80_52	1324	592	732	174	448	2	6	16	0/8
load_80_53	1074	494	580	147	341	2	7	16	0/8
load_80_54	1301	569	732	206	428	2	6	16	0/8
load_80_55	1333	550	783	229	437	2	6	16	0/8
load_80_56	1384	613	771	221	433	2	6	16	0/8
load_80_57	1294	579	715	210	448	2	5	16	0/8
load_80_58	1361	570	791	225	411	2	6	16	0/8

(*continued on next page*)

Table A.1 (*continued*).

load_80_59	1438	605	833	228	439	2	6	16	0/8
load_80_60	1369	535	834	232	419	2	6	16	0/8
load_80_61	1339	562	777	210	421	2	7	16	0/8
load_80_62	1361	601	760	216	437	2	6	16	0/8
load_80_63	1287	565	722	196	413	2	7	16	0/8
load_80_64	1450	592	858	204	456	2	7	16	0/8
load_80_65	2780	1199	1581	425	921	2	5	34	6/14
load_80_66	2427	1012	1415	365	784	2	6	30	2/14
load_80_67	1599	724	875	275	516	2	6	20	8/6
load_80_68	1593	696	897	235	503	2	6	20	8/6
load_80_69	1591	679	912	279	504	2	6	20	8/6
load_80_70	1734	791	943	270	536	2	6	20	8/6
load_80_71	1739	768	971	268	580	2	6	20	8/6
load_80_72	1795	774	1021	295	579	2	6	20	8/6
load_80_73	1668	683	985	251	560	2	5	20	8/6
load_80_74	1683	724	959	266	509	2	7	20	8/6
load_80_75	235	98	137	39	86	7	5	12	4/4
load_80_76	181	81	100	33	73	8	4	12	4/4
load_80_77	272	115	157	43	88	6	5	16	8/4
load_80_78	318	127	191	43	112	6	5	16	8/4
load_80_79	272	115	157	43	88	6	7	16	0/8
load_80_80	318	127	191	43	112	6	6	16	0/8
load_95_1	1658	722	936	263	525	3	6	28	0/14
load_95_2	1456	639	817	208	470	3	6	28	0/14
load_95_3	1495	629	866	231	508	3	5	28	0/14
load_95_4	1690	741	949	267	529	3	6	28	0/14
load_95_5	1610	674	936	220	501	3	6	28	0/14
load_95_6	1604	696	908	226	496	3	6	28	0/14
load_95_7	1654	724	930	255	529	3	6	28	0/14
load_95_8	1544	675	869	234	478	3	6	28	0/14
load_95_9	2018	886	1132	291	658	3	6	34	6/14
load_95_10	1986	865	1121	318	691	3	6	34	6/14
load_95_11	1956	845	1111	306	641	3	6	34	6/14
load_95_12	2001	842	1159	299	680	3	5	34	6/14
load_95_13	926	389	537	160	292	3	5	16	0/8
load_95_14	855	370	485	133	267	3	7	16	0/8
load_95_15	1158	501	657	163	395	3	5	18	14/2
load_95_16	1056	447	609	167	348	3	6	18	14/2
load_95_17	1085	454	631	162	359	3	5	18	14/2
load_95_18	1086	463	623	159	376	3	5	18	14/2
load_95_19	1851	797	1054	285	617	3	6	32	8/12
load_95_20	1889	835	1054	275	659	3	5	32	8/12
load_95_21	1758	770	988	265	573	3	6	32	8/12
load_95_22	1951	850	1101	297	676	3	5	32	8/12
load_95_23	2014	876	1138	284	654	3	6	32	8/12
load_95_24	1801	778	1023	278	633	3	5	32	8/12
load_95_25	1658	722	936	263	525	3	6	28	4/12
load_95_26	1479	645	834	211	497	3	5	28	4/12
load_95_27	1495	629	866	231	522	3	5	28	4/12
load_95_28	1693	743	950	268	544	3	6	28	4/12
load_95_29	1720	721	999	277	563	3	6	32	8/12
load_95_30	1768	755	1013	295	590	3	6	32	8/12
load_95_31	1718	751	967	295	580	3	5	32	8/12
load_95_32	1695	695	1000	314	537	3	6	32	8/12
load_95_33	1713	730	983	269	589	3	5	32	8/12
load_95_34	1756	754	1002	329	563	3	6	32	8/12
load_95_35	1832	826	1006	317	596	3	6	32	8/12
load_95_36	1879	785	1094	337	591	3	6	32	8/12
load_95_37	1831	761	1070	314	608	3	6	32	8/12
load_95_38	1891	781	1110	314	611	3	6	32	8/12
load_95_39	2798	1145	1653	482	839	3	7	48	0/24
load_95_40	2722	1129	1593	446	854	3	6	48	0/24
load_95_41	2169	958	1211	296	725	2	5	24	2/11
load_95_42	2199	952	1247	327	738	2	5	24	2/11
load_95_43	2568	1135	1433	377	840	2	6	28	0/14
load_95_44	2553	1127	1426	363	843	2	6	28	0/14
load_95_45	2661	1181	1480	410	836	2	6	28	0/14
load_95_46	2608	1133	1475	370	864	2	6	28	0/14
load_95_47	2588	1145	1443	380	864	2	6	28	4/12
load_95_48	2667	1170	1497	376	894	2	5	28	4/12
load_95_49	2681	1194	1487	413	842	2	6	28	4/12
load_95_50	2641	1150	1491	372	889	2	6	28	4/12
load_95_51	1394	616	778	208	432	2	5	16	0/8
load_95_52	1311	557	754	181	434	2	5	16	0/8

(*continued on next page*)

Table A.1 (*continued*).

load_95_53	1433	655	778	216	452	2	6	16	0/8
load_95_54	1501	658	843	216	507	2	5	16	0/8
load_95_55	1469	692	777	259	463	2	6	16	0/8
load_95_56	2912	1258	1654	467	898	2	6	30	2/14
load_95_57	1398	578	820	223	444	2	6	16	0/8
load_95_58	1589	677	912	245	505	2	5	16	0/8
load_95_59	1526	643	883	229	472	2	6	16	0/8
load_95_60	1664	692	972	273	514	2	6	16	0/8
load_95_61	1449	614	835	241	463	2	6	16	0/8
load_95_62	1545	653	892	259	482	2	6	16	0/8
load_95_63	1497	674	823	238	475	2	6	16	0/8
load_95_64	1569	659	910	271	509	2	5	16	0/8
load_95_65	3471	1456	2015	564	1125	2	6	34	6/14
load_95_66	2572	1087	1485	425	785	2	7	34	6/14
load_95_67	1963	865	1098	317	614	2	6	20	8/6
load_95_68	1869	803	1066	289	581	2	6	20	8/6
load_95_69	1921	820	1101	290	646	2	6	20	8/6
load_95_70	1942	812	1130	295	627	2	6	20	8/6
load_95_71	1889	848	1041	300	590	2	6	20	8/6
load_95_72	2178	952	1226	335	691	2	6	20	8/6
load_95_73	1982	847	1135	313	658	2	5	20	8/6
load_95_74	1952	849	1103	299	589	2	6	20	8/6
load_95_75	267	117	150	31	84	7	5	12	4/4
load_95_76	227	90	137	45	79	7	8	12	4/4
load_95_77	309	122	187	55	122	7	3	16	8/4
load_95_78	348	169	179	54	121	7	6	16	8/4
load_95_79	266	101	165	48	96	6	5	16	0/8
load_95_80	335	161	174	50	109	7	6	16	0/8

7.6. Uncertainty

Considering uncertainty is essential to ensure real-world applicability and validity of results. By using a dynamic time horizon, we already integrated the uncertain arrival of emergency patients. A second and equally relevant factor, however, is the uncertainty in the length of stay. It is easily possible to update a patient's planned discharge date in every iteration of our algorithm for dynamic PRA. If a patient's stay is prolonged, the feasibility check should then be repeated for the affected time periods. It is still an open question to assess the consequences of such updates on the solution quality. For wards with a high uncertainty in patient length of stay, it also might be better to integrate the uncertainty more directly in the algorithm to compute robust solutions. It is however also yet undefined what a *robust* solution in this context means. One possible avenue for that was already proposed by Ceschia and Schaerf, who encode an overcrowding risk penalty term in their objective (Ceschia & Schaerf, 2012). Looking at different ways of modelling this uncertainty, and their performance, might also constitute a promising avenue for further research.

8. Conclusion

In this work, we presented new combinatorial insights for the patient-to-room assignment problem with regard to feasibility and the assignment of private patients to single rooms. We provided closed formulas to check an instance's feasibility and to compute the maximum number of single-room requests that can be fulfilled. The computation time of those formulas is only a fraction of the time needed to build a corresponding IP. This is of special interest, e.g., in the context of appointment scheduling in hospitals.

We further explored the performance of different IP-formulations. One of our key insights here is the significant performance gap between objectives f^{trans} and f^{priv} which needs to be taken into account when designing IP-formulations. Using all our insights, we proposed a fast IP-based solution approach that obtains high quality solutions which showcases the benefits of combinatorial insights for developing solution approaches. For an extensive and reproducible computational study, we provide a large artificial data set that we generated based on our real-world data. Our computational study showed that even though PRA

Table B.2
Results of dynamic algorithm (365 iterations). All runtimes are given in seconds.

Instance	Runtime per iteration		Total	Objective value		$s^{\max}$	$\frac{f^{\text{priv}}}{s^{\max}}$
	Mean	Max	runtime	f^{trans}	f^{priv}		(%)
real_1	0.2	1.0	101	22	1154	1154	100
real_2	1.2	11.1	473	48	1215	1230	98
real_3	1.4	11.1	536	80	1211	1229	98
real_4	0.2	2.3	80	78	483	488	98
real_5	1.3	30.1	499	42	921	931	98
real_6	0.2	0.6	83	2	851	851	100
real_7	0.0	0.5	23	0	66	66	100
real_8	0.1	0.6	52	0	546	546	100
real_9	0.0	0.1	17	0	0	0	n.a.n.
real_10	0.2	0.7	90	1	751	751	100
real_11	0.1	0.2	56	0	376	376	100
real_12	0.2	0.4	90	3	887	887	100
real_13	0.2	0.6	81	3	556	556	100
real_14	0.3	0.5	111	0	1468	1468	100
real_15	0.2	0.6	77	0	432	432	100
real_16	0.0	0.5	30	0	305	305	100
real_17	0.2	0.7	79	0	794	794	100
real_18	0.2	0.8	94	43	706	706	100
real_19	0.1	1.7	67	16	892	893	99
real_20	0.0	0.4	19	0	0	0	n.a.n.
real_21	0.0	0.6	33	17	692	692	100
real_22	0.1	0.6	47	3	372	372	100
real_23	0.1	0.6	54	1	843	843	100
real_24	0.1	0.6	69	0	181	181	100
real_25	0.0	0.0	9	0	0	0	n.a.n.
real_26	0.0	0.1	35	0	235	235	100
real_27	0.1	0.2	50	0	271	271	100
real_28	0.1	0.8	48	11	588	588	100
real_29	0.1	1.6	72	19	879	881	99
real_30	0.0	0.1	21	0	256	256	100
real_31	0.0	0.5	35	9	364	365	99
real_32	0.0	0.2	16	4	69	69	100
real_33	0.2	0.7	84	0	303	305	99
real_34	0.2	0.9	108	27	1687	1687	100
real_35	0.3	1.3	119	11	1765	1765	100
real_36	0.0	0.3	34	7	82	82	100
real_37	0.3	11.2	119	37	1975	1977	99
real_38	0.6	11.5	222	46	317	335	94
real_39	0.1	0.3	59	0	391	391	100
real_40	0.1	0.2	57	0	535	535	100
real_41	0.1	0.2	43	0	465	465	100
real_42	0.0	0.1	17	0	167	167	100
real_43	0.1	0.3	67	0	526	526	100
real_44	0.2	0.3	79	0	650	650	100
real_45	0.2	0.3	73	0	1038	1038	100
real_46	0.0	0.1	23	0	213	213	100
real_47	0.1	0.2	48	1	634	634	100
real_48	0.0	0.1	17	0	35	35	100
real_49	0.3	1.0	113	6	1246	1246	100
real_50	0.3	11.3	130	27	775	781	99
real_51	0.0	0.1	26	0	451	451	100
real_52	0.5	10.9	191	13	3282	3283	99
real_53	0.0	0.3	35	9	52	52	100
real_54	0.1	0.3	39	3	0	0	n.a.n.
real_55	0.1	0.3	38	6	31	31	100
real_56	0.0	0.0	5	0	0	0	n.a.n.
real_57	0.0	0.2	31	13	134	134	100
real_58	0.2	0.5	88	5	970	970	100
real_59	0.1	0.2	51	1	947	947	100
real_60	0.2	4.8	90	46	1622	1624	99
real_61	0.4	1.3	148	16	1891	1891	100
real_62	0.1	0.7	64	12	451	451	100
load_50_1	0.1	0.2	39	0	591	591	100
load_50_2	0.1	0.2	49	0	756	756	100
load_50_3	0.1	0.2	42	0	803	803	100
load_50_4	0.1	0.4	56	5	974	974	100
load_50_5	0.1	0.2	40	0	859	859	100
load_50_6	0.1	0.2	42	0	758	758	100
load_50_7	0.1	0.3	44	1	896	896	100
load_50_8	0.1	10.3	58	0	914	914	100

(*continued on next page*)

Table B.2 (*continued*).

load_50_9	0.4	10.9	172	140	790	794	99
load_50_10	0.1	0.3	61	0	755	755	100
load_50_11	0.7	11.5	267	186	961	969	99
load_50_12	0.8	11.5	299	164	1030	1041	98
load_50_13	0.1	0.4	65	0	939	939	100
load_50_14	0.1	0.3	60	0	794	794	100
load_50_15	0.1	0.3	61	0	727	727	100
load_50_16	0.3	11.2	119	106	868	868	100
load_50_17	0.1	0.3	66	0	1079	1079	100
load_50_18	0.3	3.3	121	119	929	929	100
load_50_19	0.3	11.3	118	165	499	503	99
load_50_20	0.1	0.2	50	0	756	756	100
load_50_21	0.0	3.9	35	4	502	503	99
load_50_22	0.0	0.2	22	2	543	543	100
load_50_23	0.1	0.6	48	23	605	605	100
load_50_24	0.1	0.5	48	21	732	732	100
load_50_25	0.1	0.5	44	18	570	570	100
load_50_26	0.1	0.4	45	16	446	446	100
load_50_27	0.3	0.9	123	1	1601	1601	100
load_50_28	0.3	0.6	126	1	1702	1702	100
load_50_29	0.1	0.3	66	0	1010	1010	100
load_50_30	0.1	0.3	63	0	1021	1021	100
load_50_31	0.1	0.3	63	0	790	790	100
load_50_32	0.3	5.1	128	135	1013	1020	99
load_50_33	0.1	0.2	60	0	926	926	100
load_50_34	0.2	0.4	75	0	1181	1181	100
load_50_35	0.4	11.0	172	148	1166	1168	99
load_50_36	0.1	0.5	67	1	991	991	100
load_50_37	0.1	0.3	61	0	922	922	100
load_50_38	0.1	0.3	62	0	959	959	100
load_50_39	0.1	0.2	36	0	514	514	100
load_50_40	0.1	0.6	41	2	615	615	100
load_50_41	0.1	0.3	46	0	729	729	100
load_50_42	0.1	0.2	48	2	895	895	100
load_50_43	0.1	0.2	43	0	689	689	100
load_50_44	0.1	0.2	44	0	739	739	100
load_50_45	0.1	0.2	48	0	689	689	100
load_50_46	0.1	0.2	50	0	739	739	100
load_50_47	0.1	0.2	52	0	729	729	100
load_50_48	0.1	0.3	53	4	903	903	100
load_50_49	0.0	0.2	22	1	401	402	99
load_50_50	0.0	0.4	22	3	361	361	100
load_50_51	0.0	1.1	26	7	459	459	100
load_50_52	0.0	0.2	22	3	402	402	100
load_50_53	0.2	10.2	78	3	693	695	99
load_50_54	1.5	91.2	571	174	318	391	81
load_50_55	0.0	1.5	34	10	479	483	99
load_50_56	0.0	0.5	26	13	672	672	100
load_50_57	0.0	0.9	25	3	481	481	100
load_50_58	0.0	0.1	23	4	379	379	100
load_50_59	0.0	0.1	22	1	478	478	100
load_50_60	0.0	0.4	24	3	578	578	100
load_50_61	0.0	0.1	23	2	321	321	100
load_50_62	0.0	1.3	27	2	396	396	100
load_50_63	0.0	0.4	22	3	339	339	100
load_50_64	0.0	0.3	23	4	500	500	100
load_50_65	0.1	0.3	70	0	1036	1036	100
load_50_66	0.1	0.3	67	0	927	927	100
load_50_67	0.1	0.3	38	3	620	620	100
load_50_68	0.1	0.2	41	2	629	629	100
load_50_69	0.1	0.2	39	0	718	718	100
load_50_70	0.1	0.5	43	31	457	457	100
load_50_71	0.1	0.2	39	0	497	497	100
load_50_72	0.1	0.2	40	0	541	541	100
load_50_73	0.1	0.2	38	0	610	610	100
load_50_74	0.1	0.2	38	0	586	586	100
load_50_75	0.0	1.1	36	79	384	430	89
load_50_76	0.0	0.2	22	32	131	131	100
load_50_77	0.1	0.4	38	36	509	509	100
load_50_78	0.1	0.3	37	43	520	520	100
load_50_79	0.0	0.9	30	6	444	452	98
load_50_80	0.0	0.5	22	1	341	341	100
load_65_1	0.3	10.5	137	35	851	852	99
load_65_2	0.2	10.5	102	22	755	759	99
load_65_3	0.2	10.5	77	17	956	957	99
load_65_4	0.7	11.8	276	42	1110	1113	99
load_65_5	0.4	10.7	168	35	789	792	99
load_65_6	0.2	11.0	81	9	779	779	100
load_65_7	0.1	3.7	60	25	752	753	99
load_65_8	0.4	10.7	151	38	1092	1096	99
load_65_9	0.2	0.6	76	7	1171	1171	100
load_65_10	0.4	11.1	164	184	1026	1029	99
load_65_11	1.1	12.4	430	251	1427	1447	98
load_65_12	0.7	11.5	272	193	1063	1072	99
load_65_13	0.0	1.0	34	34	442	443	99
load_65_14	0.1	3.3	64	39	577	588	98
load_65_15	0.1	0.6	54	26	858	858	100
load_65_16	0.1	0.2	45	1	830	830	100
load_65_17	0.1	0.5	45	22	428	428	100
load_65_18	0.1	0.4	48	21	705	705	100
load_65_19	0.1	0.5	69	7	1115	1115	100
load_65_20	0.2	0.6	73	17	1128	1128	100
load_65_21	0.5	11.6	189	183	1407	1414	99
load_65_22	0.2	0.8	74	12	941	941	100
load_65_23	0.3	7.3	137	148	1025	1028	99
load_65_24	0.4	12.0	162	141	1160	1167	99
load_65_25	0.2	2.2	83	19	1232	1232	100
load_65_26	0.2	0.9	79	11	1164	1164	100
load_65_27	0.1	3.0	65	14	1004	1004	100
load_65_28	0.1	0.8	70	18	931	931	100
load_65_29	0.2	0.9	86	13	1407	1407	100
load_65_30	0.2	0.6	73	12	1183	1183	100
load_65_31	0.1	0.6	70	7	1171	1171	100
load_65_32	0.2	0.6	74	10	1420	1420	100
load_65_33	0.4	11.0	168	139	1204	1207	99
load_65_34	0.3	3.2	116	126	996	997	99
load_65_35	0.5	12.0	202	160	1405	1409	99
load_65_36	0.3	5.4	129	135	1093	1095	99
load_65_37	0.1	0.6	72	2	1098	1098	100
load_65_38	0.2	0.6	77	11	1144	1144	100
load_65_39	0.3	1.1	133	15	1771	1771	100
load_65_40	1.7	12.5	638	116	2172	2186	99
load_65_41	0.1	0.3	47	20	786	786	100
load_65_42	0.7	15.7	282	209	393	411	95
load_65_43	0.1	0.7	57	15	974	974	100
load_65_44	0.1	3.7	59	17	940	940	100
load_65_45	0.1	0.4	54	10	963	963	100
load_65_46	0.2	10.7	105	17	996	996	100
load_65_47	0.1	0.7	62	15	1014	1014	100
load_65_48	0.1	0.7	67	17	991	991	100
load_65_49	0.5	11.2	216	197	825	837	98
load_65_50	0.1	1.6	65	22	1066	1068	99
load_65_51	0.1	1.8	39	33	500	500	100
load_65_52	0.0	0.6	29	17	483	485	99
load_65_53	0.1	2.2	51	33	593	595	99
load_65_54	0.1	2.5	43	27	467	471	99
load_65_55	0.1	2.3	66	25	512	516	99
load_65_56	0.1	10.1	60	32	659	668	98
load_65_57	0.0	0.9	29	24	436	438	99
load_65_58	0.1	1.4	37	20	443	448	98
load_65_59	0.0	0.9	33	21	554	554	100
load_65_60	0.0	1.0	35	22	424	428	99
load_65_61	0.1	1.6	42	32	506	514	98
load_65_62	0.1	2.8	39	19	587	589	99
load_65_63	0.1	1.3	39	32	657	660	99
load_65_64	0.1	1.9	39	13	489	495	98
load_65_65	0.2	0.7	88	14	1375	1375	100
load_65_66	0.2	0.7	83	14	1246	1246	100
load_65_67	0.1	0.3	44	12	697	697	100
load_65_68	0.1	0.5	58	83	769	769	100
load_65_69	0.1	0.3	44	18	797	797	100
load_65_70	0.1	0.6	62	71	808	808	100
load_65_71	0.1	0.3	45	13	869	869	100
load_65_72	0.1	0.6	55	57	672	672	100
load_65_73	0.1	0.4	46	4	680	680	100
load_65_74	0.1	0.4	48	11	642	642	100
load_65_75	0.0	0.5	35	89	463	489	94

(*continued on next page*)

Table B.2 (*continued*).

load_65_76	0.0	0.4	34	88	345	358	96
load_65_77	0.1	0.4	38	61	523	523	100
load_65_78	0.0	0.2	36	5	478	478	100
load_65_79	0.2	6.4	80	10	437	455	96
load_65_80	0.1	0.9	43	24	429	441	97
load_80_1	1.1	11.0	420	99	886	894	99
load_80_2	1.4	11.5	544	89	819	842	97
load_80_3	0.9	12.2	342	97	915	924	99
load_80_4	2.0	11.5	754	103	776	791	98
load_80_5	0.8	10.9	299	85	740	750	98
load_80_6	0.8	11.1	325	111	1005	1013	99
load_80_7	1.2	10.7	456	119	717	733	97
load_80_8	1.2	10.7	469	101	875	888	98
load_80_9	0.2	0.9	98	54	1505	1505	100
load_80_10	0.8	11.9	309	244	1316	1327	99
load_80_11	1.3	13.3	487	288	1464	1479	98
load_80_12	0.3	10.7	121	56	1317	1317	100
load_80_13	0.8	10.5	300	87	450	519	86
load_80_14	0.1	4.3	60	87	393	407	96
load_80_15	0.1	0.5	60	39	1000	1000	100
load_80_16	0.1	0.6	58	32	1038	1038	100
load_80_17	0.1	0.5	53	35	741	741	100
load_80_18	0.1	0.3	47	11	786	786	100
load_80_19	0.4	11.0	174	149	1386	1390	99
load_80_20	0.5	11.2	205	230	1583	1587	99
load_80_21	0.2	0.9	102	65	1193	1193	100
load_80_22	0.4	11.0	175	171	1234	1238	99
load_80_23	0.5	10.9	201	221	1627	1636	99
load_80_24	0.5	12.5	205	189	1460	1464	99
load_80_25	0.5	11.4	191	76	1364	1369	99
load_80_26	0.5	12.8	206	93	1173	1178	99
load_80_27	1.0	13.3	373	333	999	1029	97
load_80_28	1.1	11.3	412	294	912	958	95
load_80_29	1.4	15.7	533	122	1688	1696	99
load_80_30	0.3	6.1	142	18	1509	1509	100
load_80_31	0.2	1.9	109	80	1562	1562	100
load_80_32	0.3	11.2	132	147	1508	1510	99
load_80_33	0.3	1.9	127	177	1183	1187	99
load_80_34	0.4	7.7	147	190	1481	1484	99
load_80_35	0.4	4.0	161	193	1529	1534	99
load_80_36	0.4	6.7	146	190	1364	1365	99
load_80_37	0.2	0.8	82	61	1310	1310	100
load_80_38	0.3	3.1	132	183	1314	1316	99
load_80_39	0.5	11.4	216	188	1571	1578	99
load_80_40	0.5	9.5	188	201	1616	1624	99
load_80_41	0.3	10.6	127	113	962	966	99
load_80_42	0.7	12.8	276	315	464	492	94
load_80_43	0.4	10.7	159	72	766	769	99
load_80_44	0.4	10.6	174	56	753	757	99
load_80_45	0.8	13.0	303	89	828	834	99
load_80_46	0.4	10.8	174	58	939	942	99
load_80_47	0.4	11.2	174	242	876	882	99
load_80_48	0.2	5.0	74	73	984	984	100
load_80_49	0.4	11.0	159	139	1090	1095	99
load_80_50	0.5	11.0	185	227	972	980	99
load_80_51	0.2	10.4	94	79	510	523	97
load_80_52	0.1	4.6	54	62	377	393	95
load_80_53	0.2	6.7	84	67	314	319	98
load_80_54	0.1	5.3	64	55	471	477	98
load_80_55	0.2	4.3	85	75	484	500	96
load_80_56	0.2	4.7	99	61	463	472	98
load_80_57	0.2	10.1	89	78	504	519	97
load_80_58	0.2	8.1	93	66	472	476	99
load_80_59	0.3	10.3	136	63	489	507	96
load_80_60	0.2	7.7	109	57	435	457	95
load_80_61	0.1	2.7	72	54	423	432	97
load_80_62	0.1	3.0	68	61	444	455	97

(*continued on next page*)

Table B.2 (*continued*).

load_80_63	0.3	10.3	110	71	392	405	96
load_80_64	0.1	1.5	41	55	432	437	98
load_80_65	0.5	11.4	197	162	1211	1214	99
load_80_66	0.4	11.9	165	97	1196	1198	99
load_80_67	0.1	0.5	62	98	938	938	100
load_80_68	0.1	0.5	61	85	987	987	100
load_80_69	0.1	0.8	69	85	769	769	100
load_80_70	0.1	0.6	70	95	997	997	100
load_80_71	0.1	0.5	59	58	886	886	100
load_80_72	0.1	0.4	51	39	930	930	100
load_80_73	0.1	0.5	63	87	867	867	100
load_80_74	0.1	0.5	54	42	967	967	100
load_80_75	0.1	4.6	58	104	804	845	95
load_80_76	0.1	0.6	41	60	621	654	94
load_80_77	0.1	0.5	46	57	730	730	100
load_80_78	0.1	0.9	53	63	864	879	98
load_80_79	0.2	3.4	97	46	445	474	93
load_80_80	0.5	7.6	215	48	489	541	90
load_95_1	3.1	14.7	1141	206	575	607	94
load_95_2	2.3	11.0	864	178	530	566	93
load_95_3	2.9	11.9	1089	162	570	599	95
load_95_4	4.8	12.2	1757	229	602	653	92
load_95_5	2.9	12.6	1059	181	603	633	95
load_95_6	2.4	11.7	885	179	509	535	95
load_95_7	2.0	11.4	746	148	498	520	95
load_95_8	3.9	11.6	1428	175	514	543	94
load_95_9	1.2	12.4	460	306	1618	1638	98
load_95_10	0.9	12.1	340	313	1573	1584	99
load_95_11	1.5	14.8	577	376	1819	1847	98
load_95_12	0.9	12.7	363	359	1807	1841	98
load_95_13	0.6	10.2	255	129	421	452	93
load_95_14	0.9	10.5	329	113	357	398	89
load_95_15	0.1	0.6	66	39	889	889	100
load_95_16	0.1	0.7	70	36	1256	1256	100
load_95_17	0.1	0.4	54	27	831	831	100
load_95_18	0.1	0.4	52	45	815	815	100
load_95_19	0.4	11.1	179	204	1602	1610	99
load_95_20	0.4	3.9	169	217	1817	1821	99
load_95_21	0.4	8.1	166	197	1632	1636	99
load_95_22	0.3	3.2	144	202	1467	1470	99
load_95_23	0.3	2.4	145	193	1519	1519	100
load_95_24	0.4	4.0	180	218	1699	1706	99
load_95_25	1.2	12.7	447	339	1039	1077	96
load_95_26	0.7	11.0	277	318	957	981	97
load_95_27	1.0	14.3	370	314	1040	1083	96
load_95_28	1.8	16.2	688	371	1219	1296	94
load_95_29	0.4	10.1	169	203	1646	1648	99
load_95_30	0.4	4.4	174	200	1615	1626	99
load_95_31	0.5	5.6	195	229	1803	1805	99
load_95_32	0.5	11.2	204	217	1785	1794	99
load_95_33	0.3	1.9	140	186	1400	1401	99
load_95_34	0.7	11.4	277	254	1636	1646	99
load_95_35	0.4	11.3	182	225	1877	1884	99
load_95_36	0.5	11.0	202	223	1806	1817	99
load_95_37	0.7	11.1	258	277	2123	2133	99
load_95_38	0.6	13.5	240	213	1883	1888	99
load_95_39	4.7	36.3	1722	325	940	958	98
load_95_40	3.6	19.0	1345	312	896	915	97
load_95_41	1.1	11.5	411	330	568	586	96
load_95_42	0.5	10.6	202	334	595	610	97
load_95_43	1.7	16.5	627	175	482	488	98
load_95_44	1.1	15.5	432	161	509	515	98

(*continued on next page*)

is $\mathcal{NP}$-hard, the PRA problem can be solved to optimality or at least close to optimality for realistically sized instances in reasonable time. Hence, our solution approach is a good basis for future integration of other objective functions that are needed for application in real-life. Finally, we elaborated on numerous possibilities for future work.

CRediT authorship contribution statement

Tabea Brandt: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Christina Büsing:** Supervision, Project administration, Methodology, Funding acquisition, Conceptualization. **Felix Engelhardt:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization.

Table B.2 (*continued*).

load_95_45	1.7	12.5	622	155	526	536	98
load_95_46	1.4	12.0	535	181	516	524	98
load_95_47	0.7	10.9	276	266	1102	1112	99
load_95_48	0.5	11.2	195	249	1008	1015	99
load_95_49	0.5	11.0	201	193	1221	1227	99
load_95_50	0.6	11.9	221	268	1120	1130	99
load_95_51	0.6	10.1	249	114	378	419	90
load_95_52	0.4	10.4	180	119	374	394	94
load_95_53	0.6	10.3	250	126	385	404	95
load_95_54	0.2	3.5	85	113	333	346	96
load_95_55	0.3	10.1	140	109	373	393	94
load_95_56	1.1	15.3	432	257	848	866	97
load_95_57	0.4	10.2	150	112	374	391	95
load_95_58	0.3	2.7	113	103	349	368	94
load_95_59	0.5	11.1	198	100	312	341	91
load_95_60	0.3	11.9	143	113	420	435	96
load_95_61	0.5	10.6	184	93	430	463	92
load_95_62	0.6	10.9	223	105	404	423	95
load_95_63	0.9	10.4	361	119	461	491	93
load_95_64	0.6	10.3	251	89	382	397	96
load_95_65	0.5	6.7	202	207	1588	1592	99
load_95_66	0.3	1.7	116	112	1703	1703	100
load_95_67	0.1	0.6	70	107	1181	1181	100
load_95_68	0.1	0.6	66	97	1043	1043	100
load_95_69	0.1	0.5	65	103	938	938	100
load_95_70	0.2	0.6	74	116	1309	1309	100
load_95_71	0.1	0.6	71	127	1100	1100	100
load_95_72	0.1	0.5	64	96	1070	1070	100
load_95_73	0.1	0.7	63	107	1093	1093	100
load_95_74	0.1	0.6	68	102	937	937	100
load_95_75	0.1	0.8	56	160	745	824	90
load_95_76	0.1	0.8	62	137	742	885	83
load_95_77	0.1	0.4	45	65	1009	1009	100
load_95_78	0.1	0.4	46	66	690	690	100
load_95_79	0.4	3.0	151	56	456	503	90
load_95_80	0.6	12.5	241	95	276	337	81

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We thank the team at RWTH Aachen University hospital for their support. We thank Jens Brandt for his input regarding efficient implementation and coding. Simulations were performed with computing resources granted by RWTH Aachen University.

Appendix A. Datasets

In this section, we report statistical information about our artificial instances. The corresponding information about the real-world instances is subject to non-disclosure. In the following tables, there are information for every instance about the total number of patients, as well as the number of female, male, private, and emergency patients. The average occupancy rate is specified in the instance's name. We further report the median of patients' length-of-registration (lor) and length-of-stay (los). Both lor and los are generated using a lognormal distribution. We further report the total number of beds and the number of single/double rooms of each instance (see Table A.1).

Appendix B. Full runtimes and results

In this section, we report statistical details on the solution found by our algorithms both for the 62 real-world instances and artificial instances. The solutions for the artificial instances themselves can be found at Brandt and Engelhardt (2024). In each table we report the mean and maximal computation time needed for one iteration as well as the total runtime needed for all 365 iterations. We further report the objective values found by our algorithm. For an assessment of the solution's quality we report the value $s^{\max}$ for every instance as well as the ratio of how many single-room requests could be fulfilled. Contrary to the real-world instances, we did not check whether there exists a feasible solution without transfers for the artificial instances. Hence, we cannot assess the solution's quality w.r.t. to the number of transfers (see Table B.2).

References

Abdalkareem, Z., Amir, A., Ehkan, P., & Al-Betar, M. (2018). Patients' admission scheduling techniques and approaches. *International Journal of Engineering and Technology(UAE)*, *7*, 3569–3573.

Bastos, L. S., Marchesi, J. F., Hamacher, S., & Fleck, J. L. (2019). A mixed integer programming approach to the patient admission scheduling problem. *European Journal of Operational Research*, *273*(3), 831–840.

Blay, N., Roche, M. A., Duffield, C., & Gallagher, R. D. (2017). Intrahospital transfers and the impact on nursing workload. *Journal of Clinical Nursing*, *26*, 4822–4829.

Blay, N., Roche, M., Duffield, C., & Xu, X. (2017). Intra-hospital transfers and adverse patient outcomes: An analysis of administrative health data. *Journal of Clinical Nursing*, *26*.

Brandt, T., Büsing, C., & Knust, S. (2024). Structural insights about avoiding transfers in the patient-to-room assignment problem. *Discrete Applied Mathematics*, *347*, 231–248.

Brandt, T., & Engelhardt, F. (2024). TabeaBrandt/patient-to-room_assignment: v2 - new benchmark set and results.

Brandt, T., Klein, T. L., Reuter-Oppermann, M., Schäfer, F., Thielen, C., van de Vrugt, M., et al. (2023). Integrated patient-to-room and nurse-to-patient assignment in hospital wards.

Ceschia, S., & Schaerf, A. (2009). Multi-neighborhood local search for the patient admission problem. In *Hybrid metaheuristics*.

Ceschia, S., & Schaerf, A. (2011). Local search and lower bounds for the patient admission scheduling problem. *Computers & Operations Research*, *38*, 1452–1463.

Ceschia, S., & Schaerf, A. (2012). Modeling and solving the dynamic patient admission scheduling problem under uncertainty. *Artificial Intelligence in Medicine*, *56*(3), 199–205.

Ceschia, S., & Schaerf, A. (2014). Dynamic patient admission scheduling with operating room constraints, flexible horizons, and patient delays. *Journal of Scheduling*, *19*(4), 377–389.

Chaudhury, H., Mahmood, A., & Valente, M. (2005). Advantages and disadvantages of single-versus multiple-occupancy rooms in acute care environmentsa review and analysis of the literature. *Environment and Behavior*, *37*, 760–786.

Conforti, D., Guido, R., Mirabelli, G., & Solina, V. (2018a). A decision support service for hospital bed assignment. (pp. 466–473).

Conforti, D., Guido, R., Mirabelli, G., & Solina, V. (2018b). A two-phase integrated approach for elective surgery and bed management problems. (pp. 402–408).

Demeester, P., Souffriau, W., Causmaecker, P. D., & Berghe, G. V. (2010). A hybrid tabu search algorithm for automatically assigning patients to beds. *Artificial Intelligence in Medicine*, *48*(1), 61–70.

Fekieta, R., Rosenberg, A., Hodshon, B., Feder, S., Chaudhry, S., & Emerson, B. (2020). Organisational factors underpinning intra-hospital transfers: a guide for evaluating context in quality improvement. *Health Systems*, *10*, 1–10.

Ficker, A. M. C., Spieksma, F. C. R., & Woeginger, G. J. (2021). The transportation problem with conflicts. *Annals of Operations Research*, *298*(1), 207–227.

Guido, R., Groccia, M. C., & Conforti, D. (2018). An efficient matheuristic for offline patient-to-bed assignment problems. *European Journal of Operational Research*, *268*(2), 486–503.

Hantel, S., & Benkenstein, M. (2019). Roommates in hospitals - A new and relevant dimension of health care quality models. *Journal of Service Management Research*, *3*, 82–90.

He, J., Gallego, B., Stubbs, C., Scott, A., Dawson, S., Forrest, K., et al. (2018). Improving patient flow and satisfaction: An evidence-based pre-admission clinic and transfer of care pathway for elective surgery patients. *Collegian*, *25*(2), 149–156.

Hendrich, A., & Lee, N. (2005). Intra-unit patient transports: time, motion, and cost impact on hospital efficiency. *Nursing Economics*, *23 4*, 157–64, 147.

Johnson, D., Schmidt, U., Bittner, E., Christensen, B., Levi, R., & Pino, R. (2013). Delay of transfer from the intensive care unit: A prospective observational study of incidence, causes, and financial impact. *Critical Care (London, England)*, *17*, R128.

Liu, H., Wang, Y., & Hao, J.-K. (2024). Solving the patient admission scheduling problem using constraint aggregation. *European Journal of Operational Research*, *316*(1), 85–99.

Noonan, F., O'Brien, J., Broderick, E., Richardson, I., & Walsh, J. (2019). Hospital bed management practices: A review. In *Proceedings of the 12th international joint conference on biomedical engineering systems and technologies - HEALTHINF* (pp. 326–331). SciTePress.

Ouelhadj, D., & Petrovic, S. (2009). A survey of dynamic scheduling in manufacturing systems. *Journal of Scheduling*, *12*, 417–431.

Papson, J. P., Russell, K. L., & Taylor, D. M. (2007). Unexpected events during the intrahospital transport of critically ill patients. *Academic Emergency Medicine*, *14*(6), 574–577.

Rachuba, S., Reuter-Oppermann, M., & Thielen, C. (2023). Integrated planning in Hospitals: A review.

Range, T. M., Lusby, R. M., & Larsen, J. (2014). A column generation approach for solving the patient admission scheduling problem. *European Journal of Operational Research*, *235*(1), 252–264.

Schäfer, F., Walther, M., Grimm, D., & Hübner, A. (2023). Combining machine learning and optimization for the operational patient-bed assignment problem. *Health Care Management Science*, 1–22.

Schäfer, F., Walther, M., Hübner, A., & Kuhn, H. (2019). Operational patient-bed assignment problem in large hospital settings including overflow and uncertainty management. *Flexible Services and Manufacturing Journal*, *31*.

Schmidt, R., Geisler, S., & Spreckelsen, C. (2013). Decision support for hospital bed management using adaptable individual length of stay estimations and shared resources. *BMC Medical Informatics and Decision Making*, *13*, 3.

Storfjell, J., Ohlson, S., Omoike, O., Fitzpatrick, T., & Wetasin, K. (2009). Non-value-added time the million dollar nursing opportunity. *The Journal of Nursing Administration*, *39*, 38–45.

T, B., Bollapragada, S., Akbay, K., Toledano, D., Katlic, P., Dulgeroglu, O., et al. (2013). Automated bed assignments in a complex and dynamic hospital environment. *Interfaces*, *43*, 435–448.

Turhan, A., & Bilgen, B. (2016). Mixed integer programming based heuristics for the Patient Admission Scheduling problem. *Computers & Operations Research*, *80*.

UKA Aachen (2021). Fees schedule for university hospital aachen. Accessed: 2023-11-29.

Vancroonenburg, W., Croce, F. D., Goossens, D., & Spieksma, F. C. R. (2014). The Red–Blue transportation problem. *European Journal of Operational Research*, *237*(3), 814–823.

Vancroonenburg, W., De Causmaecker, P., & Vanden Berghe, G. (2016). A study of decision support models for online patient-to-room assignment planning. *Annals of Operations Research*, *239*(1), 253–271.