Constraint-Based Optimization for Scheduling Medical Appointments

George Assaf[a], Sven Löffler and Petra Hofstedt
Programming Languages and Compiler Construction Chair, Brandenburg University of Technology,
BTU Cottbus-Senftenberg, Konrad-Wachsmann Allee 5, Cottbus, Germany
{*george.assaf, sven.loeffler, hofstedt*}*@b-tu.de*

Keywords: Constraint Programming, CSP Model, Optimized Scheduling, Medical Appointment Scheduling Problem.

Abstract: In this paper, we introduce a novel approach for solving as well as optimizing the medical appointment scheduling problem (MASP) using constraint programming. The MASP is a complex and critical task in health care management, directly impacting both patient care and operational efficiency. We formalize the MASP as a set of constraints that encode diverse requirements for scheduling medical appointments, including intuitive requirements such as the availability of both patients and medical resources, including physicians, nurses and medical equipment. Furthermore, our model accounts for patient preferences, such as favoring specific dates and/or particular resources whenever feasible. The proposed method incorporates optimization techniques that enhance the scheduling process by considering appointment urgency and balancing workload distribution among the assigned resources, thereby improving the allocation of medical resources. As an outcome, our constraint model demonstrates high efficiency by scheduling medical appointments in just milliseconds.

1 INTRODUCTION

Medical appointment scheduling problem (MASP) within health care facilities is a complex and critical task that directly impacts the quality of patient care and the efficiency of the health care operations (Abdalkareem et al., 2021; Ala and Chen, 2022). As the demand for health care services continues to rise, this calls for effective scheduling of the available medical resources ranging from specialized doctors and nurses to dedicated rooms and medical equipments.

Scheduling medical appointments becomes a challenging task due to both substantially increasing the number of resources within the medical domain and the interdependencies occurring between different resources. For instance, in a large hospital, scheduling a cardiac surgery is a complex task that requires the coordination of several critical resources such as specialized physicians and a room equipped with specialized cardiac surgery equipment. According to our medical partner the *Charité – Universitätsmedizin Berlin*, Europe's largest university hospital, traditional scheduling methods such as spreadsheets often fall short, unable to effectively balance many competing demands and constraints, especially when the search period (scheduling period) as well as the number of specialists and medical devices become substantial.

In this paper, we present a novel constraint-based model for scheduling medical appointments requiring the coordination of multiple resource types. Constraint programming ($\mathcal{CP}$) (Apt, 2003) offers an outstanding approach for this purpose by expressing problem requirements as constraints. On one hand, hard constraints must be strictly satisfied. For instance, scheduling an appointment for a cardiac consultation calls for a cardiologist to be available. On the other hand, soft constraints represent preferences or conditions that are desirable but not mandatory. For example, a patient prefers an appointment to be taken place on a specific date. Furthermore, model optimization focuses on two key objectives: (1) minimizing the starting time of appointments to ensure timely access to medical services, and (2) minimizing the overall workload distribution across available resources to promote a balanced utilization of medical staff and equipment.

The remainder of this paper is organized as follows: The next section introduces the fundamentals of constraint satisfaction and constraint optimization problems, along with a review of relevant prior research. In Section 3, we detail the domains of the medical appointment scheduling problem (MASP), which are milestones to construct our

[a] https://orcid.org/0000-0002-8878-5871

Assaf, G., Löffler, S. and Hofstedt, P.
Constraint-Based Optimization for Scheduling Medical Appointments.
DOI: 10.5220/0013091300003890
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 17th International Conference on Agents and Artificial Intelligence (ICAART 2025) - Volume 3*, pages 94-103
ISBN: 978-989-758-737-5; ISSN: 2184-433X
Proceedings Copyright © 2025 by SCITEPRESS – Science and Technology Publications, Lda.

constraint model. Section 4 then presents the constraint model and its optimization. Computational results for the proposed model are discussed in Section 5. Finally, Section 6 wraps up the paper and explores avenues for future research.

2 PRELIMINARIES

In this section, we recall the fundamentals of $\mathcal{CP}$ including both constraint satisfaction problems and constraint optimization problems ($\mathcal{COP}$). $\mathcal{CP}$ (Apt, 2003) is a powerful paradigm within artificial intelligence (AI) to model and solve complex combinatorial problems by declaring variables and defining constraints over these variables to specify problem requirements.

2.1 Constraint Satisfaction Problems

Constraint satisfaction problems ($\mathcal{CSP}$) (Freuder and Mackworth, 2006) P are formally defined as a 3-tuple $P = \langle X, D, C\rangle$, with:

- $X = \{x_1, x_2, \dots x_n\}$ is a set of variables, namely decision variables.
- $D = (D_{x_1}, D_{x_2}, \dots, D_{x_n})$ is an n-tuple of finite domains, where D_{x_i} represents the set of possible values that the variable $x_i \in X$ may take.
- $C = \{c_1, c_2, \dots, c_m\}$ is a set of constraints, whereby $c_i = (Y, R)$ consists of:
  - Y: A subset of variables $Y \subseteq X$.
  - R: A relation that specifies the allowable combinations of values for the variables in Y.

  For instance, the constraint $x^2 + y^2 \leqslant z$ comprises three variables x, y and z that are governed by the relation $\leqslant$.

Constraints described in natural languages are classified into *hard* and *soft* (Freuder and Mackworth, 2006). *Hard constraints* are constraints that must be strictly satisfied in all solutions, whereas *soft constraints* represent desirable conditions that the solutions should aim to satisfy, but it is not mandatory for them to be fully met.
$\mathcal{CP}$ offers a variety of constraints to model complex problems. Among these, *global constraints* are particularly powerful, as they capture common patterns or relations that frequently occur in many problems (Global Constraint Catalog, 2022). A commonly used *global constraint* (depending on the problem) is *allDifferent*, which ensures that a set of variables are all assigned distinct values. For instance, *allDifferent*($\{x_1, x_2, \dots, x_l\}$) ensures that each variable gets assigned a distinct value.
A solution of a $\mathcal{CSP}$ involves assigning each variable x_i a value from D_{x_i} so that the conjunction of the problem constraints is fulfilled. This implies that a $\mathcal{CSP}$ seeks any solution that satisfies the constraints.

2.2 Constraint Optimization Problems

A *constraint optimization problem* ($\mathcal{COP}$) (Freuder and Mackworth, 2006) extends a $\mathcal{CSP}$ by an objective function f. The goal of a $\mathcal{COP}$ is twofold. First, finding an assignment of values to variables that satisfies all the constraints, similar to a $\mathcal{CSP}$. Second, optimizing (i.e. minimizing or maximizing) the given objective function f for the purpose of finding the best solution according to this function. A $\mathcal{COP}$ is formally defined as a 4-tuple $P_{opt} = \langle X, D, C, f\rangle$, with:

- $P = \langle X, D, C\rangle$ is a $\mathcal{CSP}$.
- $f : D_1 \times D_2 \times \dots D_n \to \mathbb{R}$ a function that assigns a real number (often a cost or utility) to each possible assignment of values to the variables.

Please note that the notation $\mathbb{R}$ denotes the set of real numbers.

A $\mathcal{CSP}/\mathcal{COP}$ is solved by means of a *constraint solver* integrating techniques such as constraint propagation (Bessiere, 2006), i.e. domain reduction and search techniques, e.g. Backtracking (Bitner and Reingold, 1975) for exploring search space. In $\mathcal{COP}$ problems, the constraint solver seeks not only to satisfy the problem constraints but also to refine the solutions to meet an objective function. For instance, the *branch and bound* (Morrison et al., 2016) technique searches for feasible solutions that satisfy all constraints while iteratively improving the objective function.

There are different types of *constraint solvers*, each optimized for specific kinds of problems. For example, linear programming solvers like *Gurobi* and *CPLEX* (IBM ILOG CPLEX Optimization Studio, 2019) are highly efficient for solving constraint satisfaction problems with finite domains that can be expressed as linear equations. In contrast, constraint programming solvers like *Choco* (Régin et al., 2017) and *JaCoP* (Kuchcinski and Szymanek, 2012) are more versatile, handling a broader range of combinatorial problems involving discrete variables (potentially over infinite domains) and non-linear relationships. Tools like *MiniZinc* (Nethercote et al., 2007) offer a high-level modeling language that can be used with various back-end solvers, providing flexibility in choosing the best solver for a particular problem whether the domains are finite or infinite.

Subsequently, we present a $\mathcal{CP}$ model for addressing the MASP. To this end, we adopt the *Choco* solver (Régin et al., 2017), as it offers a wide range of constraints and several search strategies enhancing the efficiency of finding optimal or feasible solutions in large-scale and combinatorial settings.

2.3 Related Work

In this section, we review former efforts made in the field of appointment scheduling, focusing on the various methodologies and approaches that have been developed to address the challenges inherent in this complex problem.

Several constraint models have been proposed in the literature to describe scheduling appointment problems. For example, (Guéret et al., 1996; Rappos et al., 2022) introduced a university timetabling constraint model for scheduling lectures within an educational institute. While this problem is somewhat relevant to appointment scheduling, but it significantly differs from medical appointment scheduling problem. On one hand, doctors may have different working hours. On the other hand, appointment scheduling may be governed by patients' needs such as requesting a preferred specialist over others.

Further, (Topaloglu and Ozkarahan, 2011) introduced a mixed-integer programming (MIP) model for scheduling residents' duty hours in graduate medical education, focusing on adherence to residency program regulations, including on-call nights, days off, rest periods, and total work hours. The master problem aims to minimize deviations from desired service levels during day and night shifts by configuring individual schedules. The addressed problem falls within a very narrow context, as it does not address the broader and more complex challenges of medical appointment scheduling, which involves coordinating appointments across multiple patients and varying medical resources/services. Similarly, (Oliveira et al., 2024) recently developed a constraint model for the nurse scheduling problem, aimed at creating an optimized shift plan for medical staff within a healthcare facility.

Next, in (Hanset et al., 2010), constraint programming was utilized to model the operating theatre scheduling problem, addressing key constraints and objectives, such as scheduling surgeries within available time slots, minimizing resource idle time, and accommodating the availability of surgeons. However, this study overlooks constraints related to other resources and does not fully consider patients' preferences.

Besides, (Vermeulen et al., 2009) developed an adaptive model that dynamically allocates high-cost medical resources like CT-scans across various patient groups, taking into account their differing attributes, such as expected arrival times. The model optimizes resource usage by continuously adjusting the resource calendar based on real-time. While the introduced adaptive model relies heavily on dynamic capacity allocation for a specific resource, it neglects various aspects of appointment scheduling within health care systems.

In contrast to these existing approaches/models, our constraint-based model addresses the medical appointment scheduling problem by integrating a holistic set of constraints that account for the needs of both patients and staff. Additionally, our model takes into account appointment optimization based on two main objectives: minimizing the start time of the next appointment and ensuring a fair distribution of workload among medical resources, e.g. specialists.

3 DOMAIN DESCRIPTIONS

The MASP (Ala and Chen, 2022) comprises various *resources* including *physicians* with different medical specializations, *rooms*, and *medical equipment* of different types. Each of these resources is associated with a calendar sketching both available and occupied time slots over a given scheduling period, i.e. number of days. The goal of the MASP is to find available medical resources (at the same time) that are appropriate for the patient's medical needs, while also striving to accommodate patient preferences when feasible. The complexity of the problem stems from the fact that the search may encompass a substantial number of medical resources, each with its own calendar that may span over a large number of scheduling days, making the scheduling process highly intricate. Crucially, appointment scheduling must account for multiple objectives. These objectives include scheduling the appointment as quickly as possible and fairly distributing the workload across the medical facility's resources. While the first objective addresses the urgency of the appointment, the latter focuses on preventing overburdening any single resource by consistently assigning appointments to that resource. Balancing these objectives is essential to ensure both timely care for patients and sustainable resource management within the medical facility. To construct a constraint model, we first need to determine the variable domains. In this context, we identify two domains: the time slot domain and the resource domain.

Time Slot Domain. This domain describes the available time slots of a resource calendar. The resource calendar is constructed based on the usual *office hours* per week, *official holidays,* and *special vacations* (off days) of the associated resource. The office hours of a resource are given as a matrix whose rows and columns represent the time slots and the week days, respectively, see Figure 1 as an example.

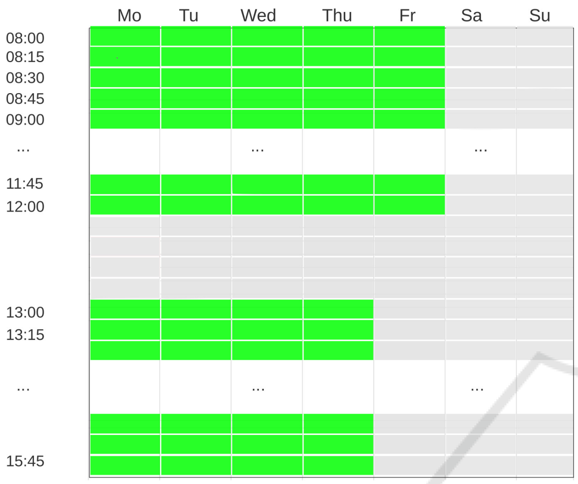


Figure 1: Office hours matrix of a resource per week, i.e. 7 columns starting from 8:00 AM to 16:00, i.e. 32 time slots (rows). Green slots depict those slots at which the given resource is available, whereas the gray slots show the unavailable slots.

Similarly, we model a resource calendar as a matrix, whose rows represent the number of time slots and the columns represent the number of days, during which appointments to be scheduled, i.e. scheduling period. Based on the office hours of a resource, the number of time slots of a resource calender is determined using Equation 1.

$$N = \frac{T_{\text{end}} - T_{\text{start}}}{D}, \qquad (1)$$

where T_{start} and T_{end} are the earliest and latest times (in minutes) at which the resource starts and finishes its usual work, respectively. D is the duration of a time slot. For example, for a resource starting at 8:00 and ending at 16:00 with 15-minute time slots, this yields 32 time slots per day.

Each time slot per day gets assigned an identifier which uniquely identifies the corresponding time slot, if a time slot is free, otherwise it gets assigned a negative value. The availability of a time slot is obtained by the associated office hour matrix. The time slot identifier is determined using Equation 2.

$$T_{id} = \begin{cases} Day \times N + i & \text{if } \texttt{slot} \quad i \quad \texttt{is free} \\ -1 & \text{otherwise} \end{cases} \qquad (2)$$

where *Day* is the day index within the calendar, N the number of time slots per day calculated by Equation 1, and i the corresponding slot index. Figure 2 gives a resource calendar over 14 days. In this example, the time slots between 12:00 and 13:00 are considered as occupied, i.e. -1, as no office hours at this period are given due to, e.g. lunch time, compare Figure 1.

As a slot identifier represents a specific time slot (row index) at a certain day (column index), information sketched in Table 1 can be determined. For instance, the time slot with the identifier $T_{id} = 33$ corresponds to Tuesday 13.08.2024 (assuming the calendar starts on Monday 12.08.2024), which is the second day (i.e., T_{id} *mod* $N = 33$ *mod* $32 = 1$) within the given scheduling period, and corresponds to the second time slot (at 8:15).

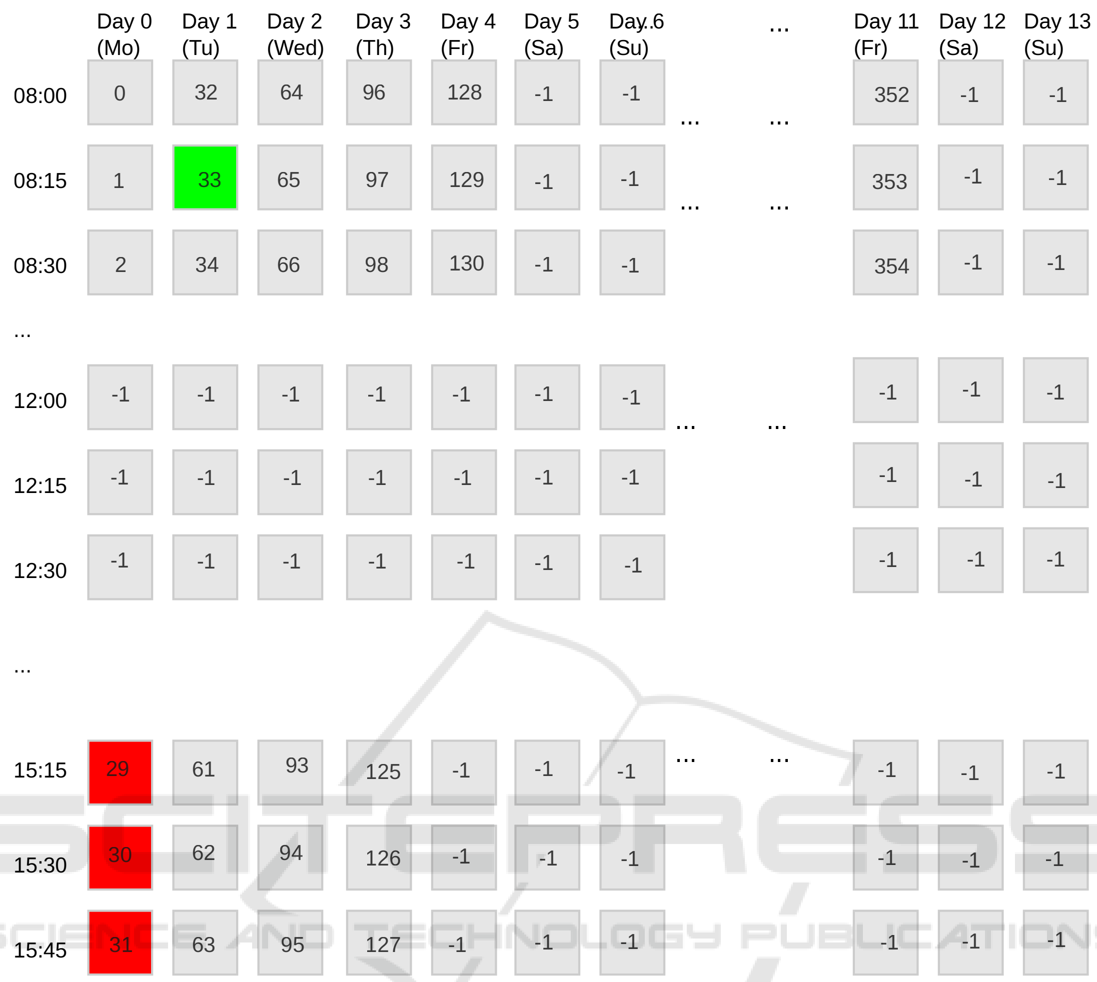
Table 1: Date/time related information determined by time slot identifier.

No.	Info.	Equation	Description
1	Specific Date	$d = \left\lfloor \frac{T_{id}}{N} \right\rfloor$	d gives the corresponding date index.
2	Day in a Week	$d_w = \left\lfloor \frac{T_{id}}{N} \right\rfloor$ mod 7	d_w gives the corresponding day (from Monday to Friday).
3	Time Slot	$t_s = T_{id}$ mod N	t_s gives the corresponding slot index.

For the purpose of reflecting an overall schedule of the facility resources, a global calendar can be constructed by help of the facility resource calendars. Both global start time slot and end time slot of the global calendar are induced by the earliest start time slot and latest end slot among all resource calendars, respectively. To access an appointment scheduled for a resource calendar within the global calendar, the start appointment slot within the global calendar is obtained by the following equation:

$$S_g = \frac{S_l - G_s}{D} \qquad (3)$$

Here, S_g denotes the start slot index of the appointment within the global calendar, S_l is the appointment start slot index within the local calendar, i.e. resource calendar, and G_s is the index of start time slot of the global calendar. For example, assuming $D = 15$ (minutes), if the start of the global calendar is at 7:30 (slot



	Day 0 (Mo)	Day 1 (Tu)	Day 2 (Wed)	Day 3 (Th)	Day 4 (Fr)	Day 5 (Sa)	Day.6 (Su)	...	...	Day 11 (Fr)	Day 12 (Sa)	Day 13 (Su)
08:00	0	32	64	96	128	-1	-1	...	...	352	-1	-1
08:15	1	33	65	97	129	-1	-1	...	...	353	-1	-1
08:30	2	34	66	98	130	-1	-1			354	-1	-1
...												
12:00	-1	-1	-1	-1	-1	-1	-1	...	...	-1	-1	-1
12:15	-1	-1	-1	-1	-1	-1	-1			-1	-1	-1
12:30	-1	-1	-1	-1	-1	-1	-1			-1	-1	-1
...												
15:15	29	61	93	125	-1	-1	-1	...	...	-1	-1	-1
15:30	30	62	94	126	-1	-1	-1			-1	-1	-1
15:45	31	63	95	127	-1	-1	-1			-1	-1	-1

Figure 2: A resource calendar within a medical facility, spanning over two-week period, i.e. 14 days. Identifier assignment respects the resource office hours given in Figure 1. It is worth mentioning that the given resource calendar neglects the appointments that have been previously scheduled for the associated resource.

0 in the global calendar), and the start of a local calendar is at 8:00, then the identifier of the start slot in global calendar would be 2.

Resource Domain. This domain describes the resources offered by a given medical facility. To accommodate the facility resources into a $\mathcal{CSP}$ model, each resource gets assigned a unique identifier. By using resource identifiers, we now are able to define sub-domains from the entire resource domain, induced by all resource identifiers. Each resource sub-domain contains resource identifiers belonging to the same medical specialization for specialists. For example, if a facility comprises n cardiologists, then the cardiologist sub-domain is defined as follows.

$\mathcal{D}_c = \{ID_{c_1}, ID_{c_2}, \ldots, ID_{c_n}\}$

where each element represents an identifier of an involved cardiologist. Similarly, resources representing medical devices with identical medical usage (referred to by medical type) form an individual sub-domain. For example, the sub-domain corresponding to medical equipment of the type *CT-scan* (computed tomography device) is given as follows:

$\mathcal{D}_e = \{ID_{e_1}, ID_{e_2}, \ldots, ID_{e_n}\}$

where each element represents an identifier associated with the medical equipment of type *CT-scan*.

In the same way, other resources such as surgical rooms and nurses are defined.

4 CONSTRAINT MODEL

The goal of the MASP is to find an allocation of available resources for a medical appointment request. In this context, the availability of these resources is determined by their associated calendars. While a $\mathcal{CSP}$

focuses solely on calculating the corresponding resources and time slots, a $\mathcal{COP}$ considers additional optimization goals, such as scheduling the next appointment and ensuring a fair workload distribution among the medical staff.

The input for the problem consists of the calendars for all resources, the search period defined in days, required resource types, and the duration of the required appointment.

To construct a $\mathcal{CSP}$ model for the MASP problem, it is necessary first to determine the decision variables and their corresponding domains.

Decision Variables. Let the tuple $\mathcal{A} = (\mathcal{R}, \mathcal{T})$ represent a medical appointment, where $\mathcal{R} = \{x_1, x_2, \ldots, x_l\}$ is the set of required resources for the appointment, and $\mathcal{T} = \{t_1, t_2, \ldots, t_m\}$ denotes time slot identifiers that are needed to cover the required duration. The time slot length m can be intuitively determined by dividing the required appointment duration by the slot duration. For example, $\mathcal{A} = (\{ID_{c_3}, ID_{e_1}\}, \{29, 30, 31\})$ which denotes an appointment requiring both a cardiologist identified by ID_{c_3} and a CT-scan device identified by ID_{e_1} for a period of time covering three consecutive 15-minute time slots, i.e. 45-minute appointment starting at 15:15. Let the calendar shown in Figure 2 represent the calendar of the resource identified by ID_{c_3}. This means that the slot identifier 29 corresponds to the starting time slot of the appointment involving ID_{c_3}.
Let x_i represent the decision variable for the i-th required resource (e.g. a medical specialization or medical type (see Section 3). The sub-domain of x_i, denoted by $\mathcal{D}(x_i)$, is the sub-set of resource IDs belonging to similar resource type (medical specialization for doctors or medical types for equipment). This can be expressed as:

$$x_i \in \mathcal{D}(x_i) \quad \text{for } i = 1, 2, \ldots, l$$

where $\mathcal{D}(x_i) = \{ID_1, ID_2, \ldots, ID_n\}$ is the set of IDs corresponding to all resources that can fulfill the type requirement of the i-th resource, l denotes the number of required resources for appointment $\mathcal{A}$ and n is the number of resources with similar type.
Let t_j be a variable representing a time slot of appointment $\mathcal{A}$. The domain of the variable t_j, denoted by $\mathcal{D}(t_j)$, is the set of all available time slots within the calendar of the resource x_i, given as follows:

$$t_j \in \mathcal{D}(t_j) \quad \textit{for } j = 0, 1, \ldots, m$$

where $\mathcal{D}(t_j) = \{0, 1, \ldots, s\}$ represents the set of the identifiers of the available time slots within the calendars of all required resources in $\mathcal{R}$. Please note that s denotes the last slot identifier of a resource calendar.

Model Constraints. To best of our knowledge, the following constraints have to be taken into account for a typical MASP.

1. **Assigning the Right Resources.** For instance, scheduling an appointment for a patient diagnosed with cardiology issues calls for one or more cardiologists. This constraint is ensured by restricting the resource variables in $\mathcal{R}$ to take values from their corresponding sub-domains based on the required resources.
2. **Resource Uniqueness.** Since an appointment may require multiple resources simultaneously, it is crucial that no resource is assigned to the same appointment more than once. This is enforced by the *allDifferent* constraint as follows.

$$\textit{allDifferent}(\{x_1, x_2, \ldots, x_l\})$$

3. **Resource Availability** This constraint is enforced by limiting the domains of the variables $t_j \in \mathcal{T}$, representing time slots, to those that are available, i.e. not occupied within each involved resource calendar.
4. **Ensuring a Substantial Appointment Duration.** The constraint is satisfied by designating a number of time slots equal to m, induced by dividing the required appointment duration by the predefined time slot duration within a resource calendar.
5. **Consecutive Time Slots.** The time slots allocated for an appointment must be consecutive within a day to ensure that the required duration is covered. This constraint implicitly covers the constraint given in the former item (No. 4). This constraint is formulated as an *arithmetic constraint*[1], expressed as follows:

$$t_j = t_{j-1} + 1 \quad \text{for } j = 2, 3, \ldots, m$$

where m is the total number of slots needed to cover the required appointment duration.
6. **Preferred Resource Assignment.** This constraint is treated as a soft constraint, reflecting the fact that patients may prefer certain resources (e.g. a specific specialized doctor), and thus at least one of the preferred resources should be chosen. However, if the preferred resource is unavailable, an alternative must be chosen. To formulate this con-

[1]An arithmetic constraint is a mathematical expression that enforces a specific condition using arithmetic operations like addition, subtraction, multiplication, or division.

straint, let $y_{j,k}$ be a binary variable [2] that equals 1 if resource k is selected for resource type j, and 0 otherwise. For each resource type j, if there are preferred resources in the set Q_j, the model encourages the selection of one of these preferred resources:

$$\sum_{k \in Q_j} y_{j,k} \geqslant 1 \quad \text{for } x_j \in \mathcal{R} \text{ where } Q_j \neq \varnothing$$

Here, Q_j is the set of preferred resources for resource type j. This constraint is implemented by creating binary variables corresponding to resource assignments and ensuring that at least one preferred resource is selected. Please note that this is a soft constraint, as the preferred resource may not always be available.

7. **Eliminating Dates on Which a Patient Can not Schedule an Appointment.** This constraint ensures that no appointment will be scheduled on dates when the patient is unavailable. Let B denote the set of time slot identifiers that correspond to the unavailable dates (refer to Table 1 for the identification of these slots). To prevent appointments from being scheduled on these dates, the starting time slot of the appointment must not fall within the undesired dates. This can be described as follows:

$$t_1 \notin B$$

where t_1 is the starting time slot of the appointment.

8. **Scheduling an Appointment on a Date Preferred by the Patient.** This constraint represents a soft constraint that seeks to schedule the patient's appointment on one of the preferred dates. Let $\mathcal{P}$ be the set of preferred dates by the patient and the variable x_p indicates whether a resource is able to schedule the appointment on a preferred date p. To encourage this preference, the model minimizes the following sum over all days (i.e. dates) p that are not in $\mathcal{P}$:

$$\sum_{p \notin \mathcal{P}} x_p$$

which represents the number of times the appointment is scheduled outside the preferred dates. We introduce the binary variable brokenDateVar to capture violations of this preference. Minimizing brokenDateVar ensures that the patient's appointment is scheduled on a preferred date whenever possible, but the constraint remains flexible to account for resource availability.

The $\mathcal{CSP}$ is now fully defined, typically yielding numerous potential solutions. To identify a most optimal solution among these, we proceed with the optimization function.

Model Optimization. In the context of MASP, we consider two objectives. First, finding the solution providing the next appointment as soon as possible, i.e. the solution with minimalistic first time slot. Second, finding the solution which fairly distribute workload among the resources, e.g. a cardiologist with less workload.

Minimizing the start time of an appointment and minimizing the workload of resources are two distinct goals that may not align naturally. If each optimized individually, we might find a solution that is optimal for one objective but suboptimal for another. Therefore, we combine the objectives, leading to a more holistic solution that considers both the efficiency of the schedule (early start time) and the fairness for workload distribution across resources (workload minimization).

Minimizing starting time slot t_1 is expressed as follows.

$$\text{Minimize} \quad t_1 \times \text{TimeSlotWeight}$$

where t_1 is the starting time slot of the appointment and $TimeSlotWeight$ is a weight assigned to prioritize the minimization of the starting time slot.

To optimize the workload, we introduce a set of variables p_i representing workload penality associated with assigning resource x_i. The penality is calculated based on the current workload of a potential resource, i.e. time amount which already assigned over search period, scaled by a penality factor. For each resource x_i and potential resource r_j in its domain:

$$\text{if} \quad x_j = r_j, \quad \text{then} \quad p_i = \text{penaltyFactor} \times \text{workload}(r_j)$$

where $\text{workload}(r_j)$ is the assigned time amount induced by scheduled appointments for resource r_j. Then, the total penality variable $totalPenalty$ across all resources is defined as:

$$\text{totalPenalty} = \sum_{i=1}^{N} p_i$$

The objective function to be minimized is

$$\text{Minimize} \quad \text{totalPenalty} \times \text{WorkloadWeight}$$

[2]A binary variable, also called *BoolVar* in the *Choco Solver* library, is a variable that can take only two values: 0 (false) or 1 (true). It is commonly used in constraint programming to represent boolean conditions and is useful for tracking the satisfaction or violation of constraints.

where WorkloadWeight is a weight assigned to prioritize the minimization of the workload penalties.

The combined objective function f to be minimized:

$$\text{Minimize} \quad f$$

$$= t_1 \times \text{TimeSlotWeight} + \text{WorkloadWeight} \times \text{totalPenalty}$$

5 MODEL EVALUATION AND RESULTS

We performed two experiments to evaluate the model's performance and its behavior.

Experiment 1. In this experiment, we aim to explore the performance of the constraint model by measuring two key metrics as the problem size varied, i.e. problem scalability. These metrics are: the time spent to solve the MASP problem, referred to by resolution time (RT) and the time taken by the $\mathcal{CSP}$ solver to reach the optimal solution, referred to as the time to best solution (TTBS). Due to data privacy regulations enforced by our medical partner, we conducted our experiments using a fictional medical facility. The experiments were conducted on a Ubuntu 22.04.4 LTS machine equipped with 14GB of RAM and an 8-core AMD Ryzen 2.02 GHz. Additionally, we utilized the *Choco* solver library, version 4.10.8.

We assessed the model's performance by gradually increasing the problem size. Specifically, the size of the MASP is defined by either the number of scheduling days or the number of resources involved. To this end, we conducted two experiments. In both experiments, all resource calendars have 80% available slots with each resource's calendar divided into 15-minute slots. Additionally, all specialists initially have no previously scheduled appointments. We scheduled a 60-minute appointment, each requiring a different number of resources per problem instance. Additionally, we considered two patient preferences: 1) excluding the dates 17.12.2024 and 18.12.2024, and 2) prioritizing the appointment to be scheduled on 20.12.2024. As an optimization criterion, we also aimed to ensure a fair distribution of workload among the available resources.

Table 2 presents the model's performance as the problem size is varied by adjusting the number of involved resources. Notably, the resolution time increases with the growth of both the problem size and the number of resources required for the appointments. Table 3 illustrates the model's performance as the number of scheduling days is gradually increased. The findings highlight the consistent performance of the constraint model, even as the complexity of the problem increases. For instance, with a problem size of 252 days (36 weeks), the RT and TTBS only increase to 392 *ms* and 360 *ms*, respectively. These results demonstrate the capability of our approach to handle large-scale scheduling problems efficiently, which is critical for practical deployment in large health care facilities such as *Charité*.

We observed variations in RT and TTBS across different problem sizes. These variations can be attributed to several factors inherent to constraint satisfaction problems and the optimization process. Although it may seem counterintuitive, smaller problem instances can sometimes take longer to solve than larger ones. This behavior may result from the heuristics employed by the solver, which can perform suboptimally on certain smaller instances, leading to longer resolution times.

Experiment 2. The purpose of this experiment is to examine the model's behavior when assigning resources to a medical appointment. Specifically, we aim to investigate the objective function's effectiveness in ensuring a fair distribution of workload among resources. To this end, we considered ten resources, all sharing the same medical specialization. Initially, all resources had zero workload, except for the first two, which started with a workload of 150 minutes. The model was tasked with scheduling an appointment requiring two cardiologists for a duration of 60 minutes. Further, we neglect the patient's preferences. After determining a solution, the appointment was confirmed, and the experiment was repeated to track the cumulative workload across resources.

Figure 3 illustrates the resource-to-appointment assignments in two scenarios: without workload optimization (left subfigure) and with workload optimization (right subfigure). When workload optimization is disabled, the solver consistently assigned the first two resources to the appointment in all rounds. As a result, their workload accumulated progressively, increasing from 210 minutes (150 initial workload + 60 for the appointment) to 450 minutes over multiple rounds. In contrast, with workload optimization enabled, the solver excluded the first two resources from scheduling. Their workload, highlighted in red in the heat map (right-hand subfigure), remained unchanged. Furthermore, it is clear that the solver selects the optimal schedule by consistently choosing the resource with the minimal cumulative workload in each scheduling round.

Table 2: Problem Size (in Resources) vs. Performance Metrics. The scheduling period was set to 49 days.

Problem Instance	Required Resources	All Resources	Specialists	Rooms	CT-Scan	RT (sec)	TTBS (sec)
1	**2**	**40**	24	8	8	0.222	0.130
2	**4**	**50**	30	10	10	0.614	0.506
3	**6**	**60**	34	13	13	0.911	0.804
4	**8**	**70**	40	15	15	1.107	1.002
5	**10**	**80**	46	17	17	1.401	1.219
6	**12**	**90**	50	20	20	1.642	1.330
7	**14**	**100**	56	22	22	1.982	1.521
8	**16**	**110**	60	25	25	2.649	2.197
9	**18**	**120**	66	27	27	3.420	2.420
10	**20**	**132**	72	30	30	4.111	3.510

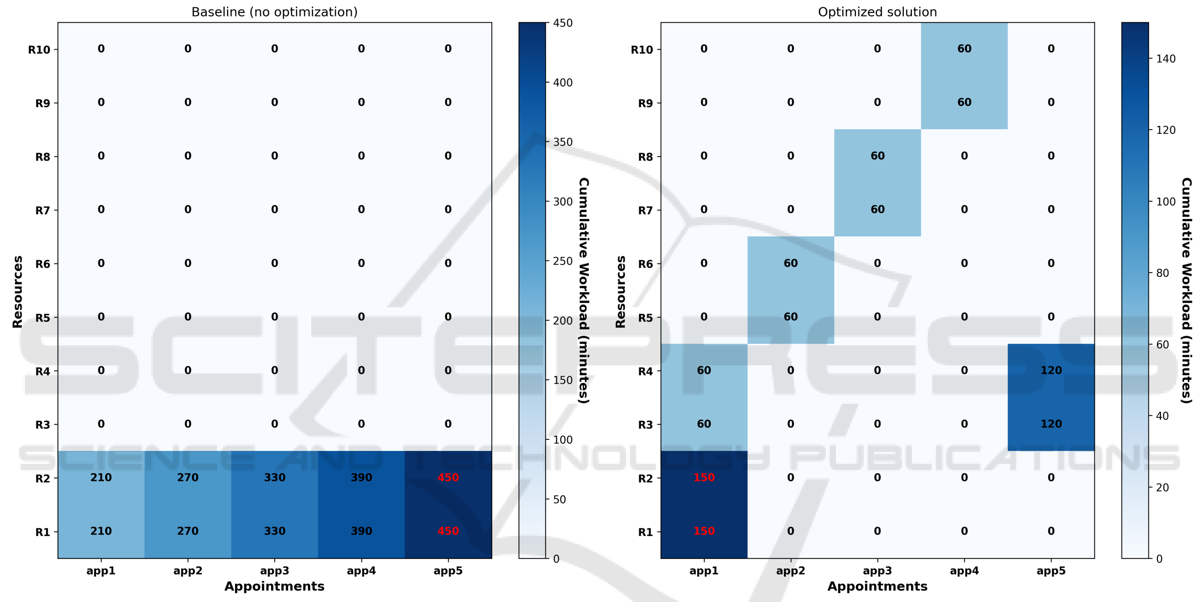


Figure 3: Workload distribution per appointment. We consider five scheduling rounds (app1 to app5) for the same appointment setting. The highest workload values are highlighted in red.

Table 3: Problem Size (in Days) vs. Performance Metrics. The involved resources are 72 specialists, 30 rooms, and 30 CT-scan devices.

Problem Instance	Days	RT	TTBS
1	**21**	208 ms	170 ms
2	**42**	234 ms	196 ms
3	**56**	219 ms	187 ms
4	**77**	249 ms	210 ms
5	**112**	243 ms	208 ms
6	**182**	274 ms	243 ms
7	**252**	392 ms	360 ms
8	**280**	425 ms	393 ms
9	**350**	487 ms	449 ms
10	**490**	638 ms	605 ms

6 CONCLUSION AND FUTURE WORK

We proposed a stepwise constraint modeling approach for the medical appointment scheduling problem, but it also can be generalized to other application fields. The problem has been formalized as a set of variables defined over finite domains and a set of constraints over these variables reflecting different requirements of the problem. These requirements cover both intuitive conditions to schedule an appointment such as the availability of required resources as well as patient preferences such as scheduling an appointment on a specific date. Furthermore, model potential output is subject to be optimized to consider both criteria scheduling an appointment as soon as possible and ensuring fair workload distribution among the facility resources.

Our experimental findings highlight the efficiency of the proposed constraint model, effectively solving instances of varying complexity within minimal computational time. Furthermore, the model demonstrated exceptional performance in optimizing the schedule and distributing the workload among resources, further highlighting the efficacy of the proposed approach.

Future work includes extending our model to accommodate the scheduling of sequences of interdependent medical appointments. This extension is particularly critical for managing treatment plans for patients with complex conditions. There are additional objectives such as finding appointment sequences with certain dependencies, like orders and spacing between the individual appointments. Last but not least, we plan to incorporate patient preferences for each appointment within the sequence, such as prioritizing specific dates, times and physicians.

ACKNOWLEDGEMENTS

The authors would like to thank the anonymous reviewers for their valuable and constructive feedback, which significantly contributed to improving this work. This research was supported by the German Ministry for Economic Affairs and Climate Action (BMWK) as part of the programme 'ZIM - Zentrales Innovationsprogramm Mittelstand' ('Central Innovation Programme for small and medium-sized enterprises') (FKZ: 16KN093238).

REFERENCES

Abdalkareem, Z. A., Amir, A., Al-Betar, M. A., Ekhan, P., and Hammouri, A. I. (2021). Healthcare scheduling in optimization context: a review. *Health and Technology*, 11(3):445–469.

Ala, A. and Chen, F. (2022). Appointment scheduling problem in complexity systems of the healthcare services: A comprehensive review. *Journal of Healthcare Engineering*, 2022(1):5819813.

Apt, K. (2003). *Principles of Constraint Programming*. Cambridge University Press.

Bessiere, C. (2006). Chapter 3 - constraint propagation. In Rossi, F., van Beek, P., and Walsh, T., editors, *Handbook of Constraint Programming*, volume 2 of *Foundations of Artificial Intelligence*, pages 29–83. Elsevier.

Bitner, J. R. and Reingold, E. M. (1975). Backtrack programming techniques. *Commun. ACM*, 18(11):651–656.

Freuder, E. C. and Mackworth, A. K. (2006). Chapter 2 - constraint satisfaction: An emerging paradigm. In Rossi, F., van Beek, P., and Walsh, T., editors, *Handbook of Constraint Programming*, volume 2 of *Foundations of Artificial Intelligence*, pages 13–27. Elsevier.

Global Constraint Catalog (2022). Global constraint catalog. http://sofdem.github.io/gccat/. [Online].

Guéret, C., Jussien, N., Boizumault, P., and Prins, C. (1996). Building university timetables using constraint logic programming. In Burke, E. and Ross, P., editors, *Practice and Theory of Automated Timetabling*, pages 130–145, Berlin, Heidelberg. Springer Berlin Heidelberg.

Hanset, A., Meskens, N., and Duvivier, D. (2010). Using constraint programming to schedule an operating theatre. In *2010 IEEE Workshop on Health Care Management (WHCM)*, pages 1–6.

IBM ILOG CPLEX Optimization Studio (2019). *CPLEX User's Manual*. IBM. Version 12.10.

Kuchcinski, K. and Szymanek, R. (2012). Jacop – a java constraint programming solver. *Foundations of Computing and Decision Sciences*, 37(4):309–324.

Morrison, D. R., Jacobson, S. H., Sauppe, J. J., and Sewell, E. C. (2016). Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning. *Discrete Optimization*, 19:79–102.

Nethercote, N., Stuckey, P. J., Becket, R., Brand, S., Duck, G. J., and Tack, G. (2007). Minizinc: Towards a standard cp modelling language. In *Principles and Practice of Constraint Programming*, pages 529–543. Springer, Berlin, Heidelberg.

Oliveira, M., Rocha, A. M. A. C., and Alves, F. (2024). Using or-tools when solving the nurse scheduling problem. In *Optimization, Learning Algorithms and Applications*, pages 438–449, Cham. Springer Nature Switzerland.

Rappos, E., Thiémard, E., Robert, S., and Hêche, J.-F. (2022). A mixed-integer programming approach for solving university course timetabling problems. *Journal of Scheduling*, 25(4):391–404.

Régin, J.-C., Schaus, P., Prud'homme, C., Lecoutre, C., Rochart, J., and Legrain, A. (2017). Choco: an open source java constraint programming library. In *International Conference on Principles and Practice of Constraint Programming*, pages 500–518. Springer, Cham.

Topaloglu, S. and Ozkarahan, I. (2011). A constraint programming-based solution approach for medical resident scheduling problems. *Computers & Operations Research*, 38(1):246–255.

Vermeulen, I. B., Bohte, S. M., Elkhuizen, S. G., Lameris, H., Bakker, P. J., and Poutré, H. L. (2009). Adaptive resource allocation for efficient patient scheduling. *Artificial Intelligence in Medicine*, 46(1):67–80. Artificial Intelligence in Medicine AIME' 07.