Manuscript

A prediction-based approach for online dynamic radiotherapy scheduling

Tu-San Pham[a,*], Antoine Legrain[a],
Patrick De Causmaecker[b], Louis-Martin Rousseau[a]

January 21, 2022

* Corresponding author.
Email: tu-san.pham@polymtl.ca

Affiliations:
[a] Polytechnique Montréal, Quebec, Canada
[b] KU Leuven, Kortrijk, Belgium

1 Abstract

Patient scheduling is a difficult task as it involves dealing with stochastic factors such as an unknown arrival flow of patients. Scheduling radiotherapy treatments for cancer patients faces a similar problem. Curative patients need to start their treatment within the recommended deadlines, i.e., 14 or 28 days after their admission while reserving treatment capacity for palliative patients who require urgent treatments within 1 to 3 days after their admission. Most cancer centers solve the problem by reserving a fixed number of treatment slots for emergency patients. However, this flat-reservation approach is not ideal and can cause overdue treatments for emergency patients on some days while not fully exploiting treatment capacity on some other days, which also leads to delaying treatment for curative patients. This problem is especially severe in large and crowded hospitals. In this paper, we propose a prediction-based approach for online dynamic radiotherapy scheduling. An offline problem where all future patient arrivals are known in advance is solved to optimality using Integer Programming. A regression model is then trained to recognize the links between patients' arrival patterns and their ideal waiting time. The trained regression model is then embedded in a prediction-based approach that schedules a patient based on their characteristics and the present state of the calendar. The numerical results show that our prediction-based approach efficiently prevents overdue treatments for emergency patients while maintaining a good waiting time compared to other scheduling approaches based on a flat-reservation policy.

Keywords: Operations Research; Radiotherapy scheduling; Integer Programming; Patient Scheduling

2 Introduction

According to the WHO [1], cancer is the leading cause of death worldwide, responsible for 10 million deaths in 2020. In Canada, the number of cancer deaths in 2020 was 83,300, accounting for 30% of all deaths in the country. The number of cancer incidents is steadily on the rise in many countries, which results in a lot of stress on the treatment facilities and medical staff and also makes timely access to cancer treatments a difficult task. It is essential to improve the treatment scheduling process to optimize the utilization of available resources and reduce the waiting time of patients.

In this paper, we focus on efficiently planning radiotherapy (RT) treatments, one of the most popular forms of cancer treatment. About 50% of cancer patients undergo RT treatment [1, 21]. During this treatment, a patient receives a high dose of radiation divided into smaller ones (called *fractions*) delivered to the cancer site over a number of consecutive days. This treatment is most commonly delivered by a linear accelerator (*linac*). As the number of linacs in a hospital is usually limited, the waiting time for cancer patients to start their RT treatments depends greatly on treatment scheduling. Every patient has a recommended due date by which their treatment should start. Overdue treatments (treatments started after the recommended deadlines) are highly discouraged. In addition, waiting time should be minimized as increased waiting time for RT treatment has a negative impact on clinical outcomes [5, 14, 4]. Therefore, better planning to reduce waiting time is crucial in improving treatment outcomes.

In this paper, we consider a scheduling problem for RT treatments arising at CHUM (Centre hospitalier de l'Université de Montréal), a large cancer center in Montréal, Canada. Given a number of linacs and an incoming flow of treatment requests, we must determine the most efficient treatment schedule for patients. The main objective is to minimize overdue treatments as well as patient waiting time, which can be achieved by deciding the starting dates and the linacs for patients' first treatments. At CHUM, patients are divided into four categories based on the urgency of treatment. Palliative pa-

[1]https://www.who.int/news-room/fact-sheets/detail/cancer

tients (categories P1 and P2) need urgent care to relieve intense pain, hence the treatment deadline is set to one or three days after their admission. Treatment deadlines in curative patients (categories P3 and P4) are 14 or 28 days after admission, depending on the cancer site and their condition. Approximately 70% of patients treated at CHUM are curative. Despite accounting for only 30% of treatments, palliative treatments are the most challenging to schedule (see [19]). As palliative patients require urgent care after admission, with little time to schedule in advance, it is important to anticipate future arrivals of patients in order to make good scheduling decisions. Many cancer centers reserve a number of treatment slots per day for emergency treatments. This approach is used in several papers [17, 13, 19, 8]. Nonetheless, it is difficult to set a proper threshold. If the reserved capacity is too high, it reduces the capacity for treating curative patients and hence lengthens their waiting time. If the reserved capacity is too low, there may not be enough slots for emergency treatments.

In the RT scheduling literature, there are so far only two classes of approach that take into account future arrivals when deciding the treatment schedule. Legrain et. al [13] combine stochastic programming with online optimization. That is, once a treatment plan is approved, the appointments are made and the patient leaves with their schedule in hand. Saure et al. [20], on the other hand, apply *batch scheduling*, where scheduling decisions for a group of patients are made daily by the end of the day using a discounted infinite-horizon Markov Decision Process (MDP). MDP is also utilized by Gocgun [9], which additionally allows for the cancellation of treatments. Compared to online scheduling as in [13], batch scheduling has the advantage of having a larger search space and hence usually results in better schedules. However, due to larger problem sizes, batch scheduling is more difficult to solve than online scheduling, which involves only one patient at a time. Online scheduling, on the other hand, is preferred by many cancer centers for the convenience of their patients and scheduling staff. Patients can leave the center with their appointments in hand while the scheduling staff does not have to call patients to notify them about their appointments. Stochastic programming and MDP, however, are both prone to scaling problems which makes it difficult to apply either approach in large hospitals. In addition, they are algorithmically heavy and may be difficult to implement and maintain in real-world applications. In the literature, few attempts have been made to tackle the scaling problem of stochastic programming using machine learning techniques. One example can be found in the context of two-stage stochastic programming [12], where the authors use supervised machine learning to predict the computationally expensive second stage and apply the algorithm to a railway demand and capacity management problem.

In this paper, we propose a novel online prediction-based approach for radiotherapy scheduling which takes future arrivals into account. As the arrival pattern of cancer patients can be learned from historical data, it can give us more information on future arrivals and hence assists scheduling decisions. Assuming that we have perfect knowledge of future arrivals, we can derive an optimal *offline schedule* using an exact model. We propose an Integer Programming (IP) model for this purpose. Given that we have a large number of instances with the same arrival rate and number of linacs, a regression model can be trained to detect the links between a patient's features, the current state of the hospital and the ideal schedule. The regression model can then predict a "good" waiting time for a patient.

Our algorithm solves the radiotherapy scheduling problem in an online manner, i.e., scheduling decisions are made upon patients' arrival. As we prioritize on-time treatments for palliative patients, they are always scheduled on the earliest possible dates. Curative patients are scheduled using a prediction model which suggests a waiting time based on their treatment plan and the current occupancy of linacs. This suggested waiting time is fed to a prediction-based scheduling algorithm that takes into account other constraints and returns an actual valid starting date. In essence, the prediction model is trained to delay treatments of curative patients to make space for palliative ones based on an observation of the current state of the hospital. This results in a robust scheduling approach that is able to take into account future arrivals. We refer to the proposed machine learning-based approach as *prediction-based approach*. It is compared to different scheduling strategies, including an online greedy heuristic that is currently used at CHUM and three batch scheduling strategies which schedule patients daily, bi-weekly, and weekly, respectively. The algorithms are tested in a simulation with a rolling horizon to evaluate their performance over a period of time. Test datasets are generated from actual data obtained from CHUM. Compared to other approaches, the prediction-based approach shows superior performance in most problem settings (problem settings are differentiated by number of linacs and arrival rate). It results in the lowest average overdue time and waiting time in palliative patients while maintaining similar numbers in curative patients compared to other approaches, which demonstrates its ability to efficiently integrate information on future arrivals into daily scheduling decisions. Notably, the superiority of the prediction-based approach is more prominent in larger hospital settings with more linacs and more patients to treat (large arrival rate).

Our approach offers many advantages. First, despite being an online scheduling strategy, it surpasses the batch scheduling strategies tested. Second, the running time of the algorithm does not depend on the instance size. This offers an enormous advantage compared to other stochastic approaches in the literature that suffer from the curse of dimension. The highest cost associated with this approach is to generate optimal offline solutions for the training phase. However, it is an offline cost and can be done easily with an average server. In our experiments, we tested successfully for instances with up to 8 linacs, which represents a hospital size that is larger than most of the cancer centers studied in literature.

The rest of the paper is organized as follows. Section 3 provides problem description and literature review. The methodology is presented in Section 4. Section 5 presents numerical results, and finally Section 6 closes the paper with our conclusions and future work.

3 Problem description and related work

In this section, we describe the radiotherapy scheduling problem arising at CHUM and then present some related work and situate our work in the literature.

3.1 Problem description

In radiotherapy treatment, a patient receives a series of radiation doses (fractions) delivered by a linac to the cancer site during the treatment course, which typically consists of several consecutive days with breaks on the weekends. Multiple appointments are therefore required. The problem studied in this paper is a real-world problem arising at CHUM, a large cancer center in Montréal, Quebec which is capable of treating around 3,500 patients per year (data from 2018). CHUM's facility consists of 12 linacs, eight of which are generalized linacs while the remaining four are specialized linacs required for special types of treatment.

At CHUM, patients are divided into four categories with different waiting time targets based on their cancer type and condition. Palliative patients (categories P1 and P2) have their waiting time target set as 1 and 3 days, respectively. In curative patients (categories P3 and P4), the target is 14 and 28 days, respectively. To minimize the waiting time of patients, we only need to decide the start date for treatment, as once a treatment starts, it must be carried out daily. The waiting time of a patient is the time elapsed between his/her admission date and the date of the first treatment. If a patient's treatment starts after the recommended deadline, the time delay is counted as overdue time.

Cancer treatment is a complicated process with many procedures and parties involved. The treatment workflow used at CHUM is illustrated in Figure 1. When a new consultation request is made, a patient first undergoes a consultation session where the doctor explains his/her condition and the available options. If the patient agrees to proceed with the treatment, they will go through several preparation steps such as external consultation and exams. The next step is planification scan, where the patient undergoes a set of tests and imaging. Then, in the geometric and dosimetric planning

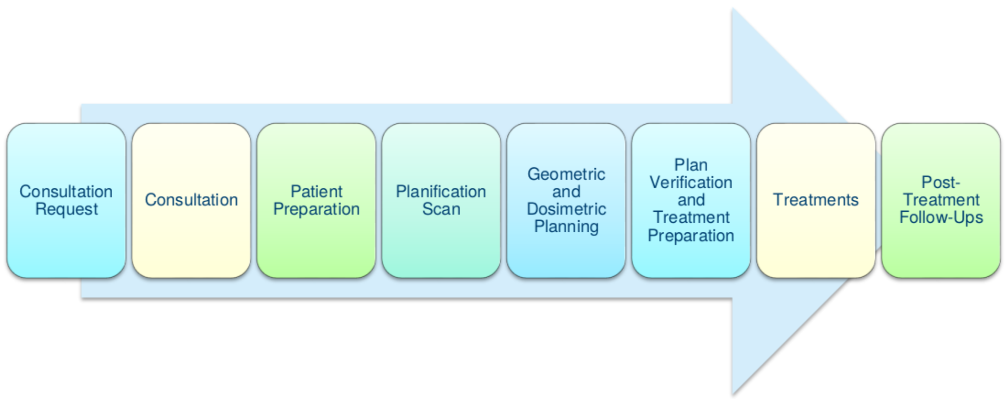
Consultation Request
Consultation
Patient Preparation
Planification Scan
Geometric and Dosimetric Planning
Plan Verification and Treatment Preparation
Treatments
Post-Treatment Follow-Ups

Figure 1: Radiotherapy treatment workflow at CHUM[2].

step, a treatment team carefully makes a treatment plan for the patient based on their diagnosis. The treatment plan will need to be verified and approved by a physicist before the treatments are prepared. The RT treatments are then carried out, with review and possibly revision during the course of treatment. Finally, the treatment ends with post-treatments and follow-ups. Currently, the scheduling of treatments is performed manually at CHUM.

Among various technical information, a treatment plan includes information about the number of fractions that a patient will receive and the duration of each fraction. Since fraction durations at CHUM are always multiples of 5 minutes, we set the granularity of our schedules to 5-minute blocks. Each linac has a treatment capacity measured in time blocks.

3.2 Literature review

Radiotherapy scheduling is a relatively young field of research. A review of RT scheduling can be found in [11, 22]. The first two mathematical models for RT scheduling are proposed by Conforti [6] to maximize the number of treated patients within a given horizon with identical treatment times and a single linac. The model assigns patients to treatment slots of equal length, which is referred as *block* scheduling. The authors then extend their models in [7] to allow for more realistic settings where treatment duration can vary in length (*non-block scheduling*). Petrovic et al. [18, 17] propose several constructive heuristics based on different prioritised rules and improve the solutions using metaheuristics. They use block scheduling where treatment on a linac is of fixed duration and depends on the energy type of the linac. Burke et al. [2] propose a non-block IP model to assign patients to days and linacs and perform several simulations to investigate whether delaying the scheduling decision can lead to better schedules. Heuristic methods are also used in [10] to schedule patients in both pre-treatment and treatment stages. Pre-treatment for RT patients is also considered in [3], where the authors solve a multi-objective problem by formulating it as a series of single-objective scheduling problems.

Another form of RT treatment is particle therapy, which uses a centralized machine to deliver an ion beam to different treatment rooms. Most studies in particle therapy hence focus on deciding specific appointment times for treatments to optimize the beam's utilization, i.e., minimizing the idle time of the machine. An exact model is proposed in [16] but is proven to be highly intractable. Metaheuristics are therefore widely used for particle therapy treatment scheduling [16, 15, 24].

In another context, Veira et al. [23] point out that most cancer centers in Europe are capable of treating their patients within the recommended waiting period. They hence focus on meeting patients' preferences for treatment times. They propose a heuristic to pre-assign patients to linacs and use an exact model to solve the resulting subproblems. In [8], the authors propose an IP and

[2]Figure provided by CHUM.

two constraint programming (CP) models for scheduling RT treatments. Their generated instances include an expected number of future arrivals, based on which they predict future resource utilization.

None of the above-mentioned studies take into account future arrivals, and all of them are tested on a static horizon. In RT scheduling, and patient scheduling in general, it is important to evaluate the performance of the algorithm over time in a rolling horizon where patient arrivals are revealed gradually. In [19], the authors propose a two-phase approach for RT scheduling and implement a simulation to evaluate the algorithm dynamically in a rolling horizon using different scheduling strategies. The approach, however, is still myopic, i.e., does not take into account future arrivals when making scheduling decisions. To the best of our knowledge, there are only two approaches for dynamic RT scheduling which take into account future events. The first approach is Markov Decision Process (MDP), which is proposed by Saure et al. [20] to provide a dynamic policy that takes into account future events. The problem is modeled as a discounted infinite-horizon MDP and the equivalent Integer Programming (IP) model is solved using column generation technique. The authors solve the problem in an offline fashion where batch scheduling decisions are made daily by the end of the day. The same approach is also used in [9], which additionally allows for the cancellation of treatments. Legrain et al. [13], in contrast, propose a hybrid method combining stochastic optimization and online optimization to solve the problem in an online fashion, i.e., one patient at a time. The aforementioned approaches, however, are computationally expensive and hence are limited to relatively small instances. In [20], their practical example has an arrival rate of 8.25 requests per day with 120 appointment slots, which is equivalent to three linacs. Legrain et al. [13] consider a 20-minute time slot for each patient and tests the algorithm on instances with up to two linacs and an arrival rate of less than 3.5. In this paper, we aim to solve the problem for larger instances with up to 8 linacs.

4 Methodology

We first present a mathematical model for the problem in Section 4.1 and then introduce different scheduling strategies in Section 4.2. The remainder of the section explains our prediction-based approach, which is the main contribution of this paper.

4.1 Mathematical model

The problem consists of finding the best schedule of treatments for a set of patients $\mathcal{P}$ given a set of linacs $\mathcal{L}$. The sets of palliative patients and curative patients are denoted as $\mathcal{P}^{\mathcal{P}}$ and $\mathcal{P}^{C}$, respectively, $\mathcal{P} = \mathcal{P}^{\mathcal{P}} \cup \mathcal{P}^{C}$. Each instance consists of a set of fixed patients with appointments made from the previous scheduling decisions. This set is denoted as $\bar{\mathcal{P}}$ while the set of new patients is denoted as $\hat{\mathcal{P}}$, $\mathcal{P} = \bar{\mathcal{P}} \cup \hat{\mathcal{P}}$. The set of days in the planning horizon is denoted as $\mathcal{T}$. Each patient $i \in \mathcal{P}$ has an admission date a_i; a ready date r_i which is the earliest date the patient can begin treatment; and a due date d_i, which is the recommended deadline to start the treatment. Each patient i is associated with a treatment plan, which specifies the number of fractions I_i that the patient needs to receive and the duration of each fraction (p_i). Each linac l has a capacity (C_l^t) measured in blocks of 5 minutes. $\hat{C}_l^t$ represents the available capacity of linac l on day t after deducting the fixed appointments from the previous scheduling decisions.

As our objective is to minimize overdue time and waiting time, we propose a model to determine the assignment of patients' first fractions to dates and linacs. Once treatment starts, it is carried out every day until the required number of fractions is achieved. Subsequent fractions of each patient should be carried out on the same linac as the first treatment. Two objective weights, ω_1 and ω_2, are introduced to control the balance between waiting time and overdue time. By setting $\omega_2 >> \omega_1$, we heavily penalize overdue time over waiting time. Capacity of linacs is a hard constraint, with a portion of linac capacity reserved for palliative patients. Parameter γ ($\gamma < 1$) indicates the percentage

Strategy	Scheduling palliative patients	Scheduling curative patients
offline	at admission	one time
greedy	at admission	at admission
daily	every day	every day
bi-weekly	every day	every Tuesday & Friday
weekly	every day	every Friday
prediction-based	at admission	at admission

Table 1: Scheduling strategies.

of reserved capacity.

On the set of new patients $\hat{\mathcal{P}}$, we define a binary variable x_{tl}^i which holds value 1 if patient i receives his/her first treatment on day t, linac l. The model is as follows:

$$\begin{aligned} \text{minimize} \sum_{i\in\hat{\mathcal{P}}}\sum_{t\in\mathcal{T},t>a_i}\sum_{l\in\mathcal{L}} \omega_1(t-a_i)log(t-a_i+1)x_{tl}^i \\ +\sum_{i\in\hat{\mathcal{P}}}\sum_{t\in\mathcal{T},t>d_i}\sum_{l\in\mathcal{L}} \omega_2(t-d_i)log(t-d_i+1)x_{tl}^i \end{aligned} \tag{1}$$

Subject to

$$\sum_{t\in\mathcal{T}}\sum_{l\in\mathcal{L}} x_{tl}^i = 1 \qquad \forall i\in\hat{\mathcal{P}} \tag{2}$$

$$x_{tl}^i = 0 \qquad \forall i\in\hat{\mathcal{P}}, l\in\mathcal{L}, t\in\{0,\dots,r_i-1\} \tag{3}$$

$$\sum_{i\in\hat{\mathcal{P}}}\sum_{t'=max\{0,t-I_i+1\}}^{t} p_i x_{t'l}^i \le \hat{C}_l^t \qquad \forall t\in\mathcal{T}, l\in\mathcal{L} \tag{4}$$

$$\sum_{i\in\mathcal{P}^C}\sum_{t'\in\{t-I_i+1,\dots,t\}} p_i x_{t'l}^i \le max\{0, \hat{C}_l^t - \gamma C_l^t\} \qquad \forall t\in\mathcal{T}, l\in\mathcal{L} \tag{5}$$

$$x_{tl}^i \in \{0,1\} \qquad \forall i\in\hat{\mathcal{P}}, t\in\mathcal{T}, l\in\mathcal{L} \tag{6}$$

The objective function (1) minimizes waiting time (the first term) and overdue time (the second term) with the respective weights ω_1 and ω_2. The log function is a non-linear function utilized to enforce an equal distribution of waiting time and overdue time between patients. Constraints (2) ensure all patients are scheduled within the planning horizon. Constraints (3) ensure no patients are scheduled before their ready date. Constraints (4) make sure linac capacity is respected for the entire course of treatment. Constraints (5) reserve a portion of linac capacity for palliative patients.

4.2 Scheduling strategies

In this section, we propose several scheduling strategies for radiotherapy treatments. All strategies utilize online scheduling for palliative patients, i.e., their treatments are scheduled for the earliest possible date on an available linac when the treatment request is submitted. A portion of linac capacity is reserved for palliative patients. The proposed strategies differ in the manner in which curative patients are handled. All scheduling strategies are summarized in Table 1.

4.2.1 Online scheduling with a greedy heuristic

This strategy is the closest one to the approach used by scheduling clerks at CHUM. Once a patient is admitted (usually when their treatment plan is approved), the algorithm looks up one or two weeks in advance depending on the patient's category (P3 or P4). The first feasible date for accomodating all of the patient's fractions is then chosen. A date is considered eligible if on this day, and the following $(I_i - 1)$ days, there is a linac with enough remaining capacity to treat the patient, with I_i representing the number of fractions of the patient. An illustration of the greedy heuristic can be found in Figure 2.

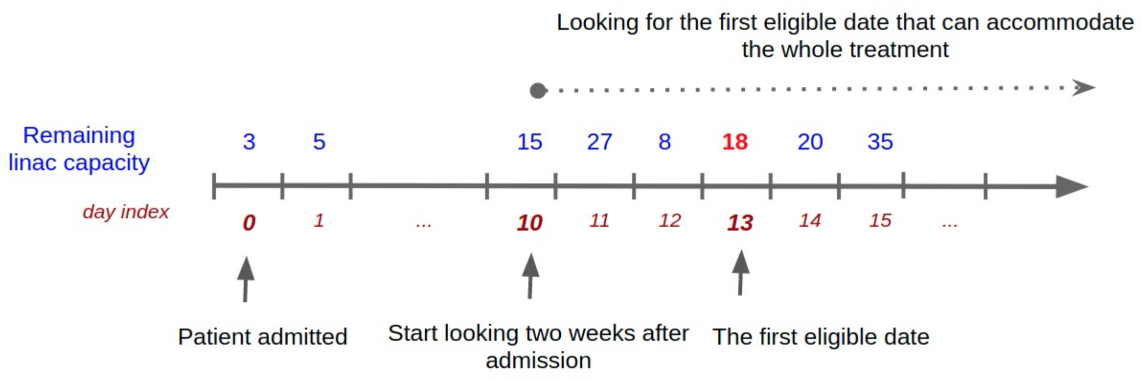
Looking for the first eligible date that can accommodate the whole treatment
Remaining linac capacity
3
5
15
27
8
18
20
35
day index
0
1
...
10
11
12
13
14
15
...
Patient admitted
Start looking two weeks after admission
The first eligible date

Figure 2: Greedy heuristic for an instance with one linac. A curative patient (category P4) with 3 fractions, each of which requires 10 time slots, is admitted on day 0. The algorithm looks up two weeks ahead, starting from day 10 (only business days are indexed). The first day after day 10 that is able to accommodate the whole treatment is day 13.

4.2.2 Batch scheduling

In batch scheduling, treatment requests are accumulated for a period of time and then a scheduling decision is made by the end of the period, e.g., daily or weekly. At each scheduling decision, a set of patients admitted from the day after the previous scheduling decision to the current day are scheduled using the IP model in Section 4.1. A visualization of batch scheduling can be found in Figure 3.

4.2.3 Offline scheduling

In this scheduling strategy, we assume that all patient arrivals are known in advance. The algorithm first schedules palliative patients for the first eligible day after their arrival. Curative patients admitted during the simulation period are then scheduled using the IP model in Section 4.1, without reserving a portion of linac capacity for palliative patients. The scheduling decision is made on the first day of the scheduling period. Having all future patient arrivals known in advance is an unrealistic assumption. However, it provides a bound for other scheduling strategies. No strategy can bypass

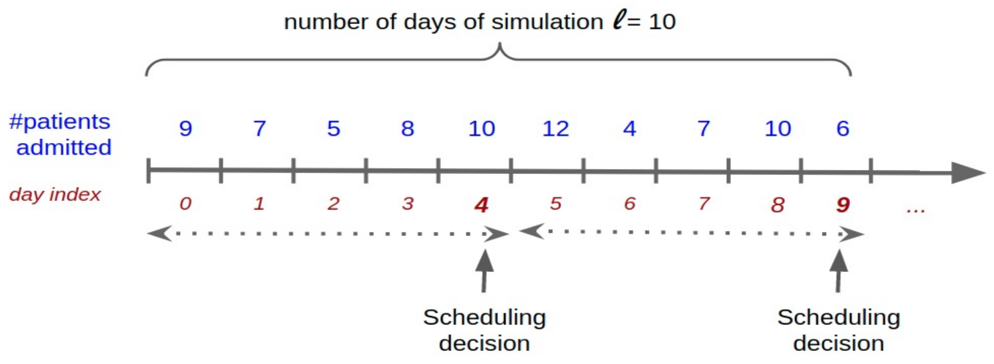

Figure 3: Batch scheduling where scheduling decision is made weekly. On day 4, patients admitted from day 0 to day 4 are scheduled. On day 8, patients admitted from day 5 to day 8 are scheduled.

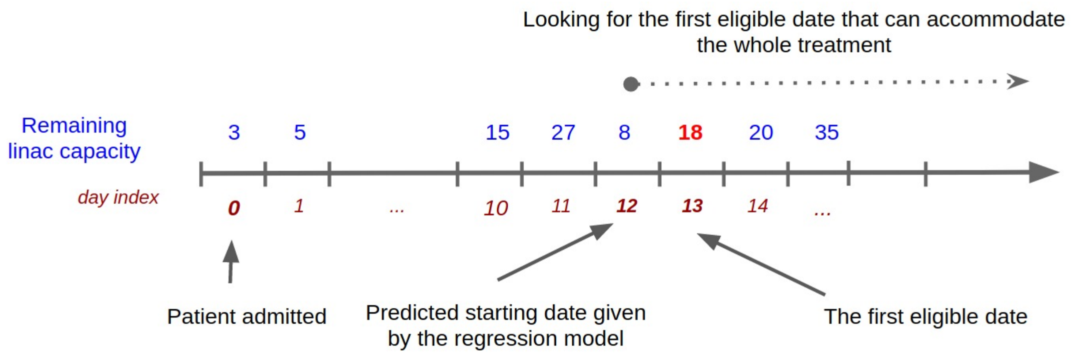
Looking for the first eligible date that can accommodate the whole treatment
Remaining linac capacity
3
5
15
27
8
18
20
35
day index
0
1
...
10
11
12
13
14
...
Patient admitted
Predicted starting date given by the regression model
The first eligible date

Figure 4: Prediction-based scheduling for an instance with 1 linac. A curative patient (category P4) with 3 fractions, 10 time slots each is admitted on day 0. The algorithm looks up two weeks ahead (day 10 - day index is for business days only). The first day after day 10 which is able to accommodate the whole treatment is day 13.

offline scheduling due to the lack of information on future arrivals. It also provides a picture of how a "perfect schedule" would look if we had a precise prediction of future arrivals.

4.2.4 Online scheduling using regression - a machine learning approach

This scheduling strategy is the main contribution of the paper. Scheduling is carried out in an online fashion similar to the greedy heuristic. However, instead of looking up one or two weeks in advance, we apply a regression model to predict a start date for a patient. The algorithm then looks for an eligible date starting from the predicted day. An illustration of the algorithm is given in Figure 4. The details of the regression model will be presented in Section 4.3.

4.3 Prediction-based approach - a machine learning-based algorithm

In most cancer centers, the number of linacs is fixed. The arrival rate of patients is usually stable and can be modelled as a Poisson distribution. Given an instance with a number of linacs and an arrival rate, an optimal offline solution (referred to simply as an *offline solution*) given by offline scheduling (Section 4.2.3) provides an "ideal" schedule where all future arrivals are known in advance. All other scheduling strategies based on the gradual revelation of patient arrivals share the same goal of achieving solutions as close as possible to offline solutions. The intuition of our approach is that if we have a large enough number of instances and their offline solutions, we might be able to learn the patterns in the ideal optimal waiting time of patients. As a result, given the current occupancy of linacs and the treatment plan of a patient, we can predict a "good" waiting time for the patient. In the following section, we describe our method of training the regression model and applying it in a machine learning-based scheduling algorithm, which we refer to as a *prediction-based approach*.

4.4 Training the regression model

A problem instance is characterized by a number of linacs $|\mathcal{L}|$ and an arrival rate λ. A pair $(|\mathcal{L}|, \lambda)$ is referred to as an *instance setting*, which represents a typical hospital/cancer center with a fixed number of linacs and a stable arrival rate. The number of time blocks per linac in all our problem settings is constant and is set as 120 blocks of 5 minutes per linac per day. Each instance setting is associated with a regression model. Our reasoning is that each hospital has its own historical ideal scheduling patterns and hence needs a separate regression model that learns from its past data. For each instance in the training set, an offline solution is obtained by solving the offline scheduling as in Section 4.2.3. A set of training examples is then constructed from the instance and its offline solution.

To explain the algorithms used to construct training examples, we describe some terms. A problem instance has a set of simulation days (denoted as $\mathcal{K}$), which consists of several days, each of which observes a set of patients admitted to the center (denoted as $\mathcal{P}_d, d \in \mathcal{K}$). A problem instance hence consists of a flow of patients admitted to the hospital daily during the simulation period, which is referred as a *patient flow* ζ, where $\zeta = \{\mathcal{P}_d, \forall d \in \mathcal{K}\}$. A solution of an instance is the assignment of patients to a set of dates ($\mathcal{D}$) and linacs ($\mathcal{L}$): $s : \mathcal{P} \times \mathcal{D} \times \mathcal{L} \rightarrow \{0, 1\}$, where $s^i_{dl} = 1$ if patient i is assigned to day d, linac l.

The *present time point* at any moment during the simulation period is denoted as ϕ, and the corresponding date is denoted as $d(\phi)$. At a time point ϕ, the *present capacity* of linacs on date d is the available capacity after deducting all treatment slots on the corresponding date that are occupied by patients admitted (and scheduled) before the time point ϕ, $\hat{c}^{\phi}_d = C - \sum_{i \in \mathcal{P}, l \in \mathcal{L} | s^i_{dl}=1, \alpha_i < \phi} p_i$, where C is the total linac capacity, p_i and α_i are the fraction duration and admission time point of patient i, respectively.

The *state* of an instance at a time point ϕ, denoted as $\hat{C}^{\phi}$, is a set of *present capacity* of all days in the *sample horizon*, $\hat{C}^{\phi} = \{\hat{c}^{\phi}_d, \forall d \in \mathcal{D}^{\phi}\}$, where $\mathcal{D}^{\phi}$ is the set of days in the sampling horizon at the time point ϕ. In our problem, the sampling horizon is set as 50 days after $d(\phi)$, $\mathcal{D}^{\phi} = \{d(\phi), d(\phi) + 1, \ldots, d(\phi) + 49\}$, since in our realistic instance settings, it is unlikely that a patient is not scheduled within 50 days after his/her admission. Therefore, when making a scheduling decision, the linac capacity search is restricted to a window of 50 days after his/her admission.

Given an instance and its offline solution s^*, a training example consisting of an input vector and a label is created for each curative patient in the patient flow. The label of a patient i is the patient's waiting time w_i in the offline solution. The input vector consists of the *state* measured at the patient's admission time $\phi = \alpha_i$ and a vector of the patient's features. We select the features that are the most relevant to the scheduling task, i.e., the patient's admission date a_i, ready date r_i, due date d_i, number of sessions I_i, fraction length p_i, and priority e_i. A training example for patient i is hence a tuple (X_i, w_i), where the input vector $X_i = \{\hat{C}^{\phi, \phi = a_i}, a_i, r_i, I_i, d_i, p_i, e_i\}$.

The algorithm to create training examples is presented in Algorithm 1. As the regression model is used to schedule curative patients only, we create one training example for each new curative patient in the instance. At the beginning of the algorithm, the present time point is set as the beginning of the first day in the simulation period $\mathcal{K}$ and the present capacity $\hat{C}^{\phi}$ is calculated accordingly. For each day $d \in \mathcal{K}$, the set of patients admitted on that day ($\mathcal{P}_d$) is iterated in a chronological order. If a patient is curative, a training example is created and added to the set of training examples. After each iteration, regardless of the patient's priority, the present capacity $\hat{C}^{\phi}$ is updated with the appointments of the corresponding patient from the offline solution s^*. The process continues until all patients in the patient flow are visited.

In the preliminary experiments, different regression models are tested, including *MLP*, *SGD*, *ElasticNet*, *Lasso*, *SVR*, *Decision Tree*, *Random Forest*, and *XGBoost*. *Random Forest* and *XGBoost* give comparable performance and outperform the other models. However, XGBoost requires less training time than Random Forest. In addition, the differences between the error rates in the training set and the testing set produced by XGBoost are smaller compared to that of Random Forest, which implies that XGBoost is less prone to overfitting. We therefore choose XGBoost for our regression model. The detailed results of the training phase can be found in A.

4.5 Using the regression model

The regression model is embedded in an algorithm to schedule RT treatments. The algorithm is presented in detail in Algorithm 2. When a new patient is admitted, there are two possibilities. If the patient is palliative, they will be scheduled on the first eligible date. If the patient is curative, an input

Algorithm 1: Generating training examples

```
input : instance $ins$, its offline solution $s^*$
output: set of training examples $\mathcal{E}$
1  $\mathcal{E} \leftarrow \{\}$ ;
2  $\phi \leftarrow$ the beginning of the first day in the simulation period $\mathcal{K}$ ;
3  calculate $\hat{C}^{\phi}$ ;
4  foreach day $d$ in the simulation period $\mathcal{K}$ do
5      foreach patient $i$ in $\mathcal{P}_d$ do
6          if $i$ is curative then
7              $X_i \leftarrow \{\hat{C}^{\phi}, d, r_i, I_i, d_i, p_i, e_i\}$ ;
8              $w_i \leftarrow$ waiting time of $i$ in $s^*$ ;
9              $\mathcal{E} \leftarrow \mathcal{E} \cup (X_i, w_i)$ ;
10         end
11         $\phi \leftarrow \alpha_i$ ;
12         update $\hat{C}^{\phi}$ with appointments of patient $i$ ;
13     end
14 end
```

vector is constructed as described in Section 4.4. The input vector is fed to the trained regression model to get a predicted waiting time. A predicted starting date is then derived and utilized in the algorithm presented in Section 4.2.4 to construct a schedule.

5 Numerical results

To evaluate the proposed algorithms, two experiments are carried out. The first experiment aims to examine the performance of the proposed algorithms on instances generated based on real data. The testbed is designed to offer an overview of the algorithms' behavior in different scenarios, i.e., different hospital sizes and crowding levels. The second experiment aims to test the algorithms on real patient flows from CHUM. Based on preliminary experiments (see B)), the reservation rate for greedy heuristic and batch scheduling strategies is set at 15% of treatment capacity. The reservation rate for the prediction-based approach is set at 10%. An overview of the historical data at CHUM is given in Section 5.1 while the generation of test instances is described in Section 5.2. The results for the two experiments are presented in Section 5.3 and 5.4, respectively.

5.1 Data overview

We have access to historical data from CHUM for a 3-year period, from September 2017 to July 2019, which consists of 4,538 patients and 68,439 treatments. CHUM is equipped with 12 treatment rooms, each room cotaining a linac. Rooms A, B, C, D, E have identical generic linacs. Most patients treated in rooms G, K, and L can be treated in the 5 generic rooms as well. However, a small percentage of treatments (2 to 3%) are specialized treatments that are bounded to those rooms. Room F is a special room dedicated to hyperfractionated radiation therapy. Room J offers cyberknife treatments. Rooms H and O are special linacs dedicated to some rare treatments which account for less than 0.1% of treatments.

Patients are classified into four categories with different treatment deadlines. The majority of patients are curative (more than 70%) with 14- or 28-day treatment deadlines in 2017-2018. The remaining patients are mostly palliative type P2 with 3-day treatment deadlines. A small portion of patients are emergency patients (type P1) who require urgent treatment with a one-day deadline. More than 70% of P2 and P3 patients at CHUM did not meet the recommended treatment deadlines.

Algorithm 2: Prediction-based scheduling

```
input : instance $ins$, a trained regression model $\mathcal{M}$
output: schedule $s$
1  Initialize an empty schedule $s$ ;
2  $\phi \leftarrow$ the beginning of the first day in the simulation period $\mathcal{K}$ ;
3  calculate $\hat{C}^{\phi}$ ;
4  foreach day $d \in \mathcal{K}$ do
5      foreach patient $i$ in $\mathcal{P}_d$ do
6          if $i$ is palliative then
7              $d^* \leftarrow$ the first eligible date for $i$ starting from $max(d, r_i)$ ;
8          end
9          if $i$ is curative then
10             $X_i \leftarrow \{\hat{C}^{\phi}, d, r_i, I_i, d_i, p_i, e_i\}$ ;
11             $\bar{w} \leftarrow$ predicted waiting time given by $\mathcal{M}(X_i)$ ;
12             $d^* \leftarrow$ the first eligible starting date for $i$ from $max(d + \bar{w}, r_i)$ ;
13         end
14         $s \leftarrow$ update the schedule by scheduling all fractions of $i$ with the starting date $d^*$ ;
15         $\phi \leftarrow \alpha_i$ ;
16         update $\hat{C}^{\phi}$ with new appointments of patient $i$ ;
17     end
18 end
```

Category	Proportion (%)	Treatment deadline (days)	Percentage of overdue treatment (%)	Average waiting time (days)
P1	0.44	1	14.29	1.09
P2	27.14	3	79.89	6.91
P3	41.36	14	74.55	18.11
P4	31.06	28	29.89	22.59

Table 2: Waiting time targets, percentage of overdue treatment and average waiting time of cancer patients at CHUM in 2017-2018 by patient category.

Patient categories along with their waiting time targets and percentage of overdue treatment and average waiting time are listed in Table 2.

The fraction length of patients at CHUM ranges from 10 to 165 minutes, with the majority of patients having 25-minute fractions (more than 50%). The distribution of fraction lengths by patient category can be seen in Figure 5. Figure 6 shows the number of fractions by patient category. From this data, we can see that most palliative patients have less than five fractions, while the number of fractions of curative patients distributes from 1 to 35 sessions with peaks at 5, 15, 20, 25, 30, and 33 sessions. A small portion of patients have more than 35 sessions.

5.2 Data generation

To describe how our instances are generated from CHUM's real data, we first define some terms. A *personal treatment plan* is a treatment plan tailored to a particular patient. It is decided by a team of physicians and specialists based on the patient's diagnosis and condition. For convenience, for the rest of the paper, we use the term *treatment plan* to refer to a *personal treatment plan*. A treatment plan provides information such as the number of fractions and fraction length. We create a *patient pool*, which consists of all patients and their treatment plans taken from CHUM's dataset. The patient pool does not include patients' personal information and their treatment timelines (dates and times of

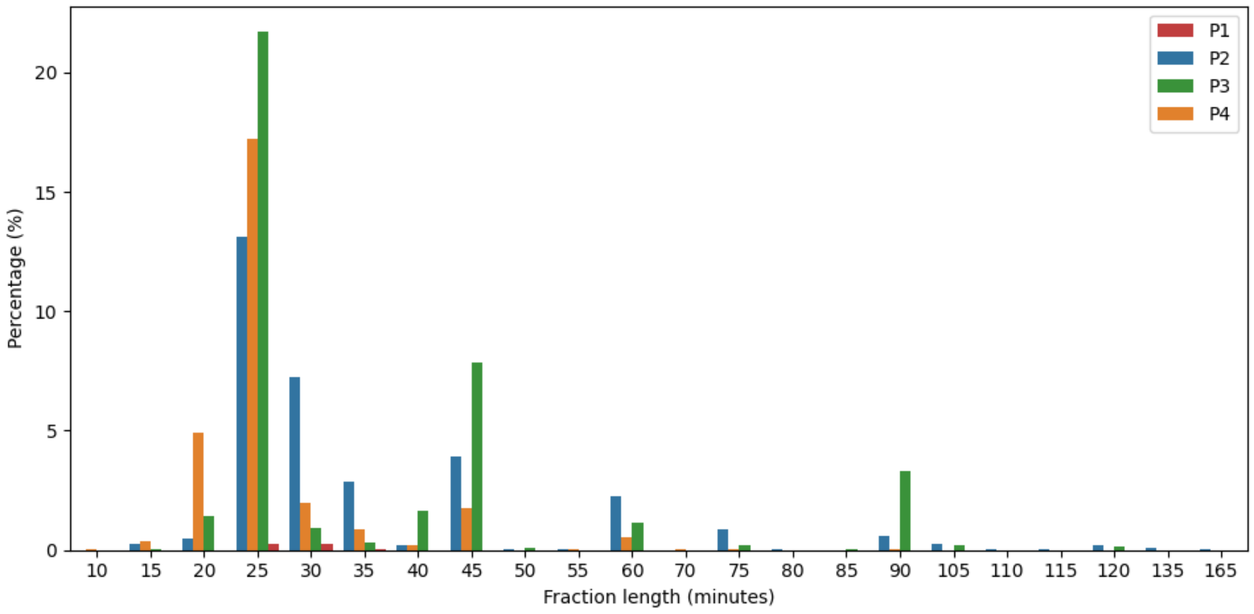

Figure 5: Fraction length by patient category.

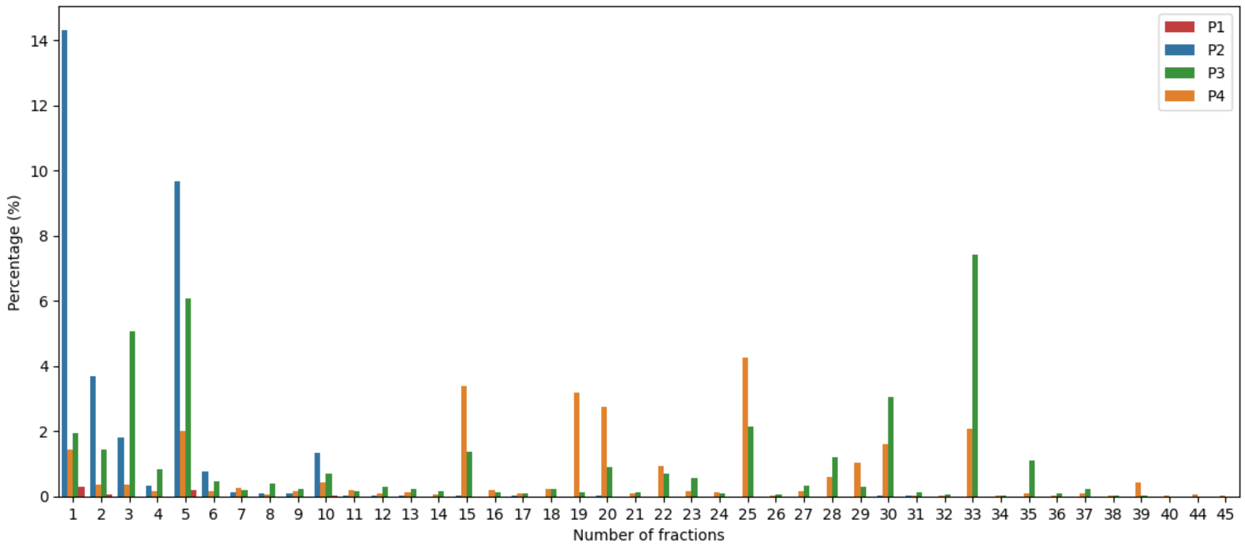

Figure 6: Number of fractions by patient category.

appointments).

A problem instance consists of an *instance scenario* and a *patient flow*. An instance scenario consists in a number of linacs and their capacity, along with a set of appointments made in the previous scheduling decisions. A patient flow consists of a daily flow of new patients admitted to the hospital during a *simulation period*. Given an arrival rate λ and a set of simulation days $\mathcal{K}$, a set of $|\mathcal{K}|$ numbers is generated respecting a Poisson distribution with the event rate of λ. For each day $d \in \mathcal{K}$, a corresponding number of virtual patients are generated and their admission dates are subsequently set as d. Each patient is assigned to a treatment plan selected randomly from the patient pool. A ready date for each patient is then generated based on their category. The ready date for P1 patients is always the same as their admission date. P2 patients have ready dates ranging from 0 to 2 days after their admission date while curative patients (P3 and P4) have ready dates ranging from 5 to 7 days after their admission date. The due date is calculated from the admission date and the patient's category listed in Table 2.

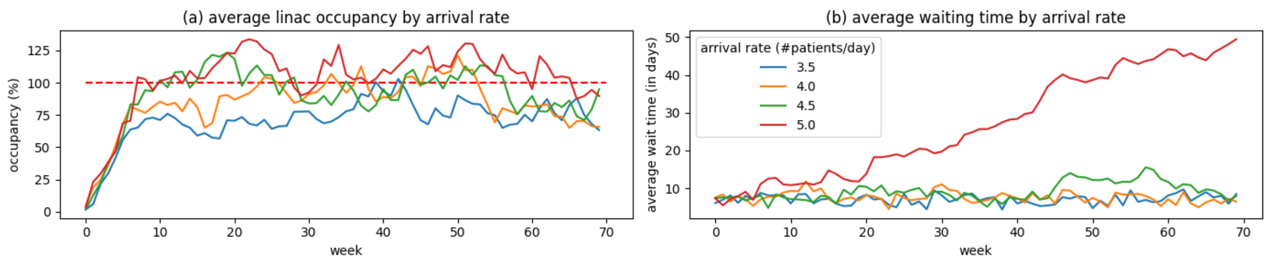

Figure 7: Capacity simulation on 3 linacs with arrival rates of 3.5, 4.0, 4.5, 5.0. The red dotted line represents linac capacity (360 time slots in this case). Starting from an arrival rate of 4.5, the average waiting time of patients increases over time.

To generate an instance scenario, we first generate an empty schedule and an instance flow. The greedy heuristic (Section 4.2.1) is run in a "warm-up" period to fill up linacs. When the occupancy of linacs reaches 90% of its capacity on any day, the warm-up simulation stops. The date with the highest occupancy is set as date 0 in the simulation period of the new instance. Appointments starting from date 0 are copied to the instance scenario of the newly-created instance.

5.3 Numerical results on generated instances

To examine the algorithms' performance on different hospital sizes and crowding levels, we generate instances with a wide range of linac numbers and arrival rates. Firstly, we identify a realistic range of arrival rates that a hospital with a given number of linacs can accommodate. For that purpose, two types of *capacity simulation* are utilized. In the first capacity simulation, we run the greedy algorithm in Section 4.2.1 without imposing the capacity constraints, i.e., we assume that the hospital has enough capacity to accept all treatment requests. We then measure and plot the average weekly occupancy rate. An example can be found in Figure 7a, which shows the results of the first capacity simulation for 3 linacs with arrival rates of 3.5, 4.0, 4.5, and 5.0, respectively. The red dotted line represents the linac capacity. From the plot, we can see that arrival rates 4.5 and 5.0 result in overload for multiple weeks during the simulation period. An arrival rate of 3.5 is in the safe zone, with the occupancy rate remaining mostly below the linac capacity, while an arrival rate of 4.0 can cause overload in some weeks but, is still generally within the safe zone.

In the second capacity simulation, we run the greedy algorithm with capacity constraints and then observe the progression of the weekly average waiting time. If the average waiting time increases over time, the arrival rate is too high for a hospital of the given size to handle. A stable average waiting time indicates that the arrival rate is reasonable. Figure 7b shows the results of the second capacity simulation for 3 linacs. As can be seen from the figure, an arrival rate of 5.0 causes the average waiting time to increase rapidly over time. This agrees with the observation from the first capacity simulation. The waiting time increase with an arrival rate 4.5 is less steep but is still potentially problematic for the hospital. Arrival rates of 3.5 and 4.0 are the most reasonable number for the hospital with stable lines. The two capacity simulations hence complement each other and we can conclude that arrival rates from 3.5 to 4.5 are reasonable for a hospital with 3 linacs, while an arrival rate of 4.5 is potentially problematic.

After identifying a reasonable range of arrival rates, we carry out experiments on those realistic instance settings. For each instance setting (a number of linacs and an arrival rate), 500 instances are generated, 400 of which are used for training the regression model while the remaining 100 are used for testing. We compare the six scheduling strategies listed in Section 4.2, namely greedy heuristic, daily batch scheduling, biweekly batch scheduling, weekly batch scheduling, and prediction-based approach. The algorithms are tested on the 100 test instances and the average waiting time and overdue time (in days) of patients by category are reported.

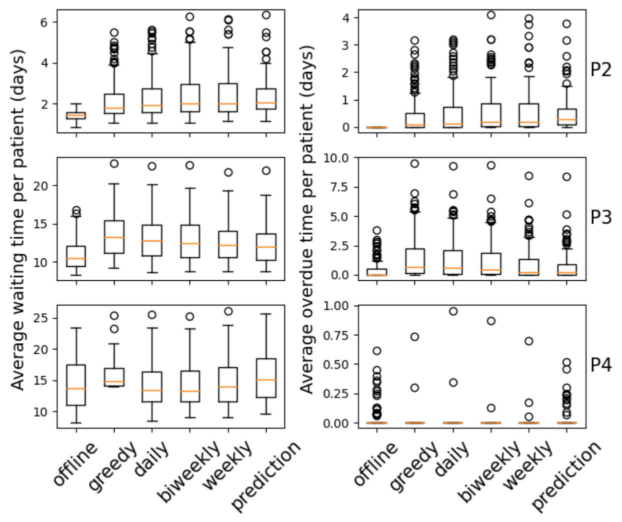

Figure 8: Simulation on 3 linacs with an arrival rate of 3.5.

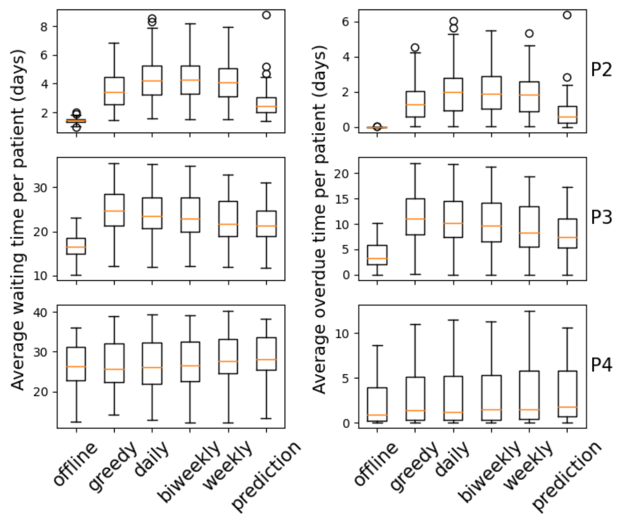

Figure 9: Simulation on 3 linacs with an arrival rate of 4.0.

Figure 8 shows the results for 3 linacs with an arrival rate of 3.5. The average waiting time is plotted on the left while the average overdue time is plotted on the right. In the test instances of this setting, there are no P1 patients or the number is not significant. As can be seen from the figure, in an ideal situation where all future arrivals are known in advance, the overdue time is insignificant in all patient categories. The greedy heuristic and the three batch scheduling strategies are similar in waiting time and overdue time. Overdue treatments mostly occur in P2 and P3 patients. The prediction-based approach shows slightly better average waiting time and overdue time in all patient categories. It also results in less deviation in waiting time and overdue time between patients.

Figures 9 and 10 show results for the same hospital size in more crowded scenarios, with arrival rates of 4.0 and 4.5, respectively. Looking at those figures, we observe an interesting trend that the more crowded the hospital is, the better the prediction-based approach performs compared to the batch scheduling strategies and the greedy heuristic. The most significant difference is at P2 patients, the most challenging category for other algorithms. As P2 patients account for almost 30% of patients and the treatment deadlines are urgent (only 3 days after their admission date), it is difficult to decide how much space should be reserved for their arrivals when scheduling curative patients. Preliminary results show that a lower rate would result in longer delays for palliative patients while a higher rate would result in a low occupancy rate and more overdue treatments for curative patients. The prediction-based approach, despite the lack of information compared to batch scheduling strategies, demonstrates the ability to learn from offline schedules how to delay the treatment of curative patients in order to make space for palliative patients based on the present state of the schedule. This explains the lower overdue rates associated with prediction-based scheduling in palliative patients.

To investigate the algorithms' performance in bigger hospitals, we run the experiments on two more hospital sizes, i.e., 5 linacs and 8 linacs. Numerical results for hospital sizes of 4 linacs, 6 linacs, and 7 linacs can be found in C.

For hospitals with 5 linacs, the realistic arrival rates identified through the capacity simulations are 6.5, 7.0, and 7.5 (see Figure 11). The waiting time and overdue time given by the proposed algorithms on those settings are shown in Figures 12, 13, and 14, respectively. What stands out in the figures is that the differences between the algorithms are more profound than in the previous smaller settings with 3 linacs, especially when the arrival rate increases. The prediction-based approach offers better waiting time and overdue time in palliative patients (both P1 and P2) compared to scheduling methods while maintaining a comparable waiting time in curative patients. Similar to the case of 3 linacs, the more crowded the hospital is, the better the prediction-based approach performs compared

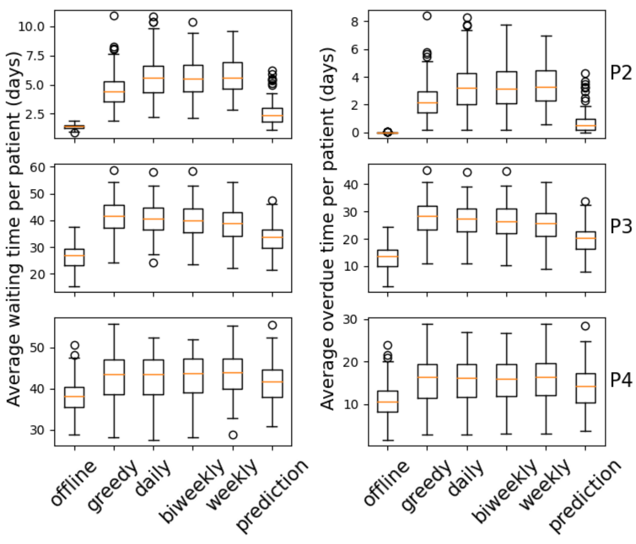

Figure 10: Simulation on 3 linacs with an arrival rate of 4.5.

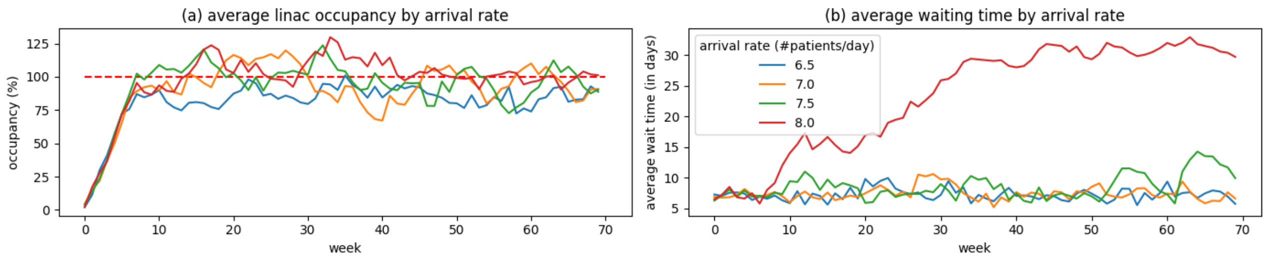

Figure 11: Capacity simulation on 5 linacs with arrival rates of 6.5, 7.0, 7.5 and 8.0. Starting from an arrival rate of 7.5, the average waiting time of patients increases over time.

to other approaches. Another observation from the boxplots is that the average waiting time (and overdue time) of P4 patients in greedy heuristic and batch scheduling is lower than the others, which also leads to higher overdue time in palliative patients. It shows that greedy heuristic and batch scheduling are not effective in delaying curative treatments to accommodate palliative ones.

Finally, for the largest setting that we study, 8 linacs, the reasonable arrival rates suggested by the capacity simulations are those with less than 12 patients per day, as an arrival rate of 13.0 causes the weekly average waiting time to increase over time (see Figure 15). We hence test three settings with arrival rates of 10.0, 11.0, and 12.0. The results are shown in Figures 16, 17 and 18, respectively. From these results, we derive two observations. First, in agreement with the experiments on the settings with 3 and 5 linacs, we conclude that the larger and more crowded a hospital is, the better the prediction-based approach performs compared to the other scheduling strategies. Second, our approach scales well on large instances. To our knowledge, most studies in the literature consider much smaller hospital sizes, and in reality, most hospitals do not have more than 8 linacs operating full-time. In large hospitals, however, as the number of linacs and patients are greater, the scheduling task becomes extremely challenging for most algorithms. Our prediction-based approach offers an online, fast approach with outstanding performance.

5.4 Numerical results on real instances

After confirming that our prediction-based approach performs well on generated instances, we now evaluate the algorithms on real instances from CHUM. We extract the real patient flow for a one-year period starting from July 2018. “Unusual” patients that are admitted during weekends or holidays (2

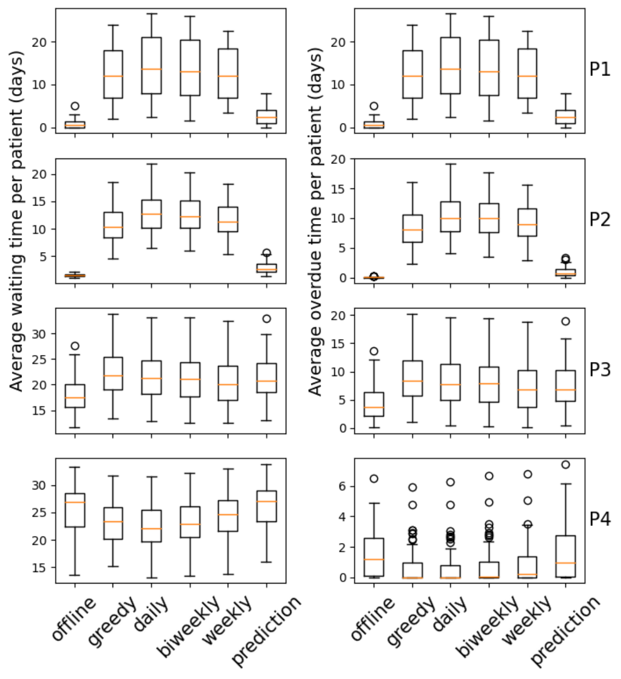

Figure 12: Simulation on 5 linacs with arrival rates of 6.5.

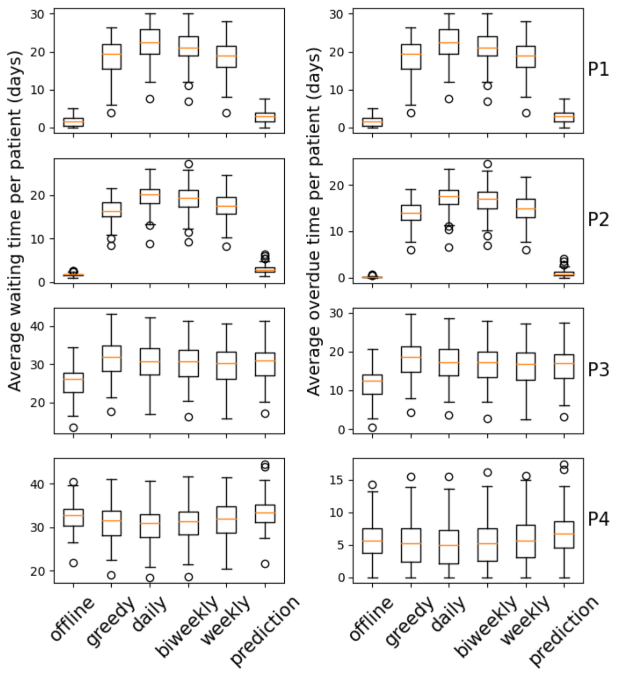

Figure 13: Simulation on 5 linacs with an arrival rate of 7.0.

out of 2962 patients) are removed from the dataset. As mentioned in Section 5.1, CHUM's facility consists of 12 linacs. Two special linacs (O and H) are special linacs reserved for very rare treatments. Linac J is a cyberknife and patients requiring this machine can not be treated on the generic linacs. We therefore remove all patients bounded to their special linacs (O, H, and J), who account for 9% of patients. This results in 2695 patients. Among the remaining 9 linacs, only 5 generic linacs (linacs A to E) operate full-time, 8 hours a day. The remaining 4 linacs operate on average for half of a working day, due to the lack of technicians. Therefore, we construct real instances using the extracted patient flow and 7 linacs operating full-time (8 hours a day). We then use the first 60 days to initialize a partially-filled schedule in the same manner described in Section 5.2. The next days in the patient flows are used for the actual patient flow of the instance. We fit the actual patient flow into a Poisson distribution and the best fit is $\lambda = 10.1$.

To train the regression model, we generate 500 instances with the scenario of 7 linacs and an arrival rate of 10.1 with a simulation period of 30 days. 400 instances are used for training and the remaining 100 instances are for testing. Similar to Section 5.3, we run all proposed scheduling strategies on 100 test instances. The results are presented in Figure 19. Similar to the results on generated instances in Section 5.3, the results show that the prediction-based approach outperforms the greedy approach as well as the batch scheduling strategies in palliative patients while maintaining a comparable waiting time and overdue time in curative patients.

We then test the real instance with a simulation period of 90 days. There are two reasons behind this decision. First, we aim to examine the performance of the algorithms in a long run. Second, we want to verify the performance of the regression model on a simulation period longer than that of the observed data during training (30 days). The results are shown in Figure 20 and Table 3. The first observation from the table is that all strategies result in no waiting and overdue treatment in P1. Regarding P2, the greedy heuristic gives the lowest waiting time of 1.8 days, while the prediction-based approach stands close with 1.83 days. The prediction-based approach, however, offers the lowest overdue time for P2 (0.19 days) among all scheduling strategies. Regarding curative patients, we observe that most overdue treatments occur in P3, which is in line with the real-life situation. The prediction-based approach continues to stand out with the lowest overdue time in P3 (2.72 days)

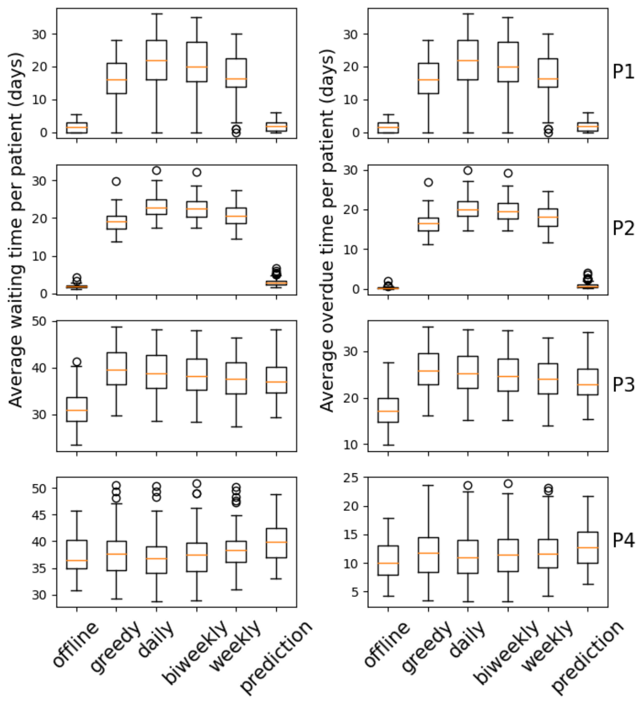

Figure 14: Simulation on 5 linacs with an arrival rate of 7.5.

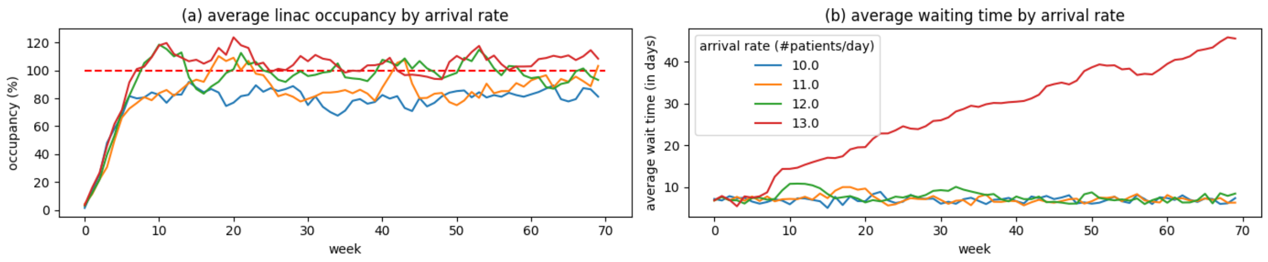

Figure 15: Capacity simulation on 8 linacs with arrival rates of 10, 11, 12 and 13. Starting from an arrival rate of 13, the average waiting time of patients increases over time.

and the second lowest overdue time in P4 (0.02 days, right behind daily batch scheduling with 0 overdue). This result is notable given that the simulation period is three times longer than that of the training data. The greedy heuristic results in high overdue time in curative patients, especially P3. The average waiting time and overdue time across all patient categories produced by the prediction-based approach are also the lowest, while the average occupancy rates during the simulation period are similar among scheduling strategies. A visualization of occupancy rates in different scheduling strategies can be found in Figure 21. As one can see, the prediction-based approach results in a more stable occupancy rate along the simulation period compared to the others.

6 Conclusions

Scheduling radiotherapy treatment is a difficult task due to the stochastic nature of patient arrivals. In this paper, we propose a machine learning-based approach that takes future arrivals into account when making a scheduling decision. A regression model is trained to recognize the scheduling patterns in offline solutions generated by an IP model, with the assumption of perfect knowledge of future arrivals. The trained regression model is then used to predict the waiting time of a given patient. The model is embedded in an online scheduling algorithm to schedule RT treatments for patients on ar-

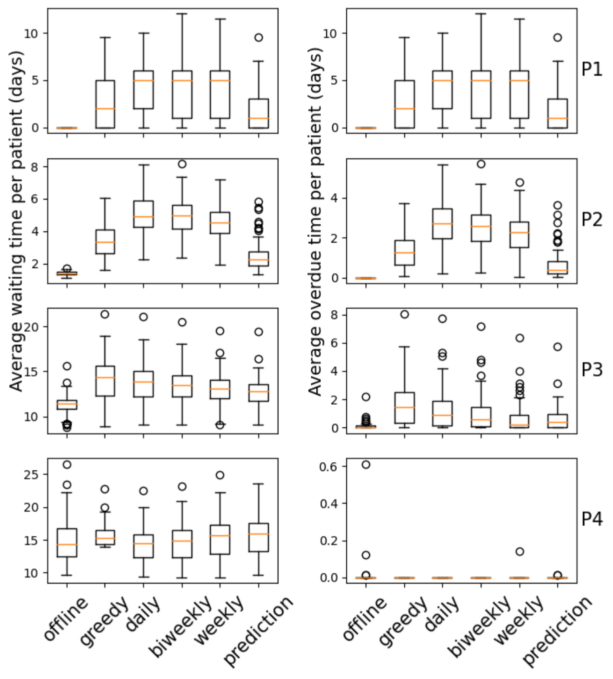

Figure 16: Simulation on 8 linacs with an arrival rate of 10.0.

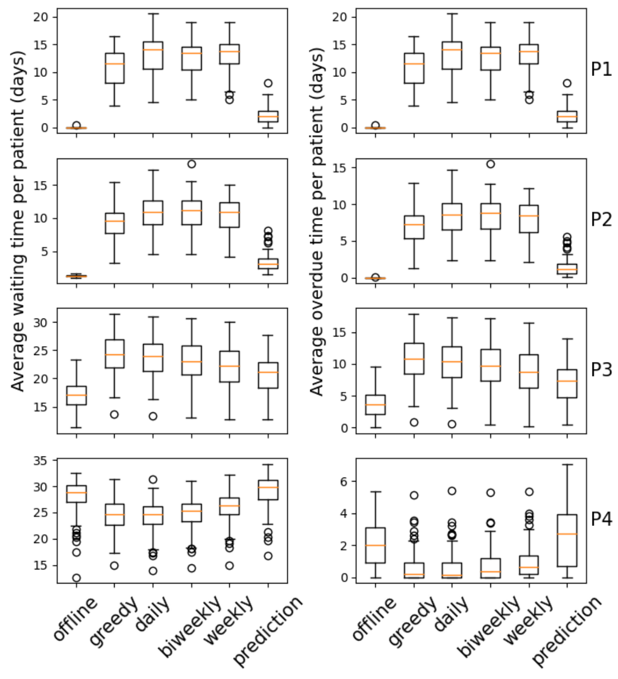

Figure 17: Simulation on 8 linacs with an arrival rate of 11.0.

Scheduling strategy	Avg. occupancy (%)	Waiting time (days)					Overdue time (days)				
		overall	P1	P2	P3	P4	overall	P1	P2	P3	P4
offline	96.76	9.63	0.00	1.45	10.66	15.59	0.36	0.00	0.00	0.41	0.63
greedy	92.89	13.96	**0.00**	**1.80**	18.32	19.61	2.04	**0.00**	0.20	5.23	0.06
daily	93.46	13.44	**0.00**	2.27	17.55	**18.56**	1.77	**0.00**	0.57	4.30	**0.00**
biweekly	93.57	13.32	**0.00**	2.24	16.99	18.85	1.56	**0.00**	0.54	3.73	0.03
weekly	93.34	13.29	**0.00**	2.05	**16.54**	19.41	1.38	**0.00**	0.37	3.35	0.06
prediction	93.56	**13.23**	**0.00**	1.83	15.92	20.11	**1.08**	**0.00**	**0.19**	**2.72**	0.02

Table 3: Average waiting time of patients in different scheduling strategies in the real instance.

rival. The algorithm is compared with a greedy heuristic and three batch scheduling algorithms using an IP model. All algorithms are evaluated in a simulation with a rolling horizon. The test instances are generated based on real data from CHUM to cover different hospital sizes and crowding levels. The numerical results show the superior performance of the proposed prediction-based approach. It results in a lower average overdue time and waiting time in palliative patients, while maintaining similar numbers in curative patients as compared with the greedy heuristic and the batch scheduling algorithms. It proves the algorithm's ability to "look ahead" while making scheduling decisions in order to prevent the lack of capacity for emergency patients. The proposed approach offers the advantages of online scheduling, i.e., convenience for patients and staff, while eliminating the disadvantage of being short-sighted due to the small search space. The approach also scales decently with instance sizes. The algorithm runs successfully for instances with up to 8 linacs. In the literature, instances of this size have been tackled only by heuristics and in a static setting without a rolling horizon. The prediction-based approach also shows its credibility in a real instance from CHUM. In addition, we show that despite the short simulation period of 30 days in test instances, the algorithm performs well for the instance with 90 days of simulation.

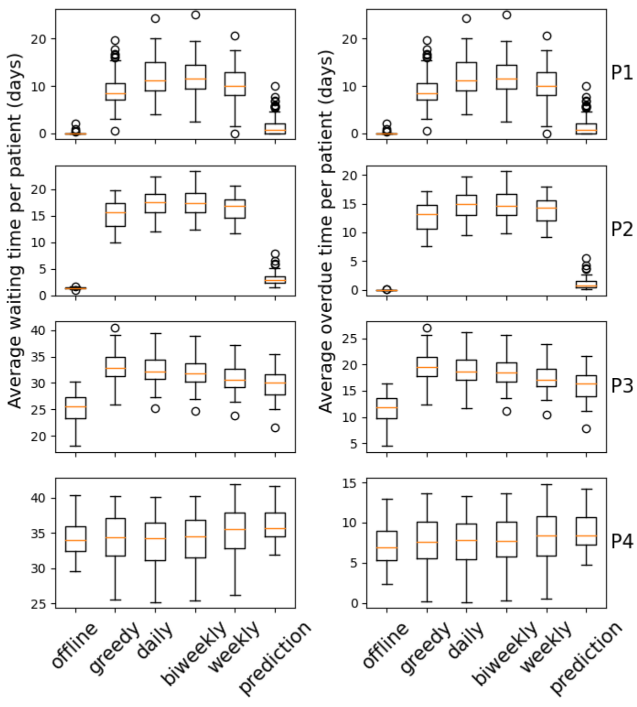

Figure 18: Simulation on 8 linacs with an arrival rate of 12.0.

Appendices

A Training regression models

In this section, we present the training results using different regression models. We present the results on different instance settings, from the easiest to the most difficult ones. Eight regression models are tested. For each model, each instance setting, we report the training time, mean squared error (MSE) and mean absolute error (MAE) on both training set and testing set.

Three instance settings are tested, from the easiest to the most difficult:

- 3 linacs, arrival rate of 3.5
- 6 linacs, arrival rate of 8.0
- 8 linacs, arrival rate of 12.0

The training results for the three settings are presented in Table 4, 5, and 6, respectively. From the tables, we observe that Random Forest and XGBoost consistenly give the best results in both training set and testing set. However, Random Forest requires more time for training and it becomes worse in difficult settings, i.e., when the size of the training test increases (51, 128.93, and 265.83 seconds in instance settings of 3, 6, and 8 linacs, respectively). Meanwhile, XGBoost takes only 7.71 seconds in 3 linacs and 37.8 seconds in 8 linacs. In addition, even though Random Forest gives the lowest error rates, the MAE in the testing set given by Random Forest and XGBoost are very close. A lower differences between the error rates in training test and setting test resulted by XGBoost also indicates that it is less prone to overfitting compared to Random Forest. Based on the above analysis, we choose XGBoost as the regression model for our algorithm.

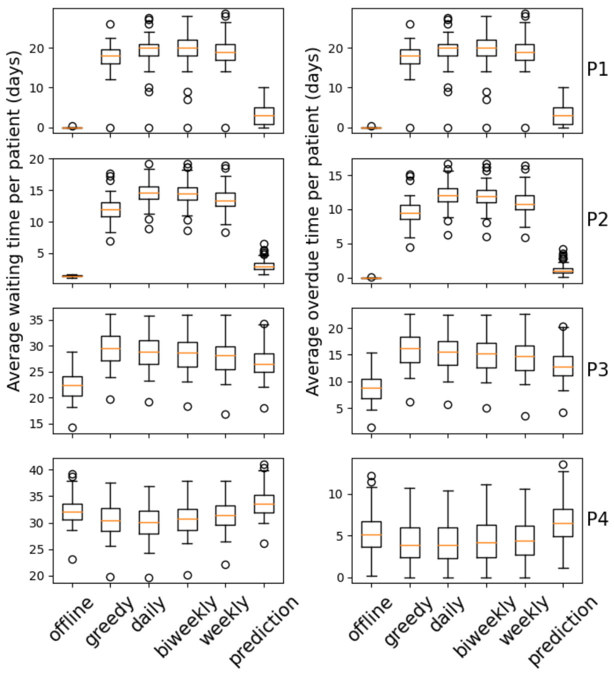

Figure 19: Testing results for 7 linacs, arrival rate 10.1.

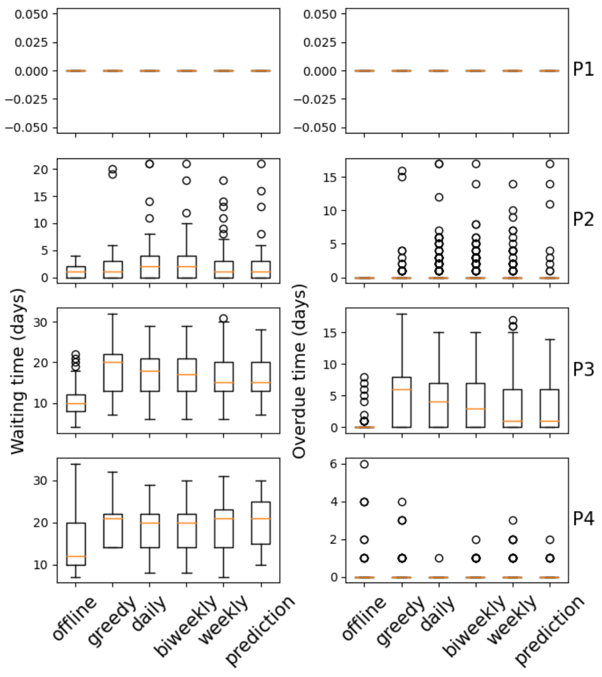

Figure 20: Testing results on the real instance with a simulation period of 90 days.

B Tuning reservation rates for palliative patients

As the percentage of linac capacity reserved for palliative patients has a great impact on the scheduling outcome, we conduct an experiment to identify the best reservation rate. The experiment is carried out on a setting of 4 linacs and an arrival rate of 6.0. 100 instances are tested with four different reservation rates, namely $5\%, 10\%, 15\%$, and 20%. As there is no reservation in offline scheduling, five scheduling strategies are tested, namely greedy heuristic, daily, biweekly, weekly, and prediction-

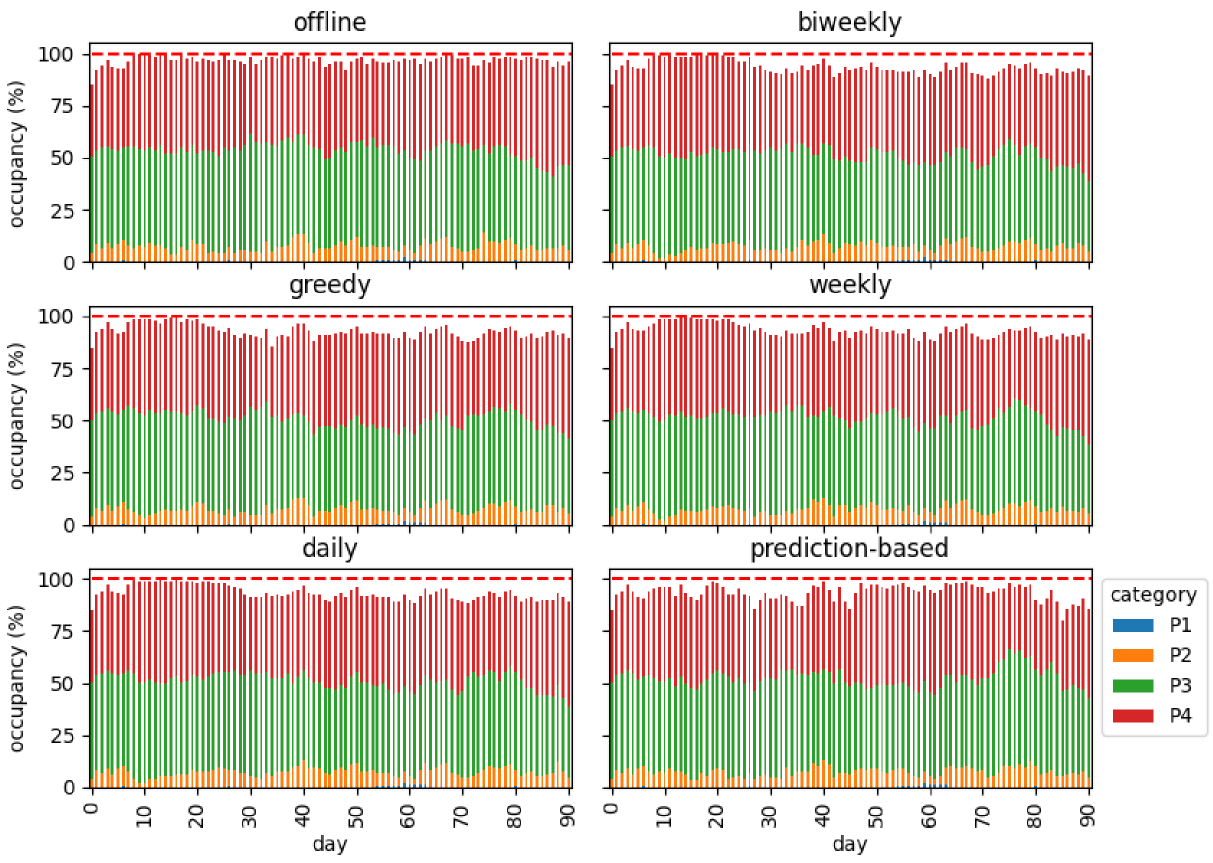

Figure 21: Occupancy rates of the real instance with different scheduling strategies.

based. The results are shown in Figure 22. Our first observation from the figure is that the higher the reservation rate, the lower the overdue time of palliative patients (P1 and P2). However, the trade-off coming with that low overdue rate is a high number of overdue times in curative patients (P3 and P4). For greedy heuristic and batch scheduling strategies (daily, biweekly, and weekly), with the consultant from CHUM, we decide that a reservation rate of 15% results in the best equilibrium between the delays in curative and palliative patients. For the prediction-based approach, the reservation rate of 10% is chosen due to a good overdue time in palliative patients while still maintaining a similar overdue time in curative patients compared to the remaining approaches.

C Additional numerical results

In this section, we present some additional numerical results for some more instance settings. For instances with 4 linacs, the capacity simulations (Figure 23) suggest that the realistic range of arrival rates is less than 6.0. We therefore test instance settings with arrival rates of 5.0, 5.5, and 6.0. The results for those settings can be found in Figure 24, 25, and 26, respectively.

For instances with 6 linacs, based on the results from the capacity simulation (Figure 27), we test instance settings with arrival rates of 7.0, 8.0, and 9.0. The results are plotted in Figures 28, 29, and 30, respectively.

For instances with 7 linacs, the suitable range of arrival rate is less than 10 (see Figure 31). As the results for an instance setting with 7 linacs and an arrival rate of 10.1 has already been shown in Section 5.4, we present the results for arrival rates of 9.0 and 10.0. Those results can be found in Figures 32 and 33.

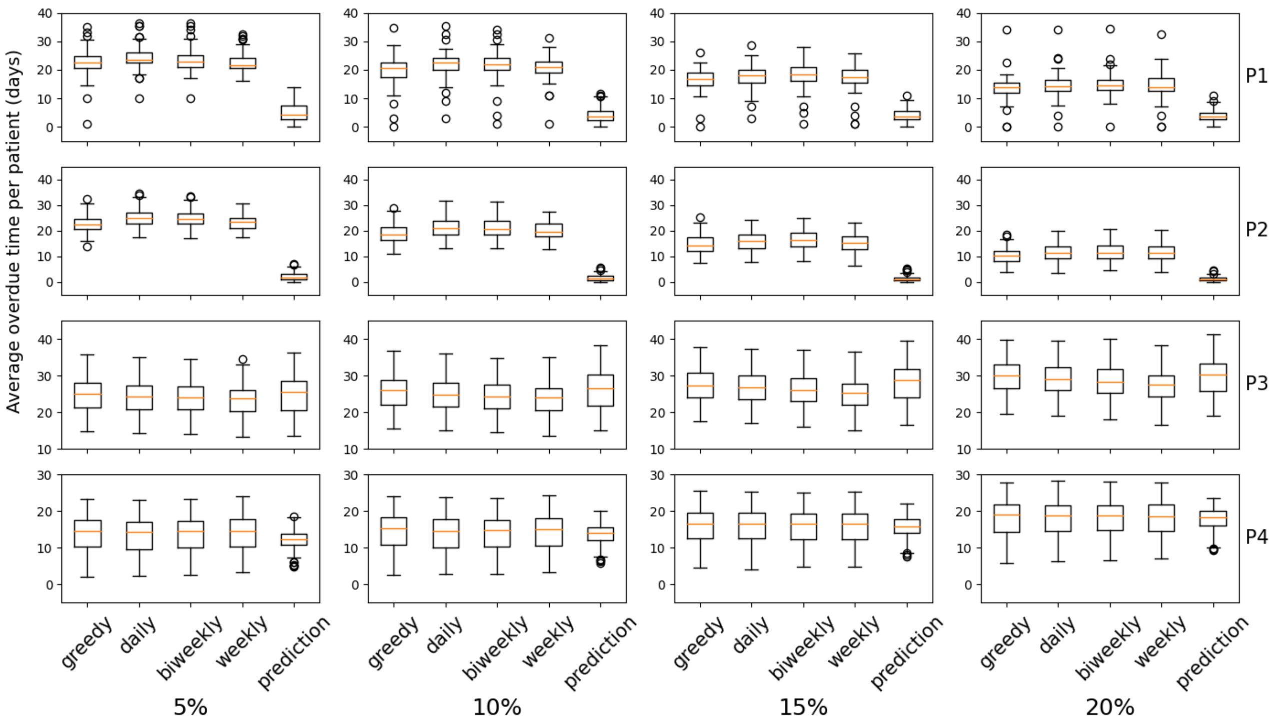

Figure 22: Average overdue time with different reservation rates.

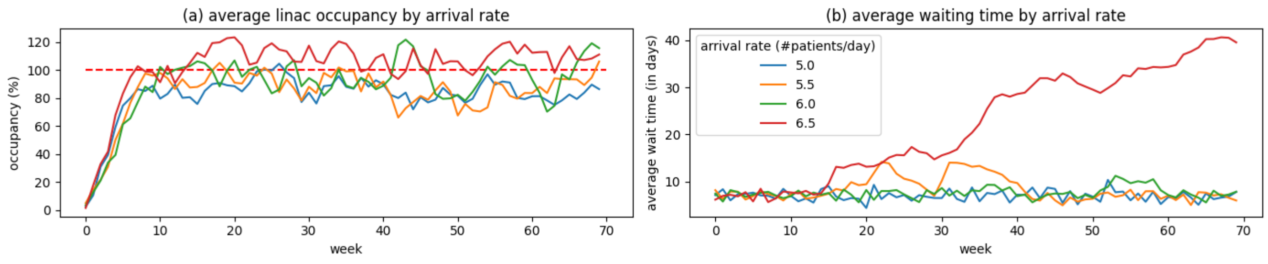

Figure 23: Simulation on 4 linacs with arrival rates of 5.0, 5.5, 6.0 and 6.5.

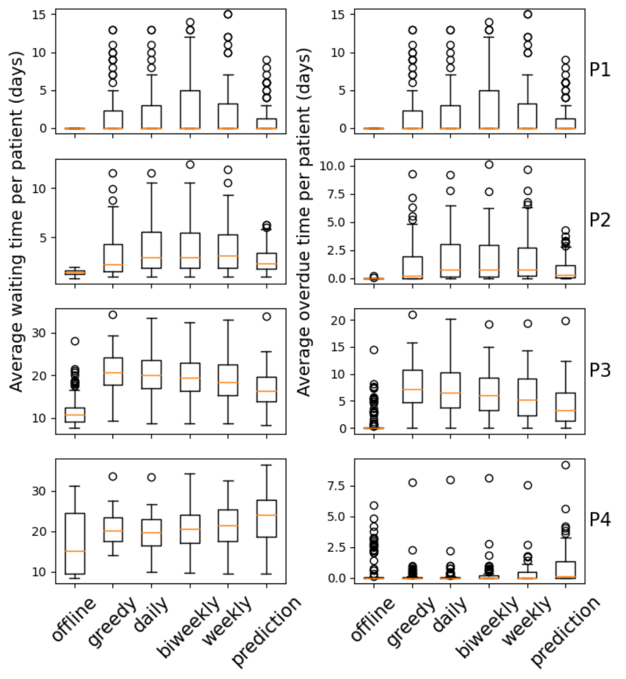

Figure 24: Simulation on 4 linacs with arrival rates of 5.0.

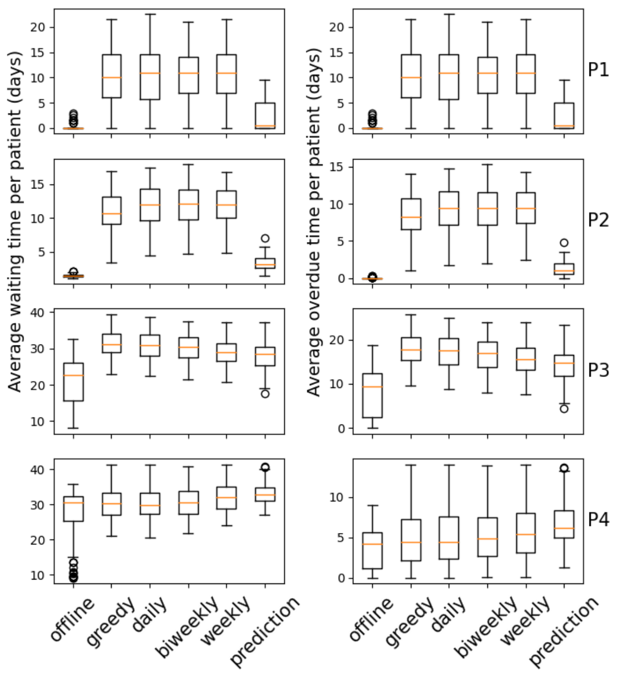

Figure 25: Simulation on 4 linacs with an arrival rate of 5.5.

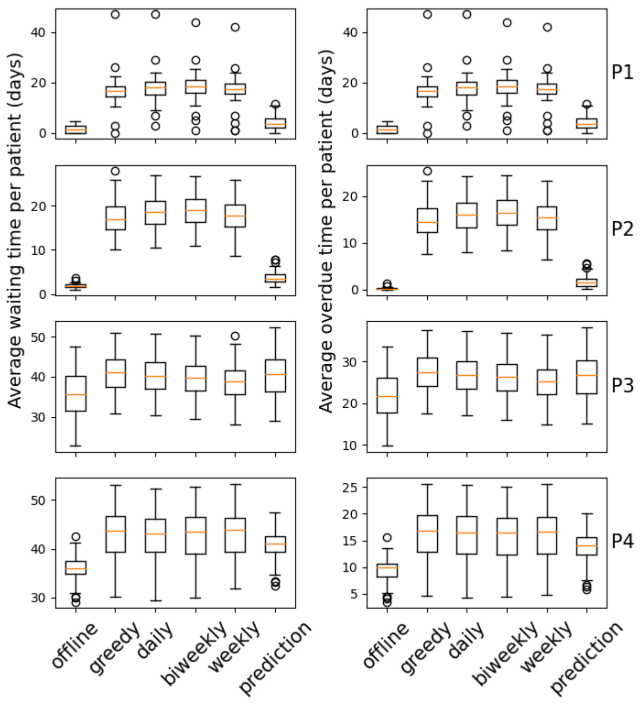

Figure 26: Simulation on 4 linacs with an arrival rate of 6.0.

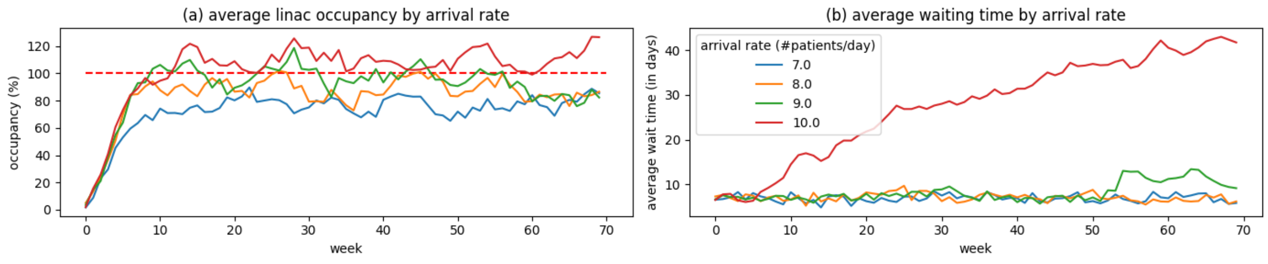

Figure 27: Simulation on 6 linacs with arrival rates of 7.0, 8.0, 9.0 and 10.0.

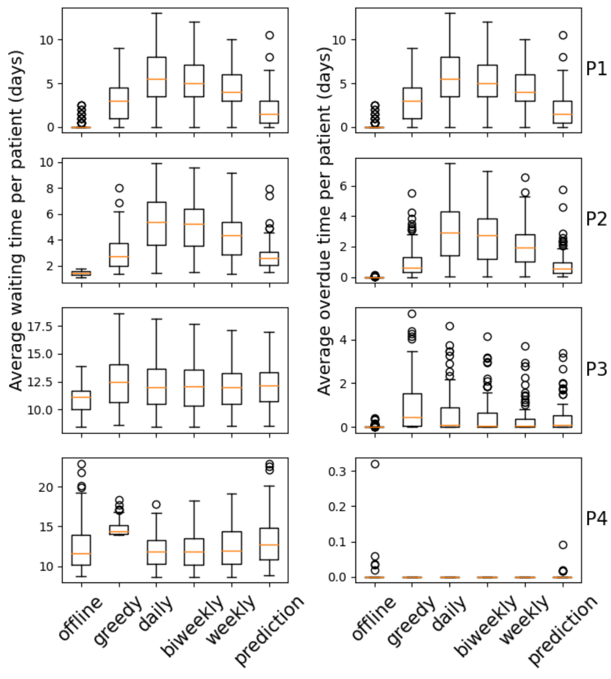

Figure 28: Simulation on 6 linacs with arrival rates of 7.0.

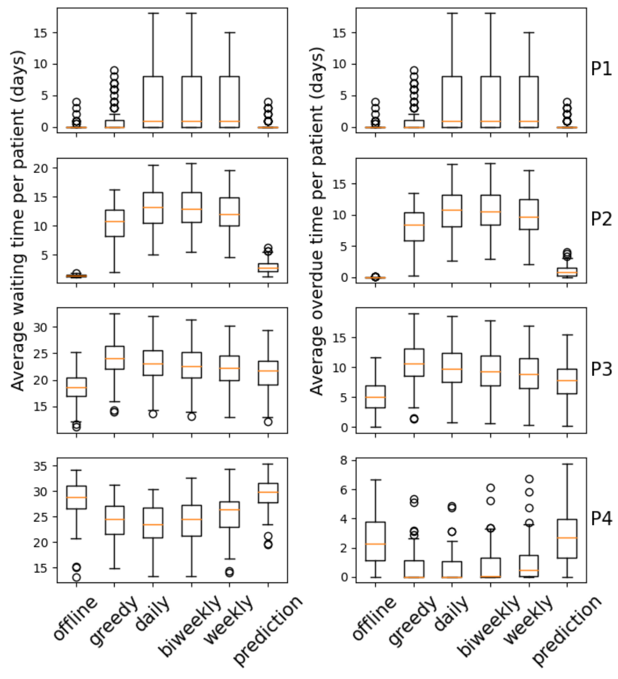

Figure 29: Simulation on 6 linacs with an arrival rate of 8.0.

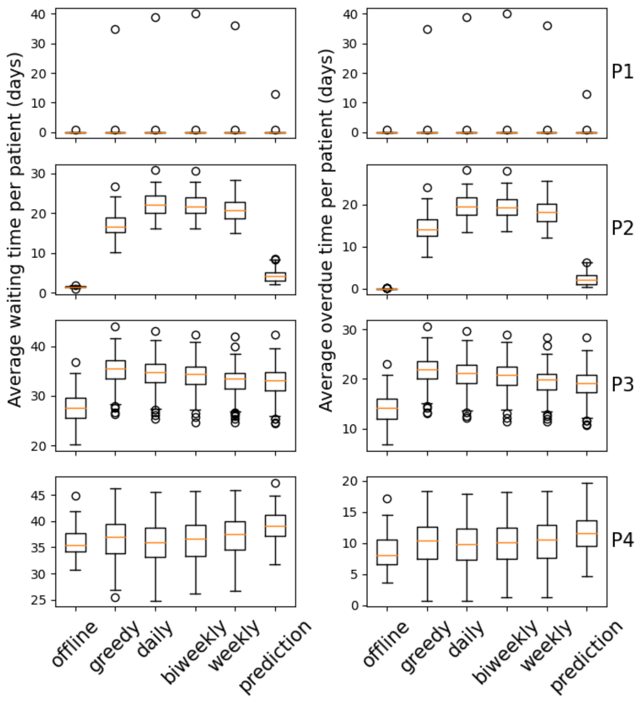

Figure 30: Simulation on 6 linacs with an arrival rate of 9.0.

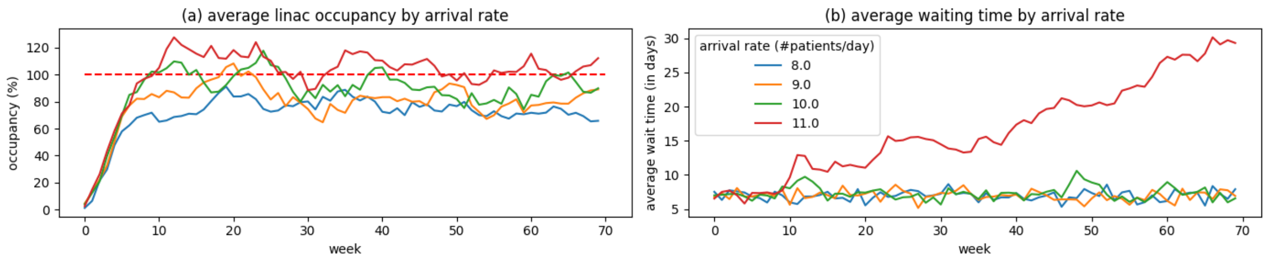

Figure 31: Simulation on 7 linacs with arrival rates of 8, 9, 10 and 11.

		Training		Testing	
	Training time	MSE	MAE	MSE	MAE
MLP	116.19	3.45	1.32	3.33	1.29
SGD	0.35	6.06	1.84	5.61	1.77
Lasso	0.44	5.97	1.81	5.52	1.74
ElasticNet	0.25	6.26	1.85	5.83	1.8
SVR	43.16	3.19	1.07	3.12	1.07
Decision Tree	0.84	2.41	0.48	6.59	1.4
Random forest	51	**0.38**	**0.39**	2.64	1.03
XGBoost	7.71	0.96	0.66	**2.44**	**0.97**

Table 4: Training results for instance setting of 3 linacs, arrival rate 3.5.

		Training		Testing	
	Training time	MSE	MAE	MSE	MAE
MLP	95.15	5.02	1.56	5.6	1.63
SGD	0.46	12.57	2.69	13.32	2.76
Lasso	1.54	12.47	2.65	13.27	2.72
ElasticNet	1	12.62	2.65	13.42	2.74
SVR	177.82	4.88	1.33	5.51	1.45
Decision Tree	2.14	3.4	0.65	8.12	1.58
Random forest	128.93	**0.42**	**0.37**	3.62	**1.13**
XGBoost	16.68	1.4	0.82	**3.55**	**1.17**

Table 5: Training results for instance setting of 6 linacs, arrival rate 8.0.

D References

[1] M. B. Barton, S. Jacob, J. Shafiq, K. Wong, S. R. Thompson, T. P. Hanna, and G. P. Delaney. Estimating the demand for radiotherapy from the evidence: a review of changes from 2003 to 2012. *Radiotherapy and oncology*, 112(1):140–144, 2014.

[2] E. K. Burke, P. Leite-Rocha, and S. Petrovic. An integer linear programming model for the radiotherapy treatment scheduling problem. *arXiv preprint arXiv:1103.3391*, 2011.

[3] E. Castro and S. Petrovic. Combined mathematical programming and heuristics for a radiotherapy pre-treatment scheduling problem. *Journal of Scheduling*, 15(3):333–346, 2012.

[4] Z. Chen, W. King, R. Pearcey, M. Kerba, and W. J. Mackillop. The relationship between waiting time for radiotherapy and clinical outcomes: a systematic review of the literature. *Radiotherapy and Oncology*, 87(1):3–16, 2008.

[5] C. Coles, L. Burgess, and L. Tan. An audit of delays before and during radical radiotherapy for cervical cancer–effect on tumour cure probability. *Clinical oncology*, 15(2):47–54, 2003.

[6] D. Conforti, F. Guerriero, and R. Guido. Optimization models for radiotherapy patient scheduling. *4OR*, 6(3):263–278, 2008.

[7] D. Conforti, F. Guerriero, and R. Guido. Non-block scheduling with priority for radiotherapy treatments. *European Journal of Operational Research*, 201(1):289–296, 2010.

[8] S. Frimodig and C. Schulte. Models for radiation therapy patient scheduling. In *International Conference on Principles and Practice of Constraint Programming*, pages 421–437. Springer, 2019.

		Training		Testing	
	Training time	MSE	MAE	MSE	MAE
MLP	321.15	7.01	1.86	7.29	1.9
SGD	0.74	19.04	3.27	19.38	3.28
Lasso	5.59	18.88	3.23	19.26	3.25
ElasticNet	2.44	19.03	3.23	19.45	3.26
SVR	699.36	7.47	1.69	7.7	1.75
Decision Tree	4.91	5.46	0.9	10	1.76
Random forest	265.83	**0.57**	**0.44**	**4.37**	**1.27**
XGBoost	37.8	2.57	1.07	**4.52**	**1.37**

Table 6: Training results for instance setting of 8 linacs, arrival rate 12.0.

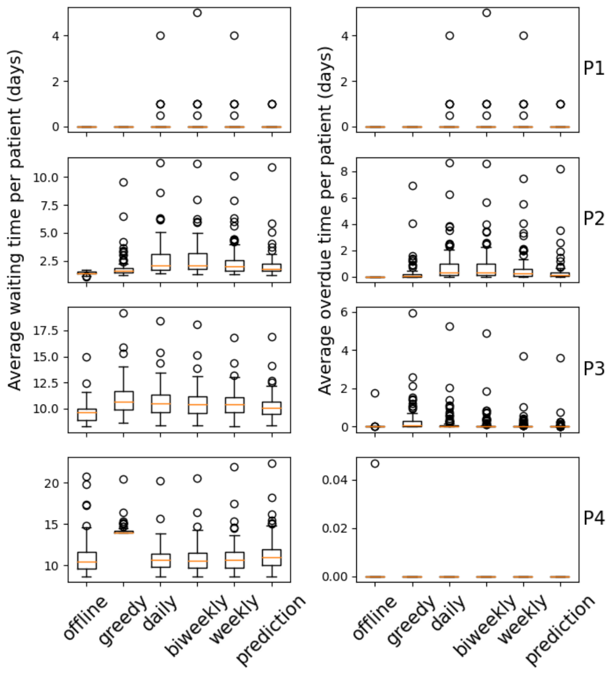

Figure 32: Simulation on 7 linacs with arrival rates of 8.0.

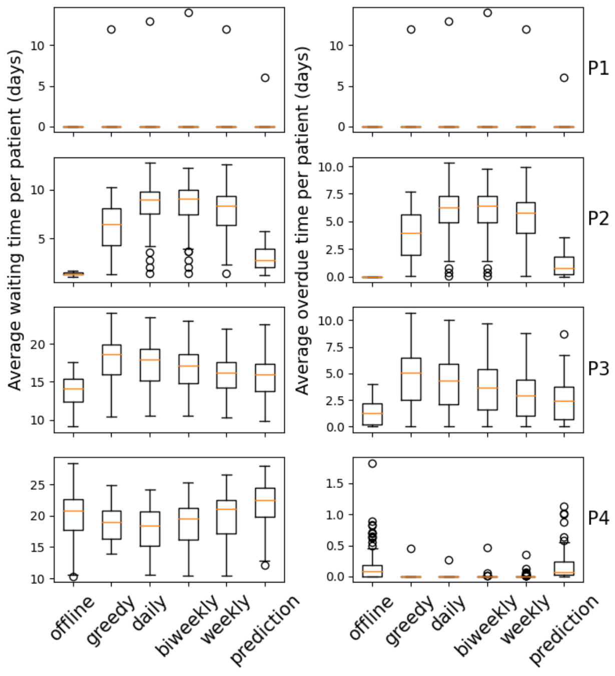

Figure 33: Simulation on 7 linacs with an arrival rate of 9.0.

[9] Y. Gocgun. Simulation-based approximate policy iteration for dynamic patient scheduling for radiation therapy. *Health care management science*, 21(3):317–325, 2018.

[10] T. Kapamara and D. Petrovic. A heuristics and steepest hill climbing method to scheduling radiotherapy patients. In *Proceedings of the 35th International Conference on Operational Research Applied to Health Services, Catholic University of Leuven, Belgium*. Citeseer, 2009.

[11] T. Kapamara, K. Sheibani, O. C. Haas, C. R. Reeves, and D. Petrovic. A review of scheduling problems in radiotherapy. In *Proceedings of the Eighteenth International Conference on Systems Engineering (ICSE2006), Coventry University, UK*, pages 201–207, 2006.

[12] E. Larsen, S. Lachapelle, Y. Bengio, E. Frejinger, S. Lacoste-Julien, and A. Lodi. Predicting tactical solutions to operational planning problems under imperfect information. *INFORMS Journal on Computing*, 2021.

[13] A. Legrain, M.-A. Fortin, N. Lahrichi, and L.-M. Rousseau. Online stochastic optimization of radiotherapy patient scheduling. *Health care management science*, 18(2):110–123, 2015.

[14] W. J. Mackillop. Killing time: the consequences of delays in radiotherapy. *Radiotherapy and Oncology 84.1*, 1(4), 2007.

[15] J. Maschler and G. R. Raidl. Particle therapy patient scheduling with limited starting time variations of daily treatments. *International Transactions in Operational Research*, 2018.

[16] J. Maschler, M. Riedler, M. Stock, and G. R. Raidl. Particle therapy patient scheduling: First heuristic approaches. In *Proceedings of the 11th Int. Conference on the Practice and Theory of Automated Timetabling. Udine, Italy*, 2016.

[17] S. Petrovic and P. Leite-Rocha. Constructive approaches to radiotherapy scheduling. In *Proceedings of the World Congress on Engineering and Computer Science*, pages 722–727, 2008.

[18] S. Petrovic, W. Leung, X. Song, and S. Sundar. Algorithms for radiotherapy treatment booking. In *25th Workshop of the UK planning and scheduling special interest group*, pages 105–112, 2006.

[19] T.-S. Pham, L.-M. Rousseau, and P. De Causmaecker. A two-phase approach for the radiotherapy scheduling problem. *Health Care Management Science*, pages 1–17, 2021.

[20] A. Saure, J. Patrick, S. Tyldesley, and M. L. Puterman. Dynamic multi-appointment patient scheduling for radiation therapy. *European Journal of Operational Research*, 223(2):573–584, 2012.

[21] S. Tyldesley, G. Delaney, F. Foroudi, L. Barbera, M. Kerba, and W. Mackillop. Estimating the need for radiotherapy for patients with prostate, breast, and lung cancers: verification of model estimates of need with radiotherapy utilization data from british columbia. *International Journal of Radiation Oncology* Biology* Physics*, 79(5):1507–1515, 2011.

[22] B. Vieira, E. W. Hans, C. van Vliet-Vroegindeweij, J. Van De Kamer, and W. Van Harten. Operations research for resource planning and-use in radiotherapy: a literature review. *BMC medical informatics and decision making*, 16(1):1–11, 2016.

[23] B. Vieira, D. Demirtas, J. B. van de Kamer, E. W. Hans, L.-M. Rousseau, N. Lahrichi, and W. H. van Harten. Radiotherapy treatment scheduling considering time window preferences. *Health care management science*, pages 1–15, 2020.

[24] P. Vogl, R. Braune, and K. F. Doerner. Scheduling recurring radiotherapy appointments in an ion beam facility. *Journal of Scheduling*, 22(2):137–154, 2019.