Synthetic Instance Generation for the Radiotherapy Scheduling Problem

Chiara Camilla Rambaldi Migliore[1,2] [0009−0000−1581−6086], Nysret Musliu[3] [0000−0002−3992−8637], Giovanni Iacca[1] [0000−0001−9723−1830], and Marco Roveri[1] [0000−0001−9483−3940]

[1] University of Trento, Italy
[2] University of Pisa, Italy
{cc.rambaldimigliore,marco.roveri,giovanni.iacca}@unitn.it
[3] Technische Universität Wien, Austria
nysret.musliu@tuwien.ac.at

Abstract. The Radiotherapy Scheduling Problem (RTSP) concerns finding an optimal appointment schedule for patients undergoing radiation treatments. Given its considerable impact on clinical outcomes, this problem has been studied in previous research, relying on datasets obtained from various hospital settings; however, such real-world datasets are rarely made public–mainly for ethical approval issues–while synthetic problem instances are rare and not standardised, which makes the comparison between scheduling algorithms harder. To address this gap, we propose a Benchmark Instance Management Tool that comprehends 1) a generator of new instances that samples from statistical distributions tuned on two real-world reference datasets from a big Belgian cancer centre and a small Italian radiotherapy unit, 2) a set of solvers, such as Mixed Integer Linear Programming, Simulated Annealing and two heuristics, and 3) a validator to check solution feasibility. Moreover, we propose a standardised input and output format. The generated instances are compared to real-world data by means of Instance Space Analysis on features extracted directly from instances. The results reveal two sharply separated clusters for the real-world reference datasets in the 2-dimensional feature space, while the newly generated instances not only populate those clusters but also fill the gap between them.

Keywords: Benchmark · Scheduling · Radiotherapy

1 Introduction

New cancer cases are predicted to exceed 27 million new diagnoses worldwide per year by 2040 [23]. Among the currently available cancer treatments, radiation therapy is widely used and is considered to be effective, especially for some specific types of cancer [15]. It consists of radiation delivery to the tumour areas in a set of fractionated sessions (called *fractions*), and an early start of treatment, followed by a precise planning of sessions, can be crucial for achieving optimal treatment results [9]. Thus, the Radiotherapy Scheduling Problem

(RTSP)–which consists of finding optimised appointment schedules balancing patient needs, resource constraints, and treatment efficacy–is a crucial field of study with major social implications.

The RTSP has been addressed by several authors [20] by introducing various problem formulations and algorithms, ranging from exact Operational Research (OR) methods [13, 7, 8] to metaheuristics [4, 22, 21, 2]. However, existing studies usually rely on data collected in specific hospitals, or generated from them, without any specific standard for input, problem formulation, or output. This contingency makes it difficult to compare how different algorithms perform on the RTSP. Moreover, the majority of the datasets are not public–mainly for ethical approval issues–or, when generated, the parameter and sampling distributions are not accurately described, making it difficult to reproduce them. To address this gap, following the state-of-the-art formulation of the problem and the solvers described and used in [7, 14], we present the following contributions:

- We ground our study in two real-world radiotherapy settings: Iridium Netwerk Cancer Centre, a major cancer hub in Antwerp, Belgium [7] (hereafter **Netwerk**), and Azienda Sanitaria Universitaria Integrata del Trentino's radiotherapy unit in Trento, Italy (hereafter **ASUIT**), a smaller but highly active facility. Using these benchmarks, we extract the empirical statistical distributions of patient arrivals, their associated treatment plans, and the occupancy of linear accelerators (LINACs) over a specified time horizon.
- We introduce a new, fully parametrised and customisable instance generator that leverages the above statistical distribution to produce synthetic data closely mirroring real-world cases–bridging the gap between the large cancer-centre dataset from [7] and the smaller ASUIT cohort.
- We embed our generator into a Benchmark Instance Management Tool that not only creates new instances on demand, but also automatically solves them using state-of-the-art solvers derived from [14], validates every solution, and stores all results for seamless analysis and reuse.
- We extract features from **Netwerk**, **ASUIT**, and our newly generated instances, then apply Instance Space Analysis (ISA) [17] to project them into a 2-dimensional feature space. This shows where each instance lies and how effectively the new instances bridge the gap between cases from small and large hospitals.

The ISA results reveal a sharp separation between the **Netwerk** and **ASUIT** datasets, which form distinct clusters in the 2-dimensional feature space, making their difference in facility size clear. The generated instances show that our generator not only fills the gap between these clusters, but also populates the clusters themselves–strong evidence that the statistical analysis successfully captures the key characteristics of the two real-world instances.

The remainder of this article is organised as follows. Section 2 summarises the main related works. Section 3 presents the definition of the RTSP, formalising the problem input, solution, and objective. Section 4 describes the Benchmark Instance Management Tool, starting from the standardisation of the input and output format, continuing with the methodology for building the statistical dis-

tribution from which to sample, and with the description of the parameters the user can provide to the generator, ending with the summary of solvers and the validator. Section 5 introduces the Instance Space Analysis results by comparing the reference datasets with the newly generated ones on a 2-dimensional feature space. Finally, Section 6 draws conclusions and outlines future work.

2 Related Works

Scheduling problems in radiotherapy have been the subject of study for more than two decades. Early works such as [5, 10, 3, 16, 11] addressed variants of the RTSP using data from real cancer centres, but, to the best of our knowledge, none have made their datasets or code publicly available, and the generation processes were absent or only partially described. Throughout the literature, there is a lack of standardised and reproducible benchmarks. One exception is [4], where both a generator and instances were published; unfortunately, the associated website is no longer available. More recent works follow the common approach of generating data from real-world sources, but none of them are standardised, and often, datasets are not even publicly available. [19] provides good detail on the generator's parameters, and [18] subsequently publishes real-world data, albeit in a non-standardised format, while the solver code remains unavailable in both cases. [13] generates instances based on real-world data with explained distributions and publicly available data, but the dataset uses a custom format, and no code is released. Other works [1, 6] share instances derived from a Thai cancer centre, but provide neither the code nor sufficient details on the generation process to allow for replication.

A separate line of work addresses scheduling in ion beam or particle therapy facilities, most notably at MedAustron; [12, 21, 22, 2] all tackle variants of this problem, with some datasets made available. However, the formulation of particle therapy differs in some aspects compared to LINAC-based RTSP (that is, the focus of this work), making it impossible to use those generators and datasets.

The work most closely related to ours is [7], which generates instances from real-world data from the Iridium Netwerk Cancer Centre, in Belgium, and provides both a public real-world-like synthetic dataset and a partially described generation process. However, their work only presents instances from a large facility, and the code is not provided. For this reason, we generate new instances of various sizes and difficulties and provide the scientific community with a Benchmark Instance Management Tool to generate, solve, and validate new instances.

3 Problem definition

The RTSP involves scheduling all patients' **fractions**–i.e., individual radiotherapy sessions delivering a portion of the total prescribed dose–by assigning each a **day**, **time window** (a predefined part of the day), and **machine**, subject to a set of constraints. The formulation and solution quality metric used here follow [7], with a more detailed description provided in [14].

Problem Input. The problem has three main inputs: a list of **patients**, a **time horizon**, and a list of **machines**. Each patient is prescribed with a specific

treatment protocol, which defines the priority, the number of fractions to schedule and their duration, the machines allowed and preferred to perform the treatment, the allowed start days and the target day (i.e., the latest day after which it is considered late to start the treatment). Patients may or may not have a preference for the time window of the day in which to be scheduled. The time horizon is defined by the number of days considered for the scheduling. Each machine has a residual time capacity for each time window of each day of the time horizon, measured in minutes. Some machines may be partially or fully beam-matched, meaning that the same radiotherapy plan can be safely administered by any of them. Machines are considered fully matched when they are located in the same building; if they are in different buildings, they are partially matched (assuming a cancer centre may be located in more than one building).

Solution. A solution to the RTSP consists of the complete appointment calendar for each patient. A feasible solution satisfies the following criteria:

1) **Singularity**: Each patient receives at most one fraction per day, scheduled in exactly one time window of one machine.
2) **Consecutiveness**: Fractions must be scheduled on consecutive weekdays.
3) **Availability**: A fraction can only be assigned to a machine's time window if sufficient capacity remains.
4) **Starting**: Treatment must begin on a protocol-allowed weekday, no earlier than the patient's arrival and no later than $T - n_f$, where T is the time horizon and n_f the total number of the patient's treatment fractions.
5) **Eligibility**: Each patient may only be treated on machines allowed by their protocol.
6) **Protocol precedence**: Among patients sharing the same protocol and same start day, those with an earlier target day must start treatment no later than those with a later one.

Objective. The objective is to minimise a weighted sum of six objectives, plus an offset of 1, as in [7, 14]:

$$\text{Obj. Sum} = 1 + \alpha_1 f_1 + \alpha_2 f_2 + \alpha_3 f_3 + \alpha_4 f_4 + \alpha_5 f_5 + \alpha_6 f_6 \tag{1}$$

where: f_1 is the weighted sum of the days patients are waiting before starting the treatment (the weights are defined by `cost` field defined in Section 4); f_2 is the weighted sum of the days beyond the deadline for patients to start their treatment (the weights are the same as for f_1); f_3 is the number of times patients switch time windows between two fractions; f_4 is the sum of the deviations from the time window preferred by the patients over all fractions; f_5 is the number of patients' fractions scheduled on a non-preferred machine as per the treatment protocol; and, f_6 is the sum of switches between machines that are partially beam-matched, i.e., machines whose treatment plans are interchangeable (they can be thought of as clones), but located in different buildings. The alpha values used for our experiments are $(50, 100, 1, 0, 10, 10)$, corresponding to the first combination discussed in [14].

4 Benchmark Instance Management Tool

4.1 Input format

The format of the standardised dataset output by the generator and expected as input by the solver, is an input parameter file[4]. The input parameter file contains various fields related to *machines* and to *patients*. In the following, we present a description of each of them.

Machines fields. The input file contains simple information fields, such as the `machinesQty` and `timeWindowsQty`, stating the number of machines and the number of time windows for each machine. The machines' maximum capacities `machinesMaxCapacity` and the residual capacities `machinesCapacity` (i.e., the remaining capacity at the start day of the scheduling; the occupied minutes are taken by patients already scheduled on that date) are designed as arrays of length $l = \texttt{machinesQty} \cdot \texttt{timeWindowsQty}$ with index $i = m \cdot \texttt{timeWindowsQty} + tw$ where m and tw represent the machine and the number of time windows targeted. Lastly, a more complex field, `machineBeamMatching`, represents the beam-matching relation between machines: the field is a matrix $\texttt{machinesQty} \times \texttt{machinesQty}$ where each cell (m_i, m_j) can take values in $\{0, 1, 2\}$ if $i \neq j$ and value 2 if $i = j$. The value 0 means that m_i is not beam-matched with m_j, the value 1 means that m_i is *partially* beam-matched with m_j, while the value 2 means that m_i is *completely* beam-matched with m_j.

Patients fields. The `patientsQty` field indicates the total number of patients to be scheduled, while `daysQty` contains the number of days in the time horizon. The `patientsGroupedByProtocol` field is an array of length equal to the number of protocols, where each element is another array filled with the ids of those patients having that same protocol treatment. The patients' ids are ordered ascending by the target day of each patient within the same protocol. This information is then used to ensure the Protocol precedence criterion described in Section 3. A more complex field is `patientInfo`, which–for each patient–is a data structure containing:

- `priority`: integer in $\{1, 2, 3\}$, where 1 denotes the highest clinical urgency.
- `cost`: scalar penalty weight associated with the priority class, set to 10, 3, and 1 for priorities 1, 2, and 3, respectively; it is used to calculate the objective function described in Section 3.
- `dMin`: the earliest day on which the patient can begin the treatment.
- `dTarget`: the clinically recommended latest start day.
- `numFractions`: the total number of treatment fractions specified by patients' protocol.
- `fractionsDuration`: an array of length `numFractions` specifying the duration in minutes of each fraction.
- `allowedDays`: a binary array of length `daysQty` indicating with 1 the days allowed to begin the treatment and with 0 the not allowed ones.

[4] The input parameter file is an ad-hoc data structure stored as a pickle and json file.

– `twPref`: an optional integer that indicates the preferred time window. If the patient does not express a preference, a placeholder "`Null`" value is inserted.

Finally, the field `machineEligibility` is a matrix `patientQty` × `machinesQty` where each cell (p_i, m_j) can take a value in $\{0, 1, 2\}$ where 0 means that machine m_j is not eligible for patient treatment p_i, 1 means that it is allowed, while 2 means it is *preferred* among the ones allowed.

4.2 Solution format

The standardised solution format output by the solvers provided in the tool is a file containing a specific data structure composed by two main fields: `patientAppointments` and `startDays`. The first is a matrix `daysQty` × `patientsQty` where each cell (d_i, p_j) contains a tuple (a, b) where $a = m \cdot$ `timeWindowsQty` $+ tw$ is a combined index indicating the machine and time window in which the patient p_j will receive the treatment fraction, and b is the duration of that fraction. The latter field is an array of length `patientsQty` specifying the start day for each patient, used for efficient calculation of the objective function.

4.3 Reference datasets

The first dataset considered for generator development–i.e., **Netwerk**–is a public one, containing real-world-like synthetic data from the Iridium Netwerk Cancer Centre based in Antwerp, Belgium [7]. It features 10 machines distributed over different buildings and encompasses instances that consider either 2 or 4 time windows. To ground the generator in real-world data from a smaller hospital, we collected, cleaned, and analysed real-world anonymised data (which we refer to as **ASUIT**) we obtained from the Azienda Sanitaria Universitaria Integrata del Trentino radiotherapy unit based in Trento, Italy. This dataset features 4 machines all in the same building and, as **Netwerk**, instances consider both 2 and 4 time windows. To make the **ASUIT** dataset publicly available, we created synthetic, real-world-like instances by sampling from the real-world instances' distribution.

4.4 Generator methodology

The generator tool has been built by analysing the statistical distribution of machines' occupation and patient arrivals of both the **ASUIT** and **Netwerk** reference datasets, in order to have a realistic range for the parameters of the statistical distribution used for the generation.

Machine occupation generation. From the reference datasets, we calculate, for each day d, machine, and time window, the empirical mean and variance of the occupation across similar instances–those with the same machines and time windows quantity–, denoted by $mean_d^i$ and var_d^i, respectively. To capture overdispersion, we use a Negative Binomial($\mathcal{NB}$) distribution to generate the machine occupation for each day, estimating its dispersion parameter via the method of moments. For each day d, we aggregate these estimates across all machines and time windows by taking the median:

$$\hat{r}_d = median\left(r_d^i = \frac{(mean_d^i)^2}{var_d^i - mean_d^i}\right), \quad i \in [0, \text{machine\_qty} \cdot \text{tw\_qty}] \tag{2}$$

The estimator is valid when $var_d^i \neq mean_d^i$; otherwise, the data are not overdispersed and are consistent with a Poisson ($\mathcal{P}$) distribution. In such cases, we store a placeholder instead of r_d^i. Since the mean occupation of all machines' time windows across instances in both reference datasets follows a decreasing pattern over days, we fit a quadratic exponential regression:

$$\mu_d^i = e^{(\beta_0^i + \beta_1^i d + \beta_2^i d^2)}, \quad i \in [0, \text{machine\_qty} \cdot \text{tw\_qty}] \tag{3}$$

where μ_d^i represents the expected occupation on day d for a specific time window of a machine averaged across similar instances, and $\ln(\beta_0)$ controls the initial occupation level. Across all machines' time windows. Only fits with the coefficient of determination $R^2 \geq 0.75$–which is considered enough for a good fitting–are retained. From the accepted fits, we calculate $\hat{\beta}_2 = mean(\beta_2^i)$–which will then be used as is by the generator during the creation process–and we bound β_1 between β_1^{min}, the minimum observed fitted value of β_1, and $\beta_1^{max} = min(0, max(\hat{\beta}_1))$–a range derived from thorough analysis across all machine time windows. The actual values of $\hat{\beta}_0$ and $\hat{\beta}_1$ used by the generator when producing new instances are calculated by means of two parameters given by the user: the initial occupation ω, which determines $\hat{\beta}_0 = \ln(\omega \times \text{maximum\_capacity})$, and the decay velocity of the occupation $\nu \in [0, 1]$, which determines $\hat{\beta}_1 = \nu\beta_1^{\min} + (1 - \nu)\beta_1^{\max}$. The generator, thus, will calculate the estimated daily occupation mean $\hat{\mu}_d$ with the $\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2$ described above, and feed it to the Negative Binomial ($\mathcal{NB}$) if the above calculated $var_d^i \neq mean_d^i$ else to the Poisson ($\mathcal{P}$) to generate the occupation Occ_d for each day–which is bounded to the maximum capacity of each machine's time window–as follows:

$$\hat{\mu}_d = e^{(\hat{\beta}_0 + \hat{\beta}_1 d + \hat{\beta}_2 d^2)} \tag{4}$$

$$Occ_d = \begin{cases} \mathcal{NB}\left(\hat{r}_d, \dfrac{\hat{r}_d}{\hat{r}_d + \hat{\mu}_d}\right) & \text{if overdispersion} \\ \mathcal{P}(\hat{\mu}_d) & \text{otherwise} \end{cases} \tag{5}$$

To model beam-matched machines, the user specifies the number of partial and complete beam-matched sets, constrained to a maximum of machines_qty/2. The generator then randomly creates that specific number of sets, with each set's size sampled randomly, constrained by the number of available machines as $\text{setLen} = randInt(2, \text{machines\_qty}/\text{sets\_qty})$.

Patient arrival generation. For both reference datasets, the number of patients arriving each day is modelled as a Poisson distribution. The mean λ is provided by the user and should be chosen within the ranges observed in the reference data: $\lambda \in [3.6, 5.6]$ for **ASUIT** and $\lambda \in [9, 10.7]$ for **Netwerk**–calculated as the minimum and maximum per-instance daily mean across all instances. The preferred and allowed machines selected for each protocol are randomly generated according to the proportions taken from the reference datasets analysis. Finally, each patient is assigned a priority, sampled according to the priority proportions defined by the user. Given a priority, a protocol is then sampled from a discrete distribution over protocols of that specific priority, where each protocol's probability is proportional to its observed frequency in the **Netwerk**

dataset instances. Protocol attributes are also taken from the **Netwerk** dataset and are applied to both datasets since the **ASUIT** protocols are not standardly defined–i.e., the physicians decide the treatment ad hoc, not following a database of standard treatments.

4.5 Generator parameters

The instance generator takes a command-line argument that contains the file with the parameters to drive the generation. Those parameters can be divided into three different categories: **dataset**, **machines**, and **patients** configurations.

Dataset configuration. Three dataset parameters control the overall process. The `dataset_title` is a string identifier used to name the output directory where the generated instances are stored. The `instances_number` specifies how many independent instances are created in a single run. Lastly, the `seed` parameter sets the random seed for reproducibility.

Machines configuration. To generate the machines, their occupation, and their compatibility (beam-matching), the generator takes several parameters as input. Firstly, the number of machines and of time windows is stated by `machines_number` and `time_windows_number` parameters. Then, the maximum capacity of each machine for each time window is stated by the `capacities` parameter. The `initial_occupation_percentage` (ω) parameter is used along with the maximum capacity to state the initial occupation $\hat{\beta}_0$ for each machine and time windows as stated in Equation (6). Moreover,`occupation_decay_velocity` (ν) controls how quickly the pre-existing occupation decreases over time, interpolating between a fast-decay coefficient $\beta_1^{\min}$ and a slow-decay coefficient $\beta_1^{\max}$ combined as described in Equation (7). Lastly, the minimum quantity of complete and partial beam-matched machine sets is stated by `min_beam_matched_sets` and `min_partial_beam_matched_sets`.

$$\hat{\beta}_0 = \ln(\omega \cdot \texttt{capacities}) \qquad (6) \qquad\qquad \hat{\beta}_1 = \nu \cdot \beta_1^{\min} + (1-\nu) \cdot \beta_1^{\max} \qquad (7)$$

Patients configuration. As far as the patient arrival generation is concerned, the generator uses two parameters. The `arrival_mean`, which is the expected average of patient arrival per day, is used in a Poisson distribution as the λ parameter from which the actual number of patients arriving each day is sampled. Finally, the `priorities_percentage` parameter is a probability vector that specifies the proportion of patients who belong to each priority class.

4.6 Solvers and Validator

The tool also includes a suite of solvers, such as a Mixed Integer Linear Programming (MILP) solver, a Simulated Annealing (SA) algorithm, and two heuristics. The details of the implementation of these solvers are described in [14]. The validator script takes the path to the *input* and *solution* files and checks the feasibility of the solution, as described by Section 3. This gives the user the ability to check the feasibility of the solutions found by the various solvers.

5 Evaluation

Generated Instances. We generated 7 new datasets (from now on referred to as `d_#` where `#` is the number of the dataset) with different parameters [5]. All the datasets are generated using the same seed (198743), and start from the same occupation percentage ($\omega = 70\%$); `d_1` contains 100 instances, while `d_2` to `d_7` contain 30 instances each. Datasets from `d_1` to `d_5` contain half of the instances generated with 7 patients arriving on average every day, and half with 10 patients, while `d_6` and `d_7` have, respectively, an average number of patients arriving of 10 and 5, for all instances. Datasets `d_1`, `d_3`, `d_4`, `d_5` have 6 machines, while `d_2` has 4 machines; for all these datasets, the machines are divided into 2 time windows with 240 minutes of capacity each. Datasets `d_6` and `d_7` try to mimic the **Netwerk** and the **ASUIT** reference datasets, respectively. In fact, `d_6` has 10 machines, and half of the instances are generated with 2 time windows of 240-minute capacity each, and the other half with 4 time windows of 120-minute capacity each. On the other hand, `d_7` has 4 machines, half instances with 2 time windows with capacity 240 and 210 minutes, and the other half with 4 time windows with capacity 120, 120, 135, and 75 minutes. Datasets from `d_1` to `d_3` have an occupation decay velocity ν set to 0.75, and beam-matched (BM) sets and partial beam-matched (PBM) sets both to 0; the other datasets have an occupation decay velocity set to 0.5, and BM sets and PBM sets both to 1, with the exception of `d_6` configured to generate 2 BM sets and 3 PBM sets. Finally, the patient priority percentages (from the most important to the least important) are [0.3, 0.3, 0.4] for `d_1` and `d_4`, [0.6, 0.2, 0.2] for `d_2`, `d_3`, and `d_5`, [0.75, 0.2, 0.05] for `d_6`, and [0.2, 0.1, 0.7] for `d_7`.

Instance Space Analysis. Since the reference datasets come from two really different facilities in terms of number of machines and daily patient arrivals, we decided to use Instance Space Analysis (ISA) through the software *Matilda* [17] to assess how they spread in the feature space, finding they form two different and really distant clusters as expected. The newly generated datasets were purposely created so as to fill the space between the two clusters. We solved the new instances with two of the solvers among the ones provided by our tool, i.e., MILP and SA, and we extracted 95 direct features (directly from the instances, based on simple statistical aggregations). The ISA selects 10 features and builds a projection matrix to map them to a 2-dimensional space, as follows:

$$\begin{pmatrix} z_1 \\ z_2 \end{pmatrix} = \begin{pmatrix} 0.4924 & 0.3562 \\ -0.3678 & -0.5242 \\ 0.776 & 0.2218 \\ 0.3738 & 0.2447 \\ -0.7577 & -0.1519 \\ -0.3962 & 1.0747 \\ 0.187 & -0.6391 \\ -0.3649 & -0.0674 \\ -0.5144 & 0.7411 \\ -0.4718 & -1.0413 \end{pmatrix}^{\top} \cdot \begin{pmatrix} \text{patientsP1PerDayGini} \\ \text{patientsP2PerDayRange} \\ \text{patientsP3PerDayMedian} \\ \text{patientsP3PerDayStd} \\ \text{patientsP3PerDayGini} \\ \text{beamMatchedMachinesQ3} \\ \text{machineResidualCapacityPercentageByDayMin} \\ \text{machineResidualCapacityPercentageByDayRange} \\ \text{machineResidualCapacityPercentageByDayStd} \\ \text{allowedMachinesForPatientStd} \end{pmatrix} \tag{8}$$

[5] All the generated instances, along with the code, can be found at https://github.com/DIOL-UniTN/RTSP-Benchmark-Instance-Generator-Tool.git

Figure 1 presents the distribution of the different instances across the previously described 2-dimensional feature space. It can be seen that the two reference datasets constitute two clusters in opposite corners of the plot. It is worth noticing that the **ASUIT** dataset is split into "real" and "synthetic"; this is due to the fact that real data cannot be made public. For this reason, a synthetic, real-like dataset has been created to be made publicly available. Thanks to the ISA analysis, we can also verify that the synthetic version of the **ASUIT** dataset overlaps with the real one, making it safe to assume the two datasets are statistically similar (i.e., they are sampled from the same distribution). Datasets `d_1`, `d_2`, `d_3` cover another corner of the feature space, specifically the lower right one. Datasets `d_4` and `d_5` cover the centre of the space, while `d_6` and `d_7` mimic the distributions of the **Netwerk** and **ASUIT** reference datasets, respectively.

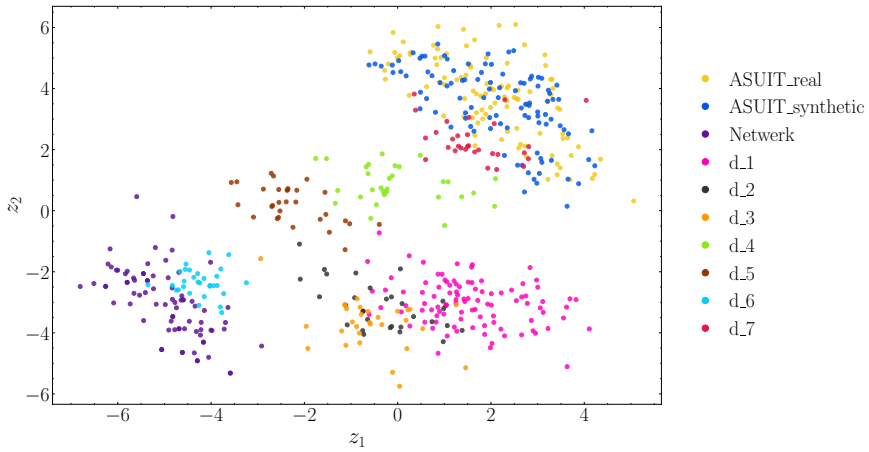

Fig. 1. A scatter plot visualizing how the different instances are spread in the 2-dimensional space found by multiplying the projection matrix by the processed feature vector.

6 Conclusion and Future Work

The RTSP is a relevant scheduling problem, which aims to ensure that timely treatments are performed and optimise patients' preferences and machine usage. Despite the problem being addressed in the existing literature, we found a lack of standardised and scalable available datasets and solvers. In this paper, we presented a Benchmark Instance Management Tool that can generate, solve, and validate new RTSP instances. The generator is constructed upon two reference datasets from real-world facilities of considerably different dimensions: a big cancer centre from Belgium [8] and a small radiotherapy unit from Italy (unpublished data). Those datasets have been statistically analysed to find the correct distributions and parameter ranges for the generation of new, realistic instances. The reference datasets' instances have been evaluated, along with some newly

generated ones, with the Instance Space Analysis (ISA) tool Matilda [17]. We extracted 95 direct features from the instances, solved them with MILP and SA solvers, and fed the results to the ISA tool, which mapped the 10 most relevant features into a 2-dimensional space. We analysed how the instances spread across the 2-dimensional space and concluded that the proposed generator is able to create real-world-like instances that can cover all the space between the instances obtained from two different-sized facilities and even explore other portions of the instance space. This makes the proposed Benchmark Instance Management Tool a significant asset in the RTSP domain, as it gives the opportunity to generate new instances, solve them with the given solvers or with new developed ones, and easily compare the results across all datasets. It could also be used to perform what-if analysis, e.g., aimed at understanding the effect of an increased number of patients, a varying number of machines, etc. Future works will focus on making the generator even further customisable, giving the users the possibility to add protocols, decide the specific sets of beam-matched machines, and preferred/allowed machines for protocols or patients.

Acknowledgments. This paper is based upon work from a scholarship supported by SPECIES (http://species-society.org).

References

1. Boonmee, C., Pisutha-Arnond, N., Chattinnawat, W., Muangwong, P., Nobnop, W., Chitapanarux, I.: Decision support system for radiotherapy patient scheduling: Thai cancer center case study. In: International Conference on Medical and Health Informatics. pp. 168–175 (2021)
2. Braune, R., Gutjahr, W.J., Vogl, P.: Stochastic radiotherapy appointment scheduling. Central European Journal of Operations Research pp. 1–39 (2022)
3. Burke, E.K., Leite-Rocha, P., Petrovic, S.: An integer linear programming model for the radiotherapy treatment scheduling problem. arXiv preprint arXiv:1103.3391 (2011)
4. Cares, J.P., Riff, M.C., Araya, I.: LS2R: A local search algorithm to solve scheduling radiotherapy problems. In: International Conference on Hybrid Intelligent Systems (HIS 2013). pp. 256–261. IEEE (2013)
5. Conforti, D., Guerriero, F., Guido, R.: Optimization models for radiotherapy patient scheduling. 4OR **6**, 263–278 (2008)
6. Emsamrit, N., Boonmee, C.: Mixed-Integer Linear programming for scheduling of radiotherapy patients. Engineering and Applied Science Research **51**(1), 106–116 (2024)
7. Frimodig, S., Enqvist, P., Carlsson, M., Mercier, C.: Comparing optimization methods for radiation therapy patient scheduling using different objectives. In: Operations Research Forum. vol. 4, p. 83. Springer (2023)
8. Frimodig, S., Enqvist, P., Kronqvist, J.: A Column Generation Approach for Radiation Therapy Patient Scheduling with Planned Machine Unavailability and Uncertain Future Arrivals (2023), arXiv:2303.10985
9. Higgins, G., Shah, K., Horne, P., Gilham, M., Bloomfield, D., et al.: The timely delivery of radical radiotherapy: guidelines for the management of unscheduled treatment interruptions. Londres: The Royal College of Radiologists (2019)

10. Jacquemin, Y., Marcon, E., Pommier, P.: Towards an improved resolution of radiotherapy, scheduling. In: 2010 IEEE workshop on health care management (WHCM). pp. 1–6. IEEE (2010)
11. Legrain, A., Fortin, M.A., Lahrichi, N., Rousseau, L.M.: Online stochastic optimization of radiotherapy patient scheduling. Health Care Management Science **18**(2), 110–123 (2015)
12. Maschler, J., Riedler, M., Stock, M., Raidl, G.R.: Particle therapy patient scheduling: First heuristic approaches. In: International Conference on the Practice and Theory of Automated Timetabling. pp. 223–244 (2016)
13. Pham, T.S., Rousseau, L.M., De Causmaecker, P.: A two-phase approach for the Radiotherapy Scheduling Problem. Health Care Management Science pp. 1–17 (2022)
14. Rambaldi Migliore, C.C., Stanicel, D., Musliu, N., Iacca, G., Roveri, M.: Comparing Optimization Models for Radiotherapy Scheduling. arXiv preprint (2026)
15. Roomi, M.W.: Radiation Therapy for Cancer Treatment. Journal of Otolaryngology Research & Reports **2**, 1–10 (2023)
16. Saure, A., Patrick, J., Tyldesley, S., Puterman, M.L.: Dynamic multi-appointment patient scheduling for radiation therapy. European Journal of Operational Research **223**(2), 573–584 (2012)
17. Smith-Miles, K., Muñoz, M.A.: Instance space analysis for algorithm testing: Methodology and software tools. ACM Computing Surveys **55**(12), 1–31 (2023)
18. Vieira, B., Demirtas, D., van de Kamer, J.B., Hans, E.W., Jongste, W., van Harten, W.: Radiotherapy treatment scheduling: Implementing operations research into clinical practice. PLOS ONE **16**(2), e0247428 (2021)
19. Vieira, B., Demirtas, D., van de Kamer, J.B., Hans, E.W., Rousseau, L.M., Lahrichi, N., van Harten, W.H.: Radiotherapy treatment scheduling considering time window preferences. Health Care Management Science **23**, 520–534 (2020)
20. Vieira, B., Hans, E.W., van Vliet-Vroegindeweij, C., van de Kamer, J., van Harten, W.H.: Operations research for resource planning and -use in radiotherapy: a literature review. BMC Medical Informatics Decis. Mak. **16**, 149:1–149:11 (2016)
21. Vogl, P., Braune, R., Doerner, K.F.: A multi-encoded genetic algorithm approach to scheduling recurring radiotherapy treatment activities with alternative resources, optional activities, and time window constraints. In: International Conference on Computer Aided Systems Theory. pp. 373–382. Springer (2017)
22. Vogl, P., Braune, R., Doerner, K.F.: Scheduling recurring radiotherapy appointments in an ion beam facility: Considering optional activities and time window constraints. Journal of Scheduling **22**(2), 137–154 (2019)
23. Wild, C.P., Weiderpass, E., Stewart, B.W.: World cancer report. Radiotherapy and oncology : journal of the European Society for Therapeutic Radiology and Oncology (2020)