An Enhanced Iterated Greedy Metaheuristic for the Particle Therapy Patient Scheduling Problem[1]

Johannes Maschler, Thomas Hackl, Martin Riedler, Günther R. Raidl

Institute of Computer Graphics and Algorithms, TU Wien
Favoritenstraße 9-11, 1040 Vienna, Austria
{maschler|riedler|raidl}@ac.tuwien.ac.at, t9.hackl@gmail.com

Abstract

The Particle Therapy Patient Scheduling Problem (PTPSP) arises in modern cancer treatment facilities that provide particle therapy and consists of scheduling a set of therapies within a planning horizon of several months. A particularity of PTPSP compared to classical radiotherapy scheduling is that therapies need not only be assigned to days but also scheduled within each day to account for the more complicated operational scenario. In an earlier work we introduced this novel problem setting and provided first algorithms including an Iterated Greedy (IG) metaheuristic. In this work we build upon this IG and exchange two main components: the construction phase and the local search algorithm. The resulting metaheuristic enhances the existing approach and yields in most of the considered benchmark instances substantially better results.

1 Introduction

Particle therapy is a relatively novel and highly promising option to provide cancer treatments. A proton or carbon beam is produced by either a cyclotron or a synchrotron and is directed into one of up to five treatment rooms, where patients are irradiated. Since several tasks have to be completed in a treatment room before and after an actual irradiation, the usually single available beam is switched between the available treatment rooms to maximize the throughput of the facility. Consequently, the main challenge is to arrange the individual treatments in such a way that idle times on the particle beam are minimized. We consider here the particle therapy treatment center MedAustron in Wiener Neustadt, Austria, which offers three treatment rooms.

The Particle Therapy Patient Scheduling Problem (PTPSP) addresses the midterm planning part of such a particle therapy treatment center and has been first introduced in our recent work [8]. In the PTPSP an effective plan has to be found for performing a larger number of therapies, each consisting of daily treatments (DTs) provided on 8 to 35 subsequent days. Therapies have to start on Mondays or Tuesdays between an earliest and a latest allowed starting day. After a therapy is started, the number of DTs that are provided each week has to stay between a lower and an upper bound. Moreover, there is a minimal and a maximal number of days that are allowed to pass between two subsequent DTs, and there has to be a break from the treatment of at least two consecutive days each week. The DTs have resource requirements that vary with time, but each specific resource is required at most once for a consecutive time period. These varying requirements originate from the different tasks involved in providing the treatments. Each resource can only be used by one DT at a time. Amongst others the considered resources involve the particle beam, treatment rooms, radio oncologists, and an anesthetist. In terms of the resource-constrained project scheduling literature (see for example [3]) DTs would be called activities with resource requests varying with time. Resources have for each working day (usually Mondays to Fridays) a regular availability period followed by an extended availability period in which they can be used, where the use of the latter induces (additional) costs. Furthermore, the availability of resources can be interrupted by so-called unavailability periods. The aim of PTPSP is to schedule a given set of therapies by determining days and times for all corresponding DTs while considering all operational constraints. The objective is to minimize the use of extended availability periods, while the therapies have to be completed as early as possible.

[1]We thank EBG MedAustron GmbH, Wiener Neustadt, Austria, for the collaboration on particle therapy patient scheduling and partially funding this work.

Midterm planning for classical radiotherapy has attracted the focus of the scheduling community starting with the works from Kapamara et al. [6] and Petrovic et al. [13]. Several further heuristic as well as exact approaches followed. Heuristic techniques range from a Greedy Randomized Adaptive Search Procedure (GRASP) [12] and steepest hill climbing methods [5, 14] to more advanced techniques using Genetic Algorithms (GAs) [10, 11]. Exact methods are based on Mixed Integer Linear Programming (MILP) models and consider different levels of granularity [1, 2]. All these works have in common that they assign treatments only to days, but do not sequence the treatments within a day. The reason is that in the considered scenarios linear accelerators are used which serve single treatment rooms exclusively. Hence, only a sequential processing of treatments is possible. This stands in contrast to PTPSP where the particle beam is shared between multiple treatment rooms and a finer-grained scheduling is necessary to maximize the throughput of the facility.

In our previous work [8] we formalized PTPSP as a MILP model. However, even solving a strongly reduced version of the model turned out to be practically intractable. Therefore, we proposed the therapy-wise construction heuristic (TWCH), which acts in two phases by assigning first all DTs to days (day assignment) and then scheduling the DTs on each day (time assignment). Moreover, a GRASP and an Iterated Greedy (IG) metaheuristic that are based on this construction heuristic were developed. Experiments indicated that the IG yields superior results in comparison to the GRASP. This is mainly due to the fact that the IG preserves substantial parts of the solution from one iteration to the next, and consequently, poor decisions made especially in the first phase of TWCH can be corrected in the course of the iterations. However, the IG proposed in [8] does not exhaust its full potential: Moving DTs between days might require to reevaluate the start times of all DTs, and this is done by simply dropping all start times of the considered days. In addition, we used a local improvement operator within the IG that is based on applying a randomized version of the time assignment phase of TWCH iteratively many times. Even though this local improvement operator is able to enhance solutions rather quickly this approach has the drawback that partly redundant work is repeatedly done, still yielding relatively similar solutions for the time assignments.

The aim of this work is to study improved variants of the IG metaheuristic from [8]. We propose novel destruction and construction methods that are able to keep relative timing characteristics of untouched DTs to a larger extent. Furthermore, we replace the so far rather simple local improvement operator by a local search method that considers a restricted DT exchange neighborhood. The new IG is compared to the IG from [8] and a variant of IG that uses the destruction and construction phase from [8] combined with the new local search method. Our experiments clearly indicate that the IG with the new destruction and construction methods as well as the new local search yields substantially better results than the other two variants.

The remainder of this work is structured as follows: After giving a formal problem definition in the next section, we present the enhanced IG metaheuristic in Section 3. The conducted experiments are then discussed in Section 4. Finally, Section 5 concludes this work with an outlook on future work.

2 Problem Definition

In the PTPSP a set of therapies $T = \{1, \dots, n_T\}$ has to be scheduled on consecutive days $D = \{1, \dots, n_D\}$ considering a set of renewable resources $R = \{1, \dots, n_R\}$.

Each therapy $t \in T$ consists of a set of DTs $U_t = \{1, \dots, \tau_t\}$. In the course of a therapy, the number of DTs applied per week has to be in the range from n_t^{twmin} to n_t^{twmax} and DTs have to be performed at least every $\delta_t^{\min} \geq 1$ and at most every $\delta_t^{\max} \geq \delta_t^{\min}$ days. In addition, between two weeks there have to be at least two days where no DT is performed. The set of possible start days for each DT $u \in U_t$ is given by the subset $\{d_{t,u}^{\min}, \dots, d_{t,u}^{\max}\} \subseteq D$ of days. For each DT $u \in U_t$ we are given a processing time $p_{t,u} \geq 0$ and a set of required resources $Q_{t,u} \subseteq R$. In the execution of a DT each resource $r \in Q_{t,u}$ is in general required during a part of the whole processing time specified by the time interval $P_{t,u,r} = [P_{t,u,r}^{\text{start}}, P_{t,u,r}^{\text{end}}) \subseteq [0, p_{t,u})$.

The planning horizon is structured into a subset $D' \subseteq D$ of working days on which the treatment

center is actually open and DTs can be scheduled on. Moreover, let $\bigcup_{v\in\{1,\dots,n_V\}} D'_v$ be the partitioning of D' into n_V subsets corresponding to the weeks. For each working day $d \in D'$ we have a fundamental opening time $\widetilde{W}_d = [\widetilde{W}_d^{\text{start}}, \widetilde{W}_d^{\text{end}})$ that limits the availability of all resources on the considered day.

Each resource $r \in R$ is available on a subset $D_r^{\text{res}} \subseteq D'$ of the working days. On such days the availability of each resource is defined by a regular service time window $W_{r,d} = [W_{r,d}^{\text{start}}, W_{r,d}^{\text{end}}) \subseteq \widetilde{W}_d$ that is immediately followed by an extended service window $\widehat{W}_{r,d} = [W_{r,d}^{\text{end}}, \widetilde{W}_d] \subseteq \widetilde{W}_d$. Moreover, for each resource $r \in R$ and each day $d \in D_r^{\text{res}}$, the availability of resource r may be interrupted by a set of unavailability intervals $\overline{W}_{r,d} = \bigcup_{w=1,\dots,\omega_{r,d}} \overline{W}_{r,d,w}$ with $\overline{W}_{r,d,w} = [\overline{W}_{r,d,w}^{\text{start}}, \overline{W}_{r,d,w}^{\text{end}}] \subset W_{r,d} \cup \widehat{W}_{r,d}$.

We represent a solution for the PTPSP as tuple (Z, S), where $Z = \{Z_{t,u} \in D : t \in T,\ u \in U_t\}$ denotes the days at which the DTs are planned and $S = \{S_{t,u} \geq 0 : t \in T,\ u \in U_t\}$ are the start times of the DTs on the respective days. A solution is feasible if all resource availabilities, precedence relations, and the remaining operational constraints are respected. The objective is to minimize the use of extended time over all resources R while finishing each treatment as early as possible. More formally, we aim at minimizing

$$\gamma^{\text{ext}} \sum_{r\in R} \sum_{d\in D_r^{\text{res}}} \max\left(0, \max_{\substack{t\in T, u\in U_t:\\ r\in Q_{t,u},\ Z_{t,u}=d}} (S_{t,u} + P_{t,u,r}^{\text{end}}) - W_{r,d}^{\text{end}}\right) + \gamma^{\text{finish}} \sum_{t\in T} \left(Z_{t,\tau_t} - Z_{t,\tau_t}^{\text{earliest}}\right), \quad (1)$$

where γ^{ext} and γ^{finish} are scalar weights and $Z_{t,\tau_t}^{\text{earliest}}$ is a lower bound on the earliest possible finishing day for the last DT of therapy t (see [8]).

Note that the definition of DTs stated here differs from the one given in [8], where DTs are composed of consecutively executed activities that are associated with minimum and maximum time lags. The simplification here is motivated by the fact that in practice the possibility to have different minimum and maximum time lags between two activities is not expected to be exploited in midterm planning. Consequently, time lags may either be replaced by "dummy" activities of fixed length or, as we do here, the subdivision of DTs into activities can be replaced by the time intervals $P_{t,u,r}$ specifying at which times which resources are needed.

3 Iterated Greedy Approach

An IG [4] algorithm starts with an initial solution and then repeatedly applies a destruction phase dissolving part of the solution, followed by a construction phase that completes the solution again, until a termination criterion is reached. The initial solution is usually obtained by applying a construction heuristic. The destruction phase removes randomly selected components from the incumbent solution, that are then reinserted by a greedy reconstruction method in the construction phase. Afterwards, an acceptance criterion is evaluated to determine whether the newly generated solution replaces the incumbent solution. Frequently, a local search algorithm is applied to the initial solution and after the construction phase to further boost the performance. In the following sections we discuss the components of the proposed IG.

3.1 Initial Solution

The initial solution is computed using TWCH from [8]. This construction heuristic acts in two phases, first assigning all DTs to days (day assignment) and afterwards determining the actual starting times of the DTs (time assignment).

In the day assignment phase therapies are processed in the order of the latest possible starting day of their first DT. For each selected therapy the corresponding DTs are then allocated sequentially to days, starting with the first DT. For each considered DT, all feasible days between the earliest and latest starting day w.r.t. the constraints imposed by the DT's predecessors are evaluated. The DT is then assigned to the candidate day minimizing on the one hand the expected use of extended service windows for the current and all subsequent DTs and on the other hand the finishing day of the last DT.

Input: A day d and a permutation π of set G_d of DTs

1 $C_r \leftarrow W_{r,d}^{\text{start}} \qquad \forall r \in R,\ d \in D_r^{\text{res}}$;
2 **for** $(t,u) \leftarrow \pi(1), \ldots, \pi(|G_d|)$ **do**
3 $\quad S_{t,u} \leftarrow \max_{r \in Q_{t,u}}(C_r - P_{t,u,r}^{\text{start}})$;
4 $\quad$ **while** $\exists r \in Q_{t,u} \wedge \exists \overline{W}_{r,d,w} \in \overline{W}_{r,d} : [S_{t,u} + P_{t,u,r}^{\text{start}}, S_{t,u} + P_{t,u,r}^{\text{end}}) \cap \overline{W}_{r,d,w} \neq \emptyset$ **do**
5 $\quad\quad S_{t,u} := \overline{W}_{r,d,w}^{\text{end}} - P_{t,u,r}^{\text{start}}$;
6 $\quad$ **end**
7 $\quad C_r \leftarrow S_{t,u} + P_{t,u,r}^{\text{end}} \qquad \forall r \in Q_{t,u}$;
8 **end**

Algorithm 1: Time assignment of a given permutation of DTs.

In the time assignment phase the scheduling of the DTs is done for each working day independently. For a particular working day always the DT with the highest priority is planned as early as possible until all DTs have been considered. The priority of the DTs is determined by a lexicographic combination of three criteria that consider the idle time that emerges on the beam resource, the earliest end of a regular service window from a required resource, and the ratio between the time the beam is required and the total processing time of the respective DT.

3.2 Local Search

The design of the neighborhood used within the IG's local search component depends on several factors. As real world instances are expected to be quite large, the main challenge is to find neighborhoods that can be searched rather fast, still allowing to complete a reasonable number of iterations of the IG, while improving the solution significantly in most cases. To achieve this, we restrict ourselves to a local search method that is only able to modify the starting times of DTs, i.e., the day assignment is considered to be fixed. Hence, we are only able to improve on the objective function term that considers the use of extended service windows. A further consequence of this restriction is that the working days become independent, which allows us to define the neighborhood and perform the local search for each day in a separate fashion.

In our scenario the DTs are heterogeneous regarding their time and resource requirements. Thus, moving DTs or exchanging the starting times of two DTs in a tightly scheduled day will lead in most cases to an infeasible solution. However, we can exploit the fact that each DT requires the beam resource exactly once to define a unique sequence of the DTs scheduled on a particular day. Let $G_d = \{(t,u) \mid t \in T, u \in U_t, Z_{t,u} = d\}$ be the set of DTs assigned to day $d \in D'$. We use as solution encoding the permutation π of G_d that is defined by sorting the DTs $(t,u) \in G_d$ in ascending order of the times from which on they use the beam B, i.e., according to $S_{t,u} + P_{t,u,\text{B}}^{\text{start}}$. On such permutations we are able to apply classical moves. To evaluate the objective function we have to decode a permutation of DTs to obtain an actual time assignment. Algorithm 1 shows this decoding for a given set of DTs G_d and a corresponding permutation π for a working day $d \in D'$. The procedure starts by initializing the time marker C_r to the earliest time a resource r becomes available. In the main loop each DT in G_d is assigned in the order of π to the earliest possible start time at which all resources are available. First, at Line 3 the start time $S_{t,u}$ is set to the earliest time at which no required resource is used before the corresponding time marker. At this time, the considered DT might still overlap with unavailability periods. If this is the case, the DT is delayed in the inner while loop until all required resources become available. At Line 7 the C_r time markers are set to the times when the corresponding resources become free after the just scheduled DT.

The DT exchange neighborhood is defined for a day d on a permutation π of DTs by considering all pairs of DTs $\pi(i)$ and $\pi(j)$, where $i, j \in \{1, \ldots, |G_d|\}$ and $i < j$. A move in this neighborhood results in a new permutation $\pi(1), \ldots, \pi(i-1), \pi(j), \pi(i+1), \ldots, \pi(j-1), \pi(i), \pi(j+1), \ldots, \pi(|G_d|)$ and is accepted if the decoded time assignment has a better objective value.

To accelerate the local search procedure we restrict the DT exchange neighborhood to the most

Input: A solution (Z, S)
1 select a set T' of $\beta^{\text{ig-dest}} \cdot n_T$ random therapies and remove their day and time assignments;
2 apply TWCH's day assignment for the set of removed therapies;
3 **foreach** $d \in D$ **do**
4 $\quad G_d \leftarrow \{(t,u) \mid t \in T \setminus T', u \in U_t, Z_{t,u} = d\}$;
5 $\quad G'_d \leftarrow \{(t,u) \mid t \in T', u \in U_t, Z_{t,u} = d\}$;
6 $\quad$ **foreach** $(t,u) \in G'_d$ **do**
7 $\quad\quad$ let π be the permutation of G_d resulting by sorting the elements according to $S_{t,u} + P^{\text{start}}_{t,u,\text{B}}$;
8 $\quad\quad$ best_obj $\leftarrow \infty$; best_MS $\leftarrow \infty$; $\pi'_{\text{best}} \leftarrow ()$;
9 $\quad\quad$ **for** $i \leftarrow 1$ **to** $|G_d| + 1$ **do**
10 $\quad\quad\quad$ $\pi' \leftarrow (\pi(1), \ldots, \pi(i-1), (t,u), \pi(i), \ldots, \pi(|G_d|))$;
11 $\quad\quad\quad$ schedule π' with Algorithm 1;
12 $\quad\quad\quad$ obj $\leftarrow$ objective value of the current partial solution;
13 $\quad\quad\quad$ MS $\leftarrow$ makespan of current day d;
14 $\quad\quad\quad$ **if** obj $<$ best_obj $\vee$ (obj $=$ best_obj $\wedge$ MS $<$ best_MS) **then**
15 $\quad\quad\quad\quad$ best_obj $\leftarrow$ obj; best_MS $\leftarrow$ MS; $\pi'_{\text{best}} \leftarrow \pi'$;
16 $\quad\quad\quad$ **end**
17 $\quad\quad$ **end**
18 $\quad\quad$ schedule π'_{best} with Algorithm 1;
19 $\quad\quad$ $G_d \leftarrow G_d \cup \{(t,u)\}$;
20 $\quad$ **end**
21 **end**

Algorithm 2: Destruction and construction phases.

promising moves. That is, a move is only evaluated if both considered DTs are either adjacent in sequence π or it is likely that an exchange produces a tighter scheduled day. The latter criterion is based on the observation that scheduling two DTs requiring the same treatment room consecutively induces substantial idle time on the beam resource. Hence, we count how many adjacent DTs of $\pi(i)$ and $\pi(j)$ are requiring the same room as $\pi(i)$ and $\pi(j)$, respectively. A move is only considered further if this number does not increase with the exchange.

3.3 Destruction and Construction

The destruction and construction phase of the IG from Maschler et al. [8] consists of removing the DTs of randomly selected therapies from the schedule, followed by applying TWCH's day assignment for the removed therapies and solving TWCH's time assignment from scratch. However, w.r.t. the local search algorithm from Section 3.2 discarding the whole time assignment during destruction and construction is disadvantageous since no parts of the old time assignment of an affected day are transferred to the new one. We overcome this drawback by replacing TWCH's day assignment with an insertion heuristic which preserves the sequence of unchanged DTs and inserts the removed ones in a greedy fashion. Note that this insertion heuristic is conceptually similar to the NEH algorithm of Nawaz et al. [9].

Algorithm 2 shows the used destruction and construction phase in detail. It starts by invalidating the day and time assignment of $\beta^{\text{ig-dest}} \cdot n_T$ randomly selected therapies, where $\beta^{\text{ig-dest}} \in (0, 1]$ is the destruction rate. Afterwards, TWCH's day assignment is applied to reassign the DTs from the removed therapies to potentially new days. The insertion heuristic for the time assignment is defined in the foreach loop at Line 3 and is applied for each working day. The heuristic starts by initializing G_d to the set of DTs that have been assigned to day d and which have not been removed by the destruction phase. Analogously, G'_d is defined as the set containing all DTs assigned to day d that have been removed and for which a new starting time has to be found. Since all DTs in G_d have valid start times, we can define a unique permutation π by sorting the DTs according to the time they first require the beam resource. In each step a not yet considered random DT from G'_d is inserted at all possible positions of the permutation

π and scheduled using Algorithm 1. All of these $|G_d| + 1$ partial time assignments are compared and finally the best one is kept. To this end a permutation is considered better if the objective value is smaller (i.e., the permutation uses less extended time). In case of a tie we prefer the option with the smaller makespan. The rationale behind the latter criterion is that in particular after destruction many insertion points allow scheduling the sequence without use of extended service windows. Preferring a smaller makespan typically results in a tighter packed schedule and hopefully retains better options for the still to be inserted DTs.

4 Computational Study

We perform in this section an experimental evaluation and comparison of the proposed enhanced IG approach, which we call from here on EIG, with the IG from Maschler et al. [8], denoted as IG-LI, and IG-LS, the variant that uses the destruction and construction phase from [8] combined with the local search method from Section 3.2, on a set of new benchmark instances. Note that there is an additional variant in which the used local search method of EIG is replaced with the local improvement operator from [8]. We exclude this variant from further considerations because the local improvement operator ignores the time assignment provided by the construction phase and, hence, it is in practice equivalent to IG-LI.

The used artificial benchmark instances are related to the expected situation at MedAustron and are available at `http://www.ac.tuwien.ac.at/research/problem-instances`. The instances' main characteristic is the number of therapies n_T. From this number we derive the length of the planning horizon and generate therapies that have to start in a window of 14 days. The instances are designed in such a way that after a ramp-up phase of a few weeks the facility is used at full capacity followed by a wind-down phase near the end of the planning horizon. For more details on the instance generation see [8]. We consider instances with 50, 70, 100, 150, 200, and 300 therapies. The used naming schema encodes first the number of therapies followed by a consecutive number. Note that we generated new instances for the experiments discussed here for two reasons. On the one hand, [8] considered DTs composed of activities associated with minimum and maximum time lags. However, as already mentioned this feature is not really considered relevant in our real world midterm planning application. On the other hand, many of the instances from [8] had rather unrealistically strict constraints concerning the starting days of therapies, which made an extensive amount of extended time unavoidable.

All algorithms have been implemented in C++11 and compiled with G++ 4.8.4, and all experiments were carried out using a single core of an Intel Xeon E5540 processor with 2.53 GHz. In a preliminary study we observed for the local search method that a next improvement strategy converges, in general, significantly faster than a best improvement strategy, while yielding similarly good solutions. Moreover, we tested the impact of randomizing the order in which the local search examines the neighboring solutions. It turned out that this randomization yields small improvements on almost all instances that are, however, still in the magnitude of the standard deviation. From a theoretical point of view, the randomization of the order of the considered moves removes a bias towards exchanges at the beginning of days. Therefore, we applied this randomization in the following experiments. Moreover, empirical investigations have shown that the restrictions of the neighborhood exclude promising moves only in rare cases. We adopt the acceptance criterion and the termination condition from [8]: The incumbent solution is replaced by a current new solution iff the latter has a smaller objective value, and the total CPU-time is limited to 20 minutes, respectively.

The metaheuristics' strategy parameters were tuned using the automatic parameter configuration tool irace [7] in version 2.1. In detail, irace was applied separately on each instance size to tune $\beta^{\text{ig-dest}}$ for EIG and IG-LS and the parameters $\beta^{\text{ig-dest}}$, $n^{\text{rta-noimp}}$, and $k^{\text{rta-rand}}$ for IG-LI. On this account, we generated for each instance size five independent instances for tuning. Moreover, each irace run had a computational budget of 1000 experiments. The resulting parameter configurations are shown in Table 1.

Table 2 depicts for IG-LI, IG-LS, and EIG averages of the final objective values *obj* and the corresponding standard deviation $\sigma(obj)$ over 30 runs for each of the 30 benchmark instances. We start

Instance size	IG-LI			IG-LS	EIG
n_T	$\beta^{\text{ig-dest}}$	$n^{\text{rta-noimp}}$	$k^{\text{rta-rand}}$	$\beta^{\text{ig-dest}}$	$\beta^{\text{ig-dest}}$
50	0.2	$\lfloor 2.25 \cdot \lvert G_d \rvert \rfloor$	2	0.092	0.068
70	0.08	$\lfloor 1.50 \cdot \lvert G_d \rvert \rfloor$	2	0.15	0.05
100	0.055	$\lfloor 1.86 \cdot \lvert G_d \rvert \rfloor$	2	0.06	0.039
150	0.08	$\lfloor 1.70 \cdot \lvert G_d \rvert \rfloor$	2	0.106	0.026
200	0.069	$\lfloor 1.34 \cdot \lvert G_d \rvert \rfloor$	2	0.056	0.022
300	0.149	$\lfloor 2.43 \cdot \lvert G_d \rvert \rfloor$	2	0.189	0.023

Table 1: Parameter settings for IG-LI, IG-LS, and EIG determined by irace.

Instance	IG-LI		IG-LS		EIG	
	obj	$\sigma(obj)$	*obj*	$\sigma(obj)$	*obj*	$\sigma(obj)$
050-01	7.512	1.179	8.945	1.953	**4.432**	1.338
050-02	43.618	2.297	50.768	4.082	**40.609**	4.003
050-03	47.355	3.373	49.522	6.438	**40.100**	4.523
050-04	22.263	1.520	**21.224**	1.828	22.536	3.136
050-05	46.567	2.978	**45.528**	2.631	50.105	5.589
070-01	26.563	3.814	33.289	2.998	**19.442**	3.187
070-02	38.545	3.958	40.184	3.390	**36.304**	5.476
070-03	**58.569**	3.720	63.721	2.517	66.849	3.316
070-04	10.615	2.566	12.764	2.655	**9.144**	2.042
070-05	44.101	2.815	46.606	2.236	**42.100**	4.075
100-01	20.210	1.477	17.861	1.328	**10.840**	0.858
100-02	36.819	3.368	30.992	3.035	**14.282**	1.993
100-03	12.045	1.321	10.363	1.039	**8.143**	0.376
100-04	21.236	2.374	19.848	2.308	**12.512**	1.333
100-05	33.870	2.126	27.253	1.924	**11.746**	1.610
150-01	25.287	2.038	27.731	2.306	**18.188**	1.662
150-02	102.486	3.085	77.613	4.166	**46.879**	3.156
150-03	36.246	2.972	41.840	3.398	**29.723**	2.955
150-04	25.392	2.803	26.329	2.994	**17.656**	1.864
150-05	34.176	2.026	34.139	2.752	**22.096**	3.504
200-01	62.931	4.031	54.072	5.134	**46.174**	4.046
200-02	62.335	4.685	60.310	3.283	**34.189**	2.393
200-03	35.798	3.109	36.137	3.163	**28.734**	2.829
200-04	51.805	3.876	49.700	3.302	**28.990**	2.581
200-05	40.434	2.669	40.231	3.207	**28.621**	2.161
300-01	23.368	1.905	26.646	2.240	**23.312**	2.965
300-02	93.680	3.613	99.561	4.988	**56.730**	4.381
300-03	**36.127**	3.920	40.530	3.472	39.223	4.111
300-04	106.270	4.870	114.920	5.685	**103.399**	6.813
300-05	20.208	1.460	21.607	1.319	**17.484**	1.750

Table 2: Average objective values *obj* of 30 runs and corresponding standard deviations $\sigma(obj)$ for IG-LI, IG-LS, and EIG.

by comparing IG-LI with IG-LS. The average objective values of IG-LI are on 17 benchmark instances smaller compared to IG-LS. On most instances the absolute differences between the objective values are still in the range of the standard deviation. For this reason we applied a Wilcoxon rank sum test with a significance level of 95% for each instance. It turned out that IG-LI performed significantly better compared to IG-LS on all instances with 70 and 300 therapies and on two instances with 50 and 150 therapies, respectively. IG-LS has significantly better results on all instances with 100 therapies, on one with 50 and 150 therapies, respectively, and on two instances with 200 therapies. Thus, although IG-LI shows better results than IG-LS on slightly more instances, no significant overall performance difference can be observed among these two approaches. This indicates that exchanging just the local improvement operator of the IG-LI with the local search component described in Section 3.2 does not yield a substantial improvement.

Most importantly, however, Table 2 clearly shows that EIG dominates the two other metaheuristics, and provides the best average objective values on 26 out of 30 benchmark instances. According to a Wilcoxon rank sum tests with a significance level of 95%, EIG is better than IG-LI on 25 instances and better than IG-LS on 26 instances. Moreover, on 16 instances the average objective values of EIG are 25% smaller compared to those of IG-LI, and on three instances the average objective values are halved compared to the ones from IG-LI. The main reason for this performance improvement is the interplay between EIG's construction phase and local search procedure. On the one hand, the local search operator is, in general, able to provide better results than IG-LI's local improvement operator. However, encoding, decoding, and evaluating the solution is computationally demanding and, hence, converging to a local optimum is time consuming, especially on strongly perturbed solutions. On the other hand, EIG's construction phase is designed in such a way that large parts of the sequence of the DTs are preserved while introducing the removed DTs in a sensible but randomized way. Starting with a solution close to a local optimum w.r.t. the DT exchange neighborhood allows to reduce the time spent in the local search procedure and, consequently, increases the total number of iterations.

The impact of EIG's construction phase is also implicitly reflected in the parameter configurations determined by irace. IG-LS' local search starts each iteration with a new time assignment produced by TWCH's time assignment from scratch. Due to the many steps required to reach a locally optimal solution it makes sense to use a rater high destruction rate to cover a larger part of the search space within the time limit. With the new destruction and construction phase, however, the situation changes, because now there is an immediate correspondence between the destruction rate and the time required to reach a locally optimal solution and with it the number of total iterations. Here, the destruction is able to either focus or broaden the exploration of the search space.

5 Conclusion

In this paper, we presented an IG metaheuristic for the PTPSP that enhances the IG from our previous work [8]. In contrast to the latter, the presented approach aims at preserving the order of the not removed DTs on the individual days. The resulting advantage is that more information from the incumbent solution is maintained. Compared to our previous IG and a variant of the previous IG that uses the presented local search procedure our new IG metaheuristic provides significantly better results on 25 and 26 out of 30 benchmark instances, respectively. The superiority of the enhanced approach over the other two can be explained by the interplay between its construction phase and the applied local search technique: The local search method yields, in general, better results than applying TWCH's time assignment randomized for many times. However, due to the required encoding and decoding steps, evaluating neighbors is time consuming. Hence, to ensure that the metaheuristic is able to perform sufficiently many iterations it is required that the neighborhood requires on average only a few steps until it reaches a local optimum. To this end, we apply in the construction phase an insertion heuristic that iteratively places the removed DTs into the permutation resulting from sorting the DTs according to the times at which they use the beam.

PTPSP, as described here, is only a simplified variant of the midterm planning part arising in practice. In particular, therapies are restricted here to consist only of DTs. In the general case, however, there is a

treatment planning phase preceding all DTs in which the DTs are prepared. Since other constraints have to be enforced for these tasks, they cannot be modeled as DTs. Moreover, there are control examinations that have to be provided once a week before or after one of the DTs. In addition, the DTs of a therapy should be planned roughly at the same time on each treatment day to provide a consistent experience for the patients. This soft constraint has several implications: The working days considered during the time assignment are not independent anymore. Furthermore, the strategy applied here to schedule the DTs as early as possible might lead to suboptimal solutions. The reason for this is that delaying a DT can improve the objective value, especially if there is still unused time in the regular service windows left.

We use in this work an acceptance criterion that accepts a current solution if it has a better objective value. An obvious next step is to also consider acceptance criteria that allow to select suboptimal solutions, e.g., in an simulated annealing like fashion (see [15]). Moreover, the proposed local search procedure exchanges the DTs only within days. Neighborhoods that can alter the days on which a therapy is applied seem promising. The main challenge here is to design and restrict the neighborhoods s.t. the resulting local search method is still fast enough to allow a sufficiently large number of IG iterations.

References

[1] E. K. Burke, P. L. Rocha, and S. Petrovic. An integer linear programming model for the radiotherapy treatment scheduling problem. *CoRR*, abs/1103.3391, 2011.

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

[3] S. Hartmann and D. Briskorn. A survey of variants and extensions of the resource-constrained project scheduling problem. *European Journal of Operational Research*, 207(1):1 – 14, 2010.

[4] L. W. Jacobs and M. J. Brusco. A local-search heuristic for large set-covering problems. *Naval Research Logistics*, 42(7):1129–1140, 1995.

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

[6] T. Kapamara, K. Sheibani, O. Haas, D. Petrovic, and C. Reeves. A review of scheduling problems in radiotherapy. In *Proceedings of the International Control Systems Engineering Conference (ICSE)*, pages 207–211, Coventry, UK, 2006. Coventry University Publishing.

[7] M. López-Ibáñez, J. Dubois-Lacoste, L. P. Cáceres, M. Birattari, and T. Stützle. The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3:43–58, 2016.

[8] J. Maschler, M. Riedler, M. Stock, and G. R. Raidl. Particle therapy patient scheduling: First heuristic approaches. In *Proceedings of the 11th International Conference on the Practice and Theory of Automated Timetabling*, pages 223–244, Udine, Italy, 2016.

[9] M. Nawaz, E. E. Enscore, and I. Ham. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega*, 11(1):91–95, 1983.

[10] D. Petrovic, M. Morshed, and S. Petrovic. Genetic algorithm based scheduling of radiotherapy treatments for cancer patients. *Proceedings of the Conference on Artificial Intelligence in Medicine (AIME)*, 5651:101–105, 2009.

[11] D. Petrovic, M. Morshed, and S. Petrovic. Multi-objective genetic algorithms for scheduling of radiotherapy treatments for categorised cancer patients. *Expert Systems with Applications*, 38(6):6994–7002, 2011.

[12] S. Petrovic and P. Leite-Rocha. Constructive and GRASP approaches to radiotherapy treatment scheduling. In *Advances in Electrical and Electronics Engineering - IAENG Special Edition of the World Congress on Engineering and Computer Science 2008*, pages 192–200. IEEE, 2008.

[13] 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, Nottingham, UK, 2006.

[14] M.-C. Riff, J. P. Cares, and B. Neveu. Rason: A new approach to the scheduling radiotherapy problem that considers the current waiting times. *Expert Systems with Applications*, 64:287–295, 2016.

[15] R. Ruiz and T. Stützle. A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. *European Journal of Operational Research*, 177(3):2033–2049, 2007.