Multi-criteria Curriculum-Based Course Timetabling—A Comparison of a Weighted Sum and a Reference Point Based Approach

Martin Josef Geiger

University of Southern Denmark
Department of Business & Economics
Campusvey 55, DK-5230 Odense M, Denmark
mjg@sam.sdu.dk

Abstract. The article presents a solution approach for a curriculum-based timetabling problem, a complex planning problem found in many universities.

With regard to the true nature of the problem, we treat it as multi-objective optimization problem, trying to balance several aspects that simultaneous have to be taken into consideration. A solution framework based on local search heuristics is presented, allowing the planner to identify compromise solutions. Two different aggregation techniques are used and studied. First, a weighted sum aggregation, and second, a reference point based approach.

Experimental investigations are carried out for benchmark instances taken from track 3 of the *International Timetabling Competition ITC 2007*. After having been invited to the finals of the competition, held in August 2008 in Montréal, and thus ranking among the best five approaches world-wide, we here extend our work towards the use of reference points.

Keywords: Multi-criteria timetabling, iterated local search, threshold accepting, reference point approach.

1 Introduction

Timetabling describes a variety of notoriously difficult optimization problems with a considerable practical impact. Important areas within this context include employee timetabling, sports timetabling, flight scheduling, and timetabling in universities and other institutions of (often higher) education [1].

Generally, timetabling is concerned with the assignment of activities to resources. In more detail, these resources provide timeslots (time intervals) to which the activities have to be assigned. In contrast to classical scheduling problems [2], the available time is therefore already discretized into these slots. The result of timetabling is the construction of a *timetable* which defines for each activity when it has to be executed using which resource.

In the case of educational timetabling problems, events are either lectures or examinations, and resources are lecturers and rooms in which the classes are held.

Obviously, the construction of such a timetable has to be done with respect to problem-specific side constraints. Prominent examples comprise:

- All activities must be assigned to timeslots.
- A timeslot may be assigned to at most a single activity.
- Some resources may be unavailable during certain timeslots.
- Individual requirements of the certain activities: Not all resources may be equally suitable for all events.

On the other hand, timetables must be designed such that they optimize several, often conflicting criteria and address the requirements of different groups of stakeholder. This means in case of educational timetabling problems, that both the needs of the students as well as those of the members of staff should be respected.

It is interesting to see that most scientific investigations of educational timetabling implicitly consider the problems to be of multi-objective nature [3, 4]. In contrast to the established terminology from multi-criteria decision making however, desired properties of solutions usually are not expressed using *criteria* nor measured involving a set of *objective functions*. In timetabling, the common approach is to introduce so-called ‘soft constraints’, each of which computes a score with respect to a particular aspect of the problem. While a violation of these soft constraint is possible, it is penalized introducing a penalty (cost) function. The overall objective may then be derived by minimizing an aggregated overall cost function.

Examples of criteria/ soft constraints include:

- As several lectures share students, they should not be assigned to timeslots of the same period.
- Precedence relations between certain activities: Some activities should be scheduled before or after others.
- Preference of lecturers for certain timeslots or rooms.
- As students have to commute from one room to the other, lectures sharing some students should be placed in rooms which are close to each other.
- Consideration of (meal) breaks.
- Other patterns: In order to ensure a certain compactness of the timetables, any lecture should be for any student adjacent to another lecture.

Most variants of timetabling problems unfortunately are $\mathcal{NP}$-hard [5], and moreover, most problem instances comprise a rather large number of decision variables. Consequently, the application of exact optimization approaches is problematic, given the fact that the solution has to be done within limited time. While there are some exact optimization approaches on the basis of integer linear programming [6], the vast amount of problem solution techniques are based on heuristics, and more recently, metaheuristics.

Heuristic approaches can be categorized with respect to how the two classes of side constraints, hard constraints and soft constraints, are treated. Three classes are commonly considered [7]:

1. *One-stage approaches.*
   One-stage approaches combine the penalty functions of hard and soft constraint violations into a single evaluation function. As generally the minimization of hard constraints is considered to be more important in comparison to the minimization of the violation of soft constraints, a considerable higher weight is given to this aspect.
2. *Two-stage approaches.*
   Contrary to one-stage approaches, two-stage algorithms divide the search for an optimal solution into two disjunct phases. In a first step, a feasible assignment of all events to timeslots is computed, and feasibility is here understood with respect to the set of hard constraints only. The succeeding phase of two-stage approaches is devoted to the minimization of the soft constraint penalties while maintaining feasibility.
3. *Relaxation-based approaches.*
   Feasibility of timetables in relaxation-based solution approaches is assured by either relaxing certain side constraints such that all events may be assigned to timeslots, or by leaving some events unassigned. In this sense, the obtained timetables are feasible not for the initial but with respect to some modified side constraints. When optimizing the timetables for the soft constrain penalties, it is then tried to accommodate all initially defined hard constraints into the solution with the ultimate goal of reaching feasibility. Left aside events are put into the timetable, etc.

Independent from the implemented strategy, most modern approaches are based on the principles of local search. The particular characteristics of timetabling problems imply that neighborhoods are generally composed of moves that unassign and reassign some events, and thus relocate particular activities in the current solution.

An extensive comparison of algorithms has been done in the first International Timetabling Competition ITC 2002 (`http://www.idsia.ch/Files/ttcomp2002/`). Metaheuristics that turned out to be especially effective for timetabling problems are Simulated Annealing, deterministic variants of Simulated Annealing, and Tabu Search. As for most operations research problems, other techniques have been used, too, including Evolutionary Algorithms, Ant Colony Optimization, and Greedy Randomized Adaptive Search. For an extensive listing of references we may refer to [7].

While previous research has primarily been considering an additive aggregation of the penalty functions, our aim is to extend this work towards other ideas from multi-criteria decision making. In detail, we study the influence of the aggregation methodology on the performance and the outcomes of the solution approach. A solution framework (also) allowing the integration of reference points is therefore presented in Section 3 and tested on benchmark data from the ITC 2007.

2 The Curriculum-Based Course Timetabling Problem

The curriculum-based course timetabling problem [8] is a particular variant of an educational timetabling problem. It consists of constructing a weekly timetable by assigning lectures for several university courses to a given number of rooms and time periods. The sketched situation can be found in many universities, where so-called *curricula* are used to describe sets of courses/ lectures that share common students. The underlying logic is based on the assumption, that students who are enrolled in the same program progress together through their studies. Consequently, these students are supposed to attend a previously well-defined set of lectures. This can be seen contrary to post-enrollment based course timetabling problems, where students explicitly register for courses they wish to attend. While in this setting, detailed information about any particular student can be obtained, in curriculum-based course timetabling, registrations of students for courses are not required. The available constraints are solely based on the definition of the curricula.

The data for the curriculum-based course timetabling problem is comparably easy to obtain. Once the curricula are defined, they usually do not change very often, and timetabling can consider to some extend last years plans. On the other hand, students not following the definitions of their curricula may end up having a problem, as several lectures will be scheduled in parallel. The application of the curriculum-based course timetabling problem therefore requires the students to follow the structure of their program.

Besides its practical relevance for many universities, the curriculum-based course timetabling problem has been chosen as one of the competition tracks of the ITC 2007 (`http://www.cs.qub.ac.uk/itc2007/`). The competition invited researchers to propose and submit novel approaches for the solution of timetabling problems. Comparison of results is possible by means of a set of newly released benchmark instances.

A technical description of the problem of the ITC 2007 is given in [8]. Important hard constraints require that no lectures belonging to a common curriculum, as well as lectures being taught by the same professor, are scheduled in parallel. Also, a set of given unavailability constraints has to be respected. These constraints define times when teachers are unavailable. Another common constraint requires that at most one lecture can be assigned to a single room at a time.

Besides these hard constraints, four soft constraints/ objectives sc_1, sc_2, sc_3, sc_4 are relevant that measure desirable properties of the solutions.

1. Objective 1 (sc_1): A room capacity soft constraint tries to ensure that the number of students attending a lecture does not exceed the room capacity. Assignments of lectures to rooms of smaller capacity are penalized with the number of students above the room capacity, multiplied with a penalty weight w_1.
2. Objective 2 (sc_2): The lectures of the courses must be spread into a minimum number of days, penalizing timetables in which lectures appear on too few distinct days. Each day below the minimum is penalized with w_2 points of penalty.
3. Objective 3 (sc_3): The curricula should be compact, meaning that isolated lectures, that is lectures without another adjacent lecture, should be avoided.

For any given curriculum, the number of isolated lectures is computed. Each isolated lecture in a curriculum is penalized with w_3 points of penalty.

4. Objective 4 (sc_4): All lectures of a course should be held in exactly one room. Each distinct room used for the lectures of a course, but the first, counts as w_4 points of penalty.

The overall evaluation of a timetable is then based on some aggregate function $SC = f(sc_1, sc_2, sc_3, sc_4)$.

3 Proposition of a Solution Approach

Figure 1 illustrates the elements of the solution framework in which the following entities play a role:

– A *human decision maker* communicates via a *graphical user interface (GUI)* with the system. Communication includes the definition of decision variables, constraints, objectives and preferences. Also, the penalization (weighting) of particular timetabling patterns and the aggregation of the objective functions are obtained in this process.
– The formal *model* of the current situation is formulated and stored in a database.
– After a *preprocessing stage*, in which some problem-specific properties are pre-computed and structured, two method bases are employed to construct and improve timetables for the quantitative model:
    1. A *constructive approach* is used to obtain a first feasible assignment of all lectures. In this phase, the chosen objective functions are not yet considered, but the method focuses on the hard constraints of the problem.

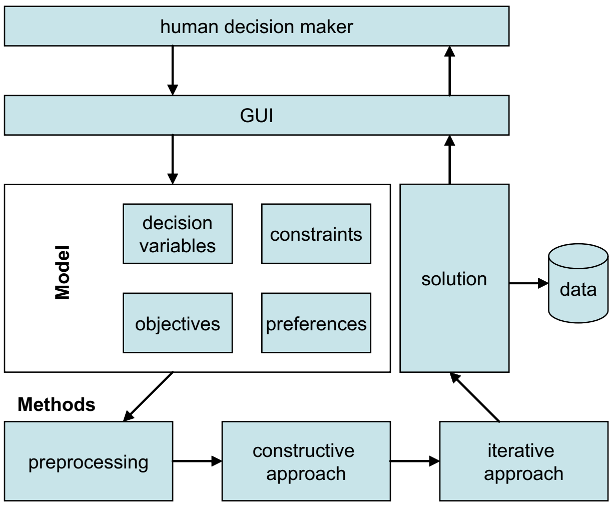

Fig. 1. Elements of the solution concept and their interaction

2. An *iterative approach* is then subsequently executed to improve the timetable obtained from the constructive approach. During this phase, reallocations of lectures are carried out with the final goal of identifying an optimal timetable.

Constructive and iterative phases follow subsequently. In this sense, the solution framework implements a two-phase-concept in which feasibility of the timetable is first considered, and optimality following later.

3.1 Preprocessing

Prior to the computation of a first solution, some preprocessing is carried out. In brief, some problem-specific characteristics are employed, adding some additional structure to the problem.

For each given lecture L_i, events $E_{i1}, \dots, E_{ie}$ are created which are later assigned to timeslots. The number of events e is given in the problem instances. Creating events for each lecture leads to a more general problem description, and the solution approach only needs to concentrate on the assignment of all events, one to a single timeslot, as opposed to keeping track of assigning a lecture to e timeslots.

Second, we categorize for each lecture L_i (and thus for each event belonging to lecture L_i) the available rooms in three disjunct classes $\mathcal{R}_{i1}, \mathcal{R}_{i2}, \mathcal{R}_{i3}$.

$\mathcal{R}_{i1}$ refers to the rooms in which the lecture fits best, that is the rooms R_k with the minimum positive or zero value of $c_k - s_i$, c_k being the room capacity, s_i the number of students of lecture L_i. The class $\mathcal{R}_{i2}$ stores the rooms in which lecture L_i fits, that is $s_i < c_k$, but not best, and $\mathcal{R}_{i3}$ contains the rooms in which lecture L_i does not fit. With respect to the given problem statement, events of lectures may be assigned to timeslots of rooms in $\mathcal{R}_{i3}$, this however results in a penalty.

The underlying assumption of the classification of the rooms is that events are preferably assigned to timeslots belonging to a room of class $\mathcal{R}_{i1}$, followed by $\mathcal{R}_{i2}$ and $\mathcal{R}_{i3}$. It has to be mentioned however, that this cannot be understood as a binding, general rule but rather should be seen as a recommendation. A randomized procedure is therefore going to be implemented when assigning events to timeslots (see Section 3.3), allowing a certain deviation from the computed room order.

3.2 Constructive Approach

Based on the results for the initial constructive approach, we propose a reactive procedure that self-adapts to the set of unassigned events from previous runs. The logic behind this approach is that the constructive procedure ‘discovers’ events that are difficult to assign, giving them priority in successive runs. To some extend, we borrow ideas from the *squeaky wheel optimization* approach [9]. In this concept, alternatives are repetitively constructed and analyzed. Unfavorable aspects of the current solution are discovered in each analysis, which are then resolved in the successive iteration. In this sense, the ‘squeaky wheel gets the grease'. Events that have not been assigned in previous construction runs are here considered to be the unfavorable aspects of the current solution.

In the following, let $\mathcal{E}^p$ be the set of prioritized events, $\mathcal{E}^{\neg p}$ the set of non-prioritized events, and $\mathcal{E}^u$ the set of events that have not been assigned during the construction phase. It is required that $\mathcal{E}^p \subseteq \mathcal{E}$, $\mathcal{E}^{\neg p} \subseteq \mathcal{E}$, $\mathcal{E}^p \cap \mathcal{E}^{\neg p} = \emptyset$, and $\mathcal{E}^p \cup \mathcal{E}^{\neg p} = \mathcal{E}$.

Algorithm 1 describes the reactive construction procedure.

Algorithm 1. Reactive construction

```
Require: $Maxloops$
 1: Set $\mathcal{E}^p = \emptyset$, $\mathcal{E}^u = \emptyset$, $loops = 0$
 2: repeat
 3:   $\mathcal{E}^p \leftarrow \mathcal{E}^u$
 4:   $\mathcal{E}^u \leftarrow \emptyset$
 5:   $\mathcal{E}^{\neg p} \leftarrow \mathcal{E} \backslash \mathcal{E}^p$
 6:   while $\mathcal{E}^p \neq \emptyset$ do
 7:     Select the most critical event $E$ from $\mathcal{E}^p$, that is the event with the smallest
        number of available timeslots
 8:     if $E$ can be assigned to at least one timeslot then
 9:       Select some available timeslot $T$ for $E$
10:       Assign $E$ to the timeslot $T$
11:     else
12:       $\mathcal{E}^u \leftarrow \mathcal{E}^u \cup E$
13:     end if
14:     $\mathcal{E}^p \leftarrow \mathcal{E}^p \backslash E$
15:   end while
16:   while $\mathcal{E}^{\neg p} \neq \emptyset$ do
17:     Select the most critical event $E$ from $\mathcal{E}^{\neg p}$, that is the event with the smallest
        number of available timeslots
18:     if $E$ can be assigned to at least one timeslot then
19:       Select some available timeslot $T$ for $E$
20:       Assign $E$ to the timeslot $T$
21:     else
22:       $\mathcal{E}^u \leftarrow \mathcal{E}^u \cup E$
23:     end if
24:     $\mathcal{E}^{\neg p} \leftarrow \mathcal{E}^{\neg p} \backslash E$
25:   end while
26:   $loops \leftarrow loops + 1$
27: until $\mathcal{E}^u = \emptyset$ or $loops = Maxloops$
```

As given in the pseudo-code, the construction of solutions is carried out in a loop until either a feasible solution is identified or a maximum number of iterations $Maxloops$ is reached. When constructing a solution, a set of events $\mathcal{E}^u$ is kept for which no timeslot has been found. When reconstructing a solution, these events are prioritized over the others. In that sense, the constructive approach is biased by its previous runs, identifying events that turn out to be difficult to assign.

After a maximum number of at most $Maxloops$ iterations, the construction procedure returns a solution that is either feasible ($\mathcal{E}^u = \emptyset$) or not ($\mathcal{E}^u \neq \emptyset$).

It becomes clear that the reactive procedure is principally based on the previous simple greedy heuristic. The choice of the most critical event, as well as the choice of the timeslot is left unchanged. Only the prioritization of the events by dividing them into two disjunctive subsets is an additional feature of the revised method.

3.3 Improvement Procedures

An iterative process continues with the alternative found in the constructive approach, searching for an optimal solution with respect to the soft constraint penalties.

In each step of the algorithm, a number of randomly chosen events is unassigned from the timetable and reinserted in the set $\mathcal{E}^u$. A reassignment phase follows. Contrary to the constructive approach, where events are selected based on whether they are critical with respect to the available timeslots, events are now randomly chosen from $\mathcal{E}^u$, each event with equal probability. The choice of the timeslot for the event is based on the logic described in the preprocessing phase, prioritizing timeslots of particular room classes. We use two possible preference structures of rooms, $\mathcal{R}_{i1}$ over $\mathcal{R}_{i2}$ over $\mathcal{R}_{i3}$, and $\mathcal{R}_{i2}$ over $\mathcal{R}_{i1}$ over $\mathcal{R}_{i3}$. Each of them is randomly chosen with probability 0.5.

The following different variants of local search have been implemented and tested:

- Hillclimbing (HC).
  In this local search variant, only improving reassignments of events are accepted. It can be expected that this strategy does not lead to the best results. However, for comparison reasons, an application will be interesting, simply because the effectiveness of alternative local search strategies can be studied in contrast to this relatively simple algorithm.
- Iterated Local Search (ILS).
  Iterated Local Search [10] is based on a hillclimbing algorithm, which is first used to compute a locally optimal solution. Then, after converging to this alternative, an escape mechanism is triggered, consisting of a worsening perturbation move by means of some neighborhood. Search continues from this perturbed alternative, again executing a hillclimbing run.
- Threshold Accepting (TA).
  The idea of Threshold Accepting has been introduced in [11]. It describes a deterministic variant of Simulated Annealing [12]. Worsening moves are accepted up to a certain threshold, thus allowing an escape of the search procedure from local optima. Throughout the execution of the local search approach, the threshold is subsequently decreased, similar to what is referred to as a 'cooling schedule' in Simulated Annealing.

  As previous research has shown that simplifications of Simulated Annealing may be very effective for timetabling problems [13, 14], we suspect that this approach turns out successful for the problem at hand.

4 Experimental Investigation

4.1 Weighted Sum Aggregation

For the first part of our experimental investigation, we consider a weighted sum aggregation of the objective functions sc_i as given in Expression (1). The values of the weights w_i have been chosen as proposed for the ITC 2007, thus $w_1 = 1$, $w_2 = 5$, $w_3 = 2$, $w_4 = 1$.

$$SC = \sum_{i=1}^{4} w_i sc_i \tag{1}$$

Different configurations of the algorithms have been tested on the benchmark data from the ITC 2007. The running time of each test run has been chosen in accordance with the regulations of the competition, allowing 375 seconds of computing time on an Intel Q6600 processor. While experiments with significantly longer running times are reported in [15], we here favor an experimental setting in which only comparably little running time is permitted. This reflects the practical circumstances of many real-world timetabling problems, in which a planner expects results in comparably short time.

The number of reassigned events in each iteration has been set to five for all variants of the improvement procedure. It should be noticed that other numbers from 1 to 10 have been tested, too. Based on some preliminary tests, in which a reassignment set of five events gave reasonable results for all benchmark instances, this number has been chosen and fixed in the following experiments.

Three configurations of the Iterated Local Search approach have been implemented. The first variant, ILS-3k, starts perturbing after 3,000 non-improving moves, the second, ILS-10k, after 10,000 moves, and the last one, ILS-25k, after 25,000 non-improving moves. Perturbations are done by a random reassignment of five events. Contrary to the usual acceptance rule with respect to the cost function SC, the perturbed alternative is accepted in any case, and search continues from this new solution.

Two different configurations of the Threshold Accepting algorithm have been tested. First, an algorithm with a threshold of 1% of the overall evaluation of the alternatives SC. Second, an algorithm using a threshold of 2% of SC. The choice of a percentage of SC as a threshold has the advantage that the algorithm performs an automated cooling when approaching small values of SC, yet maintaining high thresholds for large values of SC. This idea stands somewhat in contrast to other approaches, in which explicit cooling schedules are needed. Nevertheless, an appropriate choice of the percentage has to be made.

Average results of the metaheuristics
The following Table 1 gives the average values of the soft constraint penalties SC for the different local search strategies over ten independent test runs.

The hillclimbing algorithm never leads to best results, which already has been suspected prior to the investigation. For all benchmark instances, either an Iterated Local Search approach and/ or a Threshold Accepting algorithm leads to better average results.

Table 1. Average values for the weighted sum formulation

Instance	HC	ILS-3k	ILS-10k	ILS-25k	TA 1%	TA 2%
comp01	6.3	8.0	7.1	6.8	6.6	6.0
comp02	158.2	157.2	155.5	147.1	135.0	157.2
comp03	151.4	150.7	144.7	147.3	140.6	150.7
comp04	90.0	95.9	92.3	87.1	84.0	79.0
comp05	581.5	494.8	539.1	516.4	505.5	495.9
comp06	137.5	137.7	135.8	130.2	108.4	127.6
comp07	99.1	122.2	113.5	103.8	73.9	121.1
comp08	102.4	98.5	98.4	92.3	83.3	90.6
comp09	169.2	172.5	161.8	164.7	165.7	176.5
comp10	108.3	113.4	106.7	107.5	84.2	82.8
comp11	0.8	2.6	1.7	0.7	0.3	0.1
comp12	534.0	504.8	497.6	499.6	480.3	472.5
comp13	127.9	132.3	122.3	122.1	112.3	122.6
comp14	130.6	113.4	114.6	111.8	101.3	97.6

When analyzing and comparing the different Iterated Local Search approaches, the variant with a rather high number of evaluations before perturbing the solutions, ILS-25k, turns out to be superior to the other configurations. In nine of the 14 data sets, best results are obtained by this approach. Despite the fact that some counterexamples have been found, such as comp05, it is possible to conclude that a sufficient number of evaluations is needed before applying perturbations. In this context, 25,000 describes this 'sufficiently large number' better than 3,000 or 10,000.

Comparing the two Threshold Accepting algorithms, both TA 1% and TA 2% lead to best average result in seven of the 14 data sets. Also it can be seen, that some differences are rather small, such as in case of comp01 and comp11, while others are considerable larger (comp06, comp07, comp08, comp13). On the basis of this observation, we conclude that the comparably smaller threshold of 1% leads to better results than the larger one of 2%.

For twelve out of 14 instances, Threshold Accepting proves to be superior to Iterated Local Search. Some counterexamples exist, but the overall conclusions are rather strong in favor of Threshold Accepting.

Best obtained results
Table 2 shows the best results of the Threshold Accepting algorithm with a threshold of 1%. The results are based on 30 trials with different random seeds.

It can be seen that the TA 1% approach leads to competitive results. For some instances, comp01 and comp11, particularly good solutions are identified. Others such as comp05 and comp12 have best found alternatives with soft constraint penalties that are still quite large. We suspect that some properties of the data sets induce these results. Benchmark instance comp05 possesses a rather small average teacher availability, and so does instance comp12 [8]. It is therefore fair to assume that this property leads to the considerable differences in terms of

Table 2. Best results (out of 30 trials)

Instance	SC	$w_1 sc_1$	$w_2 sc_2$	$w_3 sc_3$	$w_4 sc_4$	Instance	SC	$w_1 sc_1$	$w_2 sc_2$	$w_3 sc_3$	$w_4 sc_4$
`comp01`	5	4	0	0	1	`comp08`	75	0	5	56	14
`comp02`	108	0	5	92	11	`comp09`	153	0	35	92	26
`comp03`	115	0	35	68	12	`comp10`	66	0	0	40	26
`comp04`	67	0	5	48	14	`comp11`	0	0	0	0	0
`comp05`	408	0	175	218	15	`comp12`	430	2	205	196	27
`comp06`	94	0	10	58	26	`comp13`	101	0	25	62	14
`comp07`	56	0	5	18	33	`comp14`	88	0	15	60	13

the best found values of SC to the other data sets. Obviously, relatively difficult side constraints complicate the identification of timetables with a small overall evaluation.

Another aspect of the obtained results is the detailed component-wise analysis of the individual objectives $sc_i, i = 1, \ldots, 4$. Recalling that the assignment of events to timeslots made use of a certain pre-computed order, we suspect that this may influence the characteristics of the results. In particular sc_1, which measures the assignment of lectures to rooms of smaller capacity, should be addressed rather well by this assignment logic.

A detailed overview of the best obtained results SC and the individual objective function values $sc_i, i = 1, \ldots, 4$ is given in Table 2. With the only counterexamples of instances `comp01` and `comp11`, sc_1 turns out to be better addressed in comparison to the other objectives. In particular for sc_2 and sc_3, the numbers are rather relatively high. It also should be noticed, that the scoring for these two objectives employed higher penalties w_i as for the others. Nevertheless, and although a weighted sum approach has been used, a considerable bias in terms of a preference of sc_1 over the other objectives becomes obvious.

4.2 Experiments Based on a Reference Point

As pointed out when analyzing the result in Section 4.1, the weighted sum aggregation approach of the objective functions $sc_i, i = 1, \ldots, 4$ leads to a situation in which objective sc_1 is significantly better addressed in comparison to the others. It can be suspected that these results not always represent the individually desired solutions. Often, the human decision maker requires a different methodology for expressing his/ her preferences, for example by stating a reference point that represents the desired outcomes for each objective function. The solution of the problem then lies in the minimization of the distance of the solution to this point.

Expression (2) reformulates the aggregated evaluation of the timetables by introducing a reference point $R = (r_1, r_2, r_3, r_4)$. By minimizing SC^{ref}, we minimize the distance of the outcomes to the given reference values r_i. Using this approach, the decision maker may primarily guide the search towards a preferred solution by stating desired values of r_i.

$$SC^{ref} = \max_i w_i \left(sc_i - r_i \right) \tag{2}$$

The experiments as described in Section 4.1 have been repeated, now using the revised formulation of the aggregated evaluation function. Again, a hillclimbing algorithm, the three variants of the Iterated Local Search algorithm, and the two Threshold Accepting algorithms have been tested. Experiments have been conducted on the identical computer hardware, once more permitting a maximum running time of 375 seconds for each trial. The reference point has been assumed with $r_i = 0, i = 1, \ldots, 4$ as we know that 0 is the smallest possible outcome for each objective $sc_i, i = 1, \ldots, 4$.

Average results of the metaheuristics
Average results for ten independent test runs are given in the following Table 3.

Table 3. Average results of SC^{ref} in the reference point based formulation

Instance	HC	ILS-3k	ILS-10k	ILS-25k	TA 1%	TA 2%
comp01	4.0	7.0	6.2	5.2	4.0	4.0
comp02	72.1	75.0	66.4	72.1	69.0	66.9
comp03	64.7	68.9	66.4	64.3	64.8	66.3
comp04	41.5	51.3	47.9	47.5	38.2	46.6
comp05	248.2	232.1	236.6	226.5	238.2	226.0
comp06	58.5	71.0	65.7	63.9	60.1	80.2
comp07	50.0	73.5	64.8	61.9	71.0	135.8
comp08	41.5	57.0	50.5	50.2	48.6	59.9
comp09	71.3	79.3	76.5	74.1	73.1	79.5
comp10	44.0	59.6	57.0	53.5	54.8	88.0
comp11	0.1	7.2	5.2	4.8	0.0	0.0
comp12	248.5	240.8	240.1	221.0	228.4	211.0
comp13	55.0	65.0	61.0	60.7	59.2	62.6
comp14	50.5	57.7	54.1	52.0	50.8	52.1

For the Iterated Local Search algorithm, we observe again that the variant with the largest number of evaluations before perturbing the alternative, ILS-25k, obtains better results than the two configurations with a faster perturbation. Also, TA 1% leads in more instances to better results than the one with the larger threshold. In addition to that, Threshold Accepting outperforms Iterated Local Search in most instances. So far, the analysis is in line with the one of the weighted sum approach.

What really is remarkable is the behavior of the hillclimbing approach. In comparison to both the Iterated Local Search as well as the Threshold Accepting algorithm, better results are obtained in many cases. This is counterintuitive, especially as the algorithm does not possess any strategy of escaping local optima while the others do. It appears as if the presence of local optima is less problematic for the chosen evaluation function. In brief, and in contrast to the experiments of Section 4.1 we are able to observe that not a single approach is suitable for all possible data sets and variants of evaluation functions, as it has been already pointed out in other contexts [16].

Best obtained results
Another analysis is concerned with the best found solutions of the reference point based evaluation SC^{ref} and the values of the objectives $sc_i, i = 1, \ldots, 4$. The best solutions are reported in the following Table 4.

Table 4. Best results of SC^{ref} and the values for each objective sc_i (out of 10 trials)

Instance	SC^{ref}	w_1sc_1	w_2sc_2	w_3sc_3	w_4sc_4	Instance	SC^{ref}	w_1sc_1	w_2sc_2	w_3sc_3	w_4sc_4
comp01	4	4	0	2	4	comp08	40	0	40	40	40
comp02	45	0	45	44	45	comp09	66	0	65	66	66
comp03	52	0	50	52	52	comp10	35	0	35	34	35
comp04	30	0	30	30	30	comp11	0	0	0	0	0
comp05	190	190	190	190	41	comp12	200	182	200	198	80
comp06	55	0	55	54	55	comp13	45	0	45	44	44
comp07	44	27	40	44	44	comp14	40	0	40	40	40

Obviously, the values are better balanced in comparison to the ones of the weighted sum aggregation. While sc_1 still often has the smallest value, it is possible to see that for example for instance `comp05`, rather high values of sc_1 are achieved in order to minimize the other objective functions further, and so to minimize the aggregation. A similar observation can be made for the instances `comp07` and `comp12`.

The results indicate that the prioritization of the timeslots when assigning events generally does favor objective sc_1 to some extent. However, there are cases in which the quality of timetables with respect to sc_1 is sacrificed for an improvement in the other objectives, and we may conclude that the solution approach does not generally overfit the investigated problem.

5 Conclusions

The article presented an optimization approach for multi-criteria timetabling problem with an application for curriculum-based course timetabling. On the basis of a general solution framework, a two-stage optimization approach has been proposed that first constructs feasible alternatives and subsequently searches for optimal solutions. For the latter purpose, different local search strategies have been implemented, and experiments have been conducted for benchmark instances taken from the literature.

By comparing a weighted sum and a reference point based approach, considerable differences in the obtained results became obvious. For the weighted sum aggregation, balancing the four objectives appears to be problematic. Even when assigning higher weights on particular objectives, they do not turn out to be better addressed than others. On the other hand, the reference point aggregation led to significantly better balanced outcomes with respect to an equal treatment of all four objectives.

In brief the experiments indicate that an interaction with the presented system employing reference points is more direct and overall preferable, a conclusion that similarly has been derived in [17]. Also, the relative performance of the different metaheuristics is affected by the chosen aggregation procedure. Although the results are overall competitive, as we have been able to demonstrate in the finals of the International Timetabling Competition ITC 2007, future research should therefore be dedicated towards the proposition of more robust solvers.

Besides, a comparison to multi-objective optimization approaches that approximate the whole Pareto-front should be carried out as part of future research. While the approach presented in this article aims to identify an alternative that optimizes the aggregated function only, the entire Pareto-front could be obtained for comparison reasons. Knowing that the weighted sum approach can be problematic, as it only allows the generation of supported efficient solutions, the effect and relevance of this potential drawback could further be explored.

Also, alternative approaches of integrating the reference point should be explored, such as described in e. g. [18]. Such approaches would allow to ensure the efficiency of the final solution, which is not necessarily the case in the implemented min-max achievement function.

References

1. Carter, M.W.: Timetabling. In: Gass, S., Harris, C. (eds.) Encyclopedia of Operations Research and Management Science, 2nd edn., pp. 833–836. Kluwer Academic Publishers, Dordrecht (2001)
2. Pinedo, M.: Scheduling: Theory, Algorithms, and Systems, 2nd edn. Prentice-Hall, Upper Saddle River (2002)
3. Landa-Silva, J.D., Burke, E.K., Petrovic, S.: An introduction to multiobjective metaheuristics for scheduling and timetabling. In: Gandibleux, X., Sevaux, M., Sörensen, K., T'kindt, V. (eds.) Metaheuristics for Multiobjective Optimisation. Lecture Notes in Economics and Mathematical Systems, vol. 535, pp. 91–129. Springer, Heidelberg (2004)
4. Burke, E.K., Petrovic, S.: Recent research directions in automated timetabling. European Journal of Operational Research 140(2), 266–280 (2002)
5. van den Broek, J., Hurkens, C., Woeginger, G.: Timetabling problems at the TU Eindhoven. European Journal of Operational Research (2008) (article in press)
6. Schimmelpfeng, K., Helber, S.: Application of a real-world university-course timetabling model solved by integer programming. OR Spectrum 29(4), 783–803 (2007)
7. Lewis, R.: A survey of metaheuristic-based techniques for university timetabling problems. OR Spectrum 30, 167–190 (2008)
8. Di Gaspero, L., McCollum, B., Schaerf, A.: The second international timetabling competition (ITC 2007): Curriculum-based course timetabling (track 3). Technical Report QUB/IEEE/Tech/ITC2007/CurriculumCTT/v1.0/1 (August 2007)
9. Joslin, D.E., Clements, D.P.: "Squeaky wheel" optimization. Journal of Artificial Intelligence Research 10, 353–373 (1999)
10. Lourenço, H.R., Martin, O., Stützle, T.: Iterated local search. In: Glover, F., Kochenberger, G.A. (eds.) Handbook of Metaheuristics. International Series in Operations Research & Management Science, vol. 57, pp. 321–353. Kluwer Academic Publishers, Dordrecht (2003)

11. Dueck, G., Scheuer, T.: Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing. Journal of Computational Physics 90, 161–175 (1990)
12. Kirkpatrick, S., Gellat, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science 220(4598), 671–680 (1983)
13. Burke, E.K., Bykov, Y., Newall, J.P., Petrovic, S.: A time-predefined approach to course timetabling. Yugoslav Journal of Operations Research (YUJOR) 13(2), 139–151 (2003)
14. Landa-Silva, D., Obit, J.H.: Great deluge with non-linear decay rate for solving course timetabling problems. In: Proceedings of the 2008 IEEE Conference on Intelligent Systems (IS 2008), pp. 8.11–8.18. IEEE Press, Los Alamitos (2008)
15. Geiger, M.J.: An application of the threshold accepting metaheuristic for curriculum based course timetabling. In: Proceedings of the 7th International Conference on the Practice and Theory of Automated Timetabling, Montréal, Canada (August 2008)
16. Wolpert, D.H., Macready, W.G.: No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation 1, 67–82 (1997)
17. Petrovic, S., Bykov, Y.: A multiobjective optimisation technique for exam timetabling based on trajectories. In: Burke, E.K., De Causmaecker, P. (eds.) PATAT 2002. LNCS, vol. 2740, pp. 149–166. Springer, Heidelberg (2003)
18. Wierzbicki, A.P.: The Use of Reference Objectives in Multiobjective Optimization. In: Fandel, G., Gal, T. (eds.) Multiple Criteria Decision Making Theory and Application, pp. 469–486. Springer, Heidelberg (1980)