Solving the Post Enrolment Course Timetabling Problem by Ant Colony Optimization

Alfred Mayer[1], Clemens Nothegger[2], Andreas Chwatal[1], and Günther R. Raidl[1]

[1] Institute of Computer Graphics and Algorithms
Vienna University of Technology, Vienna, Austria

[2] Christian Doppler Laboratory for
Spatial Data from Laser Scanning and Remote Sensing,
Institute of Photogrammetry and Remote Sensing,
Vienna University of Technology, Vienna, Austria

alfred.mayer@student.tuwien.ac.at, cn@ipf.tuwien.ac.at,
{chwatal|raidl}@ads.tuwien.ac.at

Abstract. In this work we present a new approach to tackle the problem of *Post Enrolment Course Timetabling* as specified for the International Timetabling Competition 2007 (ITC2007), competition track 2. The heuristic procedure is based on Ant Colony Optimization (ACO) where artificial ants successively construct solutions based on *pheromones* (stigmergy) and local information. The key feature of our algorithm is the use of two distinct but simplified pheromone matrices (representing event–timeslot and event–room relations, respectively) in order to improve convergence but still provide enough flexibility for effectively guiding the solution construction process. Furthermore a local improvement method is embedded. We applied our algorithm to instances used for the First and Second International Timetabling Competition (TTComp2002 and ITC2007). The results document that our approach is among the leading algorithms for this problem; in many cases the optimal solution could be found.

Keywords: timetabling, ant colony optimization, ant system, metaheuristics, combinatorial optimization;

1 Introduction

Course timetabling problems periodically arise at various universities and other educational institutions. The general course timetabling problem is known to be NP-hard and is also in practice a challenging computational task.

1.1 Problem Description

Here, we focus in particular on the *Post Enrolment Course Timetabling Problem* as specified for the International Timetabling Competition 2007 (ITC2007),

competition track 2. In this problem the challenge is to assign university courses to timeslots and rooms, where each assignment has to fulfill various constraints. According to Lewis et al. (2007) a problem instance for the ITC2007 (track 2) consists of

- a set of n events that are to be scheduled into 45 timeslots (5 days of 9 hours each),
- a set of r rooms, each of which has a specific seating capacity, in which the events take place,
- a set of f room-features that are satisfied by rooms and which are required by events,
- a set of s students who attend various different combinations of events,
- a set of *available* timeslots for each of the n events (i.e. not all events will be available in all timeslots), and
- a set of *precedence* requirements stating that certain events should occur before others.

The goal is to assign the n events to the time slots and rooms such that the following hard constraints are fulfilled (Lewis et al. 2007):

1. No student should be required to attend more than one event at the same time.
2. In each case the room should be large enough for all the attending students and should satisfy all of the features required by the event.
3. Only one event is put into each room in any timeslot.
4. Events should only be assigned to timeslots that are pre-defined as "available" for those events.
5. Where specified, events should be scheduled to occur in the correct order.

It is required that all solutions fulfill all of the hard constraints and leave some events unassigned if necessary. A solution is defined to be *feasible* if all events are assigned without hard constraint violation. Otherwise the quality of the solution is characterized by the *Distance to Feasibility* (DTF) which is the number of students of each of the unplaced events. Besides the hard constraints, solutions should fulfill the following soft constraints:

1. Students should not be scheduled to attend an event in the last timeslot of a day.
2. Students should not have to attend three (or more) events in successive timeslots occurring in the same day.
3. Students should not be required to attend only one event in a particular day.

The *Soft Constraint Penalty* (SCP) is determined as the sum of the following components:

- Count the number of students having just one class a day.

– Count the number of occurrences of a student having more than two classes consecutively (3 consecutively scores 1, 4 consecutively scores 2, 5 consecutively scores 3, etc.).
– Count the number of occurrences of a student having a class in the last timeslot of the day.

Two solutions are first compared by their DTF, and only in the case of equal DTFs the SCP is considered for comparison.

1.2 Previous Work

A comprehensive review of timetabling problems can be found in Schaerf (1999); Carter and Laporte (1998). Many metaheuristics have been applied successfully to diverse variants of this problem. For instance, various ACO algorithms have been developed, e.g. a Max-Min Ant System is presented in Socha et al. (2002), and an ant colony system in Rossi-Doria and Paechter (2003). In Rossi-Doria and Paechter (2003) other metaheuristics including evolutionary algorithms, ant colony optimization, iterated local search and simulated annealing as well as neighborhood structures are compared. As these algorithms perform very differently depending on various properties of the problem instances, the authors conclude that hybrid algorithms may be a promising way to tackle the problem. The most successful approaches at the TTComp2002 are described in detail in (Kustoch 2003; Cordeau et al. 2003; Bykov 2003; Gaspero and Schaerf 2003; Chiarandini et al. 2003; Rossi-Doria and Paechter 2004). The algorithm proposed in Kustoch (2003) first constructs a feasible soultion and then applies simulated annealing to minimize the soft constraint violations. Also in Cordeau et al. (2003), a feasible solution is constructed at the beginning, but is then optimized by a tabu search algorithm. Local search strategies are successfully applied in Bykov (2003) and Gaspero and Schaerf (2003). A hybrid algorithm, described in Chiarandini et al. (2003), is reported to find very good results for many of the TTComp2002 instances. A memetic algorithm is presented in Rossi-Doria and Paechter (2004).

Our approach is based on ant colony optimization (ACO) and can be more specifically classified as Ant System (AS). For a comprehensive introduction to ant colony optimization see Dorigo and Stützle (2004). In ACO algorithms artificial ants successively construct solutions based on global information (pheromones) and local information (e.g. some greedy criterion). The pheromones hence act as a probabilistic model for solution construction and are perpetually amplified by ants that constructed high quality solutions. Pheromone evaporation counteracts premature convergence to a poor local optimum. A generic ant colony optimization procedure is shown in Algorithm 1.

More specifically, our AS can be described by Algorithm 2. The main task in designing an ACO algorithm is to devise the pheromone structures and update rules, and to find effective local methods able to drive the process towards promising regions of the search space. The respective parts of our algorithm are described in detail in the following sections.

Algorithm 1: ACO-Metaheuristic()

```
1  while not termination-criterion fulfilled do
2      solution construction by artificial ants
3      pheromone update
4      optional deamon actions
5  end
```

Algorithm 2: TimeTabling-AS()

```
1  while time limit not yet reached do
2      for each ant k = 1, ..., m do
3          create random permutation π^e of the events
4          for each event in order π^e do
5              assign event based on pheromones
6          end
7      end
8      locally improve each constructed solution
9      for each solution with a better than average score do
10         pheromone amplification for assignments appearing in solution
11     end
12     pheromone evaporation
13 end
```

2 Pheromone Information

In our algorithm ants are basically assigning events to timeslots and rooms based on two kinds of pheromone denoted by τ^{s}_{ij} and τ^{r}_{ik} which represent the probabilities of assigning an event i to timeslot j and room k, respectively. The decision to store pheromone information in this way is a key-feature of the algorithm, as it avoids the usage of a much larger data structure implied by a more traditional encoding using individual pheromone values for all slot/room/event combinations (see e.g. Socha et al. (2002)). On the other hand it contains more information than the exclusive use of event–timeslot pheromones (e.g. Rossi-Doria et al. (2002); Socha et al. (2003)).

Obviously, distinct pheromone matrices τ^{s} and τ^{r} are not as expressive as a three-dimensional pheromone matrix (τ_{ijk}) covering all combinations of events i, timeslots j, and rooms k would be. On the other hand such a three-dimensional matrix can be expected to be sparse, i.e. there are few elements different from zero. Thus, it is likely that the elements τ_{ijk} can be sufficiently well approximated by $\tau_{ijk} \approx \tau^{\mathrm{s}}_{ij} \times \tau^{\mathrm{r}}_{ik}$. Furthermore we do not expect particularily strong mutual dependencies of event–room and event–timeslot relations. Suppose some course E is required to be held in a subset of the rooms including room R_1 and the students attending this course as well as some precedence constraints require the event to be scheduled early at the first day, say at one of the slots $S_1, \ldots, S_m$. There is no obvious need to express the demand to assign E to exactly R_1

when using S_1 for instance, as all rooms satisfying the requirements of E will typically be equally good in this situation w.r.t. a current partial solution. These considerations directly lead to the conclusion that the assignment to a room is typically less critical than the assignment to a timeslot, which is also supported by our experiments. If there are mutual dependencies, they are handled implicitly by the solution construction procedure.

3 Solution Construction

The solution construction considers the events in a uniform random order and assigns each event to a feasible room and a feasible time slot in a greedy randomized way (if possible) considering the pheromone information. In more detail, for each event randomized weighted permutations of the available slots and rooms (π^{s} and π^{r}, respectively) are derived in such a way that slots (rooms) with higher pheromone values for the current event are more likely to appear earlier than slots (rooms) with low pheromone values. This can be compared to the fitness proportional selection in genetic algorithms as for each position an item is selected (from the remaining ones) with a probability proportional to the respective pheromones.

The ant then tries to assign the current event to a slot/room combination based on their order in π^{s} and π^{r}, respectively. The first possible assignment not violating any hard constraints w.r.t. the current partial solution is accepted. To ensure that both kinds of pheromone are accounted for in a balanced way, the slot/room combinations are considered in the following order: $(\pi^{\mathrm{s}}_1, \pi^{\mathrm{r}}_1), (\pi^{\mathrm{s}}_1, \pi^{\mathrm{r}}_2), (\pi^{\mathrm{s}}_2, \pi^{\mathrm{r}}_1), (\pi^{\mathrm{s}}_1, \pi^{\mathrm{r}}_3), (\pi^{\mathrm{s}}_2, \pi^{\mathrm{r}}_2), (\pi^{\mathrm{s}}_3, \pi^{\mathrm{r}}_1), \ldots, (\pi^{\mathrm{s}}_{45}, \pi^{\mathrm{r}}_r)$. See also Fig. 1.

To speed up this process the weighted random permutations are not entirely created in advance, but rather the requested elements are calculated on demand. Algorithm 3 performs this task for the slots; the room-pheromones are treated analogously. The algorithm returns the requested j-th element from the weighted random permutation π^{s}. The global array w^{s} is assumed to be filled with the respective rows of the pheromone matrices τ^{s} for the event i under consideration. Before the method is executed the first time, let $\sigma \leftarrow \sum_{l=0}^{45} w^{\mathrm{s}}_l$.

The integer variable *pos* stores the index to which the weighted permutation has yet been created. If i is less than *pos* no further computation is required. Otherwise the remaining elements are calculated (lines 3-20). The function swap(i, j) exchanges the elements $\pi^{\mathrm{s}}_i, \pi^{\mathrm{s}}_j$ and $w^{\mathrm{s}}_i, w^{\mathrm{s}}_j$ respectively. In the special case of all remaining weights being zero (line 14-18), some arbitrary element is chosen. Note that w^{s} and π^{s} do not need to be reinitialized for each ant. Moreover, for all but the first ants the method has a much better performance, as w^{s} and π^{s} are already roughly sorted in advance.

4 Pheromone Update

After each iteration ants with the lowest number of unplaced events and a better than average SCP-score add an amount of pheromone proportional to the

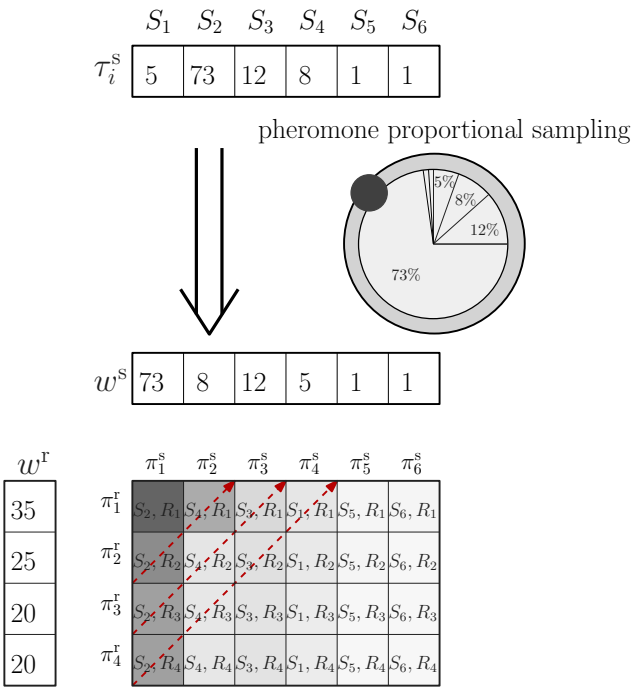

Fig. 1. Assignment of an event to a feasible room and timeslot: The pheromone matrix rows τ_i^{s} τ_i^{r} are copied into vectors w^{s} and w^{r} and sorted according to weighted random permutations π^{s} and π^{r} so that entries with higher values appear more likely at the beginning. Room/timeslot combinations are then checked in the indicated order, and the first feasible assignment is accepted.

solution quality for the performed event/slot and event/room assignments. For the event/slot assignments, this is done in detail as follows:

$$\triangle\tau_{ij} = \begin{cases} f \cdot g & \text{if } (i,j) \text{ part of the solution} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$$f = \begin{cases} \frac{100}{\#\text{unplaced}} & \text{if there are unplaced events} \\ 200 & \text{otherwise} \end{cases} \tag{2}$$

$$g = \begin{cases} \frac{1000}{\text{SCP}} & \text{SCP} \neq 0 \\ 2000 & \text{otherwise} \end{cases} \tag{3}$$

The numeric constants have been determined by preliminary experiments. They numerators are chosen such that for most solutions f and g are greater than one. The constants used when the denominator becomes zero are chosen to be twice the denominator, which is designed to boost good assignments somewhat.

If an assignment is found to cause a violation of soft constraints, the involved assignments are punished accordingly

Algorithm 3: getNextPermItem(j)

```
if j > pos then
    for j' = pos, ..., j do
        if σ > ε then
            rnd ← σ · random number
            q ← pos
            ξ ← 0
            while ξ < rnd and q < 45 do
                ξ ← ξ + w^s_q
                q ← q + 1
            end
            σ ← σ − w^s_{q−1}
            swap(q − 1, j')
        else
            // the remaining weights are all zero
            arbitrarily choose one of the remaining indizes rnd
            swap(rnd, j')
        end
        pos ← pos + 1
    end
end
return π^s_j
```

$$\Delta\tilde{\tau}_{ij} \leftarrow (1 - (1 - \gamma)^{\mathrm{SCP(e)}}) \cdot f \cdot g, \tag{4}$$

where $\mathrm{SCP}(e)$ denotes the soft constraint penalty induced by event e. This yields the following pheromone update:

$$\tau^{\mathrm{s}}_{ij} \leftarrow \max(0, \tau^{\mathrm{s}}_{ij} + \triangle\tau_{ij} - \triangle\tilde{\tau}_{ij}). \tag{5}$$

The maximum-calculation avoids negative pheromone values which might otherwise appear due to the penalization w.r.t. SCP. Finally, pheromone evaporation follows the standard AS method, which is

$$\tau^{\mathrm{s}}_{ij} \leftarrow (1 - \rho)\tau^{\mathrm{s}}_{ij}. \tag{6}$$

The pheromone update for the event–room pheromones τ^{r}_{ik} is performed analogously. In the case of stagnation, i.e. no new so-far-best solution has been found during the last 500 iterations, the pheromone values are normalized. The normalization procedure performs a linear scaling, such that the average value of the original pheromones remains the same and the maximum deviations from this average is relatively small, e.g. ten percent.

5 Improvement Method

Often, ACO approaches benefit significantly by including a local search procedure for improving candidate solutions derived by the ants. In our algorithm we employ an improvement heuristic that tries to move costly events (i.e. events which violate soft constraints) to a different timeslot if this can be achieved without violating any hard constraints while removing at most one other event from the solution. If an event needed to be removed, the procedure is applied recursively until either a suitable place is found or the maximum search depth is reached. We used a maximum search depth of 16 levels. At each level the events are ranked by their associated cost (i.e. SCP) and the most costly events are tried first. A chain of moves is accepted if it reduces the total soft constraint penalty. If the move has been accepted, the search is aborted. This improvement heuristic is applied to all events with violate soft constraints.

6 Computational Results

The algorithm is parameterized by α (importance of the pheromones), β (importance of local information), ρ (pheromone evaporation) and η (number of ants). Robust parameter values have been determined by preliminary experiments. It turned out that $\alpha \in [1.0, 1.1]$ yields good results, and the choices of γ and η are less critical. Lower values of γ and η reduce the time to obtain a solution with DTF=0, but decrease the average solution quality as well. As no local information is used in our algorithm, $\beta = 0$. The pheromone matrices τ^{r} and τ^{r} are initialized with the value 0.5. Table 1 lists the parameter settings used for the subsequent computational experiments.

Parameter	Value	Description
α	1.0	importance of pheromone
ρ	0.25	pheromone evaporation
γ	0.3	penalty
η	20	number of ants
$t_{\max}$	299/294 s	CPU-time limit for ITC2007/TTComp2002

Table 1. Parameter values used for our computational experiments.

We did not use any local heuristic information. This information is sometimes called the visibility function, especially in the case of the traveling salesman problem. In this case, however, it is very hard to find a meaningful function which expresses the desirability of a particular assignment. Furthermore, a comparison of different optimal solutions showed few similarities, which is evidence that for the solution this problem such a function is not necessarily beneficial.

We performed our computational experiments with the instances used for the First International Timetabling Competition (TTComp2002) and the early

Instance	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
SCP_{best}	55	43	61	134	134	32	52	48	39	77	39	102	94	109	47	26	78	35	119	19
SCP_{avg}	82	64	92	208	185	59	138	107	70	118	75	143	156	175	89	45	143	59	187	38
σ_{SCP}	10	9	11	41	20	16	26	19	13	15	12	17	22	29	18	9	22	8	29	7
Kostuch	45	25	65	115	102	13	44	29	17	61	44	107	78	52	24	22	86	31	44	7
Jaumard et al.	61	39	77	160	161	42	52	54	50	72	53	110	109	93	62	34	114	38	128	26
Bykov	85	42	84	119	77	6	12	32	184	90	73	79	91	36	27	300	79	39	86	0
Gaspero et al.	63	46	96	166	203	92	118	66	51	81	65	119	160	197	114	38	212	40	185	17
Chiarandini et al.	57	31	61	112	86	3	5	4	16	54	38	100	71	25	14	11	69	24	40	0
Socha	65	36	69	138	143	24	24	28	36	75	50	95	79	73	31	23	108	26	108	5

Table 2. This table shows results on the TTComp2002 instances. Lines 2 to 4 list best and average SCP scores as well as corresponding standard deviations of final solutions obtained by our AS in 100 runs/instances. DTF is zero in all cases. Lines 5 to 8 list the best results of the TTComp2002 top four contestants (which could be replicated by the organizers) (Kustoch 2003; Cordeau et al. 2003; Bykov 2003; Gaspero and Schaerf 2003). The last two rows show further excellent results obtained by Chiarandini et al. (2003); Socha et al. (2002), who did not officially participate at the competition. All results can be found on the competition website.

and late data sets used for the Second International Timetabling Competition (ITC2007). All experiments have been executed on an Intel Xeon 5160 (3.0 GHz) Windows Server 2003 Standard 64-bit with Sun Java SE 1.6.0_04 64-bit Server VM. With the benchmark-tools for the competitions, we determined $t_{\max} = 294$ seconds as permitted CPU-time for TTComp2002 and $t_{\max} = 299$ seconds for ITC2007. All results (except those presented later in Fig. 2) have been obtained within these time-limits.

Table 2 shows the results for the TTComp2002 instances. As the algorithm is especially tuned for ITC2007 (considering in particular the hard constraints 4 and 5) and no modifications have been done in order to especially support TTComp2002 instances, the results are only moderate. Nevertheless, the average values lie in the range of the best four participants of TTComp2002.

Table 3 lists the results achieved for the ITC2007 instances. The first two rows show that an optimum solution could be found within the given time limit in many cases. Furthermore the probability of finding a solution with DTF=0 is very high, and also the average SCP values are very promising. Within the given time limit, on average 5000 iterations have been performed. Figure 2 shows time-to-target plots for two typical instances.

We also tested the algorithm without applying the local improvement method. The obtained results indicate that local improvement leads to slightly but significantly better solutions on average. However, it is remarkable that even without this procedure high quality solutions are usually found.

Instance	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$\mathrm{DTF}_{\mathrm{best}}$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
$\mathrm{SCP}_{\mathrm{best}}$	0	0	110	53	13	0	0	0	0	0	143	0	5	0	0	0
$\mathrm{DTF}_{\mathrm{avg}}$	237	274	0	0	0	2	0	0	109	0	0	20	2	0	0	0
$\mathrm{SCP}_{\mathrm{avg}}$	613	556	680	580	92	212	4	61	202	4	774	538	360	41	29	101
σ_{DTF}	290	369	0	0	0	9	0	0	294	0	6	56	9	0	0	0
σ_{SCP}	612	671	255	268	55	165	24	47	492	18	247	605	167	56	104	72
$P(DTF = 0)$	.54	.59	1.0	1.0	1.0	.95	1.0	1.0	.85	1.0	.99	.86	.94	1.0	1.0	1.0

Table 3. Results for the ITC2007 early and late instances. The first two rows indicate the best results achieved within the given time limit. The distance to feasibility is denoted by DTF, soft constraint penalty by SCP. The subsequent rows list average solution values of 100 runs followed by the respective standard deviations and the probability to reach DTF=0.

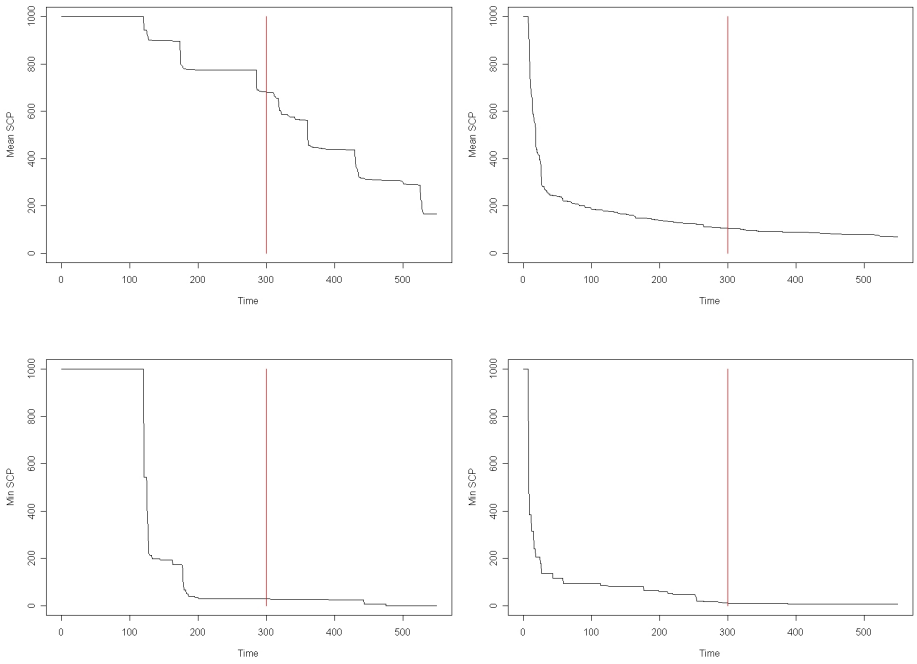

Fig. 2. Time-to-target plots showing the SCP-values for instances 1 and 5 of ITC2007. The upper plots depict average values over ten runs, the plots on the bottom show the corresponding minimum values.

7 Conclusions

The large majority of the competition instances can with high probability be solved to optimality within a couple of minutes (i.e. the time limit for the ITC2007 competition).

The algorithm ranked 4th among all submitted algorithms, which demonstrates, that the presented algorithm falls within the leading algorithms for this task. For 5 out of 24 instances our algorithm found the best solution of all 5 finalists, for a further 6 instances we tied another algorithm for the best solution. On the other hand, our algorithm also showed the largest variation in solution quality, for several instances it produced both the best and the worst solution. This is an indication that it was tuned to be too aggressive, favoring a single good solutions over repeatably good solutions. It remains to be examined if the algorithm can also be tuned to show less variation in solution quality.

From our point of view the key feature of the algorithm is the use of two distinct but relatively compact pheromone matrices in combination with an effective procedure to exploit their information in the heuristic solution construction. The algorithm is able to produce high quality solutions even without the local improvement method, but better results could be achieved when including it.

Further experimental investigations on a larger set of instances with different characteristics need to be performed in order to get a deeper understanding of the advantages and disadvantages of the described approach and the influence of its various strategy parameters. Also, combinations with other local improvement techniques may be studied for eventually further improving the performance.

Bibliography

Bykov, Y. (2003), The Description of the Algorithm for International Timetabling Competition, *in* 'International Timetabling Competition Results', http://www.idsia.ch/Files/ttcomp2002/bykov.pdf.

Carter, M. W. and Laporte, G. (1998), Recent developments in practical course timetabling, *in* 'Practice and Theory of Automated Timetabling II', Springer, London, UK, pp. 3–19.

Chiarandini, M., Socha, K., Birattari, M. and Rossi-Doria, O. (2003), International Timetabling Competition – A Hybrid Approach, *in* 'International Timetabling Competition Results', http://www.idsia.ch/Files/ttcomp2002/chiarandini.pdf.

Cordeau, J.-F., Jaumard, B. and Morales, R. (2003), Efficient Timetabling Solution with Tabu Search, *in* 'International Timetabling Competition Results', http://www.idsia.ch/Files/ttcomp2002/jaumard.pdf.

Dorigo, M. and Stützle, T. (2004), *Ant Colony Optimization*, MIT Press.

Gaspero, L. D. and Schaerf, A. (2003), Timetabling Competition TTComp 2002: Solver Description, *in* 'International Timetabling Competition Results', http://www.idsia.ch/Files/ttcomp2002/schaerf.pdf.

ITC2002 (2002), '`http://www.idsia.ch/Files/ttcomp2002/` First International Timetabling Competition (2002)'.

ITC2007 (2007), '`http://www.cs.qub.ac.uk/itc2007/` Second International Timetabling Competition (2007)'.

Kustoch, P. A. (2003), Timetabling Competition – SA-based heuristics, *in* 'International Timetabling Competition Results', http://www.idsia.ch/Files/ttcomp2002/kostuch.pdf.

Lewis, R., Paechter, B. and McCollum, B. (2007), Post enrolment based course timetabling: A description of the problem model used for track two of the second international timetabling competition, Cardiff accounting and finance working papers, Cardiff University, Cardiff Business School, Accounting and Finance Section.

Rossi-Doria, O. and Paechter, B. (2003), An hyperheuristic approach to course timetabling problem using an evolutionary algorithm, Technical Report CC-00970503, Napier University, Edinburgh, Scotland.

Rossi-Doria, O. and Paechter, B. (2004), A memetic algorithm for university course timetabling, *in* 'Combinatorial Optimisation 2004 Book of Abstracts', p. 56.

Rossi-Doria, O., Sampels, M., Birattari, M., Chiarandini, M., Dorigo, M., Gambardella, L. M., Knowles, J. and Manfrin, M. e. a. (2002), A comparison of the performance of different metaheuristics on the timetabling problem, *in* 'Practice and Theory of Automated Timetabling IV: 4th International Conference, PATAT 2002, Gent, Belgium, August 21-23, 2002: Selected Revised Papers', Springer.

Schaerf, A. (1999), 'A Survey of Automated Timetabling', *Artificial Intelligence Review* **13**(2), 87–127.

Socha, K., Knowles, J. and Sampels, M. (2002), A Max-Min ant system for the university timetabling problem, *in* M. Dorigo, G. Di Caro and M. Sampels, eds, 'Proceedings of the 3rd International Workshop on Ant Algorithms, ANTS 2002', Vol. 2463, Springer, pp. 1–13.

Socha, K., Sampels, M. and Manfrin, M. (2003), Ant algorithms for the university course timetabling problem with regard to the state-of-the-art, *in* J. Gottlieb and G. Raidl, eds, 'Proceedings of EvoCOP 2003 – 3rd European Workshop on Evolutionary Computation in Combinatorial Optimization', Vol. 2611 of *LNCS*, Springer.