Ant algorithms for the exam timetabling problem

Michael Eley

Aschaffenburg University of Applied Science
Logistics Laboratory (LlAb)
Aschaffenburg / Germany
michael.eley@fh-aschaffenburg.de

Abstract. Scheduling exams at universities can be formulated as a combinatorial optimization problem. Given a planning horizon with a fixed number of periods the objective is to avoid situations, or at least to minimize them, when a student is enrolled in two exams that are scheduled for the same period. Ant colony approaches have been proven to be a powerful solution approach for various combinatorial optimization problems. In this paper a Max-Min and a ANTCOL approach will be presented. Its performance is compared with other approaches presented in the literature and with modified graph coloring algorithms.

Key words: scheduling, exam timetabling, ant colony algorithms, Max-Min approach, graph coloring

1 Introduction

The exam timetabling problem faces the problem of scheduling exams within a limited number of available periods. As students plan to write different exams, setting up a conflict free timetable is not a trivial task due to limited resources like periods, examination rooms and teacher availability. The main objective is to balance out student's workload and to distribute the exams evenly within the planning horizon. In particular, it should be avoided that a student has to write two exams in the same period. Such situations will be referred to as conflicts of order 0 in the sequel. Additionally, as few students as possible have to attend x exams within y consecutive periods. Such conflicts can either be totally forbidden by constraints or penalized in the objective function. For example, Carter et al. proposed in [1] a cost function that imposes penalties P_ω for a conflict of order ω, i.e. whenever one student has to write two exams scheduled within $\omega + 1$ consecutive periods. In the literature ω normally runs from 1 to 5 with $P_1 = 16, P_2 = 8, P_3 = 4, P_4 = 2, P_5 = 1$.

Solving practical exam timetabling problems requires that additional constraints have to be considered, e.g. some exams have to be written before other

E. K. Burke, H. Rudová (Eds.): PATAT 2006, pp. 167–180. ISBN 80-210-3726-1.

exams or some exams can not be written within specific periods. References [2–4] give comprehensive lists of possible hard and soft constraints.

The exam timetabling problem can be formulated as a graph coloring problem. Each node represents one exam. Undirected arcs connect two nodes if at least one student is enrolled in both corresponding exams. Weights on the arcs represent the number of student enrolled in both exams. The objective is to find a coloring where no adjacent nodes are marked with the same color or to minimize the weighted sum of the arcs that connect two nodes marked with the same color. The exam timetabling problem is a generalization of the graph coloring problem as in the objective function also conflicts of higher orders are penalized.

To solve exam timetabling problems, several algorithms have recently been developed. In [1] Carter et al. applied some well known graph coloring heuristics which they combined with backtracking.

In recent time various heuristical approaches have been developed. Most of them use local search like tabu search, simulated annealing, great deluge or adaptive search methods [5, 6, 1, 7, 8, 2, 9–11]. A comprehensive survey on the literature on exam timetabling problems can be found in [4].

The aim of this paper is twofold: Originally, this research was motivated by the need for a software tool for solving a practical exam timetabling problem. As ant colony approaches have been proven to be a powerful tool for various combinatorial optimization problems (c.f. the survey in [12]), it is apparent to adapt this solution approach to the exam timetabling problem. In the literature different variants of ant colony approaches have been presented. We will compare some of these strategies with respect to their suitability for our problem.

This paper is organized as follows: In section 2 a detailed problem formulation will be presented. Section 3 will give an introduction into ant colony systems. The next sections will present a solution approach and test results for some benchmark problems that were taken from the literature. Finally, section 6 summarizes the results and suggests discussion for future work.

2 Problem formulation

Before stating the problem formally, we introduce some notation.

R	index set of rooms
I	index set of exams
T	index set of periods
Ω	index set of order of conflicts
K_{rt}	capacity of room r in period t
c_{ij}	number of students enrolled in exam i as well as in exam j
E_i	number of students enrolled in exam i
P_ω	penalty imposed if one student has to write two exams within $\omega + 1$ periods
y_{it}	binary variable equal to 1 if exam i is scheduled in period t and 0 otherwise
p_{irt}	number of students of exam i assigned to room r in period t

Using this notation, the exam timetabling problem can be formulated as follows:

$$min \sum_{\omega \in \Omega} \sum_{i,j \in I, i \neq j} \sum_{t \in T, t > \omega} P_\omega c_{ij} y_{it} y_{j(t-\omega)} \tag{1}$$

s.t.

$$\sum_{t \in T} y_{it} = 1 \quad \forall i \in I \tag{2}$$

$$p_{irt} \leq y_{it} K_{rt} \quad \forall i \in I, \forall r \in R, \forall t \in T \tag{3}$$

$$\sum_{r \in R} \sum_{t \in T} p_{irt} = E_i \quad \forall i \in I \tag{4}$$

$$\sum_{i \in I} p_{irt} \leq K_{rt} \quad \forall r \in R, \forall t \in T \tag{5}$$

$$\sum_{t \in T} c_{ij} y_{it} y_{jt} = 0 \quad \forall i, j \in I, i \neq j \tag{6}$$

$$y_{it} \in \{0, 1\} \quad \forall i \in I, \forall t \in T \tag{7}$$

$$p_{irt} \in \mathbb{N}_0 \quad \forall i \in I, \forall r \in R, \forall t \in T \tag{8}$$

The objective function (1) balances out students' workload by minimizing the weighted sum of all conflicts. Constraint (2) states that each exam is assigned to exactly one period. If an exam is not assigned within a period, then no seats should be reserved for that period in any room. This is imposed by constraint (3). Constraints (4) and (5) assure that the number of seats reserved for an exam will be equal to the number of students who are enrolled in that exam and that the room capacities are not exceeded. Finally, constraint (6) avoids conflicts of order 0, i.e. that a student has to write two exams in the same period.

The exam timetabling problem is a generalization of the graph coloring problem, which is known to be NP-hard [13]. Therefore, solution approaches try to decompose the problem in order to solve it within a reasonable amount of time [14]. One way is to split up the problem into the two following subproblems, which can be solved sequentially:

Problem I: *Scheduling of exams*, i.e. assign exams to periods in order to balance out students' workload as pursued by the objective function (1). Instead of considering capacity constraints for the single rooms, only the total capacity of all available exam rooms within a period is considered. In the IP formulation stated above this can be accomplished by replacing the set of rooms by a artificial single room. For this problem a solution approach will be presented in the next sections.

Problem II: *Room planning*, i.e. distribute the exams of one period among the available examination rooms. Finding a feasible room plan is not difficult if the exams can take place in more than one room and if more than one exam can take place in one room at the same time, provided that the room capacity is not exceeded. If exams are split up into different rooms one could consider the campus layout and try to generate a room plan where these exams are only assigned to rooms not too far from each other in order to minimize walking distances. We will not consider this problem in the following.

3 Ant algorithms

Ant colony optimization algorithms represent special solution approaches for combinatorial optimization problems derived from the field of swarm intelligence. They were first introduced by Colorni, Dorigo and Maniezzo in the early nineties [15]. See [12] for an in depth introduction into ant systems.

Ant algorithms were inspired by the observation of how real ant colonies find shortest paths between food sources and their nest. This observation was first implemented in algorithms for solving the traveling salesperson problem (TSP). This type of ant colony optimization algorithm is known in the literature as ant systems (AS). We will briefly describe the basic principle of AS algorithms by means of the TSP. This solution approach to the TSP will be adopted to solving the exam timetabling problem in the next section.

The solution approach consists of n cycles. In each of these cycles first each of the m ants constructs a feasible solution. In AS each ant builds a complete tour that visits all nodes. Obviously, this solution neither has to be optimal nor must it be even close to the (unknown) optimal value. Improved solutions can be obtained if the knowledge gathered by other ants in the past on how good solutions can be obtained is incorporated into the ant's decision. Assume that an ant is located in a node i. To choose the next node j that has not yet been visited by that ant one may apply one of the following two randomized strategies:

Strategy I: *Constructive heuristic.* Apply one priority rule like randomized nearest neighbor. Decision values for all nodes j are determined by the inverse of the distance from node i to that node j. The next node the ant moves to is then randomly chosen according to the probabilities determined by those decision values. Consequently, if node j_1 is closer to i than node j_2 it is more likely to choose node j_1. The decision values of the constructive heuristic will be later referred to as η_{ij}.

Strategy II: *Pheromone trails.* This strategy is mainly inspired by the way real ants find shortest paths. While commuting between two places on different possible pathes ants deposit a chemical substance called pheromone. The shorter the path is the more often the ant will use this path within a limited period of time and, consequently, the larger the amount of pheromone will be on that path. Thus, whenever an ant has to choose between different available paths it will prefer the one with higher amount of pheromone.

To adapt these observations to the TSP, the amount of pheromone is stored in a matrix τ which is initialized with 0 for all arcs (i, j). After an ant has completed a tour, the values of the cells that belong to the arcs the ant has chosen are updated by the inverse of the obtained objective function value, i.e. the length of the tour. The amount of pheromone trail τ_{ij} associated to arc (i, j) is intended to represent the learned desirability of choosing node j when in node i. Consequently, arcs belonging to good solutions receive a high amount of pheromone.

AS algorithms combine these two strategies. The probability that an ant ν located in node i chooses the next node j is determined by the following formula:

$$p_{ij}^{\nu} = \begin{cases} \frac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum_{k \in N_i^{\nu}} (\tau_{ik})^{\alpha}(\eta_{ik})^{\beta}} & \text{if } j \in N_i^{\nu} \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

α and β are a given weighting factors and N_i^{ν} is the set of nodes that have not yet been visited by ant ν currently located in node i.

Excepting the TSP, AS algorithms have been implemented for various combinatorial optimization problems, such as the quadratic assignment problem or the sequential ordering problem. Different variants of AS algorithms have been suggested in the literature, like e.g. ant colony systems (ACS) or Max-Min ant systems (MMAS), which obtained much better results than AS (c.f. [12]). In particular, MMAS, which was first proposed by Stützle and Hoos [16], generated significantly better solutions for the TSP than AS. Socha et al. [17] compared the MMAS variant with ACS and found out that MMAS outperformed the ACS approach for the considered timetabling problem.

The main modification of MMAS are related to the way how the matrix τ is initialized and how pheromone values are updated. Additionally, MMAS uses local search to improve the solutions found by the ants. Details will be discussed in the next section.

As far as the author is aware, ant colony algorithms to scheduling problems have only been applied by Colorni et al. [15] and by Socha et al. [17]. The former article focuses on the job shop scheduling problem, the latter one on the timetabling problems for university classes, which are slightly different from the exam timetabling problem considered here. Finally, Costa and Hertz [18] used an ant colony approach to solve assignment type problems, in particular graph coloring problems. Recently, Dowsland and Thomson as well as Vesel and Zerovnik modified and improved in [19, 20] this graph coloring algorithm with respect to the examination scheduling problem.

4 An ant algorithm for the exam scheduling problem

4.1 General modifications for the exam timetabling problem

Like in AS, the solution approach consists of n cycles. In each of these cycles first each of the m ants constructs a feasible solution using therefore the constructive

heuristic and the pheromone trails. These exam schedules are then evaluated according to the given objective function and the experience accumulated during the cycle is used to update the pheromone trails.

Depending on the choice of a constructive heuristic and the way the pheromone values are used, there are different ways how this basic solution approach can be adapted to the exam timetabling problem.

- At each stage of the construction process in the AS approach of Costa and Hertz [18] called ANTCOL the ant chooses first a node i and then a feasible color according to a probability distribution equivalent to (9). The matrix τ provides information on the objective function value, i.e. the number of colors required to color the graph, which was obtained when nodes i and j are colored with the same color.
  In contrast to elite strategies where only the ant that found the best tour from the beginning of the trial deposits pheromone, all ants deposit pheromone on the paths they have chosen. According to [12] this strategy is called ant cycle strategy.
  Different priority rules were tested as constructive heuristic. Among those chosen in each step, the node with the highest degree of saturation, i.e. the number of different colors already assigned to adjacent nodes, achieved the best results with respect to solution quality and computation times.
- In Socha et al. [17] a pre-ordered list of events is given. Each ant chooses the color for a given node probabilistically similar to the formula (9). The pheromone trail τ_{ij} contains information on how good the solution was, when node i was colored by color t. The constructive heuristic employed in their approach is not described.

For the exam timetabling problem the way the information in matrix τ is used in both approaches is not meaningful. Due to the conflicts of higher orders the quality of a solution does not depend on how a pair of exams is scheduled nor on the specific period an exam is assigned to. For example, assigning two exams i and j with $c_{ij} = 0$ to the same period can either result in a high or in a low objective function value as the quality of the solution strongly depends on when the remaining exams are scheduled. In the following we implemented a two step approach.

Step I: Determine the sequence according to the exams is scheduled. Like for the TSP we assume that an ant located in a node, corresponding to an exam, has to visit all other nodes, i.e. it has to construct a complete tour. The sequence according to this ant constructs the tour corresponds to the sequence in which the exams are scheduled.

Step II: Find the most suitable period for an exam which should be scheduled. Therefore, all admissible periods are evaluated according to the given penalty function.

Following this two step approach probabilities p_{ij}^{ν} for choosing the next node j that has to be scheduled are computed according to (9). Pheromone values τ_{ij}

along the ants' paths are updated by the inverse of the objective function value. For the heuristic value η_{ij} the following simple priority rule for graph coloring was implemented. The exam with the smallest number of available periods is selected. A period would not be available for an exam if it caused a conflict of order 0 with another exam that has already been scheduled. This priority rule corresponds to the saturation degree rule (SD) which was tested in [1]. The value η_{ij} is chosen to be the inverse of the saturation degree.

4.2 MMAS specifications

MMAS approaches mainly differ from AS algorithms in the way they use the existing information (c.f. [16]):

- Pheromone trails are only updated by the ant that generated the best solution in a cycle. The corresponding values τ_{ij} are updated by $\rho\tau_{ij} + 1/f^{best}$ where f^{best} is equal to the best objective function value found so far. For all other arcs (i, j) that are not chosen by the best ant τ_{ij} is updated by $(1 - \rho)\tau_{ij}$. $\rho \in [0, 1]$ represents the pheromone evaporation factor, i.e. the percentage of pheromone that decays within a cycle.
- Pheromone trail values are restricted to the interval $[\tau_{min}, \tau_{max}]$, i.e. whenever after a trail update $\tau_{ij} < \tau_{min}$ or $\tau_{ij} > \tau_{max}$ then τ_{ij} is set to τ_{min} or τ_{max}, respectively. The rationale behind this are that if the differences between some pheromone values were too large, all ants would almost always generate the same solutions. Thus, stagnation is avoided.
- Pheromone trails are initialized to their maximum values τ_{max}. This type of pheromone trail initialization increases the exploration of solutions during the first cycle.

The solution quality of ant colony algorithms can be considerably improved when it is combined with additional local search. In hybrid MMAS only the best solution within one cycle is improved by local search. For the exam timetabling problem a hill climber procedure has been implemented. Within an iteration of the hill climber two sub-procedures are carried out in succession. The hill climber is stopped if no improvement can be found within an iteration.

Within the first sub-procedure of the hill climber for all exams the most suitable period is examined. Beginning with the exam that causes the biggest contribution to the objective function value, all feasible periods are checked and the exam is assigned to its best period. The first sub-procedure is stopped if all exams have been checked without finding an improvement. Otherwise the contributions to the objective function value are recalculated and the process is repeated.

The second sub-procedure tries to decrease the objective function value by swapping all exams within two periods, i.e. all exams assigned to period t' are moved to period t'' and the exams of that period are moved to period t'. Therefore all pairs of periods are examined and the first exchange that leads to an improvement is carried out. Again, the process is repeated as long as the objective function value is decreased.

Finally, the use of a so called candidate list has been proven to reduce required computational times as well as to improve solution quality at the same time (c.f. [12]). Such a list provides additional local heuristic information as it contains preferred nodes to be visited from a given node. Instead of scanning all other exams only the exams in the candidate list are examined and only in case all exams in this list have already been scheduled, the remaining exams are considered.

5 Computational experiments

The proposed Max-Min algorithm was implemented in Borland Delphi 7.0. It will be referred to as MMAS-ET in the sequel. Test runs were carried out on a computer with 3.2 GHz clock under Windows XP.

5.1 Test cases

To benchmark algorithms test cases of twelve practical examination problems can be found on the site of Carter (c.f. [21]). Table 1 summarizes some characteristics of these problems. To make a comparison meaningful all algorithms must use the same objective function. Therefore, Carter proposed weighting conflicts according to the following penalty function: $P_1 = 16, P_2 = 8, P_3 = 4, P_4 = 2, P_5 = 1$, where P_ω is the penalty for the constrain violation of order ω. The cost of each conflict is multiplied by the number of students involved in both exams. The objective function value represents the costs per student. As the proposed MMAS-ET algorithm does not guarantee that no conflicts of order 0 occur, additionally, the penalty P_0 was imposed and set to 10000.

Table 1. Test cases from Cater et al. [1, 21, 22]

test case	# exams	# students	# student exams	problem density	# periods
car-f-92	543	18419	55522	13.8 %	32
car-s-91	682	16925	56877	12.8 %	35
ear-f-83	190	1125	8109	26.7 %	24
hec-s-92	81	2823	10632	42.0 %	18
kfu-s-93	461	5349	25113	5.6 %	20
lse-f-91	381	2726	10918	6.3 %	18
pur-s-93	2419	30032	120681	2.9 %	43
rye-f-92	486	11483	45051	7.5 %	23
sta-f-83	139	611	5751	14.4 %	13
tre-s-92	261	4360	14901	5.8 %	23
uta-s-92	622	21267	58979	12.6 %	35
ute-s-92	184	2750	11793	8.5 %	10
yor-f-83	181	941	6034	28.9 %	21

5.2 Adjustment of the parameters

The required parameters were specified as follows. The number of cycles was set to 50. Within each cycle 50 ants were employed to construct solutions. The candidate list contained the 20% of exams with the lowest number of available periods. Several test runs were carried out in order to determine the required parameters appropriately:

- The evaporation rate ρ was set to 0.3. Like in [16] it turned out that this parameter is quite robust, i.e. the parameter ρ does not clearly influence the performance.
- For the restrictions of the pheromone interval values to strategies were tested. Setting $\tau_{max} = 1/\rho$ obtained slightly better results than in the case of variable τ_{max} and τ_{min} as proposed in [16].
- Different values for the weighting factors α and β were tested. It turned out that the approach performed best when α was set to one and β was chosen high. Best results were obtained for β equal to 24. But the difference was on the average less than one percent when β was bigger than eight. A high β forces that exams which can be scheduled, due to zero order conflicts, only in a few remaining periods are scheduled first as they are given a much higher probability in (9). Remember that η_{ij} is the inverse of the saturation degree as explained in section 4.1. Thus, a high β value has the same effect like a candidate list. This could be a reason why the use of the candidate list did not improve the solutions. Whereas, for small values of β, i.e. values lower than 5, solutions with zero order conflicts could not always be avoided.
- As the approach is non-deterministic each test case was solved twenty times.

After determining the parameters in such a way, it turned out that less than 2 % of the solutions were generated more than once. Thus, stagnation, that is caused by the fact that many ants generate almost the same solutions, could not be observed.

5.3 Test results for the MMAS-ET approach

Table 2 displays the results for different approaches. For each approach the minimal objective function value and the average result after twenty test runs are given. Results of the proposed MMAS-ET approach are given in the second column.

In order to find out how much the hill climber contributes to the solution the MMAS-ET approach was also tested without making use of the hill climber. Comparing the results in the second and in the third column it is obvious that the hill climber considerably improves the solutions.

Thus, one could ask how much the ants contribute to the solution or if solutions of the same quality could also be achieved by applying only the hill climber on a random starting solution. Therefore a third version of the MMAS-ET approach was implemented where each ant constructs an exam timetable without interacting with the other ants, i.e. the matrix τ is not updated at all. This

approach can be seen as a randomized greedy heuristic. As in MMAS-ET with 50 ants and 50 cycles 2500 exam timetables were generated. The best solutions of this approach are displayed in the last column of table 2.

As the MMAS-ET approach without ants generates the worst solutions it is obvious, that the ant colony has a positive impact on the diversification of the solution space, i.e. the ants guide the search process into promising regions of the solution space where the hill climber can find good solutions.

Increasing the number of ants and the number of cycles to 100 in the MMAS-ET approach did not result in achieving better solutions. Neither the average value of all twenty iterations was improved nor were better solutions found during the twenty iterations.

Table 2. Results for three different variants of the MMAS-ET approach for twenty test runs

	MMAS-ET		MMAS-ET without hill climber		MMAS-ET without ants	
test case	best	avg.	best	avg.	best	avg.
car-f-92	4.8	4.9	7.8	8.0	10.9	13.3
car-s-91	5.7	5.9	9.3	9.5	11.9	13.9
ear-f-83	36.8	38.6	50.4	53.0	49.5	62.4
hec-s-92	11.3	11.5	14.8	15.8	11.6	15.5
kfu-s-93	15.0	15.5	23.9	24.6	19.5	22.0
lse-f-91	12.1	12.7	19.3	19.8	16.7	25.4
pur-s-93	5.4	5.6	12.2	12.5	11.7	14.6
rye-s-93	10.2	10.4	18.0	18.7	12.2	14.2
sta-f-83	157.2	157.5	160.6	161.9	157.3	157.7
tre-s-92	8.8	9.1	12.4	12.8	9.2	13.1
uta-s-92	3.8	3.8	6.2	6.3	8.2	9.9
ute-s-92	27.7	28.6	33.6	34.5	27.7	30.1
yor-f-83	39.6	40.3	50.5	51.3	62.9	73.0

5.4 Comparison with other exam timetabling approaches

The proposed MMAS-ET approach was compared with the following approaches:

- LD, SD, LDW and LE: Carter et al. compared in [1] four different priority rules largest degree (LD), saturation degree (SD), largest weighted degree (LWD) and largest enrollment (LE).
- Wal: Tabu search approach with longer-term memory proposed by White et al. in [11].
- GS: Tabu search approach proposed by Di Gaspero and Schaerf in [2].
- Cal: Local search approach of Caramia et al. [6].
- BN: Great deluge local search approach developed by Burke and Newall [5].

– Mal: Simulated annealing approach of Merlot et al. [9].
– Ga: Multi-neighborhood search approach presented by Di Gaspero [8]
– PS: Tabu search approach of Paquete and Stützle [10].
– CT: Randomized adaptive search algorithm of Casey and Thomson [7].

The results of the benchmarks are taken from the literature [11] and from the internet (c.f. the timetabling database at the University of Melbourne [22]). Table 3 displays the best solution and the average solution achieved when each test case was solved twenty times. The results of table 3 can be summarized as follows:

Table 3. Best (b.) and average (a.) solution after twenty test runs for the benchmark test cases from Carter et al.[1, 21, 22]

test case		LD	SD	LWD	LE	Wal	GS	Cal	BN	Mal	Ga	PS	CT	MMAS -ET
car	b.	7.6	6.6	6.6	6.2	4.6	5.2	6.0	4.0	4.3	-	-	4.4	4.8
-f-92	a.	7.6	6.6	6.6	6.2	4.7	5.6	6.0	4.1	4.4	-	-	4.7	4.9
car	b.	7.9	7.1	7.4	7.6	5.7	6.2	6.6	4.6	5.1	5.7	-	5.4	5.7
-s-91	a.	7.9	7.1	7.4	7.6	5.8	6.5	6.6	4.7	5.2	5.8	-	5.6	5.9
ear	b.	36.4	46.5	37.3	42.3	45.8	45.7	29.3	36.1	35.1	39.4	40.5	34.8	36.8
-f-83	a.	36.4	46.5	37.3	42.3	46.4	46.7	29.3	37.1	35.4	43.9	45.8	35.0	38.6
hec	b.	10.8	12.7	15.8	15.9	12.9	12.4	9.2	11.3	10.6	10.9	10.8	10.8	11.3
-s-92	a.	10.8	12.7	15.8	15.9	13.4	12.6	9.2	11.5	10.7	11.4	12.0	10.9	11.5
kfu	b.	14.0	15.9	22.1	20.8	17.1	18.0	13.8	13.7	13.5	-	16.5	14.1	15.0
-s-93	a.	14.0	15.9	22.1	20.8	17.8	19.5	13.8	13.9	14.0	-	18.3	14.3	15.5
lse	b.	12.0	12.9	13.1	10.5	14.7	15.5	9.6	10.6	10.5	12.6	13.2	14.7	12.1
-f-91	a.	12.0	12.9	13.1	10.5	14.8	15.9	9.6	10.8	11.0	13.0	15.5	15.0	12.7
pur	b.	4.4	4.1	5.0	3.9	-	-	3.7	-	-	-	-	-	5.4
-s-93	a.	4.4	4.1	5.0	3.9	-	-	3.7	-	-	-	-	-	5.6
rye	b.	7.3	7.4	10.0	7.7	11.6	-	6.8	-	8.4	-	-	-	10.2
-s-93	a.	7.3	7.4	10.0	7.7	11.7	-	6.8	-	8.7	-	-	-	10.4
sta	b.	162.9	165.7	161.5	161.5	158.0	161.0	158.2	168.3	157.3	157.4	158.1	134.9	157.2
-f-83	a.	162.9	165.7	161.5	161.5	158.0	167.0	158.2	168.7	157.4	157.7	159.3	135.1	157.5
tre	b.	11.0	10.4	9.9	9.6	8.9	10.0	9.4	8.2	8.4	-	9.3	8.7	8.8
-s-92	a.	11.0	10.4	9.9	9.6	9.2	10.5	9.4	8.4	8.6	-	10.2	8.8	9.1
uta	b.	4.5	3.5	5.3	4.3	4.4	4.2	3.5	3.2	3.5	4.1	-	-	3.8
-s-92	a.	4.5	3.5	5.3	4.3	4.5	4.5	3.5	3.2	3.6	4.3	-	-	3.8
ute	b.	38.3	31.5	26.7	25.8	29.0	29.9	24.4	25.5	25.1	-	27.8	25.4	27.7
-s-92	a.	38.3	31.5	26.7	25.8	29.1	31.3	24.4	25.8	25.2	-	29.4	25.5	28.6
yor	b.	49.9	44.8	41.7	45.1	42.3	41.0	36.2	36.8	37.4	39.7	38.9	37.5	39.6
-f-83	a.	49.9	44.8	41.7	45.1	42.5	42.1	36.2	37.3	37.9	40.6	41.7	38.1	40.3

Although, the MMAS-ET approach does not generate outstanding results its performance is comparable with other approaches. Beside the graph coloring

heuristics of Carter et al. it also finds better solutions than the Wal, the GS, the PS, the Ga and the CT approach for most test cases.

In addition, it is striking that no approach outperforms all other approaches for all test cases. Thus, there are some test cases where MMAS-ET outperforms the approaches Cal, BN and CT, although one must confirm that these three approaches generate better solutions for most of the test cases. For example, MMAS-ET found better solutions than the Ca approach in four out of the 13 test cases, i.e. for the test cases car-f-92,car-s-91, sta-f-83 and tre-s-92.

5.5 Comparison with the approach of Costa and Hertz

Finally, the results of MMAS-ET were compared with a modified version of the ANTCOL algorithm of Costa and Hertz [18], which originally was developed for solving graph coloring problems. This approach will be called ANTCOL-ET in the sequel. Within that approach the ANT_DSATUR(1) procedure was used as a constructive method as described in [18]. The objective function was modified in order to consider conflicts of higher order too. Test runs were carried out to adjust the parameters appropriately. The parameter α was set to 1, β to 35. ρ was set equal to 0.3. Again, each test case was solved twenty times.

Table 4 shows the results for the thirteen test cases and compares them with the MMAS-ET approach. Surprisingly, the simple AS like approach ANTCOL-ET outperformed the MMAS-ET for some test cases. In particular, this result is contrary to other results presented in the literature where MMAS algorithms obtained better results for various combinatorial optimization problems by avoiding stagnation (c.f. [12, 16]).

Thus, ANTCOL-ET was modified by implementing additionally the hill climber already incorporated in the MMAS-ET approach. This modified version of the Costa and Hertz approach provided on the average better solutions than the MMAS-ET approach and

Like the MMAS-ET approach the ANTCOL-ET approach in particular improves the test cases that already achieved the best solutions. For example, it again outperformed the approach of Caramia et al. in the test cases car-f-92, car-s-91, sta-f-83 and tre-s-92. White et al. argued in [11] that these test cases seem to be in a way easier.

Computing times for the MMAS-ET approach lay in the range of 10 seconds for the smallest test cases, i.e. hec-s-92, to 2.5 hours for the pur-s-93 problem. Compared to the MMAS-ET approach the computing time of the ANTCOL-ET combined with the hill climber was on the average 80 % higher. Thus, one can conclude that ANTCOL-ET takes more time but gets a better solution quality than MMAS-ET. Please note that the same stopping stopping criteria was used for both algorithms, namely, 2500 solutions.

6 Conclusion

In this paper different strategies for solving exam timetabling problems were tested. Ant colony approaches are capable of solving large real world exam

Table 4. Comparison between different ant colony approaches.

test case		MMAS-ET	ANTCOL-ET without hill climber	ANTCOL-ET with hill climber
car-f-92	best	4.8	4.5	4.3
	avg.	4.9	4.6	4.4
car-s-91	best	5.7	5.3	5.2
	avg.	5.9	5.4	5.2
ear-f-83	best	36.8	40.3	36.8
	avg.	38.6	41.4	38.3
hec-s-92	best	11.3	12.2	11.1
	avg.	11.5	12.6	11.4
kfu-s-93	best	15.0	15.4	14.5
	avg.	15.5	15.8	14.9
lse-f-91	best	12.1	11.9	11.3
	avg.	12.7	12.2	11.7
pur-s-93	best	5.4	4.8	4.6
	avg.	5.6	4.9	4.6
rye-s-93	best	10.2	10.2	9.8
	avg.	10.4	10.7	10.0
sta-f-83	best	157.2	158.2	157.3
	avg.	157.5	159.3	157.5
tre-s-92	best	8.8	8.8	8.6
	avg.	9.1	9.0	8.7
uta-s-92	best	3.8	3.6	3.5
	avg.	3.8	3.7	3.5
ute-s-92	best	27.7	28.9	26.4
	avg.	28.6	29.4	27.0
yor-f-83	best	39.6	42.2	39.4
	avg.	40.3	43.7	40.4

timetabling problems. The implemented algorithms generated comparable results like other high performance algorithms from the literature.

Unlike for other combinatorial optimization problems like the TSP or the QAP for the exam timetabling problem the MMAS approach did not outperform the simpler AS strategy. Of course, it goes without saying but proper adjusting parameters can improve the performance of an algorithm considerably.

A self-evident extension would be to incorporate additional constraints and requirements like e.g. scarce room resources or precedence constraints between exams.

References

1. M.W. Carter, G. Laporte, and S.Y. Lee. Examination timetabling algorithmic strategies and applications. *Journal of the Operational Research Society*, 47:373–383, 1996.

2. L. Di Gaspero and A. Schaerf. Tabu search techniques for examination timetabling. *Lecture Notes in Computer Science*, 2079:104–117, 2001.
3. J.P. Boufflet and S. Nègere. Three methods used to solve an examination timetable problem. *Lecture Notes in Computer Science*, 1153:327–344, 1996.
4. M.W. Carter and G. Laporte. Recent developments in practical examination timetabling. *Lecture Notes in Computer Science*, 1153:3–21, 1996.
5. E.K. Burke and J. Newall. Enhancing timetable solutions with local search methods. *Lecture Notes in Computer Science*, 2740:195–206, 2003.
6. M. Caramia, P. Dell'Olmo, and G.F. Italiano. New algorithms for examination timetabling. *Lecture Notes in Computer Science*, 982:230–241, 2001.
7. S. Casey and J. Thompson. Grasping the examination scheduling problem. *Lecture Notes in Computer Science*, 2740:233–244, 2003.
8. L. Di Gaspero. Recolour, shake and kick: A recipe for the examination timetabling problem. In *Proceedings of the fourth international conference on the practice and theory of automated timetabling*, pages 404–407, Gent, Belgium, August 2002.
9. L.T.G. Merlot, N. Boland, B.D. Hughes, and P.J. Stuckey. New benchmarks for examination timetabling. Testproblem Database, http://www.or.ms.unimelb.edu.au/timetabling.html.
10. L. Paquete and T. Stuetzle. Empirical analysis of tabu search for the lexicographic optimization of the examination timetabling problem. In *Proceedings of the fourth international conference on the practice and theory of automated timetabling*, pages 413–420, Gent, Belgium, August 2002.
11. G.M. White, B.S. Xie, and S. Zonjic. Using tabu search with long-term memeory and relaxation to create examination timetables. *European Journal of Operational Research*, 153:80–91, 2004.
12. M. Dorigo, G. Di Caro, and L.M. Gambarella. Ant algorithms for discrete optimization. *Artificial Life*, 5:137–172, 1999.
13. M.R. Garey and D.S. Johnson. *Computers and intractability: a guide to the theory of NP-completness*. W. H. Freeman and Company, New York, 1979.
14. V. Lofti and R. Cerveny. A final-exam scheduling package. *Journal of the Operational Research Society*.
15. A. Colorni, M. Dorigo, and V. Maniezzo. Distributed optimization by ant colonies. In *Proceedings of the first european conference on artificial life*, pages 134–142, Amsterdam, 1992. Elsevier Science Publishers.
16. T. Stuetzle and H.H. Hoos. Max-min ant systems. *Future Generation Computer System*, 16:889–914, 2000.
17. K. Socha, M. Sampels, and M. Manfrin. Ant algorithms for the university course timetabling problem with regard to state-of-the-art. In *Proceedings of 3rd European Workshop on Evolutionary Computation in Combinatorial Optimization (EvoCOP'2003)*, pages 334 – 345, Essex, UK, April 2003.
18. D. Costa and A. Hertz. Ants can color graphs. *Journal of the Operational Research Society*, 48:295–305, 1997.
19. K. Dowsland and J. Thompson. Ant colony optimisation for the examination scheduling problem. *Journal of the Operational Research Society*, 56:426–439, 2005.
20. A. Vesel and J. Zerovnik. How well can ants color graphs? *Journal of Computing and Information Technology - CIT*, 8:131–136, 2000.
21. ftp:\\ie.utoronto.ca/pub/carter/testprob.
22. http:\\www.or.ms.unimelb.edu.au/timetabling/ttexp2.html.