A multiple metaheuristic variable neighborhood search framework for the Uncapacitated Examination Timetabling Problem

Panayiotis Alefragis · Christos Gogos · Christos Valouxis · Efthymios Housos

Received: date / Accepted: date

Abstract The paper describes a framework that was developed to solve the Uncapacitated Examination Timetabling Problem. It also presents a way of reducing problem sizes by removing students and examinations that ultimately have no impact. Moreover, it presents some loose lower bounds that are computed for reference problems. The framework allows the collaboration of multiple metaheuristic algorithms. A common problem and solution representation is created. Multiple evaluators are available and a mechanism to select variable neighborhoods is implemented. A number of simple and complex neighborhoods is created. The application of the framework creates competitive results compared to the best ones available in the literature for the Toronto-b dataset.

Keywords uncapacitated examination timetabling · metaheuristics · framework

1 Introduction

The examination timetabling problem in general involves assigning examinations to a limited number of periods and rooms while respecting a set of hard

Panayiotis Alefragis
Electrical & Computer Engineering Department, University of Peloponnese, Patras, Greece
E-mail: alefrag@uop.gr

Christos Gogos
Informatics & Telecommunications Department, University of Ioannina, Arta, Greece
E-mail: cgogos@uoi.gr

Christos Valouxis
Electrical & Computer Engineering Department, University of Patras, Patras, Greece
E-mail: valouxis@ece.upatras.gr

Efthymios Housos
Electrical & Computer Engineering Department, University of Patras, Patras, Greece
E-mail: housos@ece.upatras.gr

constraints and at the same time trying to minimize the violation of soft constraints. The Uncapacitated Examination Timetabling Problem (UETP) does not consider room capacity requirements. The only hard constraint is that it is not allowed for a student to sit on two examinations at the same period. On the other hand, the only soft constraint is that of spreading examinations as evenly as possible from the student's perspective. A formal description of the problem as an Integer Programming problem follows: In this problem setting, C is the set of courses, P is the set of periods and S is the set of students. Each course c has a number of enrollments which is captured at E_c. In sync, each student s has a set of courses C_s that he is enrolled to. The set CNF contains triplets of the form c_1, c_2, co which represent the situation in which course c_1 and course c_2 have co in common students.

In order to define for each course the period that it will be scheduled the decision variables m_{cp} are used for each $c \in C$ and $p \in 1..P$ which are binary variables assuming value 1 if the course c is scheduled to period p or 0 otherwise. Moreover, the binary variables $y1_{t,p}$ are defined for each $t \in CNF$ and $p \in 1..|P-1|$ assuming value 1 when courses $t.c_1$ and $t.c_2$ are scheduled in consecutive periods and 0 otherwise. Likewise, binary variables $y2_{t,p}, y3_{t,p}, y4_{t,p}, y5_{t,p}$ are defined and assume value 1 when courses $t.c_1$ and $t.c_2$ are scheduled to periods with distance between 2 and 5 periods respectively.

The objective function is presented in Equation 1

$$\begin{aligned} minimize \sum_{t \in CNF} t.co\big(&16 \sum_{p \in 1...|P-1|} y1_{t,p} \\ &+8 \sum_{p \in 1...|P-2|} y2_{t,p} \\ &+4 \sum_{p \in 1...|P-3|} y3_{t,p} \\ &+2 \sum_{p \in 1...|P-4|} y4_{t,p} \\ &+ \sum_{p \in 1...|P-5|} y5_{t,p}\big) \end{aligned} \tag{1}$$

The first constraint states that each course should be scheduled at exactly one period and is captured in equation 2.

$$\sum_{p \in P} m_{c,p} = 1 \quad \forall c \in C \tag{2}$$

The second constraint shown in equation 3 prohibits every possible pair of courses having common students to be scheduled at the same period.

$$m_{t.c_1,p} + m_{t.c_2,p} \leq 1 \quad \forall p \in P, \forall t \in CNF \tag{3}$$

The third set of constraints equations 4 to 8 defines variables $y1_t$ to $y5_t$.

$$\left.\begin{aligned} y1_{t,p} &\geq m_{t.c_1,p} + m_{t.c_2,p+1} - 1 \\ y1_{t.p} &\geq m_{t.c_2,p} + m_{t.c_1,p+1} - 1 \end{aligned}\right\} \forall t \in CNF, \forall p \in 1 \ldots |P| - 1 \tag{4}$$

$$\left.\begin{array}{l} y2_{t,p} \geq m_{t.c_1,p} + m_{t.c_2,p+2} - 1 \\ y2_{t,p} \geq m_{t.c_2,p} + m_{t.c_1,p+2} - 1 \end{array}\right\} \forall t \in CNF, \forall p \in 1 \ldots |P| - 2 \quad (5)$$

$$\left.\begin{array}{l} y3_{t,p} \geq m_{t.c_1,p} + m_{t.c_2,p+3} - 1 \\ y3_{t,p} \geq m_{t.c_2,p} + m_{t.c_1,p+3} - 1 \end{array}\right\} \forall t \in CNF, \forall p \in 1 \ldots |P| - 3 \quad (6)$$

$$\left.\begin{array}{l} y4_{t,p} \geq m_{t.c_1,p} + m_{t.c_2,p+4} - 1 \\ y4_{t,p} \geq m_{t.c_2,p} + m_{t.c_1,p+4} - 1 \end{array}\right\} \forall t \in CNF, \forall p \in 1 \ldots |P| - 4 \quad (7)$$

$$\left.\begin{array}{l} y5_{t,p} \geq m_{t.c_1,p} + m_{t.c_2,p+5} - 1 \\ y5_{t,p} \geq m_{t.c_2,p} + m_{t.c_1,p+5} - 1 \end{array}\right\} \forall t \in CNF, \forall p \in 1 \ldots |P| - 5 \quad (8)$$

When courses c_1 and c_2 have distance 1 either $m_{t.c_1,p}$ and $m_{t.c_2,p+1}$ or $m_{t.c_2,p}$ and $m_{t.c_2,p+1}$ assume value 1. If this is the case variable $y1_t$ will be 1. If at least one of $m_{t.c_1,p}$ or $m_{t.c_2,p+1}$ are 0 the value of $y1_t$ could be 0 or 1 but since it is included in the objective which is minimized it will take the value 0.

This paper describes a framework that was developed to solve the UETP. The most studied dataset was introduced by Carter [14]. Characteristics of the used problem instances are presented in Table 1. The specific dataset is heavily studied, but it is still possible to generate improved solutions compared to the ones published in literature. The specific dataset has the property that while it has very low learning curve and relative simple structure it is still challenging.

Table 1 Dataset Problem characteristics (Toronto-b [22])

Problem	Exams	Students	Admissions	Density	Slots
CAR91	682	16925	56877	0.13	35
CAR92	543	18419	55522	0.14	32
EAR83	190	1125	8109	0.27	24
HEC92	81	2823	10632	0.42	18
KFU93	461	5349	25113	0.06	20
LSE91	381	2726	10918	0.06	18
PUR93	2419	30029	120681	0.03	42
RYE92	486	11483	45051	0.07	23
STA83	139	611	5751	0.14	13
TRE92	261	4360	14901	0.18	23
UTA92	622	21266	58979	0.13	35
UTE92	184	2749	11793	0.08	10
YOR83	181	941	6034	0.29	21

The paper is organized as follows. Section 2 presents related work, section 3 presents some analysis on the dataset instances, section 4 presents a high level framework design and describes the metaheuristic algorithms and the neighborhoods supported by the framework, section 5 presents generated results and finally section 6 presents conclusions and future work.

2 Related Work

Several approaches have been proposed for solving the UETP. The datasets introduced by [14], collectively known as Toronto datasets, have been typically used as a testbed for demonstrating the effectiveness of new approaches. So, a great number of published results exist over those datasets, showing that approximation methods manage to produce better results, when compared to exact methods. This might occur due to the large size of the problem instances and the symmetries among several different schedules that can be produced. Both these factors prohibit Integer Programming solvers, Constraint Programming solvers and SAT solvers to solve the full problem behind each problem instance.

A non-exhaustive list of approaches that have given good results follows. Algorithms based on local search were proposed in [13]. Several hybridizations of metaheuristics with sequential heuristics were used in [24]. The flex-deluge algorithm and the late acceptance strategy for solving UETP where proposed in [8] and [9] respectively. A hybrid variable neighborhood search was used in [12]. In [15] the problem was approached using hyperheuristics. In [4] the problem was solved using a hybrid bee colony optimization method. Another, recent hybrid approach that combines a cellular memetic algorithm with the threshold acceptance metaheuristic is described in [20].

It should be noted that the survey paper in [22] presents many details about the examination timetabling problem in general and in particular about the UETP. A review paper focusing on UETP recently appeared in [2].

3 Some thoughts on the problem and a few loose lower bounds

The simplicity of describing the UETP problem is at odds with the hardness of optimally solving it. This situation strikes resemblance with the iconic combinatorial optimization problem, Traveling Salesman Problem (TSP) [7]. For TSP, large problem instances have proven optimal solutions, while several solvers exist capable of obtaining optimal solutions for TSP instances of moderate to large sizes. In particular, the freely available Concorde TSP solver [5] has the record of obtaining optimal solutions for all TSPLIB [23] instances. It is impressive that Concorde solver can optimally solve the largest TSPLIB instance, counting 85,900 cities, while back in 1962, a problem with only 33 cities was considered very difficult to be solved and a competition was hosted by Procter & Gamble in order to solve it. Moreover, significant theoretical work has been gradually done for TSP in order to provide solutions within a guaranteed distance from the optimal ones or in order to find lower bounds (e.g. Christofides algorithm, 1-tree Lagrangean relaxation). So, new solutions can be quantified according to how close they are to the theoretically optimal ones. For example, the world tour TSP problem with 1,904,711 locations has been solved with total length of only 0.0474% greater than the length of the theoretically optimal tour. Unfortunately, UETP has not achieved the level of

maturity that TSP has reached. There are several approaches that give good results, but this is mostly justified by comparing them with results obtained for the same problem instances by other researchers. The popularity of TSP, the longest time that TSP has been at the center of interest for the scientific community, its adaptation to numerous practical combinatorial optimization problems (e.g. transportation problems, printed circuits, logistics and others) might be among the reasons why research on TSP is much more mature than in UETP.

By simple analyzing the data of the Toronto datasets a few remarks can be made. Firstly, if there are students that participate in only one examination, then their presence is irrelevant to the solution of the problem. We call these students "noise students", since they have no contribution to the cost of the final schedule. Secondly, if there are courses that have only "noise students" then these courses can also be safely removed from the problem, since they can be scheduled at any period without affecting the cost of the schedule. We call these courses "noise courses". It turns out "noise" students and courses exist for the Toronto datasets. Relevant results are displayed in Table 2 and suggest that new equivalent versions of the datasets might be constructed by omitting the "noise" students and courses. The rightmost column of the Table displays the actual identification numbers of the "noise" courses that are used at the corresponding datasets.

Table 2 Carter datasets "Noise"

Problem	#"Noise" Students	#"Noise" Courses	Set of "Noise" Courses
CAR91	3409	4	{0033, 0349, 0440, 0657}
CAR92	3969	1	{0519}
EAR83	1	0	Ø
HEC92	321	0	Ø
KFU93	276	17	{0006, 0016, 0022, 0050, 0095, 0122, 0178, 0204, 0205, 0216, 0285, 0329, 0330, 0355, 0369, 0381, 0443}
LSE91	99	2	{0168, 0256}
PUR93	2627	6	{0153, 0552, 0976, 0983, 1454, 1520}
RYE92	2025	1	{0304}
STA83	0	0	Ø
TRE92	667	1	{0186}
UTA92	6180	0	Ø
UTE92	78	0	Ø
YOR83	1	0	Ø

3.1 Loose lower bounds

Bellow, an idea for calculating some loose lower bounds for the UECP problem instances is described. Each student has a set of courses he is enrolled to. These courses should be scheduled to different periods and each assignment gives an associated cost based on the proximity among the exams. The possible permutations of scheduling E exams over P periods are $\frac{P!}{(P-E)!}$. For each permutation the associated cost is easy to be computed. Since the total number of permutations might be prohibitively large in order to enumerate all permutations and compute its cost, one can notice that the number of examinations that can be scheduled with zero cost in P periods are $L = \lfloor \frac{P-1}{6} \rfloor + 1$. So, when the number of examinations a student is enrolled into is no more than L then the theoretical best cost contribution that can be achieved for this student is zero. Else, a simple optimization problem can be solved that finds the minimum cost of positioning the E exams of a single student in P periods. A lower bound to the problem can be identified when the examinations of each student are scheduled to the exams into the period permutation with the minimum cost. Such an arrangement might not be possible since it does not consider the fact that examinations are common between students and each examination should be scheduled to the same period for all students. Nevertheless, it provides a lower bound for the problem instances. These bounds, alongside with the maximum number of courses that a student is enrolled to are presented in Table 3.

Table 3 Carter dataset lower bounds

Problem	Periods	Courses per Student (max)	Lower Bound
CAR91	35	9	0.01
CAR92	32	7	0.01
EAR83	24	10	17.85
HEC92	18	7	3.49
KFU93	20	8	5.63
LSE91	18	8	2.76
PUR93	42	9	N/A
RYE92	23	10	3.79
STA83	13	11	105.92
TRE92	23	6	0.59
UTA92	35	7	N/A
UTE92	10	6	19.25
YOR83	21	14	18.50

For two of the problem instances, STA83 and UTE92 the lower bounds are relatively close to the best values obtained in the literature. For two other problem instances, PUR93 and UTA92 no lower bound can be found in this way. Following the procedure that was described above the computed value becomes zero. This occurs because there are enough timeslots that allow even

for students that have enrolled to the maximum number of courses to have their examinations spread with no mandatory penalty.

4 The multi metaheuristic VNS framework

4.1 The multi heuristic strategy

The proposed framework which is an extension of Variable Neighborhood Search (VNS) [21] uses a number of basic metaheuristic algorithms that are orchestrated using a promising solution that has already been found from some other algorithm and each one tries to improve it and pass the result to the next one. Each metaheuristic algorithm uses a number of neighborhoods, called moves, to select the next candidate solution. Simple moves are randomly selected, while complex moves are triggered when no progress is achieved using just simple moves. Table presents the names of the used algorithms.

4.2 Supported metaheuristic algorithms

4.2.1 Simulated Annealing

Simulated Annealing (SA) was proposed by Kirkpatrick et al. [18]. It is a stochastic metaheuristic algorithm, which accepts inferior quality candidate solutions with probability $P = e^{(I-C)/T}$, where C and I are the cost of the incumbent and the candidate solutions respectively and T is a control parameter called "temperature". The application of the algorithm to solve the UETP is heavily studied. In our implementation, we have developed a number of alternative cooling schemes and reheating mechanisms.

4.2.2 Late Acceptance Hill Climbing

The Late Acceptance Hill-Climbing (LAHC) [11] is a local search metaheuristic algorithm, which may accept inferior candidate solutions if their cost is better than the solution accepted k iterations before. The algorithm maintains a cyclic buffer of the cost of the last k iterations. The performance of the algorithm is mainly effected by the value of the k parameter. The algorithm implementation in the framework adapts the value of k depending on the progress of the last execution of the LAHC algorithm. At the beginning of the search strategy a relative small value of k is used to promote the algorithm to mainly accept improving solutions. When no significant progress is achieved, the value of k is progressively increased.

4.2.3 Flex Deluge

The Flex Deluge (FD) [10] is a local search metaheuristic algorithm that is a variant of the Great Deluge (GD) algorithm. The GD algorithm may accept

inferior candidate solutions if their cost is better than an upper bound denoted by B. The FD algorithm has an additional k_f parameter that denotes the degree of its flexibility. A k_f value of 0 turns the algorithm to a Hill Climbing algorithm while a value of 1 makes the algorithm identical to the Great Deluge algorithm. The algorithm accepts a new candidate solution with cost Z if $Z < C + k_f(B - C)$ where C is the cost of the current solution.

4.3 Supported neighborhoods

The proposed framework support multiple neighborhoods that the active metaheuristic algorithms select in each iteration to generate a candidate solution. We will call these neighborhoods "moves". The implemented moves are grouped in two groups. The first group contains simple moves that perform small transformations to the incumbent solution. The second group contains more elaborate changes to the incumbent solution and may perform a relatively large local search before proposing a new candidate solution, making them more computationally expensive. The moves are presented in Table 4.

Table 4 Search space exploration

Key	Simple Moves	Key	Complex Moves
S1	BestKempeChain	C1	PenaltyDecreaserMove
S2	RandomKempeChain	C2	RuinAndRecreateMove
S3	MoveWorstExam	C3	SaturationImproverMove
S4	ForceSingleExamAtBest	C4	CyclicExchangeMove
S5	SingleExamAtBestSlot	C5	TimeRelaxerMove
S6	ExchangeExams		
S7	Move Single Exam		

4.3.1 Random and Best Kempe Chain

The first available neighborhood structures uses the concept of Kempe Chains (KCs), initially used to solve the "four color problem". Given the two sets of exams assigned to two periods, a number of chains consisting of exams belonging to either one of the periods is constructed. When an exam of a chain is moved from one period to the other then all other exams of the same chain are also moved to maintain feasibility. Initially, both moves randomly selects two slots and generates all possible KCs between these slots. Then the Random KC move shuffles the generates KCs and sequentially evaluates the difference in the cost, if the move was applied. If it finds a KC that produce a better solution than the incumbent one, it stops the process and returns the selected KC to be applied. If no improving KC is found, a random one is returned. The Best KC move on the other hand always returns the KC that generates the best reduction in the solution cost, among the generated KCs.

4.3.2 Move Worst Exam

The second available neighborhood structure ranks each exam based on the individual contribution to the overall cost. The top k exams with the worst contributions are then put in a squeaky wheel random selection scheme and one of them is selected. The move then tries to find the best legal slot to move the exam.

4.3.3 (Forced) Single Exam at Best Slot

The third available neighborhood structures randomly select an exam and then tries to find the best legal slot to move the exam. The forced version tries all possible slots, even the illegal ones, and calculates the benefit of moving the exam to the specific slot. Then, if the best slot is illegal as exams with conflicting students are already assigned to this slot, it moves the exam to the best slot and then tries to find a new feasible placement for all the conflicting exams.

4.3.4 Move Single / Exchange Exams

The fourth simple available neighborhood structures randomly selects an exam and moves it to the first feasible slot it finds. The exchange move randomly selects a non conflicting exam from the selected slot and moves this exam to the initial slot of the first exam.

4.3.5 Penalty Decreaser

The first of the complex moves either tries to optimize first very costly exams or badly placed exams. For each exam it finds the available legal slots that can be moved. It then calculates for all legal moves the benefit that each move will have to the incumbent solution. Based on the calculated benefit it select f out of all exam moves from the most beneficial to the least one. It then generates up to $k \times f$ combinations out of the selected ones and generate application sets of exam slot pairs that if applied will generate a legal assignment. Finally, it evaluates the generated sets and select to apply the best one. For our experiments, k had a value of 30 and f a value of 5. The move requires a lot of computation time, but is usually able to generate a much better solution compared to the incumbent one.

4.3.6 Ruin & Recreate

The second of the complex moves initially removes a set of exams, the ruin phase. One of the two available methods is randomly used to select exams. The first method selects a fixed percentage of the exams in the problem while the second uses a knapsack inspired method and removes exams that their individual contribution to the incumbent solution cost is used as the weight.

In the second phase, the recreate one, the removed exams are placed in slots sequentially, using a randomly selected method out of the three available . The first method, randomly select one unscheduled exam and places it to the best legal available slot, the second method to the first feasible slot it can find while the last one randomly select one the previous methods for each individual exam. The recreate phase placements may lead to an infeasible solution, as no back tracking is performed when placing unscheduled exams to slots. The move has a very strong effect, if successful, to help algorithms escape from local minimum.

4.3.7 Saturation Improver

The third of the complex moves initially calculates for all exams the available legal slots that each exam can be assigned and then sorts the exams from the most constrained to the least ones. It then tries to find for the most constrained ones if reassigning some of the exams that that have common students to other slots improves the number of the available slots that the currently selected exam can be moved. It is important to note that the reassignments are only allowed if no significant change in the cost of the incumbent solution is performed. For our experiments, we allowed up to 1% total reduction in the quality of the solution. The idea behind the move is to allow exams that due to the placement of other conflicting exams are unable to move to provide new possible slots, helping the algorithms escape from specific areas in the solution space. The move usually generate worse quality but different solutions compared to the incumbent one.

4.3.8 Cyclic Exchange

The fourth of the complex moves randomly selects a slot. It then performs the best Kempe Chain exchange with the next slot and this continues until the last available slot. When we reach the last slot, the exchange is performed between the last slot and the first slot. The exchange sequence is continued with the second slot until the initially selected slot is reached. The idea behind this move is that exams that are placed at the first or the last slots due to the cost structure have a benefit as their individual contribution to cost is calculated with only the exams that are on their right for the exams placed on the first slots and on their left for the exams placed on the last slots. With this move, we promote exams with high cost to move to the edges of the slot periods.

4.3.9 Time Relaxer

The last of the complex moves is inspired by the penalty trader move in [13]. We allow up to d slots for an exam to be moved forward, even if this slot is not available and we calculate the benefit in the cost that this move would have. All exams that have moves that would be beneficial are added to a candidate

list and the list is sorted in an ascending way. We then select the exam with the highest benefit and remove all other exams that are in the candidate list and have common students with the selected exam. The selected exam is added to a new list with non conflicting exams. The process continues until no more exams are available in the candidate list. As the non conflicting list contains exams that are compatible and can be scheduled on the same slot, the best slot for all these exams is selected. The idea behind the move is that it clusters compatible exams that individually may not have a big benefit in the cost but collectively may provide a better candidate solution.

5 Experimental results

Best results achieved using all the metaheuristic algorithms and all the available moves in the framework alongside some of the best results published for the same dataset are presented in Table 5. The results from our framework are in the column MAVN, while results from other heuristic method are from a tabu search method implemented by Di Gaspero and Schaerf[16], a graph coloring method proposed by Pillar and Allende[17], the sequential heuristic construction methods developed by Caramia et al.[13], the Ahuja-Orlin large neighbourhood search by Abdullah et al.[1], the Iterative Restart Variable Neighbourhood Search by Ayob et al.[6] and the Flex-Deluge algorithm developed by Burke and Bykov[10].

Table 5 The best results obtained by MAVN and other optimization methods applied to the Carter datasets

Dataset	MAVN	Di Gaspero & Schaerf	Allende et al.	Caramia et al.	Abdullah et al.	Ayob et al.	Burke & Bykov
CAR91	4.58	6.2	4.39	6.6	5.21	4.90	**4.32**
CAR92	3.80	5.2	3.71	6.0	4.36	4.51	**3.67**
EAR83	32.67	45.7	32.62	**29.3**	34.84	36.28	32.62
HEC92	10.03	12.4	10.05	**9.2**	10.28	11.06	10.06
KFU93	12.87	18.0	12.90	13.8	13.46	14.74	**12.80**
LSE91	10.02	15.5	9.82	**9.6**	10.24	12.08	9.78
PUR93	4.46	-	-	**3.7**	-	4.66	3.88
RYE93	8.08	-	-	**6.8**	8.74	10.67	7.91
STA83	**157.03**	160.8	**157.03**	158.2	159.28	157.32	**157.03**
TRE92	7.87	10.0	7.71	9.4	8.13	8.92	**7.64**
UTA92	3.18	4.2	3.04	3.5	3.63	3.58	**2.98**
UTE92	24.77	29.0	24.77	**24.4**	24.21	26.36	24.78
YOR83	35.11	41.0	34.70	36.2	36.11	38.97	**34.71**

6 Discussion and future work

In this paper, a framework to solve the UETP is presented. Using the framework, it is possible to experiment with the application of different algorithms and different neighborhoods very easily. In addition, the design of the framework allows the application of different evaluation methods that can use hardware accelerators [19]. Algorithmic parameters and the choice of moves that are active have a significant impact in the quality of the generated solutions. An initial experimentation for offline parameter selection can be found in [3]. In the future, we will investigate the use of online machine learning techniques to improve the sequence of the application of the available algorithms and the dynamic selection of neighborhoods.

References

1. Abdullah, S., Ahmadi, S., Burke, E.K., Dror, M.: Investigating Ahuja–Orlin's large neighbourhood search approach for examination timetabling. OR Spectrum **29**(2), 351–372 (2007)
2. Aldeeb, B.A., Al-Betar, M.A., Abdelmajeed, A.O., Younes, M.J., AlKenani, M., Alomoush, W., Alissa, K.A., Alqahtani, M.A.: A comprehensive review of uncapacitated university examination timetabling problem. International Journal of Applied Engineering Research **14**(24), 4524–4547 (2019)
3. Alefragis, P., Sofos, C.: Automated parameter selection of scheduling algorithms using machine learning techniques. In: Proceedings of the 21st Pan-Hellenic Conference on Informatics, pp. 1–5 (2017)
4. Alzaqebah, M., Abdullah, S.: Hybrid bee colony optimization for examination timetabling problems. Computers & Operations Research **54**, 142–154 (2015)
5. Applegate, D., Bixby, R., Chvatal, V., Cook, W.: Concorde TSP solver (2006). URL http://www.tsp.gatech.edu
6. Ayob, M., Burke, E.K., Kendall, G.: An iterative re-start variable neighbourhood search for the examination timetabling problem. Practice and Theory of Automated Timetabling. LNCS **1153**, 336–344 (2006)
7. Bellmore, M., Nemhauser, G.L.: The traveling salesman problem: a survey. Operations Research **16**(3), 538–558 (1968)
8. Burke, E.K., Bykov, Y.: Solving exam timetabling problems with the flex-deluge algorithm. In: Proceedings of PATAT, vol. 2006, pp. 370–372. Citeseer (2006)
9. Burke, E.K., Bykov, Y.: A late acceptance strategy in hill-climbing for exam timetabling problems. In: PATAT 2008 Conference, Montreal, Canada, pp. 1–7 (2008)
10. Burke, E.K., Bykov, Y.: An adaptive flex-deluge approach to university exam timetabling. INFORMS Journal on Computing **28**(4), 781–794 (2016)
11. Burke, E.K., Bykov, Y.: The late acceptance hill-climbing heuristic. European Journal of Operational Research **258**(1), 70–78 (2017)
12. Burke, E.K., Eckersley, A.J., McCollum, B., Petrovic, S., Qu, R.: Hybrid variable neighbourhood approaches to university exam timetabling. European Journal of Operational Research **206**(1), 46–53 (2010)
13. Caramia, M., Dell'Olmo, P., Italiano, G.F.: New algorithms for examination timetabling. In: International Workshop on Algorithm Engineering, pp. 230–241. Springer (2000)
14. Carter, M.W., Laporte, G., Lee, S.Y.: Examination timetabling: Algorithmic strategies and applications. Journal of the Operational Research Society **47**(3), 373–383 (1996)
15. Demeester, P., Bilgin, B., De Causmaecker, P., Berghe, G.V.: A hyperheuristic approach to examination timetabling problems: benchmarks and a new problem from practice. Journal of Scheduling **15**(1), 83–103 (2012)

16. Di Gaspero, L., Schaerf, A.: Tabu search techniques for examination timetabling. In: International Conference on the Practice and Theory of Automated Timetabling, pp. 104–117. Springer (2000)
17. Díaz, P.M., Allende, J.S.: Aproximación al problema de calendarios de exámenes mediante coloreado de grafos. Tecnología y desarrollo **15** (2017)
18. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. science **220**(4598), 671–680 (1983)
19. Kolonias, V., Goulas, G., Gogos, C., Alefragis, P., Housos, E.: Solving the examination timetabling problem in GPUs. Algorithms **7**(3), 295–327 (2014)
20. Leite, N., Fernandes, C.M., Melicio, F., Rosa, A.C.: A cellular memetic algorithm for the examination timetabling problem. Computers & Operations Research **94**, 118–138 (2018)
21. Mladenović, N., Hansen, P.: Variable neighborhood search. Computers & operations research **24**(11), 1097–1100 (1997)
22. Qu, R., Burke, E.K., McCollum, B., Merlot, L.T., Lee, S.Y.: A survey of search methodologies and automated system development for examination timetabling. Journal of scheduling **12**(1), 55–89 (2009)
23. Reinelt, G.: TSPLIB — A traveling salesman problem library. ORSA journal on computing **3**(4), 376–384 (1991)
24. Yang, Y., Petrovic, S.: A novel similarity measure for heuristic selection in examination timetabling. In: International Conference on the Practice and Theory of Automated Timetabling, pp. 247–269. Springer (2004)