1-2011

Solving the teacher assignment problem by two Metaheuristics

Aldy GUNAWAN
Singapore Management University, aldygunawan@smu.edu.sg

Kien Ming NG

Citation

GUNAWAN, Aldy and NG, Kien Ming. Solving the teacher assignment problem by two Metaheuristics. (2011). *International Journal of Information and Management Sciences*. 22, (1), 73-86.
Available at: https://ink.library.smu.edu.sg/sis_research/4003

International Journal of
Information
and
Management
Sciences

International Journal of Information and Management Sciences
22 (2011), 73-86

Solving the Teacher Assignment Problem by Two Metaheuristics

Aldy Gunawan and Kien Ming Ng

Singapore Management University and National University of Singapore

Abstract

The timetabling problem arising from a university in Indonesia is addressed in this paper. It involves the assignment of teachers to the courses and course sections. We formulate the problem as a mathematical programming model. Two different algorithms, mainly based on simulated annealing (SA) and tabu search (TS) algorithms, are proposed for solving the problem. The proposed algorithms consist of two phases. The first phase involves allocating the teachers to the courses and determining the number of courses to be assigned to each teacher. The second phase involves assigning the teachers to the course sections in order to balance the teachers' load. The performance of the proposed algorithms is evaluated using two sets of real data and some randomly generated problem instances. The computational results show that in general, tabu search performs better than simulated annealing and other previous work. For the real data sets, the computational results show that both proposed algorithms yield better solutions when compared to manual allocation done by the university.

Keywords: Timetabling problem, teacher assignment, simulated annealing, Tabu search.

1. Introduction

Timetabling is the allocation, subject to constraints, of given resources to objects being placed in space time, in such a way as to satisfy as nearly as possible a set of desirable objectives [41]. The timetabling problem has attracted substantial research interests due to its importance in a wide variety of application domains, including education [4], transport [27], employee/staff [33], healthcare institutions [2, 5, 22] and sport [36].

Various methods have been proposed for solving this class of problems in the literature, such as those mentioned in the reviews by Carter and Laporte [9], Burke et al. [6], de Werra [15] and Schaerf [34]. In particular, metaheuristics have been widely applied in the timetabling problem, and some examples include simulated annealing [23, 32], tabu search [7, 14, 38] and genetic algorithm [8, 24, 29, 42].

The course timetabling problem in educational timetabling can be classified into five different sub-problems: course scheduling, class-teacher timetabling, student scheduling, teacher assignment, and classroom assignment [9]. This paper focuses primarily on the teacher assignment problem.

Andrew and Collins [1] developed a procedure for assigning teachers to courses based on a simple linear programming technique. Tillett [37] noted that the model does not take into consideration some factors, such as the number of courses taught by each teacher, and proposed a zero-one integer programming algorithm for the teacher assignment problem in the secondary school level. Breslaw [3] highlighted a major shortcoming of the model proposed by Tillett [37], namely that the computation time of the algorithm can be prohibitively large as the problem size increases, and he also proposed a model that could be applicable at the university level. Goal programming has been used by Harwood and Lawless [25] in order to solve the teacher assignment problem, with improvements to the model being made by Schniederjans and Kim [35]. Metaheuristics such as the genetic algorithm were also used to solve the teacher assignment problem, as illustrated in Wang [40]. Votre [39] developed a tool for scheduling teachers to courses given their interest in lecturing.

The major contributions/highlights of this paper are as follows:

(1) The problem addressed in this paper allows each course section to be taught by more than one teacher. Also, the definition of the teachers' load in this paper is different from other models in the literature, where it is referred to as the number of courses taught. Instead, our model assumes that load is related to the number of lecture hours of the course needed per week. Moreover, the teachers could be full time or part time teachers, with different respective requirements on their teaching assignments. All these differences cause the teacher assignment problem to be more complicated and difficult to solve.

(2) We create a mathematical model that is capable of representing the problem. We also develop methods/algorithms for finding the optimal/near-optimal solutions of the problem as well as to help users make better decisions.

In Section 2, a detailed description of the problem is presented. The proposed algorithms for solving the problem, with a detailed description of how the two metaheuristics, simulated annealing (SA) and tabu search (TS), were being used are explained in Section 3. The results of computational experiments involving the proposed algorithms are also provided in Section 4, and some conclusions are given in the last section.

2. Description of the Teacher Assignment Problem

The details of the problem can be found in Gunawan et al. [24]. In this section, we only give a short description of the timetabling problem addressed in this paper. This timetabling problem arises in the Industrial Engineering Department of a university in Indonesia.

In this problem, a course refers to a subject taught one or more times within a week, and each course has a certain number of credits, where a credit refers to the number of lecture hours needed per week. Due to the capacity of the class and the number of students registered, most of the courses have to be taught repeatedly by the same or different teachers at the same or different time periods. Each of these repeated courses is called a section of the course.

The teachers are divided into full time and part time teachers. Some courses could be taught jointly by part time and full time teachers. Part time teachers' requirements, such as the course can only be taught during certain time periods, have to be satisfied. Full time teachers are further divided into Group 1 and Group 2 teachers. Group 2 teachers refer to those who are holding some key administrative positions such as the Vice Dean, Dean, or Chairperson, and hence have to be assigned a lighter teaching load, while Group 1 teachers are other full time teachers who do not hold any key administrative position.

In this study, we focus on courses which can be taught by either full time teachers only, or taught jointly by full time and part time teachers. The allocation of part time teachers to courses and course sections will be performed before the allocation of full time teachers, and this is known as pre-assignment.

The teacher assignment problem involves assigning and scheduling the teachers to the courses by taking some factors, such as the abilities of the teachers and number of courses offered, into consideration. The aim is to balance the full time teachers' load, where the load refers to the total number of credits assigned to each teacher. For example, if a 3-credit course is assigned to two teachers, the total load of each teacher will be incremented by 1.5 credits.

The requirements imposed on the problem are listed as follows:

- Each course section has a minimum and maximum number of teachers to teach (*Requirement* 1).
- Part time teachers are required to teach courses that cannot be taught by full time teachers (*Requirement* 2). Due to internal regulations, certain part time teachers, especially those from government universities, have to be involved in the teaching process.
- There are both a maximum and minimum number of teachers that can be assigned to a particular course (*Requirement* 3). This depends on the number of course sections offered.
- The total number of different courses taught by each full time teacher should not exceed a certain number (*Requirement* 4). The aim of this constraint is to reduce the amount of preparation time for each teacher.
- The teaching load for full time teachers should be balanced (*Requirement* 5). The part time teachers' load will not be considered by the department offering the courses.

The first four requirements are regarded as hard constraints that need to be satisfied, while the last one is treated as a soft constraint or preference that is considered to be equivalent to minimizing the total weighted variance of the teachers' load. Currently, the teacher assignment has always been done manually. This manual assignment procedure has gradually become more difficult and unmanageable due to an increase in the number of courses offered and the number of students.

3. The proposed algorithm for teacher assignment

The teacher assignment problem is formulated as a mathematical programming model. The details of the mathematical model and its description can be found in Gunawan et al. [24]. As described in that paper, the mathematical programming model has non-linear objective function and constraints. Solving this model to optimality may not be straightforward, especially when large-scale data sets are involved. In this section we present a two-phase algorithm devised for tackling this teacher assignment problem. The details of the algorithm are presented in Figure 1.

Phase 1 aims to find a feasible solution of good quality solution quickly. It is motivated by the following reasons. The total variance of teaching load can be minimized if the number of courses taught by each teacher, denoted by R, is restricted to a certain value. It means that the difference in terms of the number of courses taught among teachers is small. It would then be easier to allocate teachers to course sections in the next phase in order to get a small value of the total variance. In other words, Phase 1 focuses on the allocation of full time teachers to the courses based on their abilities. In this phase, we try to find the maximum number of courses that could be allocated to each full time teacher in such a way that no teacher would teach more than R, a maximum number of courses that could be taught by each full time teacher. After we have determined the value of R, we continue to allocate the full time teachers to the course sections in Phase 2. Our main objective in Phase 2 is to minimize the total weighted variance of the teachers' load as mentioned in Equation (1).

In the entire procedure, we could use either SA or TS for the metaheuristic applied in each phase. Thus, we name the procedure as the SA algorithm or the TS algorithm, respectively, depending on whether SA or TS is being applied. The details involving their implementation are described next:

3.1. Simulated annealing

SA, originally developed by Kirkpatrick et al. [26], is a type of local-search heuristic algorithm that avoids getting trapped at a local minimum by accepting neighborhood moves that increase the objective function value, using a probabilistic acceptance criterion. SA has been successfully applied to a variety of combinatorial optimization problems, such as the Traveling Salesman Problem [10], machine scheduling problem [30], DNA sequence alignment [12] and timetabling problem [23].

Several cooling schedules used in SA have been proposed in the literature [32]. In this paper, we use the geometric cooling schedule similar to that used in Gunawan et al. [23], Liu and Ong [28], and Saleh et al. [32]. We also reset the number of iterations to zero if the percentage of accepted moves in the last neighborhood move is greater than some predetermined level *min_percent*.

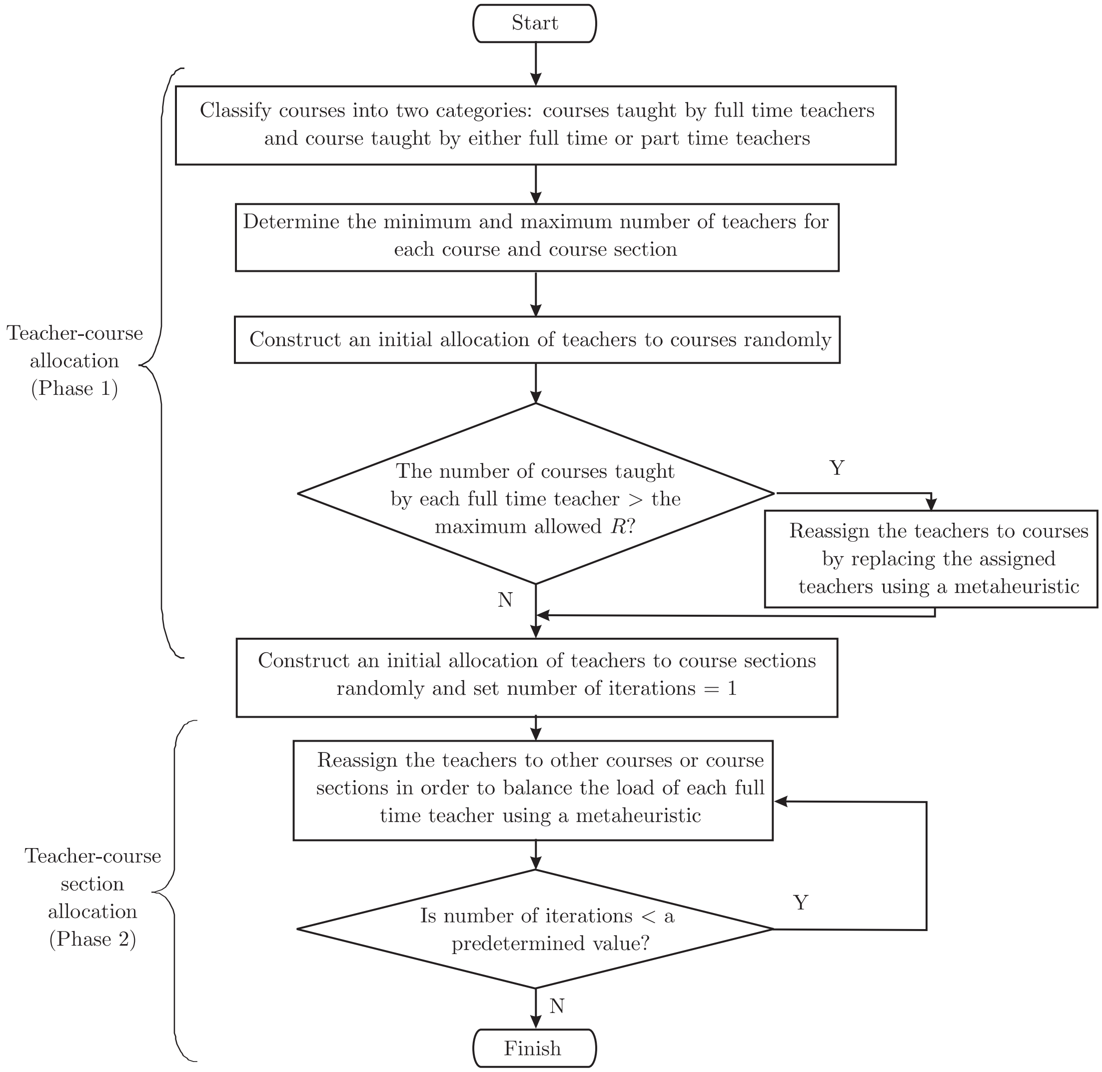

Figure 1: The flow chart of the proposed algorithm.

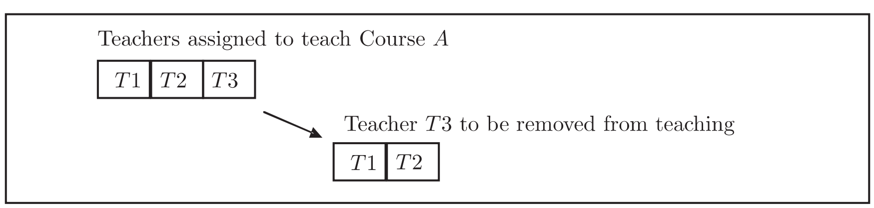

Figure 2: Example of a single move in Phase 1 of the proposed algorithm.

In Phase 1, the neighborhood structures used in SA are based on two types of movement: a single move and a double move. The idea of the single move is to pick any teacher who has been assigned to teach more than R courses and remove him/her

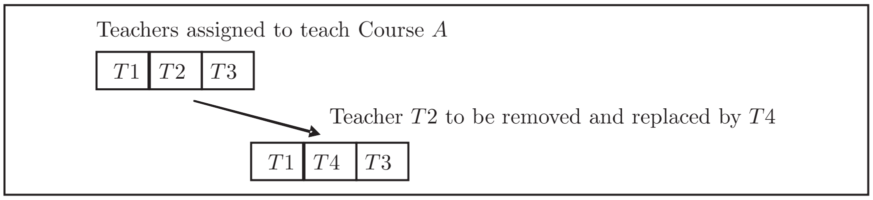

Figure 3: Example of a double move in Phase 1 of the proposed algorithm.

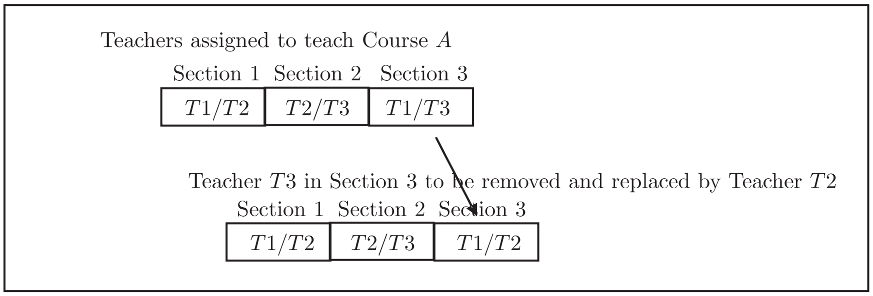

Figure 4: Example of a double move in Phase 2 of the proposed algorithm.

from teaching a particular course without reallocating the course to another teacher (see Figure 2 for illustration). The double move is to pick any teacher who has been assigned more than R courses and remove him/her from teaching a particular course by reallocating the course to another teacher who is able to teach the course (see Figure 3 for illustration). Both moves can be applied as long as there is no violation of hard constraints.

The same concept of moves could be applied in Phase 2 of the proposed algorithm. We try to improve the load of some teachers by applying the double move to a particular course. Initially, we generate a set of teachers who have higher loads, which is called the Excess List. It is assumed that the length of the Excess List is equal to a certain percentage of the total number of full time teachers. A teacher is chosen randomly from the Excess List, followed by choosing one course section that has been allocated to this teacher. This teacher is then relieved from teaching the selected course section which will be assigned to another teacher; and this is called a double move (see Figure 4 for illustration).

In this move, we keep the number of teachers who teach a particular course unchanged. The changes are only on the distribution of teachers' load in that course. Suppose instead there is no teacher available for replacement. We can simply relieve the teacher from the selected course section as long as there is no violation of any hard constraint, and this is called a single move (see Figure 5 for illustration). Instead of applying only one type of move as was commonly done in simulated annealing, we apply two different moves in order to further explore other possible solutions.

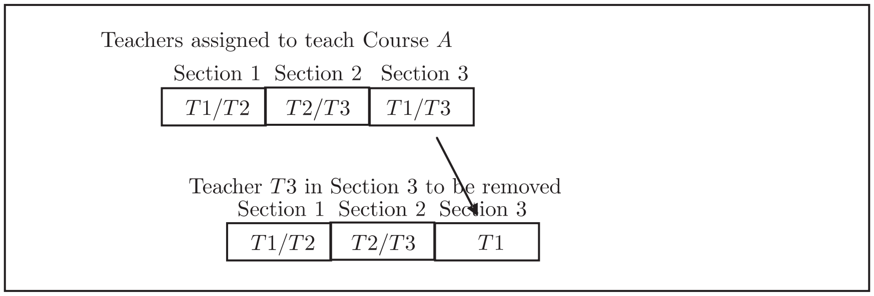

Figure 5: Example of a single move in Phase 2 of the proposed algorithm.

3.2. Tabu search

TS is a heuristic for solving discrete optimization problems. The basic idea of TS is described in detail by Glover [20, 21]. It is designed to guide other local search approaches to continue solution space exploration without falling back into local optima from which it previously emerged [17]. TS is based on gradual local improvement of a current solution of the optimization problem. The fundamental idea of TS is the use of flexible memory of search history which guides the search process to surmount local optimal solutions.

Many different papers have presented applications of Tabu Search to various combinatorial problems, such as the Quadratic Assignment Problem [16], machine scheduling problem [19] and timetabling problem [13].

Similar to SA, TS searches for a new solution in the neighborhood of the current solution. The difference between TS and SA is the method of choosing a new solution from the neighborhood. SA chooses the new solutions randomly, while TS searches the new solution from a subset or the whole neighborhood space for a good solution with certain restrictions. When we get a better solution, we treat the new one as the current solution. However, when there is no better solution, the 'best' solution (although it is actually worse than the current one) in the neighborhood is picked. This situation differs from that of SA which may accept a worse solution with a certain probability.

The basic elements of TS are the moves, tabu list, and aspiration level. The tabu list consists of certain solutions or moves that are forbidden and are called tabu moves. The size of the tabu list has a great effect on the solution quality. The aspiration criterion is introduced in TS in order to determine when a tabu move could be overridden. The main purpose is to enable tabu moves that could possibly lead to an optimal solution. In this study, we define the teacher and his/her course or course section that has been removed within a certain number of previous iterations as an element of the tabu list. The aspiration level is defined to be the case when a move could lead to a new best solution.

In Phase 1, the neighborhood structures used in TS are similar to those of SA. In Phase 2, another type of neighborhood structure is used. In TS, it is commonly found that all possible neighborhoods are examined in order to find the best one. However, this approach can be prohibitively large as the problem size increases. Here, we propose a different type of neighborhood structure. The neighborhood structure used in TS is obtained by randomly choosing only a teacher who has an excess load. The next step is to examine all the possible teachers and choose one that gives the best allowed neighbor.

In TS, we also apply the intensification strategy. Thus if the best solution has not improved within a certain number of iterations, we focus the search once again starting from the best solution obtained.

4. Computational Results

In order to evaluate the performance of the proposed algorithms numerically, we have used two real data sets and four random data sets. These data sets have also been solved by another approach, genetic algorithm (GA), in Gunawan et al. [24]. The real data sets were collected from an institution in Indonesia. The characteristics of each set of data are summarized in Table 1 with the real data sets being referred to as Real Data 1 and Real Data 2, while the random data sets are referred to as Random Data 1, Random Data 2, Random Data 3 and Random Data 4.

For the random data sets, we varied the following parameters: the number of full time teachers for Groups 1 and 2, the number of part time teachers, the number of courses offered, the maximum number of sections for each course and the maximum number of teachers allowed for each course section. Table 2 gives the minimum and maximum number of teachers that can be assigned to teach a course, which depend on the number of sections offered for the particular course.

Table 1: Characteristics of data sets used in the computational experiments.

	Real Data 1	Real Data 2	Random Data 1	Random Data 2	Random Data 3	Random Data 4
Number of full time teachers:						
- Group 1	25	22	20	20	7	90
- Group 2	3	3	5	5	3	10
Number of part time teachers	18	10	5	10	5	20
Number of courses offered	48	45	50	100	50	200
Maximum number of sections	5	6	5	5	5	5
Maximum number of teachers allowed for each course section	2	2	2	2	2	2

All the proposed algorithms for solving the problems are coded in Visual C++ 6.0 and tested on a Pentium III 497 MHz PC with 128 MB RAM under the Microsoft Windows XP Operating System. For each data set, both algorithms were executed five times. The best solutions obtained by the proposed algorithms are also compared with the manual allocation procedure performed by the university. The parameter values of the proposed algorithms are chosen experimentally to ensure a compromise between running time and solution quality. These values are summarized as follows:

Table 2: The minimum and maximum number of teachers that could teach in a course.

Number of sections	Minimum (teachers)	Maximum (teachers)
1	1	2
2	1	3
3	1	3
4	3	4
5	3	5
6	3	6

Phase 1

In the SA algorithm, we set the number of neighbor moves at each given temperature = number of full time teachers × number of courses, maximum number of iterations = 1000, initial temperature = 1000, cooling factor = 0.95, minimum percentage of accepted moves (*min_percent*) = 10%.

In the TS algorithm, we set the maximum number of iterations = 2500, number of neighbor moves = 2× number of full time teachers, length of tabu list = 7, number of iterations of intensification strategy = 100.

Phase 2

In the SA algorithm, we set the number of neighbor moves at each given temperature = 4× number of full time teachers × number of courses, maximum number of iterations = 1×10^5, initial temperature = 1000, cooling factor = 0.99, minimum percentage of accepted moves (*min_percent*) = 10%.

In the TS algorithm, we set the number of iterations = $10^7\times$ number of full time teachers, number of neighborhood moves = 2 × number of full time teachers × number of courses, length of tabu list = 7, number of iterations of intensification strategy = 100.

4.1. Results of teacher - course allocation (Phase 1)

The results of Phase 1 are summarized in Table 3. The penalty values represent the excess of the total number of courses taught by full time teachers who have to teach more than R courses. By the SA and TS algorithms, there would be nine teachers teaching more than two courses when the maximum number of courses taught by each full time teacher is set to two courses for Real Data 1, and most of teachers have to teach about four or five courses. In other words, most of teachers have the penalty value of at least two or three courses. So, the total excess number of courses or the penalty value is eighteen courses for both the SA and TS algorithms, which is better than that of the GA algorithm (twenty four courses).

Similarly for Real Data 2, if the minimum number of courses is set to three courses, there will be eight and six teachers teaching more than three courses by applying the SA and TS algorithms, respectively. In this case, the penalty values for both the SA and TS algorithms are eleven courses. Similar observations can be found for other random instances.

Table 3: The best results of teacher-course allocation (Phase 1).

Data set	Maximum number of courses taught by each teacher (R)	Number of full time teachers teaching more than R courses			Penalty Value		
		SA	TS	GA	SA	TS	GA
Real Data 1	2	9	9	9	18	18	24
	3	0	0	2	0	0	6
	4 or more	0	0	0	0	0	0
Real Data 2	3	8	6	5	11	11	19
	4 or more	0	0	0	0	0	0
Random Data 1	6	3	4	3	4	4	21
	7 or more	0	0	0	0	0	0
Random Data 2	12	11	7	3	18	18	21
	13 or more	0	0	0	0	0	0
Random Data 3	14	2	2	1	2	2	12
	15 or more	0	0	0	0	0	0
Random Data 4	5	49	43	40	96	96	89
	6 or more	0	0	0	0	0	0

Feasibility is achieved in Phase 1 only if the number of full time teachers teaching more than R courses is zero, meaning that the penalty value is also zero. For instance, Phase 1 feasibility is achieved for Real Data 1 by applying SA and TS algorithms when we set the maximum number of courses taught by each teacher to be three courses. It means that the number of courses that should be allocated to each full time teacher is three courses. GA algorithm can only achieve the feasibility when the number is four courses.

By referring to the results for Random Data 2 and Random Data 3, the minimum number of courses that should be allocated to each full time teacher is considered large since the number of full time teachers is small compared to the number of courses offered.

4.2. Results of teacher - course section allocation (Phase 2)

In order to ensure feasibility of the solution in Phase 2, we set the maximum number of courses that could be taught by each full time teacher for Real Data 1 and Real Data 2 to be three and four courses, respectively. A similar approach is also applied to the random data sets and we set the corresponding maximum number of courses to be 7, 13, 15 and 6 courses, respectively.

Phase 2 is started by using a randomly generated initial solution based on the results of Phase 1. In Phase 2, five different initial solutions are generated for each data set, followed by applying the SA or TS algorithm. The average load and average variance obtained after applying the SA and TS algorithms are presented in Tables 4 and 5. The performance of the TS algorithm is better than that of the SA algorithm in terms of the average variance obtained. In general, both SA and TS algorithms also outperform the GA algorithm [24]. For instance, the average total variances of the load by SA and TS

Table 4: Computational results of average load of Phase 2.

Data set	SA	TS
Real Data 1	4.90	4.83
Real Data 2	6.35	6.67
Random Data 1	9.16	9.44
Random Data 2	21.65	21.81
Random Data 3	26.8	26.97
Random Data 4	10.88	11.14

Table 5: Computational results of average total variance of the load in Phase 2.

Data set	SA	TS	GA
Real Data 1	0.80	0.43	2.26
Real Data 2	2.15	1.89	3.19
Random Data 1	2.52	2.07	3.85
Random Data 2	1.06	1.03	1.34
Random Data 3	2.31	0.22	0.44
Random Data 4	3.92	2.34	2.54

algorithms are only 0.80 and 0.43, respectively. On the other hand, the average total variance of the load by GA algorithm is 2.26, which is a considerably higher value.

For both real data sets, the best average results obtained by the TS algorithm are 0.43 and 1.89, respectively. The main reason for this is the exploration of the neighborhood in TS in Phase 2. Instead of random selection for teacher replacement, this algorithm focuses on examining all possible teachers and selecting one which gives a better allocation or reduces the variance of the load.

4.3 Comparison of the algorithms and manual allocation procedure

In a similar form of comparison to Chahal and Werra [11], Ergül [18] and Rankin [31], the results obtained by the proposed algorithms are also compared with that obtained from actual manual allocation procedures performed by the Industrial Engineering Department. The manual allocation procedure is started by a discussion between the heads of department and laboratories in order to come up with a draft allocation. This is followed by a general meeting that is attended by all full time teachers to discuss how to modify and finalize the allocation of courses to teachers. Table 6 summarizes these comparison results.

It can be seen that the results for the manual allocation procedure are worse than that of the SA and TS algorithms. This is because in practice, the department does not restrict the maximum number of courses to be taught by each full time teacher. Such a procedure would create unbalanced load among full time teachers since some of them would then be assigned to teach only a few courses, while others would be assigned to teach a large number of courses. Our computational experiments show that through

Table 6: Comparison of the results of the proposed algorithms and manual allocation procedure.

Data set	Best average total variance of the load		
	SA	TS	Manual Allocation
Real Data 1	0.80	0.43	3.35
Real Data 2	2.15	1.89	2.80

setting the maximum number of courses to be taught by each teacher in the proposed algorithms, the total variance of the load can be reduced.

5. Conclusions

In this paper, we describe a teacher assignment problem which allows the possibility of courses being taught by more than one teacher, and propose the use of SA and TS algorithms for solving the problem. The proposed algorithms consist of two phases. The first phase focuses on allocating the teachers to the courses and the number of courses that should be assigned to each teacher. The second phase, which is based on the results obtained from the first phase, focuses on allocating the teachers to the course sections in order to balance the teachers' load.

To evaluate the performance of the proposed algorithms, we have performed computational experiments on two sets of real data obtained from a university in Indonesia and some randomly generated data sets. For the real data sets, the computational results show that both the SA and TS algorithms yield better solutions when compared to previous work as well as manual allocation procedures. In general, both algorithms are able to generate good solutions to the problem instances.

Some possible directions for future research include applying other types of metaheuristics, such as ant colony optimization, to the teacher assignment problem, as well as the development of other types of neighborhood structures for the SA and TS algorithms. The requirements imposed in the teacher assignment problem can also be extended to adapt to other characteristics and requirements which are unique to some institutions, such as limiting the number of course sections that can be taught by each teacher and differing the maximum number of courses that can be taught by each teacher. Another possible extension is to incorporate course scheduling into the teacher assignment problem so that the teacher can not only choose the course that he/she prefers to teach, but also can choose the preferred time periods for teaching the course.

References

[1] Andrew, G. M. and Collins, R., *Matching faculty to course*, College University, Vol.46, No.2, pp.83-89, 1971.

[2] Bellanti, F., Carello, G., Croce, F. D. and Tadei, R., *A greedy-based neighborhood search approach to a nurse rostering problem*, European Journal of Operational Research, Vol. 153, No. 1, pp. 28-40, 2004.

[3] Breslaw, J. A., *A linear programming solution to the faculty assignment problem*, Socio-Economic Planning Sciences, Vol. 10, No. 6, pp. 227-230, 1976.

[4] Burke, E. K. and Petrovic, S., *Recent research directions in automated timetabling*, European Journal of Operational Research, Vol. 140, No. 2, pp. 266-280, 2002.

[5] Burke, E. K., de Causmaecker, P., Berghe, G. V. and Landeghem, H. V., *The state of the art of nurse rostering*, Journal of Scheduling, Vol. 7, No. 6, pp. 441-499, 2004.

[6] Burke, E. K., Kingston, J. H. and de Werra, D., *Applications to timetabling*, in Gross, J. and Yellen, J. (Eds.), The Handbook of Graph Theory, pp. 445-474, Boca Raton: CRC Press, 2004.

[7] Burke, E. K., McCollum, B., Meisel, A., Petrovic, S. and Qu, R., *A graph-based hyper-heuristic for educational timetabling problems*, European Journal of Operational Research, Vol. 176, No.1, pp. 177-192, 2007.

[8] Carrasco, M. P. and Pato, M. V., *A multiobjective genetic algorithm for the class/teacher timetabling problem*, in Burke, E. and Carter, M. (Eds.), Practice and Theory of Automated Timetabling III, Lecture Notes in Computer Science, Springer-Verlag, Berlin, Vol. 2079, pp. 3-17, 2000.

[9] Carter, M. W. and Laporte, G., *Recent developments in practical course timetabling*, in Burke, E. and Carter, M. (Eds.), Practice and Theory of Automated Timetabling II, Lecture Notes in Computer Science, Springer-Verlag, Berlin, Vol. 1408, pp. 3-19, 1998.

[10] Cerny, V., *A thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm*, Journal of Optimization Theory and Applications, Vol. 45, No. 1, pp. 41-51, 1985.

[11] Chahal, N. and de Werra, D., *An interactive system for constructing timetables on a PC*, European Journal of Operational Research, Vol. 40, No. 1, pp. 32-37, 1989.

[12] Chen, S-M and Lin, C-H., *Multiple DNA sequence alignment based on genetic simulated annealing techniques*, Information and Management Sciences, Vol. 18, No.2, pp. 97-111, 2007.

[13] Costa, D., *A tabu search algorithm for computing an operational timetable*, European Journal of Operational Research, Vol. 76, No. 1, pp. 98-110, 1994.

[14] de Causmaecker, P., Demeester, P. and Berghe, G. V., *A decomposed metaheuristic approach for a real-world university timetabling problem*, European Journal of Operational Research, Vol. 195, No. 1, pp.307-318, 2009.

[15] de Werra, D., *An introduction to timetabling*, European Journal of Operational Research, Vol. 19, No. 2, pp. 151-162, 1985.

[16] Drezner, Z., *The extended concentric tabu for the quadratic assignment problem*, European Journal of Operational Research, Vol. 160, No. 2, pp. 416-422, 2005.

[17] Du, D.Z. and Pardalos, P.M., Handbook of Combinatorial Optimization, Kluwer Academic Publishers, The Netherlands, 1998.

[18] Ergİ, A., *GA-based examination scheduling experience at Middle East Technical University*, in: Burke, E.K. and Ross, P. (Eds.), The Practice and Theory of Automated Timetabling I: Selected Papers (PATAT 1995), Lecture Notes in Computer Science, Springer-Verlag, Berlin, Vol. 1153, pp. 212-225, 1996.

[19] Eswaramurthy, V. P. and Tamilarasi, A., *Hybridization of ant colony optimization strategies in tabu search for solving job shop scheduling problems*, International Journal of Information and Management Sciences, Vol. 20, No. 2, 173-189, 2009.

[20] Glover, F., Tabu search - part I, *ORSA Journal on Computing*, Vol. 1, No. 3, pp. 190-206, 1989.

[21] Glover, F., Tabu search - part II, *ORSA Journal on Computing*, Vol. 2, No. 1, pp. 4-32, 1990.

[22] Gunawan, A. and Lau, H. C., *Master physician scheduling problem*, the proceedings of the 4th Multidisciplinary International Scheduling Conference, 10-12 August 2009, Dublin, Ireland.

[23] Gunawan, A., Ng, K. M. and Poh, K. L., *Solving the teacher assignment-course scheduling problem by a hybrid algorithm*, International Journal of Computer, Information and Systems Science and Engineering, Vol. 1, No. 2, pp. 136-141, 2007.

[24] Gunawan, A., Ong, H. L. and Ng, K. M., *A genetic algorithm for the teacher assignment problem*, International Journal of Information and Management Sciences, Vol. 19, No. 1, pp. 1-16, 2008.

[25] Harwood, G. B. and Lawless, R. W., *Optimizing organizational goals in assigning faculty teaching schedules*, Decision Science, Vol. 6, No. 3, pp. 513-524, 1975.

[26] Kirkpatrick, S., Gellatt, C. D. and Vecchi, M. P., *Optimization by simulated annealing*, Science, Vol. 220, No. 5, pp. 671-680, 1983.

[27] Kwan, R. S. K., *Bus and train driver scheduling*, in Leung, J. (Eds.), Handbook of Scheduling: Algorithms, Models and Performance, Boca Raton: CRC Press, Chapters 51, 2004.

[28] Liu, S. and Ong, H. L., *A comparative study of algorithms for the flowshop scheduling problem*, Asia-Pacific Journal of Operational Research, Vol. 19, No. 2, pp. 205-222, 2002.

[29] Massoodian, S. and Esteki, A., *A hybrid genetic algorithm for curriculum based course timetabling*, the proceedings of the 7th International Conference on the Practice and Theory of Automated Timetabling, 18-22 August 2008, Montreal, Canada.

[30] Radhakrishnan, S. and Ventura, J. A., *Simulated annealing for parallel machine scheduling with earliness-tardiness penalties and sequence-dependent set-up times*, International Journal of Production Research, Vol. 38, No. 10, pp. 2233-2252, 2000.

[31] Rankin, R.C., *Automatic timetabling in practice*, in Burke, E.K. and Ross, P. (Eds.), The Practice and Theory of Automated Timetabling I: Selected Papers (PATAT 1995), Lecture Notes in Computer Science, Springer-Verlag, Berlin, Vol. 1153, pp. 266-279, 1996.

[32] Saleh Elmohamed, M.A., Coddington, P. and Fox, G., *A comparison of annealing techniques for academic course scheduling*, in Burke, E. and Carter, M. (Eds.), Practice and Theory of Automated Timetabling II, Lecture Notes in Computer Science, Springer-Verlag, Berlin, Vol. 1408, pp. 92-112, 1998.

[33] Schaerf, A. and Meisels, A, *Solving employee timetabling problems by generalized local search*, in Lamma, E. and Mello, P. (Eds.), AI*IA 99: Advances in Artificial Intelligence: 6th Congress of the Italian Association for Artificial Intelligence, Lecture Notes in Computer Science, Springer-Verlag, Berlin, Vol. 1792, pp. 380-389, 2000.

[34] Schaerf, A., *A survey of automated timetabling*, Artificial Intelligence Review, Vol. 13, No. 2, pp. 87-127, 1999.

[35] Schniederjans, M.J. and Kim, G. C., *A goal programming model to optimize departmental preference in course assignments*, Computers and Operations Research, Vol. 14, No. 2, pp. 87-96, 1987.

[36] Schönberger, J., Mattfeld, D. C. and Kopfer, H, *Memetic algorithm timetabling for non-commercial sport leagues*, European Journal of Operational Research, Vol. 153, No. 1, pp. 102-116, 2004.

[37] Tillett, P. I., *An operations research approach to the assignment of teachers to courses*, Socio-Economic Planning Sciences, Vol. 9, No. 3, pp. 101-104, 1975.

[38] Valdes, R. A., Crespo, E. and Tamarit, J. M., *Assigning students to course sections using tabu search*, Annals of Operations Research, Vol. 96, No. 1, pp. 1-16, 2000.

[39] Votre, V. P., *MAGHO - A tool for teachers and physical resources assignment and for student's scheduling*, the proceedings of the 7th International Conference on the Practice and Theory of Automated Timetabling, 18-22 August 2008, Montreal, Canada.

[40] Wang, Y. Z., *An application of genetic algorithm methods for teacher assignment problems*, Expert Systems with Applications, Vol. 22, No. 4, pp. 295-302, 2002.

[41] Wren, A., *Scheduling, timetabling and rostering - a special relationship?*, in Burke, E. and Ross, P. (Eds.), Practice and Theory of Automated Timetabling I, Lecture Notes in Computer Science, Springer-Verlag, Berlin, Vol. 1153, pp. 46-75, 1996.

[42] Yu, E. and Sung, K. S., *A genetic algorithm for a university weekly course timetabling problem*, International Transactions in Operational Research, Vol. 9, No. 6, pp. 703-717, 2002.

School of Information Systems, Singapore Management University, 80 Stamford Road, Singapore 178902.

E-mail: aldygunawan@smu.edu.sg

Major area(s): Heuristic algorithms, Scheduling, Timetabling problems.

Department of Industrial and Systems Engineering, National University of Singapore, 1 Engineering Drive 2, Singapore 117576.

E-mail: isenkm@nus.edu.sg

Major area(s): Optimization, Numerical algorithms.

(Received April 2010; accepted January 2011)