Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/070529

Innovative Crossover and Mutation in a Genetic Algorithm Based Approach to a Campus Bus Driver Scheduling Problem with Break Consideration and Embedded Overtime

Razamin Ramli[*], *Haslinda Ibrahim and Lim Tze Shung*

College of Arts and Sciences, Universiti Utara Malaysia, 06010 Sintok, Kedah, Malaysia

Received: 22 Jan. 2013, Revised: 23 May. 2013, Accepted: 24 May. 2013
Published online: 1 Sep. 2013

Abstract: The unfairness of job distribution and the ineffectiveness of break-time assignment among bus drivers are factors identified as problem issues in the bus driver management in a university campus environment. The issues affect the quality of the bus service which then pointed to the efficiency or inefficiency of the bus driver scheduling system. Thus, this study investigates the problem and constraints connected to bus driver scheduling activities as a case in a campus environment. Therefore, we developed an efficient bus driver scheduling model based on the Genetic Algorithm (GA) approach. The GA model is able to schedule efficiently bus drivers to working slots and break in a particular day. It is thus, a time-saving effort and able to satisfy equal break distribution among drivers with overtime work being considered at the same time.

Keywords: Genetic algorithm, meta-heuristics, bus driver scheduling, man power scheduling

1 Introduction

The bus service is an essential internal transportation for students within the university campus area. In a particular large university campus bus service, it has been observed that one of the causes for the long wait is due to poor bus driver work schedules. Due to poor schedules, many drivers tend to take their breaks together that is, in the same timeslot. In turn, this situation has caused the unfairness in work or job distribution since the break-times of drivers are loosely determined by the drivers themselves. The problem is further worsened when there were drivers assigned to different work shifts and break durations. Thus, the unfairness of job distribution and ineffectiveness of break-time assignment among bus drivers are factors identified as problem issues in this poor bus driver management [1].

Generating a bus driver work schedule is tedious since it deals with the assignment or allocation of drivers to work shifts, break periods and off days, and at the same time, trying to fulfill certain working constraints over a planning period. In other words, all legal working shifts should be designed such that all scheduled buses have drivers assigned at all times of a day [2].

A bus driver scheduling problem (BDSP) is commonly formulated as a set covering problem [3]. The solution for BDSP is to generate sets of potential drivers' shift duties to cover sets of bus work, while fulfilling objectives such as cost minimization and quality of service provided. There has been several solution approaches attempted in previous related work such as the mathematical programming methods [4] and Constraint Programming [5]. Other approaches which are heuristics in nature include the Genetic Algorithm [6], Ant Colony System [7], Tabu Search [8], and Greedy Randomized Adaptive Search Procedure (GRASP) for large instances [9].

It has been observed that Genetic Algorithm (GA) or GA-based approaches are among the highly experimented or implemented in various problems for the respective solutions. This is true because GA is able to produce very good solutions [6], [10],. Moreover, GA can be easily coded and in a reasonable short time [11]. However, being a heuristic approach GA still provides avenues and rooms

[*] Corresponding author e-mail: razamin@uum.edu.my

for improvement in the technique, such as in the crossover and mutation for the BDSP.
Thus, the aim of this paper is to present an innovative methodology design in a GA based solution approach for the BDSP such that, it contributes towards high quality schedules within a reasonable time-frame. The solution approach is able to generate the best possible schedule through the enhancement of important GA operators, while fulfilling certain identified constraints relevant to the working environment.

2 Genetic Algorithm

Genetic Algorithm (GA) is stochastic search approach based on the mechanics of natural selection and natural genetics [12] which also possesses the concept of evolution. It has been proven to be a versatile approach to solving many complex and hard optimization problems [13], [14], [15] .
Recommended as a useful tool to solve driver scheduling problem, [6] replaced a previous solution approach, which used integer programming (IP) in a work on driver scheduling problem by a GA approach. The results gave good solution performance, although it started with a large set of GA population or candidate solutions.
As agreed by [16] GA does not impose any particular restrictions on the structure of the objective function. That may therefore, encompass different characteristics that are usually very difficult to handle by traditional algorithms. It offers a natural framework for computational parallelization, feasible solutions at the end of each iteration, and the evaluation function can become as complex as required. GA does produce quick and very satisfactory solutions, bringing automatic solutions closer to the planners' expectations as experienced in the medium size Portuguese urban bus company problem, and multi-criteria classification process in early diagnosis of Alzheimer's Disease [17].
Crossover and mutation are among the important GA operators or components in the whole mechanism. Various types of crossover are such as the single-point, multiple-point, uniform and matrix [18]. However, one can be very creative to design a mutation operator to suit a particular problem at hand. Nevertheless, the two common mutation types are binary encoded [19] and permutation encoded mutations [20].

3 Bus driver scheduling problem

The bus drivers' duty and the daily bus schedule in a university campus were examined. Based on our observation, buses departed every 15 minutes from 7.00 am – 8.00 pm daily on schooling days. However, the required number of buses for every time slot was never specifically stated. The only requirement was that there must always be drivers on stand-by to run the bus every 15 minutes. This situation added to the inefficiency of the scheduling system. Hence, to ensure a good quality of bus service in the campus, an efficient bus driver schedule should be generated especially one which could overcome the situation where many drivers were having their breaks at the same time.
The campus bus service was considered as a circular type whereby the buses transferred students from 16 residential colleges or hostels to academic zones and vice versa. There were routes that the buses needed to serve everyday (i.e. Route A, B, C and D) based on the location of these residential colleges. We the identified a fixed number of buses assigned to each route, where the respective bus drivers were expected to drive on the fixed routes every day. As identified, there were 40 buses and 40 drivers. Consequently, this has allowed one driver to be assigned to one bus only. Based on discussion with the management, a number of buses were allocated to each route as shown in Table 1, with regards to student capacity.

Table 1: Number of buses assigned to each bus route

Route	No. of Bus
A	6
B	10
C	11
D	13

The drivers needed to work for 13 hours per day (including over-time duty). Due to management requirements, an hour was divided into four to obtain four 15-minutes slots. Therefore, in a working day there would be fifty two 15-minutes time slots. The BDSP was described as having hard and soft constraints as listed below.

Hard constraints

The following were hard constraints, which mandatorily governed the generation of the bus driver schedules:

1. Mandatory working period for every driver was 8 hours per day.
2. Over-timed period for every driver was 5 hours per day.
3. Every piece of work should be at least one hour duration and does not exceed 4 hours.
4. No breaks were allowed during the first hour of a working day.
5. Drivers were given three breaks in a day with every break consisted of half an hour.

Soft constraints

It is good if all soft constraints could be satisfied as much as possible in the generation of the bus driver schedules, as given below:

1. It is preferable that the number of drivers who go for breaks together is eight persons.

2.It is preferable that two breaks were assigned during the mandatory working periods while one break during the over-timed periods.

A GA approach to solving the BDSP

The overall steps of the GA are as exhibited in the flow chart in Fig 1.

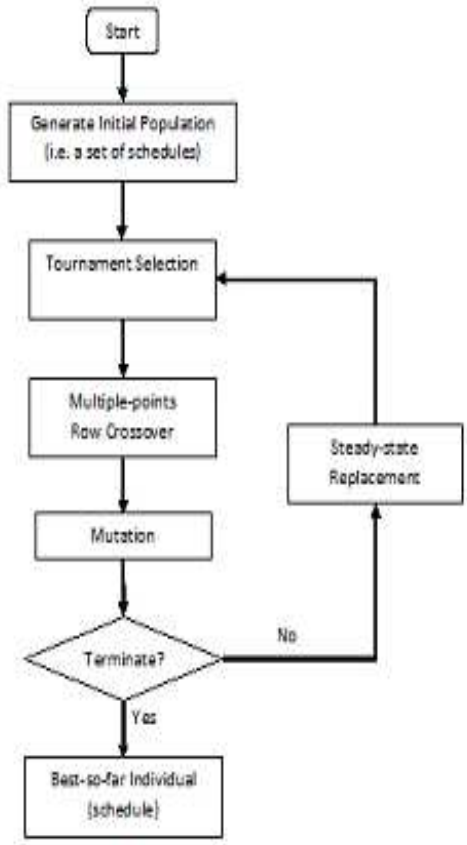

Fig. 1: The structure chart for the proposed GA in scheduling the BDSP

Solution representation

An individual or schedule is directly coded as a 2-dimensional (m x n) matrix representation, where the column is for time/working slot, $\{T_1, T_2, \ldots, T_n\}$ and the row is for driver $\{D_1, D_2, \ldots, D_m\}$. The term individual is also synonym to chromosome, which is commonly used in the GA literature. The use of direct coding in GA can provide a good solution at a fast time [21], which has motivated us to adopt this direct coding of representation.

Fitness Function

Referring to the first soft constraint, when a generated schedule has less or more than eight drivers that break together, a penalty would be enforced. The first sub-fitness function is based on the violation of this first soft constraint given as follows.

$$f_1(s) = \sum_{j=1}^{N} |\sum_{i=1}^{M} b_{ji} - B_{ideal}| p_{iy} c_{ij}, \forall y$$

The penalty values or weights were decided intuitively based on the relative importance of both soft constraints and also as a result of discussion with the experienced manager cum scheduler. For each category of soft constraint, the penalty values are distributed as in Table 2.

Table 2: Penalty values for the first sub-objective function

Soft constraint of type y	$\sum_{i=1}^{M} b_{ji} - B_{ideal}$	Penalty value, p_{1y}
1	0	0
2	1	10
3	2	20
4	≥ 3	30

The second sub-fitness function is computed based on the violation of the second soft constraint. When a generated schedule consisting of drivers that have less or more than one time break during the over-time period, a relevant penalty given. The function is as follows.

$$f_2(s) = \sum_{i=1}^{N} |\sum_{j=33}^{M} o_{ji} - O_{ideal}| p_2 c_2$$

M, N, represent respectively, the total number of drivers, and total number of timeslots, whereas B_{ideal}, O_{ideal}, represent respectively, the ideal number of break in each timeslots, and ideal number of drivers break during over-timed period. b_{ji} and o_{ij} are the decision variables whether x_{ij} = 0 or 1. c_{1j} is the decision variable whether soft constraint exist or not, whereas c_2 is the decision variable whether soft constraint exist or not. p_{1y}, p_{2z}, respectively represent penalty value given when soft constraint of type y exists, and penalty value when soft constraint of type z exists, where y = 1, 2, 3, 4. Subsequently, the penalty values assigned for p_2 are as in Table 3.

Table 3: Penalty values for the second sub-objective function

Soft constraint of type z	$\lvert\sum_{j=33}^{M} o_{ji} - O_{ideal}\rvert$	Penalty value, p_{2z}
1	0	0
2	≥ 1	10

By combining both sub-fitness functions, the overall objective function is to minimize the number of soft constraints' violations, and is calculated as follows:

To minimize:

$$f(s) = \sum_{j=1}^{N} | \sum_{i=1}^{M} b_{ji} - B_{ideal} | p_{iy} c_{ij} + \sum_{i=1}^{N} | \sum_{j=33}^{M} o_{ji} - O_{ideal} | p_2 c_2$$

Initialization of a population

The first step of GA in the BDSP starts by generating randomly a population of feasible individuals or chromosomes, that is those fulfilling all hard constraints. Different sizes of population were tested but our experiments suggested that 30 (N = 30) to be the ideal size as it lead towards promising fitness values [18].

Parent selection mechanism

Selection of parents are based on tournament selection mechanism (i.e. binary tournament) due to its efficiency as suggested by [16]. This mechanism is simple and more efficient in implementation when compared to the proportionate selection method. Two individuals were chosen at random from the initial population with their fitness evaluated and compared to each other. The fittest individual was chosen to be a parent, while the less fit individual is returned to the original population and have the chance to be selected again. The mechanism is repeated to gain the second parent.

Crossover procedure

Horizontal directions are proposed in our crossover procedure so as to ensure that hard constraints are not violated when children or offsprings are produced. The crossover is performed at the rate of 0.5 as similarly done by Morz and Musliu [22]. In cases when there is no crossover to be done, the offspring string is exactly copied from the parent with better fitness value.

We introduced a multiple-point crossover (i.e. 3-point crossover) in the procedure. Considering the GA as having small populations, a more disruptive crossover operator such as n-point (with n = 2) may yield better results as suggested by [23]. This is because it helps to overcome the limited information capacity of small populations and the tendency for more homogeneity. Thus, combining a 3-point crossover with a horizontal cut procedure as shown in Figure 2, an innovative crossover mechanism is employed in the BDSP.

Mutation procedure

In our study, random mutation may easily lead to infeasible solutions due to possible violation of hard constraints. Thus, we introduced a directed or guided mutation operator which is constraint-based. In the mutation procedure, we first randomly choose a row in the selected offspring. Then, randomly choose an allele with a value of *0*, swap the two alleles adjacent to the selected allele. If we obtain a non-zero allele, the random selection would be repeated until we obtain a zero allele.

We experimented with low [24] and high [22] mutation rates as both types have proved promising solutions in various problem contexts. Hence, we tested the mutation operator with various rates from 0.5 – 0.8.

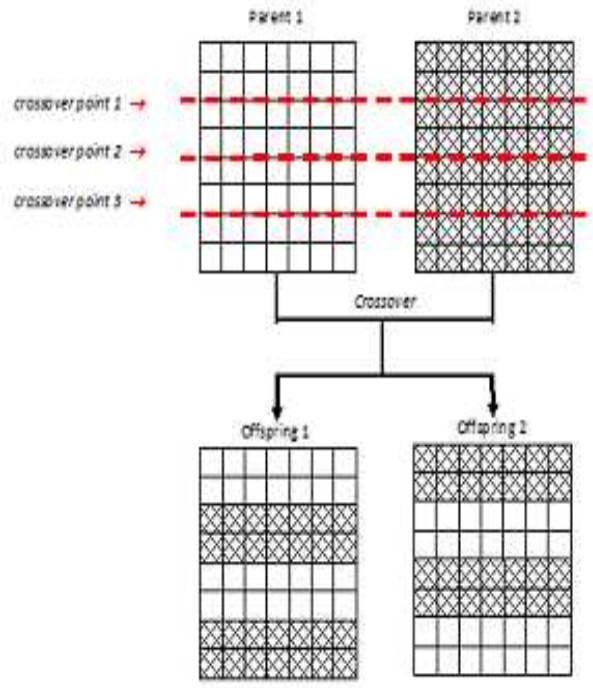

Fig. 2: An innovative mechanisme of a 3-point horizontal crossover

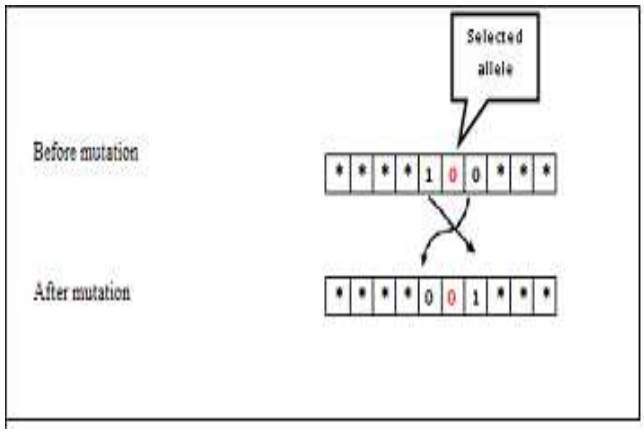

Fig. 3: An example of the proposed mutation procedure for the BDSP

Replacement procedure At the end of a mutation step, some of the individuals were replaced in the next generation following the concept of the steady-state replacement introduced by [25]. The newly produced offspring after crossover and mutation automatically replaced the worst individual in the previous population. However, duplication is allowed when the offspring

produced is same (in term of fitness) with other individual in that previous population. It means that there could be two or more individuals with same fitness values in the new population. It is noted that even with the same fitness values, two individuals may have differences in their genes (i.e. physical arrangement).

Computational Result

Many experiments were carried out to verify the performance of the proposed GA. In this paper, we present an efficient innovative model with a powerful high rate of mutation as suggested by [22], which is 0.8. The crossover rate is 0.5. Subsequently, experiments with different population sizes such as 10, 20, 30, 40, 50 and 60 were also carried out. During experiments, the number of generations and number of drivers were kept constant and run on a Pentium M with a speed of 80 megahertz with execution times in CPU minutes. The user interface for the prototype is given as in Figure 4 for parameters input.

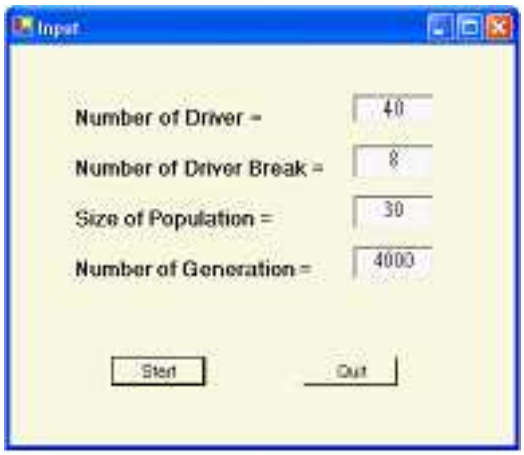

Fig. 4: A user interface for entering input

The performance of the GA with promising fitness values was obtained when the size of population was 30 as discussed in [18]. It was also observed that smaller size populations required shorter time to reach convergence, while larger size population required longer time to gain the best-so-far fitness. However, all runs showed the trend of convergence after 4000 generations.

Consequently, we continued to test the effectiveness of crossover and mutation rates with population size of 30. The parameters used in the experiments were (i) crossover rate, P_c = 0.8, mutation rate, P_m = 0.1 - 0.5, (ii) crossover rate, P_c= 0.8, mutation rate,P_m = 0.3 and 0.4, and each of those experiments was run for a total generation of k = 4000.

Subsequently, the GA performance is graphically shown in Figure 5 with crossover rate, P_c= 0.8 and mutation rates vary from 0.1 to 0.5. It is observed that the GA with crossover rate, P_c= 0.8 and mutation rate, P_m = 0.2 obtains the best result among the others, which fitness is 1060 as can also be seen in Table 4. The number of generation that shows convergence for all sets of experiments is less than 4000.

Table 4: Fitness values when crossover rate = 0.8, with various mutation rates

Parameters	"Best-so-far" fitness
Pc = 0.8, Pm= 0.5	1140
Pc = 0.8, Pm= 0.4	1740
Pc = 0.8, Pm= 0.3	1440
Pc = 0.8, Pm= 0.2	1060
Pc = 0.8, Pm= 0.1	1780

On the other hand, Figure 4 shows the performance of GA when tested with mutation rate, P_m = 0.8, with crossover rate varied from 0.3 to 0.5. It can be concluded that GA which applied a higher rate of mutation together with a lower rate of crossover gained even better result. The GA with mutation, P_m = 0.8 produces the best-so-far solution at 920 each when tested with crossover rate, P_c = 0.3, 0.4 and 0.5. The experiments show that the GA with lower crossover rate reaches convergence faster than GA with higher crossover rate as can be analyzed from Figure 6 and Table 5.

Results from these experiments show that all variations of GA are able to generate good solutions when tested with different crossover rates and mutation rates. However, the better results were obtained with high rates of mutation as

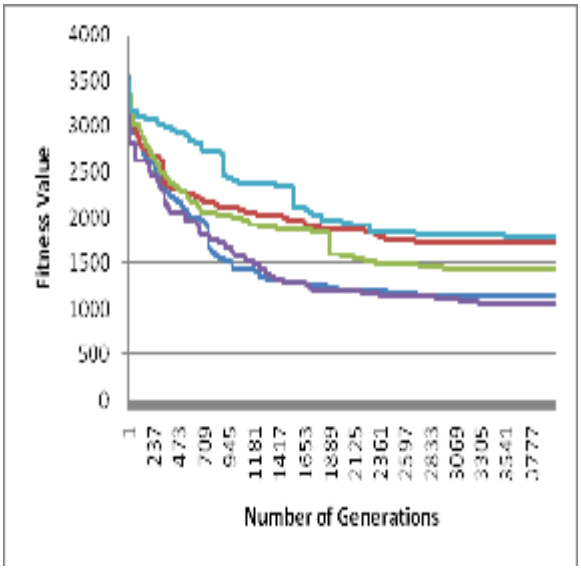

Fig. 5: The GA performance graph when crossover rate = 0.8 with different mutation rates

can be seen when comparing fitness values in Table 4 and Table 5.

Table 5: Number of generations to reach convergence when mutation rate = 0.8 with different crossover rates

Parameters	"Best-so-far" fitness	Number of generations when reached convergence
Pc = 0.3, Pm =0.8	920	2651
Pc = 0.4, Pm =0.8	920	2837
Pc = 0.5, Pm =0.8	920	3262

4 Evaluation and comparison

The GA for the BDSP is effective and efficient as the produced schedules do not violate any hard and soft constraints that are being considered. The computation time accomplished in generating a schedule is fast as compared to that of the manually generated schedule, which took almost one day to obtain a result, but sometimes violated some of the constraints.

It is difficult for the bus service manager to provide a manual schedule each time that fulfills all hard and soft constraints. The manager usually encounters the problem of too many drivers being assigned to break together. Hence, a comparison of the manual schedule with the GA generated schedule is as presented in Table 6. The comparison is based on certain factors such as the violation of hard constraints, length of computation time, and fitness value adopted from [26].

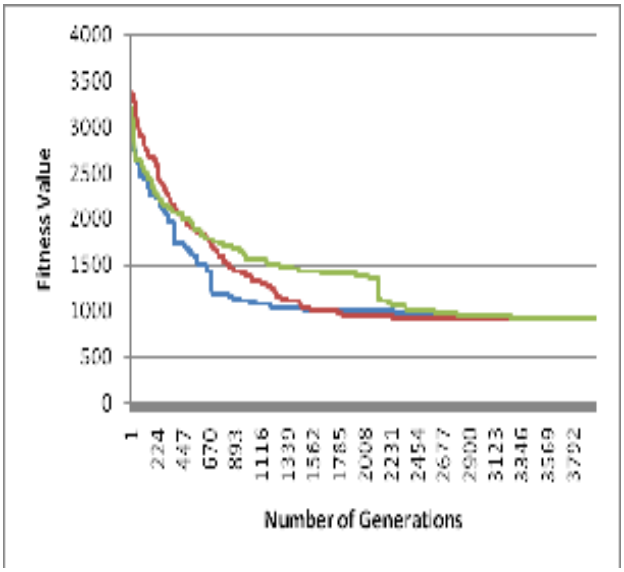

Fig. 6: The GA performance graph when mutation rate= 0.8 with different crossover rate

Table 6: Comparison of manual schedule with GA generated schedule

	Manually Generated Schedule	GA Generated Schedule
Violation of hard constraints	Yes	No
Time consumed to generate a schedule	12 hours	98 minutes
Fitness	4360	920

Referring to Table 6, the GA generated schedule gives a more desirable driver schedule compared to the manually generated driver schedule. Based on discussion with the manager, it is understood that the manually generated schedule constantly violated the hard constraints set by the management. The violation is unavoidable since human consideration is very limited. In addition, it took about 12 hours for the manager to generate a schedule and when the same fitness function was imposed, the manually generated schedule presents a fitness of 4360. Overall, the GA generated schedule is able to produce a good schedule without violating any hard constraints at any time, and in real short time, that is 98 minutes. The best-so-far fitness of the schedule has been achieved as low as 920.

5 Conclusion

We have illustrated an innovative methodology design using Genetic Algorithm approach for a bus driver scheduling problem for a campus environment. The proposed GA generated schedule has been successfully constructed to solve the BDSP in relation to work duty and break period in a day. The implementation of GA concept has actually able to generate not one but a list of acceptable schedules for the BDSP in a short time. The robustness of GA is the key reason for it to be employed in the problem.

The GA approach provides a highly efficient daily schedule for the bus drivers, which subsequently improve the quality of the campus bus service. A fairer work and break assignment can be delivered in daily basis. Eventually, the drivers' job satisfaction could be increased and thus promote better service.

Acknowledgement

The authors would like to thank Universiti Utara Malaysia and the Ministry of Higher Education Malaysia for the financial support that has helped in completing this study.

References

[1] Lim Tze Shung, Razamin Ramli & Haslinda Ibrahim . A Genetic Algorithm Approach to Bus Driver Scheduling System, In Proceedings of the International Conference on Quantitative Sciences and Its Applications (ICOQSIA), 6 - 8 December, Penang, Malaysia, (2005).

[2] Wren, A., & Rousseau, J. M. Bus driver scheduling- An overview. In Proceedings of the 5th Int. Workshop on Comp. Aided Scheduling of Pub. Trans., 6-9 July, Lisbon, Portugal, (1993).

[3] Sousa, J. P., Dias, T. G. & Cunha, J. F. Interactive multi-objective genetic algorithms for the bus driver scheduling problem. In Proceedings of the 10th EURO Working Group on Transportation Meeting and 16th Mini-EURO Conference, (2005).

[4] Beasley, J. E., & Chu, P. C. A genetic algorithm for the set covering problem. European Journal of Operation Research, **94**, 392-404 (1996).

[5] Curtis, S. D., Smith, B. M., & Wren, A. Forming bus driver schedules using constraint programming,Technical Report 99.05, University of Leeds, United Kingdom, (1999).

[6] Wren, A., & Wren, D. O. A genetic algorithm for public transport driver scheduling. Computers & Operations Research, **22**, 101-110 (1995).

[7] Forsyth, P., & Wren, A. An ant system for bus driver scheduling, Technical Report 97.25, University of Leeds, United Kingdom, (1997).

[8] Shen, Y., & Kwan, R. S. K. Tabu search for time windowed public transport driver scheduling, Technical Report, Central China Normal University, (2002).

[9] De Leone, R., Festa, P., & Marchitto, E. (2011). A bus driver scheduling problem: A new mathematical model and a GRASP approximate solution. Journal of Heuristics, **17**, 441-466 (2011).

[10] Chen, Q., & Li, C. An approach to bus-driver scheduling problem. In GCIS '10 Proceedings of the Second WRI Global Congress on Intelligent Systems, **02**, 379-382 (2010).

[11] Yoshihara, I. Scheduling of Bus Drivers' Service by a Genetic Algorithm. Advances in Evolutionary Computing, Natural Computing Series. Berlin: Springer Berlin Heidelberg, (2003).

[12] Holland, John H, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, (1975).

[13] Riedl, A. A versatile genetic algorithm for network planning. In Proceeding Open European Summer School on Network Management and Operation (EUNICE'98), Munich, Germany, (1998).

[14] Rangel-Merino, A., & Lopez-Bonilla, J. L. Optimization method based on genetic algorithm. Apeiron, **12**, 393-408 (2005).

[15] Merad, L., Meriah, S. M., & Bendimerad, F. T. . Genetic algorithm optimization for circular microstrip antenna. Algeria: University of Abou-Bekr Belkad, (2003).

[16] Dias, T. G., Sousa, J. P., & Cunha, J. F. Genetic algorithms for the bus driver scheduling problem: A case study. Journal of Operational Research Society, **53**, 324-335 (2002).

[17] Brasil, A., Pinheiro, P.R., & Coelho, A.L.V. . Towards the Early Diagnosis of Alzheimer's Disease through the Application of a Multicriteria Classification Model. In Popa, R. (ed.) Genetic Algorithms in Applications, Croatia: InTech, (2012).

[18] Lim Tze Shung, Razamin Ramli & Haslinda Ibrahim. An Evolutionary Algorithm Approach to a Bus Scheduling Problem with Break-time consideration. In Proceedings of the 5th Asian Mathematical Conference, Putra World Trade Center, Kuala Lumpur, (2009).

[19] Beasley, D., Bull, D. R., & Martin, R. R. An overview of genetic algorithms: Part 1, Fundamentals. University Computing, **15**, 58-69 (1993).

[20] Wagner, S., Affenzeller, M., & Schragl, D. Trapes and dangers when modeling problems for genetic algorithms. Cybernetics and Systems 79-84 (2004).

[21] Masuchun, R. An application of direct coding genetic algorithm to solving the problem of dynamic rescheduling. In 2nd International Conference on Autonomous Robots and Agents, Palmerston North, New Zealand, 141-145 (2004).

[22] Morz, M., & Musliu, N. Genetic algorithm for rotating workforce scheduling problem. In Proceedings of second IEEE International Conference on Computational Cybernetics, Vienna, Austria, 121-126 (2004).

[23] Spears, W., & De Jong, K. An analysis of multi point crossover. In G. J. E. Rawlins (ed.), Foundations of Genetic Algorithms . San Mateo, CA: Morgan Kaufmann Publishers, 301-315 (1991).

[24] Jaramillo, J. H., Bhadury, J., & Batta, B. On the use of genetic algorithms to solve location problems. Computers & Operations Research, **29**, 761-779 (2002).

[25] Syswerda, G. Uniform crossover in genetic algorithm. In Proceedings of the Third International Conference on Genetic Algorithms (ICGA'), **89**, 10-19 (1989).

[26] Dias, T. M., Ferber, D. F., Souza, C. C., & Moura, A. V. Constructing nurse schedules at large hospitals. International Transaction in Operational Research, **10**, 245-265 (2003).

Razamin Ramli was born in Kedah, Malaysia on 11 August 1962. She obtained her B. Sc. in Mathematics-Statistics from Wichita State University, Wichita, Kansas, USA in 1984, M. Sc. in Applied Mathematics from California State University, Hayward, California, USA in 1989, and PhD in Operational Research-Artificial Intelligence from Universiti Sains Malaysia in 2004. At present, she is an Associate Professor at the Department of Decision Science in the School of Quantitative Sciences, Universiti Utara Malaysia. Her main research interests are in optimization, meta-heuristics, evolutionary algorithms, and planning and scheduling problems. Dr. Ramli is a member of the Management Science and Operations Research Society of Malaysia, and also a member of IACSIT.

Haslinda Ibrahim obtained her bachelor degree in Mathematics in 1992 (Michigan State University, USA), Master in Mathematics in 1996 (Indiana State University) and PhD in Mathematics in 2003 (Southern Illinois University, USA). She is well known for her profound contributions in combinatorics domain especially in Combinatorial Design Theory. Her current research interests include graph theory, combinatorial design theory, enumerative theory and optimization theory. She is also a member for American Mathematical Society (AMS) and Malaysia Mathematical Science Society (PERSAMA). She is also an editorial board for several international journals

Lim Tze Shung finished her undergraduate study in Decision Science and later continued as a postgraduate student in the same field at the Department of Decision Science, School of Quantitative Sciences, Universiti Utara Malaysia. She graduated in 2009 and her area of research was specifically on Genetic Algorithm.