Kobe University Repository : Kernel

PDF issue: 2026-11-30

Solving a timetable problem using distributed genetic algorithm

Su, Yancong

Namba, Michihiro

Murao, Hajime

(Citation)
ICIC Express Letters,3(4A):1055-1060

(Issue Date)
2009-12

(Resource Type)
journal article

(Version)
Accepted Manuscript

(URL)
https://hdl.handle.net/20.500.14094/90001842

KOBE UNIVERSITY

ICIC Express Letters
Volume **3**, Number **3**, September **2009**

ICIC International ©2009 ISSN 1881-803X
pp. 1–6

SOLVING A TIMETABLING PROBLEM USING DISTRIBUTED GENETIC ALGORITHM

Yancong Su[1], Michihiro Namba[2] and Hajime Murao[1]

[1]Graduate School of Intercultural Studies
Kobe University
1-2-1 Tsurukabuto, Nada, Kobe, Japan
gensou.so@stu.kobe-u.ac.jp, murao@i.cla.kobe-u.ac.jp

[2]Faculty of Humanities
Yamanashi Eiwa College
8-8-8 Yokonecho, Kofu, Japan
namba@y-eiwa.ac.jp

Received April 2009; accepted June 2009

Abstract. *A timetabling problem is one of the combinatorial optimization problems, where large amount of lectures, rooms, and constraint conditions cause a great difficulty. We apply a distributed Genetic Algorithm (DGA) to a timetabling problem. In DGA, several GAs run in parallel and exchange solution candidates each other. Which can easily utilize multi-processor computers to reduce computation time and is also useful to overcome the premature convergence problem. The computer experiments have been done on generating a timetable for Yamanashi Eiwa College. It is shown that the performance of DGA depends on the number of sub-populations and the migration interval. However, in any configurations, DGA could generate quite good timetables scored over 6,300 of fitness values while an ordinal single population GA converged in early generations to timetables scored around 6,100. It is more than twice of the really used one which scores 3,045 only. The results show the effectiveness of DGA for the timetabling problem.*
Keywords: Timetabling, Genetic algorithm (GA), Distributed GA (DGA)

1. **Introduction.** Generating a good quality timetable for lectures in a college should consider many constraint conditions such as students' choice trends, teachers' schedules, classrooms' sizes, and so on. A satisfaction of students and teachers is also important. Generally, scheduling problems like timetabling are treated as combinatorial optimization problem. However, it is difficult to find the optimal solution for a timetabling problem because of the condition requirement stated above.

Hence, heuristic methods which search suboptimal solutions are effective in solving them. For instance, Nearest Neighbor (NN), Simulated Annealing (SA) and Taboo Search (TS) are typical methods. However, they search in the set of all solutions by the perturbation and acceptance of only one candidate of solutions. It is not appropriate to apply them to a timetabling problem, which has a much larger solution space then their search ranges. On the other hand, Genetic Algorithm (GA) makes many candidates of solutions simultaneously and stochastically. Since this feature of multi-point search, it is effective in scheduling problems. GA can be applied into various applications when selecting some parameters appropriately. However, it has problems such as requiring considerably long computation time and premature convergence on poor solution. Moreover, parameters selected have a great influence on accuracy of solutions [1].

Distributed GA (DGA) is a parallel model of GA, and has a migration operation imitating the immigration activity of animals. It can not only obtain a better solution

fast but maintain the diversity of solution, by searching in parallel and exchanging solutions between GAs. It has been reported that DGA is effective in various combinatorial optimization problems such as TSP, KP, job-shop scheduling, and so on [2, 3, 4, 5, 6].

In this paper, we apply DGA to making a timetable for college lectures, and consider the effect of the parameters on the quality of the results, since it has been reported that the performance of DGA depends on parameter selection [7]. In addition, we examine the usefulness in comparison with the really existing timetable, and demonstrate the excellence in comparison with results of Single Population GAs (SPGA).

2. Distributed GA.

2.1. Conventional GA.

The concept of GA is to produce good quality solutions in an object problem by iterating selection, crossover, mutation and other genetic operations in every generation. The flow of the conventional GA is as follows:

1. Start with a randomly generated population of n chromosomes (which encode solution candidates to a problem).
2. Calculate the fitness value $f(x)$ of each chromosome x in the population.
3. Iterate the following steps until n offspring have been created.
   a. Select a pair of parent chromosomes from the current population. Selection is done "with replacement", meaning that the same chromosome can be selected more than once to become a parent.
   b. With crossover probability, crossover the pair at a randomly chosen point to form two offspring.
   c. Mutate the two offspring at each locus with mutation probability, and place the resulting chromosomes in the new population.
4. Replace the current population with the new population.
5. Go to step 2

Compared with the traditional optimization methods, GA has a superior ability in global search by multi-point searching, and can reduce convergence on local solutions by mutation. However, GA has some problems of a great deal of computation time, risk of early convergence on local solutions and difficulty of setting parameters.

2.2. Distributed GA.

DGA, which is a parallel model of GA, can overcome the problems described above. In DGA, the population is distributed to some sub-populations called islands. DGA individually performs the genetic operations in each island which enables to utilize multi-processor system becoming popular. In addition, it requires a new operation called migration to exchange information between the islands. The flow of DGA is described as follows:

1. Start with a randomly generated population of n chromosomes (candidate solutions to a problem).
2. Distribute the initial population averagely to some islands. It enables to assign them to processors in multi-processor system.
3. Perform the genetic operations (same as the step 2 - 5 of the conventional GA) individually in each sub-population.
4. According to a migration interval, select some individuals at a migration rate, and exchange them with another island.
5. Go to step 3.

The distribution algorithm enables DGA to search solutions independently in each island at the same time. Hence DGA can search solutions much efficiently than conventional GA. Moreover, the differences between the islands can maintain the diversity of solution in the whole population.

However, it is not easy to determine parameters (migration interval, migration rate, and so on) in DGA as well as conventional GA parameters. These parameters have a great influence on the performance of DGA [7].

3. Timetabling Problem.

3.1. **Hard Constraint Conditions.** Timetabling generally requires a lot of professional knowledge. Hence teachers make timetables by hand in many schools. However, since it has many constraint conditions, it consumes a lot of time and energy. Among those constraint conditions, there are ones which timetable must satisfy. We call them "Hard Constraint Conditions". In this paper, we consider the following:

- All lectures must be assigned to a timetable.
- All different lectures cannot be held in the same room at the same time.
- All teacher cannot have different lectures at the same time.
- All teacher cannot be assigned to their inconvenient hours such as meeting.

3.2. **Soft Constraint Conditions.** There are also conditions which timetable does not have to satisfy but better to satisfy. We call them "Soft Constraint Conditions". We consider nineteen "Soft Constraint Conditions" in this paper. Their examples are as follows:

- Compulsory subjects cannot be held at the same time.
- One teacher may not have lectures continuously.
- The lectures of the same field may be assigned to different hours.
- Teachers may prefer specific time slots.
- Teachers may prefer specific rooms.

4. System Description.

4.1. **Input Data and Encoding of a Chromosome.** A chromosome contains lecture numbers, and the initial ones are encoded randomly with unique lecture numbers in their timetable frames constituted by days of week, periods and classrooms. Other data can be referred from the lecture number, because the input database consists of different number of lectures, different number of teachers, different number of courses, different number of room types and flag of compulsory subjects. Figure 1 shows a concept of encoding of a chromosome. According to the actual situation of Yamanashi Eiwa College, we digitized 396 lectures (No.0 - 395) into 5 days a week, 5 periods a day and 23 classrooms as the timetable frame for encoding. The value of a space in chromosome where is not any lectures assigned is -1.

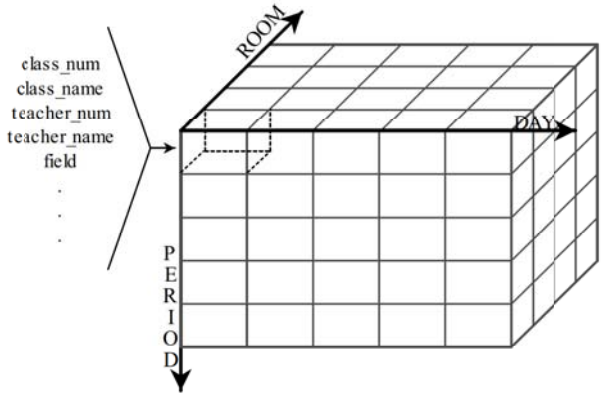


FIGURE 1. Chromosome

4.2. **Fitness Value Evaluation Based on Constraint Conditions.** Fitness value $f(x)$ for a solution candidate x obtained from a chromosome is calculated by

$$f(x) = f_{\max} - \sum_{c} p(c) \cdot \delta(c, x) \tag{1}$$

where $f_{\max}$ gives the maximum of the fitness value, and $p(c)$ is a penalty function for a soft constraint condition c. $\delta(c, x)$ is a function gives 1 if x violates the condition c and 0 otherwise. $p(c)$ is determined for each condition beforehand, which reflects priority of the conditions, i.e. larger value for conditions with higher priority, and smaller value for lower priority.The following are examples of $p(c)$ used in experiments.

- One teacher has lectures continuously → 10 points
- Compulsory subjects are held at the same time slot → 30 points
- The lectures of the same field are held at the same time slot → 20 points

The chromosomes which do not satisfy the hard constraint conditions are treated as lethal ones which have no chance to survive in the next population.

4.3. **Genetic Operations and Migration.** Genetic operations used in this study are "selection", "crossover" and "mutation", which are the same in conventional GA. The elitest strategy and the roulette selection are used for "selection" operation. To avoid the duplication of lecture number in individuals, the method showed in Fig.2 was used in crossover operation. It selects some lectures randomly to be kept and replaces others in order.

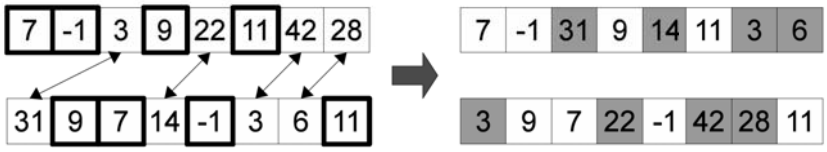


FIGURE 2. Crossover

Mutation operation selects an individual randomly from population, and exchanges its two lectures also chosen randomly.

In migration, chromosomes are selected according to fitness value and migration rate, and moved to the next island. The migrants from the last island immigrate into the first one.

5. Experimental Results.

5.1. **Experiments with SPGA.** At first, in order to examine a usefulness of GA in timetabling problem, we apply the system with SPGA to making a timetable of a college in Japan. The parameters of SPGA in the experiment are shown in Table 1. The maximum fitness value $f_{\max}$ is 8,000 points which is a large enough score in this experiment. Table 2 shows the maximum and averages of fitness values in five times experiments. We also calculate the fitness value for the timetable used in Yamanashi Eiwa College. It becomes 3,045 points, almost the half of the experiment results. It indicates that GA is effective in the this problem.

TABLE 1. Parameters of SPGA

Generation	150,000
Population	200
Mutation Rate	0.05

TABLE 2. Maximum and average of SPGA

	Maximum	Average
1st	6,130	6,130
2nd	6,195	6,195
3rd	6,245	6,245
4th	6,060	6,060
5th	6,055	6,055
Avg.	6,137	6,137

TABLE 3. Parameters of DGA

Generation	150,000
Population	200
Mutation Rate	0.05
Migration Intervals	1, 2, 5, 10
Island	2, 4, 10
Migration Rate	0.4

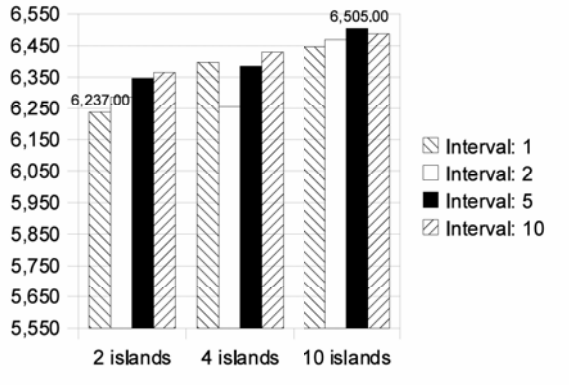


FIGURE 3. Total average of maximum

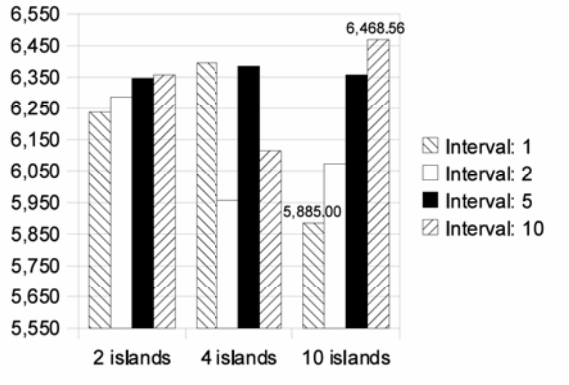


FIGURE 4. Total average of average

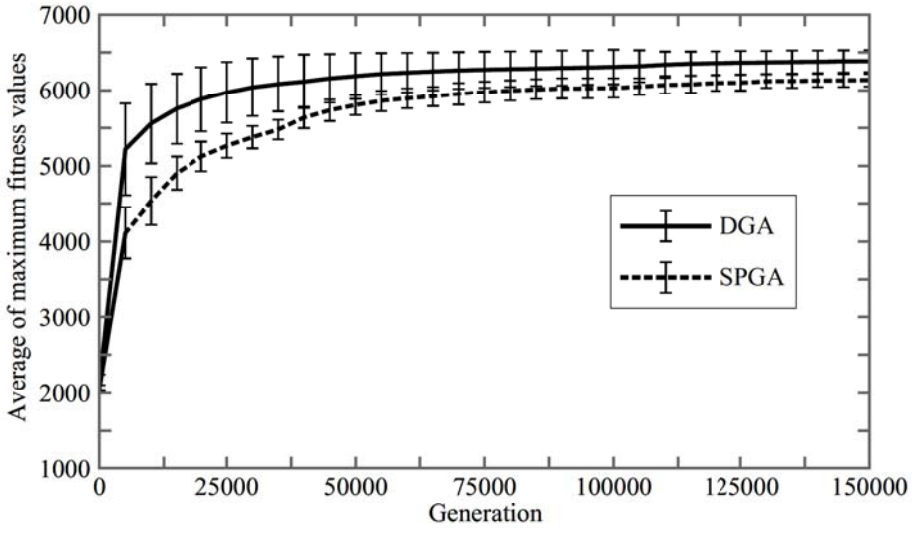


FIGURE 5. Fitness Value Transition in DGA and SPGA

5.2. **Experiments with DGA.** Next, we demonstrate the similar experiments with DGA. In order to examine the influence of DGA parameters on the quality of solution, we try different set of parameters in DGA, which are shown in Table 3. As a result, Figure 3 and 4 show the total average of maximum and average in each experiment.

Figure 3 indicates that the maximum becomes large according to increase of number of islands and migration interval, while few opposite results was observed when the number of island is more than two. This result implies that the balance of DGA parameter setting is not simple. Poor combinations of interval and island may affect the solution search ability. In any case, we can find that the quality of solutions in DGA is better than SPGA.

Differences in the maximum shown in Fig.3 and the average shown in Fig.4 indicate the diversity of the solution candidates in the population is maintained with DGA. It is not observed in the results of SPGA shown in Table 2. Dividing the population into islands and making progress individually seem to contribute well for the maintainance of the diversity and finding better solutions.

Figure 5 shows the average and standard deviation of the maximum fitness values for SPGA and DGA. According to generational change in both of them, the fitness value is improved clearly. It indicates that they are effective in the timetabling problem. By comparing them, we can observe that DGA evolves solutions higher quality and faster than SPGA with maintaining the diversity of solution. It confirmed that DGA provide a better performance than SPGA in timetabling problem.

6. **Conclusion.** In this paper, we applied DGA to making a timetabling system, and demonstrated it by using a curriculum of a college in Japan. To begin with, we could observe GA was effective in timetabling problem. Moreover, we could observe the usefulness of DGA by comparison with SPGA. In future, we will include much more constraint conditions, and examine the optimal parameters.

REFERENCES

[1] H. UEDA, D. OUCHI, K. TAKAHASHI and T. MIYAHARA, Comparisons of Genetic Algorithms for Timetabling Problems, *Transactions of the Institute of Electronics, Information and Communication Engineers*, vol.J86-D-I, no.9, pp.691-701, 2003.

[2] M. MIKI, T. HIROYASU and Y. NAKAMURA, A Study on Parallel Distributed Genetic Algorithms: Discussion on a Stepping Stone Population Model of Distributed Genetic Algorithms, *Transactions of the Japan Society of Mechanical Engineers (A)*, vol.J86-D-I, no.9, pp.2177-2183, 1999.

[3] M. MIKI, T. HIROYASU and Y. HANADA and T. MIZUTA, Distributed Genetic Algorithm with Centralized Multiple Crossovers Applied to Traveling Salesman Problems, *Transactions of the Institute of Systems*, vol.16I, no.12, pp.607-615, 2003.

[4] M. MIKI, T. HIROYASU and Y. HANADA, Application of a Distributed Genetic Algorithm to Jobshop Scheduling Problems, *The science and engineering review of Doshisha University*, vol.44, no.2, pp.119-125, 2003.

[5] K. KIM, M. GEN and M. KIM, Adaptive Genetic Algorithms for Multi-resource Constrained Project Scheduling Problem with Multiple Modes, *International Journal of Innovative Computing, Information and Control*, vol.2, no.1, pp.41–49, 2006

[6] F. Xhafa, J. Carretero and A. Abraham, Genetic Algorithm Based Schedulers for Grid Computing Systems, *International Journal of Innovative Computing, Information and Control*, vol.3, no.5, pp.1053–1071, 2007

[7] Y. MAEDA and Q. LI, Parallel Genetic Algorithm with Adaptive Genetic Parameters Tuned by Fuzzy Reasoning, *International Journal of Innovative Computing, Information and Control*, vol.1, no.1, pp.95-107, 2005