GENETIC ALGORITHM APPROACH TO MULTIOBJECTIVE RESCHEDULING ON PARALLEL MACHINES

Hitoshi Iima *

* *Kyoto Institute of Technology*
Matsugasaki, Sakyo-ku, Kyoto 606–8585, Japan
iima@si.dj.kit.ac.jp

Abstract: This paper deals with multiobjective rescheduling on parallel machines for alteration of due date. The objectives of this rescheduling problem are to minimize both the total tardiness and a difference of schedule from that before the alteration. Genetic algorithms with various individual descriptions are applied to the problem in order to find the Pareto optimal solutions. In particular, two decoding procedures are proposed to obtain schedules with a smaller difference. *Copyright © 2005 IFAC*

Keywords: Scheduling algorithms, Genetic algorithms, Multiobjective optimizations, Manufacturing systems

1. INTRODUCTION

In real manufacturing systems, alteration of problem condition such as change of due date and addition of job often obliges to revise a schedule worked out previously. Vieira *et al.* (2003) present definition appropriate for most applications of such rescheduling manufacturing systems, and review various methods to solve the rescheduling problems. The aim of most of these methods is to optimize only an original objective function, say the total tardiness. Consequently, the schedule obtained by such a method may be very different from that before the alteration. The difference of schedule incurs time and costs in re-preparing the processing for the case where the problem condition is altered after preparation of processing.

As studies considering the schedule difference, Watatani and Fujii (1992) define a problem in which the objective function is a weighted sum of the makespan and the schedule difference. Abumaizar and Svestka (1997) consider a problem for a breakdown of machine, and obtain a schedule with a small difference by rescheduling only the operations affected by the breakdown of machine. In our earlier paper, a rescheduling problem is considered in a job shop (Iima, 2005).

This paper deals with a two-objective rescheduling problem for parallel machines in the case where the due dates of some jobs are altered. The objectives of this problem are to minimize the total tardiness and the schedule difference. Genetic algorithms (GAs) (Goldberg, 1989) with various individual descriptions are applied to the problem for obtaining the Pareto optimal solutions. In particular, two decoding procedures are proposed to obtain schedules with a smaller difference. These individual descriptions are compared through the computational result.

2. GA FOR A CONVENTIONAL PROBLEM

In this section, a conventional scheduling problem P^* before the alteration is considered, and the computational result for P^* is shown by applying GAs with three individual descriptions. This result is shown to compare with that for the rescheduling problem, which will be considered in the next section.

2.1 Problem Statement

A set of I kinds of jobs J_i $(i = 1, 2, \cdots, I)$ is processed by using K identical machines M_k $(k = 1, 2, \cdots, K)$ arranged in parallel. A job J_i is completed by processing on one of these machines, and the processing time is given as PT_i irrespective of the machines. In addition, J_i should be completed by the due date D_i^*. A machine can process at most one job at a time, and no preemption of job is allowed.

The problem P^* is to determine both the processing machine number $m^*(i)$ for J_i and the processing order of jobs for each machine in such a way that the total tardiness F_1^* should be minimized. The objective function F_1^* is formulated as

$$F_1^* = \sum_{i=1}^{I} \max(c_i^* - D_i^*,\ 0) \tag{1}$$

where c_i^* is the completion time of J_i and can be calculated from the decision variables.

2.2 GA for the Parallel Machine Problem

In general parallel machine problems, there are two kinds of decision variables: the processing order and the processing machine. Therefore, a direct genotype in a GA is expressed by using two strings. In this genotype, one of them represents the processing order, and the other represents the processing machine. However, the solution space explored by this GA is enormous, and it may take a long computation time to obtain a suboptimal solution. In order to do it in a short time, a genotype in a GA may be expressed by using a single string representing the processing order or the processing machine. In this GA, the remaining decision variables are determined by using a suitable heuristic rule in the decoding procedure.

In this paper, three GAs (GA1, GA2 and GA3) are designed on the basis of the above explanation. In GA1 the two strings are used. The string S1 representing the processing order is expressed by sequencing job numbers $\{i\}$. The string S2 representing the processing machine is expressed by sequencing the processing machine numbers $\{m^*(i)\}$ in the order of i. In the decoding procedure, the processing machine is straightforwardly determined by S2, and then the schedule on each machine is determined by processing from the left job in S1. As the crossover operation, the set partition crossover (Shi *et al.*, 1996) and the uniform crossover are used for S1 and S2, respectively. As the mutation operation, the shift mutation and the standard one-point mutation are used for S1 and S2, respectively.

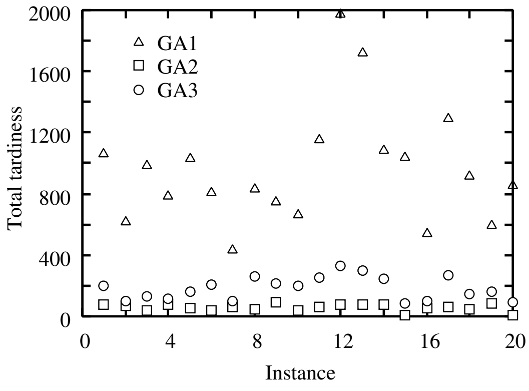


Fig. 1. Computational result for the conventional problem P^*

For GA2 only the string S1 is used. The schedules of jobs are determined in the order of S1. The machine on which a job J_i is processed is assigned in such a way that the completion time c_i^* is minimized.

For GA3 only the string S2 is used. The processing order for each machine is determined by the earliest due date rule (EDD).

GA1, GA2 and GA3 use the common flow. Each initial solution in the population is generated at random. As the selection operation, the minimal generation gap (Sato *et al.*, 1997), which is of the steady-state type, is used. The crossover and mutation operations are applied at every generation.

2.3 Computational Result

Twenty instances are used for evaluating the performance of GA1, GA2 and GA3. Each of these instances belongs to one of instance sets with four scales, and the scales of Instances 1–5, 6–10, 11–15 and 16–20 are (I, K)=(100,5), (100,10), (200,5) and (200,10), respectively. The processing time PT_i in these instances is given randomly from the range of 10 to 99. Moreover, the due date D_i is appropriately given in such a way that the total tardiness F_1^* in a reasonable schedule is less than one hundred.

In the GAs, there are two parameters: the population size PS and the final generation FG. The values of them are decided through a preliminary calculation as follows:

$PS = 300,\ FG = 15000$ (for Instances 1–10),
$PS = 300,\ FG = 30000$ (for Instances 11–20).

Each GA is performed one hundred times with various random seeds for an instance.

Fig. 1 shows the average of the total tardiness obtained by each GA. It is confirmed from this figure that GA2 is the best method. GA3 is worse than GA2, because the optimal processing order is not necessarily obtained by EDD in GA3. GA1 is much worse than GA2 and GA3 under the final generation given. This is because the solution space in GA1 is enormous, as mentioned above.

3. PARALLEL MACHINE RESCHEDULING PROBLEM

It was confirmed from the previous section that a suboptimal solution (schedule) S^* is obtained by applying GA2 to the conventional scheduling problem P*. Consider the situation that the production has been prepared on the basis of S^*. After the preparation, some due dates are altered. The due date of job J$_i$ after the alteration is denoted as D_i. Since S^* may not be optimal for D_i no longer, it should be revised. In this situation, the second objective function is defined as the magnitude of difference between S^* and a schedule S revised.

The rescheduling problem P coped with in this paper is to determine both the processing machine number $m(i)$ for J$_i$ and the processing order in such a way that both the total tardiness F_1 and the schedule difference F_2 should be minimized. The objective functions in P are formulated as follows.

$$\min \begin{pmatrix} F_1 \\ F_2 \end{pmatrix} = \begin{pmatrix} \displaystyle\sum_{i=1}^{I} \max(c_i - D_i,\ 0) \\ A + wB \end{pmatrix} \qquad (2)$$

$$A = \sum_{i=1}^{I} |a_i \cap a_i^*| \qquad (3)$$

(The symbol | | means the number of elements)

$$B = \sum_{i=1}^{I} b_i \qquad (4)$$

$$b_i = \begin{cases} 1 \ (\text{for } m(i) \neq m^*(i)) \\ 0 \ (\text{for } m(i) = m^*(i)) \end{cases} \qquad (5)$$

$$(i = 1, 2, \cdots, I)$$

c_i : Completion time of J$_i$ in S.
w : Weight.
a_i: Set of jobs processed on M$_{m(i)}$ before J$_i$ in S.
a_i^*: Set of jobs processed on M$_{m(i)}$ after J$_i$ in S^*. It is noted that $a_i^* = \{\phi\}$ for $m(i) \neq m^*(i)$.

The variable A means the schedule difference for the processing order, and is given on the basis of Watatani's definition (Watatani and Fujii, 1992). If the machine on which a job is processed in S is different from that in S^*, the schedule difference of the job is zero.

The variable B means the schedule difference for the processing machine. Moreover, the parameter w is the weight between A and B.

4. GA FOR THE RESCHEDULING PROBLEM

4.1 Individual Description

Because the decision variables in the rescheduling problem P are the same as those in the conventional problem P*, GA1, GA2 and GA3 are also applicable to P. The schedule difference F_2, however, is added as the second objective function in P. The heuristic rules used in the decoding procedures of GA2 and GA3 are introduced to obtain solutions with a smaller value of the total tardiness F_1, and may not be suitable for obtaining solutions with a smaller value of the schedule difference F_2. Thus, new heuristic rules in these decoding procedures are proposed in consideration to not only F_1 but also F_2. GA2 and GA3 with these new decoding procedures are called GA2-ND and GA3-ND, respectively.

Decoding Procedure in GA2-ND In the case where the processing order of a left job J$_{i'}$ in the string of GA2 is changed, the processing machines of many jobs decoded after J$_{i'}$ tend to be changed. Consequently, the schedule difference B for the processing machine becomes larger. In order to overcome this disadvantage, a job J$_i$ in a schedule S for P should be assigned to the same machine M$_{m^*(i)}$ as the best schedule S^* for P*. The total tardiness F_1, however, remains large, if all the jobs are assigned to M$_{m^*(i)}$. Thus, in GA2-ND, J$_i$ is assigned to M$_{m^*(i)}$, if the tardiness of J$_i$ is zero in this case. If not, J$_i$ is assigned to the machine such that the completion time c_i of J$_i$ is minimized.

Decoding Procedure in GA3-ND Although the processing order in GA3 is determined by EDD, the jobs in S^* are not necessarily processed in the order of due date. This is because S^* is obtained by means of GA2. Therefore, the schedule difference A for the processing order becomes larger necessarily in GA3. In order to overcome this disadvantage, the jobs assigned to the same machine as S^* in GA3-ND are sequenced in the processing order of S^*, and the incomplete processing order is first generated. However, if the processing order of job of which the due date became earlier is unchanged, the tardiness of the job may not become smaller in the incomplete processing order. Such a job should be processed earlier. Thus, the jobs with $D_i < D_i^*$ or $m(i) \neq m^*(i)$ are sequenced in the order of due date, and then the processing order in GA3-ND is generated by inserting these jobs into the incomplete processing order.

4.2 Initial Population

Although the initial population in GAs is generated randomly for optimization problems in general, S^* can be utilized for P. Since $F_2 = 0$ for $S = S^*$, one of the Pareto optimal solutions is already obtained for P. Therefore, S^* is used as one of PS initial solutions in the proposed GAs. Then, the $PS-1$ initial solutions are generated by applying a mutation operation to S^*. These initial solutions are close to S^* in the solution space.

4.3 Flow of GA

Most of selection operations for multiobjective optimization problems are based on the multiobjective GA (MOGA) (Deb, 2001). In MOGA the rank of solution is given on the basis of the number of solutions which dominate itself, and solutions close to the Pareto-optimal front tend to be selected. In a GA with such a selection operation, solutions out of edge of incumbent non-dominated solution set are not actively explored. Even if this GA is applied to P, it is hard to obtain Pareto optimal solutions with a smaller F_1 and a larger F_2 because the initial solutions are around S^*.

In this paper, a selection operation, selection by area ranking (SAR), is used to obtain diverse Pareto optimal solutions. In SAR, solutions are selected by using the solution space ranked on the basis of the non-dominated solution set X in the solutions explored by the incumbent generation. The solution space is ranked as follows.

Step 1 Find the area in which all the solutions are not dominated by any solution of X, and set the rank of the area to one.

Step 2 Number all the non-dominated solutions of X in the order of F_1, and set the score S_m of m-th solution to m $(S_m = m)$.

Step 3 For the area in which the solution x is dominated by at least one non-dominated solution of X, and set the rank of the area to $\displaystyle\sum_{m' \in X_{sub}} S_{m'} + 1$, where X_{sub} is the set of the non-dominated solution numbers dominating x.

A solution with a small rank is close to the Pareto-optimal front, or has a small F_1. An example of area ranking is shown in Fig. 2 for $|X| = 4$.

At every generation in the proposed GA, two solutions are first picked up as the parents. While

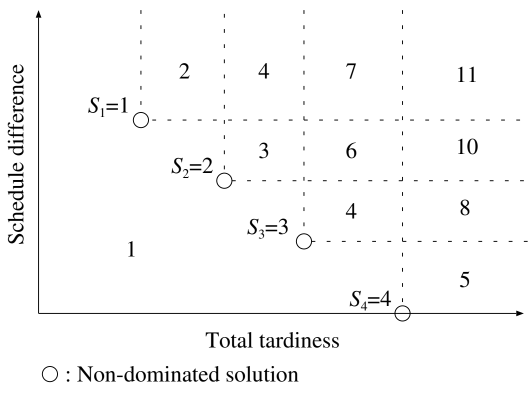


Fig. 2. Example of area ranking

one of them is picked up randomly from the population, the other is picked up randomly from X in order to explore an area around X intensively. The former solution is removed from the population. Next, two children are generated from the parents by means of the crossover operation. Moreover, two other children are generated from the parents by means of the mutation operation. Next, the two solutions with the smallest ranks are selected from the two parents and the four children, and then are added to the population. If the candidates for the second solution selected are plural, the solution with the minimum value of F_1 is selected from these candidates. In this selection, the population size increases, because only a single solution is removed from the population. In order to prevent this, another solution selected randomly is removed from the population before the two best solutions are added to the population.

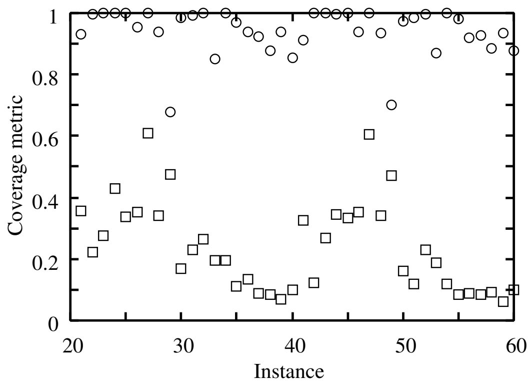
5. COMPUTATIONAL RESULT FOR THE RESCHEDULING PROBLEM

The effectiveness of five GAs is examined through the computational result.

5.1 Experimental Setup

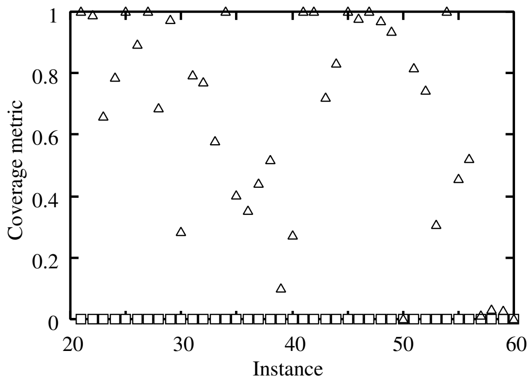
In order to generate the instances of rescheduling problem P, the due date is altered for a few jobs on the basis of the best schedule S^* for the conventional problem P*. These jobs are selected randomly from the jobs such that the tardiness in S^* is zero and the completion time c_i^* in S^* is larger than half of maximum completion time (makespan). The number of the jobs is two and four for I=100 and 200, respectively. The due date D_i of each of the jobs $\{J_i\}$ is altered to $0.9c_i^*$. Hence, they become tardy for $S = S^*$.

Two kinds of values, one and ten, are used for the weight w between the schedule difference of processing order and that of processing machine. Instances 21–40 and 41–60 are numbered for w=1



○ : C(GA2-ND,GA2) □ : C(GA2,GA2-ND)

Fig. 3. Comparison between GA2-ND and GA2



△ : C(GA3-ND,GA3) □ : C(GA3,GA3-ND)

Fig. 4. Comparison between GA3-ND and GA3

and 10 by generating from Instances 1-20, respectively.

The parameters in the GAs for P are the same as those for P*. Each GA is performed one hundred times with various random seeds for an instance.

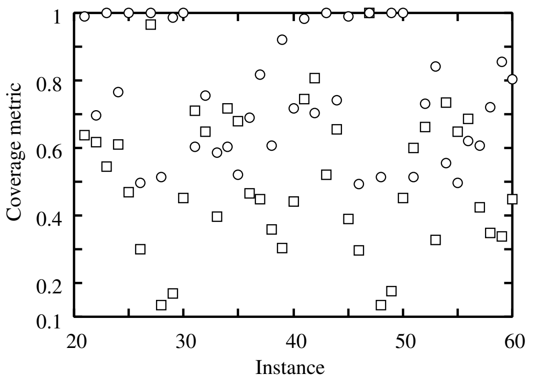
The GAs are evaluated by the coverage metric (Deb, 2001) which means relation of domination between the solution sets obtained by two methods. The coverage metric $C(\mathrm{M1}, \mathrm{M2})$ of Method 1 (M1) to Method 2 (M2) is defined by

$$C(\mathrm{M1}, \mathrm{M2}) = \frac{|\{x_2 \in X_2; \exists x_1 \in X_1 : x_1 \succeq x_2\}|}{|X_2|} \quad (6)$$

where X_1 and X_2 are the solution sets obtained by M1 and M2, respectively. Moreover, $x_1 \succeq x_2$ means that Solution x_1 dominates Solution x_2 or has the same objective values as x_2. If C(M1,M2) > C(M2,M1), M1 is better than M2.

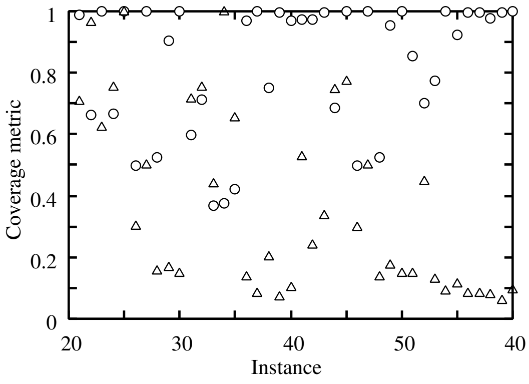
5.2 Result and Discussion

First, the effectiveness of the proposed decoding procedures is examined. Fig. 3 shows the average coverage metric between GA2-ND and GA2. Because C(GA2-ND,GA2) > C(GA2,GA2-ND) in all the instances, the proposed decoding procedure is effective. In particular, the difference between



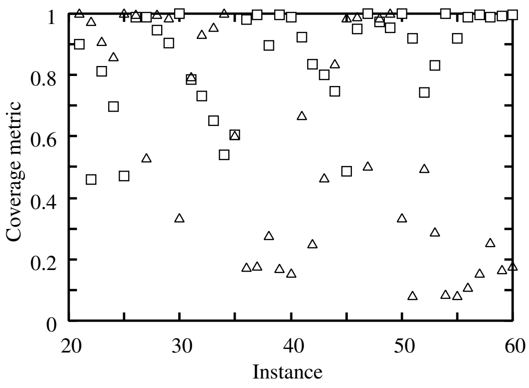
□ : C(GA1,GA2-ND) ○ : C(GA2-ND,GA1)

Fig. 5. Comparison between GA1 and GA2-ND



○ : C(GA2-ND,GA3-ND) △ : C(GA3-ND,GA2-ND)

Fig. 6. Comparison between GA2-ND and GA3-ND

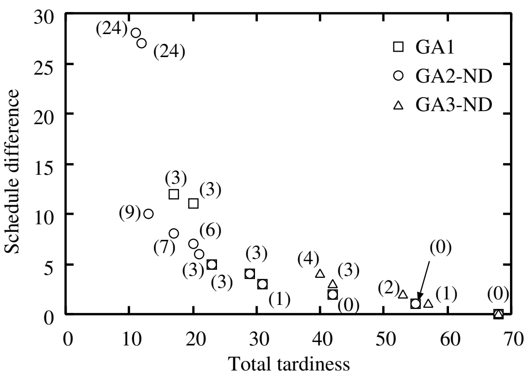


△ : C(GA3-ND,GA1) □ : C(GA1,GA3-ND)

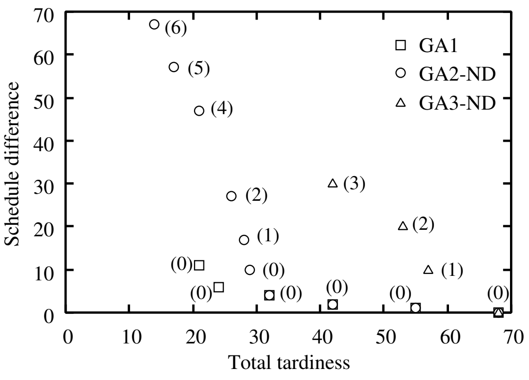
Fig. 7. Comparison between GA3-ND and GA1

these GAs is large for Instances 31–40 and 51–60 in which the number of jobs is large. As mentioned in Subsection 4.1, in the case where the processing order of a left job in the string of GA2 is changed, the processing machines of many jobs tend to be changed. Therefore, the schedule difference B for the processing machine becomes much larger in these instances.

Next, Fig. 4 shows the average coverage metric between GA3-ND and GA3. Because GA3-ND is better in almost all instances, the proposed decoding procedure is effective. Furthermore, C(GA2,GA2-ND)=0 in all the instances.



(a) Instance 20 (w=1)



(b) Instance 40 (w=10)

Fig. 8. Solution set obtained (The value in brackets means the schedule difference B)

Next, GA1, GA2-ND and GA3-ND are compared. Figs. 5, 6 and 7 show the average coverage metric between two of them, respectively. It is confirmed from these figures that GA2-ND is the best method. From the viewpoint of the coverage metric between GA3-ND and GA1, GA3-ND is better than GA1 for Instances 21–40 with w=1, and GA1 is better for Instances 41–60 with w=10. While GA1 is much worse than GA3 for the conventional problem P*, GA1 is as good as GA3 for the rescheduling problem P.

Next, solution sets obtained by GA1, GA2-ND and GA3-ND are shown in Fig. 8 for Instances 20 and 40 in which only the weight w is different. These are results in a single trial. It is found from the figure that the solution set obtained by GA2-ND is spread and is distributed uniformly. A decision maker can confirm the total tardiness and the schedule difference of each schedule by the figure, and determine a desirable schedule according to the situation at that time. The value in brackets in the figure means the schedule difference B for the processing machine. Solutions with a smaller B are obtained for the instance with the larger w.

Finally, the solution sets obtained by the three GAs are compared in Fig. 8. The solution set by GA1 is the same as that by GA2-ND in the range of larger values of the total tardiness F_1. On the other hand, no solution is obtained by GA1 in the range of smaller F_1. As for GA3-ND, the solutions obtained are not good for w=10.

6. CONCLUSION

This paper has dealt with two-objective rescheduling on parallel machines for alteration of due date. The aim of this problem is to minimize the schedule difference as well as the total tardiness. The GAs with five individual descriptions have been applied to find the Pareto optimal solutions in the problem. In particular, the decoding procedures have been proposed to obtain solutions with a small schedule difference. It is concluded from the computational result that solutions close to the Pareto-optimal front are obtained by the GA in which the decoding procedure is designed so as to process a job on the same machine as that for the schedule before the alteration.

This research was partially supported by the Ministry of Education, Culture, Sports, Science and Technology, Grant-in-Aid for Young Scientists (B), 15760292.

REFERENCES

Abumaizar, R.J. and J.A. Svestka (1997). Rescheduling job shops under random disruptions. *International Journal of Production Research*, **35**, 2065–2082.

Deb, D. (2001). *Multi-Objective Optimization using Evolutionary Algorithm.* John Wiley & Sons, Ltd., Chichester.

Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley.

Iima, H. (2005). Proposition of Selection Operation in a Genetic Algorithm for a Job Shop Rescheduling Problem. *Proc. of Third International Conference on Evolutionary Multi-Criterion Optimization*, 721–735.

Sato, H., I. Ono and S. Kobayashi (1997). A new generation alternation model of genetic algorithms and its assessment. *Transactions of the Japan Society for Artificial Intelligence*, **12**, 734–744 (in Japanese).

Shi, G., H. Iima and N. Sannomiya (1996). A new encoding scheme for solving job shop problems by genetic algorithm. *35th IEEE Conference on Decision and Control*, **4**, 4395–4400.

Vieira, G.E., W.H. Jeffrey and L. Edward (2003). Rescheduling manufacturing systems: A framework of strategies, policies and methods. *Journal of Scheduling*, **6**, 39–62.

Watatani, Y. and S. Fujii (1992). A study on rescheduling policy in production system. *JAPAN/USA Symposium on Flexible Automation*, **2**, 1147–1150.