A Method of Solving Scheduling Problems Using Genetic Algorithm with Improved Lagrangian Relaxation Method

Xiaofei Wang[†] *, Wei Wang*[†] *and Hiroki Tamura*[††] *Zheng Tang*[†]*,*

[†]Faculty of Engineering, University of Toyama, Toyama-shi, 930-8555 Japan
[††]Department of Electrical & Electronic Engineering, University of Miyazaki, Miyazaki-shi, 889-2192 japan

Summary
In this paper, an improved genetic algorithm is proposed for flow shop scheduling problems. The proposed method is improved by Lagrangian relaxation method using multipliers which can be adjusted during the search process. The simulation results based on some flow shop problems prove the proposed method can find better solution than original guided genetic algorithm.

Key words:
combination optimization problem, flow shop scheduling problem, guided genetic algorithm

1. Introduction

The combinational optimization problem, such as scheduling problem, transportation programming, delivery planning, is limited by many conditions and difficult to be resolved [1]. Furthermore, in the case of a large-scale combinational optimization problem, there are many cases that a good solution cannot be obtained even if a human being spends a few days on solving it [2]. However, in recent years it became feasible with the rapid progress of the computer to do much numerical computation in a short time [3]. But it is difficult to accelerate because many of combinational optimization problems are the problems that are NP-hard about demanding strict optimum solution.

Therefore, over the years, an approximate method and a heuristic method are taken to the combinational optimization problem and the research to which the solution that seems that considerably near to the optimal solution is done. However, a study of the Metaheuristics to achieve a highly accurate solution with more time is done recently because it is difficult at all for even these techniques to solve a complicated combinational optimization problem. As typical methods, there are genetic algorithms (GA), taboo search algorithm (TS), simulated annealing (SA). For many cases these algorithms cannot arrive at the global optimum value (we call it optimum value as follows), because if it falls into a local minimum value once (we call it local minimum as follows), it is not possible to escape from there. Therefore, planning an escape from local minimum is effective technique to discover the optimum value in the elucidation of the combinational optimization problem with Meta heuristics.

In this paper, a method that applies genetic algorithm to solve a flow shop scheduling problem is suggested. It is specialized to solve a difficult problem in the combinational optimization problem. This is a technique for making a better solution enabled the search by adjusting the Lagrangian multiplier of each function that composes the objective function. Because the accuracy of the unit is low in the genetic algorithm, the improvement is added and has been used as the combinational optimization problem with an approximation method and a heuristic method as hybrid GA till now. Among these methods, Guided Genetic Algorithm (GGA) is special, it adds penalty to the one to enlarge (reduce) the value of the energy function, to change the objective function, and to attempt the escape from a minimum value. On the other hand, suggestion method is similar technique, but a wide application is possible because there is no penalty needed, besides, when the limiting condition is not filled, it has the effect similar to a conventional Lagrangian relaxation method.

In this paper, the proposed method is actually applied to the flow shop problem and the simulation results by comparing with some of other techniques shows that the proposed method is effective.

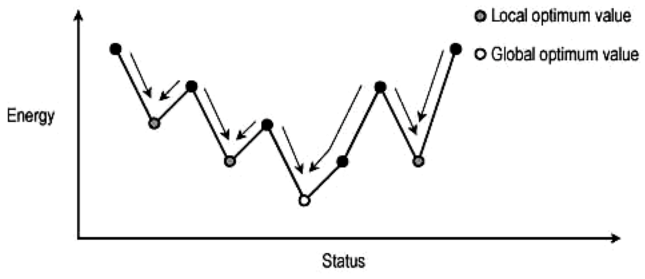

Fig. 1 Local optimum values and a global optimum value.

Manuscript received May 5, 2009
Manuscript revised May 20, 2009

2. Flow Shop Problem

The flow shop problem is solved in this paper. The conditions are described as follows.

2.1 Outline of Flow Shop Problem

1. About product X , it works with machine i in the production process only at necessary processing time $P(X,i)$. When the work on the last machine ends in the processing order, product X is completed.
2. When two or more products $X_1, X_2, ..., X_n$ are processed, each machine cannot process more than one work at the same time.
3. Each work cannot be processed at the same time with two machines or more.
4. The same product cannot be produced parallel. For example, when the number of production targets of products X is two, the second production cannot be started until the first production is complete.

2.2 Limiting Condition and Objective Function

There is a requirement should be met at least in the scheduling problem (limiting condition). In this paper, there is no repetitive use of the machine, and the processing order is defended. In many cases, there are two or more feasible solutions in a certain scheduling problem. It is caused by completion time of the product, goodness of the efficiency of the inventory control, and so on. So these situations should be considered. The objective function is used to judge the superiority or inferiority of the feasible solutions. In this paper, the following objective functions are enumerated as a definition of the best schedule, when solving a flow shop problem.

1. The earlier production is complete, the better.
2. The shorter the standby time of the production is, the better.
3. The shorter the standby time of the machine is, the better.

3. Genetic Algorithm

3.1 A Summary of the Genetic Algorithm

Genetic algorithms are implemented as a computer simulation in which a population of abstract representations (called chromosomes or the genotype of the genome) of candidate solutions (called individuals, creatures, or phenotypes) to an optimization problem evolves toward better solutions [1]. Traditionally, solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible. The evolution usually starts from a population of randomly generated individuals and happens in generations [4] [5]. In each generation, the fitness of every individual in the population is evaluated, multiple individuals are stochastically selected from the current population (based on their fitness), and modified (recombined and possibly randomly mutated) to form a new population. The new population is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population. If the algorithm has terminated due to a maximum number of generations, a satisfactory solution may or may not have been reached.

3.1.1 Initialization

When the initial individual group is generated, it is necessary to consider the variety of the chromosome [6]. In this paper, when an initial population is generated, an internal clock of the computer is used as a seed of random numbers.

3.1.2 Selection

In this paper, an original method which matches the elite strategy to the tournament method is used. The elite strategy copies individual with high evaluation value to a new population. Therefore, it is easy to fall into a minimum value though the searching ability is excellent. As the candidate of the selection, the roulette method decides the selected probability according to the evaluation value, and all of the individuals are objects [7]. Thus, there is a possibility of selecting the worst individual by this technique. Then, the original method mentioned above is to narrow the range of individual selection by elite strategy and to select individual depending on an evaluation value. On the other hand, the tournament method selects n individuals from population randomly and chooses an individual with the highest evaluation value as a parent. The tournament size (number of individuals chosen randomly) is set to be 2 in this paper. The 50% of the high-ranking is set according to the combination of roulette method and elite strategy in the individual group. In this paper, two parents' hamming distance is calculated, and the combinations of parents are selected only when the distance is larger than 1, in order to maintain the variety of the individuals.

3.1.3 Crossover

In this paper, uniform crossover was used. These crossover operators' searching abilities are assumed to be efficient in the order of one-point crossover < two-point crossover < Uniform crossover [8]. One-point crossover is that a single crossover point on both parents' organism strings is selected. All data beyond that point in either organism string is swapped between the two parent organisms. The resulting organisms are the children. The feature of one-point crossover is that it has the possibility of generating the child far apart from parents. Two-point crossover calls for two points to be selected on the parent organism strings. Everything between the two points is swapped between the parent organisms, and renders two child organisms. In the uniform crossover scheme (UX), the two parents are combined to produce two new offspring. And its individual bits in the string are compared between two parents. The bits are swapped with a fixed probability.

3.1.4 Mutation

In genetic algorithms, mutation is a genetic operator used to maintain genetic diversity from one generation of a population of chromosomes to the next. It is analogous to biological mutation. A common method of implementing the mutation operator involves generating a random variable for each bit in a sequence. This random variable shows whether or not a particular bit will be modified. The purpose of mutation in GAs is to allow the algorithm to avoid local minima by preventing the population of chromosomes from becoming too similar to each other, thus slowing or even stopping evolution. In this paper, the same method is used. Furthermore, it is made to select the individuals of the high-ranking 1% of individual groups successfully. And the processing replaced only when the evaluation value changes better than that before mutation. The state of the mutation is shown in Fig. 2.

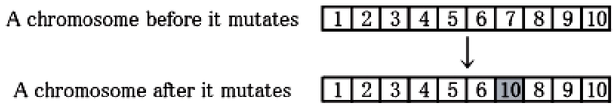

Fig. 2 Mutation.

3.1.5 Evaluation

Evolution are a sub-class of nature-inspired direct search (and optimization) methods belonging to the class of Evolutionary Algorithms (EAs) which use mutation, recombination, and selection applied to a population of individuals containing candidate solutions in order to evolve iteratively better and better solutions. In this paper, energy functions composed of objective functions and limited function are explained in detail in Chapter 4.

3.1.6 GA parameter

There is a variable called GA parameter when GA is designed. The definition of the GA parameter is important, and the quality of algorithm depends on these values greatly. There is no best conclusion yet although a lot of researches have been done up to now. The main parameter is population size N , the probability of the selection is P_{cross} , and the probability of the mutation is P_{mut} . P_{mut} is usually set between 0.001-0.01, but when this value is too small, it may fall into minimum value, and on the other hand, it suffers a long time when this value is too large. In this paper, the gene is set to be the maximum value. And the values of GA parameters are indicated in Chapter 5.

3.2 Application of Genetic Algorithm for Scheduling Problem

The population is called a genotype, and the character of the individual decided by the gene is called phenotype. In this paper, a phenotype is a Gantt chart. We define the encoding and decoding as follows: the encoding is to map phenotype to genotype, and the decoding is to map genotype to phenotype. The encoding is performed using the real numbers according to the viewpoint of manageability. If an appropriate encoding is not done, it is probable that an impracticability solution (lethal gene) is generated or the individual far apart from parents might be generated even if it is executable[13]. Fig. .3 shows the decoding from a chromosome to a Gantt chart. The genetic information expresses the stand-by time of the product with an actual number. The generation of the lethal gene is decreased as much as possible by encoding using the method described above. For example, there is a problem with 3 products and 3 machines. One machine's working hours of each product is $X_1 = 1, X_2 = 2, X_3 = 3,$ respectively, and the working line of $A \to B \to C$ is repeated twice. The structure of the chromosome is shown in Fig. .3. The chromosome gene number 1 ~ 3 shows the standby time until the work beginning of $X_1 \sim X_3$. The chromosome gene number 4 ~ 18 shows the standby time of the product. 4 ~ 8, 9 ~ 13, 14 ~ 18 show the product information of X_1, X_2, X_3 ,respectively.

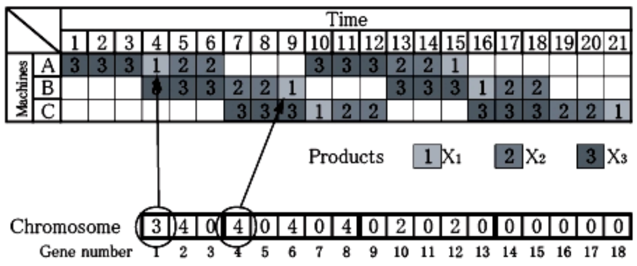

Fig. 3 Decoding of a chromosome.

4. Proposed Local Minimum Escaping Method

This section explains the minimum value escape method by the proposed improved Lagrangian relaxation method.

4.1 Improved Lagrngian Relaxation Method

The Lagrangian relaxation method is also called the penalty method, and it is a method of transferring the parameter to the limiting condition to break it.
The energy function E using the Lagrangian relaxation method is shown in equation (1).

$$E = \sum_{i=1}^{m} \lambda(i) \cdot F_c(i) + \sum_{j=1}^{n} F_o(j) \tag{1}$$

λ is the Lagrangian multiplier. F_c is the element of each parameter in constraint function, and the number of the element is m . F_o is the element of each parameter in objective function, and the number of element is n .
The parameter used in Lagrangian relaxation method is called a Lagrangian multiplier and the performance of the Lagrangian relaxation method depends on how to choose the Lagrangian multiplier. A past Lagrangian relaxation method is used to meet the limiting condition. It may not be improved after obtaining the solution that fulfills the limiting condition, when the Lagrangian relaxation method is applied to GA etc. In this paper, an improved Lagrangian relaxation method is proposed, and the Lagrangian multiplier is multiplied to element of each function. The optimum solution after the limiting condition is fulfilled can be efficiently searched. The improved energy function is shown in the following equation.

$$E = \sum_{i=1}^{m} \lambda_c(i) \cdot F_c(i) + \sum_{j=1}^{n} \lambda_o(j) \cdot F_o(j) \tag{2}$$

When the constraint function fulfilling the limiting condition is 0, the energy function can be corrected by assigning penalty to the parameter of the objective function. In addition, application is possible in other Metaheuristics not only GA but also hybrid GA because the proposed method is applicable for a combinational optimization problem with a constraint function and an objective function.

4.2 Escaping From Minimum Value by the Subgradient Method

Generally, it is difficult to obtain the best Lagrangian multiplier, and subgradient method is often used. The subgradient method is a technique for raising energy by adjusting the parameter built into the energy function to escape from a minimum value finally. The parameter is decided by evaluation function of the solution. In this paper, the gradient ascent, which differentiates the energy function by the parameter, and corrects the parameter in the direction of the most much zoom, is used.

Vector $\vec{\Lambda}$ is defined as an energy function. Therefore, if the study frequency is symbolized by s (discrete value) in parametrical space $\vec{\Lambda}$, $\vec{\Lambda}$ is updated using equation (3).

$$\vec{\Lambda}_{s+1} = \vec{\Lambda}_s + \Delta \vec{\Lambda}_s \tag{3}$$

Also, the $\Delta \vec{\Lambda}_s$, which is the correction of $\vec{\Lambda}$, is defined in equation (4).

$$\Delta \vec{\Lambda}_s = \eta \nabla e(\vec{\Lambda}_s) \tag{4}$$

Where η is the positive constant and ∇e is a gradient about $\vec{\Lambda}$ of energy function. Fig. 4 expressed a state of the escaping from minimum value by using subgradient method. Though A is a minimum value, a solution which is better than A has been discovered by changing the energy function. Not only the minimum value but also the energy of the surrounding solution synchronizes when the energy function is changed.
Therefore, other solutions may usually become the newer minimum value. It is always attempted to escape by using the subgradient method when falling into another minimum value.

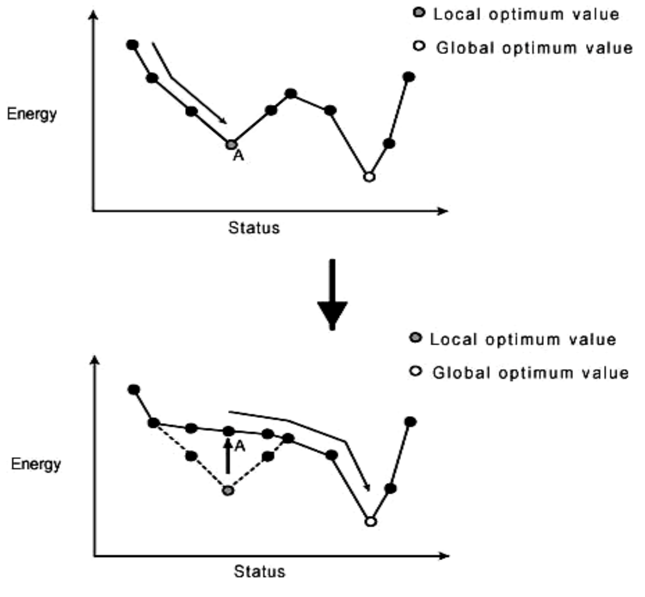

Fig. 4 The situation of escaping from the local optimum value.

5. Application of Proposed Local Minimum

5.2 Formulation of scheduling problem

In this paper, the flow shop problem which is mentioned in section 2 is resolved. The flow shop problem is formulated as follows.
Objective function:

$$\sum_{i=1}^{X_n} dE_2(i) + \sum_{i=1}^{M_n} dE_3(i) + \sum_{i=1}^{X_n} dE_4(i) \qquad (5)$$

Constraint function:

$$\sum_{i=1}^{X_n} \sum_{j=1}^{J_n(i)} dE_1(i,j) = 0 \qquad (6)$$

Where

$dE_1(i,j)$: The repetition time with a certain machine.
$dE_2(i)$: The completion time of product i .
$dE_3(i)$: The sum of stand-by time of machine i .
$dE_4(i)$: The sum of stand-by time of product i .
X_n : The number of products.
$J_n(i)$: The number of the work processes of product i .
M_n : The number of machines.

5.2 Energy function

Energy function is defined as follows.

$$E_1 = E_1 + E_2 + E_3 + E_4 \qquad (7)$$

E_1 : The repetition time of work.
E_2 : The sum of production completion time.
E_3 : The sum of the stand-by time of machines.
E_4 : The sum of the stand-by time of products.

5.3 Application of Proposed Local Minimum Escaping Method

The value of objective function E_1 is very different from that of constraint function E_2 , E_3 , E_4 . As this time, the Lagrangian multiplier is corrected, and the proposed local minimum escaping method works at the direction where the value of objective function is large to make it decreased. In addition, escaping from a minimum value is performed with the processing of constraint function and objective function respectively. First of all, when the constraint function is not fulfilled for E_1 , the Lagrangian multiplier is corrected using equation (8) by priority. λ is a Lagrangian multiplier.

$$\lambda_1(i,j) = \lambda_1(i,j) + \Delta\lambda_1(i,j) \qquad (8)$$

$$\Delta\lambda_1(i,j) = \eta_1 \cdot \frac{\partial E}{\partial \lambda_1(i,j)} = \eta_1 \cdot dE_1(i,j) \qquad (9)$$

When falling into a minimum value with an executable solution, the Lagrangian multiplier to E_2 , E_3 and E_4 is corrected using equation (10) ~ (15).

$$\lambda_2(i) = \lambda_2(i) + \Delta\lambda_2(i) \qquad (10)$$
$$\lambda_3(i) = \lambda_3(i) + \Delta\lambda_3(i) \qquad (11)$$
$$\lambda_4(i) = \lambda_4(i) + \Delta\lambda_4(i) \qquad (12)$$
$$\Delta\lambda_2(i) = \eta_2 \cdot \frac{\partial E}{\partial \lambda_2(i)} = \eta_2 \cdot dE_2(i) \qquad (13)$$
$$\Delta\lambda_3(i) = \eta_3 \cdot \frac{\partial E}{\partial \lambda_3(i)} = \eta_3 \cdot dE_3(i) \qquad (14)$$
$$\Delta\lambda_4(i) = \eta_4 \cdot \frac{\partial E}{\partial \lambda_4(i)} = \eta_4 \cdot dE_4(i) \qquad (15)$$

After the Lagrangian multiplier is corrected, it is necessary to escape from a minimum value. And it keeps correcting the Lagrangian multiplier until the superiority or inferiority relation is reversed. Moreover, after an executable solution becomes an impracticable solution, equation (8) is applied again to correct the Lagrangian multiplier. The flowchart is shown in Fig. .5.

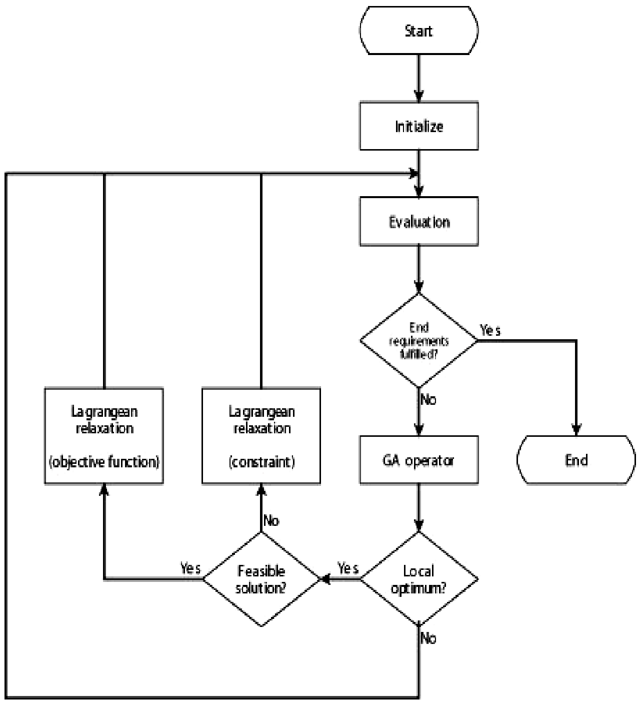
Start
Initialize
Evaluation
End requirements fulfilled?
Yes
No
End
GA operator
Local optimum?
Yes
No
Feasible solution?
Yes
No
Lagrangean relaxation (objective function)
Lagrangean relaxation (constraint)

Fig. 5 The Flowchart of the proposed method.

6. Simulation

6.1 A Summary of the Simulation

In this section, the proposed method is applied to flow shop problem. 4 products problem ($X_1 \sim X_4$) to 10 products problem ($X_1 \sim X_{10}$) of the scheduling problem are explained to prove the effectiveness of the proposed method. The following two kinds of production process models are set according to the difficulty of the scheduling problem.

Model 1: The problem is set with a low difficulty of the limiting condition. Machine $A\times 2$, $B\times 2$, $C\times 2$ and $Z\times 2$ are prepared in this model, and it is possible for all machines to perform parallel work.

Model 2: The problem is set with a high difficulty of the limiting condition. Machine $A\times 2$, $C\times 2$ and B , Z are prepared in this model, and it is possible for machine A and C to perform parallel work.
Table.1 shows the production process and the processing time of each product. Each product is performed consecutive twice as shown in Table. 1.

Table. 1 Processing order and time of products.

	Products									
Processing Order	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}
Z ↓	1	1	1	1	1	1	1	1	1	1
A ↓	2	3	4	5	6	2	3	4	5	6
Z ↓	1	1	1	1	1	1	1	1	1	1
B ↓	2	3	4	5	6	2	3	4	5	6
Z ↓	1	1	1	1	1	1	1	1	1	1
C	2	3	4	5	6	2	3	4	5	6

The following two techniques are simulated for the comparison with the proposed method.

Method 1: The scheduling problem is solved by using only GA. The state that falls into a minimum value is not corrected by proposal method.

Method 2: For the GA generation, the mutation is generated in the individual of the high rank 1% that always repeats ten times. Only when the individual with small value of the energy function is generated before mutating, the substitution is performed. In addition, the energy function is not corrected in the state of falling into a minimum value.

The GA parameter used for simulation is shown in Table. 2, and the Lagrangian multiplier used for proposed method is shown in Table. 3.

Table. 2 Setup of parameters for GA.

Parameters	Values
Population size N	300
Crossover rate P_{cross}	0.5
Mutation rate P_{mut}	0.1
End requirement	SGA 100000steps
Maximum value of genes	100

Table. 3 Setting of parameters for proposed method.

Parameters	Initial values	Values of η
Overlapping jobs	λ_1=1.0	η_1=1.1
Completion time of products	λ_2=1.0	η_2=1.0
Waiting time of machines	λ_3=0.10	η_3=0.11
Waiting time of products	λ_4=0.10	η_4=0.13

The trial frequency is set to be100 times, and the state of chromosomes is initialized randomly.

6.2 Simulation Result

6.2.1 Simulation result of model 1

The success rates of model 1 are shown in Table. 4. Through the comparison of success rates, it is shown that the proposed method is effective on searching the solution which meets the limiting condition. The shortest completion time and occurrence frequency of model 1 are shown in Table. 5. It is shown that the proposed method is effective for the objective function by comparing these values. Moreover, the average completion time is shown in Table.6. The completion time has been considerably shortened overall regardless of the number of products.

Table. 4 Success rate of model 1(%).

Number of products	Method 1	Method 2	**Proposed method**
4	100.0	100.0	**100.0**
5	100.0	100.0	**100.0**
6	100.0	100.0	**100.0**
7	100.0	100.0	**100.0**
8	100.0	100.0	**100.0**
9	100.0	100.0	**100.0**
10	100.0	100.0	**100.0**

Table. 5 Completion time of model 1 (Minimum).

	Method 1		Method 2		**Proposed method**	
Number of products	Time	Frequency	Time	Frequency	**Time**	**Frequency**
4	36	24	36	31	**36**	**83**
5	43	4	43	3	**42**	**20**
6	44	3	45	3	**43**	**2**
7	47	3	46	1	**44**	**1**
8	50	2	48	1	**48**	**3**
9	56	1	57	1	**55**	**2**
10	61	1	68	1	**61**	**1**

Table. 6 Completion time of model 1 (Average).

Number of products	Method 1	Method 2	**Proposed method**
4	38	38	**36**
5	48	47	**43**
6	57	53	**46**
7	80	64	**47**
8	115	112	**52**
9	186	200	**61**
10	265	285	**69**

6.2.2 Simulation result of model 2

Similarly, the success rates of model 2 are shown in Table. 7, and the shortest completion time and occurrence frequency are shown in Table. 8. The average completion time is shown in Table. 9. The success rates have been improved greatly in all the results, and the effect of the scheduling problem will be better if the number of products increases. The average value and the minimum value of the completion time decrease greatly though the occurrence frequency is low.

Table. 7 Success rate of model 2 (%).

Number of products	Method 1	Method 2	**Proposed method**
4	100.0	99.0	**100.0**
5	99.0	82.0	**100.0**
6	82.0	68.0	**100.0**
7	60.0	51.0	**100.0**
8	41.0	16.0	**100.0**
9	13.0	8.0	**100.0**
10	2.0	2.0	**100.0**

Table. 8 Completion time of model 2 (Minimum).

	Method 1		Method 2		**Proposed method**	
Number of products	Time	Frequency	Time	Frequency	**Time**	**Frequency**
4	43	5	41	1	**41**	**5**
5	55	1	80	1	**52**	**1**
6	171	1	170	1	**57**	**1**
7	253	1	194	1	**66**	**1**
8	309	1	373	1	**80**	**1**
9	502	1	418	1	**92**	**1**
10	416	1	492	1	**108**	**1**

Table. 9 Completion time of model 2 (Average).

Number of products	Method 1	Method 2	**Proposed method**
4	133	103	**45**
5	293	305	**61**
6	418	417	**70**
7	503	472	**81**
8	553	540	**94**
9	594	543	**109**
10	491	589	**127**

The scheduling result in 4 products problem of model 2 is shown in Fig. 6. The simulations using method 1 and proposed method respectively begin from a same initial state. The minimum value of energy function and the objective function are shown in Fig. 7, Fig. 8 and Fig. 9.

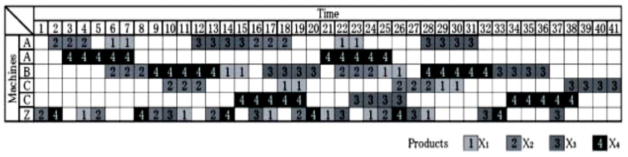

Fig. 6 A scheduling result of 4 products problem.

According to Fig. 8 and Fig. 9, the proposed method can search for more executable solutions. When the limiting condition is not fulfilled, the correction of the energy function will be performed. And after becoming an impracticable solution, an executable solution can be found again. Moreover, the energy function is corrected when falling into a minimum value (See Fig. 7).

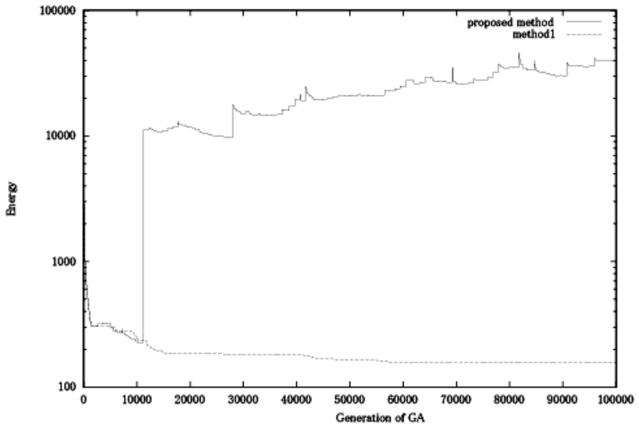

Fig. 7 Values of the energy function (4 products problem).

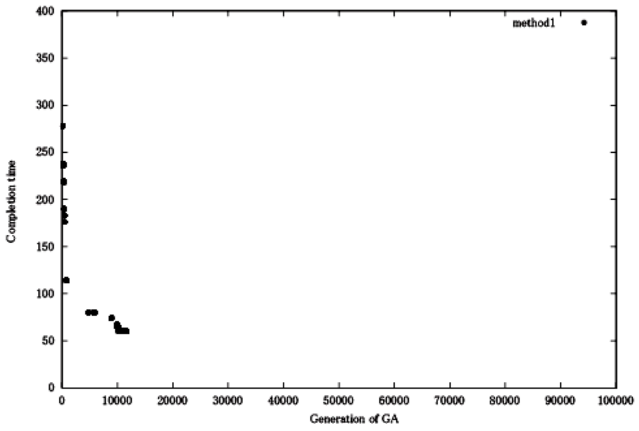

Fig. 8 Values of the objective function (Method 1).

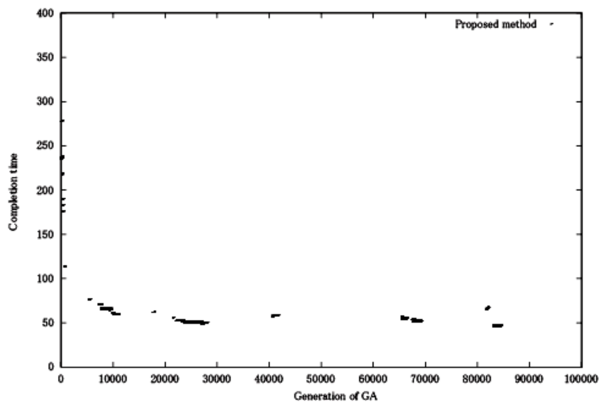

Fig. 9 Values of the objective function (Proposed method).

7. Conclusions

In this paper, as a technique of searching for the solution in the combination optimization problem efficiently, an improved genetic algorithm using Lagrangian relaxation method is proposed, and applied to the flow shop problem. From the result of the simulation, an executable solution to severe problems with the constraint condition can be obtained. Moreover, more optimum solutions can be obtained than using GA. The results give evidence of its effectiveness in light of the good quality of the solutions. The proposed method can also be extended to other hard combination optimization problems that composed of the constraint function and two or more objective functions.

References

[1] Mutsunori Yagiura and Toshihide Ibaraki, "On Metaheuristic Algorithms for Combinatorial Optimization Problems ", The Transactions of the Institute of Electronics, Information and Communication Engineers, Vol. J83-D-I, No.1, pp. 3-25, Jan. 2000.

[2] Uno Takeaki, "Uno Takeaki's Home Page", http://research.nii.ac.jp/~uno/index.html, 2005/01.

[3] Mitsuru Kuroda and Kenji Muramatsu, "Production scheduling", Asakura Bookstore, 2002/02.

[4] Mikio Kubo and Tomomi Matsui, "Combination optimization 'Short collection' ", Asakura Bookstore, 1999.

[5] Hitoshi Iba, "Base of genetic algorithm", Ohm Company, 1994.

[6] Hiroaki Kitano, "Genetic algorithm", Industrial Bookstore, 1993.

[7] T.L.Lau, and E.P.K.Tsang, "Solving the radio link frequency assignment problem with the guided genetic algorithm", University of Essex, 1998.

[8] Masao Mori and Tomomi Matsui, "Operations Research", Asakura Bookstore, 1999.

Xiaofei Wang received the B.S. degree from University of Toyama, Toyama, Japan in 2005 and M.S. degree from University of Toyama, Toyama, Japan in 2007. His main research interests are artificial neural networks and optimizations.

Wei Wang received the B.S.degree from Dalian Maritime University, Liaoning, China in 2003 and M.S. degree from University of Toyama, Toyama, Japan in 2007. His main research interests are artificial neural networks, artificial immune system and pattern recognition.

Hiroki Tamura received the B.E and M.E degree from Miyazaki University in 1998 and 2000, respectively. From 2000 to 2001, he was an Engineer in Asahi Kasei Corporation, Japan. In 2001, he joined University of Toyama, Toyama, Japan, where he was currently a Technical Official in Department of Intellectual Information Systems. In 2006, he joined Miyazaki University, Miyazaki, Japan, where he is currently an Assistant Professor in the Department of Electrical & Electronic Engineering. His main research interests are neural networks and optimization problems.

Zheng Tang received the B.S. degree from Zhejiang University, Zhejiang, China in 1982 and an M.S. degree and a D.E. degree from Tshinghua University, Beijing, China in 1984 and 1988, respectively. From 1988 to 1989, he was an Instructor in the Institute of Microelectronics at Tshinhua University. From 1990 to 1999, he was an associate professor in the Department of Electrical and Electronic Engineering, Miyazaki University, Miyazaki, Japan. In 2000, he joined Toyama University, Toyama, Japan, where he is currently a professor in the Department of Intellectual Information Systems. His current research interests include intellectual information technology, neural networks, and optimizations.