Scheduling Multi-Processor Tasks with Resource and Timing Constraints Using Genetic Algorithm

2002 International Computer Symposium (ICS2002)

Workshop on Artificial Intelligence

Authors: **Shu-Chen Cheng** (contact author), PhD student, Department of Engineering Science, National Cheng Kung University, Tainan, Taiwan, R.O.C.,
TEL: (886)6-2757575 ext: 63342-30, FAX: (886)6-2766549
E-mail address: n9889101@sparc1.cc.ncku.edu.tw

Shih-Tang Lo, PhD student, Department of Engineering Science, National Cheng Kung University, Tainan, Taiwan, R.O.C.,
TEL: (886)6-2757575 ext: 63342-30, FAX: (886)6-2766549
E-mail address: edwardlo@mail.ksut.edu.tw

Yueh-Min Huang, Professor, Department of Engineering Science, National Cheng Kung University, Tainan, Taiwan, R.O.C.,
TEL: (886)6-2757575 ext: 63336, FAX: (886)6-2766549
E-mail address: raymond@mail.ncku.edu.tw

Abstract

The job-shop scheduling problems have been categorized as NP-complete problems. The exponential growth of the time required to obtain an optimal solution makes the exhaustive search for global optimal schedules very difficult or even impossible. Recently, stochastic search techniques such as genetic algorithms have shown the feasibility to solve the job-shop scheduling problems. However, a pure GA-based approach tends to generate illegal schedules due to the crossover and the mutation operators, it is often the case that the gene expression or the genetic operators need to be specially designed to fit the problem domain or some other schemes may be combined to solve the scheduling problems.

This paper presents a GA-based approach with a feasible energy function to generate good-quality schedules. In our work, we design an easy-understood genotype to generate legal schedules without modifying the genetic algorithm or genetic operators and the proposed approach converges rapidly.

Keywords: scheduling, genetic algorithm, optimization.

Scheduling Multi-Processor Tasks with Resource and Timing Constraints Using Genetic Algorithm

Shu-Chen Cheng[1] Shih-Tang Lo[2] Yueh-Min Huang[3]

Department of Engineering Science

National Cheng Kung University

E-mail: [1] n9889101@sparc1.cc.ncku.edu.tw

[2]edwardlo@mail.ksut.edu.tw

[3]raymond@mail.ncku.edu.tw

Abstract

The job-shop scheduling problems have been categorized as NP-complete problems. The exponential growth of the time required to obtain an optimal solution makes the exhaustive search for global optimal schedules very difficult or even impossible. Recently, stochastic search techniques such as genetic algorithms have shown the feasibility to solve the job-shop scheduling problems. However, a pure GA-based approach tends to generate illegal schedules due to the crossover and the mutation operators, it is often the case that the gene expression or the genetic operators need to be specially designed to fit the problem domain or some other schemes may be combined to solve the scheduling problems.

This paper presents a GA-based approach with a feasible energy function to generate good-quality schedules. In our work, we design an easy-understood genotype to generate legal schedules without modifying the genetic algorithm or genetic operators and the

proposed approach converges rapidly.

Keywords: scheduling, genetic algorithm, optimization.

1. Introduction

The job-shop scheduling problems have been categorized as NP-complete problems. The time required to obtain an optimal solution increases exponentially with the augmentation of the number of jobs to be processed, the number of operations for each job and the number of flexible machines that can perform the processes. The exponential growth makes the exhaustive search for global optimal schedules very difficult or even impossible. Therefore, adaptive search approaches have been implemented to generate good-quality schedules instead of global optimal schedules. In 1999 and 2001, Huang and Chen [1][2] proposed an energy function for the Hopfield neural network (HNN) to schedule multiprocessor job with resource and timing constraints. Then, in 2001, they integrated fuzzy c-means clustering strategies into a Hopfield neural network to solve scheduling problems [3].

Recently, stochastic search techniques such as genetic algorithms have shown the feasibility to solve the job-shop scheduling problems. In 1999, Correa, Ferreira and Rebreyend proposed a knowledge-augmented genetic approach to schedule multiprocessor tasks [4]. In 2000, Hajra and etc. presented a controlled genetic algorithm based on fuzzy logic and belief functions to solve job-shop scheduling problems [5]. In 2001, Yang developed a GA-based discrete dynamic programming approach for scheduling in flexible

manufacturing system (FMS) environments [6]. However, a pure GA-based approach tends to generate illegal schedules due to the crossover and the mutation operators, it is often the case that the gene expression or the genetic operators need to be specially designed to fit the problem domain or some other schemes may be combined to solve the scheduling problems.

This paper presents a GA-based approach with a feasible energy function to generate good-quality schedules. Unlike other schemes suffering from rapidly converging to the local optima, the genetic algorithms provide a better chance to obtain the global optima.

This paper is organized as follows. The energy function of the scheduling problem is defined in Section 2. Next, the genetic algorithms are reviewed and the energy function is translated into the evaluation function in Section 3. The mathematical proof of the convergence of the energy function is then illustrated in Section 4. After this, the simulation examples and experimental results are presented in Section 5. Finally, conclusions of this paper are discussed in Section 6.

2. Energy Function of the Scheduling Problem

Job-shop scheduling problems differ from case to case. The scheduling problem domain to be considered in this paper is defined as follows. Suppose there are N jobs, each of which can be segmented, and there are M machines that are capable of performing the operations of all jobs. The execution time required by each job is predetermined and can be estimated by calculating the machine cycles. It is also assumed that different segments of a job cannot be

assigned to different machines, inferring that the job migration between machines is prohibited. Furthermore, the deadline constraint for each job is imposed on the proposed system. Besides, a resource is not allowed to be shared by two jobs simultaneously. Based on the above assumptions, we attempt to generate legal schedules.

To solve this problem, the energy function of the problem regarding all constraints is derived. In 1999 and 2001, Huang and Chen [1][2] proposed an energy function for the Hopfield neural network (HNN). The energy function is modified and reduced in this paper. A state variable V_{ijk} is defined as representing whether or not the job i is executed on machine j at a certain time k. Moreover, the state V_{ijk} =1 denotes that the job i is run on machine j at the time k; otherwise, V_{ijk} =0.

Since a machine j can process only one job i at any certain time k, the energy term can be defined as

$$\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{T}\sum_{\substack{i1=1\\ i1\neq i}}^{N} V_{ijk}V_{i1jk} \tag{1}$$

where V_{ijk} are as defined above; where i represents a job with a range from 1 to N, the total number of jobs to be scheduled; where j represents a dedicated machine identified from 1 to M, the total number of machines to be assigned; where k represents a specific time from 1 to T, the latest deadline of the job. The same notations are used hereinafter. The minimum value of this term is zero, which occurs when either V_{ijk} or V_{i1jk} equals zero.

As mentioned earlier, if a job is assigned on a dedicated machine, then all of its segments must be executed on the same machine. According this constraint, the energy term is defined as follows:

$$\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{T}\sum_{\substack{j1=1\\ j1\neq j}}^{M}\sum_{k1=1}^{T}V_{ijk}V_{ij1k1}. \tag{2}$$

Since a job i can be processed on either machine j or machine j_1 at any time, the minimum value of this term is zero, which occurs when either V_{ijk} or V_{ij1k1} equals zero.

As for the resource constraint, two jobs are not allowed to utilize the same resource instance simultaneously. Besides, the resource is non-preemptive so that the energy term can be defined as follows:

$$\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{T}\sum_{\substack{i1=1\\ i1\neq i}}^{N}\sum_{\substack{j1=1\\ j1\neq j}}^{M}\sum_{s}^{F}V_{ijk}R_{is}V_{i1j1k}R_{i1s} \tag{3}$$

where F denotes the quantity of available resource instances, R_{is} and R_{i1s} are the elements of the resource requested matrix for job i and i_1 respectively. The value R_{is} =1 indicates that job i requires resource s while R_{i1s} =1 implies that job i_1 requests resource s. When two distinct jobs are scheduled to be processed on different machines j and j_1 at the same time k (say V_{ijk} =1 and V_{i1j1k} =1), machines j and j_1 cannot share the same resource at the time k. Hence, either R_{is} or R_{i1s} is zero. This observation implies that the energy term becomes zero if the resource constraint is satisfied. Correspondingly, the total energy function with all constraints can be induced as Eq.(4):

$$E = \frac{C_1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{T}\sum_{\substack{i1=1\\ i1\neq i}}^{N} V_{ijk}V_{i1jk} + \frac{C_2}{2}\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{T}\sum_{\substack{j1=1\\ j1\neq j}}^{M}\sum_{k1=1}^{T} V_{ijk}V_{ij1k1}$$
$$+ \frac{C_3}{2}\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{T}\sum_{\substack{i1=1\\ i1\neq i}}^{N}\sum_{\substack{j1=1\\ j1\neq j}}^{M}\sum_{s=1}^{F} V_{ijk}R_{is}V_{i1j1k}R_{i1s}, \quad (4)$$

where C_1, C_2 and C_3 refer to weighting factors and are assumed to be positive constants in our study.

This work concentrates mainly on scheduling problems with constraint satisfaction. In the following section, genetic algorithms are introduced to solve the constraint satisfaction of the scheduling problems.

3. Genetic Algorithms

The exhaustive search for global optimal schedules is very difficult or even impossible. Therefore, adaptive search approaches have been implemented to generate good-quality schedules instead of global optimal schedules. To solve NP-hard optimization problems by using genetic algorithms have revealed their efficiency to generate good-quality schedules in relatively short computation time. Unlike other meta-heuristics such as simulated annealing, which processes a single point of the search space, genetic algorithms maintain a population of potential solutions [7]. Genetic algorithms perform a multi-directional search and encourage information exchange between different potential solutions so that the local optimum can be eliminated. The individuals in a population are called chromosomes, which consist of sets of genes. An initial population is randomly created. The population undergoes a simulated evolution by means of crossover and mutation to form a new population. At each iteration, the crossover point is randomly selected and couples of chromosomes swap the

corresponding segments to form new solutions. The mutation operator is applied to arbitrarily alter one gene of a selected chromosome. The iteration terminates when the value of the energy function reaches zero. However, a pure GA-based approach tends to generate illegal schedules due to the crossover and the mutation operators, it is often the case that the gene expression or the genetic operators need to be specially designed to fit the problem domain or some other schemes may be combined to solve the scheduling problems. In this paper, we introduce a way of representing a schedule and present a GA-based approach with a feasible energy function to generate good-quality schedules.

3.1 Representation of an Individual

Since there are N jobs, each of which can be segmented, and there are M machines that are capable of performing the operations of all jobs, a state variable V_{ijk} is defined as representing whether or not the job i is executed on machine j at a certain time k. Moreover, the state V_{ijk} =1 denotes that the job i is run on machine j at the time k; otherwise, V_{ijk} =0. Thus, P_{ijk} is defined as representing the probability of V_{ijk} =1. The chromosome, (P_{ijk} ; i=1,…,N; j=1,…,M; k=1,…,T), represents a potential scheduling solution. The dimension of the chromosome is equal to N*M*T.

3.2 Initial Population

An initial population is created by generating each gene, P_{ijk}, randomly.

3.3 Generating a Schedule

Due to the deadline constraint, no segment of a job is allowed to be assigned to a time later than the deadline of the job. Hence, P_{ijk} is set to zero if $k-d_i \leq 0$, where d_i denotes the deadline of the job i. Furthermore, the time spent by all segments of job i should be equal to PT_i, the processing time required by job i. The condition $(\sum_{j=1}^{M}\sum_{k=1}^{T} V_{ijk} = PT_i)$ must be satisfied. Therefore, for each job i, i=1,…,N, V_{ij1k} =1 if $P_{ij1k} \geq Th_{ij1}$, where Th_{ij1} is the PT_i-th highest P_{ij1k} on machine j_1, k=1,…,T;otherwise, V_{ijk} =0. Note that the job migration between machines is prohibited, so finding Th_{ij1} is restricted on machine j_1, where machine j_1 contains the highest P_{ijk} for j=1,…,M and k=1,…,T.

3.4 Evaluation Function

The evaluation function f(E) is calculated by

$$f(E) = \frac{1}{E + \varepsilon} \tag{5}$$

where E is the value of the energy function computed by Eq.(4) and ε is an extremely small value to prevent the denominator from becoming zero.

3.5 Genetic Operators

To make sure that the best member in the population survives, the elitist model is adopted. The best member of the previous generation is stored. If the best member of the current generation is worse than that of the previous generation, the latter one would replace the worst member of the current population. At each iteration, the crossover point is randomly

selected and two individuals are paired randomly to swap the corresponding segments to form new solutions with a prescribed probability P_C. The mutation operator is applied to arbitrarily alter one gene of a selected chromosome with a prescribed probability P_M.

4. Convergence of the Energy Function

The defined energy function dominates the convergence during the iteration. In this section, Eq.(4) is proven to be an appropriate Lyapunov function. Hence, the convergence is assured. Eq.(4) consists of two parts, one containing a state V_{lmn} using resource f and the other containing the rest of the states. Thus, $V_{lmn}=1$ indicates a situation in which machine m processes job l at time n using resource f. Contrarily, $V_{lmn}=0$ refers to that job l is neither executed on machine m nor utilizes resource f at time n. Herein, the equation is divided into two parts to observe the change of the energy with respect to the state V_{lmn} change. The energy before updating is shown below:

$$
\begin{aligned}
E = & \frac{C_1}{2}[V_{lmn}(\sum_{\substack{i1=1\\ i1\neq l}}^{N}\sum_{j=m}^{m}\sum_{k=n}^{n}V_{i1jk}+\sum_{\substack{i=1\\ i\neq l}}^{N}\sum_{j=m}^{m}\sum_{k=n}^{n}V_{ijk})+(\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{T}\sum_{\substack{i1=1\\ i1\neq i}}^{N}V_{ijk}V_{i1jk})_{\substack{V_{ijk}\neq V_{lmn}\\ and\\ V_{i1jk}\neq V_{lmn}}}] \\
& + \frac{C_2}{2}[V_{lmn}(\sum_{i=l}^{l}\sum_{\substack{j1=1\\ j1\neq m}}^{M}\sum_{k1=1}^{T}V_{ij1k1}+\sum_{i=l}^{l}\sum_{\substack{j=1\\ j\neq m}}^{M}\sum_{k=1}^{T}V_{ijk})+(\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{T}\sum_{\substack{j1=1\\ j1\neq j}}^{M}\sum_{k1=1}^{T}V_{ijk}V_{ij1k1})_{\substack{V_{ijk}\neq V_{lmn}\\ and\\ V_{ij1k1}\neq V_{lmn}}}] \\
& + \frac{C_3}{2}[V_{lmn}(\sum_{\substack{i1=1\\ i1\neq l}}^{N}\sum_{\substack{j1=1\\ j1\neq m}}^{M}\sum_{k=n}^{n}\sum_{s=f}^{f}V_{i1j1k}R_{i1s}R_{ls}+\sum_{\substack{i=1\\ i\neq l}}^{N}\sum_{\substack{j=1\\ j\neq m}}^{M}\sum_{k=n}^{n}\sum_{s=f}^{f}V_{ijk}R_{is}R_{ls}) \\
& + (\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{T}\sum_{\substack{i1=1\\ i1\neq i}}^{N}\sum_{\substack{j1=1\\ j1\neq j}}^{M}\sum_{\substack{s=1\\ s\neq f}}^{F}V_{ijk}R_{is}V_{i1j1k}R_{i1s})_{\substack{V_{ijk}\neq V_{lmn}\\ and\\ V_{i1j1k}\neq V_{lmn}}}]
\end{aligned}
\tag{6}
$$

where $V_{ijk} \neq V_{lmn}$ and $V_{i1jk} \neq V_{lmn}$ apply to the indexed parenthesized item only and indicates that the condition have different values. Similarly, the energy, E^{new}, after updating is derived as follows:

$$
\begin{aligned}
E^{new} &= \frac{C_1}{2}[V_{lmn}^{new}(\sum_{\substack{i1=1 \\ i1\neq l}}^{N}\sum_{j=m}^{m}\sum_{k=n}^{n} V_{i1jk} + \sum_{\substack{i=1 \\ i\neq l}}^{N}\sum_{j=m}^{m}\sum_{k=n}^{n} V_{ijk}) + (\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{T}\sum_{\substack{i1=1 \\ i1\neq i}}^{N} V_{ijk}V_{i1jk})_{\substack{V_{ijk}\neq V_{lmn}^{new} \\ and \\ V_{i1jk}\neq V_{lmn}^{new}}}] \\
&+ \frac{C_2}{2}[V_{lmn}^{new}(\sum_{i=l}^{l}\sum_{\substack{j1=1 \\ j1\neq m}}^{M}\sum_{k1=1}^{T} V_{ij1k1} + \sum_{i=l}^{l}\sum_{\substack{j=1 \\ j\neq m}}^{M}\sum_{k=1}^{T} V_{ijk}) + (\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{T}\sum_{\substack{j1=1 \\ j1\neq j}}^{M}\sum_{k1=1}^{T} V_{ijk}V_{ij1k1})_{\substack{V_{ijk}\neq V_{lmn}^{new} \\ and \\ V_{ij1k1}\neq V_{lmn}^{new}}}] \\
&+ \frac{C_3}{2}[V_{lmn}^{new}(\sum_{\substack{i1=1 \\ i1\neq l}}^{N}\sum_{\substack{j1=1 \\ j1\neq m}}^{M}\sum_{k=n}^{n}\sum_{s=f}^{f} V_{i1j1k}R_{i1s}R_{ls} + \sum_{\substack{i=1 \\ i\neq l}}^{N}\sum_{\substack{j=1 \\ j\neq m}}^{M}\sum_{k=n}^{n}\sum_{s=f}^{f} V_{ijk}R_{is}R_{ls}) \\
&+ (\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{T}\sum_{\substack{i1=1 \\ i1\neq i}}^{N}\sum_{\substack{j1=1 \\ j1\neq j}}^{M}\sum_{\substack{s=1 \\ s\neq f}}^{F} V_{ijk}R_{is}V_{i1j1k}R_{i1s})_{\substack{V_{ijk}\neq V_{lmn}^{new} \\ and \\ V_{i1j1k}\neq V_{lmn}^{new}}}]
\end{aligned}
\tag{7}
$$

Hence, according to Eq.(6) and Eq.(7), the changes of the energy can be calculated as follws:

$$
\begin{aligned}
\Delta E_{lmn} &= E^{new} - E \\
&= \frac{C_1}{2}(V_{lmn}^{new} - V_{lmn})(2\sum_{\substack{i=1 \\ i\neq l}}^{N}\sum_{j=m}^{m}\sum_{k=n}^{n} V_{ijk}) \\
&+ \frac{C_2}{2}(V_{lmn}^{new} - V_{lmn})(2\sum_{i=l}^{l}\sum_{\substack{j=1 \\ j\neq m}}^{M}\sum_{k=1}^{T} V_{ijk}) \\
&+ \frac{C_3}{2}(V_{lmn}^{new} - V_{lmn})(2\sum_{\substack{i=1 \\ i\neq l}}^{N}\sum_{\substack{j=1 \\ j\neq m}}^{M}\sum_{k=n}^{n}\sum_{s=f}^{f} V_{ijk}R_{is}R_{ls})
\end{aligned}
\tag{8}
$$

According to Eq.(8), the total energy difference is involved in the change of $(V_{lmn}^{new}-V_{lmn})$. Whenever the state changes from 0→1, 1→0, 0→0, or 1→1, the change of $(V_{lmn}^{new}-V_{lmn})$ has different effects on the energy difference. For convenience, the above energy difference ΔE_{lmn} is rewritten as Eq.(9)

$$\Delta E_{lmn} = (V_{lmn}^{new} - V_{lmn})[\frac{C_1}{2}(2\sum_{\substack{i=1 \\ i\neq l}}^{N}\sum_{j=m}^{m}\sum_{k=n}^{n}V_{ijk}) + \frac{C_2}{2}(2\sum_{i=l}^{l}\sum_{\substack{j=1 \\ j\neq m}}^{M}\sum_{k=1}^{T}V_{ijk}) + \frac{C_3}{2}(2\sum_{\substack{i=1 \\ i\neq l}}^{N}\sum_{\substack{j=1 \\ j\neq m}}^{M}\sum_{k=n}^{n}\sum_{s=f}^{f}V_{ijk}R_{is}R_{ls})]$$
$$= (V_{lmn}^{new} - V_{lmn})[\Delta E_{item1} + \Delta E_{item2} + \Delta E_{item3}] \quad (9)$$

According to Eq.(9), the change of energy concerns itself with state change of (V_{lmn}^{new}-V_{lmn}), where ΔE_{item1}, ΔE_{item2}, and ΔE_{item3} correspond to the associated items of C_1, C_2, and C_3, respectively. Closely examining Eq.(9) again obviously reveals that when $V_{lmn}^{new}=V_{lmn}$, i.e. when state changes from 0→0 or 1→1, the system is in a stable condition and the energy difference is zero ($\Delta E_{lmn}=0$). A state change from 0→1 (V_{lmn}^{new}-V_{lmn} =1) indicates that machine m is processing job *l* at time n. Hence, according to the state constraint definition, we can infer that ΔE_{item1} is zero.

$$(\sum_{\substack{i=1 \\ i\neq l}}^{N}\sum_{j=m}^{m}\sum_{k=n}^{n}V_{ijk} = 0) \quad (10)$$

As mentioned earlier, if a job is assigned on a dedicated machine, then all of its segments must be executed on the same machine. Thus, ΔE_{item2} is equal to zero.

$$(\sum_{i=l}^{l}\sum_{\substack{j=1 \\ j\neq m}}^{M}\sum_{k=1}^{T}V_{ijk} = 0) \quad (11)$$

Furthermore, ΔE_{item3} is also zero,

$$(\sum_{\substack{i=1 \\ i\neq l}}^{N}\sum_{\substack{j=1 \\ j\neq m}}^{M}\sum_{k=n}^{n}\sum_{s=f}^{f}V_{ijk}R_{is}R_{ls} = 0) \quad (12)$$

since R_{ls}=1 indicates that resource f is used by job *l*, subsequently forcing R_{is}=0. Consequently, the energy difference, ΔE_{lmn} =(1-0)(0+0+0), is equal to zero when the state V_{lmn}^{new} becomes one.

Finally, when the state changes from 1→0, (V_{lmn}^{new}-V_{lmn} =-1), ΔE_{item1} has a maximum value of C_1,

$$\sum_{\substack{i=1 \\ i\neq l}}^{N}\sum_{j=m}^{m}\sum_{k=n}^{n} V_{ijk} = 0 \text{ or } 1, \tag{13}$$

since a machine can process one job at most or does nothing at a certain time. ΔE_{item2} also has a maximum value of C_2*P_l,

$$\sum_{i=l}^{l}\sum_{\substack{j=1 \\ j\neq m}}^{M}\sum_{k=1}^{T} V_{ijk} = 0 \text{ or } P_l, \tag{14}$$

where P_l and C_2 are both positive and P_l is the total execution time of job *l*. In addition, a situation in which V_{lmn}^{new} =0 implies that resource f is not used by job *l*, which will force R_{ls} =0, and then ΔE_{item3} becomes zero,

$$\left(\sum_{\substack{i=1 \\ i\neq l}}^{N}\sum_{\substack{j=1 \\ j\neq m}}^{M}\sum_{k=n}^{n}\sum_{s=f}^{f} V_{ijk}R_{is}R_{ls} = 0\right). \tag{15}$$

Consequently, the energy difference, $\Delta E_{lmn} \leq (0-1)(C_1 + C_2 * P_l + 0)$, is less than zero when the state V_{lmn}^{new} becomes zero. Correspondingly, the proposed energy function is a Lyapunov function.

5. Simulation Examples and Experimental Results

Three sets of resource and timing constraints were applied for the simulations. The constants of the energy function, C_1, C_2, and C_3, were all set to 1 in this work. Each population of chromosomes of the genetic algorithm was initialized randomly. The population size was 100 and other parameters such as the probability of the crossover and the mutation were 0.4 and 0.06, respectively. The resource requested matrix and the timing constraints matrices for three cases are shown in Table 1 and Table 2. There are 2 machines available for Case 1 and 2 while there are 3 machines available for Case 3.

Table 1. Resource Requested Matrix

(a) Case 1

	R1	R2	R3
Job 1	1	0	0
Job 2	0	0	1
Job 3	0	1	0
Job 4	1	0	0

(b) Case 2

	R1	R2	R3	R4
Job 1	1	0	0	0
Job 2	0	1	0	0
Job 3	0	0	1	0
Job 4	0	0	0	1
Job 5	1	0	0	1

(c) Case 3

	R1	R2	R3	R4
Job 1	1	0	0	0
Job 2	0	1	0	0
Job 3	0	0	0	0
Job 4	1	0	0	0
Job 5	0	0	1	0
Job 6	0	1	0	0
Job 7	0	0	0	1
Job 8	0	0	1	0
Job 9	0	0	0	0
Job 10	0	0	1	0

Table 2. Timing Constraints Matrix

(a) Case 1

	Time Required	Time Limit
Job 1	4	6
Job 2	3	4
Job 3	3	6
Job 4	2	3

(b) Case 2

	Time Required	Time Limit
Job 1	2	3
Job 2	5	8
Job 3	3	4
Job 4	4	8
Job 5	2	5

(c) Case 3

	Time Required	Time Limit
Job 1	5	10
Job 2	3	5
Job 3	3	9
Job 4	2	5
Job 5	3	9
Job 6	2	6
Job 7	3	10
Job 8	2	5
Job 9	3	9
Job 10	4	10

Figure 1 displays the energy curve of the best member in the population for 3 cases during iterations. The simulated scheduling results are graphically represented by using the Gantt charts and are shown in Figure 2.

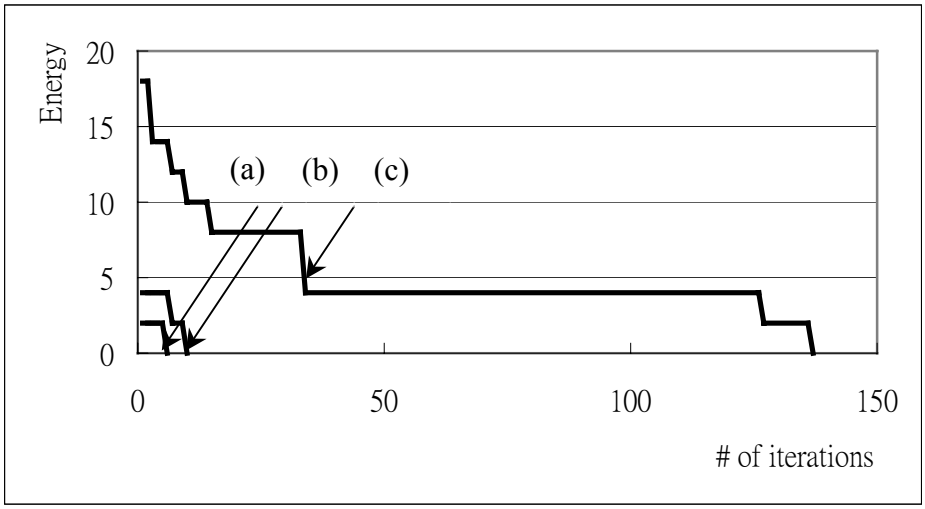

Fig. 1. The energy curve of the best member in the population.(a)Case 1 (b)Case 2 (c)Case 3.

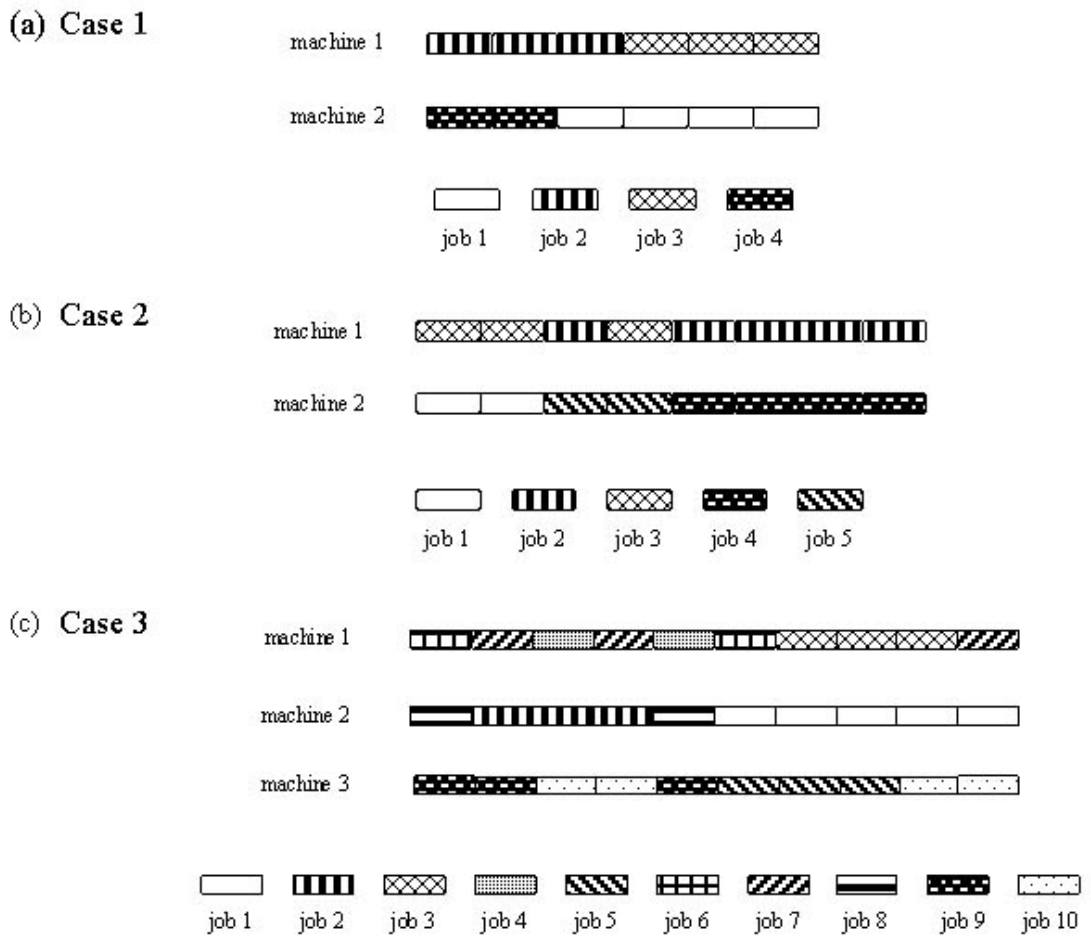

Fig. 2. The simulated scheduling results. (a)Case 1 (b)Case 2 (c)Case 3.

6. Conclusions

In this paper, we present a GA-based approach with a feasible energy function to generate good-quality schedules. Unlike other schemes suffering from rapidly converging to the local optima, the genetic algorithms provide a better chance to obtain the global optima. Besides, a pure GA-based approach tends to generate illegal schedules due to the crossover and the mutation operators, it is often the case that the gene expression or the genetic operators need to be specially designed to fit the problem domain or some other schemes may be combined to solve the scheduling problems.

In our work, we design an easy-understood genotype to generate legal schedules without modifying the genetic algorithm or genetic operators. Also, the energy function proposed work efficiently. The time required to obtain an optimal schedules using an exhaustive search increases exponentially with the augmentation of the number of jobs to be processed, the number of operations for each job and the number of flexible machines that can perform the processes. Therefore, the complexity of the search space is $O(2^N*2^M*2^T)$. However, in the three simulated cases, the proposed scheme converges rapidly. In the future, we attempt to investigate the applicability of our approach to larger-size practical examples.

Reference

[1] Yueh-Min Huang and Ruey-Maw Chen, "Scheduling Multiprocessor Job with Resource and Timing Constraints Using Neural Networks", *IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics, Vol.29, No.4*, pp. 490-502, 1999.

[2] Ruey-Maw Chen and Yueh-Min Huang, "Competitive Neural Network to Solve Scheduling Problems", *Neurocomputing*, pp. 177-196, 2001.

[3] Ruey-Maw Chen and Yueh-Min Huang, "Multiprocessor Task Assignment with Fuzzy Hopfield Neural Network Clustering Technique", *Neural Computing and Applications*, pp. 12-21, 2001.

[4] Ricardo C. Correa, Afonso Ferreira and Pascal Rebreyend, "Scheduling Multiprocessor Tasks with Genetic Algorithms", *IEEE Transactions on Parallel and Distributed Systems, Vol.10, No.8*, pp. 825-837, 1999.

[5] S. Hajri, N. Liouane, S. Hammadi and P. Borne, "A Controlled Genetic Algorithm by Fuzzy Logic and Belief Functions for Job-Shop Scheduling", *IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics, Vol.30, No.5*, pp. 812-818, 2000.

[6] Jian-Bo Yang, "GA-Based Discrete Dynamic Programming Approach for Scheduling in FMS Environments", *IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics, Vol.31, No.5*, pp. 824-835, 2001.

[7] Zbigniew Michalewicz, "Genetic Algorithms + Data structures = Evolution Programs", Third Edition, *Springer*, pp. 13-44, 1999.