Journal of Universal Computer Science, vol. 15, no. 4 (2009), 765-785
submitted: 30/6/08, accepted: 27/2/09, appeared: 28/2/09 © J.UCS

A Quantum-Inspired Immune Algorithm for Hybrid Flow Shop with Makespan Criterion

Qun Niu
(Shanghai Key Laboratory of Power Station Automation Technology
School of Mechatronic Engineering and Automation, Shanghai University, Shanghai, China
comelycc@hotmail.com)

Taijin Zhou
(Shanghai Key Laboratory of Power Station Automation Technology
School of Mechatronic Engineering and Automation, Shanghai University, Shanghai, China
zhoutaijin@hotmail.com)

Shiwei Ma
(Shanghai Key Laboratory of Power Station Automation Technology
School of Mechatronic Engineering and Automation, Shanghai University, Shanghai, China
swma@mail.shu.edu.cn)

Abstract: This paper presents a quantum-inspired immune algorithm (QIA) for Hybrid flow shop problems (HFSP) to minimize makespan. Since HFSP have been proved to be NP-hard in a strong sense when the objective is to minimize the makespan, an effective immune algorithm (IA) is used to solve the problems. IA is a kind of evolutional computation strategies, which is developed on the basis of a real immune mechanism in the human body, and has been employed to tackle complex scheduling problems and produce a reasonable manufacturing schedule. In order to achieve better results, the standard IA is combined with quantum algorithm (QA), which is based on Q-bit and uses quantum rotation gate to update. A real number representation is proposed to convert the Q-bit representation to job permutation for evaluating value of solutions. The proposed QIA can overcome the limitations of IA, quicken up convergence speed and improve the solution. Forty one benchmarks are examined to validate the efficiency of the proposed algorithm. The computational experiments show that the proposed QIA can also obtain both better and more robust results than IA and QA.

Keywords: Hybrid flow shop scheduling, Immune algorithm, Quantum algorithm, Quantum rotation gate
Categories: I.2.8, F.2.2

1 Introduction

Shop scheduling problems play a key role in industrial processes, manufacturing systems and other fields. Good scheduling makes a company have high profit and a competitive position in the market. Hybrid flow shop problems (HFSP) are quite common in a flexible flow shop environment, which can represent most of production systems. For example, manufacturing, oil food, paper, chemical and cosmetic industry can be modelled as HFSP.

HFSP have been proved to be nondeterministic polynomial-time hard (NP-Hard) [Gupta 88b] when the objective is to minimize makespan in case of two stages. HFSP consist of several production workshops or stages. Each stage has several parallel machines. Some stages may have only one facility, but at least one stage must have more than one facility. In this research, the scheduling objective is to minimize the makespan which is a maximum completion time.

Many approaches were used to solve HFSP, such as branch-and-bound (B&B), immune algorithm (IA), genetic algorithm (GA), simulated annealing (SA), tabu search (TS), and etc. In the first study on HFSP, Arthanari and Ramaswamy [Arthanari 71b] proposed a B&B to tackle this problem. The branch and bound which was used to solve the HFSP was improved by Portman et al. [Portmann, Vignier, Dardilhac and Dezalay 98b] through using genetic algorithm-based upper bounds and new lower bounds. Rubén Ruiz and Concepción Maroto [Ruiz and Maroto 06b] introduced genetic algorithm for hybrid flow shops. Allahverdi and Al-Anzi [Allahverdi and Al-Anzi 08b] developed SA as a heuristic algorithm to solve the two-stage assembly flow shop scheduling problem with a weighted sum of makespan and mean completion time criteria. A tabu search algorithm has been applied to HFSP with limited buffer capacities between stages [Wardono and Fathi 04b]. Kemal Alaykýran [Alaykýran, Engin and Döyen 07b] proposed an improved ant colony optimization to solve HFSP.

IA imitates the immune system which is against its invaders in a biological body [Mori, Tsukiyama and Fukuda 93b] [De Castro and Von Zuben 02b]. Recently, the interest in IA and its applications [Chen, Kim and Jong 97b] [Tsai, Ho, Liu and Chou 07b] [Tsai and Chou 06b] [Carrano, Guimaraes, Takahashi, Neto and Campelo 07b] have been increasing. The immune algorithm approach to hybrid flow shops scheduling with sequence-dependent setup times was proposed by M. Zandieh [Zandieh, Fatemi Ghomi and Moattar Husseini 06b]. Reza Tavakkoli-Moghaddam [Tavakkoli-Moghaddam, Rahimi-Vahed and Mirzaei 07b] developed IA to solve a multi-objective model for a no-wait flow shop scheduling problem. And an IA approach was adopted to solve the scheduling problems of a flexible flow shop for the manufacture of PCBs for communication equipment [Alisantoso, Khoo and Jiang 99b]. IA has an accelerating mechanism and a restraining mechanism. These mechanisms can restraint the over-dominance of good solutions and achieve better diversity during the search.

Quantum algorithm (QA) is based on probability [Alisantoso, Khoo and Jiang 99b] in random to search the best solution. Q-bit chromosomes can overcome premature and maintain solution diversity. Quantum rotation gate is used to update the individuals. In the last few years, many researchers studied biologically inspired systems and also gained much progress. Jang [Jang, Han and Kim 03a] proposed a new face verification algorithm using Quantum-inspired Evolutionary Algorithm (QEA). Quantum Genetic Algorithm (QGA) was employed to solve Binary Decision Diagram Ordering Problem [Layeb and Saidouni 07b]. The principle of quantum computing reduced population size and a reasonable number of iterations to find the best solution. Kuk-Hyun Han and Jong-Hwan Kim [Han and Kin 04b] researched issues on QEA such as a termination criterion, a Q-gate, and a two-phase scheme, for a class of numerical a combinatorial optimization problems. QA was also used to solve knapsack problems [Han and Kim 02b] and travelling salesman problems

[Narayanan and Moore 96a]. QA has been applied in many fields. however, very few studies have employed QA to solve the scheduling problems. Bin-Bin Li and Ling Wang have proposed a hybrid quantum-Inspired genetic algorithm for the multiobjective flow shop scheduling problem. [Li and Wang 07b]. Quantum-inspired evolutionary computing characterized by certain principles of quantum mechanisms for a classical computer also has been studied [Narayanan and Moore 96a] [Han and Kim 00a].

IA is very similar to GA. Therefore, IA is also easy to get into premature convergence and stagnation in the late of evolution. In this paper, a Quantum-Inspired Immune Algorithm (QIA) is proposed to solve HFSP. This method can overcome the disadvantages to some extent. We can apply the QIA for both exploration in permutation-based scheduling space and exploitation for good schedule solutions. And a real number representation is proposed according to the encoding of HFSP to convert the Q-bit representation to a job permutation. The job permutation is adopted to evaluate the objective value of the schedule solution.

The remainder of the paper is organized as follows. In section 2, we describe the hybrid flow shop problems. Section 3 introduces the proposed immune algorithm. The structure of quantum algorithm is given in Section 4. Section 5 presents experimental design. Simulation results which are compared by proposed Quantum-Inspired Immune Algorithm with those achieved by immune algorithm and quantum algorithm are shown in Section 6. Section 7 concludes the paper and introduces the future work.

2 Problem Description

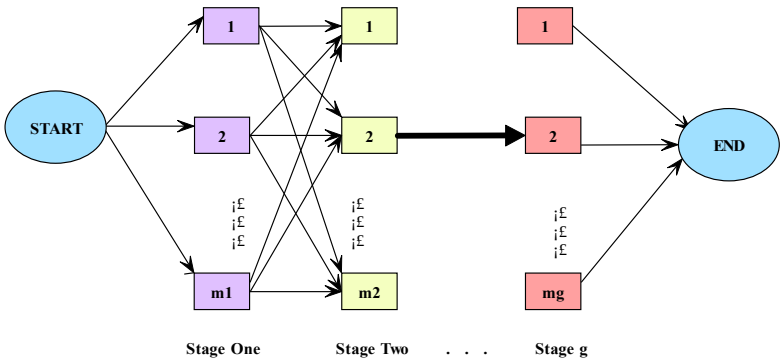

Figure 1: Schematic of a hybrid flow shop problems

HFSP are quite common and known as the flow shop with multiple machines. It is commonly defined as follows: M jobs and g stages. And each stage t ($t = 1, 2, 3, ..., g$) has m_t identical machines. These identical machines have the same effect. At least one stage t ($t = 1, 2, 3, ..., g$) has more than one machine. Each job is to be sequentially processed on stage 1, 2, 3… g. And job m ($m = 1, 2, 3, ..., M$) is processed on any one

machine at each stage. At any time, each machine also only processes one job. One machine must process another job after the former completed. The processing time $pt(i,t)$ ($i = 1,2,3,...,M$; $t = 1,2,3,...,g$) is given for each job at each stage. The objective is to find a permutation of M jobs at each stage and minimize the makespan.

As shown in Fig.1, HFSP have g stages and each stage has m_t ($t = 1,2,3,...,g$) identical machines. In this environment, each job is processed from the 'START' to the "END" in Fig.1. Each job may be processed on any one of the identical machines at each stage. Every job has the same processed stage route, that is, from start to end.

3 Immune Algorithm

The immune algorithm comes from the biological immune system which is a robust, strong, and adaptive recognition system that defends the body from bacteria or viruses (foreign pathogens). The system also includes cells that are able to kill pathogens and recognize antigens. It is able to categorize cells (or molecules) which are belonged to its own kind (self cell) or foreign origin (non-self cell) [Dasgupta 02b]. The immune defense mechanism is either non-specific (innate) which is gained through evolution from generation or specific (acquired) which is studied through its own encounter with antigens [Khoo and Situmdrang 03b].

These cells are named immune cells or antibodies. Most of immune cells are lymphocytes, like B-cells and T-cells. Receptor molecules which can recognize disease-causing pathogens or antigens are on immune cells' surfaces. Because each individual immune cell is capable of recognizing only one type of antigen, immune system has a large diversity of immune cells in the organism facing for many possible and different antigens.

Based on immune algorithm, the antibodies and the antigens in the immune system are regarded as the feasible solution and the objective function of the problem. The best antibodies are found, i.e., the best solution is also found. The objective function gets the best value.

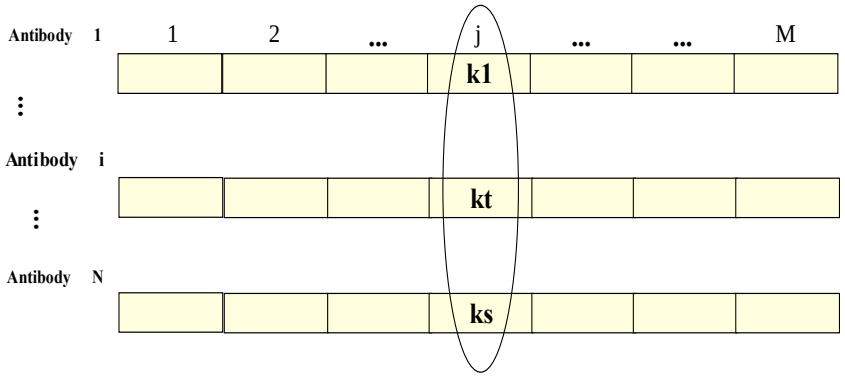

Figure 2: Antibody representation

The genetic encoding structure of the IA is similar to that of the GA. So antibody representation is shown in Fig.2. There are *N* antibodies; each of them consists of *M* genes. Each of genes includes *ks* alleles that come from i^{th} gene of each antibody.

Immune system has two kinds of affinities between antigens and antibodies or between antibodies and antibodies. Based on two affinities can select a low concentration and high fitness antibody.

The affinity can be defined by information entropy theory. Abramson [Abramson (63)] defined the information entropy $H(x)$. As Fig.2 shows, $H_j(N)$ is the information entropy of the j^{th} gene. And it is given by Eq.(1).

$$H_j(N) = \sum_{i=1}^{ks} -P_{ij} \log P_{ij} \tag{1}$$

Where P_{ij} is the probability that the i^{th} allele appears in the j^{th} gene.

If all alleles at the same position are the same, P_{ij} is equal to 1. Hence $H_j(N)$ is also equal to 0. The average information entropy $H(N)$ of all genes is defined as follows.

$$H(N) = \frac{1}{M} \sum_{j=1}^{M} H_j(N) \tag{2}$$

So, the affinity between the antibody *m* and *n* is defined:

$$ay_{m,n} = \frac{1}{1+H(2)} \tag{3}$$

If the antibody is represented by random discrete variable, we can give some definitions: If the two antibodies are the same, $H(2)$ is equal to 0. On the other hand, if the j^{th} gene is different between the antibody *m* and the antibody *n*, P_{ij} is equal to 0.5. The value of $ay_{m,n}$ is between 0 and 1.

Affinity ax_m is between antigen and antibody *m*:

$$ax_m = \frac{1}{1+opt_m} \tag{4}$$

Where opt_m : the objective value of the antibody *m*.

The concentration of antibody *m* is equal to

$$C_m = \frac{1}{N} \sum_{n=1}^{N} ac_{mn} \tag{5}$$

Where $ac_{mn} = \begin{cases} 0, otherwise \\ 1, ay_{mn} \geq Tac1 \end{cases}$ and $Tac1$, namely affinity threshold, is a constant which represents similarity degree of antibodies.

The rate of expectation propagation e_m is described as follows:

$$e_m = \frac{a_x m}{C_m} \tag{6}$$

4 Quantum Algorithm

In the early 1980s', Quantum Computer (QC), a relative concept to the classical computer was firstly proposed by Beniorff P. [Benioff 80b]. QC is a study area that contains concepts like quantum mechanical computers and quantum algorithms. And various specialized problems are solved. For example, there are famous quantum algorithms such as Shor's quantum factoring algorithm [Shor 94a] and Grover's database search algorithm [Grover 96a].

QA uses a new representation which is based on the concept of Q-bits and superposition of states. A Q-bit is the smallest unit of information stored in a two-state quantum computer. It may be in the "1" state, or in the "0" state, or in the any superposition between "0" and "1".

A Q-bit is represented as a linear combination of basic states ("0" state and "1" state): $|\psi>=\alpha|0>+\beta|1>$, Where $|\psi>,|0>,|1>$ is the quantum state, α and β are complex numbers that denote the probability amplitudes of the corresponding states. And α and β also satisfy the normalization condition $|\alpha|^2+|\beta|^2=1$, $|\alpha|^2$ is probability of state being 0 and $|\beta|^2$ gives the probability that the Q-bit will be found in the '1' state. A Q-bit chromosome with a string of *m*-qubit is defined as:

$$\left[\begin{array}{c|c|c|c} \alpha 1 & \alpha 2 & \ldots & \alpha m \\ \beta 1 & \beta 2 & \ldots & \beta m \end{array}\right]$$

For instance, following three-Q-bit:

$$\left[\begin{array}{c|c|c} \frac{\sqrt{3}}{2} & \frac{1}{3} & \frac{\sqrt{2}}{2} \\ \frac{1}{2} & \frac{2\sqrt{2}}{3} & -\frac{\sqrt{2}}{2} \end{array}\right]$$

The states can be described as follows:

$$\frac{1}{24}|000\rangle-\frac{1}{24}|001\rangle+\frac{1}{3}|010\rangle-\frac{1}{3}|011\rangle+\frac{1}{72}|100\rangle-\frac{1}{72}|101\rangle+\frac{1}{9}|110\rangle-\frac{1}{9}|111\rangle$$

The above three-Q-bit string includes eight states. And every state has a probability. Quantum algorithm is updated by a quantum rotation gate. This operator makes a current solution approach to the best solution gradually. The quantum rotation gate can be described as follows:

$$\begin{bmatrix}\alpha_i' \\ \beta_i'\end{bmatrix}=\begin{bmatrix}\cos(\Delta\theta_i) & -\sin(\Delta\theta_i) \\ \sin(\Delta\theta_i) & \cos(\Delta\theta_i)\end{bmatrix}\begin{bmatrix}\alpha_i \\ \beta_i\end{bmatrix}=U(\Delta\theta_i)\begin{bmatrix}\alpha_i \\ \beta_i\end{bmatrix}$$

Where: $U(\Delta\theta_i)$ is the quantum rotation gate and $\Delta\theta i$ is the rotation angle of the i^{th} Q-bit. $\begin{bmatrix}\alpha_i \\ \beta_i\end{bmatrix}$ denotes the i^{th} Q-bit.

In this paper, aiming at the model of HFSP, the antibody should be encoded by real numbers. In order to decrease the calculation time and the complexity of algorithm,

we proposed another observational manner. We will describe it in detail in the flowing part b of section 5.2.

5 QIA for Hybrid Flow Shop Problem

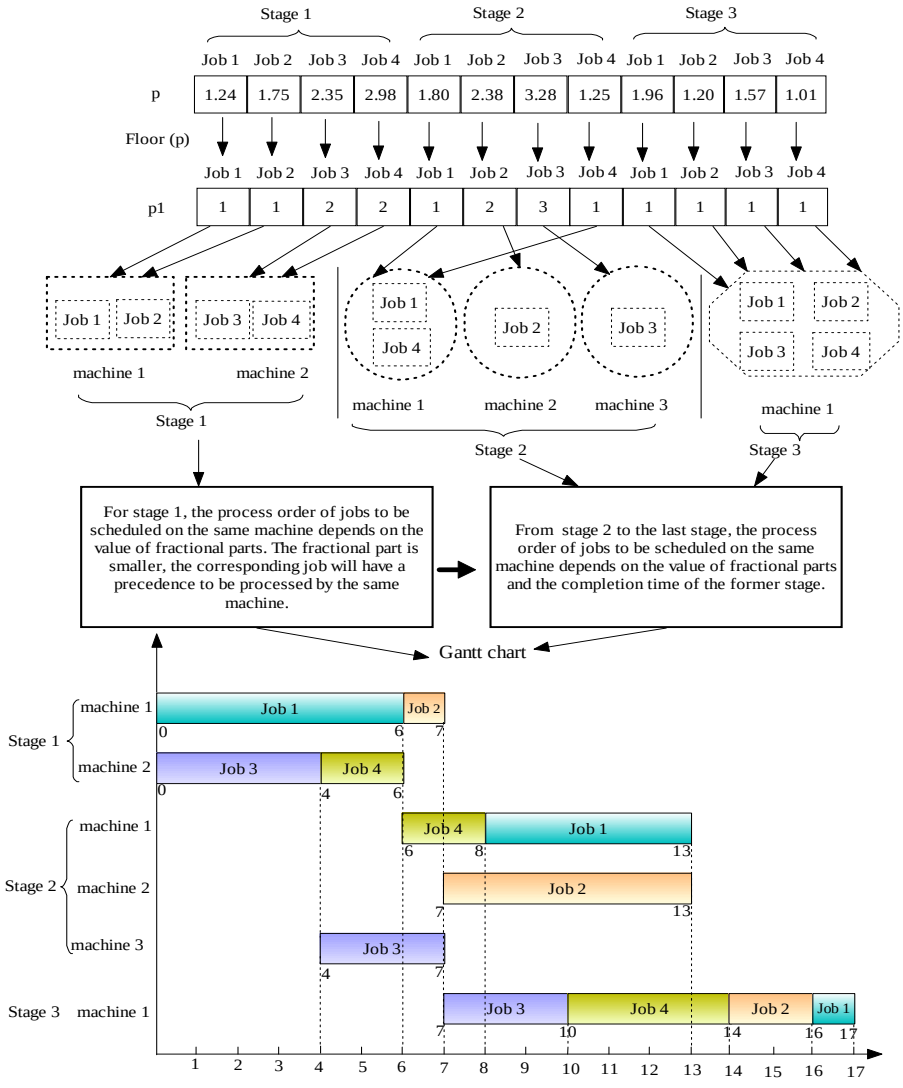

Figure 3: Solution representation

A job permutation represents the processing sequence. In the flow shop scheduling problem, it is common to use the numbers of jobs to represent a job permutation. For

instance, a flow shop scheduling problem has 3 jobs and 3 machines. A job permutation [3 1 2] denotes that job 3, job 1 and job 2 are processed by each machine in the given order, i.e. job 3 is first processed by each machine, job 1 is the second and job 2 is last one. But the hybrid flow shop scheduling problem uses a real number encoding whose fractional part is used to sort the jobs assigned to each machine and whose integer part is the machine number to which the job is assigned.

For example, HFSP has four jobs ($n=4$) and three processes ($g=3$). Stage one, stage two and stage three have two machines ($m(1)=2$), three machines ($m(2)=3$) and one machine ($m(3)=1$), respectively. We generate four real numbers randomly by uniform distribution $[1,1+m(k)]$ ($k=1,2,3$) for each stage. Processing times $pt=\begin{bmatrix} 6 & 5 & 1 \\ 1 & 6 & 2 \\ 4 & 3 & 3 \\ 2 & 2 & 4 \end{bmatrix}$ are given. As shown in Fig. 3, a solution is represented by a real number encoding $p=[1.24\ 1.75\ 2.35\ 2.98\ 1.80\ 2.38\ 3.28\ 1.25\ 1.96\ 1.20\ 1.57\ 1.01]$. At stage one, job1 and job2 are assigned to machine 1; job 3 and job 4 are assigned to machine 2. The order of jobs to be scheduled on machine 1 is job 2 followed by job 1. Because the fractional part of job 2 is greater than the fractional part of job 1.

At stage i ($i>1$), if two jobs are assigned to the same machine, we can arrange the jobs according to another rule. The rule is described as follows:

Every job bases on the completion time of the (i-1) stage to schedule its processed sequence, that is, the job which first completes the former stage will be first processed. If the completion time of the former stage is also the same, we can compare the values of genes. The job whose value of gene is smaller will be first processed. If the values are also equal, we can ensure the job processing sequence randomly.

5.1 Procedure of QIA for Hybrid Flow Shop Problem

Fig.4 shows the framework of QIA for solving the hybrid flow shop problem. From the flowchart, it can be seen that the QIA includes two parts which are QA and permutation-based immune algorithm (PIA).

In the following, we will describe the procedures of OIA for solving the HFSP.

Step 1: Let t=0 and memory cells $B(t)$ and Q-bits as an initial population are generated Quantum Computing for Beginners

$$P_Q(t)=\{p_1(t), p_2(t), p_3(t),..., p_N(t)\}$$

where $p_i(t)$ denotes the i^{th} individual (antibody) in the t^{th} generation and $p_i(t)$ is a Q-bit string.

Step 2: Convert $P_Q(t)$ to a job permutation $P_p(t)$ and evaluate population $P_Q(t)$, and record the best solution b.

Step 3: If the stopping condition is satisfied, the best solution is exported. Otherwise, go on to the following steps.

Step 4: $P_Q(t)$through rotation operation to generate $P_{Q'}(t)$ and $P_Q(t+1) = P_{Q'}(t)$. $P_p(t)$ combined with $B(t)$ can be regarded as an initial population $P_{p'}(t)$. Evaluate population $P_{p'}(t)$ and update $P_p(t)$ and $B(t)$.

Step 5: Perform operators of IA, including affinity calculation, selection, crossover, and mutation for $P_p(t)$ to generate $P_p\,'(t)$ .And convert $P_{Q'}(t)$ to $P_p(t)$.

Step 6: Combine $P_p(t)$ with $P_p\,'(t)$ and evaluate two populations, and also select the best half population as the next generation population and update the best solution b.

Step 7: Let t=t+1 and go back to Step 2.

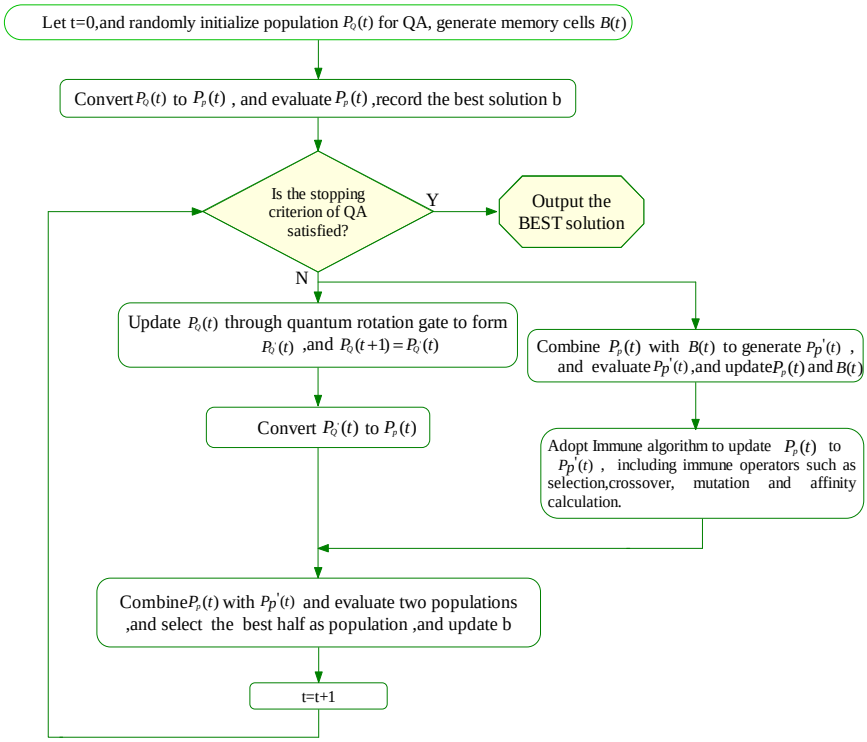

Figure 4: Flowchart of proposed QIA for Hybrid flow shop scheduling

5.2 Design of QIA for HFSP

To describe our QIA, we detail in the following the different steps of the algorithm including initial population, converting mechanism, rotation operation, selection, crossover and mutation.

a. Initial Population

Randomly generate Q-bit strings as an initial population $P_Q(t)$,

$$P_Q(t) = \{P_{Q1}(t), P_{Q2}(t), P_{Q3}(t), ..., P_{QN}(t)\}$$

Where $P_{Qi}(t)$ has m×n Q-bits and is defined as follows:

$$P_{Qi}(t)=\begin{bmatrix}\cos(t_{i1})&|&\cos(t_{i2})&|&...&|&\cos(t_{in})\\ \sin(t_{i1})&|&\sin(t_{i2})&|&...&|&\sin(t_{in})\end{bmatrix}$$

Where $t_{ij}=2*\pi*rand$ and rand is a random between 0 and 1; $i=1,2,...,m$; $j=1,2,...,n$; m is a number of work stages and n is a number of jobs.

b. Convert $P_Q(t)$ to Job Permutation $P_p(t)$

As mentioned above, job permutations are real numbers in HFSP. However, traditional observational manner of QA converts a Q-bit string to a binary string, which is not suitable for solving HFSP. If we want to calculate the scheduling objective, Q-bit strings must be converted to real number strings. Wang Ling [Li and Wang 07b] converts a Q-bit string to a binary string. Such a binary string should be further converted to a random-key representation. A random-key representation can be easily converted to a job permutation. Since the encoding of hybrid flow shop is different from flow shop, this method of converting method is not suitable for our problem. Therefore, we propose another new observational manner according to the encoding of HFSP.

The new observational manner proposed can be described as follows:

$$P_{pi}(t)=[((\cos(t_{k1}))^2\times m(k)+1),...,((\cos(t_{kj}))^2\times m(k)+1),...,((\cos(t_{kn}))^2\times m(k)+1)] \qquad (7)$$

where $P_{pi}(t)$ is a job permutation of the i^{th} individual at the t^{th} generation;

$m(k)$ denotes that stage k has $m(k)$ parallel machines.

$k\,(k=1,2,...,m)$ represents the k^{th} stage; j $(j=1,2,...,n)$ denotes the job j;

For example, there is a hybrid flow shop problem with two stages and three jobs. Each stage has 3 identical parallel machines. A Q-bit string is described as follows:

$$P_Q=\begin{bmatrix}\cos(t_{11})&|&\cos(t_{12})&|&\cos(t_{13})&|&\cos(t_{21})&|&\cos(t_{22})&|&\cos(t_{23})\\ \sin(t_{11})&|&\sin(t_{12})&|&\sin(t_{13})&|&\sin(t_{21})&|&\sin(t_{22})&|&\sin(t_{23})\end{bmatrix}$$

$$=\begin{bmatrix}0.3956&|&0.8299&|&0.6982&|&0.8555&|&-0.6737&|&0.8180\\ -0.9184&|&-0.5580&|&0.7159&|&-0.5178&|&-0.7390&|&0.5752\end{bmatrix}$$

The Q-bit string can be converted to real number string by using the new observational manner (Eq.(7)). The job permutation after converting can be given as follows:

$$P(t)=[(0.3956)^2\times3+1),(0.8299)^2\times3+1),(0.6982)^2\times3+1),(0.8555)^2\times3+1),(-0.6737)^2\times3+1),(0.8180)^2\times3+1)]$$

$$=[1.4695\ 3.0662\ 2.4624\ 3.1956\ 2.3616\ 3.0074]$$

This converting method can guarantee each solution is feasible and can be used to evaluate the objective value easily.

c. Rotation Operation

Because of the real number encoding, rotation operation used in this paper is different from that used in other papers such as [Huang and Tang 07a]. Rotation operation is defined as follows:

$$\begin{bmatrix} \alpha_i' \\ \beta_i' \end{bmatrix} = \begin{bmatrix} \cos(\Delta\theta_i) & -\sin(\Delta\theta_i) \\ \sin(\Delta\theta_i) & \cos(\Delta\theta_i) \end{bmatrix} \begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix} = U(\Delta\theta_i) \begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix}$$

Where $\begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix}$ denotes the present solution. $U(\Delta\theta_i)$ represents a Q-gate. The $\begin{bmatrix} \alpha_i' \\ \beta_i' \end{bmatrix}$ denotes the solution after performing rotation operation, $\Delta\theta_i$ is a rotation angle whose lookup table is shown in Table 1. In this Table, $f(r) < f(b)$ means that the solution b is preceded the solution r; x_i and $best_i$ are the i^{th} bit of the x solution and the best solution respectively.

$x_i > best_i$	$x_i = best_i$	$x_i < best_i$	$f(x) \geq f(best)$	$\Delta\theta_i$
false	false	true	false	-0.01π
false	false	true	true	-0.001π
false	true	false	false	-0.001π
false	true	false	true	0.001π
true	false	false	false	0.01π
true	false	false	true	0.001π

Table 1: Lookup table of rotation angle.

d. Selection

The roulette wheel selection operation is adopted in QIA. This selection strategy is widely used in standard GA. Chromosomes for reproduction are selected based on their fitness. Chromosomes with higher fitness will have a higher probability to be selected to enter into the next generation. In this paper, selection is not only based on fitness, but also based on concentration of antibody. Therefore, we use rate of expectation propagation as the selection standard. Selection probability of i^{th} individual is calculated as follows:

$$p_i = \frac{e(x_i)}{\sum_{i=1}^{N} e(x_i)}, i = 1, 2, ..., N \quad \textbf{(8)}$$

Where $e(x_i)$ denotes expectation propagation of x_i.

The roulette wheel selection operation can be described as follows:

First generate a number between 0 and 1 randomly, that is r=random (0, 1).

If $r \leq p_1$ then select the first individual;

If $p_1 + p_2 + ... + p_{i-1} < r \leq p_1 + p_2 + ... + p_{i-1} + p_i$

Then x_i will be selected to enter the next generation.

e. Expectation propagation

Expectation propagation includes affinity between antigen and antibody and affinity between the antibody *m* and antibody *n*.

Affinity between antigen and antibody m is very easy to be calculated, which can be found in Eq.(4). In order to obtain the affinity between the antibody *m* and

antibody *n*, the first step should calculate the similarity degree. We adopt the information entropy to calculate the similarity degree. Because HFSP have a encoding of real numbers, we must convert real numbers to discrete numbers, i.e., we take out the integer parts of the real numbers to compare the similarity degree. Then the affinity between the antibody m and antibody n can be calculated though Eq.(3). In this paper, the expectation propagation of each antibody is calculated by the Eq.(9).

$$em = (T / t) \times \frac{a x m}{C m} \tag{9}$$

where T is the maximal generation and t denotes the current generation.

f. Crossover operation

This paper uses a single point crossover for each process. This crossover is very easy to implement. Each process has a different crossover point and changes genes after the crossover point between two parents. For instance, the parents are described as follows:

parent one:[1.2 2.3 3.4 |2.7 4.5] & parent two:[2.3 1.6 3.1 |1.6 2.3]

Randomly generate one point and regard this point as a crossover point. If it satisfies the conditions of crossover, the two parents will change genes to generate the offspring. Supposed the crossover point is equal to 3, the offspring will be generated as follows:

offspring one:[1.2 2.3 3.4|1.6 2.3] & offspring two:[2.3 1.6 3.1|2.7 4.5]

g. Mutation operation

Mutation adopts two points at each process. A mutation operator is used to change the machine assignment. It is described as follows:

Randomly generate two points and do not change the genes between the two points. Interchange the genes in front of the left point with the genes after the right point. For example, the chromosome is [2.1 1.5| 2.3 1.9 4.2| 1.85 3.14 6.8]. Randomly generate two points. If generating point 3 and point 5, the offspring will be [1.85 3.14 6.8|2.3 1.9 4.2|2.1 1.5].

6 Computational Results

Experiments are carried out to evaluate the efficiency and effectiveness of the proposed QIA, which is implemented in MATLAB software and run on a Pentium(R) Dual 1.60Ghz PC. The test instances used in experiments are some benchmarks taken from Carlier and Neron [Engin and Döyen 04b]. The problem sizes vary from 10 jobs×5 stages to 15 jobs×10 stages. The processing times of jobs are uniformly distributed over the interval [3, 20]. Three characteristics are used to represent a problem, which are the number of jobs, the number of stages, and the number of identical machines at each stage. For instance, the notation of j10c5a2 means a 10-job, 5-stage problem. The letters j and c denote job and stage, respectively. Letter a defines the structure of the machine distribution at the stages. The last number 2 is the instance index for a specific type. Same instances are also studied in [Neron, Baptiste and Gupta 01b] and [Alaykýran, Engin and Döyen 07b] [Carlier and Neron 00b].The

lower bounds (LB) of Cmax can be found in [Neron, Baptiste and Gupta 01b]. In order to validate the performance of the proposed method, we compare the QIA with IA and QA on 41 benchmarks. Some Tables and figures are presented to show the comparison results.

6.1 Parameter settings

In this section, QIA and IA are fine-tuned to find good parameter settings that would lead to high quality solutions within reasonable computational time when applied to HFSP. All the parameters are listed in Table 2. In this paper, there are three main parameters, which are crossover probability (Pc), mutation probability (Pm) and affinity threshold (Tac1), respectively. Therefore, we conduct a small pilot experiment with the following values $Pc \in \{0.70, 0.75, 0.80, 0.85\}$, $Pm \in \{0.05, 0.1, 0.2, 0.3\}$ and $Tac1 \in \{0.70, 0.80, 0.90, 1.0\}$. j10c5a2 and j10c5b6, whose LB values are 88 and 115 respectively, are tested in the experiment. Population size M and iteration number T are selected different values for different type instances, which can be found in Table 3 and 4.

Parameter	Symbol
Population size/ Iteration number	M/T
Crossover probability/Mutation probability	Pc/ Pm
Affinity threshold	Tac1
Number of tests	S
Times number of reaching LB value among the 10 times	N
Minimum/Maximum/ Average value of makespan among the 10 times	Min/ Max/ Avg

Table 2: Computational Parameters

IA								
Pc (M=20; T=20;Pm=0.1; Tac1=0.9; S=10)								
INS	**Min/Avg**	**N**	**Min/Avg**	**N**	**Min/Avg**	**N**	**Min/Avg**	**N**
	Pc=0.70		Pc=0.75		Pc=0.80		Pc=0.85	
j10c5a2	88/88.8	6	88/88.5	7	88/88.6	7	88/89.1	6
j10c5b6	115/115.6	7	115/115.4	8	115/115.5	7	115/116	7
Pm (M=20; T=20;Pc=0.75; Tac1=0.9; S=10)								
INS	Pm=0.05		Pm=0.10		Pm=0.20		Pm=0.30	
j10c5a2	88/88	10	88/88.5	7	88/88.6	7	88/88.7	7
j10c5b6	115/115.2	9	115/115.4	8	115/115.7	8	115/115.5	8
Tac1(M=20; T=20;Pc=0.75; Pm=0.05; S=10)								
INS	Tac1=0.7		Tac1=0.8		Tac1=0.9		Tac1=1.0	
j10c5a2	88/88.2	9	88/88.3	9	88/88	10	88/88.2	9
j10c5b6	115/116.6	8	115/116.4	8	115/115.2	9	115/115.8	8

Table 3: Comparison results of IA with different parameters

QIA								
Pc (M=20; T=20;Pm=0.1; Tac1=0.9; S=10)								
INS	**Min/Avg**	**N**	**Min/Avg**	**N**	**Min/Avg**	**N**	**Min/Avg**	**N**
	Pc=0.70		Pc=0.75		Pc=0.80		Pc=0.85	
j10c5a2	88/88.3	7	88/88.1	9	88/88.3	9	88/88.4	8
j10c5b6	115/115.3	9	115/115.2	9	115/115.4	9	115/115.3	9
Pm (M=20; T=20;Pc=0.75; Tac1=0.9; S=10)								
INS	Pm=0.05		Pm=0.10		Pm=0.20		Pm=0.30	
j10c5a2	88/88	10	88/88.1	9	88/88.3	9	88/88.3	8
j10c5b6	115/115	10	115/115.2	9	115/115.3	8	115/115.2	9
Tac1(M=20; T=20;Pc=0.75; Pm=0.05; S=10)								
INS	Tac1=0.7		Tac1=0.8		Tac1=0.9		Tac1=1.0	
j10c5a2	88/88.2	9	88/88.2	9	88/88	10	88/88.1	9
j10c5b6	115/115.4	9	115/115.5	9	115/115	10	115/115.4	9

Table 4: Comparison results of QIA with different parameters

Table 3 and 4 show the comparison results of IA and QIA with different parameters. According to the results, we give the best parameters about Pc, Pm and Tac1, which are Pc=0.75; Pm=0.05; Tac1=0.90, respectively. It can also be seen that the results for the parameters with different values are close, which means the parameters have a small impact on the performance of IA and QIA, i.e., the methods are stable. These parameter settings will be used when comparisons are made in the forthcoming experiments in this paper.

6.2 Comparison of a and b Type Benchmarks

INS	**LB**	**IA**		**QA**		**QIA**	
		Min/Avg/Max	**N**	**Min/Avg/Max**	**N**	**Min/Avg/Max**	**N**
j10c5a2	88	88/88/88	10	88/88.8/91	6	88/88/88	10
j10c5a3	117	117/117.1/118	9	117/117.2/118	8	117/117/117	10
j10c5a4	121	121/121/121	10	121/121.6/123	6	121/121/121	10
j10c5a5	122	122/122/122	10	122/122/122	10	122/122/122	10
j10c5a6	110	110/110.8/112	5	110/110.9/112	4	110/110/110	10
j10c5b1	130	130/130.1/131	9	130/130.3/132	8	130/130/130	10
j10c5b2	107	107/107.6/110	8	107/108.1/110	4	107/107/107	10
j10c5b3	109	110/110.8/112	0	110/111/113	0	109/109.3/110	7
j10c5b4	122	122/125.5/127	1	124/127.1/129	0	122/122.2/124	9
j10c5b5	153	153/153/153	10	153/153/153	10	153/153/153	10
j10c5b6	115	115/115.2/117	9	115/116.4/119	5	115/115/115	10

Table 5: Comparison of IA, QA and QIA
(M=20;T=100;Pc=0.75;Pm=0.05;Tac1=0.90;S=10)

Table 5 illustrates the comparison results of 11 easy instances. The maximum generation is 100. If T is equal to 100, the search will stop and the best solution is

accepted as the final schedule. Each instance has been run 10 times randomly. From Table 5, it can be seen that QIA is able to find the LBs of Cmax for each instance. Except for instance j10c5b3, IA can also find the LBs of the remaining instances. For QA, it can obtain the LBs except for j10c5b3 and j10c5b4. Notice the value of N on each instance, which denotes the times number of reaching LB value among the 10 times. Except for instance j10c5b3 and j10c5b4, QIA can find the LBs 10 times on the remaining instances, which means QIA performs better than both IA and QA. In terms of average value, IA provides better results than QA.

INS (LB)	**M**	**IA**		**QA**		**QIA**	
		Min/Avg/Max	**N**	**Min/Avg/Max**	**N**	**Min/Avg/Max**	**N**
j15c10b1 (222)	20	223/223.2/225	0	223/225.2/227	0	223/223/223	0
	40	223/223/223	0	223/224/227	0	222/222.9/223	1
j15c10b2 (187)	20	187/188/189	5	189/190.2/192	0	187/187/187	10
	40	187/187.5/189	7	187/188.9/191	2	187/187/187	10
j15c10b3 (222)	20	222/222/222	10	222/222.3/224	8	222/222/222	10
	40	222/222/222	10	222/222.1/223	9	222/222/222	10
j15c10b4 (221)	20	221/221.2/223	9	221/222.6/227	5	221/221/221	10
	40	221/221/221	10	221/221.8/224	6	221/221/221	10
j15c10b5 (200)	20	202/203/205	0	204/205.6/209	0	201/201.9/202	0
	40	202/202.7/204	0	203/204.6/208	0	200/201.8/202	1
j15c10b6 (219)	20	219/221.3/222	1	221/222/223	0	219/219.4/220	6
	40	219/220.6/222	2	220/221.8/222	0	219/219.2/220	9

Table 6: Comparison of IA, QA and QIA
(T=200;Pc=0.75;Pm=0.05;Tac1=0.90;S=10)

INS (LB)	M	IA		QA		QIA	
		Min/Avg/Max	N	Min/Avg/Max	N	Min/Avg/Max	N
j15c10a1 (236)	20	236/236.2/238	9	236/238.2/240	0	236/236/236	10
	40	236/236/236	10	236/237/240	5	236/236/236	10
j15c10a2 (200)	20	209/212/214	0	211/213.6/218	0	204/204.9/213	0
	40	205/207.6/211	0	210/213.3/217	0	200/203.7/205.1	1
j15c10a3 (198)	20	201/203.7/206	0	205/210.8/217	0	198/199.8/200	4
	40	200/201.8/205	0	202/208.9/214	0	198/199.2/200	6
j15c10a4 (225)	20	229/229.7/231	0	229/230.1/231	0	229/229/229	0
	40	228/229/230	0	229/229.4/231	0	225/228.6/229	1
j15c10a5 (182)	20	189/191.2/193	0	191/193.9/197	0	183/183/183	0
	40	186/189.2/192	0	189/193.6/196	0	182/182.9/183	1
j15c10a6 (200)	20	202/204.3/207	0	205/210.7/215	0	200/200/200	10
	40	200/202.7/204	1	205/209.7/215	0	200/200/200	10

Table 7: Comparison of IA, QA and QIA
(T=400;Pc=0.75;Pm=0.05;Tac1=0.90;S=10)

Table 6, 7 and 8 summarize the computational results obtained by the three methods on another 18 instances. The tests use different population sizes or iteration numbers to compare the performance of the three methods.

Table 6 and 7 show that when the population size is larger, the Min values of each algorithm are getting better and the times of obtaining the LB of Cmax are correspondingly increasing. It is clear that the QIA can get better solutions than IA and QA, which indicates the consistency of the proposed technique in reaching the region of better solutions. IA performs better than QA, which means IA is a feasible and effective method for hybrid flow shop scheduling problem. It should be pointed out that for some instances, such as j15c10b2 and j15c10a6, the results of QIA in population size 20 are better than the results of IA and QA in population size 40, which means QIA can reduce the population size of the algorithm and improve the searching efficiency.

INS	LB	T	IA	QA	QIA
			Min/Avg/Max	Min/Avg/Max	Min/Avg/Max
j10c10a1	139	400	142/144/148	143/145.8/147	139/139/139
		600	141/143.7/147	142/145.3/147	139/139/139
j10c10a2	158	400	163/163.8/165	164/165.8/168	160/160.7/163
		600	160/163.3/165	163/164/167	158/159.9/163
j10c10a3	148	400	150/151.5/153	152/153.5/155	148/148.6/151
		600	150/150.4/152	151/152.9/155	148/148.5/150
j10c10a4	149	400	151/153.6/156	154/155.3/157	149/150.9/151
		600	151/153.2/154	154/154.6/156	149/150.3/151
j10c10a5	148	400	149/150.6/153	150/153.1/155	148/148/148
		600	149/150.3/152	150/152/154	148/148/148
j10c10a6	146	400	147/149.6/152	147/150.4/153	146/146.1/147
		600	147/149.2/150	147/150.2/152	146/146/146

Table 8: Comparison of IA, QA and QIA
(M=20; Pc=0.75; Pm=0.05; Tac1=0.90; S=10)

Table 8 lists the computational results of instance j10c10a1 to j10c10a6 in different iteration numbers. These instances are little harder than the above instances. QA and IA can't obtain the LBs of Cmax, although they can near the LBs, while QIA can find the optimal values of all the instances, which means QIA can avoid premature convergence and escape the local optima.

Fig. 5, 6 and 7 illustrate the typical convergence of the above-mentioned cases which are j10c5b3, j10c5b4, j10c15a2, j10c15a5, j10c10a2 and j10c10a5, respectively. It can be observed that QA is easy to get into local optimization. But QIA which combines IA and QA Quantum can quicken up convergence speed and ensure global astringency.

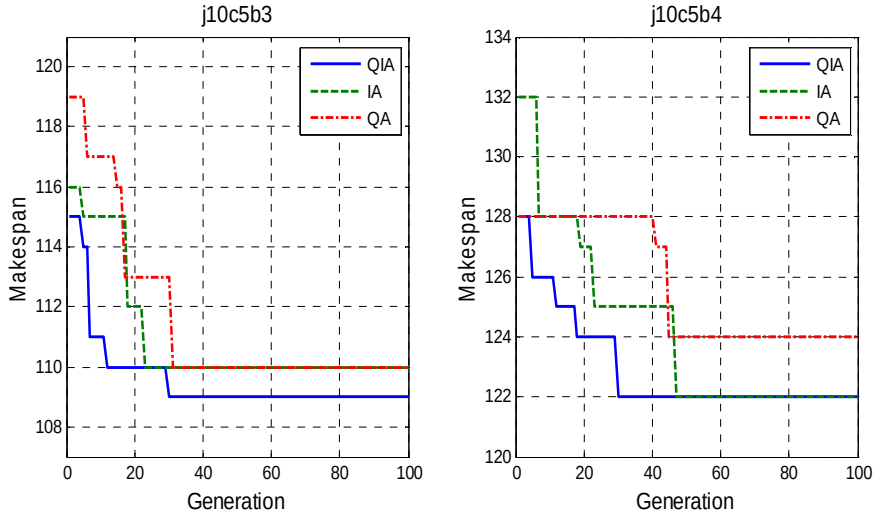

Figure 5: Comparison of convergence rates about j10c5b3 and j10c5b4

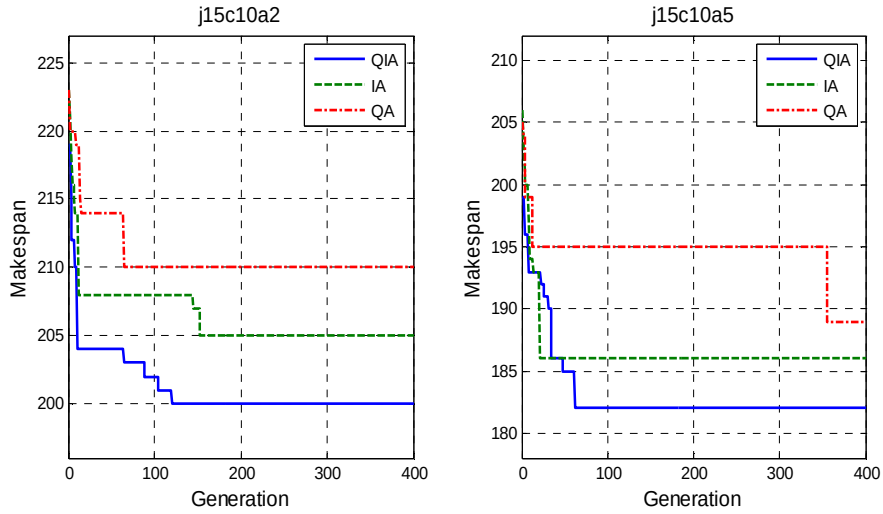

Figure 6: Comparison of convergence rates about j10c15a2 and j10c15a5

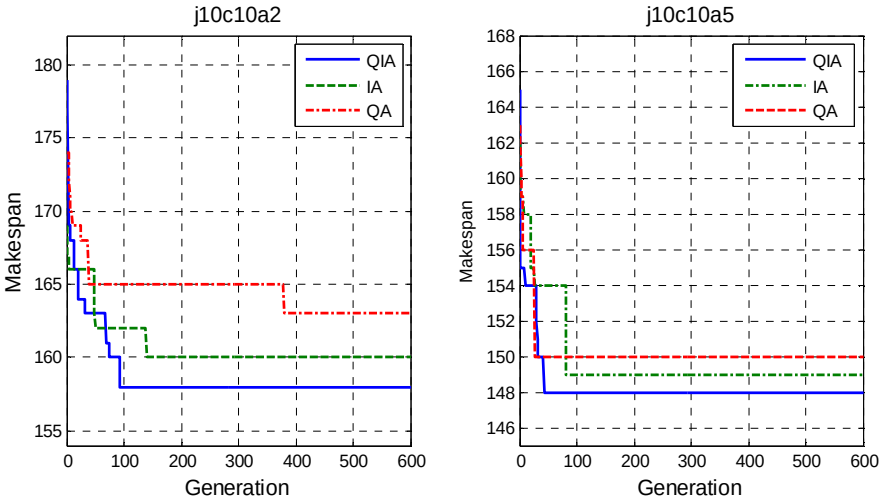

Figure 7: Comparison of convergence rates about j10c10a2 and j10c10a5

6.3 Comparison of c and d Type Benchmarks

In [Neron, Baptiste and Gupta 01b], some instances are grouped as hard problems. For these instances, they could not reach the optimal solutions in a short time. To further demonstrate the effectiveness of QIA, an additional simulation is carried out to compare the QIA with IA and QA on 12 hard instances, which can be found in Table 9.

INS	LB	IA	QA	QIA
		Min/Avg/Max	Min/Avg/Max	Min/Avg/Max
j10c5c1	68	76/77.1/80	79/80.5/83	69/73/75
j10c5c2	74	80/81.8/83	82/84.1/86	76/78.5/82
j10c5c3	71	77/79.5/80	81/84.3/86	74/75.8/77
j10c5c4	66	81/82.9/84	81/85.8/89	75/77.7/81
j10c5c5	78	83/85.7/88	87/88.2/90	79/81.7/84
j10c5c6	69	77/78.5/80	80/82.4/85	72/74.4/76
j10c5d1	66	73/76.4/78	75/78.5/82	69/71.7/74
j10c5d2	73	81/83.6/86	85/88/90	76/80.9/84
j10c5d3	64	71/72.1/73	75/76.7/79	68/70.4/73
j10c5d4	70	79/81.2/82	83/85.5/87	75/77/81
j10c5d5	66	75/77.1/79	78/80.7/83	71/72.8/75
j10c5d6	62	68/70.1/71	74/76/78	64/66.9/70

Table 9: Comparison of IA、QA and QIA
(M=40;T=800;Pc=0.75;Pm=0.05;Tac1=0.90; S=10)

From Table 9, we can see that QIA gets better solutions than IA and QA. Overall, these results indicate that QIA not only demonstrates the effective utilization in

population training, driving the immune evolutionary process in an appropriate direction and the fast convergence, but it also performs exploitation for solution improvement.

However, another point we can observe is that all the methods could not reach the LB values in 800 generations. That is because the complexity of these instances is significantly higher than the easy instances. It should be noted that for some easy instances, like j10c5a2 and j10c5b1, QIA can obtain the optimal solutions even within 10 generations. However, the two instances have the same size with the instances in Table 9. Therefore, it is very hard to obtain the LBs for the proposed methods in the short computational time.

7 Conclusions

This paper discusses the problem of hybrid flow shop scheduling, whose objective is to minimize makespan. The proposed QIA inspired by concept of quantum computer and immune algorithm has been applied to HFSP. A novel technique of evolving qubit to enable QIA to handle real number encoding directly has also been developed.

The experimental results have showed the superiority of the QIA compared with IA and QA by allowing IA and QA to have the same generations. The proposed QIA is capable of finding most optimal solutions of the benchmarks and often better than those reported for other algorithms. QIA provides the advantage of giving a greater diversity by using qubit encoding, which can avoid premature convergence and obtain better solutions.

The proposed method can be further extended and applied to other optimization problems in the future. Additionally, other performance indicators and process constraints about HFSP can be considered.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (Grant No. 60804052), Innovative Foundation of Shanghai University, and Scientific Research Special Fund of Shanghai Excellent Young Teachers. In addition, the authors would like to thank Dr. Emmanuel Neron's help.

References

[Abramson (63)] Abramson, N.: "Information Theory and Coding"; McGrawHill, New York (1963)

[Alaykýran, Engin and Döyen 07b] Alaykýran, K., Engin, O., Döyen, A.: "Using Ant Colony Optimization to Solve Hybrid Flow Shop Scheduling Problems"; Int. J. Adv. Manuf. Technol (International Journal of Advanced Manufacturing Technology), 35, 5-6 (2007), 541-550.

[Alisantoso, Khoo and Jiang 99b] Alisantoso, D., Khoo, L,P., Jiang, P.Y.: "An Immune Algorithm Approach to the Scheduling of a Flexible PCB Flow Shop"; Int J Adv Manuf Tech (International Journal of Advanced Manufacturing Technology), 22, 11-12 (2003), 819-827.

[Allahverdi and Al-Anzi 08b] Allahverdi, A., Al-Anzi, F.S.: "The Two-Stage Assembly Flowshop Scheduling Problem with Bicriteria of Makespan and Mean Completion Time"; Int J

Adv Manuf Tech (International Journal of Advanced Manufacturing Technology), 37, 1-2 (2008), 166-177.

[Arthanari 71b] Arthanari, T.S., Ramamurthy, K.G.: "An Extension of Two Machines Sequencing Problem"; Opsearch, 8, 1 (1971), 10-22.

[Benioff 80b] Benioff , P.: "The Computer As a Physical System: A Microscopic Quantum Mechanical Hamiltonian Model of Computers As Represented by Turing Machines"; J. Stat. Phys (Journal of Statistical Physics), 22, 5 (1980), 563-591.

[Carlier and Neron 00b] Carlier, J., Neron, E.: "An Exact Method for Solving the Multiprocessor Flowshop"; R.A.I.R.O- R.O. (RAIRO-. Operations Research), 34, 1 (2000), 1-25.

[Carrano, Guimaraes, Takahashi, Neto and Campelo 07b] Carrano, E.G., Guimaraes, F.G., Takahashi, R.H.C, Neto, O.M., Campelo, F.: "Electric Distribution Network Expansion under Load-Evolution Uncertainty Using an Immune System Inspired Algorithm"; IEEE Trans. on Power Sys.(IEEE Transactions on Power Systems), 22, (2007), 851-861.

[Chen, Kim and Jong 97b] Chen, J.S., Kim, M.K., Jong, H.K.: "Shape Optimization of Electromagnetic Devices Using Immune Algorithm"; IEEE Trans. Magn (IEEE Transactions on Magnetics), 33, 2 (1997), 1876-1879.

[Dasgupta 02b] Dasgupta, D.: "Special Issue on Artificial Immune System"; IEEE Trans. Evol. Comput (IEEE Transactions on Evolutionary Computation), 6, 3 (2002), 225-256.

[De Castro and Von Zuben 02b] De Castro, L.N., Von Zuben, F.J.: "Learning and Optimization Using the Clonal SelectionPrinciple". IEEE Trans. Evol. Comput (IEEE Transactions on Evolutionary Computation), 6, 3 (2002), 239-251.

[Engin and Döyen 04b] Engin, O., Döyen, A.: "A New Approach to Solve Hybrid Flow Shop Scheduling Problems by Artificial Immune System". Future. Gener Comput Syst (Future Generation Computer Systems), 20, 6 (2004), 1083-1095.

[Grover 96a] Grover, L.K.: "A Fast Quantum Mechanical Algorithm for Database Search"; In: Proc. of the 28th Annual ACM Symp on the Theory of Computing, ACM Press, (1996), 212-22.

[Gupta 88b] Gupta, J.N.D.: "Two-Stage, Hybrid Flow shop Scheduling Problem"; J. Oper. Res. Soc. (Journal of the Operational Research Society), 39, (1988), 359-364.

[Han and Kim 00a] Han, K.H., Kim, J.H.: "Genetic Quantum Algorithm and Its Application to Combinatorial Optimization Problem"; In: Proc.Congr.Evol.Comput, (2000), 1354-1360.

[Han and Kim 02b] Han, K.H., Kim, J.H.: "Quantum-Inspired Evolutionary Algorithm for a Class of Combinatorial Optimization"; IEEE Trans. Evol. Comput (IEEE Transactions on Evolutionary Computation), 6, 6 (2002), 580-593.

[Han and Kin 04b] Han, K.H., Kim, J.H.: "Quantum-Inspired Evolutionary Algorithms with a New Termination Criterion, HεGate, and Two-Phase Scheme"; IEEE Trans. Evol. Comput (IEEE Transactions on Evolutionary Computation), 8, 2 (2004), 156-169.

[Huang and Tang 07a] Huang, Y.R., Tang, C.L., Wang, S.: "Quantum-Inspired Swarm Evolution Algorithm"; In: International Conference on Computational Intelligence and Security Workshops. (2007), 208-211.

[Jang, Han and Kim 03a] Jang, J.S., Han, K.H., Kim, J.H.: "Quantum-Inspired Evolutionary Algorithm Based Face Verification"; In: Proc. Genetic Evolutionary Computation Conf. LNCS, Springer, Berlin, Germany (2003), 2147-2156.

[Khoo and Situmdrang 03b] Khoo, L.P., Situmdrang, T.D.: "Solving the Assembly Configuration Problem for Modular Products Using an Immune Algorithm Approach"; Int. J. Prod. Res (International Journal of Production Research), 41, 15 (2003), 3419-3434.

[Layeb and Saidouni 07b] Layeb, A., Saidouni, D.E.: "Quantum Genetic Algorithm for Binary Decision Diagram Ordering Problem"; IJCSNS (International Journal of Computer Science and Network Security), 7, 9 (2007), 130-136.

[Li and Wang 07b] Li, B.B., Wang, L.: "A Hybrid Quantum-Inspired Genetic Algorithm for Multiobjective Flow Shop Scheduling"; IEEE Trans Syst, Man Cybern, Part B, Cybern (IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics), 37, 3 (2007), 576-591.

[Mori, Tsukiyama and Fukuda 93b] Mori, K., Tsukiyama, M., Fukuda, T.: "Immune Algorithm with Searching Diversity and Its Application to Resource Allocation Problem"; Trans Inst Electr Eng Jpn (Transactions Institute of electronics, information, and communication engineers, Japan), 113-C, 10 (1993), 872-878.

[Narayanan and Moore 96a] Narayanan, A., Moore, M.: "Quantum-Inspired Genetic Algorithm"; In :Proc. IEEE Int. Conf. Evol. Comput, Piscataway (1996), 61-66.

[Narayanan 99a] Narayanan, A.: "Quantum Computing for Beginners"; In: Proc. of the 1999 Congress. Evol. Comput, (1999), 2231-2238.

[Neron, Baptiste and Gupta 01b] Neron, E., Baptiste, P., Gupta, J.N.D.: "Solving Hybrid Flow Shop Problem Using Energetic Reasoning and Global Operations"; Omega, 29, 6 (2001), 501-511.

[Portmann, Vignier, Dardilhac and Dezalay 98b] Portmann, M.C., Vignier A., Dardilhac, D., Dezalay, D.: "Branch and Bound Crossed with GA to Solve Hybrid Flowshops"; Eur J Oper. Res (European Journal of Operational Research), 107, 2 (1998), 389-400.

[Ruiz and Maroto 06b] Ruiz, R., Maroto, C.: "A Genetic Algorithm for Hybrid Flowshops with Sequence Dependent Setup Times and Machine Eligibility"; Eur.J.Oper.Res (European Journal of Operational Research), 169, (2006), 781-800.

[Shor 94a] Shor, P.W.: "Algorithms for Quantum Computation: Discrete Logarithms and Factoring"; In: Proc. of the 35th FOCS, (1994), 124-134.

[Tavakkoli-Moghaddam, Rahimi-Vahed and Mirzaei 07b] Tavakkoli-Moghaddam, R., Rahimi-Vahed, A., Mirzaei, A. H.: "A Hybrid Multi-Objective Immune Algorithm for a Flow Shop Scheduling Problem with Bi-Objectives: Weighted Mean Completion Time and Weighted Mean Tardiness"; Inf.Sci (Information Sciences), 177, (2007), 5072-5090.

[Tsai, Ho, Liu and Chou 07b] Tsai, J.T., Ho, W.H., Liu, T.K., Chou, J.H.: "Improved Immune Algorithm for Global Numerical Optimization and Job-Shop Scheduling Problems"; Appl. Math. Comput (Applied Mathematics and Computation), 194, (2007), 406-424.

[Tsai and Chou 06b] Tsai, J. T., Chou, J.H.: "Design of Optimal Digital IIR Filters by Using an Improved Immune Algorithm"; IEEE Trans Sign Proce (IEEE Transactions on Signal Processing), 54, 12 (2006), 4582-4596.

[Wardono and Fathi 04b] Wardono, B., Fathi, Y.A.: "Tabu Search Algorithm for the Multistage Parallel Machine Problem with Limited Buffer Capacities"; Eur. J. Oper. Res ((European Journal of Operational Research), 155, 2 (2004), 380-401.

[Zandieh, Fatemi Ghomi and Moattar Husseini 06b] Zandieh, M., Fatemi Ghomi , S.M.T., Moattar Husseini, S.M.: "An Immune Algorithm Approach to Hybrid Flow Shops Scheduling With Sequence-Dependent Setup Times"; Appl. Math. Comput (Applied Mathematics and Computation), 180, 1 (2006), 111-127.