12

Hybrid particle swarm algorithm for job shop scheduling problems

Xiaoyu Song
School of Information and Control Enineering, Shenyang Jianzhu University
China

1. Introduction

Particle swarm optimization (PSO) algorithm is a kind of random optimization algorithm based on swarm intelligence. Swarm intelligence of PSO is produced by cooperation and competition between particles, which is used for guiding optimization search. PSO has been studied widely in many applications due to its good global searching ability. Currently PSO has been widely used in function optimization, neural network training, pattern classification, system control and other applications. The research on PSO in recent years indicates that PSO has fast convergence speed and good quality in solutions and fine robustness on optimization in multidimensional space functions or in dynamic objectives, which is suitable for project applications. In this chapter, we firstly introduce searching mechanism and algorithm processes of PSO. Then, some important problems are solved when PSO is used for job shop scheduling problems (JSSP), such as hybrid algorithms between particle swarm and other algorithms (HPSO), its deadlock issues, and the proof of PSO and HPSO convergence. This chapter can provide guides effectively for readers who apply particle swarm optimization algorithm.

2. Particle Swarm Optimization Algorithm for JSSP

Particle swarm optimization (PSO) is an evolutionary computation technique developed by Kennedy and Eberhart in 1995. The particle swarm concept was motivated by the simulation of social behaviors. PSO algorithm constitutes the simple conduct rules for each particle, remembers the best position of the particles, and shares the information between particles. That is, PSO algorithm achieves the optimization through cooperation and competition between the individuals of population. Comparing with other evolutionary algorithms, PSO algorithm retains the global search strategy based on population, and belongs to the simple model of movement and velocity. PSO algorithm can dynamically adjust the current search with unique memory. Considering the currency and validity of the algorithm, PSO algorithm has been studied in many applications.
Job shop scheduling problem (JSSP) is the simplification model of an actual problem, and among the most typical and hardest combinatorial optimization problems, which is a NP

complete problem. JSSP is often used to test the performance of the intelligent algorithms, which has important research and actual engineering meanings.

2.1 Introduction of PSO

PSO algorithm simulates the prey behavior of a bird flock. We can imagine a scene, a group of birds are random searching the food, and there is only a piece of food in this region. All the birds don't know the place of food, but they know distance from the current location to the place of food. What is the optimal strategy of searching the food? The most simple and effective strategy is to search the areas where are close to the birds.

PSO algorithm is motivated from the model, and is used to solve optimization problems. Each optimization is considered as a bird in the search space called a particle. Each particle has a fitness value that is decided by an optimization function, has a velocity to determine its flight direction and distance. PSO algorithm constructs an initial particle swarm (random solutions), then find the optimal solution through iterations. In each iteration, particles update their velocities and positions by tracking the two extreme values. An optimal solution is the individual extremum *pBest* that is found by the particle itself, and another optimal solution is the global individual extremum *gBest* that is found by the current population.

In traditional PSO algorithm, the particle swarm searches results in space of m*n dimension, each particle position means a result of the problem. The particle continuously adjusts itself position X to search new results. Let P_{id} denote the optimal result that the particle obtains. Let P_{gd} denote the optimal position that the particle swarm passed, the best total result in the search domain. Let V denote the speed of the particle.

$$V_{id}(t+1)= \omega \times V_{id}(t) + \eta_1 \times rand() \times (P_{id} - X_{id}(t)) + \eta_2 \times rand() \times (P_{gd} - X_{id}(t)) \quad (1)$$

Let $V_{id}(t)$ denote the speed of d dimension of particle i in generation t, ωdenote inertia weight, and '-' denote distance. Let η_1 and η_2 denote parameter, which can adjust P_{id} and P_{gd} respectively. *rand()* is the random number generation function. Therefore, we can get the next particle position.

$$X_{id}(t+1) = X_{id}(t) + V_{id}(t+1) \quad (2)$$

Considering the formula (1) and (2), we can find that the moving direction of particle is decided by three parts. That is, the initial speed $X_{id}(t)$ of the particle, and the optimum distance $P_{id} - X_{id}(t)$ that the particle passed, and the optimum distance $P_{gd} - X_{id}(t)$ that the particle swarm passed. The relative importance of three parts is decided by weighting coefficient ω, η_1, η_2。

The traditional PSO algorithm is described as follows.

STEP 1: Construct an initial particle swarm, that is randomly set the initial position X and the initial velocity V of each particle;

STEP 2: Calculate fitness value of each particle;

STEP 3: Compare each particle fitness value and its best position fitness value P_{id}, if better, update P_{id};

STEP 4: Compare each particle best position P_{id} and the best position of particle swarm P_{gd}, if better, update P_{gd};
STEP 5: adjust the velocity and position according the formula (1) and (2);
STEP 6: If termination conditions are satisfied (good enough position or the maximum number of iterations), then end; otherwise, go to 2.
PSO algorithm is a kind of evolutionary algorithm, which has several typical characteristics. First, the individual of population has been randomly initialized a random solution in the initialization process. Secondly, the better solutions of a new generation are obtained by searching the solution space. At last, a new generation of population is produced on the basis of the previous generation.

2.2 Convergence of PSO

The convergence of intelligence optimization algorithm is an important problem for the application of intelligent optimization algorithms. It is necessary that we discuss the convergence of PSO algorithm before solving a practical problem.

2.2.1 Convergence of Traditional PSO

It is a difficult problem to prove the convergence for an intelligent optimization algorithm. Two assumptions H1 and H2 proposed by Solis and Wet were introduced, which were used to prove the global convergence of the pure optimization algorithm with probability *1*. General requirements of stochastic optimization algorithm convergence are described as follows.
An optimization problem $\langle A, f \rangle$ and stochastic optimization algorithm D are given. x_k is the results of the *k-th* iterations, and results of the next iteration is x_{k+1} ($x_{k+1} = D(x_k, \zeta)$), where ζ is the solution that has been searched by algorithm D.
Condition H1: $f(D(x, \zeta)) \le f(x)$, if $\zeta \in \boldsymbol{A}$, set $f(D(x_k, \zeta)) \le f(\zeta)$, where A is the subset of the R^n, and A denotes the constraint space of the problem.
Conditions H1 random algorithm can guarantee the correctness; their objective is to ensure optimization of the solution to the fitness value of f (x) non-incremental.
A global convergence of the algorithm, which means sequence $\{f(x_k)\}_{k=0}^{\infty}$ can reach infimum $inf(f(x) : x \in \boldsymbol{A})$ in the feasible solution A. Because it is possible that the feasible solution A of optimization problem exist discontinuity spaces or isolated spots, infimum and other fitness value is incontinuous. Considering this potential problem, search infimum is defined in Lebesgue measure space as shown in formula 3, where $v[X]$ denotes Lebesgue measure in set X.

$$\Psi = inf(t : v[x \in A \mid f(x) < t] > 0) \tag{3}$$

Formula (3) implies that non-empty set of the search space is existent, where the fitness of its members infinitely are close to Ψ. The definition of $v[X]$and A guarantee that nonempty point does not exist in set A. So the algorithm can reach or be close to the infimum without searching all points of set A.
Therefore, the optimal region can be defined as the following formula, where $\varepsilon > 0$, $M \to \infty$.

$$R_{\varepsilon,M} = \begin{cases} \{x \in A \mid f(x) < \Psi + \varepsilon\}, & \text{finite}\Psi \\ \{x \in A \mid f(x) < -\mathrm{M}\}, & \Psi = -\infty \end{cases}$$

If the optimization algorithm find a point among $R_{\varepsilon, M}$, the point is an acceptable global optimal or near to the global optimum.

Condition H2: $\prod_{k=0}^{\infty}(1 - v_k[B]) = 0$, $\forall B \subseteq A$, s.t.$v[B] > 0$

Where $v_k[B]$ is probability measure in set B, and B is the *kth* iteration set of algorithm D. Algorithm D satisfies condition H2. It means, it is impossible that algorithm D searches the points among set B, and let $v[B] > 0$. Because $R_{\varepsilon, M} \subseteq$ A, it is possible that the global optimum can be found.

Theorem 1 (**Global Convergence):** Supposing that f is measurable and feasible solution space A is measurable subset of R^n, algorithm D satisfies condition H1 and H2. And algorithm D generates series $\{x_k\}_{k=0}^{\infty}$. Then

$$\lim_{k \leftarrow +\infty} P[x_k \in R_{\varepsilon,M}] = 1$$

$P[x_k \in R_{\varepsilon, M}]$ is probability measure in $R_{\varepsilon, M}$, and $R_{\varepsilon, M}$ is the *kth* iteration set of algorithm D. Considering theorem 1, the global convergence of stochastic algorithms must satisfy the conditions H1 and H2. Because each iteration of PSO algorithm has kept the best position, conditions H1 must be satisfied. However, utilizing Markov chain theory and mathematical theory of real variable, Dr. Van den Bergh has proved that PSO algorithm does not satisfy conditions H2.

2.2.2 Convergence of Improved PSO

Because the traditional PSO algorithm does not guarantee global convergence, the position and velocity update equations are improved for solving JSSP. Considering the formula (1) and (2), although v_k and x_k is multidimensional variable, each dimension is independent. Therefore the convergence analysis can be simplified to the one-dimensional. In order to expand the solution space of PSO algorithm, we adopt the velocity update equation and position update equation of particle i as follows:

$$v_i(t+1) = \alpha(P_i - x_i(t)) + \beta(P_g - x_i(t)) \quad (4)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (5)$$

In formula (4) and (5), α, $\beta(\alpha, \beta \in [0, 1])$ are random numbers. The part $\alpha(P_i - x_i(t))$ represents that the best private distance experience of the particle i in group t is inhabited by probability α; And the part $\beta(P_g - x_i(t))$ represents that the best group distance experience of all the particles in group t is inhabited by probability β.

It can be obtained from formula (4) and (5) that the bigger the value of α is the larger impact of P_i is, and the greater possibility of particle's moving to the local optimum will become;

Similarly, the bigger the value of β is the larger impact of P_g is and the greater possibility of particle's moving to the global optimum will become.
The particle i will stop moving when $x_i(t) = P_i = P_g$. Namely $x_i(t+1) = x_i(t)$. In order to expand the solution space of PSO algorithm, we save P_g as historical global best position and regenerate position $x_i(t+1)$ of particle i randomly in the solution space, which will make the particle i continue to search.
Through the operation, equation (5) can be deformed as follows:

$$x_i(t+1) = (1 - c)\, x_i(t) + c_1 p_i + c_2 p_g \tag{6}$$

When p_i, p_g fixed, equation (6) is a simple linear difference equation, when $x_i(0) = x_{i0}$, its solution is:

$$\begin{aligned} x_i(t) &= k + (x_{i0} - k)(1 - c)^t \\ k &= \frac{c_1 p_i + c_2 p_g}{c}, \quad c = c_1 + c_2 \end{aligned} \tag{7}$$

Considering the formula (7), the formula (6) has convergence if $|1 - c| < 1$. That is, if $t \to \infty$, then $x_i(t) \to \frac{c_1 p_i + c_2 p_g}{c}$. If $|1 - c| < 1$, then $0 < c_1 + c_2 < 2$. That is, if $0 < c_1 + c_2 < 2$, the evolution equation of improved PSO algorithm is asymptotic convergence. The convergence region shown in Fig. 1.

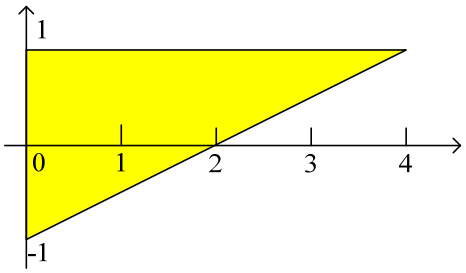

Fig. 1. Convergence region

Theorem 2: if $|1 - c| < 1$, $\lim_{t \to \infty} x_i(t) = p_g$.
See the formula (7), if $|1 - c| < 1$, then

$$\begin{aligned} \lim_{t \to \infty} x_i(t) &= k = \frac{c_1 p_i + c_2 p_g}{c} \\ x_i(t+1) &= x_i(t) - (c_1 + c_2)\, x_i(t) + c_1 p_i + c_2 p_g \end{aligned} \tag{8}$$

if $t \to \infty$,

$$\lim_{t\to\infty} x_i(t+1) = \lim_{t\to\infty} x_i(t) \tag{9}$$

Then

$$-(c_1+c_2)\lim_{t\to+\infty} x_i(t) + c_1 p_i + c_2 p_g = 0 \tag{10}$$

Considering theorem 1, if a random optimization algorithm satisfies condition H1 and H2, we can guarantee that the algorithm can converge to global optimal solution with probability *1*. We will discuss the problem whether the improved PSO algorithm is able to satisfy condition H1 and H2.

In the improved PSO algorithm, the solution sequence is$\{ p_{g,\ t}\}$, where t denotes evolutional generation, and $p_{g,\ t}$ denotes the best position of particle swarm in generation t. The function D is redefined by the improved PSO algorithm.

$$D(p_{g,t} x_i(t)) = \begin{cases} p_{g,t}, & f(p_{g,t}) \le f(x_i(t)) \\ x_i(t), & f(p_{g,t}) > f(x_i(t)) \end{cases} \tag{11}$$

It is easy to prove that the condition H1 is satisfied.

In order to satisfy the conditions H2, the sample space of particle swarm A must contain A. Namely,

$$A \subseteq \bigcup_{i=1}^{a} M_{i,t}, \tag{12}$$

Where $M_{i,\ t}$ is the support set of the sample space of particle i in generation t. Considering particle j, let $M_{i,\ t} = A$ when $x_j(\ t\ ) = p_i = p_{g.}$ Let the other particle i:

$$M_{i,t} = x_i(t-1) + \varphi_1(p_i - x_i(t-1)) + \varphi_2(p_g - x_i(t-1)) \tag{13}$$

Because of $0 \le \varphi_1 \le c_1$ and $0 \le \varphi_2 \le c_2$, $M_{i,\ t}$ is a super rectangle with vertices, where $\varphi_1 = \varphi_2 = 0$, $\varphi_1 = c_1$, and $\varphi_2 = c_2$. Let $v[\ M_{i,\ t} \cap A\ ] < v(\ A\ )$, when $max(\ c_1 \mid p_i - x_i(\ t - 1\ ) \mid,\ c_2 \mid p_g - x_i(\ t - 1\ ) \mid\ ) < 0.5 \times \mathrm{diam}(A)$, where diam(A) denotes the length of A. Considering condition H2, the length of $M_{i,\ t}$ is near to *0* when $t \to \infty$. Therefore, the measure $v[\ M_{i,\ t}\ ]$ of each $M_{i,\ t}$ is decreasing with the growth of generation t. And the measure $v[\bigcup_{i\neq j} M_{i,t}\ ]$ of union $\bigcup_{i\neq j} M_{i,t}$ is decreasing. Therefore there is N, let $v[\ \bigcup_{i\neq j} M_{i,t} \cap A\ ] < v[\ A\ ]$ when $t > N$.

Because of $M_{i,\ t} \subseteq$ A, Let $\bigcup_{i=1}^{A} M_{i,t}$ = A. In order to satisfy the conditions H2,we define A is the Borel subset($S = M_{i,\ t}$), then $v[\ S\ ] > 0$ and $v_t[\ S\ ] = \sum_{i=0}^{a} v_{i,t}[S] = 1$. Considering theorem 1, the improved algorithm can converge to a global optimal solution with probability *1*.

There is almost identical convergence between the improved PSO algorithm and the traditional PSO algorithm. That is, the parameter $x_i(t)$ can converge to the best location within the finite range. The traditional PSO algorithm does not guarantee global convergence, but the improved PSO algorithm can converge to a global optimal solution with probability *1* when generation t is near to ∞.

2.3 Convergence of PSO

Job shop scheduling problems (JSSP) is an important part of production scheduling of an enterprise, which is one kind of the most typical and hardest combinatorial optimization problems, an NP complete problem. The main task in scheduling, in terms of production targets and constraints, is to determine the precise process route, time, machine and operation for every process object. JSSP is often used to test the performance of the intelligent algorithms, which is significant for research and actual engineering.

Particle swarm optimization (PSO) algorithm is a kind of random optimization algorithm based on continuous optimization problems. PSO algorithm is less studied to solve JSSP. The PSO algorithm design of solving JSSP is difficult, and the efficient PSO algorithm design of solving JSSP is more difficult.

Leticia etc. construct the single machine scheduling algorithm based on random coding of JSSP, and the algorithm is a kind of retardation minimum time algorithm. The algorithm utilizes the dynamic mutation operators to ensure the diversity of particle populations. The algorithm has been tested respectively with 40 jobs and 50 jobs, and the algorithm achieves good results. Lina etc. construct PSO algorithm based on operation code to solve JSSP. They apply the crossover and mutation operation of GA in place of the update operations of velocity and position of PSO algorithm.

In the hybrid particle swarm optimization, Jerald.J etc. apply GA, SA and PSO algorithm to solve scheduling problems of flexible manufacturing systems. The hybrid algorithm optimizes machine idle time and reduces the cost of production tardiness. Liu etc. combine PSO algorithm and VNS. The hybrid algorithm minimizes the makespan of the flexible JSSP. Xia etc. design the hybrid PSO algorithm based on SA local search algorithm. The hybrid algorithm can solve multi-objective flexible JSSP. In order to minimize the makespan, Sha etc. construct the hybrid algorithm based on Hash table to solve JSSP. In the hybrid algorithm, Giffler-Thompson (G&T) algorithm is adopted to construct the feasible solution from the particle location of Hash table, and SWAP operation updates the particle velocity. The hybrid algorithm combines with TS algorithm based on block structure.

2.3.1 JSSP Description

Each instance of the problem J/ $/C_{max}$ is defined by a set of jobs, a set of machines and a set of operations. Each job consists of a sequence of operations, each of which has to be performed on a given machine for a given time. A schedule is an allocation of the operations to time intervals on the machines. The problem is to find the schedule that minimizes the

makespan subject to the following constraints: (i) the precedence of operations given by each job are to be respected, (ii) each machine can perform at most one operation at a time and (iii) the operations cannot be interrupted.
Let:

- J = {1, … ,n} denote the set of jobs;
- M = {1, … ,m} denote the set of machines;
- V = {0,1, … , $\tilde{n}$ +1} denote the set of operations, where 0 and $\tilde{n}$ +1 represent the dummy start and finish operations, respectively;
- A be the set of pair of operations constrained by the precedence relations, as in (i);
- V_k be the set of operations to be performed by the machine k;
- $E_k \subset V_k \times V_k$ be the set of pair of operations to be performed on the machine k and which therefore have to be sequenced, as specified in (ii);
- p_v and t_v denote the (fixed) processing time and the (variable) start time of the operation v, respectively. The processing time of the 0 and $\tilde{n}$ +1 operations is equal to zero, i.e., $p_0 = p_{\tilde{n}+1} = 0$.

Given the above assumptions, the problem can be stated as searching minimize $t_{\tilde{n}+1}$
subject to

$$\begin{aligned} t_j - t_i \geq p_i, \quad & (i, j) \in A, \\ t_j - t_i \geq p_i \vee t_i - t_j \geq p_j, \quad & (i, j) \in E_k, k \in M, \\ t_i \geq 0, \quad & i \in V. \end{aligned} \tag{14}$$

The first set of constraints represents the precedence relations among the operations of the same job, whereas the second set of constraints describes the sequencing of the operations on the same machine. These constraints impose that either $t_j - t_i \geq p_i$ or $t_i - t_j \geq p_j$. Any feasible solution of the problem (1) is called a schedule.
In this framework, it is useful to represent the job shop scheduling problem in terms of a disjunctive graph G:=(V,A, E), where V is the set of nodes, A the set of ordinary arcs (conjunctive) and E the set of disjunctive arcs. The nodes of G correspond to operations, the directed arcs correspond to precedence relations, and the disjunctive arcs correspond to operations to be performed on the same machine. More precisely, $E = \bigcup_{k=1}^{m} E_k$, where E_k is the subset of disjunctive arcs is related to a machine k; each disjunctive arc of E can be considered as a pair of opposite directed arcs. The length of an arc (i,j) ∈ A is p_i, the length of an disjunctive arc (i,j) ∈ E is either p_i or p_j depending on its orientation. The selection of a processing order on each machine involves the orientation of the disjunctive arcs, in order to produce an acyclic directed graph. A schedule on a disjunctive graph G consists in finding a set of orientations that minimizes the length of the longest path (*critical path*) in the resulting acyclic directed graph.
According to the Adams et al. method, the graph G can be decomposed into one direct subgraph D=(V , A), by deleting disjunctive arcs, and in m cliques G_k=(V_k , E_k), obtained from G by deleting both the conjunctive arcs and the dummy nodes 0 and $\tilde{n}+1$. A selection S_k in E_k contains exactly one arc between each pair of opposite arcs in E_k . A selection is acyclic since it does not contain any directed cycle. Moreover, sequencing the operations on

the machine k is equivalent to choosing an acyclic selection in E_k . A complete selection S is the union of selections S_k , one for each E_k , $k \in M$. S generates the directed graph $D_S = (V, A \cup S)$;S is acyclic if the associated directed graph D_S is acyclic. An acyclic complete selection S infers a schedule, i.e., a feasible solution of Problem.

In order to solve the job shop scheduling problem the best acyclic complete selection S^* that minimizes the value of the length of the longest critical path in the direct graph $D_{S^*}=(V,A \cup S^*)$must be determined.

The neighbourhood of the current solution can be formed by the solutions generated by inverting the direction of a disjunctive arc in the critical path of D_S . To this end, as stated by other authors, it is useful to decompose the critical path into a sequence of r blocks $(B_1,B_2, \ldots ,B_r)$. Each block contains the operations processed on the same machine; for each pair of consecutive blocks B_j,B_{j+1} with $1 \le j \le r$ the last operation of B_j and the first of B_{j+1} belong to the same job but are performed on different machines.

2.3.2 PSO for Solving JSSP

As for applying PSO to the job shop scheduling problem, the problem can be described as that n jobs are processed by m machines. A certain list such as $S_m = (O_i)$, $i=1, \ldots, n$, demonstrates the list of jobs processed on a machine, then the amount of possible lists is $n!$, list set $S = \{ S_k \mid k = 1, 2, \ldots, m\}$ is used to express the process that n jobs done by m machines, the whole possibility of solutions is $(n!)^m$. As job shop scheduling problem, when all the operations in the solution is configured, the best processed list that satisfies the efficiency index can be seeked. Therefore, for solving job shop scheduling problem by PSO, we only need to change m encoding of each particle to seek optimal solution. According to the above, definition of operating list in job shop problem is given here.

Definition 1: exchanging operation. In the operation list, operation O_i on position i and operation O_j on position j change their positions each other. This behavior is called exchanging operation, the operator is denoted as ⊗. For the list S, the exchange of O_i and O_j is expressed as $S \otimes (O_i,O_j)$, where, (O_i,O_j) denotes the operation exchange, which can be simply expressed as $O_{i \otimes j}$. Then $S' = S \otimes (O_i,O_j) = S \otimes O_{i \otimes j}$, S' denotes the list which has been disposed.

Example 1: with regard to the job shop problem in which 6 jobs are processed on m machines, the list done on machine m is $S_m = (2\ 4\ 6\ 1\ 3\ 5\ )$, for the list S_m, if operation 2 and operation 6 exchanges, their position are respectively 1 and 3, the exchange process can be described as following formula.

Here, $S'_m = S_m \otimes (O_1,O_3) = (2\ 4\ 6\ 1\ 3\ 5) \otimes (2,6) = (\ 6\ 4\ 2\ 1\ 3\ 5)$.

Definition 2: exchange list. The operation list composed of no less than one exchanges among operations is named as exchange list, which is denoted as CS, and $CS = (O_{1i \otimes 1j}, O_{2k \otimes 2l}, \ldots, O_{np \otimes nq})$. When the list only have one time exchange operate, $CS = (O_{1i \otimes 1j},)$, where the sequence $O_{1i \otimes 1j}$, $O_{2k \otimes 2l}$, …, $O_{np \otimes nq}$, denotes the sequence of exchanging operations in the list S.

Exchange list acts on certain fraction of S_m, and it means that all the exchange operation in the list acts on S_m one by one, namely $S'_m = S_m \otimes CS = S_m \otimes (O_{1i \otimes 1j}, O_{2k \otimes 2l}, \ldots, O_{np \otimes nq}) = [[S_m \otimes (O_{1i},O_{1j})\ ] \otimes (O_{2k},O_{2l})\ ] \ldots \otimes (O_{np},O_{nq})$.

Definition 3: Equal set of exchange list. Different exchange list acts on the same solution, maybe the same solution is obtained. All the exchange lists which products the same solution is called the equal set.

Definition 4: united operation of exchanged list. When more than two exchanging lists such as CS_1, CS_2, …, CS_n, which act on one list according to the sequence is named as united operate, moreover, the operator is denoted as ⊕, the unit of exchange list is expressed as HS, HS= $CS_1 \oplus CS_2 \ldots \oplus CS_n$. Through the principle stated above, it can be described as $S' = S \otimes HS = S \otimes (CS_1 \oplus CS_2 \ldots \oplus CS_n)$, where S' denotes the new operation list that S had been exchanged according to the exchange list.

Through definition 3 and definition 4, a new solution S' can be obtained after acting on solution S with CS_1 and CS_2. Supposed that there is another exchange list that acts on the solution S, if a same solution S' can be obtained, then the unite of CS_1 and CS_2 is equal to CS, which can be expressed as $CS = CS_1 \oplus CS_2$, CS and $CS_1 \oplus CS_2$ belong to the same equal set, generally speaking, CS is not sole.

Definition 5: Basic exchange list. In the equal set $\{CS_i\}$ of exchanging list, exchange list BS with least exchange operators is called basic exchange list of this equal set. Supposed X and Y are operation list on machine m, constructing an exchange list BS which satisfies $X = Y \otimes BS$, if BS is of least exchange operation, then BS is a basic exchange list, which is denoted as $BS = Y \rightarrow X$.

According to the following method, a basic exchange list can be constructed, supposed that two solutions of problem FT06 are given, the lists on machine m are respectively X and Y.

Eg: X= (1 2 3 4 5 6), Y= (2 6 3 1 5 4).

It can be seen that, in the operation X, $O_1 = 1$. and in operation Y, $O_4 = 1$, let Y's first operate exchanging $O_{1i \otimes 1}$ be $Y \otimes (O_1, O_4)$, then $Y1 = Y \otimes (O_1, O_4)$, there exists $Y1 = (1\ 6\ 3\ 2\ 5\ 4)$; in X, $O_2 = 2$, and in $Y1$, $O_4 = 2$, let the second exchange operate $O_{2k \otimes 2l}$ be $Y1 \otimes (O_2, O_4)$, then $Y2 = Y1 \otimes (O_2, O_4)$, there exists $Y2 = (1\ 2\ 3\ 6\ 5\ 4)$. Similarly, the third exchange operate $O_{3p \otimes 3q}$ is $Y2 \otimes (O_4, O_6)$, there exists $Y3 = Y2 \otimes (O_4, O_6) = X$. Spontaneously, $BS = (O_{1i \otimes 1j}, O_{2k \otimes 2l}, O_{3p \otimes 3q})$ is of the minimal exchange operations, which is named as a basic exchange list, namely, $BS = Y \rightarrow X$. Here, $BS = Y \rightarrow X = (O_{1i \otimes 1j}, O_{2k \otimes 2l}, O_{3p \otimes 3q}) = ((O_1, O_4) \oplus (O_2, O_4) \oplus (O_4, O_6))$.

Aiming at PSO used to solve job shop problem, formula of basic PSO is not fit for this new type algorithm, so the formulas are recreated as follows:

$$V_{id} = \alpha(X_{id} \rightarrow P_{id}) \oplus \beta(X_{id} \rightarrow P_{gd}) \quad (15)$$

$$X'_{id} = X_{id} \otimes V_{id} \quad (16)$$

Where α, β are random number and (α, $\beta \in [0, 1]$). $\alpha(X_{id} \rightarrow P_{id})$ expresses that all the exchange operations of basic exchange list $(X_{id} \rightarrow P_{id})$ are withheld by the probability α. Similarly, $\beta(X_{id} \rightarrow P_{gd})$ expresses that all the exchange operations of basic exchange list $(X_{id} \rightarrow P_{gd})$ are withheld by the probability β.

According to the formula (15) and (16), it can be seen that, the greater α is, the stronger P_{id} affects, the probability of moving towards to the local optimization is magnified. In the same way, the greater β is, the stronger P_{gd} effect, the probability of moving towards to the global optimization is magnified.

Due to the regularity of object functions, the optimal solution must be in the active scheduling set, so PSO uses the solution produced with G&T as the initial solution. For the random and widespread searching ability, the exchanging list based PSO is used to search globally. In the process of running PSO algorithm, if any infeasible solution appears, it must be adjusted. When there exists $P_i(t) = P_{id} = P_{gd}$ for the particle $P_i(t)$ of generation t, then recreate this particle, so that PSO algorithm for job shop problem is constructed.
The steps of solving JSSP by PSO are described as following:
Step1: Use G&T algorithm to produce an initial solution, initialize P_{id} with the initial solution, initialize P_{gd} with the best P_{id};
Step2: If the end condition is satisfied, go to Step6;
Step3: According to the position of X_{id}, calculate X_{id} 's next position X'_{id}, namely new solution;

a) $A = X_{id} \rightarrow P_{id}$ denotes that A acts on X_{id} to get P_{id} , where, A is a basic exchange list,;
b) $B = X_{id} \rightarrow P_{gd}$, where B is also a basic exchange list;
c) Calculate validity V_{id} of particle according to formula (15);
d) Calculate new position X'_{id} (solution) according to formula (16);

Step4: Adjust infeasible solution;
Step5: Calculate fitness:

a) If a better solution is got, then update P_{id};
b) If a better solution of the whole swarm is searched out, then update P_{gd}, simultaneously adopt G&T to recreate a new particle instead. Go Step2;

Step6: Show the optimal solution obtained by this algorithm (P_{gd}).
Adjustment of infeasible solution is described in hybrid PSO algorithm.

2.4 Summary

Particle swarm optimization (PSO) is an evolutionary computation technique developed by Kennedy and Eberhart in 1995. The particle swarm concept was motivated by the simulation of social behaviors. The original intent was to graphically simulate the graceful but unpredictable choreography of bird flock. In the section, we introduce search mechanisms and processes of PSO, and analyze the convergence of PSO theoretically. A new PSO algorithm is proposed based on exchanged factors and exchanged lists, which is put the PSO idea into the discrete field of JSSP.

3. Hybrid Particle Swarm optimization Algorithm for JSSP

Recently, the theorem of No Free lunch (NFL) is proposed for evaluating optimization algorithms by professor Wolpert and Macready of Stanford University. It is shown that there isn't a single solution that adapts to all problems effectively. Radcliffe and Surry have the same conclusion.
For example, if GA algorithm is better than SA algorithm when solving the problem set A, then SA algorithm must be better than GA algorithm when solving the problem set B. Considering all the circumstances, two algorithms have the same performance. Therefore, there is no kind of intelligent optimization algorithm better than the other intelligent optimization algorithms. That is, every method has its corresponding application circumstances.

In theory and practice, adopting a single intelligent algorithm is not enough for solving JSSP. The hybrid algorithm is an effective method, which enlarges the application domain and improves their performance. A hybrid algorithm combines effectively some features of several algorithms, such as optimization mechanism, process, search behavior, operation, and so on. The hybrid algorithm will have better optimization efficiency.

3.1 HSPSO

If adopting a single algorithm to solve job shop problems, it is hard to improve the local optimization after some running time of the algorithm, it is necessary to find out a method to escape from this local optimization. Therefore, a hybrid PSO algorithm based on exchanging list is proposed.

The design ideas of hybrid optimization algorithm HPSO are as follows: (1) Due to the regularity of object function, the optimal solution must be in the active scheduling set, so HPSO uses the solution produced with G&T as the initial solution. (2) For the randomly and widespread searching ability, the exchanging list based PSO is used to search globally. (3) In the process of running PSO algorithm, if an infeasible solution appears, it must be adjusted. (4) In order to avoid algorithm falling in a local optimization too early, TS exploiting strategy embedded critical operations based on exchanging neighbors is adopted to realize local parallel search, simultaneously improve the local search ability.

When there exists $P_i(t) = P_{id} = P_{gd}$ for the particle $P_i(t)$ of generation t, then adopt G&T algorithm to regenerate the particle, so that hybrid PSO algorithm for solving JSSP is constructed. The arithmetic frame is shown as Fig. 2.

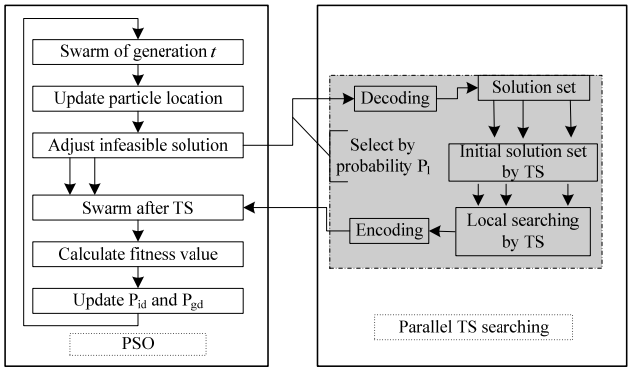

Fig. 2. Frame of the hybrid PSO algorithm

The steps of solving job shop problem by HPSO are described as following:

Step1: Use G&T algorithm to produce initial solution, initialize P_{id} with an initial solution, initialize P_{gd} with the best P_{id};

Step2: If the end condition is satisfied, go to Step7;

Step3: According to the position of X_{id}, calculate X_{id} 's next position X'_{id}, namely a new solution;

a) $A = X_{id} \rightarrow P_{id}$ denotes that A acts on X_{id} to get P_{id} , where, A is a basic exchange list,;
b) $B = X_{id} \rightarrow P_{gd}$, where B is also a basic exchange list;

c) Calculate validity V_{id} of particle according to formula (8);
d) Calculate new position X'_{id} (solution) according to formula (9);

Step4: Adjust infeasible solutions;
Step5: Select some solutions by the probability P_l to perform TS;
Step6: Calculate fitness:

a) If a better solution is gotten, then update P_{id};
b) If a better solution of the whole swarm is searched out, then update P_{gd}, simultaneously adopt G&T to recreate a new particle instead. Go Step2;

Step7: Show the optimal solution obtained by this algorithm (P_{gd}).

3.2 TS based on neighbor exchanging of critical operation

Taboo search(TS) algorithm is one of the best algorithms for solving job shop scheduling problem. So far, its running speed is faster, and it may provide a better induct within the whole searching field compared with other algorithms.

In order to obtain better searching results and higher efficiency, neighbors must be highly constrained and can be rapidly assessed. The possibility of moving to high quality solutions should be increased.

The local searching function is TS algorithm. To improve the efficiency of the local searching, we modify the TS algorithm. Firstly, the algorithm reduces the maximum that doesn't evolution. Secondly, a new exchanging strategy of neighbors is proposed based on critical operations so that TS algorithm can rapidly assess neighbors. We firstly indicate the neighbor exchanging based on the critical operation.

The feasible solution of job shop scheduling is usually denoted by the gantt graph. The gantt graph of 6×6 problem is illustrated in Fig. 3. In the figure, x-axis denotes the process time, y-axis denotes the machines, and every rectangular block marked (i, j) denotes the operation j of task i, and it is denoted as O_{ij}.

The optimized result of job shop scheduling problem is related to the length of the critical paths. The critical path is that the longest path without time intervals between operations in an available schedule. A solution always has many critical paths. For example, in Fig. 3, there are two critical paths. The first one is (4,1) (3,2) (5,1) (5,2) (4,2) (4,3) (5,3) (3,4) (3,5) (6,4) (5,5) (5,6) and another one is (4,1) (3,2) (5,1) (5,2) (4,2) (3,3) (3,4) (3,5) (6,4) (5,5) (5,6).

Furthermore, operations of the critical path can be decomposed into blocks. A block is a set of consecutive operations in a critical path in one machine. For example, operation (5,2), (4,2) and operation (4,3), (5,3), (3,4) in the first critical path forms the block respectively. Operation (5,2), (4,2), (3,3) and (3,4) in the second critical path forms the block respectively. For the two consecutive blocks, the last operation of the anterior block and the first operation of the latter block are always in the same task. For example the operation (3,3) and (3,4) of task 3 are in the same task.

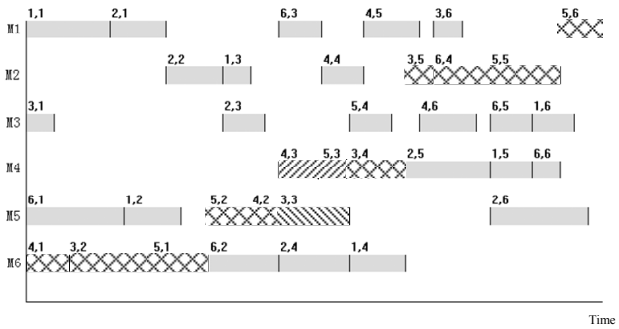

Fig. 3. 6×6 problem solution Gantt figure

Let $J_p(v)$ represent the previous operation of operation v in the same job, and $M_p(v)$ denote the previous operation of operation v processed in the same machine, $S_t(v)$ and $E_t(v)$ denote the start time and the end time of the operation v respectively.

Definition 6: Critical operation. In critical path, if operation satisfy the condition that $S_t(v)= E_t(M_p(v))=E_t(J_p(v))$, then v is called a critical operation.

When critical path is not unique, not all the neighbor exchanges can shorten the critical path. For example, Fig. 3 describes a gantt graph which shows the 6×6 problem, the exchange of (4,2) and (3,3) is unable to shorten the critical path. Because St(3,4) = MAX(Et(5,3), Et(3,3)), operation (3,4) is a critical operation, due to the dependency of critical operation (3,4) on operation (3,3), (5,3), although operation (3,3) is shortened, the neighbor exchange before the critical operation is unable to shorten the critical path.

The method of choosing neighbors based on the critical operations is as follows: when the critical path is sole, exchangeable neighbors in the critical path is considered as a set for neighbor selection; when the critical path is not sole, the exchangeable neighbors between the last critical operation and the last operation is viewed as a set for neighbor selection; TS algorithm selects an exchangeable neighbor (usually the best neighbor) from the above neighbors set to commute. If the set described above is null, then stop the current search with TS.

When TS algorithm search process runs for certain times, the quality of solution can not be improved, then TS algorithm stops.

Because of adopting new exchanging strategy of neighbors based on critical operations, TS algorithm reduces invalid neighbor exchanges, enhances searching efficiency, increases the possibility of escaping from the local optimization, and expands the searching range. Simultaneity, when there is no exchangeable neighbor, it indicates that the cost of improving the solution is too large, or the current solution is already the optimal solution. Then the searching is terminated.

3.3 HPSO Convergence

Dr. Van den Bergh has proved that PSO algorithm diverges both in the local region and the global region with the criteria presented by Solis and Wets, under which stochastic search algorithms can be considered as a global search algorithms, or merely locally search algorithms. We analyze the convergence of PSO algorithm with an optimum keeping

strategy and TS algorithm by Markov chain theory at a different aspect in this book, and HPSO algorithm based on PSO and TS algorithm is proved to be convergent. First of all, we give an introduction of Markov chain theory as follows.

Definition 7 (Markov chain) A stochastic sequence $\{X_n, n\in T\}$ and a discrete temporal series $T=\{0,1,2,\ldots\}$ are given, all state values corresponding to each X_n constitute the set of discrete state $S=\{s_0,s_1,s_2,\ldots\}$. The stochastic sequence $\{X_n, n\in T\}$ is called Markov chain as soon as the conditional probability satisfies the formula (17) as for each integer $n\in T$ and any $s_0,s_1,s_2,\ldots,s_{n+1}\in S$.

$$P\{X_{n+1}=s_{n+1} \mid X_0=s_0, X_1=s_1,\ldots,X_n=s_n\}=P\{X_{n+1}=s_{n+1} \mid X_n=s_n\} \tag{17}$$

Definition 8 (Transit ionprobability matrix) The conditional probability $p_{i,j}=P\{X_{n+1}=j \mid X_n=i\}$ is called transition probability of Markov chain $\{X_n, n\in T\}$, where $i,j \in S$. The matrix $\{P_{i,j}: i,j=1,\ldots,k\}$ is called $k\times k$ transition probability matrix.

Definition 9 (Finite homogeneous Markov chain) Markov chain is called finite homogeneous Markov chain if conditional probability $p_{i,j}(n)$ of Markov chain $\{X_n, n\in T\}$ has nothing to do with n and its set of state $S=\{s_0,s_1,s_2,\ldots s_k\}$ is finite, where $i,j \in I$. Then $p_{i,j}(n)$ is always regarded as $p_{i,j}$.

Lemma 1 Markov chain $\{X_n, n\in T\}$ with transition probability matrix P is irreducible if and only if the conditional probability satisfies formula (11) for any $s_i,s_j\in S$, where the set of state is $S=\{s_0,s_1,s_2,\ldots s_k\}$.

$$P\{X_{m+n}=s_j \mid X_m=s_i\}>0 \tag{18}$$

Lemma 2 Transition probability matrix P is irreducible if P can be turned into the form $\begin{pmatrix} C & 0 \\ R & T \end{pmatrix}$ by the same line and row transformation, where C is a strict positive irreducible stochastic matrix with dimension m, $R,T\neq 0$.Then the matrix.

$$P^{\infty}=\lim_{n\to\infty} P^n=\begin{pmatrix} C^{\infty} & 0 \\ R^{\infty} & 0 \end{pmatrix} \tag{19}$$

is a stable stochastic matrix, where $R^{\infty}=\lim_{k\to\infty}\sum_{i=0}^{k-1} T^i R C^{k-i}$, $P^{\infty}=1' p^{\infty}$, $p^{\infty}=p^0 P^{\infty}$ has nothing to do with the initial distribution, and $p_i^{\infty}>0 (1\le i\le m)$, $p_i^{\infty}=0 (m\le i\le k)$.

In this book, we set the change of the group made up of social collaboration S, self adapting A and competition C three basic evolution operations, where social collaboration S means that the particle adjusts its movement by cooperating with the best position P_g of the group; the self adapting A indicates that the particle adjusts its movement at the next moment by cooperation between cognition part $\alpha(P_i - x_i(t))$ and social collaboration part $\beta(P_g - x_i(t))$; All old particles $x_i(t)$ are totally replaced by new particles $x_i(t+1)$ with optimum keeping strategy to update their self best position and group position. Therefore, the course of transformation can be presented respectively by stochastic matrix P_S, P_A and P_C, and the transition probability matrix of TS algorithm is presented by stochastic matrix P_T.

Theorem 3 The hybrid algorithm HPSO based on PSO and TS algorithm is finite homogeneous Markov chain.

Proof: Since the probability of group in next state rests with the current state, which is independent of the past state, HPSO algorithm with the set of finite state $S=\{s_0,s_1,s_2,\ldots,s_k\}$ is Markov chain. Suppose that P_S, P_A, P_C and P_T are independent of time intervals, then the searching course of HPSO can be noted by a transition probability matrix with one step $P=P_T[P_C(P_SP_A)]$, which is independent of time intervals as well. Therefore, the whole search course of HPSO is finite homogeneous Markov chain.

The design of the neighborhood is the key factor to impact on the quality and efficiency of algorithm as for the neighborhood search algorithm TS. Therefore, we first give two assumptions about the neighborhood structure as follows to ensure the convergence of TS algorithm.

Assumption 1: The neighborhood structure is supposed to be symmetrical. That is, $\forall s_i,s_j\in S, s_i\in N(s_j)\Leftrightarrow s_j\in N(s_i), i,j=0,\ldots,k$;

Assumption 2: On the point view of the graph theory, the graph G_N is supposed to be strongly connected. Namely, there must be a path from s_i to s_j for any $s_i,s_j\in S$, where $i,j=0,\ldots,k$.

Theorem 4 HPSO algorithm with the optimum keeping strategy is global asymptotic convergence when time is endless, namely the algorithm will converge to the optimal group.

Proof: Compared with the standard PSO velocity update equation, the equation has abandoned the previous velocity $\omega v_i(t)$ of particle i, which will make at least one particle of the particle swarm stop evolution of each generation due to its best history position. The optimal strategy algorithm is adopted in this hybrid algorithm. For convenience, the optimal individual reserved from each generation is saved in the left side of the population, but it does not participate in the evolutionary process. The state which contains the same optimal solution is arranged in order as same as which in the original state space, and the one which contains the different optimal solution is arranged in order according to the fitness value. Then new social collaboration transition probability matrix, self adapting transition probability matrix and competition transition probability matrix can be presented respectively as $P_S^*=diag(P_S,P_S,\ldots,P_S)$,$P_A^*=diag(P_A,P_A,\ldots,P_A)$, and $P_C^*=diag(P_C,P_C,\ldots, P_C)$. After the competition, we'll compare the optimal solution of the current population with the optimal solution reserved from the former generation, such an operation is presented by $U=(u_{ij})$. Set $Z^t=max\{f(pop_i^{t+1}),i=1,2,\ldots,N\}$ be the optimal fitness, then the transition probability from $Pop^t=[Z^{t-1},pop_1^t,pop_2^t,\ldots,pop_N^t]$ to $Pop^{t+1}=[Z^t,pop_1^{t+1},pop_2^{t+1},\ldots,pop_N^{t+1}]$ is presented as follows:

$$P(Pop^{t+1}=s_j \mid Pop^t=s_i)=\begin{cases}1 & \text{if } \max\{f(pop_i^t), i=1,2,\cdots,N\}=f(Z^t) \\ & \text{and } (pop_1^t,pop_2^t,\cdots,pop_N^t)=(pop_1^{t+1},pop_2^{t+1},\cdots,pop_N^{t+1}) \\ 0 & \text{otherwise}\end{cases} \quad (20)$$

Thus, there is unique element 1 in every line of the U, the others are 0. Meanwhile, U is lower triangular matrix considering that the individual or is replaced by better or remains unchanged. Therefore, U is noted as follows:

$$U=\begin{pmatrix} U_{11} & & & \\ U_{21} & U_{22} & & \\ \vdots & \vdots & \ddots & \\ U_{k1} & U_{k2} & \cdots & U_{kk} \end{pmatrix} \tag{21}$$

Where U_{ij} is $k\times k$ matrix, and U_{11} is unit matrix. That is to say that the transition probability matrix with one step of PSO algorithm is lower triangular matrix.

$$\begin{aligned} P^* &= P_C^* (P_S^* P_A^*)U \\ &= diag(P_C(P_SP_A), P_C(P_SP_A),\ldots, P_C(P_SP_A))U \\ &= \begin{pmatrix} P_C(P_SP_A)U_{11} & & & \\ P_C(P_SP_A)U_{21} & P_C(P_SP_A)U_{22} & & \\ \vdots & \vdots & \ddots & \\ P_C(P_SP_A)U_{k1} & P_C(P_SP_A)U_{k2} & \cdots & P_C(P_SP_A)U_{kk} \end{pmatrix} \\ &= \begin{pmatrix} C & 0 \\ R & T \end{pmatrix} \end{aligned} \tag{22}$$

Obviously, P^* is an irreducible stochastic matrix.

In theory, it has been proved that if the search space S of TS is limited, and neighborhood structure satisfies the above assumption 1 and assumption 2, TS algorithm will converge to optimal solutions inevitably. Then the transition probability matrix with one step of TS algorithm is irreducible stochastic matrix as well. Apparently, the transition probability matrix of HPSO algorithm $P=P_TP^*$ is irreducible stochastic matrix. This shows that the probability of individual staying in the non-global optimal solution tends to 0, therefore HPSO algorithm with the optimum keeping strategy will converge to the optimal group when time is endless. Namely, $\lim_{t\to\infty} P(Z_t\in S_{opt})=1$, where S_{opt} is the optimal solution set.

3.4 Experiments and Analysis

According to the above analysis, the global asymptotic convergence of HPSO algorithm can be guaranteed theoretically. However, the proof is based on perfect operation situations such as sufficiently large taboo list, infinite time and so on. Considering the reality of computer limitations and the limited time, we just take the convergence theory as the guidance in the specific computational experiments, some relaxations are made in accordance with the actual conditions on aspects of taboo length, search steps. Therefore, the solutions of some problems we obtained can just go nearly to rather than reach the optimal solution.