2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

A Chaotic Particle Swarm Optimization Algorithm for the Jobshop Scheduling Problem

YAN Ping
School of Economics and Management
Shenyang Aerospace University
Shenyang, China
0316yanping@126.com

JIAO Minghai
Computing Center
Northeastern University
Shenyang, China
mhjiao@cc.neu.edu.cn

***Abstract*—An improved Chaotic Particle Swarm Optimization (CPSO) algorithm for a jobshop scheduling problem, with minimization of makespan as the criterion, is proposed in this research. A real-valued encoding scheme based on a matrix representation is developed, which converts the continuous position value of particles in PSO to the processing order of job operation. A compound chaotic search strategy that integrates both Tent and Logistic chaotic search process is employed to the global best particle to enhance the local searching ability of PSO. In addition, a gaussian disturbance technology is embedded in the CPSO algorithm to improve the diversity of the particles in the swarm. The performance of CPSO is compared with the standard PSO algorithm on a benchmark instance of jobshop scheduling problems. The results show that the proposed CPSO algorithm has a superior performance to the PSO algorithm.**

Keywords*—*scheduling; particle swarm optimization; chaotic search

I. INTRODUCTION

Among all types of scheduling problems, job shop scheduling problem (JSSP) has important applications in different industries. This problem is known to be NP-hard in the strong sense [1]. Thus, it is unlikely to obtain the optimal schedule through polynomial time-bounded algorithms. Over the years there has been a great deal of research to develop efficient approaches for the problem. A new hybrid swarm intelligence algorithm to solve the job-shop scheduling problem is proposed by Tsung-Lieh Lin et al. [2]. Roshanaei et al. [3] develop new solution methodologies including mathematical modelling and a meta-heuristic for the flexible job shop scheduling problem. Mencia et al. [4] apply a hybrid search algorithm that interleaves best-first and depth-first search to the job shop scheduling problem. Yuan and Xu [5] integrate hybrid harmony search (HHS) and large neighborhood search (LNS) for the flexible job shop scheduling problem with makespan criterion. Among these solution methodologies of JSSP, intelligent optimization algorithms are relatively easy to implement and they could conveniently be adapted for different kinds of scheduling problems. This has made the research on them increasingly popular in the recent years. PSO is one of the latest metaheuristic methods in the literature. Based on the metaphor of social interaction and communication such as bird flocking and fish schooling, PSO was introduced to optimize both continuous and discrete problems. Recently, PSO algorithms were successfully applied to a wide range of applications such as aggregate production planning by Wang and Yeh [6], data envelopment analysis by Meng [7], trajectory planning problem of underactuated spacecrafts by Zhuang and Huang [8], and medical technology by Yang et al. [9]. However, the converging processes of the standard version of PSO algorithm tend to be too slow for practical-scale JSSPs. In this paper, some effort has been devoted to the modification and improvement of PSO algorithm that is applied to the JSSP.

The reminder of the paper is organized as follows: In the following section, details of the JSP are described. Then we give a brief introduction to PSO algorithm and its inspiration. The structure of the proposed PSO algorithm is presented in Section 3. A series of comparative experiments are conducted in Section 4 to evaluate the performance of the proposed algorithm. Finally, conclusions are given in Section 5.

II. PROBLEM DESCRIPTION

Scheduling for job shops is an important topic in production management. Generally for the job-shop scheduling problem there are a set of n jobs are waiting to be processed on a set of m machines where the processing of each job consists of m operations performed on these machines in a specified sequence. In this paper we consider the deterministic and static job-shop scheduling problem. Some basic assumptions are in the following:

1) All the jobs are available at time zero.
2) Each machine can process only one job at a time.
3) Each job can be processed by only one machine at a time.
4) Once started operations cannot be interrupted.
5) All the machines are continuously available throughout the production stage.
6) The transportation time to deliver relevant jobs between different machines is neglected.
7) The setup time for the machines to switch between different jobs is neglected.

The manufacturing procedure to be performed on any one machine is called an operation of the job. The operation of job i has to be performed on machine j with deterministic processing time t_{ij}. The goal for optimization is to minimize the makespan ($C_{\max}$). In other words, the scheduling objective considered in this paper is to determine the processing sequence of the

This work was supported in part by Humanistic social science fund of China's Ministry of education (NO.12YJC630312),Science and Technology Projects of Zhejiang Province under Grant #2011C03004.

operations on each machine such that the completion time of the last operation could be minimized, i.e.,

$$Minimize\ \ F = \underset{i\in\{1,\cdots,n\}}{Max}(C_i)$$

where C_i is the completion time of job i.

III. CPSO Algorithm For Job Shop Scheduling

A. Standard PSO

Particle swarm optimization (PSO), originally designed by Kennedy and Eberhart [10] in 1995, is inspired by observing the bird flocking or fish school. In the PSO algorithm, birds are called particles, each representing a potential solution. All particles have their position, velocity, and fitness values. To find the optimal solution, each particle adjusts its flying according to its own flying experience and its companions' flying experience. For n-dimensional search space, each particle in the swarm population has the following attributes: a current position represented as $X_i^k = [x_{i1}^k, x_{i2}^k, \cdots, x_{in}^k]$; a current velocity represented as $V_i^k = [v_{i1}^k, v_{i2}^k, \cdots, v_{in}^k]$; a current personal best position represented as $P_i^k = [p_{i1}^k, p_{i2}^k, \cdots, p_{in}^k]$. Let $G^k = [g_1^k, g_2^k, \cdots, g_n^k]$ denote the current global best position. During the evolution process of the swarm, the new position and velocity of each particle i are determined by the following equations:

$$v_{ij}^{k+1} = w^k v_{ij}^k + c_1 r_1 (p_{ij}^k - x_{ij}^k) + c_2 r_2 (g_j^k - x_{ij}^k) \quad (1)$$

$$x_{ij}^{k+1} = x_{ij}^k + v_{ij}^{k+1} \quad (2)$$

B. Coding Scheme and Initial Population Generation

In solving the jobshop scheduling by PSO, first task is to represent a solution of a problem as a particle. We utilize a priority-based representation where a particle is encoded as a matrix. Suppose n jobs are to be scheduled on m machines. An $n\times m$ matrix is given by a particle as follows:

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix},$$

where x_{ij} denotes the priority value of job i which is processed on machine j. Then, we use Giffler and Thompson's heuristic [11] to create a feasible schedule based on the priorities values of job operations for each particle. Below is a brief outline of the G&T algorithm for obtaining schedules. Let Φ contain the schedulable operation of each job and e_{ij} denote the earliest time at which the operation (i, j) can start.

Step 1. Let e_{ij} =0, for all operation (i, j) in Φ.

Step 2. Compute $f^* = \min_{(i,j)\in\Phi}\{e_{ij} + t_{ij}\}$ where t_{ij} is the processing time of operation (i, j). Let m^* denote the machine on which the minimum is achieved.

Step 3. Find out the conflict set Ψ of all operations (i, m^*) on machine m^* such that $e_{im^*} < f^*$.

Step 4. Select the operation r^* with the largest priority value from Ψ and schedule it. If more than one operation exists according to the priority values, tie is broken by a random choice.

Step 5. Delete the operation r^* from Φ, and put its immediate successor into Φ.

Step 6. Update e_{ij} in Φ and return to step 2 until all operation are scheduled.

An initial population is generated randomly where x_{ij} is drawn from [x_{ij}^{low}, x_{ij}^{up}]. x_{ij}^{low} and x_{ij}^{up} are the lower and upper bounds of x_{ij}, respectively. As for the range of particle velocity, we clip the range of particle velocities v_{ij} within $[-(x_{ij}^{up} - x_{ij}^{low}), x_{ij}^{up} - x_{ij}^{low}]$.

C. A Compound Chaotic Search Technology for the Global Best Particle

Chaos is a deterministic, random-like process found in non-linear, dynamical system, which is non-period, non-converging and bounded. The two main characteristics of the chaos method concentrate on randomicity and ergodicity that will strengthen the performance of the traditional PSO algorithm. In order to emphasize exploitation ability, a compound chaotic search for the global best particle is embedded in the proposed PSO algorithm. The chaotic search process integrates Tent and Logistic chaotic maps which are defined as equations (3) and (4) respectively.

$$x_{n+1} = 1 - 2|x_n|, \quad -1 < x_n \le 1 \quad (3)$$

$$x_{n+1} = \lambda x_n (1 - x_n), \quad 0 < x_n < 1, \ 0 < \lambda \le 4 \quad (4)$$

A compound chaotic map can be obtained by incorporating the above two maps as follows:

$$x_{n+1} = 1 - 2\lambda x_n (1 - x_n), \quad 0 < x_n \le 1 \quad (5)$$

Some good results have been shown in some applications when λ is set to 2 [12]. Let x_{ij}^{low} and x_{ij}^{up} be the lower and upper bounds of x_{ij} in a particle matrix respectively. The local search algorithm is described in the following:

Step 1. Set $k = 1$.

Step 2. Map matrix variables x_{ij}^k to chaotic variables cx_{ij}^k in terms of (6):

$$cx_{ij}^k = \frac{x_{ij}^k - x_{ij}^{low}}{x_{ij}^{up} - x_{ij}^{low}} \quad (6)$$

Step 3. Generate chaotic variables cx_{ij}^{k+1} based on the compound chaotic map (5) where λ is set to 2, i.e.:

$$cx_{ij}^{k+1} = 1 - 4cx_{ij}^k (1 - cx_{ij}^k), \quad 0 < cx_{ij}^k \le 1 \quad (7)$$

Step 4. Translate the chaotic variables cx_{ij}^{k+1} into matrix variables x_{ij}^{k+1}:

$$x_{ij}^{k+1} = x_{ij}^{low} + cx_{ij}^{k+1}(x_{ij}^{up} - x_{ij}^{low}) \quad (8)$$

Step 5. Decode the matrix representation of the particle into a feasible processing sequence of the operations on all machines.

Step 6. Compute the fitness f^{k+1} of the particle.

Step 7. Update the best position of the particle if a better fitness ($f^{k+1} < f^{k}$) is obtained by the chaotic search.

Step 8. Set $k = k + 1$. Go to step 2 until the maximum number of iteration for chaotic search is achieved.

D. Gaussian Disturbance Strategy

One of the major drawbacks of the PSO is its premature convergence, especially while handling problems with more local optima. Particles tend to be stagnant when their velocities are near to zero. In this paper, a gaussian disturbance strategy is introduced in the proposed PSO to solve the global optimization problem. When matrix variables x_{ij} of a particle are unchanged in μ consecutive iterations, the disturbance strategy will be applied to the particle in the swarm. Suppose $G^k = [g_{ij}^k]_{n \times m}$ be the current global best position in the kth iteration of CPSO. Formula (9) gives the gaussian disturbance process.

$$x_{ij}^k = \begin{cases} g_{ij}^k \times [1 + C \times N(0,1)], & 0 \le rnd \le 0.98 \\ r_{ij}^k, \quad r_{ij}^k \text{ is a random real number} & \\ \quad \text{and } r_{ij}^k \in [x_{ij}^{low}, x_{ij}^{up}], & \textit{otherwise} \end{cases} \tag{9}$$

where C is a predefined constant parameter. $N(0,1)$ represents the standard normal distribution where mean is set to 0 and variance is set to 1. For each matrix variable x_{ij} to be disturbed, a random value *rnd* is generated uniformly from [0, 1]. Whether a gaussian disturbance or a random immigrant technique will be applied in terms of the value of *rnd*.

E. The Framework of Proposed CPSO

Based on the above design, the procedure of the proposed CPSO algorithm for solving JSSP is summarized as follows.

Step 1. Generate randomly an initial population within the range of position and velocity values.

Step 2. Translate matrix representations of particles into feasible scheduling schemes.

Step 3. Evaluate all particle individuals in the population based on the fitness.

Step 4. Update the personal best particle Pbest as well as global best particle Gbest in the swam.

Step 5. Apply the gaussian disturbance strategy to each particle in the current population. Update the personal best position and the current global best position whenever a better fitness is obtained by the gaussian disturbance.

Step 6. Perform the chaotic local search algorithm on the current global best particle Gbest.

Step 7. Update the velocities and positions of particles according to equations (1) and (2).

Step 8. Go back to Step 2, unless the termination condition is met.

IV. COMPUTATIONAL EXPERIMENTS

A. Experiment Design

To test the performance of the proposed CPSO algorithm for solving JSSP, some comparative experiments are carried out. A benchmark instance of JSSP which is prefixed by LA01 and described by Lawrence [13] is employed in the experiments. There are 10 jobs to be processed on 5 machines in LA01 and the optimal objective function value of the test problem is 666 as reported in the literature for the makespan criterion. Table I and II list the processing machine sequence and processing time of each job on all the machines in details.

In the experiment, the parameters for the proposed CPSO algorithm are assigned as follow. The number of the population size of the particles is set to 30. Default values for the parameters c_1 and c_2 have been used: $c_1 = c_2 = 2$. The inertia weight w is set to decrease linearly from 0.9 to 0.4 during a run of the PSO. The time decreasing inertia weight allows the PSO to explore a large area at the start of the run, and to refine the search later by using a smaller inertia weight. To clip the range of positions and velocities for the encoding scheme of particles, we set x_{ij}^{up} (i=1, 2, …, n; j=1, 2, …, m) to be 4 while x_{ij}^{low} is set to −4. The maximum number of iteration for compound chaotic search is set to 10. The gaussian disturbance parameter C is set to 0.9. The termination criterion of the proposed CPSO is the maximum of 1000 iterations in population evolution.

TABLE I. PROCESSING MACHINE SEQUENCES OF JOBS

	M1	**M2**	**M3**	**M4**	**M5**
J1	2	1	5	4	3
J2	1	4	5	3	2
J3	4	5	2	3	1
J4	2	1	5	3	4
J5	1	4	3	2	5
J6	2	3	5	1	4
J7	4	5	2	3	1
J8	3	1	2	4	5
J9	4	2	5	1	3
J10	5	4	3	2	1

TABLE II. PROCESSING TIME OF JOBS

	M1	**M2**	**M3**	**M4**	**M5**
J1	21	53	95	55	34
J2	21	52	16	26	71
J3	39	98	42	31	12
J4	77	55	79	66	77
J5	83	34	64	19	37
J6	54	43	79	92	62
J7	69	77	87	87	93
J8	38	60	41	24	83
J9	17	49	25	44	98
J10	77	79	43	75	96

All of the compared algorithms are coded in Visual C++ and the experiments are executed on a Pentium PIV 3.0 GHz PC with 512 MB memory.

B. Computational Results

Firstly, some experiments are designed in order to find an optimal value for parameter μ in the gaussian disturbance strategy. The experiments of the JSSP optimization are done three times using the CPSO algorithm by adjusting the values of μ from 1 to 100. Three values including 1, 10 and 100 are tested for μ. The results of the experiments are presented in Figure 1. The results show that the performance of CPSO varies when μ set to different values. When μ is set to 100, CPSO is to be stagnant in local minima and does not converge to the global minima in the end. CPSO with μ=1 converges much faster than CPSO with μ=10 although both of them converge to the optimal makespan value. It seems that a smaller μ will achieve more improvement in solution than a larger one. Thus, we select μ=1 for CPSO.

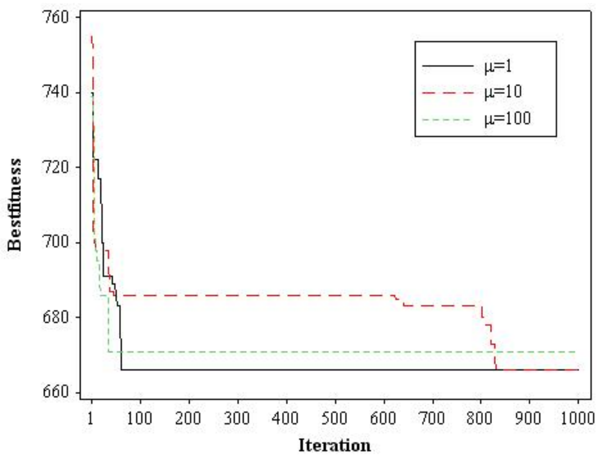

Fig. 1. The results of CPSO with different μ.

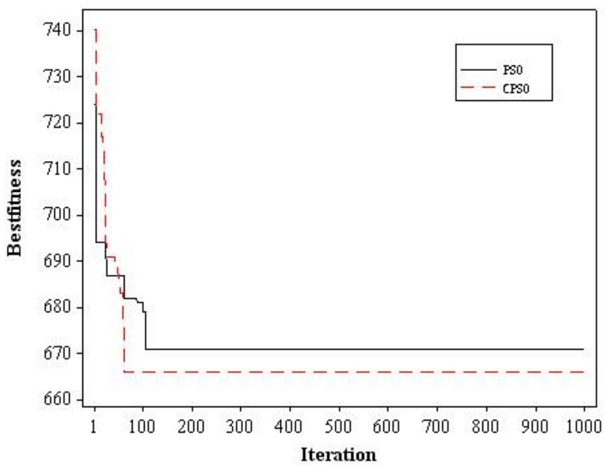

Fig. 2. The representative convergence curves for PSO and CPSO.

Then, we compare the experimental results of proposed CPSO with standard PSO. Table III summarizes the results of the experiments. From Table III it can be seen that the best and average solutions found by PSO are 671 and 672.6 respectively while the results of CPSO is much better than the original design. CPSO obtains the optimal solution 666 and its average solution is 668 which are much closer to the optimal solution. The convergence trends of PSO and CPSO are shown in Fig. 2. The proposed CPSO algorithm can reach the best solution in 60 iterations. Whereas the particles of PSO do not converge to global optima after 1000 iterations. The performance of CPSO can be attributed to the compound chaotic search and gaussian disturbance improvements which enhance search behavior of PSO and allow it to avoid local optima.

TABLE III. COMPUTATIONAL RESULTS OF STANDARD PSO AND CPSO

PSO			CPSO		
Best	Worst	Average	Best	Worst	Average
671	724	672.6	666	740	668

V. CONCLUSION

In this research, the idea of applying chaotic search principle in particle swarm optimization to the jobshop scheduling problems has been explored. By applying a special encoding mechanism, the continuous position values of particles in PSO are converted to the processing order of job operation. A compound chaotic local search strategy is designed to perform exploration for promising solutions within the entire region. The approach also incorporates a gaussian disturbance technique that is used to enhance diversity to the population. Simulation results and comparisons demonstrate the effectiveness of the proposed CPSO in terms of searching quality and robustness. The future work is to investigate the applications of the proposed method to some other kinds of combinatorial optimization problems.

REFERENCES

[1] J.K. Lenstra, A.H.G.R. Kan, and P. Brucker, “Complexity of machine scheduling problems,” *Anal. Discrete Math.* , vol. 7, no. 1, pp. 343–362, 1977.

[2] T. Lin, S. Horng, T. Kao, Y. Chen, R. Run, R. Chen, J. Lai, and I. Kuo, “An efficient job-shop scheduling algorithm based on particle swarm optimization,” *Expert Syst Appl.* , vol. 37, no. 3, pp. 2629-2636, 2010.

[3] V. Roshanaei, A. Azaba, and H. ElMaraghy, “Mathematical modelling and a meta-heuristic for flexible job shop scheduling,” *Int J Prod Res.*, vol. 51, no. 20, pp. 6247-6274, 2013.

[4] C. Mencia, M. R. Sierra, and R. Varela, “An efficient hybrid search algorithm for job shop scheduling with operators,” *Int J Prod Res.*, vol. 51, no. 17, pp. 5221-5237, 2013.

[5] Y. Yuan and H. Xu, “An integrated search heuristic for large-scale flexible job shop scheduling problems,” *Comput Oper Res.*, vol. 40, , no. 12, pp. 2864–2877, December 2013.

[6] S. Wang and M. Yeh, “A modified particle swarm optimization for aggregate production planning,” *Expert Syst Appl.* , vol. 41, pp. 3069–3077, 2014.

[7] M. Meng, “A hybrid particle swarm optimization algorithm for satisficing data envelopment analysis under fuzzy chance constraints,” *Expert Syst Appl.* , vol. 41, pp. 2074–2082, 2014.

[8] Y. Zhuang and H. Huang, “Time-optimal trajectory planning for under actuated spacecraft using a hybrid particle swarm optimization algorithm,” *Acta Astronaut.* , vol. 94, pp. 690–698, 2014.

[9] C. Yang, H. Chang, Y. Cheng, and L. Chuang, “Novel generating protective single nucleotide polymorphism barcode for breast cancer using particle swarm optimization,” *Cancer Epidemiology*, vol. 33, pp. 147–154, 2009.

[10] J. Kennedy and R. C. Eberhart, “Particle swarm optimization,” in *Conf. Rec. 1995 IEEE Int. Conf. on neural networks* , New York: Piscataway, pp. 1942–1948.

[11] J. Giffler and G. L. Thompson, “Algorithms for solving production scheduling problems,” *Oper. Res.*, vol. 8, pp. 487–503, July-August 1960.

[12] X. Yi and K. Yang, “Improvement and performance research of hybrid algorithm based on compound chaotic-artificial fish swarm,” *Computer Engineering & Science*, vol. 35, no. 8, pp. 89-95, 2013.

[13] S. Lawrence, “Resource constrained project scheduling: an experimental investigation of heuristic scheduling techniques (supplement),” in *Technical Report*, Graduate School of Industrial Administration, Carnegie Mellon University, Pittsburgh, 1984.