ISSN:2321-1156

International Journal of Innovative Research in Technology & Science(IJIRTS)

An APSO-based Decomposition Approach for Scheduling Parallel Machines with Eligibility Constraints

Sun Fudong, Chen Jiyun. Patent Examination Cooperation Center of Patent Office, SIPO Beijing, 100083, China

Abstract

In this paper, we consider identical parallel machine scheduling problem where every job is only allowed to be processed on a specified subset of machines. We develop an efficient APSO (Adaptive Particle Swarm Optimization) based decomposition approach to solve the above problem with the objective of minimizing total number of tardy jobs. In the proposed approach, the problem is firstly decomposed into a machine assignment subproblem and a job sequencing subproblem, then the job sequencing subproblem is solved optimally by a heuristic algorithm in polynomial time, and the machine assignment subproblem is solved by the proposed APSO algorithm, where the key parameters of APSO are adjusted in an adaptive manner dynamically based on the newly generated data during the search process of the algorithm itself. Also, the problem features are incorporated into the initialization and particle moving process to further enhance the performance of the algorithm. Numerical computations on random generated instances show that the proposed approach can obtain an excellent solution in an acceptable time.

Introduction

Many scheduling problems in practice can be regarded as parallel machine scheduling problem, such as scheduling bottleneck workcenters in wafer fabs, scheduling processors in a huge computer, and some nurse scheduling problems in a hospital. Parallel machine scheduling problem is a kind of difficult combinatorial optimization problems, for which many efficient algorithms have been presented [1-4]. However, the situations where jobs can only be assigned to a specified subset of machines are more common in the practical manufacturing environments. Such scheduling problems can be generally described as follows [5]: We are given a set of n jobs $\{J_1, J_2, \cdots, J_n\}$ and a set of m machines $\{M_1, M_2, \cdots, M_m\}$. Each job J_i has a processing time p_i, a duedate d_i and a set of machines $\mu_i \in \mu$ to which it can be assigned. The goal is to assign each job to one machine in μ_i and determine the processing sequence of jobs for each machine such that some scheduling criterions, e.g. the makespan, the total number of tardy jobs, are optimized. Aiming at solving this kind of scheduling problems, Yixun Lin [6] proposed several polynomial algorithms for solving parallel machine scheduling problem with unit-length jobs and different scheduling objectives, Oguzhan Alagoz [7] studied the rescheduling problem in parallel machine environments under processing set restrictions and presented an optimizing algorithm for minimizing the stability measure subject to the constraint that the efficiency measure is at its minimum level, and Liu Min et al. has proposed several efficient genetic algorithms [8-11] for solving parallel machine scheduling problems with processing set restrictions or special constraints.

Among the parallel machine scheduling problems, the problem with the objective of minimizing the total number of tardy jobs, which can be denoted as $Pm/\mu_i/\sum U_i$ using the 3-field notation [5], is one of the problems frequently encountered in the practical intensive manufacturing environments. However, there are little research reports focusing on solving this type of problem except recent work by Yin and Hao [9,11], and Hao [5]. In their approaches, Yin [9] presented a genetic algorithm (EOXGA) with an EOX (Extended Order Crossover) operator for generating feasible schedules automatically in the process of crossover operation, while Hao [5,11] developed a decomposition based evolutionary programming algorithm (DEPA) and chaotic particle swarm optimization algorithm (CPSO) where a polynomial algorithm are incorporated to solve the single machine scheduling problems optimally. DEPA and CPSO are more efficient than EOXA since the solution space is reduced greatly while keeping its performance by introducing a polynomial algorithm. However, as the problem scale is getting larger, the convergence speed and the performance of DEPA

and CPSO are not very satisfactory. In this paper, based on the above framework, we developed an APSO (Adaptive Particle Swarm Optimization) based decomposition approach to solve the problem $Pm/\mu_i/\sum U_i$ efficiently.

Particle swarm optimization (PSO), which was first introduced by Eberhart and Kennedy [13], has been successfully applied to solve continuous and discrete optimization problems [14-15,20]. Compared with general population-based algorithms, such as genetic algorithm, tabu search, ant-colony algorithm, PSO has a higher convergence speed, but is easier to get trapped into local optimum and the performance of traditional PSO mainly depends on the parameters. Researchers have presented many methods by incorporating new problem-specific search mechanisms to enhance its global optimization performance, including chaotic PSO [5,16-18], hybrid approaches with genetic algorithm [21], etc.

In this paper, we develop an efficient APSO based hybrid approach to solve the problem $Pm/\mu_i/\sum U_i$. In the proposed approach, the problem is naturally decomposed into a machine assignment subproblem and a job sequencing subproblem, then the job-sequencing subproblem is solved optimally by a heuristic algorithm with in polynomial time O(*mnlogn*), and the machine-selection subproblem is solved using the proposed APSO algorithm, where the key parameters, including velocities and directions of moving particles, of APSO are adjusted in an adaptive manner dynamically based on the newly generated data during the search process of the algorithm itself. By adaptive tuning, the population is inclined to search in more prominent area. Also, the problem features are incorporated into the initialization and particle moving process to further enhance the performance of the algorithm.

Numerical computations on random generated instances and production data from a wafer fab show that the proposed approach can obtain an excellent solution in an acceptable time.

This paper is organized as follows: In Section 2, we formulate the machine assignment subproblem and the job sequencing subproblem according to literature. Then, in Section 3 we propose APSO for the machine assignment subproblem and describe the polynomial time algorithm for the job sequencing subproblem. In Section 4 we compare the proposed algorithm with the algorithms of [5,11]. Finally, we give conclusions for this algorithm in Section 5.

Problem Decomposition

The problem studied in this paper $Pm/\mu_i/\sum U_i$ can be naturally decomposed into two subproblems: machine-assignment subproblem and job-sequencing subproblem. in the machine-assignment subproblem, each job needs to be assigned to a machine while in the job-sequencing subproblem, jobs assigned to the same machine need to be sequenced, which can be formulated as below.

(1) Machine assignment subproblem

We are given a set of n jobs $\{J_1, J_2, \cdots, J_n\}$ and a set of m machines $\{M_1, M_2, \cdots, M_m\}$. Each job J_i has a processing time p_i, a duedate d_i and a set of machines $\mu_i \in \mu$ to which job J_i can be assigned. Given a job sequencing algorithm for each machine, the goal is this subproblem to assign job J_i to one of the machines in μ_i such that the total number of tardy jobs is minimized.

(2) Job sequencing subproblem

We are given a set of n jobs $\{J_1, J_2, \cdots, J_n\}$ and a set of m machines $\{M_1, M_2, \cdots, M_m\}$. Each job J_i has a processing time p_i, a duedate d_i and a predetermined processing machine. The goal is to sequence all the jobs for each machine such that the total number of tardy jobs is minimized.

The algorithm

1. Heuristic algorithm for solving the job sequencing subproblem

In our approach, the job sequencing subproblem is solved optimally in polynomial time based on the heuristic algorithm [5,19]. The heuristic algorithm is illustrated in detail as below.

(1) Initialization: let $i=1$, $j=1$, S_j be the set of jobs to be processed on machine M_j, $n_j = |S_j|$ be the cardinality of set S_j, T_j be the set of tardy jobs in set S_j, $T_j = \varnothing\,(j=1,2,\cdots,m)$, $Q_j = S_j / T_j$, $t=0$.

(2) sort the jobs in S_j in increasing order according to

duedate d_i. The sorted job sequence in S_j can be denoted as $d_{[1]} \le d_{[2]} \le \cdots \le d_{[n_j]}$ without loss of generosity.

(3) If $t + p_{[i]} \le d_{[i]}$, then $Q_j = Q_j \bigcup \{J_{[i]}\}$, $t = t + p_{[i]}$, go to step (5).

(4) If $t + p_{[i]} > d_{[i]}$, then we consider two cases:

a) If $p_{[i]} < p_{[k]}$ (assume $J_{[k]}$ be the job with the longest processing time in set Q_j. If $Q_j = \varnothing$, let $p_{[k]} = 0$), then let $Q_j = Q_j \setminus \{J_{[k]}\}$, $Q_j = Q_j \bigcup \{J_{[i]}\}$, $t = t - p_{[k]} + p_{[i]}$, and go to step (5).

b) If $p_{[i]} \ge p_{[k]}$, go to step (5).

(5) If $i < n_j$, then $i = i + 1$, go to step (3).

(6) If $j < m$, then $i = 1$, j=j+1, $t = 0$, go to step (2). If $j \ge m$, then stop.

The time complexity of the above heuristic algorithm for solving $1\|\sum U_i$ (from step (2) to step (5)) is $O(n \log n)$. Since there are m independent single machine scheduling problems, the time complexity of the above heuristic algorithm for solving the job-sequencing subproblem is $O(mn \log n)$.

Since the job sequencing subproblem can be solved optimally by the above heuristic algorithm, the search space can be reduced greatly. However, the machine assignment subproblem remains difficult, especially for large instances from industrial practice. Assume the average value of μ_i is m/2, then the total number of solutions of the machine assignment subproblem is $(m/2)^n$.

2. APSO for machine assignment subproblem

Particle swarm optimization is a population based global optimization technology, which has been applied in many area[20-21]. Generally, the traditional PSO algorithm includes the following key steps: 1) solution representation; 2) swarm initialization; 3) particle moving mechanism and 4) velocity updating mechanism.

The key steps of the proposed APSO are introduced as follows.

(1) Solution representation

In the proposed APSO, since the job sequencing subproblem is solved by a polynomial time, each particle in APSO only consists of the machine assignment information. Therefore, particle X_k can be defined as:

$$X_k = [x_{k,1}, x_{k,2}, \cdots, x_{k,n}], k = 1, 2, \cdots, N$$

where $x_{k,i}$ ($x_{k,i} \in \mu_i$) represents the machine on which job J_i is to be processed, and N is the total number of particles in each iteration.

In the decoding process for each particle, we firstly determine the jobs to be processed on each machine based on the above representation, and then obtain the final schedule by the heuristic algorithm in 3.1.

(2) Swarm initialization

In our approach, In order to improve the performance and diversity of the initial population, we propose a problem-specific method to generate the initial swarm.

In this method, we first classify all of the jobs into three classes in average according to the slack time of each job, that is, jobs with low slack time belong to class A, jobs with middle slack time belong to class B, and the remaining jobs belong to class C. the numbers of jobs in class A, B and C should be almost equal. Then, we present a load-balance based machine assignment heuristic (LBH), which can be formulated as follows.

Procedure LBH

Step 1: For each machine M_j, $j = 1, 2, \cdots, m$, calculate the sum of processing time of jobs belonging to class A, B and C separately which are capable to be processed on machine M_j, denoted as $P(j, A), P(j, B)$ and $P(j, C)$.

Step 2: Sort the jobs in a random order.

Step 3: For each job in class A, assign it to the machine with the least $P(j, A)$ value in its processing set. After one job is assigned to a machine, recalculate the machine load $P(j, A), P(j, B)$ and $P(j, C)$ for each machine.

Step 4: For each job in class B, assign it to the machine with the least $P(j, A) + P(j, B)$ value in its processing

set. After one job is assigned to a machine, recalculate the machine load $P(j,A), P(j,B)$ and $P(j,C)$ for each machine.

Step 5: For each job in class C, assign it to the machine with the least $P(j,A)+P(j,B)+P(j,C)$ value in its processing set. After one job is assigned to a machine, recalculate the machine load $P(j,A), P(j,B)$ and $P(j,C)$ for each machine.

The jobs with different slack time can be balanced to different machines through the above machine assignment heuristic procedure.

Besides, in the proposed algorithm, the particle velocity $V_k = [v_{k,1}, v_{k,2}, \cdots, v_{k,n}]$ is an *n*-dimensional vector for each particle, of which $v_{k,i}$ $(i=1,2,\cdots,n)$ represents the velocity of the particle's *n*-th element.

(3) Update personal best and global best particle

We evaluate each particle by its corresponding objective value (minimizing the total number of tardy jobs) which is obtained by the heuristic algorithm for solving the job-sequencing subproblem. After that, update the personal-best X_i^{pbest} for each particle and the global-best X^{gbest}.

(4) Adaptive particle moving mechanism

Since the solution representation in our approach is a discrete manner and the particle moving mechanism in traditional PSO is designed to move the particle to its personal-best (X_i^{pbest}) and global-best (X^{gbest}) according to its velocity in each iteration, we develop a new discrete particle moving mechanism combined with problem features and historical data of the algorithm in order that the particle can be moved to the personal-best or global-best particle adaptively. Assume without generosity that we need to move particle $X_k = [x_{k,1}, x_{k,2}, \cdots, x_{k,n}]$ to its personal-best particle $X^{pbest} = [x_1^p, x_2^p, \cdots, x_n^p]$, and the velocity of particle X_k is $V_k = [v_{k,1}, v_{k,2}, \cdots, v_{k,n}]$, then the proposed adaptive particle moving mechanism can be described as follows.

Procedure APMM:

Step 1: Let $i = 1$, representing the index of jobs.

Step 2: Generate *a* random real number *r* in $(0,1)$. If *r* < $v_{k,i}$, then let $x_{k,i} = x_i^p$, and randomly select a job assigned to machine x_i^p, and reassign it to another machine in its processing set.

Step 3: If $r \geq v_{k,i}$, $i = i+1$. If $i = n+1$, then stop. Else go to step 2.

There is a big difference between PSO and genetic algorithm (GA), that each solution need to move to its personal best and global best and thus generating a new solution (in a different place), while in GA each new solution is generated by crossover and mutation operators. Therefore, PSO is easy to get trapped into a local optimum. So in order to improve the global optimization performance of PSO, we add a new procedure, in which each particle can move to an unknown place with a small probability.

(5) Adaptive velocity updating mechanism

We propose an adaptive tuning method for velocities of each particle, in which the key algorithm parameters of APSO, including velocities of moving particles, are adjusted in an adaptive manner dynamically based on the newly generated data during the search process of the algorithm itself. The historical data of the search process of the algorithm reflects a lot of useful information of particles, which can be used to guide the velocities of particles. In our approach, we mainly explore the information of the total sum of tardiness of all jobs assigned for each machine. Considering that if the total sum of tardiness of the current machine is comparatively small, it seems unnecessary to assign jobs to other machines, and vice versa, we update each element of particle's velocity vector V_k at iteration t by:

$$v_{k,i}(t+1) = w^t \cdot v_{k,i}(t) \cdot \left(\frac{T_{t,i,tn}}{T_{t,avg}} + \lambda_1 \frac{T_{t-1,i,tn}}{T_{t-1,avg}} + \lambda_2 \frac{T_{t-2,i,tn}}{T_{t-2,avg}} \right) \Big/ 3 + c \times r$$

where $v_{k,i}(t)$ is the *i*-th element of velocity of particle V_k at current iteration t, $T_{t,i,tn}$ is the total tardiness of machine x_i at iteration t, $T_{t,avg}$ is the average value of total tardiness among all machines at iteration t, c is a control parameter, , λ_1, λ_2 is the impact factor of historical tardiness data, r is a random real number ranging in $(-1,1)$. It can be seen that when $T_{i,tn}$ is less, the velocity $v_{k+1,i}$ is to decrease with a larger probability, and vice versa. Thus, the machine assignment of jobs on the machine with larger number of tardy jobs can be adjusted faster. Besides, the inertia weight is

updated by the equation $w^{k+1} = w^k \times \alpha$ where α is the decrement factor, which is used to control the convergence speed of the algorithm.

It can be seen that the exploration of the proposed APSO is enhanced by the incorporation of problem specific particle moving mechanism and velocity updating mechanism.

Numerical Computations

1. Experiment design

We use the randomly generated parallel machine scheduling problem with different scale to test the performance of the proposed APSO. When generating scheduling problem, we randomly select a subset from all of the machines as μ_i. All of the jobs are released at 0. The processing time and the duedate are generated with the following distribution:

$$p_i \sim U(1,10)$$

$$d_i \sim U\left(1, \beta \cdot n \cdot p_{avg} / m\right)$$

where p_{avg} is the sum of processing time of all the jobs and β is duedate tightness. In this paper we set $\beta = 1$.

We compare the proposed APSO algorithm with the algorithms presented in [5,11], in which the CPSO and the DEPA are proposed for solving the same problem studied in this paper.

The parameters of the proposed APSO are set as follows: the population is $N = 100$, the iteration number $K = 200$, the control parameter $c = 0.1$, $\alpha = 0.99$, the initial value of inertia weight $w^1 = 1$, the impact weight $\lambda_1 = 0.5, \lambda_2 = 0.2$. In CPSO and DEPA, the population is 100 and the iteration number is 200, other parameters are set according to the paper [5,11].

2. Numerical results

Under the given parameters configuration above, Table 1 lists the average best objective value (total number of tardy jobs, denoted as N_{tardy}) of 10 independent runs. Figure 1 show the converging performance of the above three algorithms.

Table 1. Average results of 10 runs

$n \times m$	DEPA		CPSO		APSO	
	N_{tardy}	Time (s)	N_{tardy}	Time (s)	N_{tardy}	Time (s)
30×10	11.4	4	9.6	5	9.0	5
100×10	13.1	8	12.2	9	11.7	8
500×10	14.3	39	12.7	43	13.4	44
800×20	36.8	68	29.9	73	28.3	74
1500×20	58.0	141	49.4	143	48.5	143
2000×50	102.1	181	96.8	174	94.2	175

From Table 1, it can be seen that the results of the proposed APSO combined with problem features and CPSO are better than DEPA on almost all of the problems, and APSO is better than DEPA on the problems with large scale. It is concluded that the proposed APSO is more effective on the studied kind of problems, especially on the problems with large scale. Besides, the running times of the CPSO and APSO are very close.

Acknowledgments

The authors are thankful to IJIRTS Journal for the support to develop this document.

References

[1] Gursel, E. B. Suerand, “Minimizing the number of tardy jobs in identical machine scheduling”, Computers and Industrial Engineering, 25(4), 1993, pp. 243-246.

[2] Liu M, Wu C, “Scheduling algorithm based on evolutionary computing in identical parallel machine production line”, Robotics and Computer-Integrated Manufacturing, 19(5), 2003, pp. 401-407.

[3] H. G. Kahlbacher, T. C. E. Cheng, “Parallel machine scheduling to minimize costs for earliness and number of tardy jobs”, Discrete Applied Mathematics, 45(2), 1993, pp. 139-164.

[4] G.A. Suer, F. Pico, and A. Santiago, "Identical machine scheduling to minimize the number of tardy jobs when lot-splitting is allowed", Computers and Industrial Engineering, 33(1), 1997, pp. 277-280.

[5] Hao Jinghua, Liu Min, Wu Cheng. CPSO-Based Hybrid Approach for Scheduling Parallel Machines with Processing Set Restrictions. WRI Global Congress on Intelligent Systems, 2009.

[6] Yixun Lin, Wenhua Li, “Scheduling unit-length jobs with machine eligibility restrictions”, *European Journal of Operational Research*, 174(2), 2006, pp. 1325–1328

[7] Oguzhan, Alagoz, “Rescheduling of identical parallel machines under machine eligibility constraints”, Euro-

pean Journal of Operational Research, 149(3), 2003, pp. 523–532.
[8] Liu Min, Wu Cheng, and Yin Wen-jun, "Solving identical parallel machine production line scheduling problem with special procedure constraint by genetic algorithm", ACTA AUTOMATICA SINICA, 27(3), 2001, pp. 381-386.
[9] Yin, Wen-jun, Liu Min, Wu Cheng, "A new genetic algorithm for parallel machine scheduling with process constraint", ACTA ELECTRONICA SINICA, 29(11), 2001, pp. 1482-1485.
[10] Liu Min, Hao Jinghua, Wu Cheng, "A genetic algorithm for parallel machine scheduling problems with procedure constraint and its applications", Chinese Journal of Electronics, 15(3), 2006, pp. 463-466.
[11] Hao Jinghua, et al., "Decomposition based optimization algorithm for complex parallel machine scheduling problems", Control Engineering of China, 12(6), 2005, pp. 520-522.
[12] R.L. Graham, E.L. Lawler, J.K. Lenstra, and A.H.G. Rinnooy Kan, "Optimization and approximation in deterministic sequencing and scheduling: a survey", Annals of Discrete Mathematics. 5(1), 1979, pp. 287–326.
[13] Kennedy, J, Eberhart, R, "Particle swarm optimization", Proceeding of the 1995 IEEE International Conference on Neural Network, Perth, 1995, pp. 1942-1948.
[14] Abido, M A, "Optimal power flow using particle swarm optimization", International Journal of Electrical Power and Energy Systems, 24(7), 2002, pp. 563-571.
[15] Hong Zhang, Xiaodong Li, Heng Li, and Fulai Huang, "Particle swarm optimization based schemes for resource-constrained project scheduling", Automation in Construction, 14(3), 2005, pp. 393-404.
[16] Bo Liu et. al., "Improved particle swarm optimization combined with chaos", Chaos, Solitons and Fractals, 25(5), 2005, pp. 1261–1271.
[17] Alatas B et al., "Chaos embedded particle swarm optimization algorithms", Chaos, Solitons &Fractals, doi:10.1016/j.chaos.2007.09.063, 2008.
[18] Hong W-C, "Chaotic particle swarm optimization algorithm in a support vector regression", Energy Conversion and Management, doi:10.1016/j.enconman. 2008.08.031, 2008.
[19] Peter Brucker, *Scheduling Algorithms, Second Edition.* Springer Press, 1998.
[20] Yanchao Wang, Jian Lv, Li Zhu, and Yanming Ma. "Crystal structure prediction via particle-swarm optimization", *Physical Review Letters.* 2010,82(09).
[21] MH Moradi, M Abedini. "A combination of genetic algorithm and particle swarm optimization for optimal DG location and sizing in distribution systems", *International Journal of Electrical Power & Energy Systems*, 34(1), 2012, pp:66-74.

Biographies

SUN FUDONG received the B.S. degree in Civil Engineering from the China University of Mining Technology, Xuzhou, JiangSu, in 2002, the M.S. degree from the same university, in 2008. Currently, she is an examiner Patent Examination Cooperation Center of Patent Office. She may be reached at sunfudong@126.com.