Energy-Efficient Job Shop Scheduling Problem Using an Improved Particle Swarm algorithm

Gu Wenbin, Li Yuxin, Wang Yi

College of Mechanical and Electrical Engineering, Hohai University, Changzhou, China
guwenbing@yeah.net; 2817984422@qq.com; 20021592@hhu.edu.cn

***Abstract*—To deal with the fastly increasing energy consumption and energy costs in the manufacturing process, manufacturing companies are forced to search some methods to reduce energy cost without affecting the yield of their products or sacrificing quality. In this paper, an extended job shop scheduling problem, where machines can work at different speeds with different energy consumption, is modified. A serial of analytical functions are developed to denote the relationship between Energy-efficiency and Makespan. And a improved particle swarm algorithm, inspired by the hormone mechanism, is developed to solve the energy-efficient job shop scheduling problem. In term of practical application aspect, the result from computational experiments indicate that durations of processing periods and processing speed of machines have great effect on energy-efficiency scheduling solution, and our algorithm could solve instances with a good solution quality.**

Keywords—* *energy consumption, extended Job shop scheduling problem, IPSO, makespan

I. INTRODUCTION

With the development of industry, on the one hand, human life is improving constantly, but on the other hand, the natural resources are freely burned for the high demand of energy. Form several years, environmental impacts and energy consumption play an important role on policy making of the enterprise [1]. According to the relevant research data, manufacturing activities should be responsible for about 90% of industrial energy consumption, and generated about 84% of industrial CO2 emissions[2] . This also becomes one of the major issue for governments, who care of the ecological impact of the industrial sector upon the planet and on the society[3]. Therefore, it is significant for enterprise to reduce energy consumption and environmental impacts in their manufacturing activities.

A job-shop problem(JSP) represents a particular case of scheduling problems where there are some specific resources or machines which have to be used to carry out some tasks[4]. In actual manufacturing processing, job-shop scheduling is one of the essential criteria that influence cost, tardiness, and quality. In addition, the difference in job-shop scheduling solution will also lead to the different result in resource consumption and emissions[5][6].

Recently, there has been growing interest in energy efficient job shop scheduling problem due to the fast rising energy consumption of production activities. Most of the literatures about energy efficient has focused on two aspects[6]: (1) the local energy consumption (machine or product), and (2) the global energy consumption (job shop floor or factory level). For the local energy consumption, it is important to develop the energy-efficient machines and the design approaches embodied the product energy frameworks. On the other hand, for the global energy consumption, research is focused on production planning and scheduling,facility operation, and supply chain designing. In addition, some scientific experiments affirm that it is important for energy consumption to set up the process parameters, especially cutting speed[7].

In this work, we mainly focus on a classical job-shop scheduling problem, which was proposed as $J||C_{max}$ problem[8], with controllable processing speed for minimizing energy consumption within several constrains. And we develop an improved Particle Swarm Optimization Algorithm(IPSO), which is inspired by the hormone mechanism, to solve the energy efficient job-shop scheduling problem. Finally, the results of the experiments show that the IPSO can solve the energy efficient scheduling model effectively.

II. PROBLEM DESCRIPTION

Scheduling is concerned with allocating limited resources to tasks to optimize certain objective functions. Formally, job shop scheduling problem(JSSP) is a classical operations research problem which has been considered as a NP-hard problem due to its computational complexity since the 1950s[9]. In traditional JSSP, there are a series of m machines M=(m_1, m_2, m_3, ... , m_n) and a series of n jobs J=(j_1, j_2, j_3, ... , j_n), and each job j_i has been separated into a finite set of processing sequence which can be operated on a different machine and in specified processing times, in a given job-dependent order. A typical objective of this problem is to minimize the makespan(the maximum of job completion time) or the other cost function.

But as an extensional JSSP considering the energy efficient as the main objective, we can find that each job j_i is composed of a sequence of processing task P_t with corresponding energy consumption e_{Po} used by the corresponding processing speed in the operation machine. By assuming that the processing speed in a machine which executes a task can be adjusted, the extended JSSP can be described based on the classic JSSP. There are a set of n job and a set of m machine. Each job consists of several processing task P= (p_1, p_2, p_3, ..., p_k) which can be operated by several machines in a given cutting speeds. In addition, a series of cutting speeds S=(v_1, v_2, v_3, ..., v_s) is presented in the job shop. The scheduling objective is to determine the processing sequence with given cutting speeds to make the scheduling problem achieve the optimal makespan with minimized energy consumption. Moreover, the processing sequence should meet the following constraint condition:

- All the machine are available at beginning time;
- Each machine can only execute one task at each moment;
- It cannot be interrupted midway when the task is processed on the machine;
- Procedure constraints exist within each job, but not among different jobs;
- There are the same priority among the different jobs.

III. Modeling a Mathematical Formulation for the Extend JSSP

In previous research, a predetermined production plan is barely considered for an energy-efficient job shop scheduling. Hence, the energy consumption is integrated in our model. There are four steps to calculate the energy consumption. The first step is calculate the energy consumption (E_S) for startup, affected by cutting speed of machine tool. And a mixed integer programming mathematical model for energy consumption (E_p) of the processing operation is proposed in the second step. In the third step, the idle energy consumption (E_u), which is also affected by rotating speed of machine tool, is calculated. Finally a total energy consumption in the job shop scheduling problem is obtained, and it will be considered in the extend job shop scheduling problem.

A. Calculate the total energy consumption for a job-shop floor

As mentioned above, the energy consumption of each machine can be decomposed into three parts.

When a job comes to a machine, the energy is consumed to activate machine component and to make it sure that the machine is available for the processing operation. The energy consumption (E_S) for startup can be expressed as

$$\mathrm{E_s} = \sum_{m=1}^{M} f_m(n) \tag{1}$$

where $f_m(n)$ is the input energy required to activated the mth machine, n is its spindle speed.

When a machine is at the processing operation stage, the main energy consumption is composed of material removal energy and basic energy for maintaining the normal operation of machine components. The energy consumption (E_p) for processing operation stage can be expressed as

$$\mathrm{E}_p = \sum_{m=1}^{M} (P_{um} + (1+\alpha_1)P_{cm} + \alpha_2 P_{cm}^2) t_{cm} \tag{2}$$

Where P_{um} is the idle power of the mth machine, P_{cm} is the cutting power of the mth machine, a_1 and a_2 is the additional load loss coefficients, and t_{cm} is the processing time on the mth machine.

When the machine is running at the idle mode stage, the machine component will still require energy to implement some operations (such as changing cutting tools, clamping or loosing the workpiece, and etc.). The energy consumption (E_u) for processing operation stage can be expressed as

$$\mathrm{E}_u = \sum_{j=1}^{P} \sum_{m=1}^{M} P_{ujm}((C_{(j+1)m} - C_{jm}) - t_{jm}) \tag{3}$$

Where P_{ujm} is the idle power for job j on the mth machine, which is affected by the spindle speed. C_{jm} is the finishing time of the job j processed on the mth machine, and t_{jm} is the processing time of the job j processed on the mth machine.

Therefore, the total energy consumption can be expressed as

$$E_{total} = E_s + E_p + E_u \tag{4}$$

B. Mathematical model

Solving the extended JSSP considering the energy efficient requires us schedule the process sequence with the machine speeds for relevant operations on each machine. A solution to this extend JSSP model consist of two parts, where $\Pi = (\pi_1, \ \ldots, \ \pi_m)$ represents the process sequence, and S=[v_{ik}] represents the machine speed settings for each process. The following is a mixed integer programming model for the extended JSSP.

$$\begin{cases} \min f1 = \min(\max_{m\in M}(\max_{i\in J}(c_{im}))), c_{im} \geq 0 \\ \min f2 = \min E_{total} = E_s + E_p + E_u \end{cases} \tag{5}$$

s.t.

$$c_{im} - t_{im} + \delta(1 - a_{ihm}) \geq c_{ih}, a_{ihm} = 0,1 \tag{6}$$

$$c_{jm} - c_{im} + \delta(1 - a_{ihm}) \geq t_{jm}, x_{ijm} = 0,1 \tag{7}$$

$$\sum\nolimits_{v\in S} X_{\pi jmv} = 1, \pi \in \Pi; j \in J; m \in M. \tag{8}$$

Where δ is a positive number; c_{ik} denotes finishing time of job i on the mth machine; t_{im} is the operation time of job i on the mth machine. a_{ihm} is an integer variable that has two possible values:0 or 1. It is equal to 1 if job i is processed on the hth machine before on the mth machine, otherwise it is equal to 0. x_{ijm} is an integer variable that has two possible values: 0 or 1. It is equal to 1 if job i is processed on the mth machine before job j, otherwise it is equal to 0. $X_{\pi jmv}$ is an integer variable that has two possible values:0 or 1. It is equal to 1 if operation π of job j is required to be processed on the mth machine with cutting speed v, otherwise it is equal to 0.

Eq.(5) is a multi-object function for the extend job shop scheduling problem. Constraint(6) represents job sequence restriction, and ensures that the operation sequence corresponds to the predetermined order. Constraint(7) represents machine restriction, namely, each machine can process only one job at a time. Constraint (8) represents the speed constraint, and ensures that each operation of one job is processed with one given speed on one machine tool.

IV. An IAPSO for the Extended JSSP

The particle swarm optimization(PSO) method is a metaheuristic population-based optimization technique, developed by Kenney and Eberhart, that mimics the movement of a flock of flying birds. As opposed to its well-developed counterparts, there are a lot of assoiated problems that should be studied, such as the information sharing mechanism among particles, the cooperation and competition between the local optimization and the global optimization, and the balance between explorations and exploitations, etc. But it is still successful to be applied to the engineering problem by the advantage of simplicity. Moreover, scientists found that PSO had a very strong optimization ability to search the optimal solutions for those multi-objective problem. Therefore, inspired by the hormone modulation mechanism in the biological body, an IAPSO is proposed to solve the extended JSSP for better searching efficiency and quality.

A. Encoding scheme and initial swarm

One of the key issues in applying PSO sucessfully to JSSP is how to find a suitable mapping between problem solution and particles. There are two kinds of PSO encoding methods for JSSP, direct encording method (job-based encoding, operation-based encoding, finishing-time-based encoding, and random key-based encoding) and indirect encoding method (priority rule-based encoding, rule-based encoding, machine-based encoding, etc).

In this article, the operation-based encoding method is adopted to encode a schedule, and we set up a matrix X(k) with n rows and s columns, as shown in the following formula:

$$\mathrm{X(k)}=\begin{bmatrix} x_{11}(k) & x_{12}(k) & \ldots & x_{1s}(k) \\ x_{21}(k) & x_{22}(k) & \ldots & x_{2s}(k) \\ \ldots & \ldots & x_{jt}(k) & \ldots \\ x_{n1}(k) & x_{n2}(k) & \ldots & x_{ns}(k) \end{bmatrix} \quad (9)$$

Where $x_{jt}(k)$ represents the absolute position of job j at the PSO evolution iteration k for each operation t.

B. Velocity update with the variable HF inspired by hormone mechanism

In the normal PSO, the position update formula is decided by tracing the anterior position and two “extreme points” which are the present best solution and the historical best solution. When the PSO runs, each particle is individual, and they can perceive each other and interact with each other. But the evolutionary rules among the individual particles of the normal PSO are so simple which cause the limitations of poor convergence and premature. To improve these drawbacks, an adaptive variable HF, which is inspired by the hormone modulation mechanism, is devised to strengthen the information among particles. By referring the Hill function, the adaptive variable HF is designed as follows.

$$HF_i = (V_{\max} - V_{\min}) \times \frac{T^q}{T^q + k^q} \quad (10)$$

Where $V_{\max}$ is the upper bound of single particle flying velocity in the PSO, and $V_{\min}$ is the lower bound of single particle flying velocity in the PSO. T is a threshold value, and k is the current iteration number, and q is the Hill coefficient of the PSO. Using this novel variable HF, the update formula of velocity, shown in Formula (11), can adjust the its own value to avoid premature convergence.

$$\begin{aligned} v_{id}(k+1) = \omega_0 \cdot v_{id}(k) + HF + c_1 \cdot r_1 \cdot (p_{id}(k) - x_{id}(k)) \\ + c_2 \cdot r_2 \cdot (p_{gd}(k) - x_{id}(k)) \end{aligned} \quad (11)$$

where $v_{id}(k)$ represents the velocity of the dth dimensional space for particle i at iteration k. $x_{id}(k)$ represents the position of the dth dimensional space for particle i at iteration k. c_1 represents the cognition learning coefficient. c_2 represents social learning coefficient. $p_{id}(k)$ represents the best solution of the dth dimensional space for particle i at iteration k. $p_{gd}(k)$ represents the best solution of the dth dimensional space for all the particles at iteration k.

V. Experiments and Results

To illustrate the effectiveness and performance of the proposed mathematical model and optimization algorithm (IPSO), some experiments from the benchmarks are calculated. All test instances can be described as a set $J=(j_1, j_2, j_3, ..., j_n)$ of j_n jobs with a range of processing time p and a set $M=(m_1, m_2, m_3, ..., m_t)$ of m_t machines with three types of cutting speeds (such as, low cutting speed, medium cutting speed and full cutting speed). For each problem from the benchmarks, 15 test instances are randomly produced, and the Matlab program of IPSO algorithm was run to simulate the instances with a 3.20 GHz Intel Pentium (R) PC.

Set coefficient c_1=2, social learning coefficient c_2=2, inertia weight ω_0=0.3, $\omega_{\max}$=1.2, $\omega_{\min}$=0.2, Threshold coefficient T=12, swarm size 100, maximum iteration K=300. In order to highlight the performance of the proposed energy efficient JSSP model, we compared the results of the energy consumption with the same makespan between the traditional JSSP and the proposed extended JSSP. The optimization results of the instances are shown in Table I, where E_{JSSP} denotes the energy consumption of the JSSP, E_{EJSSP} denotes the energy consumption of the proposed extended JSSP, and ΔE_{SA} denotes the average energy saving ratio (percentage) of each problem size.

TABLE I. THE OPTIMIZATION RESULTS OF THE TEST INSTANCES

Problem size (n×m)	Time range (*p*)	Makespan (*t*)	E_{JSSP}	E_{EJSSP}	ΔE_{SA} (%)
3×5	[1, 50]	192	770.2	709.3	7.91
3×7	[1,100]	532.3	2041.7	1889.6	7.45
5×5	[1, 50]	166.3	731.9	672.5	8.12
5×10	[1,100]	610.3	2806.2	2589.6	7.72
7×5	[1, 50]	169.7	655.3	608	7.22
7×10	[1,100]	625.9	2941.1	2670.1	9.217
10×10	[1,200]	941.2	10320.3	9875.6	4.31
15×15	[1,200]	1555.3	23151.2	22504.3	2.79

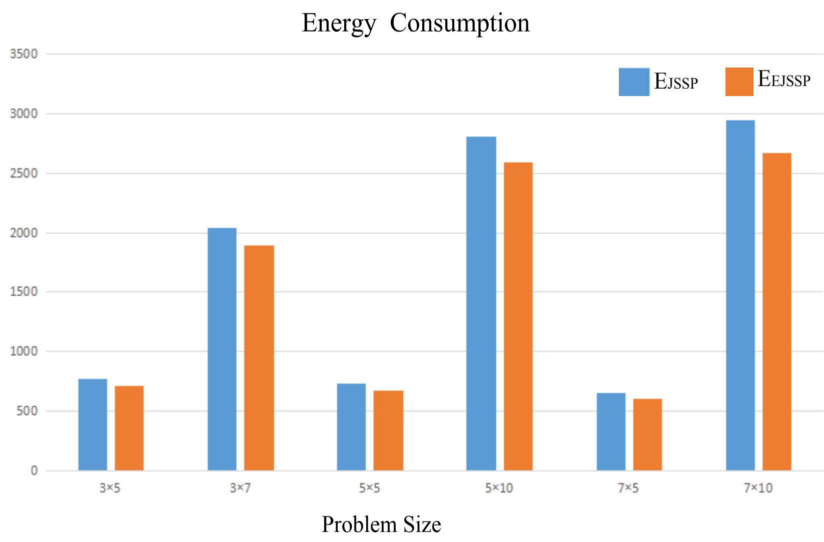

Fig.1 The comparison of the energy consumption between JSSP and extended JSSP

It can be clearly observed that the proposed model can save energy consumption with scheduling the cutting speeds of the jobs on the machines. And we can also obtain that the range of the ΔE_{SA} varies from 2% to 10% at the same makespan.

The comparison histogram of the average energy consumption between JSSP and the extended JSSP is shown in Fig.1. From the Fig.1, we can find that the total energy consumption in the job shop level can be reduced by adjusting the cutting speeds of the operations. In addition, if we divide the instances into two parts: small-size instances(n<10) and large-size instances(n≧10), it can be observed that the energy efficient JSSP model has more effect on the small-size instances than the larger-size instance by adjusting the cutting speeds. Through analyzing the reason, we find that the load balancing between machine utilization and task allocation will be more effective in large-size instances than in small-size instances.

VI. CONCLUSIONAND FUTURE WORK

In this paper, we have investigated an extended JSSP with consideration of the total energy consumption. To solve the extended JSSP effectively, we devised an improved particle swarm optimization algorithm (IPSO) with the variable *HF*, which has the ability to guarantee the particles searching optimal solution in the feasible solution space efficiently. The proposed energy-efficient job shop scheduling model and IAPSO are applicable in a wide rang of manufacturing systems. The computational results show that the performance of the proposed model and algorithm can reduce the energy consumption effectively by adjusting the cutting speeds of the operations on the machines. In the future, the energy efficient JSSP model will be improved to the large-size job shop scheduling problem, and extended to the dynamic job shop scheduling problem.

ACKNOWLEDGEMENTS

This paper is supported by National Science Foundation of China (No.51505126), Changzhou Sci&Tech Program (CJ20159052) and China Scholarship Council Fund (G27882737). The authors would like to thank the referees for their helpful comments and suggestions.

REFERENCES

[1] T. Gutowski, C. Murphy, D. Allen, D. Bauer, et al. Environmentally benign manufacturing: observations from Japan, Europe and the United States. Journal of Cleaner Production, 2005, 13, 1–17.

[2] M. Schipper. Energy-related carbon dioxide emissions in U.S. manufacturing[R]. Washington D.C.: Energy Information Administration, 2006.

[3] S. Kemmoe, D. Lamy, N. Tchernev. A job-shop with an energy threshold issue considering operations with consumption peaks, The 15th IFAC Symposlum on Information Control Problems in Manufacturing, 2015, Ottawa, Canada, 823-828

[4] J.R. Duflou, J.W. Sutherland, D. Dornfeld, et al. Towards energy and resource efficient manufacturing: A processes and systems approach[J]. CIRP Annals - Manufacturing Technology, 2012, 61(2): 587-609.

[5] K. Fang, N. Uhana, F. Zhao, & J.W. Sutherland. A new approach to scheduling in manufacturing for power consumption and carbon footprint reduction. Journal of Manufacturing Systems, 2011, 30, 234-240.

[6] M. Dai, D.B. Tang, A. Giret, et al. Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm. Robotics and Computer-Integrated Manufacturing. 2013, 29(5), 418-429.

[7] N. Diaz, E. Redelsheimer, D. Dornfeld. Energy consumption characterization and reduction strategies for milling machine tool use[C], Proceedings of the 18th CIRP International Conference on Life Cycle Engineering: Glocalized Solutions for Sustainability in Manufacturing, Braunschweig, Germany, May 2–4, 2011: 263–267.

[8] J.bBlazewicz, W. Cellary, R. Slowinski, and J. Weglarz. Scheduling under resource constraints-deterministic models. Annals of Operations Research. 1986, 7: 13–56.

[9] D.P. Bunde. Power-aware scheduling for makespan and flow[C], Proceedings of the Eighteenth Annual ACM Symposium on Parallelism in Algorithms and Architectures, New York, 2006, pp 190–196. ACM