Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 61 (2015) 521 – 526

Procedia Computer Science

Complex Adaptive Systems, Publication 5
Cihan H. Dagli, Editor in Chief
Conference Organized by Missouri University of Science and Technology
2015-San Jose, CA

A Bayesian Optimization-based Evolutionary Algorithm for Flexible Job Shop Scheduling

Lu Sun[a], Lin Lin[a,b,*], Yan Wang[a] ,Mitsuo Gen[b,c] and Hiroshi Kawakami[d]

[a]*Dalian University of Technology, 116620, China*
[b]*Fuzzy Logic Systems Institute, 820-0067, Japan*
[c]*Tokyo University of Science, 113-8656, Japan*
[d]*Kyoto University, 606-8306, Japan*

Abstract

Flexible Job-shop Scheduling Problem (fJSP) is a typical and important scheduling problem in Flexible Manufacturing System (FMS). The fJSP is an extended version of Job-shop Scheduling (JSP) that is NP hard problem. Due to it according with the real production system, we adopt a hybrid evolutionary computation algorithm to solve the fJSP problems. Among them, the Bayesian Optimization Algorithm (BOA) is introduced to the characteristics of scheduling and uncertainty characteristics of the time in the fJSP. On this basis, we propose a distributed evolutionary algorithm and parameter adaptive mechanism. Finally, through experiments, we conclude that the proposed hybrid evolutionary algorithm based on BOA with grouping mechanism get better solution than original algorithm and improve robustness of algorithm. Meanwhile, the paper also have objective perspective, that is we can group the data different from each other, make the whole population into sub-populations, and then make the experiment separately on different and parallel machines in distributed environment, so that not only optimizes the best solution, but also enhance the efficiency and shortened the time.

Peer-review under responsibility of scientific committee of Missouri University of Science and Technology

Keywords: Bayesian Optimization Algorithm; Flexible Job-shop Scheduling Problem; Evolutionary Algorithms

1. Introduction

In today's society, the scheduling models often need to adapt to different application requirements, and designing the scheduling model one by one for each customer's demands is clearly unrealistic, so the highly flexible scheduling is particularly important. In this paper, we introduce a fJSP (flexible Job-shop Scheduling Problem) model to simulate the automatic scheduling for a flexible manufacturing model. The fJSP as an extension of Job-shop Scheduling Problem (JSP) is a typical combinatorial optimization problem and it is a NP-hard problem under

Peer-review under responsibility of scientific committee of Missouri University of Science and Technology
doi:10.1016/j.procs.2015.09.207

the constraint of priorities and resources (Lawler, 1993; Gen & Cheng, 2000; Gen, Gao & Lin, 2009). Meanwhile, it has the characteristics of resources non-uniqueness: for an operation, it can choose the machine in available set to complete. For a traditional JSP (tJSP), researchers often propose the following assumption: for an operation, machine and processing time are fixed. With the development, multi-purpose equipment is replacing traditional equipment in order to increase efficiency and decrease cost in manufacturing. Therefore, considering the fJSP research is closer to the actual production mode.

Recently, Xing provided an effective integration between Ant Colony Optimization (ACO) models and applies the existing knowledge to guide the current heuristic searching, and indicates that the proposed algorithm is effective (Xing, *et al,* 2010). Zhang proposed algorithm with Global Selection (GS) and Local Selection (LS) routines designed to generate high-quality initial population in the initialization stage for minimizing makespan (Zhang, 2011). Nouri, *et al* proposed a combined genetic algorithm and tabu search with a scheduler agent applying a Neighborhood-based Genetic Algorithm (NGA) to guide the research in promising regions (Nouri, *et al*, 2015). Chang and Liu proposed a Hybrid Genetic Algorithm (HGA) for solving the distributed and flexible job-shop scheduling problem (DfJSP) and demonstrated the effectiveness of the algorithm through the experiment (Chang & Liu, 2015). All best solutions by above algorithms are generated by repeated iteration, but not learning from a learning criterion and existing research only considered the processing time and processing sequences, however, does not take process machine resources of multiple factors into account. So it is lack of related research for flexibility and the effectiveness analysis of the scheduling; design of optimization method, especially for the craft flexibility route and machine optional of flexible scheduling problem.

In this paper, we propose a hybrid evolutionary algorithm with Bayesian Network (BN) to solve S-fJSP that aims to minimize the makespan of processing time. We choose the Particle Swarm Optimization (PSO) proposed by (Kennedy & Eberhart, 1995) as the basic algorithm and group the chromosomes randomly at first. Generate the candidate network by PSO and choose the best BN that is closest to the real structure with the maximum likelihood function. Then, group the chromosomes again according to the BN structure. Do same operations mentioned above every m generations. Among the generations, adapt the parameters of PSO in the process adaptively to ensure to get better solutions. Section 2 introduces formulating process of fJSP model; Section 3 proposes the hybrid evolutionary algorithm with BN; Section 4 provides the detailed numerical experiment and result; finally, Section 5 gives the conclusion of the whole paper.

2. Modeling of fJSP

For the fJSP model, different operations of different tasks are processed on different machines, the processing time of each operation is fixed, and the machine for one operation is only one at one time. In fact, the fJSP model is an extension of JSP, so we can describe the formulation process as follows:

(1) Machine allocation: choose a machine for each operation from the available machine set.

(2) Operation sequence: all necessary operations for completing the jobs which satisfying the precedence constraints.

(3) Sequencing robustness: a best solution is not sensitive to data.

In the fJSP, each job i consists of n_i operations (O_{i1},O_{i2},...,O_{in_i}). For each operation O_{ik}, processing machine must be from the machine set A_{ik}. The operations must be completed in sequence for one job.

The symbols used in the S-fJSP are defined as follows:

Indices:

i, h: job index, $i, h = 1,2.,\ldots,n$

j: machine index, $j = 1,2,...,m$

k, g: operation index, $k, g = 1,2,\ldots, n_i$

Parameters:

n: total number of jobs

m: total number of machines

n_i: total number of operations of job i

t_{ikj}: processing time of kth operation of job i

Decision variables:

c_{ik}: completed time of O_{ik}
x_{ikj}: machine j is selected for O_{ik}

The objective function is to minimize the makespan and the mathematical programming model of the fJSP formulated as follows:

$$\min \quad C_M = \max_{1\le i\le n}\{c_{in_i}\} \tag{1}$$

$$\text{s. t.} \quad c_{ik} - c_{i(k-1)} \ge t_{ikj} x_{ikj},\ k = 2,...,n_i, \forall i, j \tag{2}$$

$$[(c_{hg} - c_{ik} - t_{hgj})x_{hgj} \ge 0] \vee [(c_{ik} - c_{hg} - t_{ikj})x_{ikj} \ge 0],\ \forall i, j, g, h \tag{3}$$

$$\sum_{x_{ikj}\in A_{ik}} x_{ikj} = 1,\ \forall i, k, j \tag{4}$$

$$c_{ik} \ge 0,\ \forall i, k \tag{5}$$

$$x_{ikj} \in \{0,1\},\ \forall i, k, j \tag{6}$$

The constraints (2) - (3) represent the operating sequence constraint. The constraint (4) guarantees machine allocation that for each operation can only process on one machine from machine set at one time. The constraints (5) - (6) are nonnegative or 0 – 1 binary variable which are restrictions on decision variables, respectively.

3. Hybrid Evolutionary Algorithm

3.1 PSO-based HEA Procedure

Genetic algorithm (GA) and PSO are typical Evolutionary Algorithm (EA). PSO uses real-number decoding method while GA uses 0-1 decoding method, this makes PSO has larger solution space and larger probability to get a best solution. Here is the proposed HEA based on PSO with several distributed sub-populations and BN in Fig. 1.

```
procedure: Hybrid Evolution Algorithm
input: problem data, parameters
output: best solution of S-fJSP
begin
    t ← 0;
    initialize population p(t) by random solution candidates satisfying the constraints;
    get sub-p(t) by randomly grouping;
    g(t) ← 0;
    while (not meeting termination condition)
        evaluate p(t) by PSO with grouping mechanism and parameters adaptive;
        get the training data set Data from each sub-p(t);
        g(t) ← g(t+1);
        keep best solution of population gbest(t) and personal best pbest(t);
        if (g(t) meets condition)
            select the best BN structure by Data;
            readjust the grouping by BN structure for each sub-p(t);
            g(t) ← 0;
    end
    output best solution of S-fJSP gbest(t)
end;
```

Fig.1. Pseudo-code for Hybrid Evolution Algorithm

When compared to GA, PSO has stronger searching ability for avoiding getting into local optimal solutions (Kennedy, 2010). So, the proposed algorithm chooses PSO as the basic algorithm, then starts with giving the value randomly to each gene of the chromosomes to form a number of initial chromosome population. Grouping the population randomly by given number of sub-populations which means how many genes one group has. Then, we use the cooperative optimization framework to evaluate each sub-population until the generation reaches a given

number in advance. After that, we use PSO with two parts which one part presents input sequence of data and the other part provides the values to confirm the network structure. Finally we find out the best network, which is closest to the real structure, by using the maximum likelihood evaluation function. Now, change the grouping strategy with this network structure and restart the process, which mentioned above, until the generation reaches the max generation given in advance.

PSO uses the following formulas to update velocity and position in each generation t, respectively. The initial values set randomly. The ω presents inertia weight, $rand_1$ and $rand_2$ are random value within [0,1]. The bigger ω presents affecting by chromosome's own value; the smaller ω presents affecting by population's social factor. The adaptation of parameters used in PSO is adapted according to the methods studied in the paper (Yang, Tang & Yao, 2008).

$$v_i(t+1) = \omega v_i(t) + c_1 rand_1[p_{pbest}(t) - x_i(t)] + c_2 rand_2[p_{gbest}(t) - x_i(t)] \quad (7)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (8)$$

3.2 Bayesian Network Structure Learning

Bayesian Network is a typical probabilistic graphical model and it has two parts: nodes and edges which nodes present the variables and the edges between variables present the causal relationship, and it was proved that it is a directed acyclic graph (DAG) (Friedman, 1997). A simple BN is shown in Fig.2 (b), it presents the network has 3 variables, x_1 depends on x_2, x_3 depends on x_1 and x_2. Due to the operations of S-fJSP have incidence relation between each other, so the algorithm proposes to use BN to find the incidence relation and group the operations according to the network structure. Because we have reason to believe that grouping by BN can get better solutions than grouping randomly. However, the structure learning of BN is a NP hard problem (Chickering, 1994), so we choose a simple structure learning method.

Initialization a population with two parts: variable sequence and variable value by PSO for generate candidate network structures for each one group. We give an example with 3 variables one group in figure 2. We can get variable sequence: x_2-> x_1-> x_3, then get the BN structure by compare the values between variables. If the value of x_i is bigger than the value of x_j, add the edge from x_i to x_j, otherwise, take no actions. Each generation can get one network structure, after m generation, we can m candidate networks.

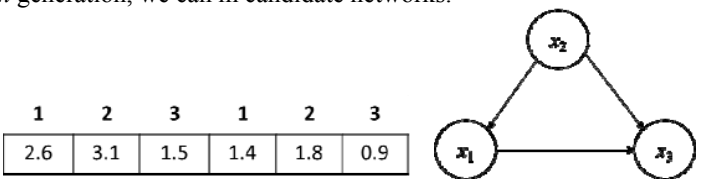
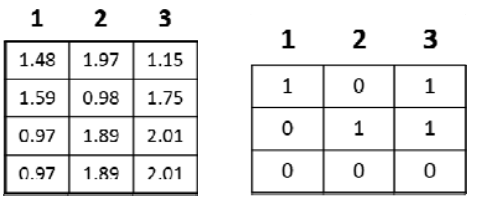


Fig. 2. (a) chromosome for learning BN structure; (b) BN structure example

3.3 Getting Training Data from the changing of Chromosomes



1	2	3
1.48	1.97	1.15
1.59	0.98	1.75
0.97	1.89	2.01
0.97	1.89	2.01

1	2	3
1	0	1
0	1	1
0	0	0

Fig.3. (a) chromosome of solution of S-fJSP; (b) dataset to evaluate the candidate networks

Now, we record down the change of each gene of solution chromosomes between the current generation and last generation. If the gene becomes bigger, then records 1, becomes smaller or keeps invariant, records 0. Then we can get the network structure evaluation training data set, shown in Figure 3. After evaluating the network structure, we

can find a best network with maximum likelihood score. Then, we readjust the grouping that put the discrete point into the other groups randomly.

4. Numerical Experiment and Result

In order to verify the effectiveness of the proposed algorithm in this paper, we do experiment under certain environment. Compare the hybrid cooperative algorithm with Bayesian grouping (CCBhEA) with a classic genetic algorithm (GA), a binary genetic algorithm (Binary GA), particle swarm optimization (PSO), differential evolution (DE) algorithm, cooperative coevolution group (CCPSO), and particle swarm algorithm with adaptive grouping differential evolution algorithm (SaNSDE). The experiment repeats 30 times, get the mean value. Test machine is Intel(R) Core(TM) i3-2120 CPU @3.3GHZ, 4GB.

Table 1 Experimental parameters Settings

	5*5	10*10	15*15	20*20
Pop. size	10	10	50	100
Crossover prob.	0.5	0.5	0.5	0.5
Mutation prob.	0.5	0.5	0.5	0.5
End con.	Evolved Indiv=5000	Evolved Indiv=5000	Evolved Indiv=5000	Evolved Indiv=10000
group size	10	10	20	40
BayesChInter.	5	5	5	5
BayesPSOInter	500	500	1000	5000
BayesPSOPop	10	10	20	50

Table 2 Experimental results under uncertain

		GA	Binary GA	DE	PSO	SaNSDE	CCPSO	CCBhEA
	max	332.0	367.0	344.0	314.0	334.0	339.0	289.0
	min	252.0	236.0	276.0	245.0	269.0	212.0	157.0
5*5	mean	291.5	305.76	314.1	273.43	310.66	279.866	217.2
	variance	573.25	1068.44	318.75	831.64	203.2	708.0488	774.35
	meanTime	37960.5	101558.08	27964.17	17935.07	42886.177	39735.37	483662.26
	max	868.0	891.0	876.0	857.0	869.0	854.0	801.0
	min	689.0	706.0	755.0	742.0	746.0	694.0	569.0
10*10	mean	778.4	834.06	809.9	748.966	805.7	766.86	696.1
	variance	2028.10	1905.72	907.29	468.4322	1102.74	2105.64	3148.75
	meanTime	134732	634998.01	119094.75	68724.85	132903.60	184064.67	4417871.8
	max	1383.0	1481.0	1354.0	1406.0	1322.0	1388.0	1317.0
	min	1238.0	1227.0	1284.0	1209.0	1231.0	1074.0	979.0
15*15	mean	1301.5	1365.8	1316.06	1277.4	1238.166	1232.7	1132.63
	variance	1242.71	3612.49	304.46	1886.37	548.595	4389.41	5596.76
	meanTime	56829.7	1.36E7	2415844.5	1227041.4	2457802.63	5812630.7	8.99E7
	max	1932.0	1945.0.	1853.0	1820.0	1860.0	1902.0	1759.0
	min	1711.0	1869.2	1730.0	1784.0	1706.0	1677.0	1372.0
20*20	mean	1832.46	1869.2666	1816.96	1794.83	1778.533	1774.96	1555.06
	variance	2841.64	2370.02	705.09	216.672	1170.51	4211.4	6958.72
	meanTime	1.49E7	1.00E8	1.43E7	7375768.1	1.49E7	1.77E7	1.189E9

Table 2 shows the experimental results of 4 scales of problems under uncertain, we use a different color to label each the best of each attribute, we can find that, CCBhEA has better performance for different scales of problems. Meanwhile, we do experiments for different algorithms to get the convergence of different, the result is shown in Figure 2. The compared algorithm reference (Della *et.al*, 1995, Sinha *et.al*, 2003, Kenddy *et.al*, 1995, Li et.al, 2012, Yang *et.al*, 2008).

5. Conclusion

We proposed an effective hybrid evolutionary algorithm (HEA) to solve the Flexible Job-shop Scheduling Problem (fJSP) in which it is based on Particle Swarm Optimization (PSO) with real-number encoding as the basic algorithm to increase the search space and avoid getting into local optimum solutions. Then, we used Bayesian Network (BN) structure to find out the relationship between the variables and according to the relationship to regroup, at the same time, using parameter adaptive mechanism to dynamic adjust parameters of PSO, minimize the makespan of fJSP within a reasonable amount of calculating time. The proposed algorithm can get better solutions and increase the robustness. Meanwhile, in our future work, numerical experiments will be processed by group under the distributed environment for each sub-population of the different and parallel processing, so that not only optimize the optimal solution, but also shorten the time more efficiency.

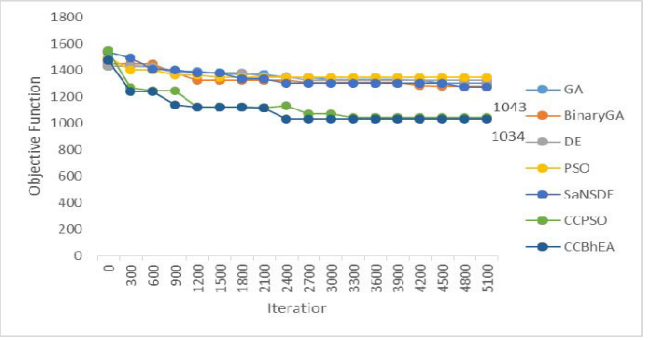


Figure 2 Algorithm convergence

Acknowledgments

This work is partly supported by the Fundamental Research Funds for the Central Universities No. DUT15QY10, and the Grant-in-Aid for Scientific Research (C) of Japan Society of Promotion of Science (JSPS) No. 15K00357.

References

[1] Lawler EL, Lenstra JK, Kan AHGR, Symoys D B. Sequencing and scheduling: Algorithms and complexity. *Handbooks in Operations Research and Management Science*, 1993, 4: 445-522.
[2] Gen M, Cheng R. *Genetic Algorithms and Engineering Optimization*. John Wiley & Sons, 2000.
[3] Gen, M., Gao, J., & Lin, L. Multistage-based genetic algorithm for flexible job-shop scheduling problem. In M. Gen, D. Green, O. Katai, B. McKay, A. Namatame, R. Sarker, et al. (Eds.), *Intelligent and evolutionary systems*, vol. 187, 2009:183–196, Springer, Berlin.
[4] Xing L N, Chen Y W, Wang P, Zhao QS, Xiong J. A knowledge-based ant colony optimization for flexible job shop scheduling problems. *Applied Soft Computing*, 2010, 10(3): 888-896.
[5] Zhang G, Gao L, Shi Y. An effective genetic algorithm for the flexible job-shop scheduling problem. *Expert Systems with Applications*, 2011, 38(4): 3563-3573.
[6] Nouri H E, Driss O B, Ghédira K. A Holonic Multiagent Model Based on a Combined Genetic Algorithm— Tabu Search for the Flexible Job Shop Scheduling Problem, *Springer International Publishing*, 2015: 43-54.
[7] Chang H.C., Liu T.K. Optimisation of distributed manufacturing flexible job shop scheduling by using hybrid genetic algorithms. *Journal of Intelligent Manufacturing*, 2015: 1-14.
[8] Kennedy J. Particle swarm optimization, *Encyclopaedia of Machine Learning. Springer US*, 2010: 760-766.
[9] Yang Z, Tang K, Yao X. Self-adaptive differential evolution with neighbourhood search, *Evolutionary Computation*, 2008:1110-1116.
[10] Friedman N, Geiger D, Goldszmidt M. Bayesian network classifiers. *Machine Learning*, 1997, 29(2-3): 131-163.
[11] Chickering D M, Geiger D, Heckerman D. Learning Bayesian networks is NP-hard. *Tech. Report* MSR-TR-94-17, *Microsoft Res.*, 1994.
[12] Della Croce F, Tadei R, Volta G. A genetic algorithm for the job shop problem. *Computers & Operations Research*, 1995, 22(1): 15-24.
[13] Sinha N, Chakrabarti R, Chattopadhyay P K. Evolutionary programming techniques for economic load dispatch. *IEEE Transactions on Evolutionary Computation*, 2003, 7(1): 83-94.
[14] Kenndy J, Eberhart R C. Particle swarm optimization, *Proceedings of IEEE Inter.Conference on Neural Networks*. 1995, 4: 1942-1948.
[15] Price K, Storn R M, Lampinen J A. Differential evolution: a practical approach to global optimization. *Springer Sci. & Bus. Media*, 2006.
[16] Li X, Yao X. Cooperatively coevolving particle swarms for large scale optimization., *IEEE Transactions on Evolutionary Computation,* 2012, 16(2): 210-224.
[17] Yang Z, Tang K, Yao X. Self-adaptive differential evolution with neighborhood search, *Congress on Evolutionary ComputationIEEE*. I2008: 1110-1116.