USE DATA MINING TO IMPROVE GENETIC ALGORITHM EFFICIENCY FOR A JOB SHOP SCHEDULING PROBLEM

S. Wesley Changchien & Ya-Tai Lin

Department of Information Management

Chaoyang University of Technology

168 GiFeng E. Rd., Wufeng, Taichung county, Taiwan, R.O.C. 413

PHONE: 886-4-23323000 ext. 4204 FAX: 886-4-23742337

EMAIL: {swc, s8914601}cyut.edu.tw

ABSTRACT

This paper proposes a new improved Genetic Algorithm (GA) by utilizing a Data Mining technique, and demonstrates how it is superior to traditional GA on a popular job shop scheduling problem. GA has long been widely applied to solve complex optimization problems in a good variety of areas. It has advantages of adaptive capability, efficient search, potential to avoid local optimum, etc. In recent literature, researchers have proposed a good number of new GAs by combining basic GA with other techniques, such as heuristic rules, simulated annealing, neural networks, fuzzy sets, and so on, in order to improve the efficiency for various optimization problems.

Data mining is a new evolving technology for knowledge extraction, classification, clustering, estimation, etc. The capability of finding frequent patterns in large data set is the key reason why it is integrated with GA in this research. Due to the fundamental concept of GA' s randomness during evolution, a traditional GA may become less efficient in search for optimum. By embedding the frequent schemata into the GA evolution process, the new improved GA could reduce the search time by preserving segments of good solutions without accidentally being lost due to random crossover or mutation. The proposed new GA was experimented on a popular 6x6 job shop scheduling problem. The results have shown its better efficiency than traditional GAs and potential for further research works.

1. INTRODUCTION

Nowadays scheduling problems require more efficient algorithms and integrated solutions, as the development of value chain, diversified product mix, and new technology leads to complex procurement, production, and distribution processes. In order to respond to customers' requirements for orders and to take full advantages of limited resources, a company must have high demand for advanced scheduling systems.

Popular production types consist of Single Machine, Parallel Machines, Flow-shop, Job-shop, and so on. Job-shop scheduling problem (JSP) is a very important issue. A job shop model, typical in manufacturing, can be described as a set of n jobs $\{J_j\}$, $1 \leqq j \leqq n$ composed of sequences of operations that need to be processed on a set of m machines $\{M_r\}$, $1 \leqq r \leqq m$. The operation of job J_j on machine M_r is called the operation O_{jr}, and the corresponding processing time is denoted T_{jr}. Operations of a job have to be executed in order and each operation can only be served on a machine and it cannot be interrupted. A schedule is a set of completion times for all the operations that conform to the above assumptions. The total time required to complete all the jobs is called the makespan. The problem is to find the best schedule that allocations of the operations to time intervals on the machines with minimum makespan. JSP is usually NP-hard, therefore it has long been researched.[14]

In the literature, scholars have developed different solutions to JSP, such as dispatching rules, heruistic algorithms, Branch and Bound and so on [2][11][13]. The use of dispatching rules is according to the properties of a job or an operation to decide its order in scheduling. The calculation is simple and quick, but its major disadvantage is that it only applies for specific scheduling objective or

data element; for example, Shortest Processing Time (SPT), Earliest Due Date (EDD), First Come First Served (FCFS), and Least Work Remaining (LWR) [9][11]. The method of heruistic algorithms usually tackles problems by dispatching rules under complex considerations. During the past decades, many probabilistic search methods are also applied to JSP extensively, such as simulated annealing, tabu search, beam search, and genetic algorithm [11][12][13]. Among these methods, Genetic algorithm (GA) is preferably applied to this problem because of its high efficiency of search. Meanwhile, scholars have continued their good progress on improvement of the representation, crossover, and mutation of GA. Compared with other methods, although GA can reach solutions efficiently through evolution by means of random numbers, the good experiences of developing solutions cannot be accumulated to accelerate the evolution. This motivates our research on the improvement of GA for JSP.

This research work makes use of the technology of Data Mining to find the schemata from GA's solutions in preliminary search then utilizes them to improve GA search efficiency.

2. LITERATURE REVIEW

2.1 Genetic Algorithm (GA)

John Holland [5] proposed GA in 1975. It is an adaptive method based on the theory of evolution for searching optimal solutions. GA improves the searching by comparing the fitness values one generation after another, and finally reaches the optimal or near optimal solution(s). It has the advantages of parallel search and capability of avoiding being restricted to local optimum, and it hence is a very popular method for complex optimization problems.

Within recent decades, a good number of genetic algorithms have been developed to solve job shop problems, and many of them focused on encoding issues. Cheng et al. [2][3] well compared and analyzed the major encoding methods including preference list based, job pair relation based, disjunctive graph based, job based, operation based, random key, and so on. Yamada and Nakano [14] submitted a method that combined local search and scheduling heuristics to design neighborhood search based crossover and Multi-Step Crossover Fusion (MSXF), and applied it to large-scale scheduling problems. Wang and Zheng [13] submitted a hybrid optimization strategy that integrated simulated annealing and genetic algorithm to solve JSP. It showed that these methods outperformed traditional methods for JSP.

Although GA is applicable for JSP, but GA may reach a different solution for each attempt and it is difficult to explain how solution was developed. Therefore, GA can be improved by accumulating the experiences of evolution in achieving good solutions. Recently, some scholars have combined the technology of data mining with GA, and improved the efficiency of GA for JSP by finding patterns in GA solutions. Koonce and Tsai [7] used an Attribute-oriented induction algorithm to classify the operations and induce corresponding priority rules from GA solutions to a JSP which then can be used later for other similar problems. According to the schemata theorem by Holland [4][5], the quantity of highly fit short-defining-length schemata will increase during the process of competition from generation to generation. The best fit schemata along with other survivors in a generation will produce better fit schemata in the next generation. If we can find those schemata from GA solutions with good fitness values and embed them into GA, the GA efficiency may be greatly improved.

2.2 Data Mining

Data mining is a technique which helps people to find information or knowledge form data for decision support. Data mining tasks include finding association rules, clustering, classification, estimation, etc. An association rule represented by $X \Rightarrow Y$, with confidence C in a transaction set D means that C is the percentage of transactions in D containing X and containing Y as well. It can explain cause and effect relationship between two sets of objects. For example, given all the GA chromosomes and its fitness values, we can find frequent schemata with high fitness values. A number of algorithms have been

developed to mine association rules from large data set such as Apriori, FP-tree, Rough Set Theory, etc [6]. Rough Set Theory is developed by Pawlak in 1982 [10]. It can analyze imprecise, inconsistent and incomplete data to extract hidden knowledge or underlying rules. It facilitates association rules mining but it requires repeated scan of the original data. Recently, a new efficient data structure, CIT (Class Inheritance Tree) [1], has been proposed using the concept of inheritance in object-oriented design for the cause equivalence class in Rough Set. It only scans the original data once to construct the CIT. In finding all association rules and calculating the confidences, CIT only search classes on related sub-trees. It hence significantly reduces the searching time and space in finding association rules.

3. A NEW PROPOSED GENETIC ALGORITHM

In this section, an improved GA by data mining is proposed. The efficiency of GA depends mainly on the encoding method of chromosomes and evolutionary operations. The elements of the encoded chromosomes are parameters of the fitness function. The characteristic of random search of GA may result in multiple optimal or good solutions. With a good design of encoding method and a data mining algorithm, good fitting hidden schemata in terms of association rules from preliminary GA solutions may be used to improve GA. Based on this motivation, this paper presents a new improved genetic algorithm by data mining technology. The algorithm is divided into 3 steps as show in Figure 1. Step 1 use a standard genetic algorithm to generated GA solutions. These encoded chromosomes then will be processed by a data mining algorithm to obtain schemata for accelerating the evolution of GA in Step 2. In Step 3, a new improved, GA with mined schemata as heuristic rules are applied to accelerate the search for optimal solution(s). The following sections describe the details of the three steps.

Step 1：GA for preliminary results

In Step1, a standard GA is applied to the problem to be solved for R generations of evolution. All the chromosomes and their corresponding fitness values are collected for mining purpose is Step 2.

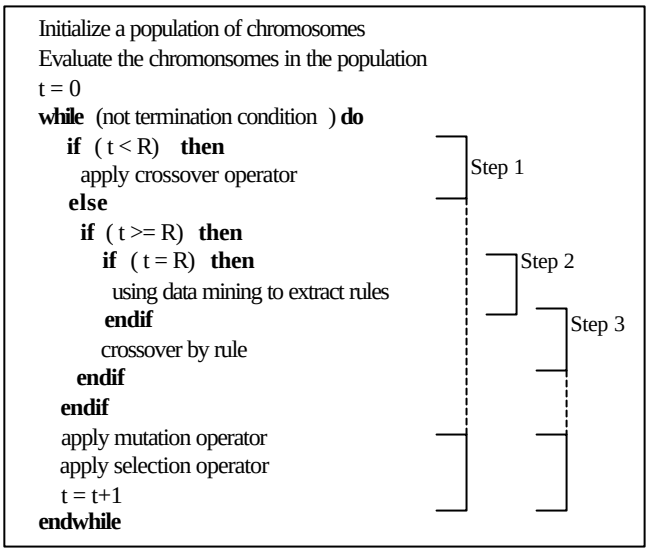

FIGURE 1 The algorithm of an improved GA by data mining.

The first task of applying GA to a complex optimization problem is to encode the problem parameters as a long digital code on which crossover and mutation operations can be performed. For a (m x n) job shop schedule problem, this research use 1,2, .., and (m x n) as operations identification numbers. For example, a 3×3 job shop problem is encoded as a 9 integer array (e.g., [7,4,1,5,6,2,8,3,9] where 1-3 denotes the operations of the first job, 4-6 the second job, and 7-9 the third job, respectively). Fitness function is defined as the total processing time for the ordered operations being scheduled and processed at the corresponding machine shops.

The order-based crossover method (OC2) [12] is adopted by this research. OC2 randomly selects two chromosomes as parent chromosomes and randomly chooses a number of exchange positions. From parent 1 chromosome, copy the chromosome at non-exchange positions to the child chromosome. The genes at the exchange positions are then inserted to the empty positions of the child chromosome in the order the genes are placed in parent 2 chromosome. **TABLE 1** is an example of order-based crossover. In parent 1, the genes at exchange positions (2,3,5,8) are (2,5,9,7). They are placed into the child chromosome in the order they are placed in parent 2 (i.e., 5→2→7→9).

Gene positions	1	2	3	4	5	6	7	8	9
Parent 1	1	2	5	6	9	8	3	7	4
Parent 2	5	7	2	1	7	8	9	3	4
Exchange positions		★	★		★			★	
Temporary Child	1			6		8	3		4
Order in Parent 2	5	7	2	1	7	8	9	3	4
New Child	1	5	2	1	7	8	3	9	4

TABLE 1 An example of order-based crossover.

Mutation is proceeded by changing the genes at the randomly selected positions. **TABLE 2** is an example of mutation. Positions 2 and 5 are selected for mutation. The gene at first mutation position (2) is allocated to the second mutation position (5). The genes at the selected mutation positions except the mutation end position are then shifted one position forward in the mutated gene.

Gene positions	1	2	3	4	5	6	7	8	9
Genes selected for Mutation	1	2	5	6	9	8	3	7	4
Mutation positions		★			★				
Mutated gene	1	5	6	9	2	8	3	7	4

TABLE 2 An example of mutation.

Selection strategy used in this paper is the popular Roulette wheel selection [4],[12] to avoid trapped in local optimum during evolution. The total area of the Roulette wheel is the summation of the fitness values of all chromosomes in the population. The area of each chromosome is its corresponding fitness value. Randomly generate random numbers and according to the values of the random numbers pick the corresponding genes as chromosomes of next generation of population. Basically the higher fitness value a gene has, the higher possibility it will be selected.

Step2 ： Data mining for schemata

This research combined Rough set theory and CIT structure to mine association rules from GA solutions in Step 1. According to Rough set theory, association rules may found in the form of X⇒Y with support C where X is the cause item set, Y the result item set and C the count of satisfied chromosomes (fitness values ≧ minimum fitness value). In JSP problem, X could contain different genes and Y refers the fitness values. All the chromosomes obtained in step 1 could be scanned to construct a CIT structure of cause equivalent classes. The rough set theory can be used to find all the lower, upper, and multidimensional association rules. For instance, an association rule could be "IF P_{32} = 28 and P_{33} = 35 THEN Fitness value ≧ 139 with count =100." It means that there are 100 chromosomes whose operation 28 is assigned to gene position 32, operation 35 assigned to gene position 33, and all the 100 chromosome's fitness values are lager than or equal to 139. Rules like this are attributed-oriented induction schemata which indicate some potential fragments of good schedules with higher fitness values. In general, the higher minimum fitness value and count number a rule has, the potentially valuable the rule is to help GA find the optimal solutions. In our research, a minimum count number is defined to ensure the high appearing frequency of extracted rules. The rules will be used for improving GA efficiency in next step.

Step3 ： Improved GA using data mining

In this step, mined rules are embedded into GA to increase the search efficiency. To avoid falling into local optimum, we randomly selected the association rules with higher numbers of count and applied to only a portion of crossover operations. A crossover method with rules is designed to ensure the genes under rule mask unchanged. **TABLE 3** gives an example of GA crossover with rules. In the example, the rule is that "IF P_2=3 and P_3=5 and P_4=6 THEN Fitness value=140 with count = 5." To maintain the rule mask, genes at positions 2 and 6 are exchanged.

Gene positions	1	2	3	4	5	6	7	8	9
Parent 1	1	3	5	6	2	7	4	8	9
Parent 2	2	5	4	6	1	7	9	3	8
Exchange positions		★	★		★	★		★	
Temp Child (Using OC2)	1	2	5	6	7	3	4	8	9
Rule Mask		3	5	6					
				Swap					
New Child	1	3	5	6	7	2	4	8	9

TABLE 3 An example of GA crossover with rules.

4. EXPERIMENT RESULTS FOR A JOB SHOP SCHEDULING PROBLEM

To illustrate our proposed algorithm, we experimented on a 6×6 JSP which Muth and Tompson [9] have researched. The JSP has 6 jobs and 6 operations for each job to be processed on 6 different machine shops. The optimal

schedule requires 55 time units. Table 4 is the details of the JSP. For each cell, the first number is the machine shop number, and the second the processing time.

Job	Operation					
	1	2	3	4	5	6
1	3(1)	1(3)	2(6)	4(7)	6(3)	5(6)
2	2(8)	3(5)	5(10)	6(10)	1(10)	4(4)
3	3(5)	4(4)	6(8)	1(9)	2(1)	5(7)
4	2(5)	1(5)	3(5)	4(3)	5(8)	6(9)
5	3(9)	2(3)	5(5)	6(4)	1(3)	4(1)
6	2(3)	4(3)	6(9)	1(10)	5(4)	3(1)

TABLE 4 A 6x6 Job shop problem (Muth & Thomposon, 1963).

The proposed improved GA by data mining was applied to solve the JSP. In Step 1, combinations of crossover rate (P_c) and mutation rate (P_m) of different values were tested. In the experiment, the population size is set to 30 and the maximum number of generations is 500. 50 runs were tested for each combination of P_c and P_m. The numbers of runs reaching the optimum and the average numbers of generations reaching the optimum are listed in Table 5. The best combination, P_c=40 and P_m=30, was further used for data mining in step 2.

Population size = 30 Termination : Max generations = 500 # of Test runs = 50									
P_c	40	40	40	30	30	30	20	20	20
P_m	10	20	30	10	20	30	10	20	30
# of runs reached Opt.	2	9	10	3	7	10	3	3	10
AVG # of generations reached Opt.	103	188	115	286	218	190	293	129	123

TABLE 5 Test results of different combinations of crossover and mutation rates.

In mining association rules, the population size was set to 30, P_c=40, and P_m=30. 100 GA generations were tested for 100 runs, then the CIT was used for mining association rules, given the minimum fitness value and minimum number of count. Further in Step 3, GA without rules and GA with rules at different percentages of rules being adopted (50%, 60%, and 70%) were experimented on the same JSP. The results were shown in Table 6. It shows that GA with rules reached optimum for more number of runs and requiring smaller numbers of generations than GA without rules. Besides, for GA with rules, the higher percentage of rules were adopted, the more number of runs reached the optimum and the smaller number of generations required to reach the optimum.

GA with Rules				GA without Rules
Population size = 30 Pc = 40 ; Pm = 30 Termination : Max # of generations = 300 # of Test runs = 100				Population size = 30 Pc = 40 ; Pm = 30 Termination : Max # of generations = 500 # of Test runs = 100
% of genes of a generation rules applied	50	60	70	-
# of runs reached Opt.	19	23	30	18
% improved	5.56	16.67	66.67	-
AVG# of generations reached Opt.	114.68	108.38	102.576	145
% improved	20.91	25.26	29.26	-

TABLE 6 Test results of experiments on a 6x6 JSP using new GA by data mining.

5. CONCLUSIONS AND FUTURE WORKS

A great number of new genetic algorithms have been proposed and applied successfully in various areas. In this research, an improved GA by data mining technique was developed and tested. The experiments on a 6×6 JSP problem has demonstrated its superior evolution efficiency. However, this is only the result of our preliminary attempt in combining GA and Data mining. A number of related issues such as the experiments on more complex scheduling problems, the impact of different strategies of rules adoption, the integration of GA and different data mining methods may all be the future works of great interest and potential.

REFERENCES

[1]. Changchien, S.W. & Lu, T.-C., "A New Efficient Association Rules Mining Method Using Class Inheritance Tree," Proceedings of the 12th International Conference of Information Management (ICIM2001), Taipei, Taiwan, May 18-19, 2001.

[2]. Cheng, R., Gen, M. & Tsujimura, Y. "A tutorial survey of job-shop scheduling problems using genetic

algorithms: I. Representation," Computers and Industrial Engineering, 1996,30(1), 983-997.

[3]. Cheng, R., Gen, M. & Tsujimura, Y. "A tutorial survey of job-shop scheduling problems using genetic algorithms: II. Hybrid genetic search strategies," Computers and Industrial Engineering, 1999,36(2), 343-364.

[4]. Goldberg, D.E. *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.

[5]. Holland, J.H. *Adaptation natural and artificial system*, Ann Arbor, MI: University of Michigan Press, 1975.

[6]. Jiawei, H. & Micheline, K. *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers, 2000.

[7]. Koonce, D.A. & Tsai, S.-C. "Using data mining to find patterns in genetic algorithm solutions to a job shop schedule," Computers & Industrial Engineering, 2000,38, 361-374.

[8]. Laarhoven, Van., Aarts, P. E. & Lenstra, J. "Job shop scheduling by simulated annealing," Operations Research, 1992,40(1), 113-125.

[9]. Muth, J.F. & Thomposn, G.L. *Industrial Scheduling*, Prentice Hall, Englewood Gliffs, Ney Jersay, 1963.

[10]. Pawlak, Z. "Rough sets," International Journal of Information and Computer Sciences, 1982,11(1), 341-356.

[11]. Sule, D.R. *Industrial scheuling*, PWS Pulishing Company, chapter 7, 1997.

[12]. Syswerda, G. *Schedule optimization using genetic algorithms, Handbook of Genetic Algorithm*, New York, NY: Van Nostrand Reinhold, 1991.

[13]. Wang, L. & Zheng, D.-Z. "An effective hybrid optimization strategy for job-shop scheduling problems," Computer & Operations Research, 2001,28, 585-596.

[14]. Yamada, T. & Nakano, R. "Genetic Algorithms for Job-shop Scheduling Problem," Proceedings of Modern Heuristic for Decision Support, pp.67-81, March 18-19, 1997, London, UK.