Journal of Software Engineering and Applications, 2016, 9, 208-214
Published Online May 2016 in SciRes. http://www.scirp.org/journal/jsea
http://dx.doi.org/10.4236/jsea.2016.95017

A Promising Initial Population Based Genetic Algorithm for Job Shop Scheduling Problem

Vedavyasrao S. Jorapur[1], Vinod S. Puranik[2], Anand S. Deshpande[3], Mahesh Sharma[4]

[1]Department of Industrial and Production Engineering, Basaveshwar Engineering College, Bagalkot, India
[2]Basaveshwar Engineering College, Bagalkot, India
[3]Gogte Institute of Technology, Belagavi, India
[4]Fr. Conceicao Rodrigues College of Engineering, Bandra, West Mumbai, India
Email: Jorapur@fragnel.ac.in, Vinod.puranik536@gmail.com, anandeshpande@yahoo.co.in, Mahesh@fragnel.ac.in

Received 29 September 2015; accepted 24 May 2016; published 27 May 2016

Abstract

Job shop scheduling problem is typically a NP-Hard problem. In the recent past efforts put by researchers were to provide the most generic genetic algorithm to solve efficiently the job shop scheduling problems. Less attention has been paid to initial population aspects in genetic algorithms and much attention to recombination operators. Therefore authors are of the opinion that by proper design of all the aspects in genetic algorithms starting from initial population may provide better and promising solutions. Hence this paper attempts to enhance the effectiveness of genetic algorithm by providing a new look to initial population. This new technique along with job based representation has been used to obtain the optimal or near optimal solutions of 66 benchmark instances which comprise of varying degree of complexity.

Keywords

Job Shop Scheduling, Job Based Representation, NP-Hard, Recombination Operators etc.

1. Introduction

Scheduling is one of the most critical issues in planning and managing of manufacturing activities. Mathematically it is treated as NP-Hard problem. An optimal schedule for a given problem (a manufacturing industry) depends on so many factors like shop floor condition, constraints with which each process is carried out and so on.

How to cite this paper: Jorapur, V.S., Puranik, V.S., Deshpande, A.S. and Sharma, M. (2016) A Promising Initial Population Based Genetic Algorithm for Job Shop Scheduling Problem. *Journal of Software Engineering and Applications*, **9**, 208-214.
http://dx.doi.org/10.4236/jsea.2016.95017

Job shop scheduling is one of the most difficult problems in this area [1]. Fisher and Thompson introduced benchmark problems in 1963 [2] and since then many researchers have studied these problems and proposed exact methods and approximate algorithms [3]-[7]. Exact methods like branch and bound, linear programming and Lagrangian relaxation are able to solve small instances and will require a large amount of time with increase in problem size. In most of the cases it is reasonable to use a technique which may yield a near optimal solution requiring a lesser amount of time compared to the methods listed above. This has given rise to the use of heuristics, meta-heuristics or hybrid search algorithms (for ex. shifting bottleneck procedure, tabu search, ant colony etc.) by many researchers in the recent past. These algorithms have potential to find high quality solutions in a reasonable computational time. These algorithms have a special quality of adapting themselves to different kinds of scheduling problems and are easy to implement. Genetic algorithms were first successfully applied by Davis in 1985 [8]. Scheduling rules such as Shortest Processing Time (SPT), Most Work Remaining (MWKR) were used by Zhou and Feng [9], in his proposed hybrid heuristics GA for JSSP. Extensive use of genetic algorithms to solve job shop scheduling problems can be seen through literature survey [10]. But it can be observed that no systematic approach has been adopted in modifying the genetic algorithm while using it for JSSP. It may be noted that simple genetic algorithm essentially consists of mainly three aspects and needs critical observation. They are initialization, cross over and mutation operations. According to the authors a systematic approach to modify the simple genetic algorithm would be to initiate the modification in initializing the population itself which helps in lowering the make span and with remaining two operators, it may be further reduced.

If initial population is diverse enough then it is possible to choose best solutions for recombination operations and this may reduce the computational time required. Dispatching Rules (DRs) have been applied consistently to scheduling problems. They are procedures designed to provide good solutions to complex problems in real time. Many authors claim that priority dispatching rules can be successfully used in solving large JSSPs and even other scheduling problems [11]. Mahanim *et al*. [12] used Genetic Algorithm (GA) with some modifications to deal with problem of job shop scheduling which generated an initial population randomly including the result obtained by some well-known priority rules such as shortest processing time and longest processing time. Kuczapski *et al*. [13] presented an efficient method of enhancing Genetic Algorithms (GAs) for solving the Job-Shop Scheduling Problem (JSSP), by generating near optimal initial populations.

2. Problem Formulation

Because of its practical importance, Job shop Scheduling Problems have been modeled in different ways based on assumptions as well as situations of the production system. This study considers a single objective of minimizing the make span. The assumptions under which this objective holds well are as follows:

- At a time only one job is processed on a machine.
- No pre-emption.
- Processing in strict adherence to the precedence constraints.
- No re-working.
- No set up times.

Mathematically it may be expressed as:

Objective function:

$$\max\left\{C_i \middle| i = 1, \cdots, n\right\}$$

Subjected to:

$$St_i + p_i \le St_j \tag{1}$$

$$\forall i, j \in I, m(i) = m(j) \text{ and } J(i) \ne J(j) \tag{2}$$

where, St_i is the starting time, P_i is the processing time of i^{th} operation and St_j is the starting time of operation "*j*". The second equation resolves the conflict between two jobs to be operated at the same time on the same machine.

So many models were presented in the past [14]-[16] and were used either to obtain minimum make span or simply to make representation simpler. Algorithms like immediate selection and shifting bottleneck heuristics were proposed by Carlier [17], Adams [18], and Lars Monch [19]. And these algorithms were due to disjunctive graph model.

Table 1. Results of instances as % deviation.

Data Set	Type	Best	Min. Make span in 12 runs	Min. Deviation %
mt06	(Opt)	55	**55**	**0.000**
mt10	(Opt)	930	**960**	**3.226**
mt20	(Opt)	1165	**1192**	**2.318**
abz05	(Opt)	1234	**1241**	**0.567**
abz06	(Opt)	943	**964**	**2.227**
abz07	(LB)	654	**719**	**9.939**
abz08	(LB)	635	**738**	**16.220**
abz09	(LB)	656	**742**	**13.110**
car01	(Opt)	7038	**7038**	**0.000**
car02	(Opt)	7166	**7166**	**0.000**
car03	(Opt)	7312	**7422**	**1.504**
car04	(Opt)	8003	**8003**	**0.000**
car05	(Opt)	7702	**7767**	**0.844**
car06	(Opt)	8313	**8313**	**0.000**
car07	(Opt)	6558	**6558**	**0.000**
car08	(Opt)	8264	**8344**	**0.968**
la01	(Opt)	666	**666**	**0.000**
la02	(Opt)	655	**655**	**0.000**
la03	(Opt)	597	**599**	**0.335**
la04	(Opt)	590	**590**	**0.000**
la05	(Opt)	593	**593**	**0.000**
la06	(Opt)	926	**926**	**0.000**
la07	(Opt)	890	**890**	**0.000**
la08	(Opt)	863	**863**	**0.000**
la09	(Opt)	951	**951**	**0.000**
la10	(Opt)	958	**958**	**0.000**
la11	(Opt)	1222	**1222**	**0.000**
la12	(Opt)	1039	**1039**	**0.000**
la13	(Opt)	1150	**1150**	**0.000**
la14	(Opt)	1292	**1292**	**0.000**
la15	(Opt)	1207	**1207**	**0.000**
la16	(Opt)	945	**946**	**0.106**

Continued

la17	(Opt)	784	**784**	**0.000**
la18	(Opt)	848	**853**	**0.590**
la19	(Opt)	842	**866**	**2.850**
la20	(Opt)	902	**913**	**1.220**
la21	(LB)	1040	**1081**	**3.942**
la22	(Opt)	927	**970**	**4.639**
la23	(Opt)	1032	**1032**	**0.000**
la24	(Opt)	935	**1002**	**7.166**
la25	(Opt)	977	**1023**	**4.708**
la26	(Opt)	1218	**1273**	**4.516**
la27	(LB)	1235	**1317**	**6.640**
la28	(Opt)	1216	**1288**	**5.921**
la29	(LB)	1120	**1233**	**10.089**
la30	(Opt)	1355	**1377**	**1.624**
la31	(Opt)	1784	**1784**	**0.000**
la32	(Opt)	1850	**1851**	**0.054**
la33	(Opt)	1719	**1719**	**0.000**
la34	(Opt)	1721	**1749**	**1.627**
la35	(Opt)	1888	**1888**	**0.000**
la36	(Opt)	1268	**1334**	**5.205**
la37	(Opt)	1397	**1467**	**5.011**
la38	(LB)	1184	**1278**	**7.939**
la39	(Opt)	1233	**1296**	**5.109**
la40	(Opt)	1222	**1284**	**5.074**
orb01	(Opt)	1059	**1099**	**3.777**
orb02	(Opt)	888	**906**	**2.027**
orb03	(Opt)	1005	**1056**	**5.075**
orb04	(Opt)	1005	**1032**	**2.687**
orb05	(Opt)	887	**909**	**2.480**
orb06	(Opt)	1010	**1038**	**2.772**
orb07	(Opt)	397	**411**	**3.526**
orb08	(Opt)	899	**917**	**2.002**

Average deviation is found to be **2.461** across all the instances.

3. Representation of the Problem in GA and GA Operators

The very objective of using evolutionary algorithms like Genetic Algorithms is to make the search process computationally efficient. This is so because techniques like branch and bound etc. are guaranteed techniques to provide the optimal solution but are computationally inefficient. Random walk or gradient search for example is basically random search or gradient descent techniques which will search one solution at a time. Hence these methods are also computationally inefficient with the problems of larger size. Genetic algorithms are well suited in such cases to find the best possible solution close to optimal solution in a computationally efficient manner. Different mathematical models may lead to different representations for the same problem [20]. Attempts to explain different available representations and explore the use of a better representation scheme for the job shop scheduling problems while using genetic algorithms was done and the study was conducted over 66 bench mark instances. An attempt was made in [21] to classify the representations as direct and indirect type which was further classified as model based and algorithm based in [22]. Giffler and Thompson proposed an algorithm way back in 1960 [23] which makes use of priority rules and have enough potential to provide feasible or good solutions. It was due to Bean [24], Beirwirth [25]-[27] the representations like random keys representation, permutation with repetition, machine based representation and job based representation respectively were developed and studied for their effectiveness.

4. Methodology

With the job based representation [20], the previous study had shown favorable results. Therefore, by using the same representation in genetic algorithms, this study aims to establish the effect of initial population scheme on the overall convergence of each benchmark instance. For initial population only random job based selection is used in place of other schemes. The pseudo-codes for the same are given below:

Step1: Obtain an eligible set of jobs from the given instance. Let it be $\varnothing = \langle 1 \cdots n \rangle$.

Step 2: Choose randomly any one job from the eligible set and place its corresponding operation in the sequence set. Let it be $S = \langle 1 \cdots k \rangle$.

Step 4: Update the job status.

Step 5: Delete the sequenced job from eligible set Ø, if all the operations of the job are sequenced and then update it with eligible jobs.

Step 6: Repeat the step 2 until all the operations on all the jobs are sequenced.

Uniform crossover [28], One-Point and Two-Point crossover techniques are used in the study as one of the recombination operators or in some proportion. Flipping is used as a mutation operator. For better convergence, a **Static Critical Path** was also found for each instance and mutation was carried out among the critical operations (*i.e.* to add the processing times of all the jobs getting processed on all the machines and the job with longest duration is to be considered as critical path for the instance). General flow chart of GA is given below:

Step 1: Generate the initial population using random job selection method. Evaluate the fitness of each individual. Let $t = 0$.

Step 2: Use crossover reproduction operator to generate the offspring.

Step 3: Carry out mutation on each offspring to generate new individuals. Calculate the fitness value of each offspring.

Step 4: If the stop criterion is satisfied, then stop. Otherwise let $t = t + 1$ and turn to step 2.

In this study, Random Job Based algorithm is used as a first step to generate the initial population to analyze whether this algorithm has any effect on the overall performance of GA which is presented above.

5. Results and Analysis

With the random job selection for initial population followed by job based representation scheme adopted, the study was conducted with 50 generations and a population size of 1000.Mutation probability varies with 0.1 to 0.9 values dynamically and elite population size is 20%. Reproduction probability used in this study is 0.1 Parents in this study were selected from two groups sorted out based on fitness value (*i.e.* minimum make span). Each parent is selected from these groups probabilistically. In the study, GA is programmed with different reproduction and mutation operators like PPX, single point and N-point crossover mechanisms. The benchmark problems used in this paper are taken from OR library [29] available in World Wide Web. All the experiments are conducted with a Pentium-4 single core processor with clock speed of 2.06 GHz and RAM of 512 Mbs. 66 benchmark instances were considered and in the twelve runs, the best values obtained are compared with lower bound or optimum value of the benchmark instances reported in the literature. The percentage deviation for each instance is calculated by the formula shown below:

$$\frac{\text{BKS or Opt.} - \text{Curr.Solution}}{\text{BKS or Opt.}} * 100$$

where BKS = Best Known Solution; Opt. = Optimum Solution and Curr.Solution = Solution obtained by the method used in the study. Results as % deviation are shown in **Table 1**.

6. Conclusion & Future Scope

In this paper, new method is introduced for initializing population. By using simple job based representation followed by random job based initialization along with reproduction operators like PPX, single point and three points cross over techniques, it is possible to get optimal or near optimal results. The average deviation obtained is much less compared with any other results obtained in literature [30], where other advanced GA operators are also used. In continuation with the theme of applying a common algorithm for maximum number of instances available in the literature, authors intend to use a new approximation method to detect critical operations based on the schedule and this will be followed by CPM based mutation operator to improve the solutions further.

References

[1] Lenstra, J.K., Kan, A.H.G. and Brucker, P. (1977) Complexity of Machine Scheduling Problem. *Annals of Discrete Mathematics*, **1**, 343-362. http://dx.doi.org/10.1016/S0167-5060(08)70743-X

[2] Fisher, H. and Thompson, G.L. (1963) Probabilistic Learning Combinations of Local Job-Shop Scheduling Rules. Prentice-Hall, Englewood Cliffs, 225-251.

[3] Ge, H.W. and Sun, L. (2008) An Effective PSO and AIS-Based Hybrid Intelligent Algorithm for Job-Shop Scheduling. *IEEE Transactions on System*, *Man*, *and Cybernetics-Part A*: *Systems and Humans*, **38**, 358-368. http://dx.doi.org/10.1109/TSMCA.2007.914753

[4] Dey, S., Sarkar, D. and Basu, A. (2010) A Tag Machine Based Performance Evaluation Method for Job-Shop Schedules. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **29**, 1028-1041. http://dx.doi.org/10.1109/TCAD.2010.2049067

[5] Wang, L., Cai, N., Feng, H.Y. and Ma, J. (2010) ASP: An Adaptive Setup Planning Approach for Dynamic Machine Assignments. *IEEE Transactions on Automation Science and Engineering*, **7**, 2-14. http://dx.doi.org/10.1109/TASE.2008.2011919

[6] Gokbayrak, K. and Selvi, O. (2010) Service Time Optimization of Mixed-Line Flow Shop Systems. *IEEE Transactions on Automatic Control*, **55**, 395-404. http://dx.doi.org/10.1109/TAC.2009.2037273

[7] Wang, S.F. and Zou, Y.R. (2003) Techniques for the Job Shop Scheduling Problem: A Survey. *Systems Engineering—Theory & Practice*, **23**, 49-55.

[8] Davis, L. (1985) Job Shop Scheduling with Genetic Algorithms. *Proceedings of the First International Conference on Genetic Algorithms*, **5**, 136-140.

[9] Zhou, H. and Feng, Y. (2001) The Hybrid Heuristic Genetic Algorithm for Job Shop Scheduling. *Computers & Industrial Engineering*, **40**, 191-200. http://dx.doi.org/10.1016/S0360-8352(01)00017-1

[10] Jain, A.S. and Meeran, S. (1999) Deterministic Job-Shop Scheduling: Past, Present and Future. *European Journal of Operational Research*, **113**, 390-434. http://dx.doi.org/10.1016/S0377-2217(98)00113-1

[11] Ho, A. and Tay, J. (2005) Evolving Dispatching Rules for Solving the Flexible Job-Shop Problem. *IEEE Congress on Evolutionary Computation*, **3**, 245-276.

[12] Omar, M., Baharum, A. and Abu Hasan, Y. (2006) A Job-Shop Scheduling Problem (JSSP) Using Genetic Algorithm—SHOP. *Proceedings of the* 2nd *IMT-GT Regional Applications*, University Sains Malaysia, 13-16.

[13] Kuczapski, M., Micea1, V., Maniu, A. and Cretu, A. (2010) Efficient Generation of Near Optimal Initial Population to Enhance Genetic Algorithm for Job-Shop Scheduling. *Information Technology and Control*, **39**, 32-37.

[14] Manne, A.S. (1960) On the Job-Shop Scheduling Problem. *Operations Research*, **8**, 219-223. http://dx.doi.org/10.1287/opre.8.2.219

[15] Park, B.J., Choi, H.R. and Kim, H.S. (2003) A Hybrid Genetic Algorithm for the Job Shop Scheduling Problems.

Computers & Industrial Engineering, **l4**, 597-613. http://dx.doi.org/10.1016/S0360-8352(03)00077-9

[16] Roy, B. and Sussmann, B. (1964) Les Problems d'Ordon Ordonnancement Avec Constraints Disjunctives. SEMA, Note D.S., Paris.

[17] Carlier, J. and Pinson, E. (1989) An Algorithm for Solving the Job-Shop Problem. *Management Science*, **35**, 164-176. http://dx.doi.org/10.1287/mnsc.35.2.164

[18] Adams, J., Balas, E. and Zawack, D. (1988) The Shifting Bottleneck Procedure for Job Shop Scheduling. *Management Science*, **34**, 391-401. http://dx.doi.org/10.1287/mnsc.34.3.391

[19] Monch, L., Schabacker, R., Pabst, D. and Fowlerb, J.W. (2007) Genetic Algorithm Based Sub Problem Solution Procedures for a Modified Shifting Bottleneck Heuristic for Complex Job Shops. *European Journal of Operational Research*, **3**, 2100-2118. http://dx.doi.org/10.1016/j.ejor.2005.12.020

[20] Jorapur, V., Puranik, V.S., Deshpande, A.S. and Sharma, M.R. (2014) Comparative Study of Different Representations in Genetic Algorithms for Job Shop Scheduling Problem. *Journal of Software Engineering and Applications*, **7**, 571-580.

[21] Cheng, R., Gen, M. and Tsujimura, Y. (1996) A Tutorial Survey of Job-Shop Scheduling Problems Using Genetic Algorithms—I. Representation. *Computers and Industrial Engineering*, **30**, 983-997. http://dx.doi.org/10.1016/0360-8352(96)00047-2

[22] Abdelmaguid, T.F. (2010) Representations in Genetic Algorithm for the Job Shop Scheduling Problem: A Computational Study. *Journal of Software Engineering and Applications*, **3**, 1155-1162.

[23] Giffler, B. and Thompson, G.L. (1960) Algorithms for Solving Production-Scheduling Problems. *Operations Research*, **8**, 487-503. http://dx.doi.org/10.1287/opre.8.4.487

[24] Bean, J. (1994) Genetic Algorithms and Random Keys for Sequencing and Optimization. *ORSA Journal on Computing*, **6**, 154-160. http://dx.doi.org/10.1287/ijoc.6.2.154

[25] Bierwirth, C. (1995) A Generalized Permutation Approach to Job Shop Scheduling with Genetic Algorithms. *Operations-Research-Spektrum*, **17**, 87-92. http://dx.doi.org/10.1007/bf01719250

[26] Anderson, E.J., Glass, C.A. and Potts, C.N. (2003) Local Search in Combinatorial Optimization. Princeton University Press, Princeton.

[27] Holsapple, C.W., Jacob, V.S., Pakath, R. and Zaveri, J.S. (1993) A Genetics-Based Hybrid Scheduler for Generating Static Schedules in Flexible Manufacturing Contexts. *IEEE Transactions on Systems, Man, and Cybernetics*, **23**, 953-972. http://dx.doi.org/10.1109/21.247881

[28] Syswerda, G. (1989) Uniform Crossover in Genetic Algorithms. *Proceedings of the* 3rd *International Conference on Genetic Algorithms*, Fairfax, June 1989, 2-9.

[29] Beasley, J.E. (2008) Job Shop Scheduling. http://people.Brunel.ac.uk/~mastjjb/jeb/orlib/jobshopinfo.html

[30] Qing-dao-er-ji, R. and Wang, Y.P. (2012) A New Hybrid Genetic Algorithm for Job Shop Scheduling Problem. *Computers and Operations Research*, **39**, 2291-2299.