RESEARCH ARTICLE

International Journal of Multidisciplinary and Scientific Emerging Research

©2014 IJMSER, All Rights Reserved
Available at http://www.ijmser.com/ (ISSN 2349 – 6037)

Optimization of Job Scheduling in Flow Shop Environment using Genetic Algorithm Considering Sequence Dependent Setup Times

Rajdeep Singh[1], Upender Dhull[2], Sanjeev Sharma[3] and Sandeep Jindal[4]

[1]Professor, Department of Mechanical Engineering, CGC Chandigarh Engineering College, Landran, Mohali, Punjab
[2]Assistant Professor, Department of Mechanical Engineering, University Institute of Engineering and Technology, Kurukshetra, Haryana,
[3]Associate Professor, Department of Mechanical Engineering, Chandigarh Engineering College, Landran, Mohali, Punjab
[4]Sandeep Jindal, Department of Mechanical Engineering, CGC Chandigarh Engineering College, Landran, Mohali, Punjab

Accepted **15** November 2015, Available online **14** January 2016, **Vol.4, No.2 (December 2015)**

Abstract

Among typical production scheduling problems, job shop scheduling is one of the strong NP-complete combinatorial optimization problems. Using an enhanced genetic algorithm, an effective crossover operation for real coded job-based representation can be used to guarantee the feasibility of solutions, which are decoded into active schedules during the search process. This paper attempts to assign an optimum job sequence on a machine considering the objective of minimizing total makespan time in a flow shop environment based on the philosophy of Genetic Algorithms. The setup time or changeover time is taken as the main factor involved in the makespan time where setups are sequence dependent. For analyzing the results simulations are performed with various population sizes and crossover probabilities.

Keywords: *Job-Shop Scheduling, Flow-Shop Scheduling, Scheduling Strategies, Genetic Algorithm ps.*

1. INTRODUCTION

Scheduling decisions allocate workloads to specific work-centers and determine the sequence in which operations are performed within the available capacity. Since scheduling is an allocation decision, it uses the resources made available by aggregate planning. Scheduling is a multiple level activity and at the top level emphasis is given to the scheduling of production and plant operations over an extended period of time. At a lower level, scheduling is mostly concerned with the sequencing of job orders to the machines and the operators, popularly called as shop scheduling. Shop scheduling is the act of sequencing the different jobs to the machine in an attempt to minimize the waiting time of the jobs while maintaining well balanced utilization of the machines.

Bector et al (1966) developed optimal solution for a Sequencing problem of n jobs with one machine in which all the jobs had a common due date. Goyal (1975) analyzed the flow-shop sequencing problem with no wait in process and with jobs, requiring dissimilar technological order. Yamamoto (1985) Investigated decomposition method to obtain an approximate solution for the large scale job shop scheduling problem. Halim and Ohta (1993) developed a heuristic algorithm for solving scheduling problem of flow shop with receiving and delivery just in time. The job-shop scheduling problem is to find the sequence of jobs on each machine in order to minimize a given objective function. Some objective functions can be the minimization of makespan, mean weighted tardiness, etc. Davis (1985) started off GA research on the job-shop scheduling problem, Nakano (1991) reports on using the edge recombination operator (designed initially for the TSP) on job-shops.

Most of the conventional heuristic procedures use a dispatching (priority) rule under the situation of choosing an operation from unscheduled operations. In recent years, better solution approaches have been attempted such as simulated annealing (SA), Glover (1989) tabu search (TS) and genetic algorithms (GA). There have been various approaches using set of dispatching rules and combining search strategies with dispatching rule. Genetic algorithm is among such attempts and has been applied to a variety of areas of decision-making including many in scheduling

problems. The most frequently used objectives of job shop scheduling problems are minimization of mean flow time of jobs and minimization of the make span.

Genetic algorithm is stochastic search procedure for combinatorial optimization problems based on the mechanism of natural selection and natural genetics. These use the idea of survival of the fittest by progressively accepting better solutions to the problems. The elements and mechanisms of genetic algorithm are representation, population, evaluation, selection, operators and parameters. The algorithm starts with a randomly generated initial set of population consisting of so called chromosomes that represent the solution to problems. These are evaluated for the fitness function or one of the objective functions and are selected according to their fitness value. In the job shop scheduling problem, the objective function is the make span or mean flow time of given schedule which has to be minimized.

GA differs from normal optimization and search procedures in four ways:

a) GA works with a coding of the parameter set, not the parameters themselves.
b) GA searches from a population of points, not a single point.
c) GA uses payoff (objective function) information, not derivatives or other auxiliary knowledge.
d) GA uses probabilistic transition rules and not deterministic rules.

Above characteristics of GA provide it the flexibility to adapt itself to changing optimization criteria and constraints. That is why GA is used in this work to find the optimal job sequence in such a way so as to minimize the total makespan time of jobs.For this, a problem in flow shop environment requiring n jobs to be sequenced on one of the bottleneck machine is chosen, as this is the machine which decides the production rate of the whole line.

Assumptions made during this work are:

a) All machines are available at time zero and there is no breakdown.
b) Process times are deterministic and
c) Setup or changeover times are sequence dependent.

In the present work an experiment is designed to find out the effect of different GA parameters on the average fitness. For this, simulation of a sophisticated genetic algorithm is performed using software developed in C++.

2. METHODOLOGY

In this work a sophisticated genetic algorithm involving real coding chromosome along with a cyclic crossover is used.

1. Representation

Real coding or permutation representation is used where the numbers 1, 2, 3 etc. are actually the job ID's. eg. ajob sequence of 1, 3,2,4,5 is represented by the chromosome 1 3 2 4 5.

2. Crossover

Cyclic crossover is being adopted in this work and is represented below:

Before crossover	After crossover
3 1 2 4 5 6	5 1 2 4 3 6
6 5 4 2 1 3	1 5 4 2 6 3

Cyclic Crossover

Crossover operator chosen is cyclic due to two reasons:

a) Operators other than PMX and CX give redundant sequences after crossover.
b) CX gives better results than PMX as comes out from the literature survey.

Cyclic crossover operator selects few positions randomly from one parent sequence and places these elements in the other parent in the same order. eg. in the case given above from first parent sequence randomly three positions 2, 4 and 6 are selected and elements at these positions i.e. 1, 4 and 6 are searched and placed in the same order in the second parent sequence to get the first offspring and similar operation is done for second parent to get second offspring. Mutation operator is not used in the program as the cyclic operator creates enough variation to get variety in next generation.

Job ID	**0**	**1**	**2**	**3**	**4**
Processing Time (in min.) per 100 units	120	130	140	100	90

Table 1: Test Data for Processing Times.

Simulation using GA based job sequencing software:
Computer Implementation of Genetic Algorithm

Algorithm is based on the Genetic Algorithm (GA) with elitism developed in the C++ and various steps are as follows:

Step 1: Initialize Population (Function name-: initialize)
Initial Population consists of strings of pop size no. of job sequences generated randomly where "pop size" is the population size. In the program we have taken pop size =20 .Thus after initialization we get an array pop array of dimensions pop size X njob where njob is the number of jobs.

Step 2: Data entry (Function name-: data entry)
The values of setup and processing times are given through this function. This function results in the

formation of two arrays proc_time [njob] & Setup time [njob][njob].

Step 3: Make span time calculation (Function name-: time_ calc)
In this function the total make span time is calculated by using the relation
Total _ time factor
=∑ proc_time [njob] +
∑ Setup _time [njob] [njob]

This objective function is of minimization type so we convert it to maximization type by taking the new objective function as
Total time=1.0/ (1.0+total_timefactor)

Step 4: Retain the best individual (Function name-: keep_the_best)
This function keeps track of the best member of the population and puts the best individual as the last entry in the population array.

Step 5: Select the parents of high fitness from the old population to form new generation (Function name-: select)
In this function new pop array is created based on the comparison of the cumulative fitness with the random numbers. If random number is less than the cumulative fitness of nth job sequence than the nth job sequence is selected in the new pop array. In this way pop size numbers of random numbers are compared with the cumulative fitness of the sequences and new pop-array is made after this steps completion from *w*hich few of the job sequences are again to be selected randomly for crossover.

Changeover Matrices			
Setup Time Matrix	**Rsetup_Time matrix**	**Setup Time**	**Rsetup_Time**
0 0	0 0	100	50
0 1	1 0	120	145
0 2	2 0	234	56
0 3	3 0	140	175
0 4	4 0	160	120
1 0	0 1	45	75
1 1	1 1	130	112
1 2	2 1	100	115
1 3	3 1	230	180
1 4	4 1	122	144
2 0	0 2	134	102
2 1	1 2	100	155
2 2	2 2	105	120
2 3	3 2	112	133
2 4	4 2	245	200
3 0	0 3	75	80
3 1	1 3	120	140
3 2	2 3	190	77
3 3	3 3	80	88
3 4	4 3	45	70
4 0	0 4	170	180
4 1	1 4	145	122
4 2	2 4	166	220
4 3	3 4	110	90
4 4	4 4	208	112

Table 2: Test Data for Setup Time.

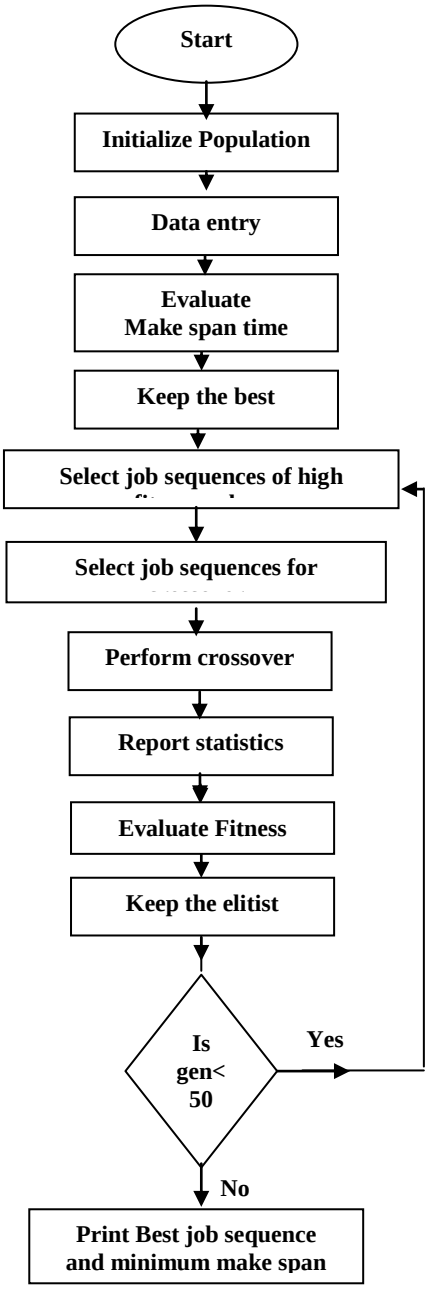

Figure 1: Flow Chart for Job Sequencing Using GA.

Step 6: Select job sequences for crossover (Function name-: X over)
In this step few job sequences are selected for crossover randomly. This function returns the job sequence number for the function crossover which actually performs the crossover on these sequences.

Step 7: Perform crossover (Function name-: crossover)
This function actually performs cyclic crossover on the job sequences return by the Xover function and replaces the respective parent in the pop array with the offspring obtained after crossover.

Step 8: Select the Elitist (Function name-: elitist)
This function compares the best member of the current generation with the best member of the previous generation and replaces it with the latter if it is worse than the latter

Step 9: Report the statistics (Function name-: report)
This function is used to report the various statistics related to the GA such as best value, average fitness, standard deviation etc. in a file named "galog.txt"

Step 10: Iterate for n generations the same procedure from Step 5 to 9 and report the best job sequence and minimum make span time at the end.

Parameters used-
1.) Population size=5, 10, 15, 20, 30
2.) Probability of crossover, Pxover=0.10, 0.20, 0.25
3.) Probability of mutation= 0.0
4.) Maximum number of generations =20, 50,100,200

RESULTS

The standard deviation showed a drastic decrease in few generations (more than 50 % in 5 generations) of the program. Also average fitness increases to optimum level within very small number of generation runs. Cyclic crossover proved really effective in minimizing the number of generations to get the optimum values(less than 100) in comparison to PMX and ordered crossovers which took more than 500 runs to reach the optimum value of time [1].Also with population size the average fitness increases and standard deviation decreases as evident from the graphs plotted from the output shown in figures 2 to 5.

The minimum time for the above test data was found to be equal to 949 minutes and the minimum make span time job sequence is 3 4 2 0 1 for GA parameters pop_size=20,max. No. of generations=50, Pxover=0.25 and njob=5.

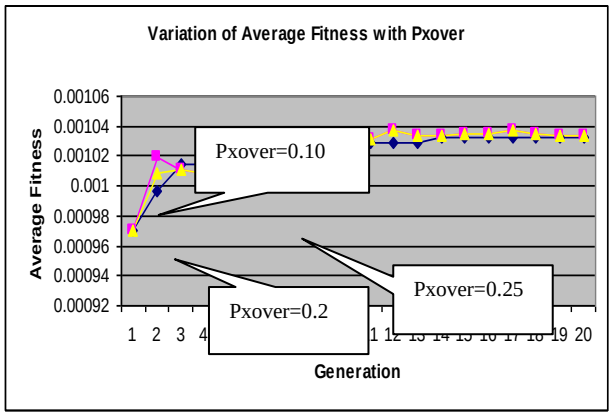

Figure 2: Effect of Probability of Crossovers on Average Fitness.

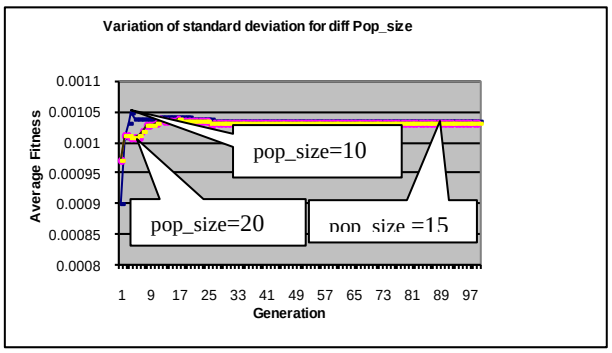

Figure 3: Effect of Probability of Crossovers on Standard Deviation.

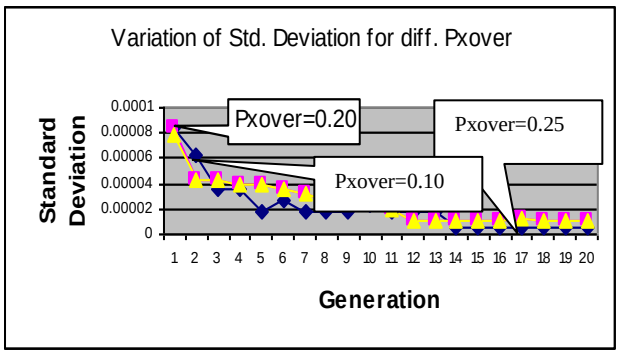

Figure 4: Effect of Pop Size on Average Fitness.

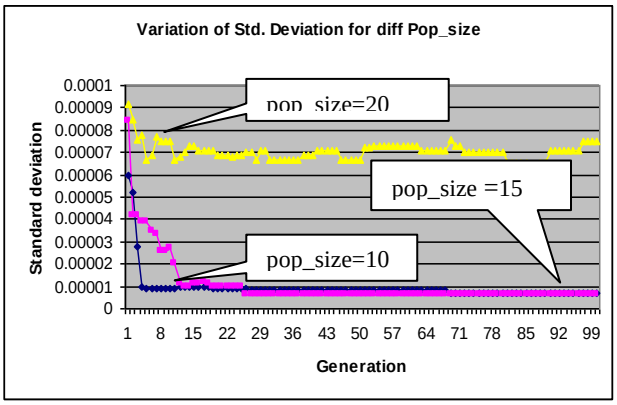

Figure 5: Effect of Pop Size on Standard Deviation.

REFERENCES

[1] Bector, C.R., Gupta, Y.F. and Gupta, N.C. (1966), "Determination of an optimal common due date and optimal sequence in a single machine job shop", International Journal of production research, Vol.26, No.4

[2] Conway, R.W., Maxwell, W.L., Miller, L.W. (1967), "Theory of Scheduling", Addison Wesley, Reading, MA.

[3] Davis, L. (1985), "Job-shop scheduling with genetic algorithms" In J.J. Grefenstette, editor, Proceedings of the First International Conference on Genetic Algorithms, pp 136-140. Lawrence Erlbaum Associates

[4] Dorndorf, U., Pesch, E. (1995), "Evolution based learning in a job-shop scheduling environment", Computers and Operations Research 22 (1), 25-40.

[5] Glover, F.(1989), "Tabu search Part I." ORSA Journal on Computing 1 (3), 190-206.

[6] Goldberg D. E. (1989), "Genetic Algorithms in Search, Optimization and Machine Learning", Addison-Wesley.

[7] Gupta, S.K. (1982), "Computerized production scheduling", Productivity, XXIII, Vol.1.

[8] Halim, A. H. and Ohta, H., "Batch scheduling problem through the flow shop with both receiving and delivery just in time", International Journal of production research, Vol.31, No.8 (1993).

[9] Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G. (1982), "Recent developments in deterministic sequencing and scheduling: A survey" In: Dempster, M.A.H. et al. (Eds.), Deterministic and Stochastic Scheduling. Reidel, Dordrecht, pp. 35-73.

[10] Littger, K.W., "A new approach for the optimum solution of the M by J production scheduling problem", International Journal of production research, Vol.14, No .l.

[11] Nakano, R. and Yamada, T., (1991), "Conventional genetic algorithms for job-shop problems" In R.K. Belwl and L.B. Booker, editors, Proceedings of the Fourth International Conference on Genetic Algorithms, ICGA-9, pages 474-479, Morgan Kaufman.

[12] Parker, R.G. (1995), "Deterministic Scheduling", Chapman and Hall, London.

[13] Yamamoto, N., and N of S.Y. (1985), "Scheduling in the manufacturing operating system environment", International Journal of production research, Vol. 23, No. 8.