Performance Comparison Of New Heuristic With Genetic Algorithm In Parallel Flow Line Set Up

K.Balasubramanian, Raja Rajeswari College of Engineering, Chennai, Tamilnadu, India
A.Noorul Haq, National Institute of Technology, Tamilnadu, India
N.Rajeswari, Sri Venkateswara College of Engineering, Chennai, Tamilnadu, India

Abstract

A new heuristic has been developed to solve the problem in parallel flow line scheduling. It involves the minimization of the makespan by the optimal allocation of a finite number of jobs to finite number of lines in the first phase and the optimal sequencing of allocated jobs in each line in the second phase. Here new heuristic and genetic algorithm for analyzing the parallel flow line scheduling are discussed and executed on a set of randomly generated problems. The results obtained for the test problems suggest that the developed new heuristic can be used successfully to solve large scale parallel flow line scheduling problems.

Keywords
Scheduling, Genetic Algorithm, New Heuristic

Introduction

Scheduling has been defined as "the art of assigning resources to tasks in order to ensure the termination of these tasks in a reasonable amount of time" [1]. According to French [2], the general job shop problem is to find a sequence in which the jobs pass between the resources, which a feasible schedule, and optimal with respect to some criterion. Scheduling problems are among the most important problems in the industry because they have an impact on the ability of the manufacturer to meet the customer demands and make a profit and also on the ability of autonomous system to optimize their operations, the deployment of intelligent systems, and the optimization of communication systems. Parallel Line Job Shop Scheduling involves the optimal allocation and scheduling of jobs in multiple processing lines and the approaches to solve these problems include Mathematical Programming, Neural Networks, Fuzzy Logic, Expert Systems and algorithms like Genetic Algorithm, Tabu Search etc. In this paper, a new heuristic has been developed to tackle the problem of parallel line job shop scheduling. The commonly used algorithms like Genetic Algorithm and Tabu Search can be applied on account of their characteristics such as easy realization. However, these algorithms are optimal in case of huge parallel architecture and suffer from redundancy.

David B.Shymos et al., [3] discuss scheduling of parallel machines on a single processing line. Amotz Bar-Noy et al., [4] solve the problem of approximating the makespan of multiple machines in real time scheduling. Jeffrey W.Sherman et al., [5] have discussed global job shop scheduling using genetic algorithm without considering multiple processing lines. C.Chekuri et al., [6] also consider minimization of completion time as the objective for their approximation techniques. George J.Kyparisis and Chrisos Koulamas [7] address the assembly line scheduling with concurrent operations per stage and parallel machines. Zhiwei Fu et al., [8] deal with "A Genetic Algorithm based approach for building Accurate Decision Trees". The data set is divided into training, scoring and test sets, and they find that the approach can be used effectively on very large data sets. Zong-Zhi Lin et al., [9] have developed and evaluated an exact algorithm, GAMP, which combines a genetic algorithm and mixed integer program to construct a minimal set of function that describes the value function. Zvi Drezner [10] deals with a new genetic algorithm for the Quadratic Assignment problem. In his paper, he proposes several variants of a new genetic algorithm for the solution of the quadratic assignment problem. S.Binato et al., [11] have developed a Greedy Randomized Adaptive Search Procedure (GRASP) for solving the scheduling problem. An intensification strategy and Proximate Optimality Principle are incorporated in the construction phase. Albert Jones [12] has discussed the various job shop scheduling techniques comprehensively. Albrecht et al., [13] have developed fast parallel heuristics for the scheduling problem. They have utilized neighborhood relationship to solve the problem. In comparison with the above papers, this paper is unique in the sense that a new heuristic has been developed to optimize the scheduling and the performance compares with the genetic algorithm results.

Problem Environment

Parallel flow Line setup consists of a number of parallel lines, each consisting of a numbers of machines. The corresponding machines on any two given parallel lines are identical and the number of machines on all the lines is equal. For example, machine M1 on line L1 is identical to machine M1 on line L2 and so on. The processing time of job varies from machine to machine along a given line. Also the processing times of a job on corresponding machines of different lines vary due to the difference in capabilities of the corresponding machines. The quantity of the job also varies from job to job. Each job is allocated to a particular line and a job cannot move between lines. The jobs enter their respective lines simultaneously and are processed in parallel. The time when the last job is completed on any line is the span of that line. The objective is to minimize the makespan (maximum of the spans of the lines).

This work has been carried out with the following assumptions of non-preemption of jobs and machines, processing of any job in any line, deterministic individual processing time, negligible transportation time between two workstations, no breakdown of machines, setup time dependent sequence and no prescribed priority between individual jobs.

The following notations shall be used to define the problem.

t_{ilk}-processing time for job i in line l on machine k

{Where i= 1,2,3…n, l= 1,2,3…j & k= 1,2,3…m}

q-batch quantity of job i

s_{i-i} – set up time for job to job

The objective function of the problem is to determine which job is to be allocated in which line and in what sequence so as to minimize the makespan.

$$\text{Min } Z(X) = \sum_{i=1}^{n} \sum_{l=1}^{j} \sum_{k=1}^{m} t_{ilk} + (q_i - l) \max (t_{ilk}) + s_{i-i}$$

NEW HEURISTIC (NH):

The newly developed heuristic for the parallel flow line scheduling problem is described below. It has two phases: Phase I is used for allocation of jobs in lines and Phase II is used for sequencing the allocated jobs in the allocated lines.

Phase: I Allocation of jobs

Step I: Calculate the total processing time for each job on each line

Step II: Arrange the total processing time in ascending order

Step III: Allocation of jobs to lines

a) Scan the sorted table from the last element

b) Note the job number having maximum total processing time

c) The line in which the job number has the least total processing time should be used to allocate the job.

d) Leaving the job out, among the remaining combinations the next allocation is made following the same procedure.

e) It is to be noted that the number of jobs allocated in a line does not exceed the integral value of the number of jobs / number of machines.

f) After each line is allocated to an equal number of jobs which is the integral value mentioned above, only the combinations for the remaining number of jobs are considered. The job – line combination having maximum total processing time should be allocated to the line having minimum processing time.

Phase: II Sequencing of jobs

Step I: The jobs are sequenced in SPT order of each job.

Step II: The span of each line is calculated with setup time and the maximum span denotes the makespan.

Genetic Algorithm (GA)

Genetic Algorithm (GA) is an adaptive heuristic search algorithm premised on the evolutionary ideas of natural selection and genetic. It has been widely studied, experimented and applied in many fields of engineering. In this study, we follow the basic notation and idea addressed by Michalewics [14], which allows us to use any data structure suitable for a problem together with any set of meaningful genetic operators. There are three common genetic operators such as selection, crossover and mutation. The selection procedure has a significant influence on driving the search towards a promising area and finding good solutions in a short time. Crossover operates on two chromosomes at a time and generates offspring by combining both chromosome features. Mutation operates on one chromosome and generates offspring by altering one or more genes.

Methodology

Coding

We form a list of job symbol and partitioning symbol as the coding scheme for multiple parallel machines scheduling problem, which can be essentially viewed as a kind of extended permutation representation. The job symbols denoted with an integer represent all possible permutation of jobs (sequence of jobs) and the partitioning symbols, denoted with slash, designate the partition of jobs to lines. With an example of 8 jobs and 3 lines, the chromosome can be represented as follows:

4 3 3 / 2 7 5 1 4 8 3 6

2 4 4 / 5 2 7 1 4 3 8 6

Fitness function

Chromosomes are selected to form new solutions (off spring) according to their fitness function value. The more suitable they are, the more chances they have to reproduce. In this study, the fitness function value for each chromosome is equal to the schedule objective function, i.e. Fit (X) = Z (X).

Crossover

Some common crossover operations are one-point crossover, two-point crossover, cycle crossover and uniform crossover. The proposed single point crossover takes two parents and creates two off springs by propagating at single point from partition symbol " / ". Both the sides from "/" and we get offspring as shown below:

P1	4 3 3 / 2 7 5 1 4 8 3 6
P2	2 4 4 / 5 2 7 1 4 3 8 6
Off 1	4 3 3 / 5 2 7 1 4 3 8 6
Off2	2 4 4 / 2 7 5 1 4 8 3 6

Mutation

We use random exchanging as our mutation, *i.e., se*lect two random genes and then exchange their positions, which are only one side from slash. The randomly selected genes may be either job or line. The different combinations of job and line result in two basic types of mutation. One case is that two selected jobs are exchanged to form offspring 1 and the other case is that two lines are exchanged to form offspring 2 as shown below.

P1	4 3 3 / 2 7 5 1 4 8 3 6 9 10
Off 1	4 3 3 / 2 7 5 **6** 4 8 3 **1** 9 10
Off 2	**3 4** 3 / 2 7 5 1 4 8 3 6 9 10

Now we summarize our implementation of genetic algorithms as follows:

Set population size, p_c, p_m & maximum generation.

1. Generate initial population of chromosomes.
2. Evaluate the fitness value in terms of the objective function for each chromosome in the population and calculate average fitness and standard deviation,
3. Cross over the parents with the crossover probability (p_c) to form new off springs.

4. Mutate new off springs at a random with a mutation probability (p_m).
5. Place new off springs in a new population.
6. If generation equals to maximum generation, stop the evolutionary process; otherwise return to step 2.

Numerical results

Three examples are given here to show that the algorithm presented in this article is effective. We have compared the different results by GA with NH. Both the algorithms have been written in C++ and they were run with PentiumIV system.

Example 1. A small scale parallel flow line scheduling problem has been considered of 4 jobs, 2 lines and 3 machines. The processing times of each job and sequence setup time matrix are in Table1 and Table 2 respectively.

Table: 1 Processing time

Job	Line	Machine 1	Machine 2	Machine 3
J1	LI	7	3	2
	L2	4	6	5
J2	LI	6	5	6
	L2	2	4	2
J3	LI	1	7	5
	L2	2	8	6
J4	LI	3	6	4
	L2	5	3	7

Table – 2 Sequence setup time matrix

Job	1	2	3	4
1	---	5	6	2
2	4	---	7	7
3	2	5	---	3
4	3	6	7	---

Table – 3 Result

Algorithm	Line	Sequence	Makespan
Genetic Algorithm	L1	J3 – J1	177
	L2	J4 – J2	
New Heuristic	L1	J2 – J4	220
	L2	J1 – J3	

From the results shown in Table 3, it is obvious that GA can yield the best solution for this problem**.**

Example 2. A small scale identical parallel machine scheduling problem has considered 5 jobs, 3 lines and 4 machines. The processing time of each job and sequence setup time matrix are listed in Table 4 and Table 5 respectively.

Table: 4 Processing time

Job	Line	Machine 1	Machine 2	Machine 3	Machine 4
J1	L1	3	7	8	7
	L2	4	6	2	9
	L3	1	5	7	2
J2	L1	3	8	2	8
	L2	6	1	8	3
	L3	4	7	6	1
J3	L1	5	4	3	2
	L2	1	7	9	4
	L3	3	8	5	4
J4	L1	6	9	2	3
	L2	9	6	4	2
	L3	1	2	9	7
J5	L1	3	2	1	8
	L2	7	5	2	2
	L3	8	2	2	7

Table – 5 Sequence setup time matrix

Job	1	2	3	4	5
1	---	5	2	3	5

2	2	---	5	5	3
3	5	4	---	4	2
4	1	8	9	---	4
5	4	6	6	7	---

Table – 6 Result

Algorithm	Line	Sequence	Makespan
Genetic Algorithm	L1	J4 – J3	76
	L2	J2 – J5	
	L3	J1	
New Heuristic	L1	J2 – J3	77
	L2	J4 – J5	
	L3	J1	

The result of Example 2 is shown in Table 6. From this, it is inferred that the difference between the two algorithms is insignificant compared to the previous case.

Example 3. A larger scale identical parallel flow line scheduling problem has considered 10 jobs, 3 lines and 3 machines. The processing time of each job and sequence setup time matrix are listed in Table 7 and Table 8 respectively. The result of Example 3 is shown in Table 9.

Table - 7 Processing time

Job	Line	Machine 1	Machine 2	Machine 3
J1	L1	2	3	4
	L2	5	6	4
	L3	3	5	6
J2	L1	3	2	4
	L2	4	3	4
	L3	5	2	6
J3	L1	6	3	2
	L2	5	4	8
	L3	7	4	5
J4	L1	2	8	6
	L2	5	4	3
	L3	2	6	4
J5	L1	3	2	6
	L2	4	8	7
	L3	3	3	4

J6	LI	2	6	3
	L2	3	4	8
	L3	4	3	3
J7	L1	3	2	6
	L2	4	3	3
	L3	3	8	4
J8	L1	5	6	4
	L2	4	8	2
	L3	2	8	6
J9	L1	6	4	3
	L2	2	3	3
	L3	6	8	7
J10	L1	5	4	3
	L2	2	4	3
	L3	3	8	6

Table – 8 Sequence setup time matrix

Job	1	2	3	4	5	6	7	8	9	10
1	---	2	4	3	2	1	5	6	7	2
2	3	---	2	2	7	3	2	1	6	5
3	4	5	---	2	5	7	6	4	3	2
4	2	1	1	---	4	3	5	7	8	1
5	2	2	3	1	---	2	2	4	2	4
6	3	1	2	7	6	---	5	4	3	5
7	1	3	2	1	3	4	---	6	7	2
8	5	6	7	1	2	2	3	---	3	2
9	6	2	5	2	3	3	2	4	---	1
10	4	7	7	3	3	1	1	3	5	---

Table – 9 Result

Algorithm	Line	Sequence	Makespan
Genetic Algorithm	L1	J1 – J6 – J 7	123
	L2	J8 – J9 – J2 – J4	
	L3	J10 – J5 – J3	
New Heuristic	L1	J1 – J3 – J8	115
	L2	J7 – J10 – J9 – J4	
	L3	J5 – J6 – J2	

The result shows that a more satisfactory solution can still be obtained by new heuristic. When the problem scale becomes larger, the quality of the best solution and the average solution has advantage over that obtained by GA.

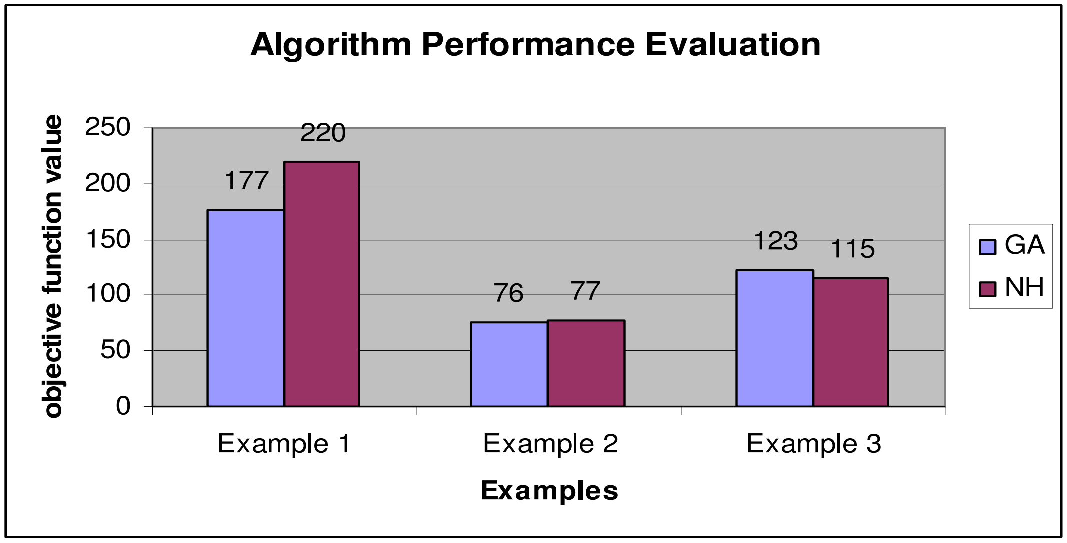

Figure – 1 Algorithms Comparison

From the results of examples cited and figure -1, it is concluded that when the degree of complexity of a problem increases, the new heuristic performs better than GA. The solution of numerous randomly generated large-scale problems reveals that the newly developed heuristic outperforms GA.

Conclusion

Genetic Algorithm is used for the scheduling problem successfully and it has shown great search advantages in solving NP problem in the scope of optimization. Although it is seen easily that there is no demand for differentiability, convex of objective function and wide adaptability for large size problems, the heuristic developed provides better solution. In future the developed heuristic can be used to solve various objective functions and its performance can be compared with other non-traditional techniques.

References

1.Dempster.M, Lenstra.J and Kan.K, (1981). Deterministic and Stochastic Scheduling; Introduction, *Proceedings of the NATO Advanced Study and Research Institute on Theoretical Approaches to Scheduling Problems*, D Reidel Publishing Company.

2. French. S, (1982). Sequencing and Scheduling, New York Halsted Press.

3. David B. Shmoys, Joel Wein, and David P. Williamson, (1995). Scheduling Parallel Machines Online, *Society for Industrial and Applied Mathematics Journal on Computing*, Vol.24, No.6 pp, 1313 – 1331.

4.Amotz Bar – Noy, Sudipto Guha Joseph Noar, Bruch Schieber, (2001). Approximating the throughput of Multiple Machines in Real Time Scheduling, *Society for Industrial and Applied Mathematics Journal on Computing*, Vol.31, No.2, pp.331 - 352.

5. Jeffrey W.Herman, Chung – Yee Lee, hinchman, (1995). Global Job Shop Scheduling with Genetic Algoirthm, *Production and Operations Management*, Vol.4, Issue: 1, pp.30 – 45.

Chekuri.C, Motwani.R, Natarajan.B, Stein.C, (2001). Approximation Techniques for Average Completion Time Scheduling, *Society for Industrial and Applied Mathematics Journal on Computing,* Vol.31, No.1, pp.146-166.

George J.Kyparisis, Christos Koulmas, (2002). Assembly Line Scheduling with Concurrent Operations and Parallel Machines, *INFORMS Journal on Computing* Vol.14, Issue:1, pp.68 – 80.

8. Zhiwei Fu, Bruce R.Golden, Shreevardhan Lele, S.Raghavan, Edward A.Wasil, (2003). AGenetic Algorithm Based Approach for Building Accurate Decision Trees, *INFORMS Journal on Computing,* Vol.15, Issue: 1, pp. 3-22.

9. Zong – Zhi Lin, James C Bean, Chelsea C.White, (2004). A Hybrid Genetic / Optimization Algorithm for Finite – Horizon, Partially Observed Markov Decision Processes, *INFORMS Journal on Computing,* Vol.16, Issue: 1, pp. 27-38.

10. Zvi Drezner, (2003).A New Genetic Algorithm for the Quadratic Assignment Problem, *INFORMS Journal on Computing,* Vol.15, Issue:3, pp.320-330.

11. Binato.S, Hery.W.J, Lowenstern.D.M, Resende.M.G.C, (2000). A GRASAP for Job Shop Scheduling, AT & T Labs Research Technical Report, Vol.1, pp.16.

12. Albert Jones, juis C.Rebelo, (2000). Survey of Job Shop Scheduling Techniques, *National Institute of Standards and Technology.*

13.Alberecht.A, Steinhofel.K, Wong.C.K, (2002). Fast Parallel Heuristics for the Job Shop Scheduling Problem, *Computers and Operations Research*, Vol.21, pp.151 – 169.

14. Michalewics, Z. (1992). “Genetic Algorithms + data structure = evolution programs”, *Berlin: Springer*.