Intelligent Information Management, 2011, 3, 204-214
doi:10.4236/iim.2011.35025 Published Online September 2011 (http://www.SciRP.org/journal/iim)

GA Based Heuristic to Minimize Makespan in Single Machine Scheduling Problem with Uniform Parallel Machines

Panneerselvam Senthilkumar[1*], Sockalingam Narayanan[2]
[1]*Program Management Office*, *Ashok Leyland Limited*, *Chennai*, *India*
[2]*VIT University*, *Vellore*, *India*
E-mail: [*]*psenthilpondy@gmail.com*
Received June 19, 2011; *revised July* 21, 2011; *accepted July* 30, 2011

Abstract

This paper considers the single machine scheduling problem with uniform parallel machines in which the objective is to minimize the makespan. Four different GA based heuristics are designed by taking different combinations of crossover methods, viz. single point crossover method and two point crossover method, and job allocation methods while generating initial population, viz. equal number of jobs allocation to machines and proportionate number of jobs allocation to machines based on machine speeds. A detailed experiment has been conducted by assuming three factors, viz. Problem size, crossover method and job allocation method on 135 problem sizes each with two replications generated randomly. Finally, it is suggested to use the GA based heuristic with single point crossover method, in which the proportionate number of jobs allocated to machines based on machine speeds.

Keywords: Uniform Parallel Machines, Genetic Algorithm, Crossover Method, Job Allocation Method

1. Introduction

The single machine scheduling problem with parallel machines is classified into the following three categories.

- Single machine scheduling with identical parallel machines;
- Single machine scheduling with uniform parallel machines;
- Single machine scheduling with unrelated parallel machines.

Let, t_{ij} be the processing times of the job j on the machine i, for $i = 1, 2, 3, \cdots, m$ and $j = 1\ 2, 3, \cdots, n$.

Then the types of parallel machines scheduling problem are defined using this processing time.

1) If $t_{ij} = t_{1j}$ for all i and j, then the problem is called as *identical parallel machines scheduling problem*.

2) If $t_{ij} = t_{1j}/s_i$ for all i and j, where s_i is the speed of the machine i and t_{1j} is the processing time of the job j on the machine 1, then the problem is termed as *uniform* (*proportional*) *parallel machines scheduling problem*.

The speeds are assumed as s_1, s_2, s_3, $\cdots$, and s_m for the parallel machines 1, 2, 3, $\cdots$, and m, respectively with the relation $s_1 < s_2 < s_3 < \cdots < s_m$. For a given job, its processing times on the parallel machines will be in the ratio of $1/s_1$: $1/s_2$:$1/s_3$: $\cdots$:$1/s_m$.

3) If t_{ij} is arbitrary for all i and j, then the problem is known as *unrelated parallel machines scheduling problem*.

In this paper, the single machine scheduling problem with uniform parallel machines is considered with the objective of minimizing the makespan. When n jobs with single operation are scheduled on m parallel machines, then each parallel machine will have its completion time of the last job in it.

The maximum of such completion times on all the parallel machines is known as the makespan of the parallel machines scheduling problem, which is an important measure of performance [1].

The characteristics of the uniform parallel machines scheduling problem are as listed below.

- It has n single operation jobs.
- It has m parallel machines with different speeds ($s_1 < s_2 < s_3 < \cdots < s_m$).
- m machines are continuously available and they are never kept idle while work is waiting.
- t_{1j} is the processing time of the job j on the machine 1

for $j = 1, 2, 3, \cdots, n$.

- For each job, its processing times on the uniform parallel machines are inversely proportional to the speeds of those parallel machines ($1/s_1$:$1/s_2$: $1/s_3$: $\cdots$: $1/s_m$), where s_1 is the unit speed.
- $t_{ij} = t_{1j}/s_i$ for $j = 1, 2, 3, \cdots, n$ and $i = 2, 3, \cdots, m$.

In this paper, off-line, non-preemptive single machine scheduling problem with uniform parallel machines is considered.

2. Literature Review

In this section, the review of off-line, non-preemptive single machine scheduling problem with uniform parallel machines is presented.

Panneerselvam Senthilkumar and Sockalingam Narayanan [2] have done a comprehensive review of literature of single machine scheduling problem with uniform parallel machines, in which 17 classifications were discussed. Prabuddha De and Thomas E.Morton [3] have developed a new heuristic to schedule jobs on uniform parallel processors to minimize makespan. It is tested on a large number of problems for both uniform and identical processors. They found that the solutions given by the heuristic for the uniform parallel machines scheduling are within 5% of the solutions given by the branch and bound algorithm. Bulfin and Parker [4] have considered the problem of scheduling tasks on a system consisting of two parallel processors such that the makespan is minimized. In particular, they treated a variety of modifications to this basic theme, including the cases of identical processors, proportional (uniform) processors and unrelated processors. In addition, they suggested a heuristic scheme when precedence constraints exist.

Friesen and Langston [5] examined the non-preemptive assignment of n independent tasks to a system of m uniform processors with the objective of reducing the makespan. It is known that *LPT* (*longest processing time first*) schedules are within twice the length of the optimum makespan [6]. They analyzed a variation of the MULTIFIT algorithm derived from the algorithm for bin packing problem and proved that its worst-case performance bound on the makespan is within 1.4 times of the optimum makepsan. Gregory Dobson [7] has given a worst-case analysis while applying the *LPT* (longest processing Time) heuristic to the problem of scheduling independent tasks on uniform processors with the minimum makepsan. In this research, a bound of 19/12 is derived on the ratio of the heuristic to the optimal makespan. Friesen [8] examined the non-preemptive assignment of independent tasks to a system of uniform processors with the objective of minimizing the makespan. The author showed that the worst case bound for the *largest processing time first* (LPT) algorithm for this problem is tightened to be in the interval (1.52 to1.67). Hochbaum and Shmoys [9] devised a polynomial approximation scheme for the minimizing makespan problem on uniform parallel processors. The technique employed is the dual approximation approach, where infeasible but super-optimal solutions for a related (dual) problem are converted to the desired feasible but possibly suboptimal solution.

Chen [10] has examined the non-preemptive assignment of independent tasks to a system of m uniform processors with the objective of minimizing the makepsan. The author has examined the performance of *LPT* (*largest processing time*) schedule with respect to optimal schedules, using the ratio of the fastest speed to the slowest speed of the system as a parameter.

Mireault, Orlin, Vohra [11] have considered the problem of minimizing the makespan when scheduling independent tasks on two uniform parallel machines. Out of the two machines, the efficiency of one machine is q times as that of the other machine. They computed the maximum relative error of the LPT (*largest processing time first*) heuristic as a function of q.

Burkard and He [12] derived the tight worst case bound $\sqrt{6/2+(1/2)^k}$ for scheduling jobs using the MULTIFIT heuristic on two parallel uniform machines with k calls of *FFD* (first fit decreasing) within MULTIFIT. Burkard, He and Kellerer [13] have developed a linear compound algorithm for scheduling jobs on uniform parallel machines with the objective of minimizing makespan. This algorithm has three subroutines, which run independently in order to choose the best assignment among them. Panneerselvam and Kanagalingam [14] have presented a mathematical model for parallel machines scheduling problem with varying speeds in which the objective is to minimize the makespan. Also, they discussed industrial applications of such scheduling problem. Panneerselvam and Kanagalingam [15] have given a heuristic to minimize the makespan for scheduling n independent jobs on m parallel processors with different speeds.

Agarwal, Colak, Jacob and Pirkul [16] have proposed new heuristics along with an augmented-neural-netwrok (AugNN) formulation for solving the makespan minimization task-scheduling problem for the non-identical machine environment. They explored four task and three machine-priority rules, resulting in 12 combinations of single-pass heuristics. They gave the AugNN formulation for each of the 12 heuristics and showed computational results on 100 randomly generated problems of sizes ranging from 20 to 70 tasks and 2 to 5 machines. The results clearly showed that AugNN provides significant improvement over single-pass heuristics.

Panneerselvam Senthilkumar and Sockalingam Narayanan [17] have developed a simulated annealing algorithm to minimize the makespan in the single machine scheduling problem with uniform parallel machines. In the first phase, a seed generation algorithm is presented and then it is followed by three variations of the simulated annealing algorithm. They compared these three simulated annealing algorithms and found that there is no significant difference among them in terms of makepsan. So, they suggested to use all the three simulated annealing algorithms for a given problem and select the best solution.

Cristina Mihaila and Alin Mihaila [18] have developed an evolutionary algorithm for single machine scheduling problem with uniform parallel machines, in which the objective is to minimize the makespan. They also, compared their algorithm with other meta-heuristics and reported the results. Alin Mihaila and Cristina Mihaila [19] have developed a genetic algorithm to minimize the makespan of the uniform parallel machines scheduling under single machine scheduling and experimented with instance problems and reported that their algorithm performs better when compared to other algorithms.

From, the literature, it is clear that the objective of minimizing the makespan in single machine scheduling problem with uniform parallel machines comes under combinatorial category. Hence, development of heuristic is inevitable for this problem. Hence, in this paper, an attempt has been made to design a GA based heuristic to minimize the makespan in single machine scheduling problem with uniform parallel machines.

3. Factors Affecting GA Based Heuristic

In this paper, a GA based heuristic is designed to minimize the makespan in single machine scheduling problem with uniform parallel machines.

The genetic algorithm mimics the mechanism of selection and evaluation. It generates successive population of alternate solutions until a solution is obtained that yields acceptable results. It is based on the fundamental processes that control the evolution of biological organisms, namely natural selection and reproduction.

The skeleton of the genetic algorithm is given below [20].

Step 1: Input the maximum number of successive population to be generated (Q). Let the generation count (GC) be 1

Step 2: Generate a random initial population with N chromosome. Let this population be L.

Step 3: Evaluate the fitness function $f(x)$ of each chromosome x in L.

Step 4: Sort L by ascending /descending as per the objective and copy a specified percentage of chromosomes (30%) into a subpopulation P.

Step 5: Randomly select two chromosomes and do the following:

5.1 Perform Crossover operation.

5.2 Perform Mutation of each offspring for a mutation probability, α.

5.3 Replace the new two offspring in L along with their fitness function values

Step 6: Repeat Step 5 until all the chromosomes in P are considered.

Step 7: GC = GC + 1

Step 8: IF GC ≤ Q, then go to Step 4; else go to Step 9.

Step 9: From L, identify the chromosome which has the best fitness function value and print its results.

Step 10: Stop.

The performance of the genetic algorithm is suspected to be affected by the crossover method, mutation, the way in which the initial population is generated and the problem size.

In the GA based heuristic, the design factors considered in this paper are as listed below.

- Problem size (Factor A) in terms of number of machines and number of jobs, for which the levels are 2 × 11, 2 × 12, 2 × 13, ···, 2 × 25, 3 × 11, 3 × 12, 3 × 13, ···, 3 × 25, ···, 10 × 11, 10 × 12, 10 × 13, ···, 10 × 25.
- Crossover method (Factor B), for which the levels are "Single point crossover method" and "Two pint crossover method".
- Method of allocation of jobs to machines while generating initial population (Factor C), for which the levels are "Equal number of allocation of jobs to machines" and "Proportionate number of allocation of jobs to machines, which is based on the speed of the machines".

3.1. Methods of Job Allocation to Machines

This section explains the methods of allocation of jobs to different machines while generating the initial population (Factor C), viz. equal number of jobs allocation to machines and proportionate number of jobs allocation to machines.

3.1.1. Equal Number of Jobs Allocation to Machines

In the method which assigns equal number of jobs to each machine, the construction of chromosome is explained below.

Let, NJ_i be the number of jobs assigned to machine i, $i = 1, 2, 3, \cdots, m$

$NJ_i = n/m$, if (n/m) is integer; for $i = 1, 2, 3, \cdots, m$.

Otherwise,

$$NJ_i = \text{Int}(n/m), \text{ for } i = 1, 2, 3, \cdots, m - 1$$

$$NJ_m = n - \sum_{i=1}^{m-1} NJ_i$$

If the number of jobs is 9 and the number of machines is 3, then a sample chromosome is as presented by Chromosome 1 in **Table 1** by randomly assigning each machine number to three jobs. If the number of jobs and the number of machines are 10 and 3, respectively, then a sample chromosome is as presented by Chromosome 2 in **Table 1**.

In the above representation of chromosomes, each gene represents a machine to which the corresponding job is assigned. The generation of genes is random subject to fulfilling the number of jobs assigned to each of the machines.

The determination of the makespan for the Chromosome 1 is shown in **Figure 1** by assuming the processing times as in **Table 2**.

3.1.2. Proportionate Number of Jobs Allocation to Machines

The construction of a chromosome in the method of proportionate number of jobs allocation to machines is explained below.

The speed ratio of the machines be $S_1:S_2:S_3:\cdots:S_m$, in which $S_1 < S_2 < S_3 < \cdots < S_m$

Let, NJ_i be the number of jobs assigned to machine i, $i = 1, 2, 3, \cdots, m$

$NJ_i = \text{Int}\{[S_i/(S_1 + S_2 + S_3 + \cdots + S_m)] \times n\}$, if the integer value is more than 0; = 1, otherwise,
for $i = 1, 2, 3, \cdots, m - 1$

Table 1. Representation of chromosomes using equal number of jobs assignment to machines.

	Job Number									
	1	2	3	4	5	6	7	8	9	10
Chromosome 1:	3	1	3	2	1	3	2	1	2	
Chromosome 2:	2	3	2	1	3	1	3	2	3	1

Table 2. Processing times of jobs shown in chromosome 1.

		Speed ratio	Job								
			1	2	3	4	5	6	7	8	9
Machine	1	1	6	9	24	12	6	18	24	12	6
	2	2	3	4.5	12	6	3	9	12	6	3
	3	3	2	3	8	4	2	6	8	4	2

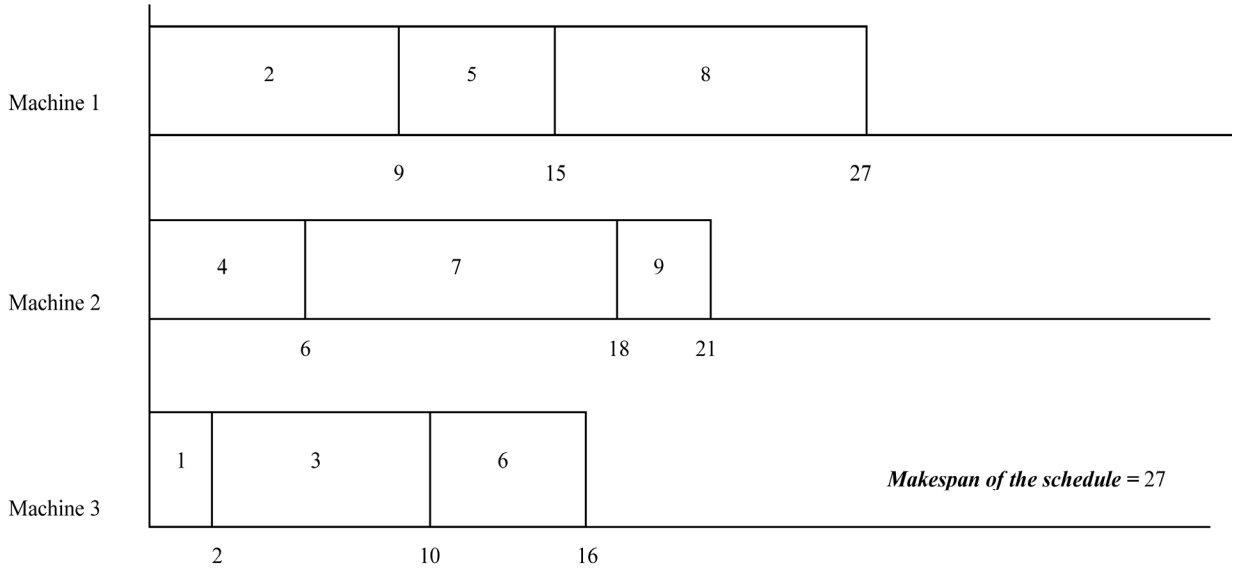

Figure 1. Gantt chart to determine makespan of chromosome 1.

$$NJ_m = n - \sum_{i=1}^{m-1} NJ_i$$

Let the number of jobs be 10 and the number of machines be 4 with speed ratio 1:2:3:4 for the machines 1, 2, 3 and 4, respectively. A sample chromosomes for this situation is as presented by the Chromosome 3 in **Table 3** by randomly assigning Machine 1 to one job, Machine 2 to two jobs, Machine 3 to 3 jobs and Machine 4 to four jobs as per their speed ratio. Assume another situation in which the number of jobs and the number of machines are 10 and 5, respectively. Let the speed ratio of the machines be 1:2:3:4:5 for the machines 1, 2, 3, 4 and 5, respectively. A sample chromosome for this situation is as presented by the Chromosome 4 in **Table 3** by assigning Machine 1 to one job, Machine 2 to 1 job, Machine 3 to 2 jobs, Machine 4 to 2 jobs and Machine 5 to four jobs as per speed ratio.

3.2. Crossover Methods

In this paper, single point crossover method and two point crossover method are used in the experiment conducted to select the factors affecting the performance of the GA based heuristic to minimize the makespan of the single machine scheduling problem with uniform parallel machines. These are demonstrated using the chromosomes 5 and 6 which are given below in which the number of jobs is 10 and the number of machines is 4.

Chromosome 5: 2 1 3 4 3 4 4 2 3 4
Chromosome 6: 4 3 4 3 2 1 2 4 3 4

3.2.1. Single Point Crossover Method

The single point crossover method is explained using the chromosomes 5 and 6. Let the random position selected in the range 1 to 10 (positions of the job numbers in the chromosomes) be 4. Then, the chromosome 5 is divided into two parts, P and Q and the chromosome 6 is divided into two parts, X and Y as shown below.

	P	Q
Chromosome 5:	2 1 3 4	3 4 4 2 3 4

	X	Y
Chromosome 6:	4 3 4 3	2 1 2 4 3 4

The offspring 1 and offspring 2 generated using the single point crossover performed on these chromosomes are shown below.

	X	Q
Offspring 1:	4 3 4 3	3 4 4 2 3 4

	P	Y
Offspring 2:	2 1 3 4	2 1 2 4 3 4

3.2.2. Two Point Crossover Method

The two point crossover method is explained using the same set of chromosomes 5 and 6. Let the two random positions selected in the range 1 to 10 (positions of the job numbers in the chromosomes) be 3 and 6. Then, the chromosome 5 is divided into three parts, P, Q and R, and the chromosome 6 is divided into three parts, X, Y and Z as shown below.

	P	Q	R
Chromosome 5:	2 1 3	4 3 4	4 2 3 4

	X	Y	Z
Chromosome 6:	4 3 4	3 2 1	2 4 3 4

The offspring 3 and offspring 4 generated using the two point crossover performed on these chromosomes are shown below.

	X	Q	Z
Offspring 3:	4 3 4	4 3 4	2 4 3 4

	P	Y	R
Offspring 4:	2 1 3	3 2 1	4 2 3 4

4. GA Based Heuristic to Minimze Makespan

As stated earlier, the performance of the GA based heuristic to minimize the makespan of the single machine scheduling problem with uniform parallel machines is mainly suspected to be affected by the factors, viz., "Crossover Method" and "Job Allocation Method", each having two methods. So, four (2 × 2 = 4) GA based heu-

Table 3. Representation of chromosomes using proportionate number of jobs allocation to machines.

	Job Number									
	1	2	3	4	5	6	7	8	9	10
Chromosome 3:	2	1	3	4	3	4	4	2	3	4
Chromosome 4:	5	3	5	3	4	5	2	5	4	1

ristics by combining the levels of these factors to minimize the makespan are presented in this section.

4.1. GA Based Heuristic with Single-Point Crossover Method and Equal Number of Job Allocation Method

The steps of the GA based heuristic with single point crossover method and equal number of job allocation method to minimize the makespan of the single machine scheduling problem with uniform parallel machines are presented below.

Step 1: Input the following.

Number of machines (m)

Number of jobs (n) [It is assumed that $n \geq m$]

Speed ratio of the machines: S_1:S_2:S_3: ⋯:S_m, in which $S_1 < S_2 < S_3 < \cdots : < S_m$

Processing times $T_{i,j}$, i = 1, 2, 3, ⋯, m & j = 1, 2, 3, ⋯, n.

Mutation probability, α (0.3).

Step 2: Set the genetic algorithm parameters.

Size of population, N

Size of subpopulation, P (30% of N)

Number of iterations to be carried out, Q

Step 3: Construct N chromosomes of the population by allocating equal number of jobs to the machines in each chromosome, $CHROM_{K,J}$, K = 1, 2, 3, ⋯, *N and J* = 1, 2, 3, ⋯, n.

Step 4: Compute the makespan of each chromosome in the population, MS_K, K = 1, 2, 3, ⋯, N.

Step 5: Set the Iteration Number q to 1.

Step 6: Sort the chromosomes in the ascending order of their makespans.

Let the sorted chromosomes be, $SCHROM_{K,J}$, K = 1, 2, 3, ⋯, N, J = 1, 2, 3, ⋯, n and the array of their makespan be $MAKESPAN_K$, K = 1, 2, 3, ⋯, N.

Step 7: Copy the sorted chromosomes, $SCHROM_{K,J}$, K = 1, 2, 3, ⋯, N, J = 1, 2, 3, ⋯, n into $CHROM_{K,J}$, K = 1, 2, 3, ⋯ N, J = 1, 2, 3, ⋯, n.

Step 8: Update the best makespan, BEST_MS = $MAKESPAN_1$ and

The best chromosome, BCHROM = $SCHROM_{1,J}$, J = 1, 2, 3, ⋯, n.

Step 9: Treat the topmost 30% of the population (0.3N = P) of the sorted population as subpopulation for crossover operation.

Step 10: Set chromosome number, C = 1

Step 11: Perform single-point crossover between the chromosomes C and C + 1 as listed below and obtain the offspring C and C + 1.

Crossover between: $CHROM_{C,J}$, J = 1, 2, 3, ⋯, n & $CHROM_{C+1,J}$, J = 1, 2, 3, ⋯, n

Offspring: $OSPRING_{C,J}$, J = 1, 2, 3, ⋯, n & $OSPRING_{C+1,J}$, J = 1, 2, 3, ⋯, n

Step 12: Perform mutation in each of the offspring for a mutation probability of α.

Step 13: Compute the makespan of each of the offspring. [MS_C and MS_{C+1}].

Step 14: Increment chromosome number by 2, C = C + 2.

Step 15: If C ≤ P then go to Step 11.

Step 16: Copy the new offspring [$OSPRING_{C,J}$, C = 1, 2, 3, ⋯, P and J = 1 ,2 3, ⋯, n] to the chromosomes vector, $CHROM_{C,J}$, C = 1, 2, 3, ⋯, P and J = 1, 2, 3, ⋯, n, respectively.

Step 17: Increment the iteration number by 1 ($q = q$ + 1).

Step 18: If $q \leq$ Q, then go to Step 6.

Step 19: Print the following results.

Best makespan, BEST_MS

Best chromosome, BCHROM, which is $SCHROM_{1,J}$, J = 1, 2, 3, ⋯, n.

Step 20: Stop

4.2. GA Based Heuristic with Two Point Crossover Method and Equal Number of Jobs Allocation Method

The steps are same as given in the Section 4.1, except the Step 11. The Step 11 of the two-point crossover is shown below.

Step 11: Perform two point crossover between the chromosomes C and C + 1 as listed below and obtain the offspring C and C + 1.

Crossover between: $CHROM_{C,J}$, J = 1, 2, 3, ⋯, n & $CHROM_{C+1,J}$, J = 1, 2, 3, ⋯, n

Offspring: $OSPRING_{C,J}$, J = 1, 2, 3, ⋯, n & $OSPRING_{C+1,J}$, J = 1, 2, 3, ⋯, n

4.3. GA Based Heuristic with Single Point Crossover Method and Proportionate Number of Jobs Allocation Method

The steps are same as given in the Section 4.1, except the Step 3. The Step 3 is shown below.

Step 3: Construct N chromosomes of the population by allocating proportionate number of jobs to the machines in each chromosome, $CHROM_{K,J}$, K = 1, 2, 3, ⋯, N and J = 1, 2, 3, ⋯, n.

4.4. GA Based Heuristic with Two Point Crossover Method and Proportionate Number of Jobs Allocation Method

The steps are same as given in the Section 4.1, except the Step 3 and Step 11. The Step 3 and the Step 11 are

shown below.

Step 3: Construct N chromosomes of the population by allocating proportionate number of jobs to the machines in each chromosome, $CHROM_{K,J}$, K = 1, 2, 3, ···, N and J = 1, 2, 3, ···, n.

Step 11: Perform two point crossover between the chromosomes C and C + 1 as listed below and obtain the offspring C and C + 1.

Crossover between: $CHROM_{C,J}$, J = 1, 2, 3, ···, n & $CHROM_{C+1,J}$, J = 1, 2, 3, ···, n

Offspring: $OSPRING_{C,J}$, J = 1, 2, 3, ···, n & $OSPRING_{C+1,J}$, J = 1, 2, 3, ···, n.

5. Experimentation

In the GA based heuristic, the design factors are as listed below.

- Problem size (Factor A) in terms of number of machines and number of jobs, for which the levels are 2 × 11, 2 × 12, 2 × 13, ···, 2 × 25, 3 × 11, 3 × 12, 3 × 13, ···, 3 × 25, ···, 10 × 11, 10 × 12, 10 × 13, ···, 10 × 25.
- Crossover method (Factor B), for which the levels are "Single point crossover method" and "Two point crossover method".
- Method of allocation of jobs to machines while generating initial population (Factor C), for which the levels are "Equal number of allocation of jobs to machines" and "Proportionate number of allocation of jobs to machines, which is based on the speed of the machines"

A comparison is made between the GA based heuristics with these three factors to minimize the makespan of the single machine scheduling problem with uniform parallel machines.

The problems are generated by varying the number of machines (m) from 2 to 10 with an increment of 1 and the number of jobs from 11 to 25 with an increment of 1. The speed ratio of the machines is assumed as the ratio of the machine numbers. If a problem has four machines, the speed ratio of the machines 1, 2, 3, and 4 is 1:2:3:4, respectively.

The problem sizes are 2 × 11, 2 × 12, 2 × 13, ···, 2 × 25, 3 × 11, 3 × 12, 3 × 13, ···, 3 × 25, ···, 10 × 11, 10 × 12, 10 × 13, ···, 10 × 25. The total number of problem sizes is 135.

For each combination of the factors, two replications have been carried out. So, 270 problems (135 problems sizes with two replications in each problem size) were generated randomly as per the layout shown in **Table 4**.

The values of the makespan of the problems under each experimental combination are obtained. The formula to compute the percent deviation of the makespan of a problem from the minimum makespan of that problem is given by the following formula.

$$\text{Percentage deviation of makespan for a given experimental condition in a replication of a problem} = \left\{\frac{\text{Makespan for given experimental condition} - \text{Minimum of the four Makespan values in the replication}}{\text{Minimum of the four Makespan values in the replication}}\right\}\times 100$$

The respective ANOVA model [21] is presented below.

$$Y_{ijkl} = \mu + A_i + B_j + AB_{ij} + C_k + AC_{ik} + BC_{jk} + ABC_{ijk} + e_{ijkl}$$

where, Y_{ijkl} is the percentage deviation of makespan w.r.t. l^{th} replication under i^{th} problem size, j^{th} crossover method and k^{th} job allocation method.

μ is the overall mean of the percent deviation of the makespan values.

A_i is the effect of the i^{th} problem size on the percent deviation of the makespan value.

B_j is the effect of the j^{th} crossover method on the percent deviation of the makespan value.

AB_{ij} is the interaction effect of the i^{th} problem size and j^{th} crossover method on the percent deviation of the makespan value.

C_k is the effect of k^{th} job allocation method on the percent deviation of the makespan value.

AC_{ik} is the interaction effect of the i^{th} problem size and k^{th} job allocation method on the percent deviation of the makespan value.

BC_{jk} is the interaction effect of the j^{th} crossover method and k^{th} job allocation method on the percent deviation of the makespan value.

ABC_{ijk} is the interaction effect of the i^{th} problem size, j^{th} crossover method and k^{th} job allocation method on the percent deviation of the makespan value.

e_{ijkl} is the random error associated with the l^{th} replication under i^{th} problem size, j^{th} crossover method and k^{th} job allocation method.

The different hypotheses of this model are listed below.

Factor: Problem Size (A)

H_0: There is no significant difference between problem sizes in terms of the percent deviation of makespan value.

H_1: There is significant difference between problem sizes in terms of the percent deviation of makespan value.

Factor: Crossover Method (B)

H_0: There is no significant difference between crossover methods in terms of the percent deviation of makespan value.

H_1: There is significant difference between crossover

Table 4. Layout of problem generation.

		Crossover Method (**B**)			
		Single Point		Two Point	
		Method of Allocation of Jobs (**C**)		Method of Allocation of Jobs (**C**)	
		Equal Allocation	Proportionate Allocation	Equal Allocation	Proportionate Allocation
Problem Size (**A**)	2 × 11				
	2 × 12				
	2 × 13				
	.				
	.				
	2 × 25				
	3 × 11				
	3 × 12				
	3 × 13				
	.				
	.				
	3 × 25				
	.				
	.				
	10 × 11				
	10 × 12				
	10 × 13				
	.				
	.				
	10 × 25				

methods in terms of the percent deviation of makespan value.

Interaction: Problem Size (A) × Crossover Method (B)

H_0: There is no significant difference between different pairs of interaction terms of problem size and crossover method in terms of the percent deviation of makespan value.

H_1: There is significant difference between different pairs of interaction terms of problem size and crossover method in terms of the percent deviation of makespan value.

Factor: Job Allocation Method (C)

H_0: There is no significant difference between job allocation methods in terms of the percent deviation of makespan value.

H_1: There is significant difference between job allocation methods in terms of the percent deviation of makespan value.

Interaction: Problem Size (*A*) × Job Allocation Method (C)

H_0: There is no significant difference between different pairs of interaction terms of problem size and job allocation method in terms of the percent deviation of makespan value.

H_1: There is significant difference between different pairs of interaction terms of problem size and job allocation method in terms of the percent deviation of makespan value.

Interaction: Crossover Method (B) × Job Allocation Method (C)

H_0: There is no significant difference between different pairs of interaction terms of crossover method and job allocation method in terms of the percent deviation of makespan value.

H_1: There is significant difference between different pairs of interaction terms of crossover method and job allocation method in terms of the percent deviation of makespan value.

Interaction: Problem Size (A) × Crossover method (B) × Job Allocation Method (C)

H_0: There is no significant difference between different combinations of interaction terms of problem size, crossover method and job allocation method in terms of the percent deviation of makespan value.

H_1: There is significant difference between different combinations of problem size, crossover method and job allocation method in terms of the percent deviation of makespan value.

The results of the corresponding ANOVA model are shown in **Table 5**.

The hypotheses for which the effects are significant are as listed below.

Factor "Problem Size (A)"

In the **Table 5**, the calculated F ratio for the factor "Problem Size (A)" is 4.65, which is more than the corresponding table F value of 1 for (134, 540) degrees of freedom at a significance level of 0.05. Hence, the alternate hypothesis is accepted. This means that *there is significant difference between the problem sizes* in terms of percent deviation of makespan.

Factor "Job Allocation Method" (C)

In the **Table 5**, the calculated F ratio for the factor, "Job Allocation Method (*C*)" is 546.169, which is more than the table F value of 3.84 for (1, 540) degrees of freedom at a significance level of 0.05. Hence, the corresponding alternate hypothesis is accepted. This means that there is significant difference between the job allocation methods in terms of percent deviation of makespan.

Interaction "Problem Size × Job Allocation Method" (A × C)

In the **Table 5**, the calculated F ratio for the interaction "*Problem Size* × *Job Allocation Method*" is 2.72,

Table 5. Results of ANOVA.

Source of Variation	Sum of squares	Degrees of freedom	Mean sum of squares	$F_{Calculated}$	$F_{table\ at\ \alpha = 0.05}$	Inference
A (Problem Size)	62602.35000	134	467.190	4.650	1.00	**Significant**
B (Crossover Method)	72.96875	1	72.969	0.726	3.84	Insignificant
AB	9760.92200	134	72.8430	0.725	1.00	Insignificant
C(Job Allocation Method)	54869.26000	1	54869.260	546.169	3.84	**Significant**
AC	36622.24000	134	273.300	2.720	1.00	**Significant**
BC	9.0156000	1	9.106	0.0897	3.84	Insignificant
ABC	9056.37500	134	67.584	0.673	1.00	Insignificant
Error	54249.32000	540	100.462			
Total	227242.50000	1079				

which is more than the table F value of 1 for (134, 540) degrees of freedom at a significance level of 0.05. Hence, the corresponding alternate hypothesis is accepted. This means that there is significant difference between the interaction terms, A_1C_1, A_1C_2, A_2C_1, A_2C_2, ⋯, $A_{135}C_1$, $A_{135}C_2$ in terms of makespan.

Since, there is significant difference between the job allocation methods (Equal number of jobs allocation and proportionate number of jobs allocation), the best method of allocation of the jobs to the machines can be based on the least mean percentage deviation of the makespan values. The mean percentage deviation of the makespans of the equal number of allocation of jobs to the machines and the proportionate number of allocation of jobs to the machines are 17.70365 and 2.26711, respectively. Since, the mean percentage deviation of the proportionate number of allocation of jobs to the machines is less when compared to that of the equal number of allocation of jobs to the *machines*, *the method of proportionate number of allocation of jobs to the machines is the best.*

It is observed that there is no significant difference between the single point crossover method and two point crossover method in terms of the percentage deviation of the makespan. So, any of these two crossover methods can be selected for implementation.

However, the selection is done based on the least mean percent deviation of the makespan values of "Single Point Crossover Method with Proportionate Allocation of Jobs" and "Two Point Crossover Method with Proportionate Allocation of Jobs". The mean percentage deviation of the makespan of the "Single Point Crossover Method with Proportionate Allocation of Jobs" is 2.2457 and that of the "Two Point Crossover Method with Proportionate Allocation of Jobs" is 1.8823. Since, the mean percentage deviation of the makespan of the "Two Point Crossover Method with Proportionate Allocation of Jobs" is less than that of the "Single Point Crossover Method with Proportionate Allocation of Jobs", the GA based heuristic with two point crossover method with proportionate allocation of jobs is the best.

Based on the above discussions, it is recommended to use the GA based heuristic with two point crossover method, in which the initial population is generated by way of proportionate allocation of jobs to the machines based on the machine speeds.

6. Conclusions

Production scheduling plays a vital role in industries. The single machine scheduling problem with uniform parallel machines which is presented in this paper is a special kind of parallel machines scheduling problem. The minimization of the makespan is considered as the objective of the problem. This problem comes under combinatorial category. Hence, the development of a meta-heuristic is inevitable for better solution. Among the meta-heuristics, very little work has been done on GA based heuristic to minimize the makespan for this problem. Hence, in this paper, four different GA based heuristics have been designed by combining the two crossover methods (single point crossover method and two point crossover method) and the two job allocation methods (equal number of allocation of jobs to machines and proportionate number of allocation of jobs to machines based on machine speeds).

To select the best GA based heuristic, a comprehensive three factor ANOVA experiment has been conducted by assuming the factors as "Problem Size (A)", "Crossover Method (B)" and "Job Allocation Method (C)". The problem sizes are 2 × 11, 2 × 12, 2 × 13, ⋯, 2 × 25, 3 × 11, 3 × 12, 3 × 13, ⋯, 3 × 25, ⋯, 10 × 11, 10 × 12, 10 × 13, ⋯, 10 × 25. The total number of problem sizes is 135 and each problem size is with two replications. So, in total 270 problems are solved using four different GA based heuristics to obtain their corresponding makespan values. Then the percent deviation of each makespan is obtained from the minimum makespan of each replication of each problem size to draw inferences using ANOVA.

From ANOVA, the following are observed.

- There is significant difference between the problem sizes in terms of percent deviation of makespan.
- There is significant difference between the job allocation methods in terms of percent deviation of makespan,
- There is significant difference between the interaction terms, A_1C_1, A_1C_2, A_2C_1, A_2C_2, A_3C_1, A_3C_2, ⋯, $A_{135}C_1$, $A_{135}C_2$ in terms of percent deviation of makespan.

Since, there is significant difference between the job allocation methods, and the mean percentage deviation of the proportionate number of allocation of jobs to the machines (2.064055) is less when compared to that of the equal number of allocation of jobs to the machines (17.01832), the method of proportionate number of allocation of jobs to the machines is the best.

Further, it is observed that there is significant difference between different problem sizes in terms of mean percent deviation of makespan as well as between different combinations of problem size and job allocation method in terms of percent deviation of makespan. These two facts further support the strong significant difference between the job allocation methods in terms of percent deviation of makespan.

It is observed that there is no significant difference

between the single point crossover method and two point crossover method in terms of the percentage deviation of the makespan. So, any of these two crossover methods can be selected for implementation. However, the two point crossover method is selected because of its reduced mean percent deviation of the makespan values.

Based on the above discussions, it is recommended to use the GA based heuristic with two point crossover method, in which the initial population is generated by way of proportionate allocation of jobs to the machines based on the machine speeds.

The research reported in this paper is a significant contribution under the single machine scheduling problem with uniform parallel machines in terms of designing a meta-heuristic to minimize the makespan. Future researches may be focused on uniform parallel machines scheduling with stochastic processing times for the jobs.

7. References

[1] R. Panneerselvam, "Production and Operations Management," 2nd Edition, PHI Learning Private Limited, New Delhi, 2005.

[2] P. Senthilkumar and S. Narayanan, "Literature Review of Single Machine Scheduling Problem with Uniform Parallel Machines," *Intelligent Information Management*, Vol. 2, No. 8, 2010, pp. 457-474. doi:10.4236/iim.2010.28056

[3] P. De and T. E. Morton, "Scheduling to Minimize Makespan on Unequal Parallel Processors," *Decision Sciences*, Vol. 11, No. 4, 1980, pp. 586-602. doi:10.1111/j.1540-5915.1980.tb01163.x

[4] R. L. Bulfin and R. G. Parker, "Scheduling Jobs on Two Facilities to Minimize Makespan," *Management Science*, Vol. 26, No. 2, 1980, pp. 202-214. doi:10.1287/mnsc.26.2.202

[5] D. K. Friesen and M. A. Langston, "Bounds for Multifit Scheduling on Uniform Processors," *SIAM Journal on Computing*, Vol. 12, No. 1, 1983, pp. 60-70. doi:10.1137/0212004

[6] R. L. Graham, "Bounds for Multiprocessing Timing Anomalies," *SIAM Journal of Applied Mathematics*, Vol. 17, No. 2, 1969, pp.416-429. doi:10.1137/0117039

[7] G. Dobson, "Scheduling Independent Tasks on Uniform Processors," *SIAM Journal of Computing*, Vol. 13, No. 4, 1984, pp. 705-716. doi:10.1137/0213044

[8] D. K. Friesen, "Tighter Bounds for LPT Scheduling on Uniform Processors," *SIAM Journal on Computing*, Vol. 16, No. 3, 1987, pp. 554-560. doi:10.1137/0216037

[9] D. S. Hochbaum and D. B. Shmoys, "A Polynomial Approximation Scheme for Scheduling on Uniform Processors: Using the Dual Approximation Approach," *SIAM Journal on Computing*, Vol. 17, No. 3, 1988, pp. 539-551. doi:10.1137/0217033

[10] B. Chen, "Parametric Bounds for LPT Scheduling on Uniform Processors," *Acta Mathematicae Applicateae Sinica*, Vol. 7, No. 1, 1991, pp. 67-73. doi:10.1007/BF02080204

[11] P. Mireault, J. B. Orlin and R. V. Vohra, "A Parametric Worst Case Analysis of the LPT Heuristic for Two Uniform Machines," *Operations Research*, Vol. 45, No. 1, 1997, pp. 116-125. doi:10.1287/opre.45.1.116

[12] R. E. Burkard and Y. He, "A Note on MULTIFIT Scheduling for Uniform Machines," *Computing*, Vol. 61, No. 3, 1998, pp. 277-283. doi:10.1007/BF02684354

[13] R. E. Burkard, Y. He and H. Kellerer, "A Linear Compound Algorithm for Uniform Machine Scheduling," *Computing*, Vol. 61, No. 1, 1998, pp. 1-9.

[14] R. Panneerselvam and S. Kanagalingam, "Modelling Parallel Processors with Different Processing Speeds of Single Machine Scheduling Problem to Minimize Makespan," *Industrial Engineering Journal*, Vol. 17, No. 6, 1998, pp. 16-19.

[15] R. Panneerselvam and S. Kanagalingam, "Simple Heuristic for Single Machine Scheduling Problem with Two Parallel Processors Having Varying Speeds to Minimize Makespan," *Industrial Engineering Journal*, Vol. 18, No. 6, 1999, pp. 2-8.

[16] A. Agarwal, S. Colak, V. S. Jacob and H. Pirkul, "Heuristics and Augmented Neural Networks for Task Scheduling with Non-identical Machines," *European Journal of Operational Research*, Vol. 175, No. 1, 2006, pp. 296-317. doi:10.1016/j.ejor.2005.03.045

[17] P. Senthilkumar and S. Narayanan, "Simulated Annealing Algorithm to Minimize Makespan in Single Machine Scheduling Problem with Uniform Parallel Machines," *Intelligent Information Management*, Vol. 3, No. 1, 2011, pp. 22-31. doi:10.4236/iim.2011.31003

[18] C. Mihaila and A. Mihaila, "An Evolutionary Algorithm for Uniform Parallel Machines Scheduling," 2*nd UKSIM European Symposium on Computer Modelling and Simulation*, Liverpool, 8-10 September 2008, pp. 76-80. doi:10.1109/EMS.2008.34

[19] A. Mihaila and C. Mihaila, "Uniform Parallel Machines Scheduling Using a Genetic Algorithm," 8*th International Conference on Intelligent Systems Design and Applications*, Kaohsiung, 26-28 November 2008, pp. 401-406.

[20] R. Panneerselvam, "Design and Analysis of Algorithms," PHI Learning Private Limited, New Delhi, 2007.

[21] R. Panneerselvam, "Research Methodology," PHI Learning Private Limited, New Delhi, 2004.

