# Determination of Job Scheduling for a Flexible Flow Shop with Unrelated Parallel Machines and Non-Uniform Performance

Wimonsiri Prapatigul
*Master of Engineering Technology, Engineering Faculty*
*Thai-Nichi Institute of Technology*
Bangkok, Thailand
pr.wimonsiri_st@tni.ac.th

Anchalee Supithak
*Industrial Engineering, Engineering Faculty*
*Thai-Nichi Institute of Technology*
Bangkok, Thailand
anchalee_s@tni.ac.th

***Abstract*— This research aims to study and develop a heuristic method to solve for the job scheduling of a flexible flow shop with unrelated parallel machine and nonuniform performance. The flexible flow shop with unrelated parallel machine represents the general manufacturing in the industry. There are at least two stages of process. Each processing stage in the flexible flow shop contains at least two difference performance machines. The performance difference of machines in each stage is not uniform. This research focuses on generating the job processing time from a uniform distribution. The heuristic based on the genetic algorithm is developed with the minimization of the makespan value from the job scheduling. The job processing time of each machine is randomly generated from the uniform distribution to represent the non-uniform performance machine. The solution from the heuristic is compared to the Brute Force method by measuring the deviation of makespan from the developed heuristics and the Brute Force method. The results show that, for the problem of six jobs and three stages which each stage contains two difference performance machines, the developed heuristics give approximal the same solution as the solution from brute force method. From the experiment of six jobs, the developed heuristic gives the same value of makespan as the Brute Force method for 8 from 10 instances. The developed heuristic is proved with solving the large problems of 20 jobs of 10 replications, with two minutes computational time. The developed heuristic gives the lower makespan value than the brute force method around 5.5%.**

***Keywords—Flexible Flow Shop, Genetic Algorithm, Job Scheduling, Makespan, Non-Uniform Performance Machines, Unrelated Parallel Machines)***

## I. INTRODUCTION

The Flexible Flow Shop or FFS represent the real manufacturing process which composes of a series of processes or stages. Flexible Flow Shop is also known as Hybrid Flow Shop which each stage has more than one machine. The Flexible Flow Shop with unrelated parallel machines problem has been studied wildly, since it represents the real manufacturing shop floor. The order scheduling is complicated when each stage involves machines with difference performance. In the real situation, the FFS with non-identical parallel and non-uniform performance machines is interesting and complicated to find the job scheduling solution. The jobs scheduling in FFS with unrelated parallel machines problem is considered as NP hard problem [1]. Most research focuses on solving for the schedule with the minimization of average flow time or makespan and meeting due date. However, jobs scheduling decision in FFS with unrelated parallel machines is complicated and becomes combinatorial explosion problem since the makespan depends on the job sequence and the selected machine in each stage which effect processing time. Thus, the Enumeration method or Branch-and-Bound may not be able to find the optimal solution within the practical period [2]. The Genetic Algorithm or GA is wildly applied in research to solve the variance FFS problems. The study of [3] illustrates the significance of the FFS unrelated parallel machine problem which represents the realistic manufacturing such as textile process. [3] considers the sequence dependent setup time in the problem with the starting date and due date of job. The constructive heuristic with Genetic Algorithm is developed to solve for the job schedule that minimizes makespan and the number of tardy jobs. The research of [4] study Flexible Job Shop with unrelated parallel machine and resources-dependent processing time which is different from FFS only the workflow route. The study of [5] also focuses on solving FFS unrelated parallel machines as the dual objective function of trading off between the makespan and the production cost by applying the GA based heuristic. According to the literature study, the FFS unrelated parallel machine problem is wildly study since it represents the realistic manufacturing. Most research apply the heuristic which mostly based on GA [2], [3], [5] and other heuristic such as Tabu Search [4]. The Genetic Algorithm is wildly applied in solving complex problem [6], [7]. However, the FFS unrelated parallel machine problem in the shop floor involves non-uniform performance machines which resulting in the more complicate to schedule the jobs especially when there are many jobs to proceed.

The key characteristics of non-uniform performance machine in this research is the machine-job dependency. The processing time of job i on machine k is unique to that specific pair. A machine may be faster than the other and this difference speed is not in constant proportion. This characteristic makes the problem more complex. The problem of flexible flow shop with unrelated parallel machine and non-uniform performance is solved by develop the heuristic based on Genetic Algorithm. This research focuses on solving the Flexible Flow Shop of three stages and each stage contains parallel machines which are difference in speed or capacity. Moreover, the machines have non-uniform performance. The machine processing time is difference for difference job and difference machine. For the same stage, the machine speed is not proportional difference since machine's technology may be difference. This research considers the job processing time that is generated form a uniform distribution. The heuristic method is developed based on Genetic Algorithm. Finally, the validation of the developed heuristic is performed by the experiment of small problems and compared with the Brute Force method.

## II. RESEARCH METHODOLOGY

### A. *Problem Description and Assumptions*

This research considers three stages flexible flow shop and each stage contain two machines which are different in performance. One machine is lower speed than the other machine. The problem is illustrated by five jobs which are proceed by three stages in series. The job processing time is random variable and is generated form the uniform distribution. The characteristic of non-uniform performance of machines in this research is the machine dependent processing time. The processing time of job *i* on machine *k* is unique to that specific pair. The developed heuristic is proposed to solve the problem as the processes in figure 1.

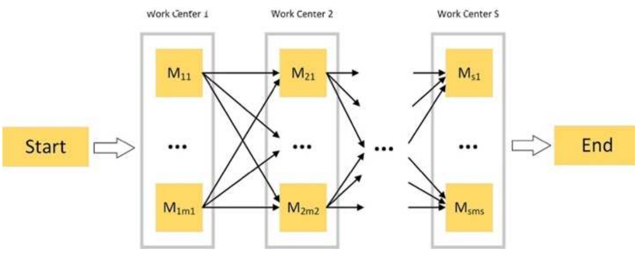



Fig. 1. The flexible flow shop and unrelated parrallel machines problem

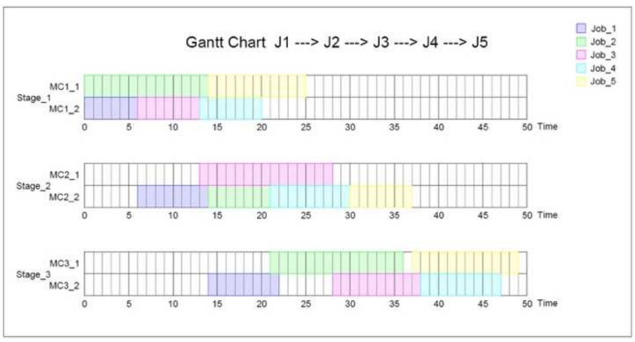



Fig. 2. The example of grant chart for five jobs of the Ffexible flow shop unrelated parallel machine problem

The related assumptions are as follow:

*1)* The job processing time is constant and generated from the uniform distribution.

*2)* The job processing time is independent from the job schedule.

*3)* The machine setup time is not considered.

*4)* The parallel machines are not identical. One machine is faster than the other machine and the machine performance difference is not the same proportion for different job.

*5)* Non-preemptive scheduling

The related notations are as follow:

$J_i$ is job i

$M_{k,j}$ is machine k at stage j

$j$ is processing stage index

$P_{i,j,k}$ is processing time of job i at stage j of machine k

$\bar{F}_i$ is flow time average of job i

$C_i$ is completion time of job i

$C_{max}$ is the maximum completion time

$S_{i,j,k}$ is starting time of job i at stage j on machine k

$C_{i,j,k}$ is completion time of job i at stage j on machine k

$x_{i,j,k}$ is 1 if job i is proceed on stage j on machine k, otherwise is 0

The objective function is to minimize $C_{max}$ as shown in equation (1). Equation 2 represents average flow time and equation 3 illustrates the relationship of completion time, starting time and processing time.

$$C_{max} = Maximum[C_i] \tag{1}$$

$$\bar{F}_i = \frac{1}{n}\sum_{i-1}^{n} F_i \tag{2}$$

$$C_{i,j,k} = S_{i,j,k} + P_{i,j,k} \tag{3}$$

### B. *Instance Generation and Data Structure*

The processing time of job i on machine k is generated from a uniform distribution. The research problem involves two parallel machines in each stage. One machine is higher performance than the other to represent the unrelated parallel machines and non-uniform performance. The job processing times on both machines are generated from the uniform distribution of [5,10] and [11-15] for the high-speed machine and low-speed machine, respectively. The ten instances of six jobs are generated. The developed heuristic based on genetic algorithm is applied to solve the ten instances. The solutions from the developed heuristic are compared to the results from the brute force method.

### *C. The Genetic Algorithm Configuration*

The production scheduling problem in a Flexible Flow Shop (FFS) with unrelated parallel machine system is highly complex and classified as NP-hard. Since finding an absolute global optimum via Brute Force becomes computationally prohibitive as the number of jobs increases, this research employs a Genetic Algorithm (GA). GA is a metaheuristic search algorithm capable of finding high-quality solutions within a reasonable computational timeframe. The GA mimics the process of natural evolution based on the principle of "Survival of the Fittest" through three core operators: Selection, Crossover, and Mutation. The goal is to evolve a population of solutions toward an optimal state over successive generation. The heuristic is developed based on GA to search for the job schedule that minimize makespan and average flow time according to figure 3.

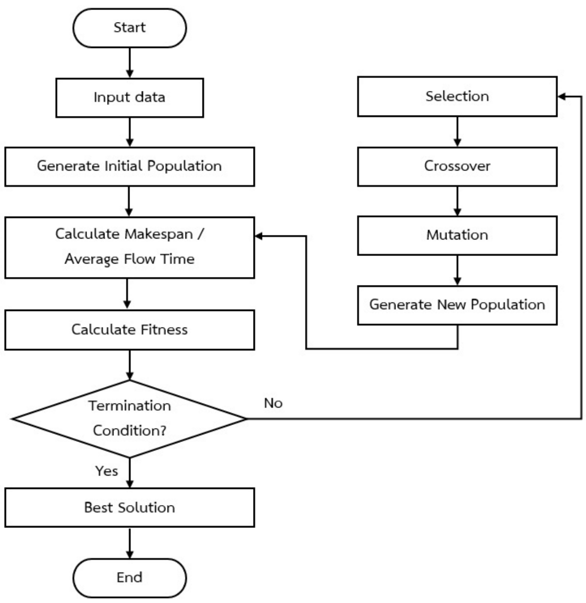



Fig. 3. Flow Chart of the Developed Heuristic Based on Genetic Algorithm

The developed heuristic is designed as follow.

**Step 1** Generating Instances

For each instance, generate job processing time of every job and every stage. For the low performance and high performance machine in each stage, the job processing time is randomly generated from the uniform distribution of [11, 15] and [5, 10] respectively.

**Step 2** Chromosome Representation

In this study, a chromosome is represented as a job sequence, which determines the order in which jobs enter the production system such as the following example.

[J3, J1, J5, J2, J4]

Each position in the chromosome is a Gene, representing a specific job. To assign machines at each stage, the Fastest Available Machine rule is applied: the system selects the machine that can complete the job at the earliest possible time among all available machines. If all machines in the considering stage are not available, the job has to wait for the machine.

**Step 3** Initial Population

The initial population is generated by randomly shuffling the job sequences. Each chromosome must contain a complete set of jobs without duplicates. Random initialization ensures high population diversity, which is crucial for exploring the solution space effectively.

**Step 4** Fitness Function

The fitness function evaluates the quality of each chromosome. In this research, fitness is inversely proportional to the Makespan.

$$Fitness = \frac{1}{C_{max}}$$

Under this definition, chromosomes with a lower Makespan yield a higher Fitness value, granting them a higher probability of being selected for the next generation.

**Step 5** Selection Process

This study utilizes Tournament Selection. A subset of chromosomes is randomly chosen from the current population, and the one with the highest fitness is selected as a "parent" for reproduction. This method ensures that superior traits are passed down while maintaining competitive pressure.

**Step 6** Crossover Operator

To create offspring, the Order Crossover (OX) method is used. OX is specifically designed for permutation-based problems. It preserves the relative order of a segment of genes from one parent while filling the remaining slots with genes from the second parent in the order they appear, ensuring no jobs are duplicated.

**Step 7** Mutation Operator

Mutation prevents the algorithm from converging prematurely on a Local Optimum. The Swap Mutation method is applied here: two random genes (jobs) within a chromosome are selected and their positions are swapped. For example, the position of J4 and J5 is swapped.

Before Mutation: [J2, J4, J1, J5, J3]

After Mutation: [J2, J5, J1, J4, J3]

**Step 8** Termination Conditions and Parameters

The algorithm continues to evolve the population until the following predefined parameters are met:

- Population Size: 50
- Maximum Generations: 200
- Crossover Rate: 0.9 (90%)
- Mutation Rate: 0.1 (10%)

The ten instances of six jobs problem are solved by the developed heuristic and the brute force method. The solutions of both methods are compared to validate the performance of the heuristic with GA based

## III. Research Result

### A. Experimental Design

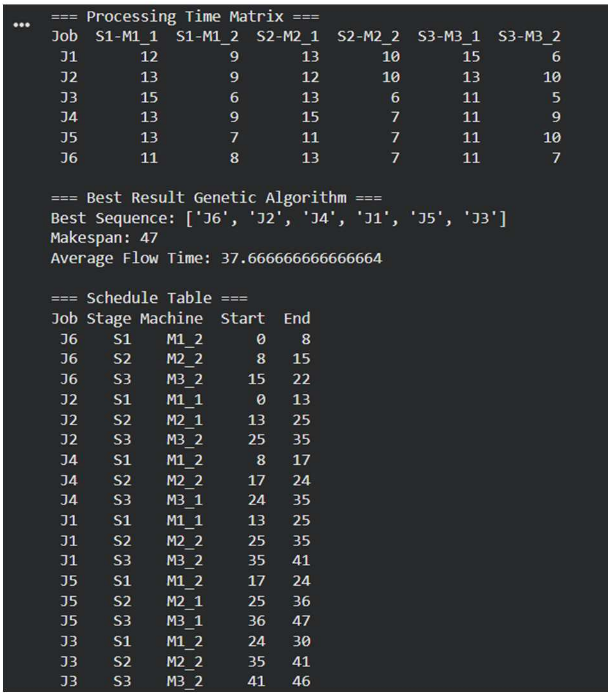
The job processing time of the ten instances of six jobs problem is generated from the uniform distribution of [11, 15] and [5, 10] for machine 1 and machine 2, respectively. The example an instance is shown in table 1.

TABLE I. THE EXAMPLE OF GENERATED PROCESSING TIME OF INSTANCE1

| **Job** | **Stage1 M1_1** | **Stage1 M1_2** | **Stage2 M2_1** | **Stage2 M2_2** | **Stage3 M3_1** | **Stage3 M3_2** |
|---|---|---|---|---|---|---|
| J1 | 12 | 9 | 13 | 10 | 15 | 6 |
| J2 | 13 | 9 | 12 | 10 | 13 | 10 |
| J3 | 15 | 6 | 13 | 6 | 11 | 5 |
| J4 | 13 | 9 | 15 | 7 | 11 | 9 |
| J5 | 13 | 7 | 11 | 7 | 11 | 10 |
| J6 | 11 | 8 | 13 | 7 | 11 | 7 |

Note: M1_1 denotes stage1 with machine1.

From the data in table I, the best job scheduling result from the developed heuristic based on GA is illustrated in figure 4.



```
=== Processing Time Matrix ===
Job  S1-M1_1  S1-M1_2  S2-M2_1  S2-M2_2  S3-M3_1  S3-M3_2
 J1       12        9       13       10       15        6
 J2       13        9       12       10       13       10
 J3       15        6       13        6       11        5
 J4       13        9       15        7       11        9
 J5       13        7       11        7       11       10
 J6       11        8       13        7       11        7

=== Best Result Genetic Algorithm ===
Best Sequence: ['J6', 'J2', 'J4', 'J1', 'J5', 'J3']
Makespan: 47
Average Flow Time: 37.666666666666664

=== Schedule Table ===
Job Stage Machine  Start  End
 J6    S1    M1_2      0    8
 J6    S2    M2_2      8   15
 J6    S3    M3_2     15   22
 J2    S1    M1_1      0   13
 J2    S2    M2_1     13   25
 J2    S3    M3_2     25   35
 J4    S1    M1_2      8   17
 J4    S2    M2_2     17   24
 J4    S3    M3_1     24   35
 J1    S1    M1_1     13   25
 J1    S2    M2_2     25   35
 J1    S3    M3_2     35   41
 J5    S1    M1_2     17   24
 J5    S2    M2_1     25   36
 J5    S3    M3_1     36   47
 J3    S1    M1_2     24   30
 J3    S2    M2_2     35   41
 J3    S3    M3_2     41   46
```

Fig. 4. The Computational Result of the Example in Table 1

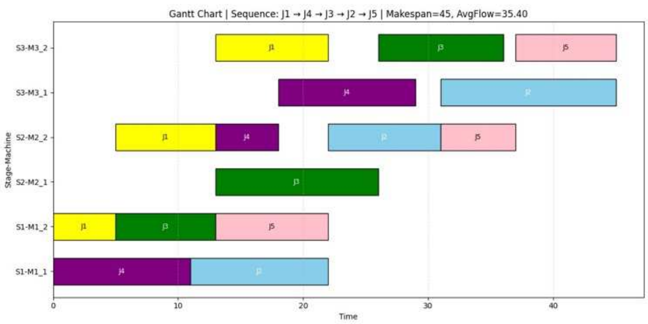



Fig. 5. The Grant Chart for the Example of Six Jobs

### B. Experimental Results

The computational results from running ten instances of six jobs by the developed heuristic based on GA comparing to the Brute force method are illustrated in table II.

TABLE II. THE COMPUTATIONAL RESULTS OF TEN INSTANCES FROM DEVELOPED HEURISTIC (GA) AND BRUTE FORCE

| | Makespan | | Average Flow Time | |
|---|---|---|---|---|
| Instance | GA | Brute Force | GA | Brute Force |
| 1 | 46 | 46 | 34.6 | 34.7 |
| 2 | 47 | 47 | 34.8 | 35.5 |
| 3 | 47 | 47 | 36.5 | 36.2 |
| 4 | 49 | 47 | 36.6 | 37.2 |
| 5 | 48 | 48 | 35.2 | 36.2 |
| 6 | 46 | 49 | 39.5 | 39.5 |
| 7 | 48 | 48 | 36.5 | 37.5 |
| 8 | 47 | 47 | 35.8 | 35.8 |
| 9 | 47 | 47 | 36.2 | 37.7 |
| 10 | 47 | 47 | 37.6 | 38.2 |

From table II, the makespan values from the developed heuristic (GA) are equal to the Brute Force method for 8 instances from 10 instances. The results confirm that the developed heuristic can practically solve the flexible flow shop unrelated parallel machine with non-uniform performance. The limitation of the Brute Force method is finding an absolute global optimum becomes computationally prohibitive as the number of jobs increases. On the other hand, the GA based heuristic is practical to solve the problem.

### C. The Results for Large Problem

The developed heuristic is applied to simulate and solve problems with 20 jobs. The job processing time is randomly generated from the uniform distribution of [11, 15] and [5, 10] for machine 1 and machine 2, respectively. The experiment is performed for 10 instances by solving with the developed heuristic and the brute force method. Since, the large size problem is not practically solved by the brute force method under the acceptable computational running period, the computation stopping criteria is set to 2 minutes of searching the solution on the laptop with 12$^{th}$ Gen Intel® Core ™ i7-12700H (2.30 GHz), 16.0 GB RAM. The results are illustrated in table III.

TABLE III. THE SIMULATION RESULTS OF IMPLEMENTING DEVELOPED HEURISTICS TO LARGE PROBLEMS

| **Case** | **GA Makespan** | **Brute Force Makespan** | **GA Avg Flow Time** | **BF Avg Flow Time** |
|---|---|---|---|---|
| 1 | 106 | 107 | 66.55 | 64.25 |
| 2 | 105 | 115 | 66.2 | 71.8 |
| 3 | 112 | 117 | 71 | 73.6 |
| 4 | 106 | 113 | 67.45 | 72.5 |
| 5 | 106 | 113 | 67.25 | 71.85 |
| 6 | 108 | 115 | 69.55 | 74.25 |
| 7 | 105 | 113 | 67.3 | 72.65 |
| 8 | 111 | 117 | 70 | 73.7 |
| 9 | 104 | 110 | 64.9 | 69.4 |
| 10 | 111 | 117 | 71.3 | 73.15 |
| Average | 107.4 | 113.7 | 68.15 | 71.715 |

The developed heuristic can solve the large problems. For ten instances and two minutes of computational period, the average makespan values of the schedule from the developed heuristic and the brute force method are 107.4 and 113.7, respectively. Under the 20 jobs scenario with 10 replicates, the developed heuristic gives the lower makespan value than the brute force method around 5.5%.

## IV. CONCLUSION

The developed Genetic Algorithm (GA) heuristic presents a highly effective and practical solution for scheduling within flexible flow shops utilizing unrelated parallel machines with non-uniform performance. Testing demonstrated that the GA is highly accurate, matching the absolute global optimum (established by the Brute Force method) in 80% of the evaluated instances. Crucially, while Brute Force methods guarantee an optimal solution, they suffer from severe scalability issues, becoming computationally prohibitive as the number of jobs increases. The results from 20 jobs simulation under the two minutes computational time confirm that the developed heuristic give the lower makespan value around 5 percent. Therefore, the GA-based heuristic successfully overcomes this limitation. It provides an optimal balance between high-quality results (makespan minimization) and computational efficiency, making it a viable and superior approach for solving larger, real-world scheduling problems where brute-force calculation is impossible.

## REFERENCES

[1] J. Blazewicz, K. H. Ecker, E. Pesch, G. Schmidt, and J. Weglarz, Scheduling Computer and Manufacturing Processes, 2nd ed. Berlin, Germany: Springer, 2001

[2] E. Taillard, “Benchmarks for basic scheduling problems,” European Journal of Operational Research, vol. 64, no. 2, pp. 278–285, 1993.

[3] J. Jungwattanakit, M. Reodecha, P. Chaovalitwongse, and F. Werner, “Algorithms for Flexible Flow Shop Problems with Unrelated Parallel Machines, Setup Times, and Dual Criteria,” The international Journal of Advanced Manufacturing Technology, vol. 37, pp. 354-370, 2008.

[4] M. Hajibabaei and J. Behnamian, “Flexible job-shop scheduling problem with unrelated parallel machines and resources-dependent processing times: a tabu search algorithm,” International Journal of Management Science and Engineering Management, vol. 16, Issue 4, pp. 242-253, 2021.

[5] A. Hasani, S. M. H. Hosseini, and S. S. Sana, “Scheduling in a flexible flow shop with unrelated parallel machines and machine-dependent process stages: Trade-off between Makespan and production costs,” Sustainable Analytic and Modeling,

[6] A. Hassanat et al., “Choosing mutation and crossover ratios for genetic algorithms: A review,” Information, vol. 10, no. 12, 2019.

[7] D. E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning. Reading, MA, USA: Addison-Wesley, 1989.