World Journal of Advanced Research and Reviews
eISSN: 2581-9615 CODEN (USA): WJARAI
Cross Ref DOI: 10.30574/wjarr
Journal homepage: https://wjarr.com/

(RESEARCH ARTICLE)

Check for updates

Comparative Analysis of Heuristic Rules for Flow Shop Scheduling Using ANOVA

Sundus Alfaitouri *

Department of Industrial and Manufacturing System Engineering, Faculty of Engineering, University of Benghazi, Benghazi, Libya.

World Journal of Advanced Research and Reviews, 2025, 27(02), 2084-2094

Publication history: Received on 18 July 2025; revised on 26 August 2025; accepted on 28 August 2025

Article DOI: https://doi.org/10.30574/wjarr.2025.27.2.3048

Abstract

This paper compares the performance of five heuristic rules with the objective of minimizing the makespan time. The selected heuristics are namely SAI, Palmer, Gupta, RA and CDS. The heuristics are tested on twelve flow shop problems of different sizes using a MATLAB program. The comparison was made by using the Analysis of Variance (ANOVA) under Completely Randomized Block Design (CRBD), where the problem size was considered as a blocking factor and the heuristic rule as the main factor. Makespan time was used as the response variable, with a significance level of $\alpha = 0.05$. The findings indicated significant differences between the CDS, SAI, and Palmer heuristics, as the mean makespan achieved by the CDS heuristic is lower than that of the SAI and Palmer heuristics, while its performance was not significantly different from that of RA and Gupta.

Keywords: Flow shop scheduling; Heuristics; Makespan; ANOVA

1. Introduction

Production Scheduling is a fundamental decision-making process in manufacturing and operations management. It involves allocating available resources (machines, workers, materials) over time to perform a collection of tasks (jobs, operations) to optimize one or more objectives while satisfying constraints [1] .

Scheduling problems are characterized by machine layouts: single machine, identical parallel machines, flow shops and job shops [2].

Flow Shop Scheduling (FSS) represents one of the most important and widely studied machine environments within production scheduling. It models production systems where a set of n jobs must be processed sequentially on a set of m machines (or stages) in the same technological order [3] .

Generally, there are two classes of methods for solving flow shop scheduling problem (FSSP); exact and heuristic methods. The first-class methods include: integer programming, dynamic programming, branch and bound, etc. Another class is called heuristic algorithms such as Gupta, Palmer, CDS, RA, SAI, etc. Although those methods do not guarantee the finding of an optimal solution, those have been reported useful in solving many challenging optimization problems within a reasonable computational time.

The main objective of FSSP is to determine the best machine schedule to do all job with the best objective value, i.e., minimizing makespan (Cmax), mean flow time, mean tardiness, earliness, maximum lateness, etc.

For solving FSSP with the objective of minimizing makespan, several researchers have reported the robustness of heuristic methods. These include Gupta and Ali [4], Gupta and Chauhan [5], Syarif et. al. [3], Sule [6], Dannenbring [7],

* Corresponding author: Sundus Alfaitouri

Nawaz et al. [8], Gupta [9] and so on. Despite these interests, however, no researcher said the best method to solve FSSP all-time optimally. This fact shows that researches on the performance evaluation of the heuristic rules for FSSP are very crucial.

Previous research has consistently compared heuristic performance for flow shop scheduling. For instance, Hossain S. et al. [10] applied the Palmer, CDS, and NEH heuristics to a 4-job and 10-machine problem aimed at makespan minimization. Interestingly, all three methods yielded identical results for this specific instance. In a more targeted comparison of small-scale problems (5 jobs, 5 machines), Rayhan and Chakma [11] found that the NEH heuristic outperformed both Palmer's and CDS's methods by a margin of 3–5%. Scaling up the analysis, Gupta & Chauhan [5] developed a weightage-based heuristic tested on large-scale Taillard benchmarks (up to 500 jobs). Their approach reduced the average makespan gap to best-known solutions by 8%, demonstrating superior performance over the Palmer, CDS, and RA heuristics.

While these studies provide valuable pairwise comparisons, a significant gap remains in the application of robust statistical methods for multi-heuristic evaluation. This paper is motivated by the notable absence of Analysis of Variance (ANOVA) in this context. ANOVA offers a critical advantage: it efficiently manages comparisons across multiple heuristics while accounting for the inherent variability introduced by different problem instances. This methodology controls statistical error rates and, when combined with post-hoc tests, provides a rigorous foundation for identifying truly superior algorithms. We argue that this leads to more reliable, objective, and insightful conclusions on algorithmic performance, which are essential for both advancing research and guiding practical implementation.

1.1. Problem Statement

The assumptions regarding FSSP in this study are general and common in nature. The same are adapted from Baker [12], Ruiz and Maroto [13] and others.

- Each job i can be processed at most on one machine j at the same time.
- Each machine m can process only one job i at a time.
- No preemption is allowed, i.e. the processing of a job i on a machine j cannot be interrupted.
- All jobs are independent and are available for processing at time 0.
- The set-up times of the jobs on machines are already included within the processing times.
- The machines are continuously available.
- In-process inventory is allowed. If the next machine on the sequence needed by a job is not available, the job can wait and joins the queue at that machine.
- All machines process jobs in the same order (permutation flow shops).

2. Heuristic Rules for FSSP

In this section, the selected heuristic algorithms for solving flow shop scheduling problem are presented.

2.1. SAI Heuristic

SAI method is used to frame a sequence of jobs for processing the n-jobs on m-machines in such a way that the total elapsed time is minimized. The steps of this method, as stated by Gupta and Ali [4], are as follow:

- **Step 1:** Examine the jobs and select the least job processing time among all n jobs for each machine and then marked it with (−) sign
- **Step 2:** Similarly, select the least processing time among all m machines for each job and then marked it with (+) sign .
- **Step 3:** Again, examine the rows and columns of table, select the cell with (∓) sign and is placed in the optimal job sequence.
- **Step 4:** Step 1 to 3 are repeated until all the jobs are placed in the optimal job sequence.

There may be a situation where a tie has occurred:

- If (∓) occurs at more than one place, then the job with least processing time is selected and is placed in the optimal job sequence.

- If (∓) occurs at more than one place and the processing time for the allocated jobs is same. Then the job which will process on the lower order positional machine is selected that is by ignoring the other higher order of machines.

Step 5: Lastly, we calculate the ideal time and total elapsed time of machines.

2.2. Palmer's Heuristic

To minimize the makespan in static flow shop, palmer has developed a heuristic which is known as the Palmer's Heuristic. This heuristic rule sometimes gives optimal solution (Makespan) but not guarantee to give optimal solution for all the time. Palmer has developed a concept of slope to compute the optimum makespan and job sequence in flow shop. Based on this slope (Aj), this heuristic can evaluate only one optimal sequence of job.

Palmer heuristics comprises two following steps (Syarif et. al. [3]):

Step1: If the number of jobs is n and machine is m, then the slope Aj for jth job is computed as follows;

$$A_j = \sum_{i=1}^{m} \{m - (2i - 1)\} P_{ij}$$

Where, P_{ij} denotes the processing time of the job.

Step 2: Jobs are arranged in the sequence according to their descending (decreasing) order of slope Aj value.

2.3. CDS Heuristic

The method generates m - 1 sequences, one for each value of I, where i = 1,2,...m-1. from which the best is chosen. The sequence number 1 is constructed by solving a two-machine problem using Johnson's rule where the two pseudo factors Ail and Bil are generated using the following expressions (Sule [6]):

$$A_{il} = \sum_{j=1}^{l} P_{i.j} \quad , \quad B_{il} = \sum_{j=1}^{l} P_{i,m-j+1}$$

A_{il}: The sum of processing times for job i on machines from 1 to j.
$P_{i.j}$: is the processing time of job i on machine j.
B_{il}: The sum of processing times for job i on machines from m down to m – j +1.
$P_{i,m-j+1}$: is the processing time of job i on machine m – j +1 counted in reverse order.

The steps of Johnson's rule are described as following (Johnson [14]):

Step 1: First of all, the different processing times for all different jobs are assigned to both machines.

Step 2: The least/smallest processing times of each job from all the jobs is noticed and if the minimum processing time of the particular job is on machine 1, then the job is put into the first sequence otherwise if the minimum processing time is on machine 2, then it is put into the last place of the sequence.

Step 3: The selected job is then removed from the sequence.

Step 4: Then again, the minimum processing time of any job from the list of jobs is detected and if the least processing time is on machine 1, and then it is put accordingly in the second position if the previous job is in fist position or in the first position if the previous job is in last position. While if the least processing Ume is on machine 2, then it is put accordingly in the last position if the previous Job is in the first position otherwise in the second last position.

Step 5: Accordingly, all the jobs are sequenced in the above-mentioned procedure.

2.4. Gupta's Heuristic

Gupta thought a transitive job ordering in the form of follows that would produce good schedules (Gupta [9]). Where m > 3, arrange the jobs in descending order of the Sj.

$$s_j = \frac{e_j}{\min\limits_{1 \le k \le m-1} \{t_{jk} + t_{j,k+1}\}}$$

Where:

$$e_j = \begin{cases} 1 & if \quad t_{j1} < t_{jm} \\ -1 & if \quad t_{j1} \ge t_{jm} \end{cases}$$

t_{jm} is the processing time of job j on machine m.

2.5. Rapid Access (RA) Heuristic

The Rapid Access Procedure (RA), as presented by Dannenbring [7], uses a weighting scheme similar to that for the Slope Order and Campbell, Dudek and Smith methods. A single, two-machine subproblem is formed where the processing times are determined from the weighting scheme, and the subproblem is solved using Johnson's two-machine algorithm.

The weighting scheme chosen was selected for its simplicity.

Where t_{ij} is the processing time for the i^{th} job on the j^{th} machine in the main problem, and p_{ij} is the subproblem processing time for the i^{th} job on the j^{th} machine (i = 1, 2, ..., n and j = 1, 2).

$$P_{i1} = \sum_{j=1}^{m} (m - j + 1) t_{ij}, \qquad p_{i2} = \sum_{j=1}^{m} (j) t_{ij}.$$

Step 1: Calculate U = { i | Pi1' < Pi2' } and V = { i | Pi1' ≥ Pi2' }

Step 2: Sort jobs in U with ascending order of Pil'.

Step 3: Sort jobs in V with descending order of Pil'.

Step 4: Optimal sequence is obtained as schedule S by U and V. S={U,V}

3. Results

3.1. Computational Results

To evaluate and compare the performance of heuristic rules, twelve FSSP (from table1 to table 12), were solved by using five heuristics namely SAI, Palmer, Gupta, RA and CDS. Table 13 summarizes the results by presenting the makespan (C_{max}) for each problem by using the selected heuristic rules.

Table 1 Problem 1 (3jobs and 3machines)

MC/Jobs	**1**	**2**	**3**
M1	4	3	6
M2	6	7	2
M3	2	1	5

Table 2 Problem 2 (4jobs and 3machines)

MC/ Jobs	**1**	**2**	**3**	**4**
M1	6	8	3	4
M2	5	1	5	4
M3	4	4	4	2

Table 3 Problem 3 (5jobs and 3machines)

MC/Jobs	**1**	**2**	**3**	**4**	**5**
M1	3	8	7	5	2
M2	3	4	2	1	5
M3	5	8	10	7	6

Table 4 Problem 4 (3jobs and 4machines)

MC / Jobs	**1**	**2**	**3**
M1	20	19	20
M2	20	19	20
M3	5	1	5
M4	5	2	5

Table 5 Problem 5 (4jobs and 4machines)

MC / Jobs	**1**	**2**	**3**	**4**
M1	20	17	21	25
M2	10	7	8	5
M3	9	15	10	9
M4	20	17	21	25

Table 6 Problem 6 (5jobs and 4machines)

MC / Jobs	**1**	**2**	**3**	**4**	**5**
M1	7	11	2	14	18
M2	15	18	13	4	11
M3	14	18	11	27	32
M4	21	6	16	14	16

Table 7 Problem 7 (3jobs and 5machines)

MC/Job	**1**	**2**	**3**
M1	6	5	4
M2	4	5	3
M3	1	3	4

M4	2	4	5
M5	8	9	7

Table 8 Problem 8 (4jobs and 5machines)

MC/Job	**1**	**2**	**3**	**4**
M1	6	5	4	7
M2	4	5	3	2
M3	1	3	4	2
M4	2	4	5	1
M5	8	9	7	5

Table 9 Problem 9 (5jobs and 5machines)

MC/Job	**1**	**2**	**3**	**4**	**5**
M1	5	9	9	4	3
M2	9	3	4	8	5
M3	8	10	5	8	6
M4	10	1	8	7	3
M5	1	8	6	2	7

Table 10 Problem 10 (4jobs and 6machines)

MC/Job	**1**	**2**	**3**	**4**
M1	15	17	21	18
M2	8	7	7	6
M3	6	9	12	11
M4	14	10	9	12
M5	6	15	11	14
M6	26	22	19	17

Table 11 Problem 11 (5jobs and 6machines)

MC/Job	1	2	3	4	5
M1	1	5	5	1	1
M2	5	75	75	5	5
M3	5	1	5	5	5
M4	5	1	5	5	5
M5	1	5	2	1	1
M6	2	3	3	1	1

Table 12 Problem 12 (6jobs and 6machines**)**

MC/ Job	**1**	**2**	**3**	**4**	**5**	**6**
M1	12	25	10	15	20	18
M2	18	10	20	12	15	22
M3	22	14	25	30	10	12
M4	15	30	12	20	25	10
M5	20	12	18	10	15	25
M6	10	18	15	25	12	20

Table 13 Comparative results of the makespan.

	Problem size (n*m)											
Algorithm	3*3	4*3	5*3	3*4	4*4	5*4	3*5	4*5	5*5	4*6	5*6	6*6
SAI	23	30	42	89	122	133	38	45	65	134	181	213
Palmer	24	26	43	82	122	130	39	45	61	133	181	223
Gupta	24	26	42	82	122	127	38	43	64	133	178	216
RA	24	27	42	82	122	127	38	43	58	133	182	214
CDS	23	26	42	82	122	123	38	43	59	133	176	212

It can be noticed that the minimum makespan can be achieved by more than one heuristic rule. Also, CDS produced the minimum makespan for almost all the problems, but it could not be considered as the best heuristic until a statistical test was made to ensure that the differences between the heuristics are statistically significant and not from a chance or random error.

3.2. The Analysis of Variance

3.2.1. The statistical model and hypothesis

Randomized Complete Block Design (RCBD) was used to compare the performance of the selected heuristics by considering the size of problems used in the test as a blocking factor. That would separate the variance between different problems' size (blocking effect) from the variance between heuristics. This reduces noise and provides a clearer picture of true heuristics differences. The statistical model for the RCBD can be written in several ways. The traditional model is an effects model:

$$y_{ij} = \mu + \tau_i + \beta j + \varepsilon_{ij}$$

Where μ is an overall mean, τ_i is the effect of the ith heuristic algorithm, βj is the effect of the jth problem instance, and ε_{ij} is the usual NID $(0, \sigma^2)$ random error term. [i=1,2,…,5 and j=1,2,….,12].

In RCBD, we are interested in testing the equality of the treatment means. Thus, the hypotheses of interest are:

$$H_0: \mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5$$

$$H_1: \text{at least one } \mu_i \neq \mu_j$$

3.2.2. Model adequacy checking

The use of ANOVA requires that certain assumptions be satisfied. Specifically, these assumptions are that the observations are adequately described by the model and that the errors are normally and independently distributed with mean zero and constant but unknown variance. If these assumptions are valid, the analysis of the variance procedure is an exact test of the hypothesis of no difference in treatment means.

In order to check these assumptions, plots of residuals are constructed using Minitab17. Figure 1 shows the normal probability plot of these residuals. The general impression from examining this display is that the error distribution is approximately normal.

A plot of residuals versus heuristic rules is shown in Figure 2. There is no reason to suspect any violation of the independence or constant variance assumptions (see figures 2 and 3).

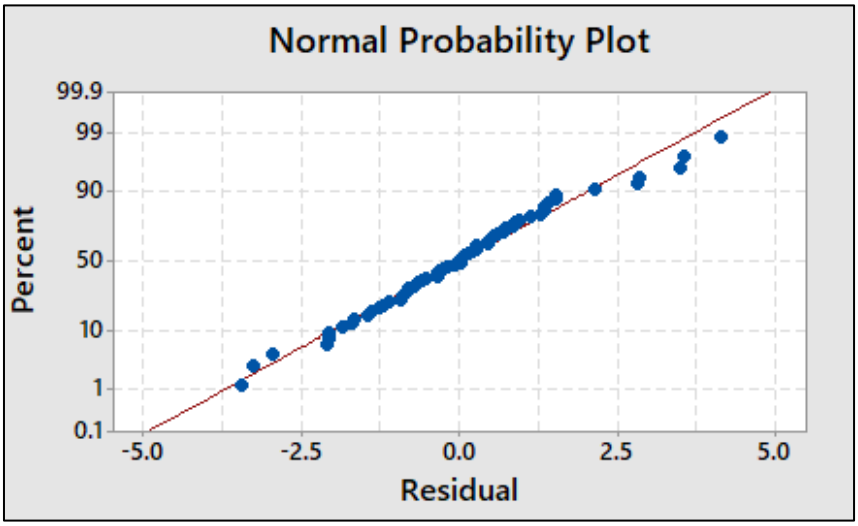

Figure 1 Normal probability plot of residual.

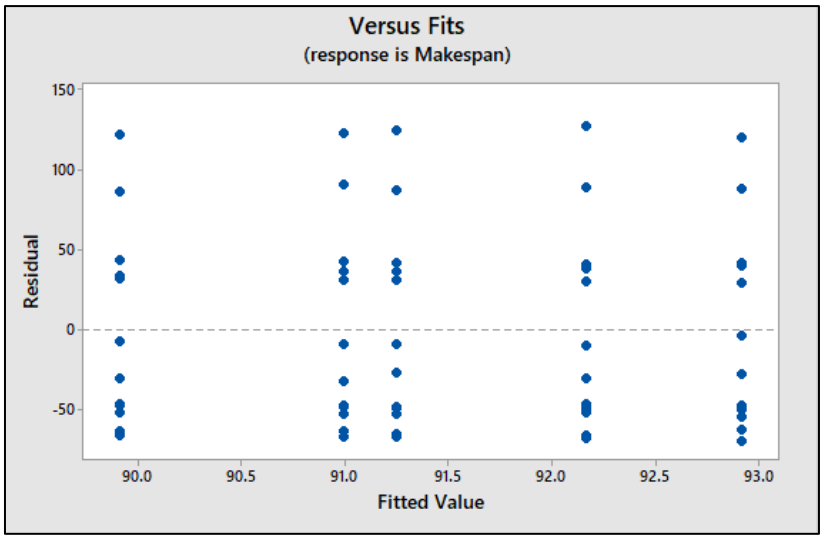

Figure 2 Plot of residual versus heuristic algorithm.

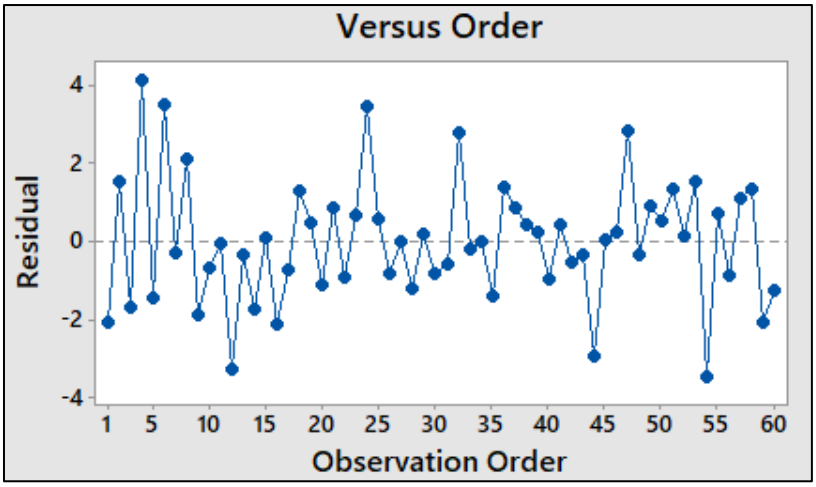

Figure 3 Plot of residual versus observation order.

3.2.3. Results of ANOVA

Table 14 presents the results of the analysis of variance, from these results it can be concluded that there is a significant difference between the heuristic rules covered by this study, since the mean makespan of these heuristics differ significantly (p-value = 0.003 less than 0.05).

Table 14 The results of ANOVA.

Source of variation	Degrees of freedom	Sum of squares	Mean of squares	F_0	P-value
Heuristic rule	4	63	15.4	4.62	0.003
Problem size	11	221274	20115.8	6018.33	
Error	44	147	3.3		
Total	59	221483			

A post-hoc analysis (Tukey's test) was conducted to determine which heuristics differed from each other. The findings indicated that the CDS heuristics is significantly different from the SAI and Palmer heuristics but does not differ significantly from the RA and Gupta heuristics (see Table 15).

Table 15 Grouping information using the Tukey method and 95% confidence.

Heuristic algorithm	N	Mean	Grouping	
SAI	12	92.91	A	
Palmer	12	92.17	A	
Gupta	12	91.25	A	B
RA	12	91.00	A	B
CDS	12	89.91		B

4. Discussion

Results of the present study provide a statistically sound comparison between five heuristic rules (SAI, Palmer, Gupta, RA and CDS) for flow shop makespan. Based on the findings that the CDS heuristics is significantly different from the SAI and Palmer heuristics but do not differ significantly from the RA and Gupta heuristics. While the present study reveals significant performance variations, earlier research by Hossain et.al. [10] found Palmer, CDS, and NEH equally

optimal when applied to a smaller-scale scenario (4 jobs, 10 machines). This contrasts with the current study's conclusion that CDS significantly outperforms Palmer, suggesting that problem scale (tested here across varied sizes) critically influences heuristic efficacy. Similarly, a study by Rayhan and Chakma [11] used a FSSP of 5 jobs and 5 machines, positioned NEH as dominant and CDS as secondary, with Gupta underperforming both—diverging from the current results where CDS and Gupta were statistically indistinguishable. This reinforces that heuristic performance is context-dependent, influenced by factors like job-machine configurations and problem complexity. A study by Syarif et al. [3] further highlights this contingency: while advanced metaheuristics like PBO achieved exceptional results on benchmark problems, the current research demonstrates that simpler heuristics like CDS remain highly competitive for makespan minimization, particularly when statistical validation confirms their robustness across heterogeneous instances.

Collectively, these comparisons underscore two key insights: First, CDS consistently demonstrates strong performance across studies but its relative dominance fluctuates with problem structure. Second, the current research's application of ANOVA fills a significant methodological gap in flow shop scheduling literature. Previous studies relied heavily on case-specific comparisons or error-rate metrics without accounting for inter-instance variability or family-wise error control. By contrast, this study's statistical design not only validates CDS as a top-tier heuristic but also establishes a replicable framework for future evaluations. For both researchers and practitioners, this emphasizes that heuristic selection must balance statistical evidence with problem-specific constraints, and that ANOVA-driven approaches offer a gold standard for objective algorithm assessment in scheduling optimization.

5. Conclusion

This paper presented a comparative analysis of five heuristic rules for solving FSSP, with the primary objective of minimizing the makespan time. The selected heuristics were applied to a set of problem instances of varying sizes. After obtaining the makespan results for each method, the data were analyzed using Analysis of Variance (ANOVA) under a Completely Randomized Block Design (CRBD).

The results of ANOVA and Tukey's test indicated that the CDS heuristic significantly outperformed both the SAI and Palmer heuristics in minimizing makespan, while its performance was not significantly different from that of RA and Gupta. Based on the findings, it can be concluded that CDS demonstrates relatively better performance in most cases and can be considered a favorable option when focusing on minimizing makespan time. However, all five heuristics remain applicable, and their effectiveness may vary depending on the specific characteristics of the scheduling problem. These findings add to a growing body of literature on the applications of heuristics. The study also highlights the value of using statistical tools such as ANOVA for conducting reliable and data-driven heuristic comparisons.

Future studies should apply the heuristics to a broader variety of problem instances, particularly those with higher complexity and scale. Furthermore, evaluation should not be limited to makespan alone. Additional criteria such as flow time, waiting time, machine utilization, or computational efficiency may provide deeper insights. Also, the fuzzy processing time should be covered in evaluating the performance of heuristic rules.

References

[1] Pinedo ML. Scheduling: theory, algorithms, and systems. 3rd ed. New York: Springer; 2008.

[2] Graham RL, Lawler EL, Lenstra JK, Rinnooy Kan AHG. Optimization and approximation in deterministic sequencing and scheduling: a survey. Ann Discrete Math. 1979; 5:287-326.

[3] Syarif A, Pamungkas A, Kumar R, Gen M. Performance evaluation of various heuristic algorithms to solve job shop scheduling problem (JSSP). Int J Intell Eng Syst. 2021;14(2):334-343.

[4] Gupta S, Ali I, Ahmed A. A new algorithm for solving job shop sequencing problem. Int J Comput Sci Eng (IJCSE). 2016;5(2):93-100.

[5] Gupta A, Chauhan SR. A heuristic algorithm for scheduling in a flow shop environment to minimize makespan. Int J Ind Eng Comput. 2015; 6:173-184.

[6] Sule DR. Production planning and industrial scheduling: examples, case studies and applications. Boca Raton: CRC Press; 2007.

[7] Dannenbring DG. An evaluation of flow shop sequencing heuristics. Manage Sci. 1977;23(11):1174-1182.

[8] Nawaz M, Enscore E, Ham I. A heuristic algorithm for the m machine, n job flow shop sequence problem. OMEGA. 1983;11(1):91-95.

[9] Gupta J. A functional heuristic algorithm for the flow shop scheduling problem. Oper Res Q. 1971; 22:39-47.

[10] Hossain S, Asadujjaman, Nayon AA, Bhattacharya P. Minimization of makespan in flow shop scheduling using heuristics. In: Proceedings of the International Conference on Mechanical, Industrial and Energy Engineering; 2014.

[11] Rayhan DSA, Chakma S. A comparative analysis of heuristics for optimizing the makespan in flow shop scheduling. In: Proceedings of the International Conference on Mechanical, Industrial and Materials Engineering; 2015.

[12] Baker KR. Introduction to sequencing and scheduling. New York: John Wiley & Sons, Inc; 1974.

[13] Ruiz R, Maroto C. A comprehensive review and evaluation of permutation flowshop heuristics. Eur J Oper Res. 2005;165(2):479-494.

[14] Johnson SM. Optimal two- and three-stage production schedules with setup times included. Nav Res Logist Q. 1954; 1:61-68.