MÜHENDİSLİK FAKÜLTESİ

Uluslararası Mühendislik Araştırma ve Geliştirme Dergisi
International Journal of Engineering Research and Development

UMAGD, (2022) 14(1), 155-163.

10.29137/umagd.978415

Cilt/Volume:14 Sayı/Issue:1 Ocak/January 2022

Araştırma Makalesi / Research Article

A Substantially Improved New Algorithm for Flowshop Scheduling Problem with Uncertain Processing Times

Muberra Allahverdi[1]

[1] *Department of Mathematical Sciences, School of Natural Sciences, Kean University, 1000 Morris Ave, Union, NJ 07083, USA*

***Başvuru/Received**: 03/08/2021* ***Kabul / Accepted**: 18/09/2021* ***Çevrimiçi Basım / Published Online:** 31/01/2022*

***Son Versiyon/Final Version**: 31/01/2022*

Abstract

The performance measure of total completion time (TCT) plays a key role in manufacturing to improve performance, e.g., reducing inventory levels. Moreover, since uncertainty is an inevitable part of certain manufacturing environments, it is especially important to address cases with uncertain processing times. This paper addresses the four-machine flowshop scheduling problem to minimize TCT with uncertain processing times. Due to the NP-hardness of the problem, different algorithms were presented as solutions in scheduling literature. In this paper, a new substantially improved algorithm is proposed and parameters of the algorithm are fine tuned. The proposed algorithm is compared to the best existing algorithm (RAIRO Operations Research 54, 529–553, 2020) in scheduling literature using extensive computational experiments and statistical analysis. Computational methods using the programming language python, along with statistical inference, is used to confirm the effectiveness of the proposed algorithm over the existing ones. Computational methods reveal that the proposed algorithm is, on average, 86.8% more effective than the best existing one in literature with similar computational times. A test of hypothesis further confirms the effectiveness of the proposed algorithm with a p-value of less than 0.00001, which is practically zero.

Key Words

“Algorithm, uncertain processing times, flowshop scheduling, total completion time”

***Corresponding Author:** mu.allahverdi@gmail.com*

1. Introduction

A manufacturing process where jobs flow from one machine to the next is known as a flowshop. Total completion time (TCT) is the sum of the completion times of all jobs in the last machine. Minimizing the total completion time is pivotal in increasing performance during manufacturing and ensuring competition among companies, Chen (2015). Furthermore, the total completion time is directly related to inventory levels, lead times, and work in process (Framinan and Perez-Gonzalez 2017a, Framinan and Perez-Gonzalez 2017b), which makes it an essential performance measure to consider for better production.

The processing time of job j on a certain machine is the amount of time required to process job j on that machine. In some cases, processing times are deterministic. Nonetheless, it is not always possible to correctly predict processing times due to a number of uncertainties in a manufacturing environment, including machine operator fatigue, new technology, tool conditions, and poor experience, Tayanithi et al. (1992). Further uncertainties include the breakdown of equipment, incorrect predictions, the absence of workers, and new jobs with no past data. Moreover, Mahjoub et al. (2011) stated that data might be subject to some uncertainties, and thus, it is impossible in practice to accurately estimate the parameters of some scheduling problems. Seif et al. (2020) stated another case (maintenance) where job processing times can be modelled as uncertain.

Attempting to reduce TCT while ignoring these uncertainties may severely affect performance. This is especially true with manufacturing environments that are known to have a wide range of uncertainties. In such environments, not only is it inefficient to consider deterministic processing times, it is also disadvantageous to consider a certain probability distribution. Firstly, past data is necessary to predict a probability distribution which may not always be available. Secondly, a probability distribution is useful only with a large number of realizations. When the number is small, the probability distribution will not be of much use. See Kouvelis and Yu (1997) for further discussion.

Taking the above into account, we address the problem of minimizing the total completion time of a four-machine flowshop problem with uncertain processing times. No assumptions are made about this job descriptor and no particular probability distribution is considered. The only conditions assumed are possible lower and upper bounds for processing times.

Similar problems have been solved for cases related to two and three machines, including Allahverdi and Sotskov (2003) which establishes dominance relations to minimize Cmax for a two-machine flowshop problem, Allahverdi and Aydilek (2010a) which proposes some polynomial time algorithms for the same problem, Allahverdi and Aydilek (2010b) which solves the two-machine problem with the objective of minimizing maximum lateness when processing times are uncertain, Sotskov et al. (2004) which establishes dominance relations for the two-machine problem with uncertain processing times to minimize total completion time, and Allahverdi and Aydilek (2010c) proposes an improved heuristics for the same problem and indicates the improvement through computational experiments.

Furthermore, Sotskov et al. (2004) investigates the case of three machines with uncertain setup times with the objective of minimizing TCT. Some other papers, including Aydilek et al. (2013, 2015, 2017), Lai et al. (1997) and Lai and Sotskov (1999) also look into various scheduling environments with uncertain processing or setup times.

Finally, Allahverdi and Allahverdi (2020) addresses the four machine flowshop scheduling problem to minimize TCT with uncertain processing times. Ten algorithms are proposed to minimize TCT with uncertain processing times, and one of them is established as the best performing algorithm with the lowest TCT. In this paper, we develop a new algorithm using the python programming language which is substantially better than the best algorithm given in Allahverdi and Allahverdi (2020). We confirm the effectiveness of the new algorithm through computational methods and statistical inference. The percentage improvement of the one established in this paper is on average 86.8% better than the best existing one in literature.

2. A New Algorithm

The problem of a four machine flowshop with the objective of minimizing TCT with uncertain processing times was addressed by Allahverdi and Allahverdi (2020). They proposed different algorithms to solve the problem and showed that one of the algorithms performs the best. In this section, we propose a new algorithm and show that the new algorithm significantly outperforms the best algorithm of Allahverdi and Allahverdi (2020).

Let n be the number of jobs. Let $t_{j,k}$ be the processing time of job j on machine k and let $t_{[j,k]}$ be the processing time of the job in position j on machine k. Let $Lt_{j,k}$ and $Ut_{j,k}$ be the lower and upper bounds of $t_{j,k}$.

The shortest processing time (SPT) sequence is known to give an optimal solution for a single machine problem with the TCT performance measure. Since the case of two-machines is known to be NP-hard, our problem is NP-hard as well. Hence, we attempt to use a variation of the SPT for a four-machine case. Essentially, we propose formulas to combine the processing times (or the upper and lower bounds of the processing times when the processing times are uncertain) of a particular job into a single processing time by giving the processing time on each machine a different weight. In other words, a single machine problem is obtained after using the formula and the SPT is used on the single machine problem to predict a solution.

The formula used to combine the four upper and lower bounds of the processing times for job *i* on the four machines into a single processing time is as follows,

$$\alpha(Lt_{r,1}+ Ut_{r,1})/2+\beta(Lt_{r,2}+ Ut_{r,2})/2+\gamma(Lt_{r,3}+ Ut_{r,3})/2+ \delta(Lt_{r,4}+ Ut_{r,4})/2 \qquad (1)$$

Given a value of n and generated values for the upper $Ut_{i,k}$ and lower bounds $Lt_{i,k}$, our aim is to find appropriate values for the variables $\alpha, \beta, \gamma, \delta$ to minimize the TCT of a sequence which is computed using the new algorithm proposed in this paper.

Steps of the New Algorithm

```
1. Generate the upper bounds $Ut_{i,k}$ for processing times from $U(1,100)$
2. Generate the lower bounds of processing times $Lt_{i,k}$ from $U(\max(1,Ut_{i,k} - \Delta),Ut_{i,k})$ where $\Delta$ is set to the values 10,20, and 30.
3. Let $r$=1
4. Compute $w_i$ as
   $\alpha(Lt_{i,1}+ Ut_{i,1})/2+\beta(Lt_{i,2}+ Ut_{i,2})/2+\gamma(Lt_{i,3}+ Ut_{i,3})/2+ \delta(Lt_{i,4}+ Ut_{i,4})/2$ for i=1, ..., n.
5. Let $r = r + 1$
6. If $r < n$, go to Step 4. Otherwise, continue.
7. Sequence the jobs according to SPT based on the 'processing times' $w_i$.
```

To summarize, upper bounds of processing times $Ut_{i,k}$ are generated from $U(1,100)$ and lower bounds $Lt_{i,k}$ are generated from $U(\max 1,Ut_{i,k} - \Delta,Ut_{i,k})$ where Δ is set to 10, 20, and 30. Hence, Δ determines how to generate the lower bound based on the upper bound. Finally, Formula (1) is used to transform a four machine problem into a single machine problem so that the SPT can be used to determine a sequence.

3. Fine Tuning Parameters of the Algorithm

We aim to find the best values for $\alpha, \beta, \gamma, \delta$ by checking all possibilities from 0.05 to 1 with an increment of 0.05 such that $\alpha+\beta+\gamma+\delta = 1$. The following is a summary for the code which is written in the python programming language.

```
1. A list is created containing all combinations of $\alpha, \beta, \gamma, \delta \in [0.05,0.1,0.15,0.20,0.25,\cdots,1]$ such that $\alpha + \beta + \gamma + \delta = 1$. It is allowed for a certain variable to take the same value more than once as long as the combination $(\alpha, \beta, \gamma, \delta)$ is different.
2. Let $r = 1$
   a  For a given n, do the following:
      i  For each $\Delta$ in [10, 20, 30], where $\Delta$ is defined as above, do the following:
         1  Upper and lower bounds of processing times are generated based on $n$ and $\Delta$
         2  For every possibility of $\alpha,\beta,\gamma,\delta$ such that $\alpha + \beta + \gamma + \delta = 1$, we do the following: we use the following formula to combine the four machine problem into a single machine problem, where the "processing time" for the ith job is $\alpha(Lt_{i,1}+ Ut_{i,1})/2+\beta(Lt_{i,2}+ Ut_{i,2})/2+\gamma(Lt_{i,3}+ Ut_{i,3})/2+ \delta(Lt_{i,4}+ Ut_{i,4})/2$ and the SPT is used to determine the order of the sequence based on this formula.
   b  After completing Step a, we take the sequence with the shortest total completion time and add the parameters $\alpha,\beta,\gamma,\delta$ used for that sequence to Table 1.
3. If $r \leq 30$, let r=r+1 and go to Step 3
```

Table 1 displays the best parameters for each replication. As evident from the table, the numbers are more or less alike.

Table 1. Best parameters for each of the 30 replications

Replication	α	β	γ	δ
1	0.35	0.35	0.25	0.05
2	0.25	0.35	0.2	0.2
3	0.25	0.3	0.3	0.15
4	0.25	0.25	0.25	0.25
5	0.4	0.2	0.2	0.2
6	0.4	0.35	0.15	0.1
7	0.25	0.3	0.25	0.2
8	0.3	0.25	0.2	0.25
9	0.4	0.35	0.2	0.05
10	0.25	0.3	0.25	0.2
11	0.3	0.25	0.2	0.25
12	0.3	0.3	0.25	0.15
13	0.3	0.2	0.25	0.25
14	0.4	0.25	0.2	0.15
15	0.3	0.25	0.2	0.25
16	0.25	0.25	0.25	0.25
17	0.35	0.3	0.15	0.2
18	0.35	0.25	0.2	0.2
19	0.3	0.3	0.2	0.2
20	0.3	0.25	0.25	0.2
21	0.35	0.35	0.15	0.15
22	0.35	0.15	0.25	0.25
23	0.35	0.3	0.15	0.2
24	0.3	0.25	0.2	0.25
25	0.35	0.25	0.25	0.15
26	0.35	0.3	0.15	0.2
27	0.3	0.25	0.3	0.15
28	0.25	0.25	0.25	0.25
29	0.25	0.25	0.25	0.25
30	0.3	0.25	0.3	0.15

The average and median of each parameter are given in Table 2.

Table 2. Mean and median for each parameter

Replication	α	β	γ	δ
Mean	0.313	0.273	0.222	0.192
Median	0.3	0.25	0.225	0.2

As seen in the Table 2, the mean and median are very similar, which implies the accuracy of the parameter predictions. The averages were rounded to 3 decimal places since they must add up to 1.

4. Comparing the proposed algorithm (new-alg) and the existing best algorithm (old-alg)

Substituting the parameters $\alpha,\beta,\gamma,\delta$ with the averages of the above predictions, the following formula is obtained

$$0.313(Lt_{r,1}+ Ut_{r,1})/2+0.273(Lt_{r,2}+ Ut_{r,2})/2+0.222(Lt_{r,3}+ Ut_{r,3})/2+ 0.192(Lt_{r,4}+ Ut_{r,4})/2 \qquad (2)$$

We follow the algorithm outlined in Section 2, New Algorithm, with Formula 2 above and conduct computational experiments to compare the new algorithm to the best algorithm given in Allahverdi and Allahverdi (2020).

For every combination of *n* and Δ, where *n* = 20,40,60,80,100,120,140,160,180,200 and Δ = 10,20,30, we generate 1000 replications and compute the error of each algorithm using the formula
Error(alg)=100(TCT(alg)-min(TCT(old-alg), TCT(new-alg)))/ min(TCT(old-alg), TCT(new-alg)) (3)
for alg = old-alg and new-alg.

In total we look at (10)(3)(1000) = 30000 instances. The results are given in Table 3. The first two columns list the n and Δ values. The table has 30 rows since there are 30 different combinations of *n* and Δ. The third and four columns list the average errors computed using Formula 3. The fifth and sixth columns list the standard deviations of the old and new algorithms and the percentage improvement shows the improvement of the new algorithm's error compared to that of the old algorithm.

Table 3. Comparing old-alg and new-alg

n	Δ	old-alg error	new-alg error	old-alg st dev	new-alg st dev	percentage imp.
20	10	2.78	0.58	3.3	1.24	79.14
20	20	2.76	0.6	3.19	1.36	78.26
20	30	3.06	0.52	3.24	1.23	83.01
40	10	2.73	0.36	2.63	0.94	86.81
40	20	2.7	0.38	2.65	0.98	85.93
40	30	2.91	0.38	2.87	0.97	86.94
60	10	2.67	0.27	2.43	0.73	89.89
60	20	2.83	0.34	2.63	0.93	87.99
60	30	2.61	0.31	2.38	0.85	88.12
80	10	2.41	0.31	2.18	0.85	87.14
80	20	2.66	0.28	2.37	0.77	89.47
80	30	2.65	0.3	2.35	0.84	88.68
100	10	2.42	0.31	2.17	0.79	87.19
100	20	2.49	0.31	2.23	0.8	87.55
100	30	2.49	0.3	2.29	0.82	87.95
120	10	2.22	0.26	1.99	0.72	88.29
120	20	2.36	0.26	2.15	0.7	88.98
120	30	2.25	0.3	2.13	0.78	86.67
140	10	2.21	0.26	1.98	0.68	88.24
140	20	2.33	0.27	2.14	0.7	88.41
140	30	2.31	0.29	2.06	0.76	87.45
160	10	2.26	0.24	1.95	0.64	89.38
160	20	2.23	0.32	2.03	0.78	85.65
160	30	2.31	0.27	2.07	0.71	88.31
180	10	2.08	0.24	1.93	0.64	88.46
180	20	2.06	0.29	1.93	0.73	85.92
180	30	2.18	0.32	1.95	0.77	85.32
200	10	2.04	0.27	1.83	0.71	86.76
200	20	2.09	0.26	1.93	0.69	87.56
200	30	2.1	0.32	1.97	0.79	84.76

As seen in Table 3, there is a considerable difference between the two algorithms. The average percentage improvement is 86.8% with a median of 87.5%, which is a very significant improvement with similar computational times.

The considerable gap between the errors of old-alg and new-alg are evident from Figure 1. The average error of old-alg is 2.44 while that of new-alg is 0.324. While the gap decreases slightly as *n* grows larger, it is more or less stable.

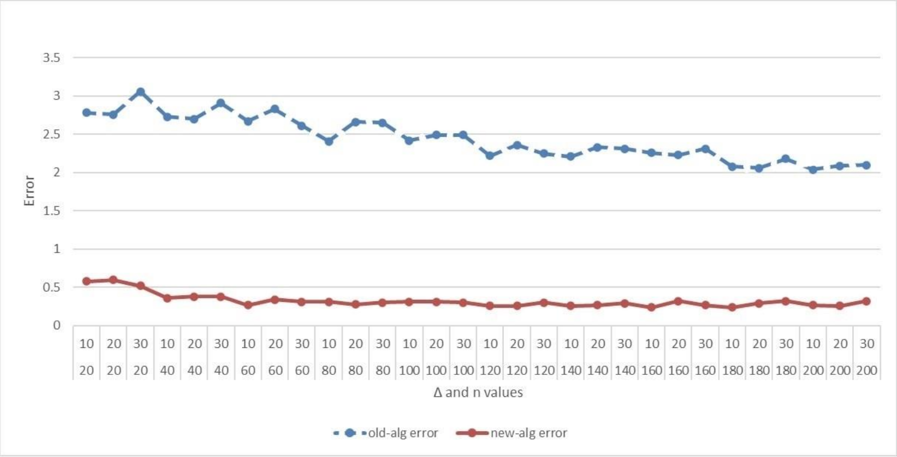

Figure 1. Error vs n and Δ

Figures 2 and 3 illustrate the percentage errors with different values of Δ and *n*. The percentage errors are almost the same for the three different values of Δ with the largest difference being around 0.5%. As *n* grows larger, there is a noticeable difference in the percentage error between *n* = 20 and *n* = 60, after which it seems to stabilize.

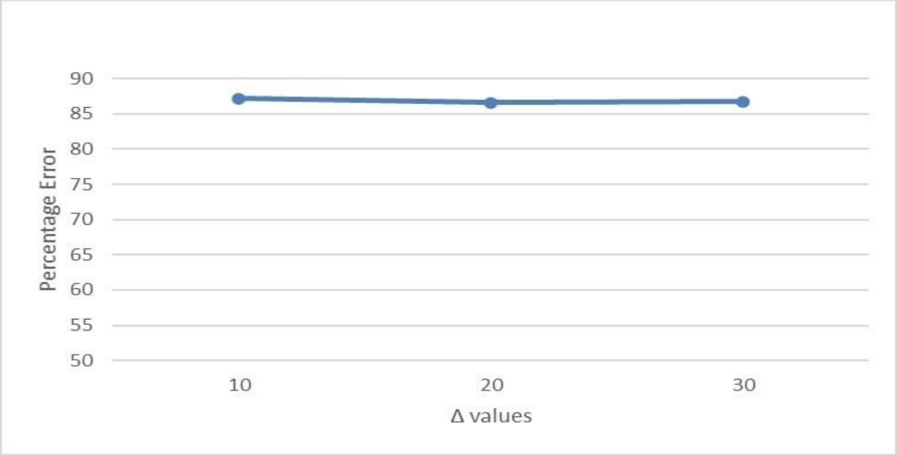

Figure 2. Percentage Error vs Δ

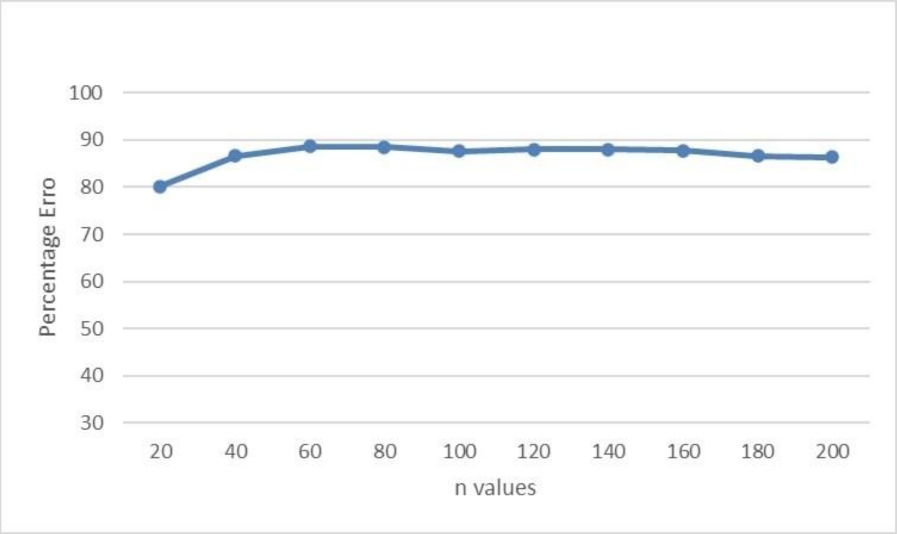

Figure 3. Percentage Error vs n

Finally, the last two figures illustrate the gap between the errors of old-alg and new-alg with different values of n and Δ. In both Figures 4 and 5, the gap seems more or less stable. However, while the gap is relatively stable, we do see a slight increase in the errors of both old-alg and new-alg in Figure 5, which is expected since as Δ increases so does the uncertainty.

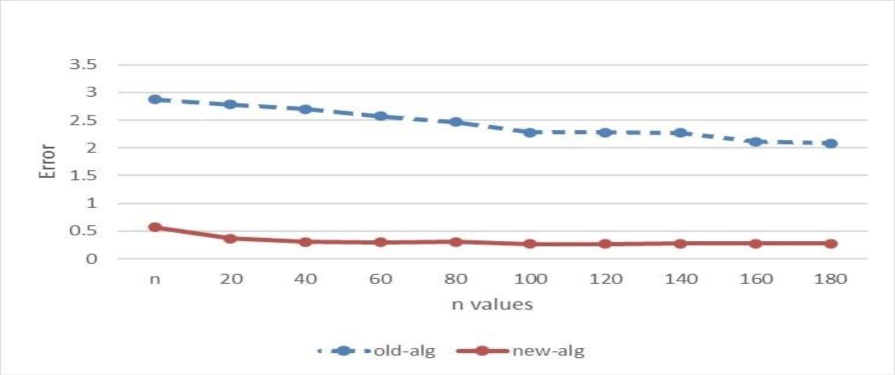

Figure 4. Error vs n

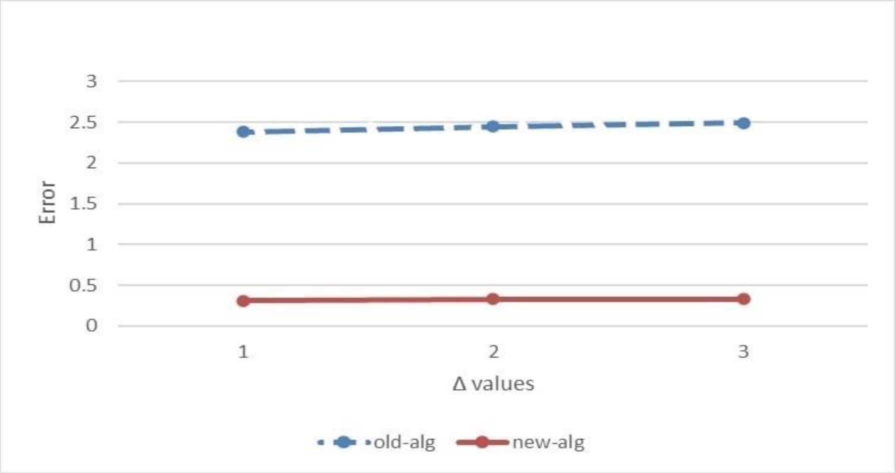

Figure 5. Error vs Δ

5. Hypothesis Testing

We conduct a hypothesis test to confirm the effectiveness of new-alg over old-alg as follows.

$$H_0 : \mu_0 - \mu_1 = 0$$

$$H_1 : \mu_0 - \mu_1 > 0$$

where μ_0 denotes the population mean of the TCT corresponding to old-alg and μ_1 to that of new-alg. The computed z-scores are listed in Table 4.

Table 4. Z-scores

n	delta	z-score
20	10	177.03
20	20	179.62
20	30	211.48
40	10	303.83
40	20	290.62
40	30	275.67
60	10	372.8
60	20	319.98
60	30	360.11
80	10	383.57
80	20	383.27
80	30	377.32
100	10	395.65
100	20	388.39
100	30	370.15
120	10	437.65
120	20	410.76
120	30	378.99
140	10	444.92
140	20	406.34
140	30	418.98
160	10	479.57
160	20	403.87
160	30	425.98
180	10	445.04
180	20	415.71
180	30	423.17
200	10	459.38
200	20	435.61
200	30	395.12

The average z-score is 372.35 and the median is 391.76. If we take $\alpha = 0.01$, we must check whether the calculated z-scores are greater than 2.33. This is obviously the case for every computed z-score. Hence, we reject the null-hypothesis that the means of old-alg and new-alg are the same and accept the alternative hypothesis that the mean of old-alg is greater than the mean of new-alg. The minimum z-score in the table is 177.03 and the p-value is $1-\Phi(177.03) < 0.00001$, which is almost zero.

6. Conclusion

Reducing the total completion time (TCT) is an essential performance measure in scheduling since the TCT is directly related to performance, inventory levels, work-in-process, and so on. For scheduling environments with uncertain processing times, ignoring uncertainty can result in very poor performance. As a result, research has recently been conducted to address this issue and an algorithm was proposed by Allahverdi and Allahverdi (2020).

In this paper, we propose a new algorithm through computational methods and test the new algorithm by comparing it with the best existing algorithm in literature. Both algorithms have similar computational times and the percentage improvement of the proposed algorithm over the existing one is 86.8% on average, which is a substantial improvement.

Furthermore, a test of hypothesis is conducted and the null hypothesis that the proposed algorithm is equal to the existing one is refuted with a p-value of less than 0.00001. This strongly confirms the effectiveness of the new algorithm established in this paper.

We assume that setup times are included in processing times. Even though this might be true for some cases, it is not true for all. When setup times take considerable time and are not negligible compared to processing times, it is crucial to consider them separately in order to meet deadlines, eliminate waste, and increase productivity, Allahverdi (2015), Ha (2020), Lee and Kim (2021). Hence, an extension to the problem is to consider the processing times and setup times separately. Yet another extension is to consider the problem with a due date related performance measure, e.g., Costa et al. (2020).

References

Allahverdi, A. 2015. The third comprehensive survey on scheduling problems with setup times/costs. European Journal of Operational Research 246, 345–378.

Allahverdi, M., Allahverdi, A. 2020. Minimizing total completion time for flowshop scheduling problem with uncertain processing times. 54, 529–553.

Allahverdi, A., Aydilek, H. 2010a. Heuristics for two-machine flowshop scheduling problem to minimize makespan with bounded processing times. International Journal of Production Research 48, 6367–6385.

Allahverdi, A., Aydilek, H. 2010b. Heuristics for two-machine flowshop scheduling problem to minimize maximum lateness with bounded processing times. Computers and Mathematics with Applications 60, 1374–1384.

Allahverdi, A., Aydilek, H. 2010c. Two-machine flowshop scheduling problem with bounded processing times to minimize total completion time. Computers and Mathematics with Applications 59, 684–693.

Allahverdi, A., Sotskov, Y.N. 2003. Two-machine flowshop minimum length scheduling problem with random and bounded processing times. International Transactions in Operational Research 10, 65–76.

Aydilek, A. Aydilek, H., Allahverdi, A. 2013. Increasing the profitability and competitivess in a production environment with random and bounded setup times. Int. Journal of Production Research 51, 106–117.

Aydilek, A. Aydilek, H., Allahverdi, A. 2015. Production in a two-machine flowshop scheduling environment with uncertain processing and setup times to minimize makespan. Int. Journal of Production Research 53, 2803–2819.

Aydilek, A. Aydilek, H., Allahverdi, A. 2017. Algorithms for minimizing the number of tardy jobs for reducing production cost with uncertain processing times. Applied Mathematical Modelling 45, 982–996.

Chen, J.F. 2015. Unrelated parallel-machine scheduling to minimize total weighted completion time. Journal of Intelligent Manufacturing 26, 1099–1112.

Costa, M.R.C., Valente, J.M.S., Schaller, J.E. 2020. Efficient procedures for the weighted squared tardiness permutation flowshop scheduling problem. Flexible Services and Manufacturing Journal 32, 487-522.

Framinan, J.M., Perez-Gonzalez, P. 2017a. The 2-stage assembly flowshop scheduling problem with total completion time: Efficient constructive heuristic and metaheuristic. Computers and Operations Research 88, 237–246.

Framinan, J.M., Perez-Gonzalez, P. 2017b. New approximate algorithms for the customer order scheduling problem with total completion time objective. Computers and Operations Research 78, 181–192.

Ha, C. 2020. Evolving ant colony system for large-sized integrated process planning and scheduling problem considering sequence-dependent setup times. Flexible Services and Manufacturing Journal 32, 523-560.

Kouvelis, P., Yu, G. 1997. Robust discrete optimization and its applications. Kluwer Academic Publisher.