A Non-delay Algorithm for The Job-shop Scheduling Problem

Nur Ezha Vidawati
Supply Chain
FMCG Company

Puryani, Apriani Soepardi, and Mochammad Chaeron
Department of Industrial Engineering
Faculty of Industrial Engineering
Universitas Pembangunan Nasional Veteran, Yogyakarta, Indonesia
apriani.soepardi@upnyk.ac.id

Abstract

The previous research applied the Artificial Immune System Algorithm in job shop scheduling with five jobs and three machines with a makespan result of 61.15 time-units. The algorithm is considered inaccurate because it requires complicated steps for operators to understand, such as determining random numbers of initialization and clones, donor seeds, and repetition of gene fragments to produce a smaller makespan. The proposed algorithm is derived from the Non-delay Algorithm with a modification in the form of ranking based on the criteria of the earliest start time, the earliest finish time, the longest remaining total processing time, and the total remaining operations. Modifications are also made by giving the priority order of machines based on the most used machines in each operation. The results showed that the proposed algorithm could shorten makespan by 3.99% to 57.16 time-units with a reliability percentage of 55.55%. The proposed algorithm gave the same or better results in the first ten cases with two to six jobs. The small number of jobs and machines resulting in a small combination of scheduling sequences and cases that might be resolved optimally on the reference data. The proposed algorithm could not give a shorter makespan result in the eleventh to eighteenth cases with more than six jobs. The proposed algorithm only provided one scheduling sequence, with the advantage of being a few easy steps.

Keywords

Job shop scheduling, Makespan, Non-delay algorithm, Scheduling, Sequence.

1. Introduction

Production scheduling is one of the main activities in the manufacturing industry. In production scheduling, there is an activity of dividing jobs into available machines or workstations. This activity requires an algorithm or steps in the preparation that might determine the makespan or total completion time. Makespan will affect the speed of job completion or product delivery. Using the proper steps, methods, or algorithms can reduce makespan to enable the next job to be started earlier.

Astuti (2013) applied Artificial Immune System (AIS) Algorithm to minimize the overall time used in job shop scheduling, which produced a makespan of 61.15 minutes. This study was accomplished at PT. Aneka Adhilogam Karya. The company was initially established in 1968 and is currently engaged in metal casting, producing pipe fittings with gray cast iron and ductile cast iron specifications. It produces five kinds of products: the collar, flange socket, flange spigot, giboult joint, and clamp saddle by three machines, namely lathes, drills, and hand grinders. The algorithm application above is considered relatively complex as they applied random numbers to the initialization stage, clone, and donor seed and re-calculate to aim lower makespan.

A simple job shop problem with five jobs on three machines can be solved with the Non-Delay (ND) Algorithm. The advantage of this algorithm is the relatively easy processing process by scheduling work based on the earliest start time. However, the makespan result using the ND Algorithm can give a longer result than applying the AIS Algorithm. Based on the above problems, modifying the ND Algorithm to minimize the makespan with more straightforward algorithm steps is necessary. The benefit of designing a job shop-type production scheduling algorithm is that the company can reduce the potential for delays in fulfilling customer orders.

2. Literature Review

A Pinedo (2012) defines *scheduling* as a form of decision-making that plays an essential role in the manufacturing and the service industry. Scheduling is one of the strategies to survive in the competitive era between companies. The company is required to be able to meet the target supply to customers. Failure to comply with requests can result in losses. Effective production scheduling is needed in using available resources to achieve company goals. The form of problems faced in scheduling can be related to the company's available resources and the goals to be achieved.

Production scheduling in the manufacturing industry begins with orders received by companies which are then broken down into jobs or work steps with a deadline. This division of steps or jobs will be carried out on the machine at the workstation in a specific order or arrangement. Processing of steps or jobs inside the workstation can be delayed if a machine is busy or an arrival of a job with a higher priority. Unpredictable events such as machine failure can also have an impact on job execution. Scheduling is required with consideration of efficiency and control of operations within the workstation. Production scheduling is not only a matter of supervising and organizing a workstation. Scheduling also includes planning for long-term use of resources such as inventory monitoring and forecasting future demand. Production scheduling makers are obliged to be responsive in dealing with uncertain aspects such as uncertain processing time, the possibility of machine failure, and the arrival of orders near the deadlines.

According to Leung (2004), scheduling is concerned with allocating scarce resources to activities to optimize one or more performance measures. Meanwhile, Baker and Trietsch (2013) stated that in scheduling, information on the type and amount of available resources is needed to make it clear whether a job is possible or not. Other additional information needed regarding determining the feasibility of a task is the resource requirement, duration, earliest time the job can be started, and the due date. There are two common problems encountered in production scheduling, namely: (1) limited engine capacity, (2) determining the order of processing orders. Therefore, a solution is needed to these two problems that meet the following requirements: (1) What resources should be allocated to do each job, and (2) When should the job be done.

3. Methods

The case in Astuti (2013) can be solved efficiently using the ND algorithm, but it gives a longer makespan result. The algorithm has a scheduling rule by scheduling operations with the earliest start time to avoid idle machines. ND algorithm modification could be done by adding scheduling criteria such as the earliest finish time, the longest remaining total processing time, and the highest total remaining operations. Modifications are also made by prioritizing the machines based on the most used machines for each working operation.

The problem was obtained from previous studies conducted by Fisher and Thompson (1963), Lawrence (1984), Taillard (1993), Willens and Rooda (1994), Zhang et al. (2010), Thamilselvan and Balasubramanie (2011), Lei (2011), Astuti (2013), Bouzidi and Riffi (2014), Zheng et al. (2014), Kuhpfahl (2015), Kurdi (2015), Prasetyo et al. (2015), Sinambela (2017), Zang et al. (2019). The conceptual model shows the relationship between the variables that determine the characteristics of the system. At this stage, an analysis of the relationship between the variable's problem and limitations with the observed system was carried out. The analysis was adjusted to reference journals and previous studies as references. The decision variable in this study was to determine the order of product processing for each job with the following parameter: (1) Number of jobs, and (2) Job operation processing time

The following are the limitations of the study carried out: (1) This study did not take into account the element of cost, (2) This study did not consider the labor aspect, (3) The modification only considers the criteria of the earliest start time, earlier finish time, most extended remaining operation, and most minor remaining operation. Whereas, the assumptions made in designing the job shop type on the production scheduling algorithm are the machine is in good shape with optimal production capacity and there is no change or cancellation of the scheduled jobs.

The model formulation is the process of formulating model behavior in the form of a variable and notation. At this stage, a model development was carried out based on the problems to be solved. The notation used in this study can be seen below.

(i,j,k)	:	step j of job i on machine k (operation) without precedence in *MkN*
(i,j,k)*	:	step j of *job i* on machine k (operation) with precedence in *MkN*
Sij	:	starting time on step j of job i (operation)
CT	:	completion time, where *CT* = *Sij* + *Pijk*
Ji	:	job i, where $i = 1, 2, \ldots, n$
Mk	:	machine k,where $k = 1, \ldots, N$
w	:	machine priority, where $w = 1, 2\ldots., N$
N	:	number of machines
P_{ijk}	:	processing time of job i on machine k of step j
PS_t	:	a partial schedule containing some j scheduled operation
PTR	:	sum of *pijk* from leftover (i,j,k), where *PTR* (i,j,k) = $\sum_{j+1}^{n} Pijk$
PCT	:	the difference between PTR and CT
Pref	:	preference of criteria, starting from 1 for the best
RS	:	the difference between *CT* and the next *Sij*.
S_t	:	set of operations ready to be scheduled for every *Mkw*
t	:	iterations
TOL	:	the number of leftover operations on a job
Total Pref	:	the total of all preference values from the assigned criteria
Val	:	the value of a criteria

The steps of the proposed model are as follows:

Step 1: At iteration step $t = 0$, *PSt* = 0, determine the order of *Mkw* based on the most used machines in each operation, starting with the first operation in the top order. If in one operation, all machines are used equally, then the next operation can proceed.

Step 2: Determine *St* = $\{(i,j,k)\}$ for each *Mkw*.

Step 3: For *Mkw*, w= 1,2…N-1, give preference values for *St* with the criteria *Sij* (minimization), *CT* (minimization), *PTR* (maximization), and *TOL* (maximization). If the option *Sij*>1, then:

a. If *Pijk* > *RS*, schedule *St* to *Sij* after RS.

b. If *Pijk* ≤ *RS*, schedule *St* to *RS*.
Schedule *St* of the smallest preference total. If there are the same total preferences, it is selected based on the most significant *PCT*. If there are the same *PCT* values, then the operations are sorted independently. Update data, input the data to *PSt* at the next stage, go to step 4.

Step 4: For each new partial schedule generated in step 3, update data:

a. Remove the scheduled (i,j,k) from *St* for *Mkw*, $w = 1,2, \ldots, N$-1.

b. The iteration of t changes to t+1, proceed to step 5.

Step 5: If *St* = {}, go to step 6.
If *St* ≠ {}, back to step 3.

Step 6: Repeat step 3 for *MkN;* go to step 7.

Step 7: Determine *St* = $\{(i,j,k)^*\}$.
If *St* = {}, scheduling is complete.
If *St* ≠ {}, go to step 8.

Step 8: Repeat step 3 with *St* = $\{(i,j,k)^*\}$ for *Mkw*, w= 1,2…N-1and change the value of t to t+1, go back to step 7.

4. Data Collection

The modification is focused on affecting makespan criteria, such as the earliest time to start, earliest completion time, longest processing time remaining, minimum total operation, and machine prioritization by the number of usage for every operation. Makespan is chosen as the performance measurement. The proposed algorithm is conducted at eighteen job shop cases with different numbers of jobs from various sources. It can be seen in Table 1.

Table 1. Number of job cases

No.	Data source	Number of jobs	Number of machines
1	Zhang et al. (2010)	2	3
2	Willens and Rooda (1994)	2	3
3	Bouzidi and Riffi (2014)	3	3
4	Kurdi (2015)	3	3
5	Thamilselvan and Balasubramanie (2011)	4	3
6	Lei (2011)	4	3
7	Astuti (2013)	5	3
8	Sinambela (2017)	5	10
9	Fisher and Thompson (1963)	6	6
10	Prasetyo et al. (2015)	6	4
11	Taillard (1993)	7	7
12	Taillard (1993)	7	7
13	Zang et al. (2019)	8	6
14	Zheng et al. (2014)	8	8
15	Kuhpfahl (2015)	9	9
16	Kuhpfahl (2015)	9	9
17	Fisher and Thompson (1963)	10	10
18	Lawrence (1984)	10	5

To put the algorithm into perspective, here are the steps of the proposed algorithm using the data of Astuti 2013. The machine routing and processing time are shown in Table 2 and Table 3, respectively.

Table 2. Machine routing of Astuti (2013)

Job	Operation		
	1	2	3
1	1	2	3
2	2	1	3
3	2	1	3
4	1	3	
5	1	2	3

Table 3. Processing time of Astuti (2013)

Job	Operation		
	1	2	3
1	6.95	7.00	3.89
2	13.02	13.02	7.16
3	13.07	14.02	7.41
4	7.67	4.10	
5	7.24	7.63	4.81

In operation 1, engine one is used the most at three times. Therefore, engine 1 has priority. In operation 2, machines 1 and 2 are used as much as two times. Engine 1 already occupies the priority. Then machine 2 has a second priority. In operation 3, the machine used the most is machine three at four times, then machine 3 takes the third priority. Visualization of determining the order of engine priority and St can be seen in Table 4 and Table 5, respectively.

Table 4. Assigning machine priority (step 1)

Machine	Operation		
	1	2	3
M1	3	2	0
M2	2	2	0
M3	0	1	4

Table 5. *St* for every *Mkw* (step 2)

M11	M22	M33
{(1,1,1), (5,1,1), (4,1,1), (2,2,1), (3,2,1)}	{(2,1,2), (3,1,2), (1,2,2), (5,2,2)}	{(4,2,3), (1,3,3), (2,3,2), (5,3,3), (3,3,3)}

On the priority for engine 1, operations (2,2,1) and (3,2,1) cannot be scheduled because they have an unscheduled predecessor. However, operations (2,2,1) and (3,2,1) still belong to *St* because the predecessor does not lie in the engine with the lowest priority (*MkN*). The *pref values* for M11 and M22 can be seen in Table 6. Update the data and proceed into the next stage of *PSt,* where *PSt* = {(1,1,1), (5,1,1), (4,1,1), (2,1,2), (3,1,2), (1,2,2) (5,2,2). Gantt chart visualization when *t* = 0 can be seen in Figure 1.

Table 6. Assigning preference values when *t* = 0

St, t=0	(*i,j,k*)	*Sij*		*CT*			*PTR*		*TOL*		*Total Pref*	Sequence	*PCT*
		Val	*Pref*	*Pijk*	*Val*	*Pref*	*Val*	*Pref*	*Val*	*Pref*			
M11	1,1,1	0	1	6.95	6.95	1	10.89	2	4	1	5	2	3.94
	5,1,1	0	1	7.24	7.24	2	12.44	1	4	1	5	1	5.2
	4,1,1	0	1	7.67	7.67	3	4.1	3	2	2	9	3	3.57
	2,2,1												
	3,2,1												
M22	2,1,2	0	1	13.02	13.02	1	20.18	2	2	1	5	2	7.16
	3,1,2	0	1	13.07	13.07	2	21.43	1	2	1	5	1	8.36
	1,2,2	1419	3	7	21.19	4	3.89	4	1	2	13	4	3.11
	5,2,2	7.24	2	7.63	14.87	3	4.81	3	1	2	10	3	2.82

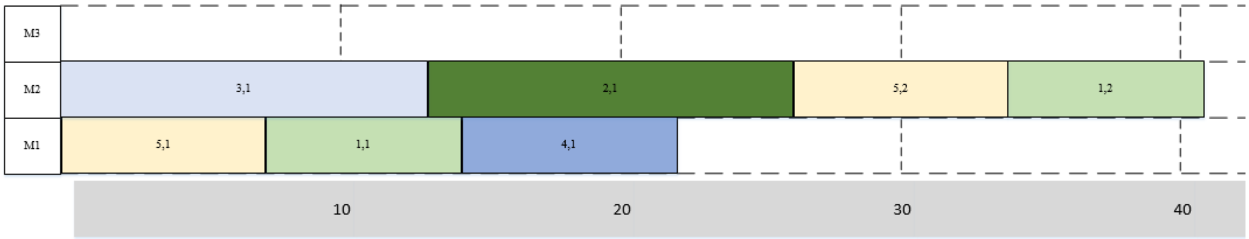

Figure 1. Gantt chart when *t* = 0

Visualization of determining the leftover *St* can be seen in Table 7.

Table 7. Leftover St, t=0 (step 4)

M11	M22
{(2,2,1),(3,2,1)}	{}

The *preference values* when *t*=1 can be seen in Table 8. Update the data and proceed into the next stage of *PSt,* where *PSt* = {(1,1,1),(5,1,1),(4,1,1),(2,1,2),(3,1,2),(1,2,2),(5,2,2),(2,2,1),(3,2,1)}. Gantt chart visualization when *t* = 1 can be seen in Figure 2.

Table 8. Assigning preference values when *t* = 1

St, t=1	*(i,j,k)*	*Sij*		*CT*			*PTR*		*TOL*		*Total Pref*	Sequence	*PCT*
		Val	*Pref*	*Pijk*	*Val*	*Pref*	*Val*	*Pref*	*Val*	*Pref*			
M11	1.1.1	0	1	6.95	6.95	1	10.89	2	4	1	5	2	3.94
	5.1.1	0	1	7.24	7.24	2	12.44	1	4	1	5	1	5.2
	4.1.1	0	1	7.67	7.67	3	4.1	3	2	2	9	3	3.57
	2.2.1	21.86	1	13.02	34.88	1	7.16	2	1	1	5	1	5.86
	3.2.1	26.09	2	14.02	40.11	2	7.41	1	1	1	6	2	6.61

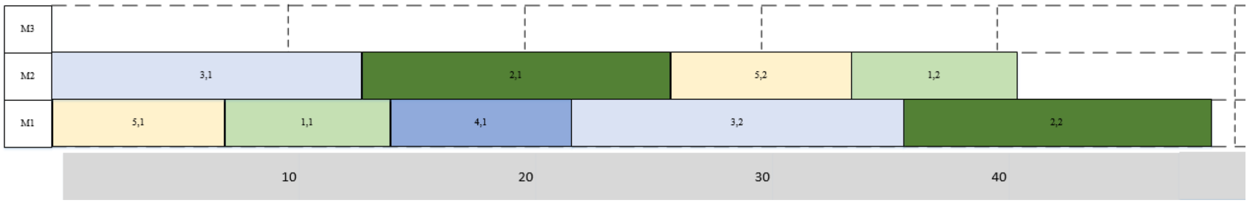

Figure 2. Gantt chart when *t* = 1

Visualization of determining the leftover *St* can be seen in Table 9. The *preference values* when *t* =2 can be seen in Table 10. Update the data and proceed into the next stage of *PSt*, where *PSt* = {(1,1,1), (5,1,1), (4,1,1), (2,1,2), (3,1,2), (1,2,2), (5,2,2), (2,2,1), (3,2,1), (4,2,3), (1,3,3), (2,3,2), (5,3,3,),(3,3,3)}.Gantt chart visualization when *t* = 1 can be seen in Figure 3.

Table 9. Leftover St, t=0

M11	M22
{}	{}

Table 10. Assigning preference values when *t* = 2

St, t=2	*(i,j,k)*	*Sij*		*CT*			*PTR*		*TOL*		*Total Pref*	Sequence	*PCT*
		Val	*Pref*	*Pijk*	*Val*	*Pref*	*Val*	*Pref*	*Val*	*Pref*			
M33	4,2,3	21.86	1	4.1	25.96	1	0	1	0	1	4	1	4.1
	1,3,3	40.72	4	3.89	44.61	4	0	1	0	1	10	4	3.89
	2,3,2	48.9	5	7.16	56.06	5	0	1	0	1	12	5	7.16
	5,3,3	29.49	2	4.81	34.3	2	0	1	0	1	6	2	4.81
	3,3,3	35.88	3	7.41	43.29	3	0	1	0	1	8	3	7.41

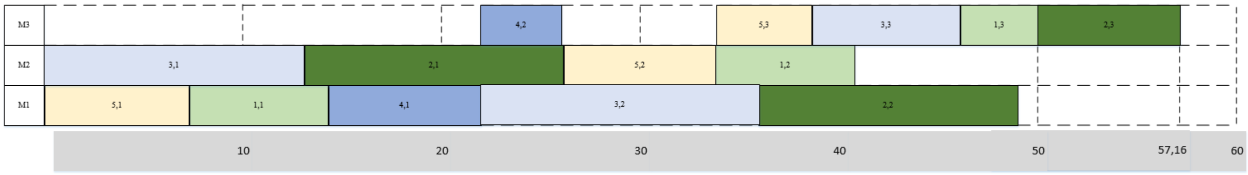

Figure 3. Gantt chart when *t* = 2

5. Results and Discussion

This study is conducted to design a modified model of the ND Algorithm to minimize the makespan of a job. Scheduling using the modified algorithm is carried out in four stages. *First*, calculate the most machines used in each operation. The results of the calculation of the most machines used will determine the priority of the machine. The algorithm modification is designed to sort the machine priorities by selecting the most machines used in each operation. It is intended to allocate operations on high-priority machines in advance. For example, machine priority 1 is the most used machine in operation 1. Therefore, the initial operations which become precedence can be scheduled in advance to shorten the makespan.

Second, scheduling is based on the order of machine priority from the highest priority (1), the second priority, to the last priority (*N*-1). Sorting is carried out by ranking under several criteria, namely the earliest start time, the shortest completion time, the longest remaining processing time, and the highest remaining operations. According to the machine in the process sequence, the ranking results will determine the order of scheduling operations of the job. The criteria used are chosen since they affect the length of the makespan. The highest ranking is given for operations with the earliest start time and the shortest completion time to achieve a shorter makespan. The most prolonged remaining processing time criteria are used to rank operations with the highest total remaining processing time to schedule the operation with the longest remaining processing time in advance. Thus, the next stage will not extend the makespan. According to the concept of a ND algorithm, the operation that is almost ready to be scheduled (scheduled precedence) can be scheduled. This is reflected in the use of the notations (*i,j,k*) and (*i,j,k*)* to distinguish between operations that are ready to schedule and not ready to be scheduled. The remaining total operations are also used as a criterion for scheduling jobs with the most remaining operations time. Therefore, it does not extend the makespan compared to scheduling the final operation on a job in advance. At this stage, there is also an insertion space, namely the gap between job execution. Thus, operations can be scheduled in the insert space to streamline machine usage and minimize makespan. However, the operation to be scheduled must not shift the previously scheduled operation. Therefore, if the insert space is less than the processing time, the operation is scheduled in the following insert space.

Third, after all, operations that can be scheduled on the first to second priority machines have been scheduled. A ranking of the machines with the last priority or the *N* priority will be carried out. The ranking results will determine scheduling operations from jobs on the machine with the last priority. The machine of last priority is executed when all operations on the machine with the highest priority to the second lowest have been scheduled. This is intended to streamline scheduling to schedule the initial operation of a job in advance. This scheduling is following the way machine priority is sorted. The machine occupies the initial priority with the most used number of the initial operation.

Fourth, perform scheduling of the remaining operations (operations with a predecessor on the machine with the last priority) with the same criteria in the second stage, unit all operations from the job have been scheduled and the scheduling is complete. After all the work has been done with the steps described, a *Gantt chart* can be made to visualize the scheduling results for 18 cases of data obtained from various sources.

The proposed algorithm was tested in eighteen cases with a different number of jobs. The result and comparison to previous makespan data could be seen in Table 11.

Table 11. Comparison of makespan results (time-units)

No.	Data source	Makespan	Makespan of Proposed Algorithm	Conclusion
1	Zhang et al. (2010)	13.00	13.00	No difference
2	Willens and Rooda (1994)	22.00	22.00	No difference
3	Bouzidi and Riffi (2014)	26.00	26.00	No difference
4	Kurdi (2015)	23.00	21.00	Shorter
5	Thamilselvan and Balasubramanie (2011)	14.00	13.00	Shorter
6	Lei (2011)	1.61	1.15	Shorter
7	Astuti (2013)	61.15	57.16	Shorter
8	Sinambela (2017)	9,998.00	6,674.00	Shorter
9	Fisher and Thompson (1963)	55.00	55.00	No difference
10	Prasetyo et al. (2015)	49.00	43.00	Shorter
11	Taillard (1993)	435.00	667.00	Longer
12	Taillard (1993)	443.00	698.00	Longer
13	Zang et al. (2019)	213.00	233.00	Longer
14	Zheng et al. (2014)	14.00	17.00	Longer
15	Kuhpfahl (2015)	636.00	1105.00	Longer
16	Kuhpfahl (2015)	586.00	1055.00	Longer
17	Fisher and Thompson (1963)	930.00	1297.00	Longer
18	Lawrence (1984)	666.00	816.00	Longer

The comparison of the settlement results between the proposed algorithm and other case data is used to check the algorithm's reliability in providing a shorter makespan. The proposed algorithm can give the same or better results in the first ten cases out of 18 cases with a reliability percentage of 55.55%. The first ten cases have two to six jobs. In 11 to 18 cases, the proposed algorithm cannot give better or comparable results than makespan in the source case. Cases 11 to 18 have many jobs totaling seven to 10 jobs.

In two jobs, the proposed algorithm gives the same results as the makespan data derived from the data source. It has occurred because the number of jobs is small. Therefore, the possibility of changing positions for scheduling between jobs is limited. Furthermore, in three jobs, the proposed algorithm gave better results in case fourth. In cases with four, five, and six jobs, the proposed algorithm can produce a better makespan than the makespan from the case data, except in the case by Fisher and Thompson (1963) with six jobs; the proposed algorithm produces the same makespan which is 55.00 time-units. A previous study by Thamilselvan and Balasubramanie (2011) used a genetic algorithm, while Kurdi (2015) used the Island Migration Algorithm, like the genetic algorithm using random numbers to determine scheduling sequences. The case of Prasetyo et al. (2015) also applied random numbers to scheduling work. Thus, the resulting schedule does not necessarily produce the best or the shortest makespan.

If the number of jobs is more than seven, the proposed algorithm cannot give similar or better results. The more jobs that must be scheduled and the number of machines available, the more combinations can occur for scheduling. In this case, the proposed algorithm with criteria such as earliest start time, shortest completion time, longest remaining processing time, and the highest number of remaining operations gives a sequence that results in a makespan longer than makespan from the case data. Thus, for cases with a many number of jobs with vast combinations, the algorithm with random numbers can produce a shorter makespan. However, the algorithm with random numbers must go through a certain amount of work to get good results. It is different from the proposed algorithm, which only requires one work, although the results are not necessarily the best.

6. Conclusion

From the conducted research, it could be concluded that the proposed algorithm can provide a shorter makespan compared to Astuti (2013) by 3,99%, from 61,15 time-units to 57,16-time units. Applying the proposed algorithm in eighteen cases with a different number of jobs has a reliability percentage of 55,55%. The proposed algorithm produces the same or shorter makespan for the first ten cases, with fewer jobs between two and six jobs. The proposed algorithm could not provide a better or the same result for the cases with more than seven jobs, which generate a high number of possible combinations. Therefore, it is better to apply an algorithm with random numbers as the generator.

References

Astuti, M., Studi penjadwalan job shop untuk meminimalkan waktu keseluruhan menggunakan pendekatan Algoritma Artificial Immune, *Jurnal Angkasa*, vol. 5, no. 1, pp. 19-28, 2013.

Baker, K. R., and Trietsch, D., *Principles of sequencing and scheduling*. John Wiley & Sons, 2013.

Bouzidi, A., and Riffi, M. E., Cat swarm optimization to solve job shop scheduling problem, Proceeding of the *Third IEEE International Colloquium in Information Science and Technology (CIST)*, October 2014, pp. 202-205.

Fisher, H., and Thompson, G.L., *Probabilistic Learning Combinations of Local Job-Shop Scheduling Rules*, Prentice-Hall, Englewood Cliffs, 225-251, 1963.

Kuhpfahl, J., *Job shop scheduling with consideration of due dates: Potentials of local search-based solution techniques*. Springer, 2015.

Kurdi, M., A new hybrid island model genetic algorithm for job shop scheduling problem. *Computers & Industrial Engineering*, vol. 88, pp. 273-283, 2015.

Lawrence, S., Resource constrained project scheduling: An experimental investigation of heuristic scheduling techniques (Supplement). *Graduate School of Industrial Administration, Carnegie-Mellon University*, 1984.

Lei, D., Simplified multi-objective genetic algorithms for stochastic job shop scheduling, *Applied Soft Computing*, vol. 11, no. 8, pp. 4991-4996, 2011.

Leung, J., *Handbook of Scheduling* (1st ed.). Florida: CRC Press, 2004.

Pinedo, M., *Scheduling* Volume 29, New York, Springer, 2012.

Prasetyo, K., Zaini, E., dan Imran, A., Algoritma penjadwalan no-wait job shop menggunakan Greedy Randomized Adaptive Search Procedure with Fixed Threshold dengan kriteria minimisasi makespan, *Reka Integra,* vol. 3, no. 1, 2015.

Sinambela, Y., Penjadwalan produksi job shop dengan menggunakan metode Shifting Bottleneck Heuristic (SHB), *Jurnal Ilmiah Teknik Industri Prima*, vol. 1, no. 1, 2017.

Taillard, E., Benchmarks for basic scheduling problems, *European Journal of Operational Research*, vol. 64, no. 2, pp. 278-285, 1993.

Thamilselvan, R., and Balasubramanie, P., Analysis of various alternate crossover strategies for genetic algorithm to solve job shop scheduling problems, *European Journal of Scientific Research*, vol. 64, no. 4, pp. 538-554, 2011.

Willems, T. M., and Rooda, J. E., Neural networks for job-shop scheduling, *Control Engineering Practice*, vol. 2, no. 1, pp. 31-39, 1994.

Zang, Z., Wang, W., Song, Y., Lu, L., Li, W., Wang, Y., and Zhao, Y., Hybrid deep neural network scheduler for job-shop problem based on convolution two-dimensional transformation, *Computational Intelligence and Neuroscience*, 2019.

Zhang, Z., Zhang, J., and Li, S., A modified Ant Colony algorithm for the Job Shop Scheduling Problem to minimize makespan, *Proceeding of IEEE International Conference on Mechanic Automation and Control Engineering*, June, 2010, pp. 3627-3630.

Zheng, Y., Lian, L., and Mesghouni, K., Comparative study of heuristics algorithms in solving flexible job shop scheduling problem with condition-based maintenance, *Journal of Industrial Engineering and Management*, vol. *7,* no. 2, pp. 518-531, 2014.

Biography

Nur Ezha Vidawati is a supply chain at FMCG Company. She has been exposed to the entire company value chain, a leading multinational company with more than 2000 brands in over 150 countries. Ms. Vidawati holds a Bachelor of Science degree in Industrial Engineering from Universitas Pembangunan Nasional Veteran, Yogyakarta, Indonesia.

Puryani is a lecturer in Industrial Engineering Department at the Universitas Pembangunan Nasional Veteran, Yogyakarta, Indonesia. She holds a Master of Industrial and Management Engineering from Institut Teknologi Bandung, Indonesia. Mrs. Puryani has taught courses in production planning and control and operational research. Her paper is related to scheduling and production systems.

Apriani Soepardi is an Associate Professor in Industrial Engineering Department at the Universitas Pembangunan Nasional Veteran, Yogyakarta, Indonesia. She earned Doctoral Program in Mechanical Engineering from Universitas Brawijaya Indonesia. In 2017, Mrs.Soepardi was a Sandwich Program student in the Department of Management and

Engineering, Division of Energy Systems, Linköping University, Sweden. She has published journals and conference papers. Her research areas include industrial energy efficiency, agile and sustainable manufacturing, and system modelling.