JOURNAL OF ADVANCED MECHANICAL ENGINEERING APPLICATIONS
e-ISSN: 2716-6201

Vol. 6 No. 2 (2025) 100-109
https://publisher.uthm.edu.my/ojs/index.php/jamea

JAMEA

Performance Investigation of Artificial Bee Colony (ABC) Algorithm for Flexible Job-shop Scheduling Problem (FJSP)

Ten Jia Yee[1], Noor Azizah Sidek[2]*, Salleh Ahmad Bareduan[1]

[1] *Faculty of Mechanical and Manufacturing Engineering, University of Tun Hussein Onn Malaysia, Batu Pahat, 86400, MALAYSIA*
[2] *Centre for Diploma Studies, University of Tun Hussein Onn Malaysia, Pagoh, 84600, MALAYSIA*

*Corresponding Author: noorazizah@uthm.edu.my
DOI: https://doi.org/10.30880/jamea.2025.06.02.010

Article Info

Received: 17 May 2025
Accepted: 21 November 2025
Available online: 31 December 2025

Keywords

Artificial bee colony algorithm, flexible job-shop scheduling, factorial experiment

Abstract

Flexible Job-Shop Scheduling Problem (FJSP) allows a job to be processed on any machine of a set of alternative machines. Therefore, it acts as an extension of the classical Job-Shop Scheduling Problem (JSP). The complexity is portrayed by dealing with both routing and scheduling sub-problems. Major weakness of classical Artificial Bee Colony (ABC) algorithm is shown by trapping in local optimum easily during dealing with complicated multimodal problems such as FJSP despite the proof of ABC's capability to solve FJSP [1], [2]. Therefore, this study aims to study the nature of ABC algorithm. At the same time, the best parameter setting is investigated by using factorial experiment. Brandimarte benchmark (MK01 – MK10) was adopted in the study for the best makespan. The results indicate that a limit value of 5 provides the lowest average makespan for 70% of the benchmark instances. Additionally, selecting MCN = 1000 reduces computational time by approximately 50% while maintaining comparable solution quality.

1. Introduction

Most of the manufacturing companies especially small-scale companies that deal with high-mix and low-volume product type applies job-shop manufacturing layout. The manufacturing layout groups similar equipment or functions together and offers a flexible setup. As a result, the layout leads to complexity of scheduling. Therefore, an effective production scheduling system is essentially important in companies that utilizing job-shop manufacturing layout. Production scheduling consists of managing all production activities including but not limited to material, method, machine and man (4M). The main objective of the scheduling is generally preventing job tardiness which might result in product shipment delay [3]. In this case, minimum production makespan acts as one of the main elements for an effective production scheduling system.

Generally, job-shop scheduling consists of n jobs $J = \{J_1, J_2, \dots, J_i, \dots, J_n\}$ that to be processed on m machine $M = \{M_1, M_2, \dots, M_k, \dots, M_m\}$. Operation O_{ik} represents the operation of job i on machine k. According to the established production schedule, each operation is performed only once on each machine, with a specified processing time. Makespan C_{max} refers to the overall time required to finish all jobs. Equation 1 shows the calculation of makespan.

$$C_{max} = max\{C_i\}, 1 \le i \le n \quad (1)$$

The constraints of Job Shop Scheduling (JSP) can be categorized into three types: precedence, capacity, and release and due dates. Further details on each type of constraint are provided in Table 1.

Table 1 *Classic constraints of JSP* [4]

Precedence	Capacity	Release and due date
i. Processing of each job is based on predetermined order of machine sequences. ii. No limitation on machine orders among different jobs. iii. No precedence constraints among different jobs	i. Machines and jobs are independent of each other ii. Machines cannot be idle when waiting operation to be processed iii. One machine can process only one job at a time iv. One machine can operate one job only once	i. Starting time of a job must not be negative numbers ii. Each operation has a certain processing time iii. Process of each operation cannot be interrupted (pre-emption is not allowed)

The Flexible Job-Shop Scheduling Problem (FJSP) is an extension of the classical Job Shop Scheduling Problem (JSP) because it involves assigning each operation to a chosen machine from a selection of alternative machines [5], [6]. In this scenario, both the routing and scheduling sub-problems must be addressed to create a productive schedule [7]. Routing sub-problem involves assigning machines from a set of alternative machines for a job, where the solution is shown in Equation 2. On the contrary, scheduling sub-problem relates with sequencing each job on machine chosen. Equation 3 displays the solution of the sub-problem. Additional constraints of FJSP as compared to classical JSP are multipurpose machines and unrelated parallel machines [4].

$$y_{j,k,l} \leq a_{j,k,l} \qquad j = 1, \dots, n; \quad k = 1, \dots, m; \quad l = 1, \dots, p_k \tag{2}$$

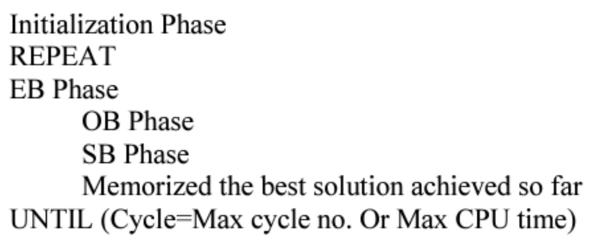
$$\sum y_{i,k,p} = 1 \qquad i = 1, \dots, n; \quad k = 1, \dots, m; \quad p = 1, \dots, p_j \tag{3}$$

Where $a_{j,k,l}$ is the machine capable of processing operations O_{jl}. Value 1 stipulates the machine k able to process the operation and vice versa for value 0. Besides, $y_{i,k,p}$ indicates the result of choosing machine. Value 1 indicates the machine is assigned from the alternative lists of available machines for operation O_{ip} and vice versa for value 0. In this case, the machine-assignment process was constrained by equation 2 to confirm only the machine from the alternative list was chosen. On the other hand, equation 3 ensures that each operation is processed on only one machine.

FJSP is categorized as an NP-hard problem due to its complexity, which involves solving both the optimal sequencing of operations and the machine assignment simultaneously [8]. Due to the weakness of heuristics approaches that capable of dealing with fundamental problems only, metaheuristics approaches were created. General optimization problem solving model was implemented in metaheuristics approaches [9]. Common metaheuristics approaches includes Genetic Algorithms (GA) [10], Particle Swarm Optimization (PSO) [11] and Artificial Bee Colony (ABC) [12], [13]. In this study, ABC algorithm was implemented for solving FJSP. At the same time, the suitable parameter setting was varied to investigate the performance of ABC algorithm.

2. Artificial Bee Colony (ABC) Algorithm

Artificial Bee Colony (ABC) optimization was developed by simulating the foraging behaviour of bees. ABC was implemented in several fields of area, including but not limited to traveling salesman problem [14], spanning tree problems [15], [16], and production scheduling [17-19]. Whole process of ABC algorithm was shown in Fig. 1.



```
Initialization Phase
REPEAT
EB Phase
        OB Phase
        SB Phase
        Memorized the best solution achieved so far
UNTIL (Cycle=Max cycle no. Or Max CPU time)
```

Fig. 1 *Initial food sources are generated in initialization phase. Employed Bees (EB) tests and potentially adopts better food sources in local neighbourhood whereas Onlooker Bees (OB) selects the food sources based on quality and further exploits them. Scout Bees (SB) abandon exhausted food sources to initiate random global exploration for new ones* [20]

2.1 Initialization Phase

The classical ABC algorithm commences with an initialization phase, during which food sources are generated. The algorithm then iterates until the stopping criteria are met. In this study, the stopping criteria are defined by a predetermined trial counter. Consequently, scout bees (SB) carry out the initialization phase after each iteration, provided the trial counter has not yet been reached. Food source generation is described in equation 4 [21].

$$x = L + rand * (U - L) \tag{4}$$

Where x is the food source, *rand* is a random variable between 0 to 1, U and L represent the upper and lower bounds.

2.2 Employed Bees (EB) Phase

The solution parameters were defined by the system before the optimization process began. Following this, the employed bees (EB) undertook the task of exploring food solutions within the specified region. An initial solution was created to direct the bees toward areas with superior food sources. After pinpointing the region that offered high-quality food sources, the EB were dispatched to search for food in the nearby vicinity. The information gathered during this search was subsequently relayed to the onlooker bees (OB) by the EB. Equation 5 represents the mathematical modelling of generating candidate solutions.

$$x_{new} = x + \emptyset * \left(x - x_p\right), \emptyset \in [-1,1] \tag{5}$$

Where x_{new} and x denote the new and old solutions respectively, x_p is the partner solution whereas $\emptyset$ is a random number range from -1 to 1.

2.3 Onlooker Bees (OB) Phase

The onlooker bees (OB) are tasked with selecting food sources based on the quality or fitness of the solutions presented by the employed bees (EB). Following this, a new set of neighbours is generated using the same method as the EB. The selected food sources are typically located in proximity to those chosen by the EB, which increases the likelihood that the OB will identify a high-quality food source. Equation 6 shows the calculation of food source probability according to fitness value.

$$P_i = \frac{fit_i}{\sum fit_i} \tag{6}$$

Where fit_i and SN denote the fitness value of the food source X_i and the number of solutions respectively.

2.4 Scout Bees (SB) Phase

After a predetermined number of trials, food sources that show no potential for improvement will be abandoned. In other words, low-quality food sources will be discarded by the system. Concurrently, scout bees (SB) will be deployed to search for new food sources.

3. Factorial Experimental Design

Factorial design is a statistical technique employed in experimental research to simultaneously investigate the effects of multiple factors. By examining interactions between factors, factorial experimental designs offer insights into how these variables may influence specific outcomes, facilitating a more nuanced interpretation of survey results.

There are three key parameters of the ABC algorithm, that is population size, limit and maximum cycle numbers. Population size can be defined as the number of employed bees or onlooker bees in the colony, limit represents the number of trials of an employed bee or onlooker bee attempts to improve the solution before becoming a scout whereas the maximum cycle number indicates the total number of iterations the algorithm run before terminating. Table 2 shows the key parameters of ABC algorithm implemented by various researchers.

4. Excel VBA Program for ABC Algorithm

The ABC algorithm was implemented in the Excel VBA environment, with the dataset sourced from the Brandimarte benchmark. Brandimarte benchmark (MK01-MK10) was employed in this study. The details of the benchmark datasets are shown in Table 3.

Table 2 *Key parameters of ABC algorithm implemented by other researchers*

Citation	Population Size	Limit	Maximum Cycle Number (MCN)
[22]	5*number of jobs	number of jobs*number of machine/2	10*n*m
[23]	2*number of jobs	number of operations	5*n*m
[24]	10	10	300
[25]	50	50	3000
[26]	100	10-20	30*n*m

Based on table 2, the parameter ranges are defined as MCN = [1000, 2000] and Limit = [5, 10, 15].

Table 3 *Brandimarte dataset details*

Instance	N_{job}	$N_{machine}$	$N_{operation}$	$Max_{machine\ allowed}$	Duration
MK01	10	6	5-7	3	1-7
MK02	10	6	5-7	6	1-7
MK03	15	8	10	5	1-20
MK04	15	8	3-10	3	1-10
MK05	15	4	5-10	2	5-10
MK06	10	15	15	5	1-10
MK07	20	5	5	5	1-20
MK08	20	10	5-10	2	5-20
MK09	20	10	10-15	5	5-20
MK10	20	15	10-15	5	5-20

In contrast to other applications of the ABC algorithm in optimization that utilize the algorithm's formula to determine the optimal makespan value, this study focuses solely on the conceptual framework of the ABC algorithm. This approach was applied to rearrange the job and machine sequences in a Flexible Job Shop Scheduling Problem (FJSP). The insertion and swapping methods [27] were employed to represent the optimization process, wherein the positions of two jobs are exchanged within these mechanisms. Fig. 2 provides a visual representation of these mechanisms for enhanced understanding. In this study, swapping and insertion mechanisms were employed to evaluate the fitness of the population. Following the evaluation and selection of food sources for both the employee bees (EB) and onlooker bees (OB), the best solution identified up to that point was stored in the system.

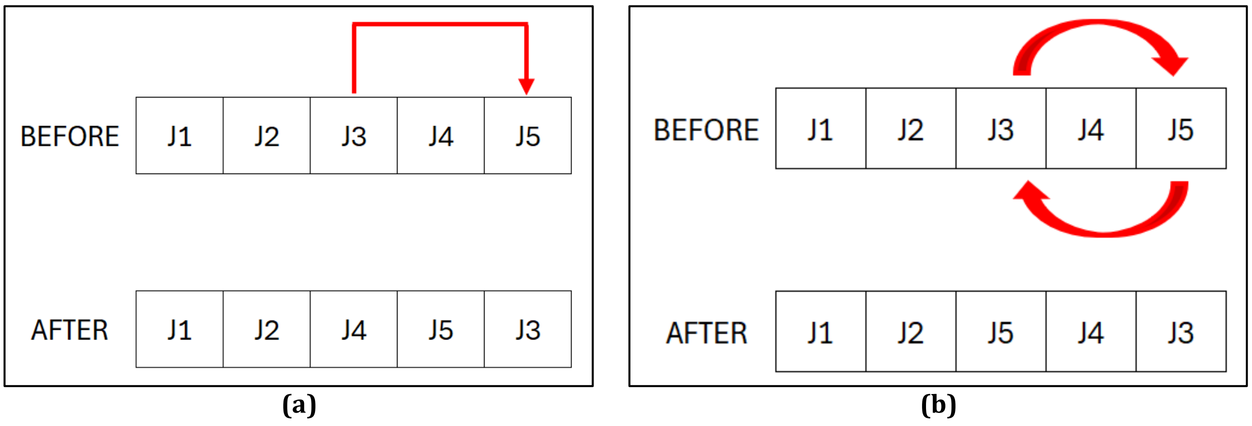


Fig. 2 *Sequences rearranging mechanisms (a) Insertion; (b) Swapping*

5. Parameter Settings of ABC in FJSP

The main scheduling objective in FJSP is to reduce the production makespan. Minimum production makespan guarantees the effectiveness of the manufacturing company and allows higher utilization of work centres.

In the ABC algorithm, if there is no enhancement in results after multiple iterations, the food source will be deemed abandoned. The limit determines the maximum number of iterations permitted before discarding a food source. The limit plays a crucial role in initiating the search in new regions and serves as a mechanism to balance exploration and exploitation efforts. A high limit value encourages the bees to remain in the same region, thereby enhancing their exploitation capabilities. When a food source is abandoned, scout bees are deployed to search for new areas, an activity referred to as exploration. Conversely, a lower limit value promotes exploration, as scout bees will more frequently venture out to discover new sources.

Optimization process will be terminated once the maximum cycle number (MCN) regardless of the completion of the optimization process. A higher maximum cycle number enables the algorithm to perform more iterations, facilitating a more comprehensive exploration of the search space, which may enhance the accuracy of the optimal solution. In contrast, a significantly low cycle number could lead to premature convergence, causing the algorithm to halt before thoroughly investigating potential solutions. At the same time, while prolonged exploration can be advantageous, an excessively high MCN value may elevate the risk of premature convergence to local optima. If the algorithm focuses too much on optimizing a suboptimal solution, it may overlook more promising solutions located in other areas of the search space.

6. Result and Discussion

Each benchmark instance (MK01–MK10) was repeated for 5 times in Excel VBA. The experiment studies the effect of limit and maximum cycle number (MCN) on the effectiveness of the ABC algorithm. Each parameter setting for each benchmark instance was repeated for five times. After that, the consistency of the computational result for each instance was investigated. Figures 3 to 5 visualize the data consistency of each instance for limit values of 5, 10 and 15. From the figures 3 to 5, it was clearly shown that most of the instances (90%) display relatively consistent values across repetitions, suggesting stable performance in these job shop schedules. On the contrary, MK08 shows more variability where deeper analysis was required.

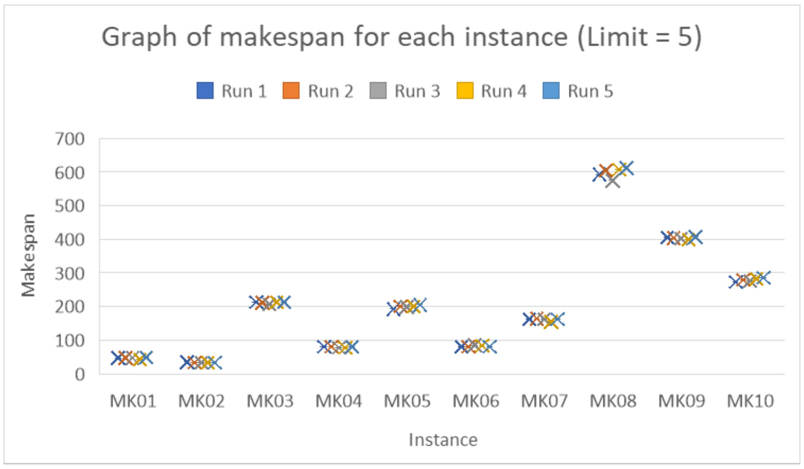


Fig. 3 *All instances except MK08 shows consistent result across repetitions for limit value = 5*

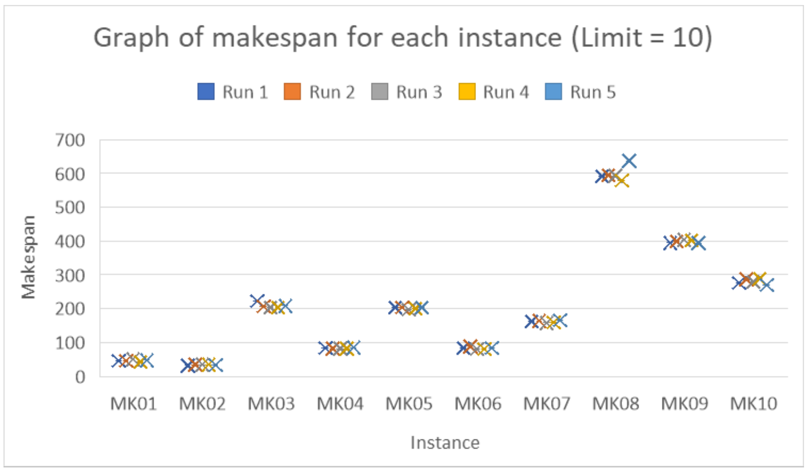


Fig. 4 *All instances except MK08 shows consistent result across repetitions for limit value = 10*

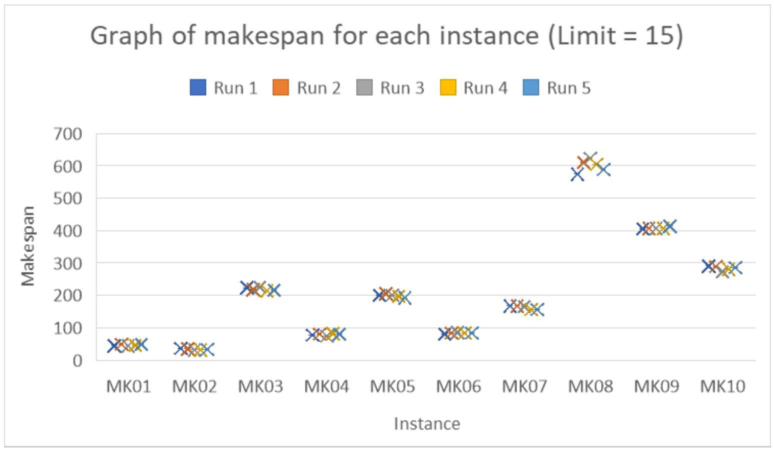


Fig. 5 *All instances except MK08 shows consistent result across repetitions for limit value = 15*

The average value for the result of each instance was calculated and tabulated in Table 4. Shortest average value comparing each limit value for MCN value of 1000 and 2000 was bolded. According to the experimental result, most of the instances (70%) have the minimum mean makespan with limit value of 5. Lower mean indicates that more data has a lower makespan. Hence, limit value of 5 gives the best result for most of the benchmark instances.

The primary limitation of the standard ABC algorithm is its poor exploitation, which results in a high susceptibility to stagnation in local optima. A lower limit value encourages faster abandonment of poor solutions, improving exploration. This prevents the algorithm from spending too much time exploiting suboptimal regions, leading to more efficient convergence.

The comparison data between each MCN value for limit value of 5, 10 and 15 was investigated by calculating T-value. Figure 6 shows the plot of T-value for each limit value. From the figure 6, all of the data is greater than 0.05, which means there is no major statistical difference between the data according to the T-test.

A feasible production scheduling requires balance between the best result and the computational time. Long computational time greatly reduce the effectiveness of the scheduling despite the good result. Total computational time repeated five times for each instance was recorded. After that, the average data of the total computational time for limit value of 5, 10 and 15 for each MCN was calculated and recorded in Table 5.

Table 4 *Comparison of computational result between various limit and MCN*

Instance	Mean for MCN = 1000 Limit =5	Mean for MCN = 1000 Limit =10	Mean for MCN = 1000 Limit =15	Mean for MCN = 2000 Limit =5	Mean for MCN = 2000 Limit =10	Mean for MCN = 2000 Limit =15
MK01	**46.4**	46.8	46.8	**46**	48.2	48
MK02	**33.4**	**33.4**	33.6	34	33.6	**33.4**
MK03	211.4	**209.2**	219.6	**209.4**	215.4	221.4
MK04	**79.2**	83	79.4	**79.4**	83.6	79.8
MK05	199.6	201.2	**199**	**197.8**	199.2	202
MK06	**81.8**	83.4	83.2	83	**82.4**	85.8
MK07	**161.2**	162	163	**160.6**	162.4	160.8
MK08	**598.4**	599.4	600.4	**589.6**	614	598.2
MK09	404.2	**399**	408.8	405.4	**396.8**	409.6
MK10	**279.2**	280.6	283.8	**277.6**	277.8	283.2

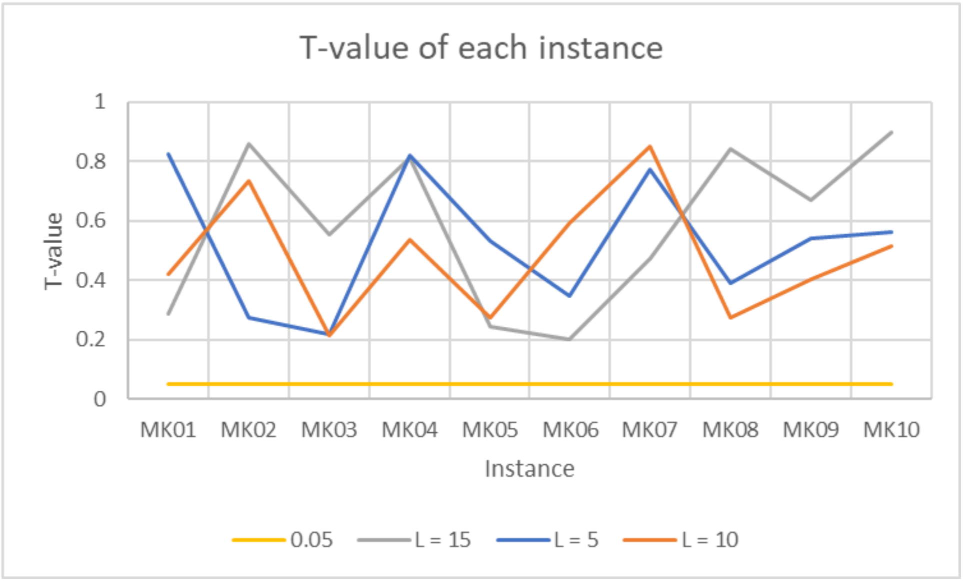


Fig. 6 *T-value of ten instances for each limit value*

Table 5 *Average total computational time for various MCN*

Instance	Average total computation time for MCN = 1000	Average total computation time for MCN = 2000	Time difference (%)
MK01	8.69	16.54	47.48
MK02	8.90	18.03	50.62
MK03	33.68	66.41	49.29
MK04	15.03	28.99	48.15
MK05	13.97	29.35	52.39
MK06	48.39	95.87	49.53
MK07	13.55	26.52	48.91
MK08	56.46	113.13	50.10
MK09	62.07	123.86	49.89
MK10	81.17	161.33	49.69

Table 5 shows that by choosing MCN value of 1000, average time saving of 49.6% can be achieved. Besides, Figure 6 confirmed that both MCN value of 1000 and 2000 may generate feasible data. Therefore, the suitable parameter setting for this dataset is Limit = 5 and MCN = 1000.

7. Conclusion

The relationship between limit and MCN on performance of the ABC algorithm was investigated in this paper. Limit value and MCN value were varied to study the effect on the makespan and compromise between exploration and exploitation. The results were evaluated for consistency to validate the appropriateness of the parameter settings. 90% of the instances show a consistent result. Subsequently, the average makespan values were compared to identify the most suitable parameter settings for the limit value. The experimental results indicate that a limit value of 5 yields the lowest average makespan for the majority of instances. T-test value was then calculated and concluded that there is no major difference between the data with MCN value of 1000 and 2000. The total computational time was subsequently recorded to assess the effectiveness of the scheduling algorithm. The results demonstrate that a MCN value of 1000 produces effective solutions with reduced computational time, facilitating feasible production scheduling. This study should be expanded to investigate the effects of population size and various manufacturing environments.

Acknowledgement

This research was supported by Universiti Tun Hussein Onn Malaysia (UTHM) through Tier 1 (Vot Q944).

Conflict of Interest

Authors declare that there is no conflict of interests regarding the publication of the paper.

Author Contribution

The authors are responsible for the study conception, research design, data collection, data analysis, result interpretation and manuscript drafting.

References

[1] Han, W., Feng, Z. & Cai, J. (2023) Dynamic scheduling study of flexible job shop considering machine failure rate, Proceedings of ISEMSS, Atlantis Press SARL, doi: 10.2991/978-2-38476-126-5_6

[2] Zhang, Z., Fu, Y., Gao, K., Pan, Q. & Huang, M. (2024) A learning-driven multi-objective cooperative artificial bee colony algorithm for distributed flexible job shop scheduling problems with preventive maintenance and transportation operations, Computers & Industrial Engineering, 196, 110484, Oct. 2024, doi: 10.1016/j.cie.2024.110484

[3] Isa, N. A., Sidek, N. A., Bareduan, S. A. & Nawawi, A. (2024) Optimization of new constructive heuristic algorithms for permutation flow shop scheduling problem, Journal of Information System and Technology Management, 9(37), 203–219, doi: 10.35631/JISTM.937016

[4] Abdolrazzagh-Nezhad, M. & Abdullah, S. (2017) Job shop scheduling: classification, constraints and objective functions, International Journal of Computer and Information Engineering, 11(4), 429–434, [Online]. Available: https://waset.org/publications/10006691/job-shop-scheduling-classification-constraints-and-objective-functions

[5] Lingkon, M. L. R. & Dash, A. (2024) Multi-objective flexible job-shop scheduling in hospital using discrete particle swarm optimization algorithm with adaptive inertia weight (DPSO-AIW), Journal of Project Management, 9(September), 0–16, doi: 10.5267/j.jpm.2024.7.007

[6] Yuan, Y. & Xu, H. (2013) Flexible job shop scheduling using hybrid differential evolution algorithms, Computers & Industrial Engineering, 65, 246–260, doi: 10.1016/j.cie.2013.02.022

[7] Gao, K., Cao, Z., Zhang, L., Chen, Z., Han, Y. & Pan, Q. (2019) A review on swarm intelligence and evolutionary algorithms for solving flexible job shop scheduling problems, IEEE/CAA Journal of Automatica Sinica, 6(4), 904–916, doi: 10.1109/JAS.2019.1911540

[8] Xu, S., Li, Y. & Li, Q. (2024) A deep reinforcement learning method based on a transformer model for the flexible job shop scheduling problem, Electronics, 13(18), 3696, Sep. 2024, doi: 10.3390/electronics13183696

[9] Alorf, A. (2023) A survey of recently developed metaheuristics and their comparative analysis, Engineering Applications of Artificial Intelligence, 117(Nov. 2022), 105622, doi: 10.1016/j.engappai.2022.105622

[10] Onwuachu, U., Amaefule, I. A. & Miriam, R. (2024) Fuzzy-genetic model for load balancing and machine utilization in job shop scheduling, International Research Journal of Modern Engineering and Technology Science, 6(9), 1017–1026, doi: 10.56726/IRJMETS61521

[11] Zhang, Q. & Zhang, B. (2024) Research on dynamic flexible job shop scheduling problem based on discrete particle swarm optimization with readjustment strategy, Proceedings of SPIE, 13261, 1093–1097, Sep. 2024, doi: 10.1117/12.3046596

[12] Osman, M. A. H., Rashid, M. F. F. A., Mohamed, N. M. Z. N. & Mu'tasim, M. A. N. (2024) Energy-aware scheduling optimization in hybrid flow shops using artificial bee colony algorithm, Journal of Mechanical Engineering Science, 18(3), 10171–10180

[13] Youssefi, K. A., Gojkovic, M. & Schranz, M. (2024) Artificial bee colony algorithm: bottom-up variants for the job-shop scheduling problem, Proceedings of International Conference on Simulation and Modeling Methodologies, Technologies and Applications (Simultech), pp. 103–111, doi: 10.5220/0012765900003758

[14] Tong, M., Peng, Z. & Wang, Q. (2025) A hybrid artificial bee colony algorithm with high robustness for the multiple traveling salesman problem with multiple depots, Expert Systems with Applications, 260, 125446, Jan. 2025, doi: 10.1016/j.eswa.2024.125446

[15] Pan, X., Peng, D. & Li, S. (2024) Quantum binary improved artificial bee colony algorithm to solve the spanning tree construction problem in vehicular ad hoc network, IEEE Internet of Things Journal, pp. 1–1, doi: 10.1109/JIOT.2024.3412692

[16] Singh, K. & Sundar, S. (2021) Artificial bee colony algorithm using permutation encoding for the bounded diameter minimum spanning tree problem, Soft Computing, 25(16), 11289–11305, Jul. 2021, doi: 10.1007/s00500-021-05913-z

[17] Zhu, Q., Gao, K., Huang, W., Ma, Z. & Slowik, A. (2024) Q-learning-assisted meta-heuristics for scheduling distributed hybrid flow shop problems, Computers, Materials & Continua, 80(3), 3573–3589, doi: 10.32604/cmc.2024.055244

[18] Ma, R., Gui, J., Wen, J. & Guo, X. (2024) Chaos quantum bee colony algorithm for constrained complicate optimization problems and application of robot gripper, Soft Computing, 28(19), 11163–11206, Jul. 2024, doi: 10.1007/s00500-024-09877-8

[19] Li, Y., Zhou, Y. & Wang, Y. (2024) Optimization of production scheduling for turbocharger manufacturing reprocessing, Proceedings of 2024 International Conference on Energy and Electrical Engineering, pp. 1–4, Jul. 2024, doi: 10.1109/EEE59956.2024.10709701

[20] Sidek, N. A., Bareduan, S. A. & Nawawi, A. (2019) Performance investigation of artificial bee colony (ABC) algorithm for permutation flowshop scheduling problem (PFSP), Journal of Physics: Conference Series, 1150(1), doi: 10.1088/1742-6596/1150/1/012060

[21] Liu, Y. F. & Liu, S. Y. (2013) A hybrid discrete artificial bee colony algorithm for permutation flowshop scheduling problem, Applied Soft Computing, 13(3), 1459–1463, Mar. 2013, doi: 10.1016/j.asoc.2011.10.024

[22] Li, J. Q., Pan, Q. K. & Gao, K. Z. (2011) Pareto-based discrete artificial bee colony algorithm for multi-objective flexible job shop scheduling problems, International Journal of Advanced Manufacturing Technology, 55(9–12), 1159–1169, doi: 10.1007/s00170-010-3140-2

[23] Wang, L., Zhou, G., Xu, Y., Wang, S. & Liu, M. (2012) An effective artificial bee colony algorithm for the flexible job-shop scheduling problem, International Journal of Advanced Manufacturing Technology, 60(1–4), 303–315, doi: 10.1007/s00170-011-3610-1

[24] Li, J. Q., Pan, Q. K. & Tasgetiren, M. F. (2014) A discrete artificial bee colony algorithm for the multi-objective flexible job-shop scheduling problem with maintenance activities, Applied Mathematics Modelling, 38(3), 1111–1132, doi: 10.1016/j.apm.2013.07.038

[25] Gao, K. Z., Suganthan, P. N., Chua, T. J., Chong, C. S., Cai, T. X. & Pan, Q. K. (2015) A two-stage artificial bee colony algorithm scheduling flexible job-shop scheduling problem with new job insertion, Expert Systems with Applications, 42(21), 7652–7663, doi: 10.1016/j.eswa.2015.06.004

[26] Nucera, D. D., Negri, E. & Fumagalli, L. (2022) A timetabling heuristic and a hybrid self-learning artificial bee colony for the no-wait flexible job-shop scheduling, SSRN Electronic Journal, doi: 10.2139/ssrn.4080717

[27] Sidek, N. A., Bareduan, S. A., Nawawi, A. & Yee, T. J. (2024) Development of guided artificial bee colony (GABC) heuristic for permutation flowshop scheduling problem (PFSP), Journal of Advanced Research in Applied Sciences and Engineering Technology, 33(3), 393–406, doi: 10.37934/araset.33.3.393406