PROGRESS IN ENGINEERING APPLICATION AND TECHNOLOGY
e-ISSN: 2773-5303

PEAT

Vol. 5 No. 1 (2024) 369-377
https://publisher.uthm.edu.my/periodicals/index.php/peat

Optimization of Permutation Flowshop Scheduling Problem (PFSP) Using First Sequence Artificial Bee Colony (FSABC) Algorithm

Lailatul Fikri Razali[1], **Azli Nawawi***[1]

[1] *Department of Mechanical Engineering Technology, University Tun Hussien Onn Malaysia, Hab Pendidikan Tinggi Pagoh, KM1, Jalan Panchor, Muar, Johor, 84600, Malaysia*

*Corresponding Author: azle@uthm.edu.my

DOI: https://doi.org/10.30880/peat.2024.05.01.039

Article Info

Received: 26 December 2023
Accepted: 18 January 2024
Available online: 15 June 2024

Keywords

Artificial Bee Colony (ABC) Algorithm, Permutation Flowshop Scheduling Problem (PFSP)

Abstract

The flowshop is a widely used production system, with efforts to enhance its functionality. Employed Bees and Onlooker Bees are two key components of this system. Employed Bees are a type of bee that are used to produce honey, while Onlooker Bees are used to produce honey. The flowshop uses NEH heuristics, which are heuristics used to find superior results. This study aims to improve NEH by using an improved version of the slow-to-converge Artificial Bee Colony (ABC) method. The Total Greedy method (5+0+0 & 10+0+0) was chosen for Employed Bees and Onlooker Bees, and the First Sequence Artificial Bee Colony (FSABC) was proposed. The study compared the performance of NEH, ABC, and FSABC, revealing that FSABC may achieve the required results, but ABC often produces inconsistent results and inferior data.

1. Introduction

Operations research has extensively explored the well-known problem of flowshop scheduling, employing various approaches such as heuristics, metaheuristics, and precise algorithms to address this issue [1]. Genetic algorithms and exact algorithms are advanced algorithms used in big situations, but they may be slow and not suitable for all scenarios [2].

Derviş Karaboga introduced the Artificial Bee Colony (ABC) optimization, a metaheuristic algorithm based on honeybee foraging behavior. ABC aims to create a multi-agent system, a colony of artificial bees, capable of solving combinatorial optimization problems, exhibiting partial similarities and differences from natural bee colonies. However, this method got slow to converge the results and took a long time to produce it [3].

There are three main objectives of this research study which are to develop the coding for Permutation Flowshop Scheduling Problems (PFSP). Next, is to develop the coding for the First Sequence Artificial Bee Colony (FSABC) algorithm for optimizing PFSP and the last is to assess the performance of FSABC algorithm against the current ABC algorithm in the context of optimizing PFSP.

2. Methodology

This research study uses a new ABC algorithm which was conducted in Microsoft Excel VBA. Then the new ABC algorithm which is First Sequence Artificial Bee Colony Algorithm will be validated with NEH as the benchmark for the performance test. This research study will also compare the performance and improvement between normal ABC and FSABC. Taillard, 1993 datasets have been used to look consistent with other studies in the same field.

2.1 Investigate the current ABC

The research initiates with the ABC algorithm's initialization step, where the system generates initial food sources. This step continues iteratively until a stopping mechanism is met. Scout bees handle the startup step, setting the trial counter (TC). The three main phases—Employed Bee (EB), Onlooker Bee (OB), and Scout Bee (SB)—are executed. Employed bees search for food sources within defined regions, starting from an initial solution. The algorithm operates in the PFSP, altering the work sequence in the flowshop, using job rearrangements to create a flowshop job arrangement instead of numerical values.

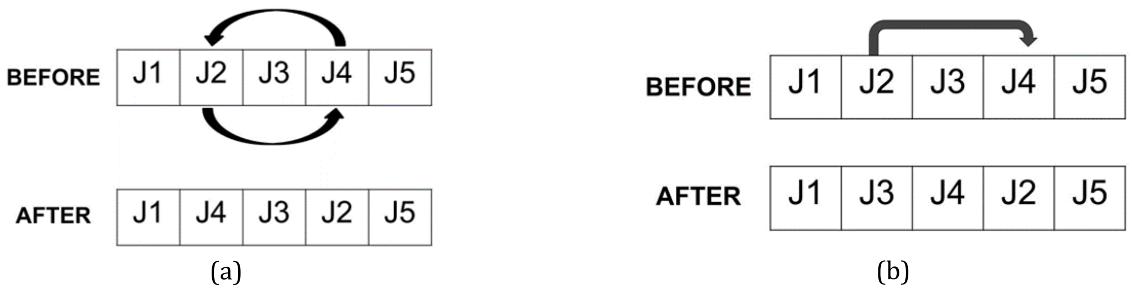


Fig. 1 (a) *The method for swapping job locations in the flowshop scheduling;* (b) *The insert method between jobs*

The selection process involves onlooker bees (OB) and employed bees (EB) identifying food sources based on their characteristics. OB's input includes fitness value and mapping computation. After the selection process, OB generates new neighbors using the same method as EB, favoring high-quality sources. In ABC, if there is no improvement in the values after a predefined number of trials, the bees will forsake the food source. The system will reject the low-quality food sources before releasing the scout bees (SB) to look for other food sources.

2.2 Development of Excel VBA Program & Greedy Variation of locations for OB.

The research developed Excel VBA coding for ABC, dividing it into two sections: Employed Bees (EB) and Onlooker Bees (OB). The EB section manages employed bees' behavior, generating random insert operations within specified ranges and storing the best solution found. It updates the solution's fitness measure if improvements are made. Similarly, the OB section controls onlooker bees, generating random insert operations and checking differences but does not explicitly store or update the best solution found. The study employed a variation for onlooker bees' locations, using 10 bees for 20 jobs and 5 machines, and 20 bees for 50 jobs and 5 machines. Initially, the system created a random solution, with employed bees seeking improvements by adjusting work sequences, using the first task as a fixed reference point.

2.3 Development of First Sequence Artificial Bee Colony (FSABC) Heuristic

The study utilized the original ABC method, with significant adjustments made to the original answer. These modifications aim to accelerate convergence and identify areas with high-quality solutions. The First Sequence Artificial Bee Colony (FSABC) approach was used to promote the bee population and provide high-quality solutions. The ABC algorithm was adapted to study high-quality solutions using the first job sequence strategy and the NEH layout solution. This approach helps identify areas with high-quality solutions faster than the original ABC algorithm.

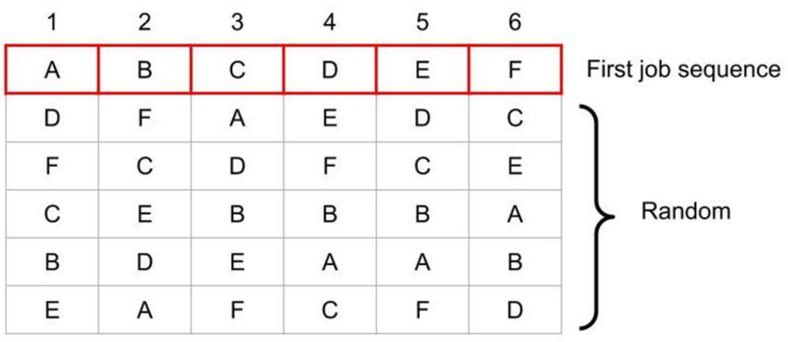


Fig. 2 *The option of employing a first job sequence method*

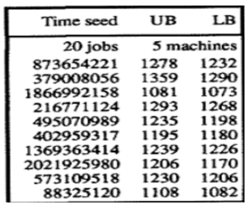
The pseudocode for the First Sequence Artificial Bee Colony (FSABC) algorithm involves initializing parameters and a population. It evaluates solutions, iterates through cycles, using employed and onlooker bees to generate, evaluate, and select the best solutions. It replaces abandoned solutions with scouts, memorizes the best solution, and refines it using a job sequence approach. This iterative process continues until reaching the maximum cycle number (MCN). The algorithm aims to optimize solutions by refining them through various bee-inspired mechanisms and memorizing the best-found solution.

2.4 Validation Stage

The NEH method was used as the performance benchmark, while comparing the standard ABC and FSABC approaches. Datasets from Taillard (1993) were utilized to align with prior research, showcasing a subset in **Figure 3**. These data scopes by E. Taillard (1993) typically outline dataset characteristics in this section.

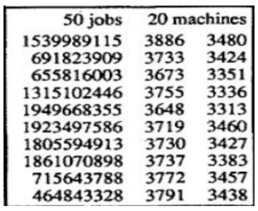
i. Twenty jobs, five machines (20J5M).

ii. Fifty jobs, five machines (50J5M).



Time seed	UB	LB
20 jobs	5 machines	
873654221	1278	1232
379008056	1359	1290
1866992158	1081	1073
216771124	1293	1268
495070989	1235	1198
402959317	1195	1180
1369363414	1239	1226
2021925980	1206	1170
573109518	1230	1206
88325120	1108	1082

(a)



50 jobs	20 machines	
1539989115	3886	3480
691823909	3733	3424
655816003	3673	3351
1315102446	3755	3336
1949668355	3648	3313
1923497586	3719	3460
1805594913	3730	3427
1861070898	3737	3383
715643788	3772	3457
464843328	3791	3438

(b)

Fig. 3 (a) *The sample of datasets from (Taillard, 1993) for 20 Jobs 5 Machines;*
(b) *The sample of datasets from (Taillard, 1993) for 50 Jobs 5 Machines*

The 20J5M requires five employed bees (EB) and five onlooker bees (OB). In the 50J5M, there are ten employed bees (EB) and ten onlooker bees (OB).

Table 1 *The total number of EB and OB for each flowshop configuration (Colony Size)*

Flowshop Setting	No. of Employed	No. of Onlooker	Trial Counter	Iteration Limit

	Bees (EB)	Bees (OB)		
20 jobs 5 machines	5	5	10	500 1000 2000
50 jobs 5 machines	10	10	20	500 1000 2000

2.5 Performance Assessment of FSABC Heuristic against ABC Algorithm

The percentage improvement achieved by the FSABC algorithm over the NEH algorithm is calculated using the formula:

$$Percentage\ of\ improvement = \frac{FSABC - NEH}{NEH} \times 100\% \tag{1}$$

The equation measures FSABC's efficiency compared to NEH, showing the percentage improvement or deviation in solution quality. This same equation applies to assess ABC's improvement based on NEH, using their solution qualities accordingly.

2.6 Generating Charts

This study compares the outcomes based on two parameters which is The Makespan Value and The Percentage of Improvements over the Standard ABC Algorithm. Microsoft Excel spreadsheets are used to create and arrange the findings. The author will produce the chart using Microsoft Excel spreadsheets when it has been generated and structured methodically. Microsoft Excel is the simplest and most straightforward way to construct charts from data.

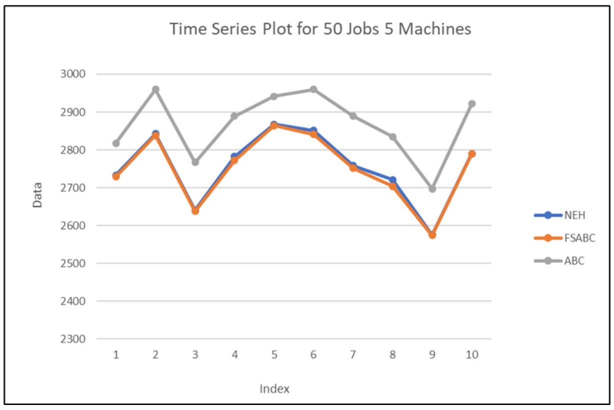
The Design of Experiments (DOE) approach is used in the study to illustrate the behaviour of all elements (parameters) and responses (outcomes).

3. Results and Discussion

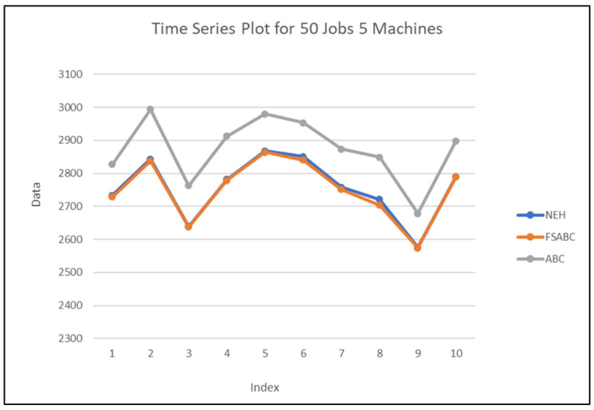
This section presents study results using tables and charts, detailing FSABC advancements in each phase. Taillard Benchmark datasets were used for comparability. VBA in Ms Excel facilitated the trials, while the experiment data was statistically analyzed using Ms Excel. The author segmented the findings into sections such as Performance Comparison and Performance Statistics for each job and iterations (Validation Stage), aligning with the research goal of FSABC development.

3.1 Analysis for 50 jobs 5 machines

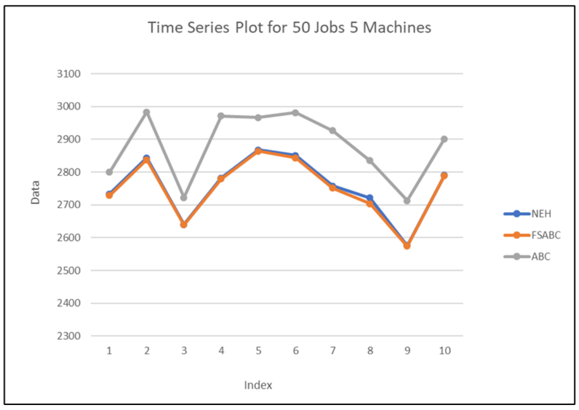
The following figure displays the makespan values for ten datasets, featuring three competitors (NEH, FSABC, and ABC) across iterations of 2000, 1000, and 500. Currently, NEH stands as the top performer and serves as the primary benchmark for comparison. Subsequently, the datasets are converted into a percentage to showcase improvements concerning NEH. Each adversary is visually compared with NEH to illustrate their respective performances. A positive value indicates a superior makespan compared to NEH, while a negative value suggests the opposite.



(a)



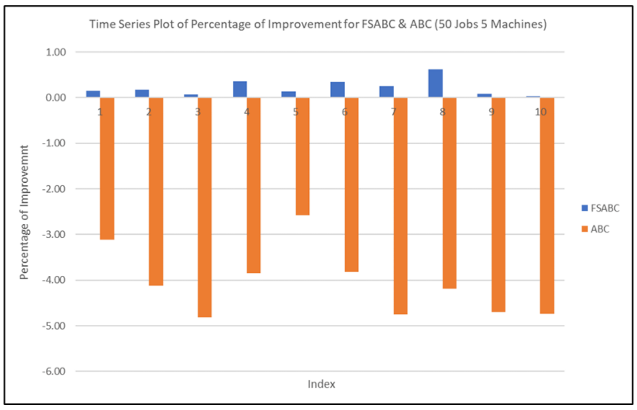
(b)



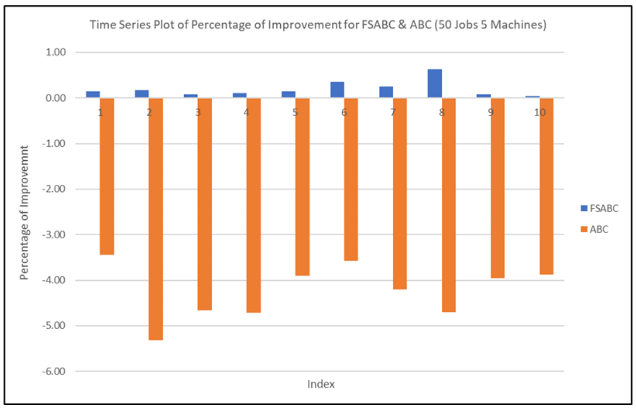
(c)

Fig. 4 (a) *Time Series Plot of Makespan Values for all datasets for 2000 iterations;,* (b) *Time Series Plot of Makespan Values for all datasets for 1000 iterations;* (c) *Time Series Plot of Makespan Values for all datasets for 500 iterations;*

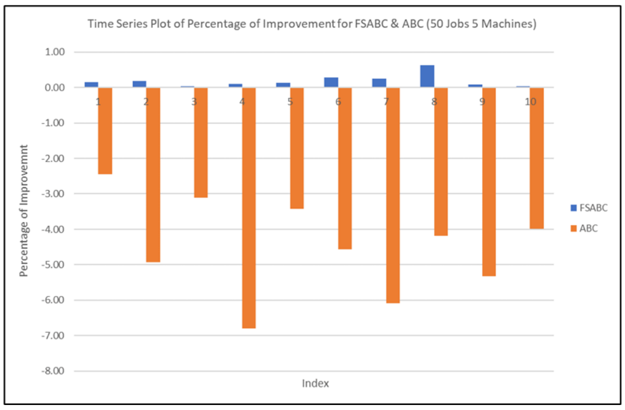
The graph shows in Fig.4 that although the patterns in each technique are similar, the values on the line plots for FSABC and NEH are almost the same in 2000, 1000 and 500 iterations. The close equivalence of the FSABC is reassuring, since the NEH is the main benchmark used in this comparison. A closer look reveals that datasets 7 and 8 perform better than NEH & ABC (refer to Fig. 4). This is quite beneficial to the author because the study's results indicate a somewhat better picture than that of ABC and NEH.



(a)



(b)



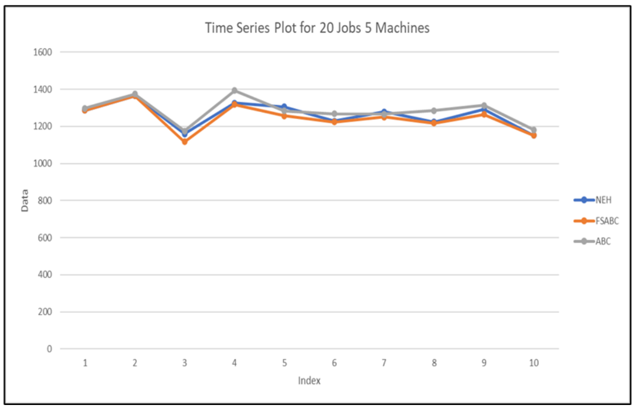
(c)

Fig. 5 (a) *Time Series Plot of Percentage of Improvement (Comparison with NEH) 2000 iterations;* (b) *Time Series Plot of Percentage of Improvement (Comparison with NEH) 1000 iterations;* (c) *Time Series Plot of Percentage of Improvement (Comparison with NEH) 500 iterations;*

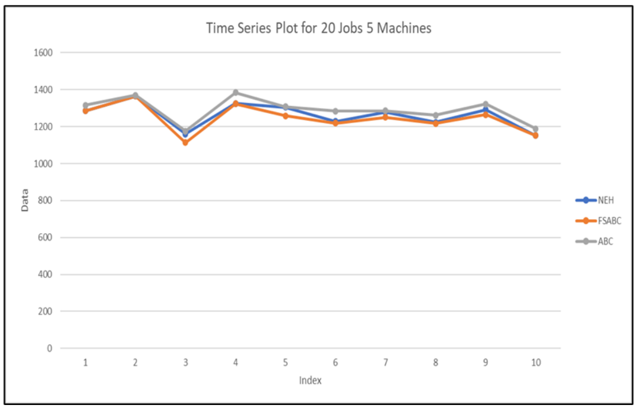
The negative value in Fig. 5 above indicates that ABC appears to be performing the worst, with its makespan value surpassing NEH's value for 2000, 1000 and 500 iterations. When compared to FSABC, all datasets exhibit the same performance level thus far. Slightly better performance is displayed by FSABC. As previously, the graphic demonstrates that FSABC is superior to ABC.

3.2 Analysis for 20 jobs 5 machines

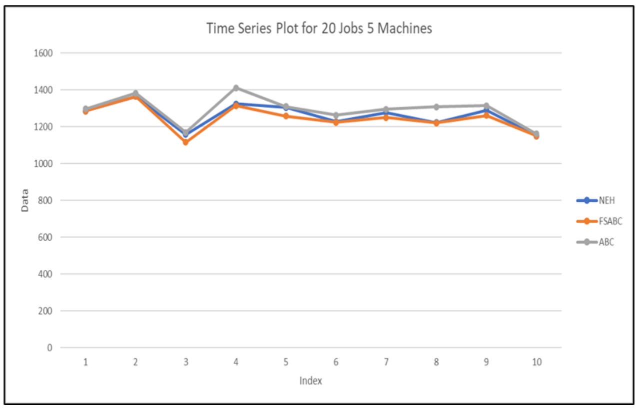
The figure displays makespan values for ten datasets comprising three competitors (NEH, FSABC, and ABC), each involving 20 Jobs and 5 Machines. NEH remains the main benchmark due to its superior performance. The datasets are then converted into a percentage representing improvements relative to NEH. Visual comparisons are made with NEH, where positive values indicate superior makespan, and negative values suggest the opposite.



(a)



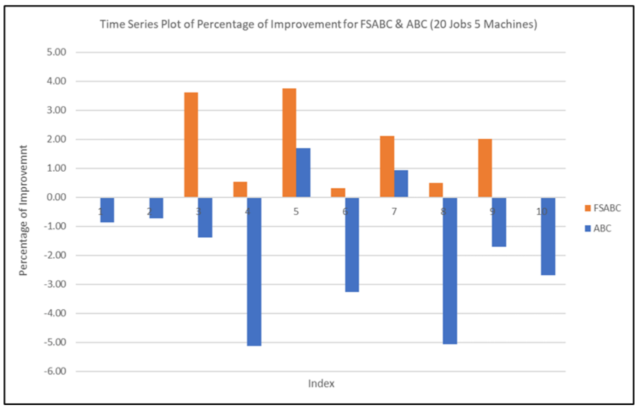
(b)



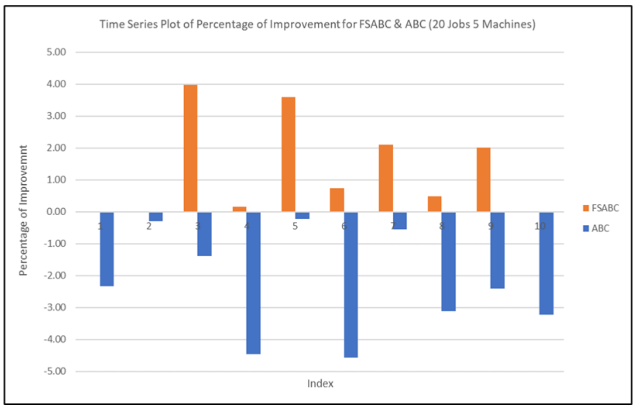
(c)

Fig. 6 (a) *Time Series Plot of Makespan Values for all datasets 2000 iterations;* (b) *Time Series Plot of Makespan Values for all datasets 1000 iterations;* (c) *Time Series Plot of Makespan Values for all datasets 500 iterations;*

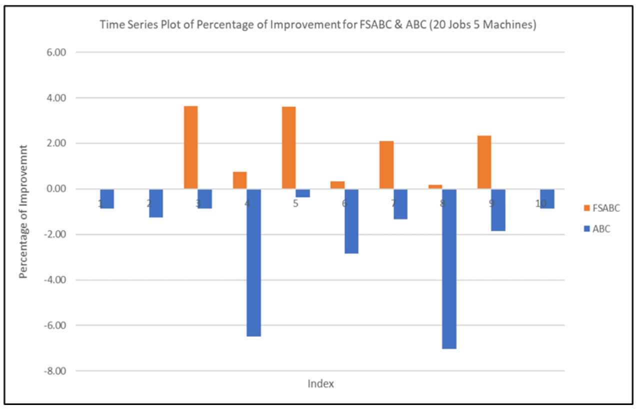
The figure illustrates that although every method presents a comparable pattern, the values on the line plots for NEH, FSABC, and ABC is almost the same for 2000, 1000 and 500 iterations. The close equivalence of the FSABC is reassuring, since the NEH is the main benchmark used in this comparison. After more examination, datasets 3rd, 5th, 7th, and 9th show that FSABC performs better than NEH & ABC (refer to Fig. 6). This is quite beneficial to the author because the study's results indicate a somewhat better picture than that of ABC and NEH.



(a)



(b)



(c)

Fig. 7 (a) *Time Series Plot of Percentage of Improvement (Comparison with NEH) 2000 iterations;* (b) *Time Series Plot of Percentage of Improvement (Comparison with NEH) 1000 iterations;* (c) *Time Series Plot of Percentage of Improvement (Comparison with NEH) 500 iterations;*

Fig. 7 shows that, out of all the data, FSABC produced the best results for 2000, 1000 and 500 iterations. It is far more effective than ABC. The figures show a significant percentage improvement, with the ABC having the poorest set of percentage errors and data 3rd having a greater percentage value than the others. This is because ABC's slower convergence problem makes it take longer for the system to provide high-quality outputs.
The chapter details performance comparisons and statistics during each work iteration (Validation Stage). Emphasizing the importance of balancing exploration and exploitation, it highlights the reliability of the ten-cycle setting with lower error percentages. It explores parameters like bee positions, cycles (TC), and iterations, favoring the 5+0+0 & 10+0+0 (Total Greedy) approach. More iterations generally yield better outcomes but result in longer runtimes. Comparing FSABC with NEH and ABC in the validation stage reveals FSABC's contradictory yet mostly superior performance. The study's resolution of ABC's convergence issue supports these findings.

4. Conclusion

The study proposes an enhanced ABC algorithm, FSABC, addressing scheduling issues in industrial systems. Divided into three primary parts, the research aims to improve the original ABC's convergence speed. The FSABC heuristic is developed using Greedy bee characteristics within Microsoft Excel VBA. It employs a swap and insert mechanism to optimize PFSP, focusing on NEH layout solutions.

The study extensively compares NEH, ABC, and FSABC performances using Taillard Benchmark datasets. It highlights that increased iterations lead to better outcomes, but longer runtimes. The "5+0+0 & 10+0+0" technique is noted as most effective, concentrating bees on optimal solutions. FSABC shows notable improvement over ABC, yet its results remain inconsistent compared to NEH, emphasizing the significance of small improvements for overall production. The study's resolution of ABC's convergence issue supports these conclusions.

Acknowledgement

The author expresses gratitude and thank for giving opportunities finishing this research paper to the Faculty of Engineering Technology, University Tun Hussien Onn Malaysia.

Conflict of Interest

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

Author Contribution

This journal requires that all authors take public responsibility for the content of the work submitted for review. The contributions of all authors must be described in the following manner:
The authors confirm contribution to the paper as follows:
study conception and design: *Lailatul Fikri Bin Razali, Azli Bin Nawawi*
data collection: *Lailatul Fikri Bin Razali*
analysis and interpretation of results: *Lailatul Fikri Bin Razali, Azli Bin Nawawi*
draft manuscript preparation: *All authors reviewed the results and approved the final version of the manuscript.*

References

[1] Agarwal, A., Colak, S., & Eryarsoy, E. (2006). Improvement heuristic for the flow-shop scheduling problem: An adaptive-learning approach. European Journal of Operational Research, 169(3), 801–815. https://doi.org/10.1016/J.EJOR.2004.06.039
[2] Aggoune, R. (2004). Minimizing the makespan for the flow shop scheduling problem with availability constraints. European Journal of Operational Research, 153(3), 534–543. https://doi.org/10.1016/S0377-2217(03)00261-3
[3] Ahmadizar, F. (2012). A new ant colony algorithm for makespan minimization in permutation flow shops. Computers & Industrial Engineering, 63(2), 355–361. https://doi.org/10.1016/J.CIE.2012.03.015
[4] Andrade, C. E., Silva, T., & Pessoa, L. S. (2019a). Minimizing flowtime in a flowshop scheduling problem with a biased random-key genetic algorithm. Expert Systems with Applications, 128, 67–80. https://doi.org/10.1016/J.ESWA.2019.03.007
[5] Andrade, C. E., Silva, T., & Pessoa, L. S. (2019b). Minimizing flowtime in a flowshop scheduling problem with a biased random-key genetic algorithm. Expert Systems with Applications, 128, 67–80. https://doi.org/10.1016/J.ESWA.2019.03.007
[6] Avci, M. (2023). An effective iterated local search algorithm for the distributed no-wait flowshop scheduling problem. Engineering Applications of Artificial Intelligence, 120, 105921. https://doi.org/10.1016/J.ENGAPPAI.2023.105921
[7] Avrahami, N., & Azar, Y. (2003). Minimizing total flow time and total completion time with immediate dispatching. Annual ACM Symposium on Parallel Algorithms and Architectures, 11–18. https://doi.org/10.1145/777412.777415
[8] Azizoğlu, M., Çakmak, E., & Kondakci, S. (2001). A flexible flowshop problem with total flow time minimization. European Journal of Operational Research, 132(3), 528–538. https://doi.org/10.1016/S0377-2217(00)00142-9
[9] Banharnsakun, A., Achalakul, T., & Sirinaovakul, B. (2011a). The best-so-far selection in Artificial Bee Colony algorithm. Applied Soft Computing, 11(2), 2888–2901. https://doi.org/10.1016/J.ASOC.2010.11.025
[10] Banharnsakun, A., Achalakul, T., & Sirinaovakul, B. (2011b). The best-so-far selection in Artificial Bee Colony algorithm. Applied Soft Computing, 11(2), 2888–2901. https://doi.org/10.1016/J.ASOC.2010.11.025
[11] Becchetti, L., Leonardi, S., Marchetti-Spaccamela, A., & Pruhs, K. (2008). Flow Time Minimization. Encyclopedia of Algorithms, 320–322. https://doi.org/10.1007/978-0-387-30162-4_146
[12] Ben-Daya, M., & Al-Fawzan, M. (1998). A tabu search approach for the flow shop scheduling problem. European Journal of Operational Research, 109(1), 88–95. https://doi.org/10.1016/S0377-2217(97)00136-7
[13] Bhattacharyya, T., Chatterjee, B., Singh, P. K., Yoon, J. H., Geem, Z. W., & Sarkar, R. (2020). Mayfly in Harmony: A new hybrid meta-heuristic feature selection algorithm. IEEE Access, 8, 195929–195945. https://doi.org/10.1109/ACCESS.2020.3031718
[14] Blum, C., & Roli, A. (2003). Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. ACM Computing Surveys, 35(3), 268–308. https://doi.org/10.1145/937503.937505
[15] Cao, D., Chen, M., & Wan, G. (2005). Parallel machine selection and job scheduling to minimize machine cost and job tardiness. Computers & Operations Research, 32(8), 1995–2012. https://doi.org/10.1016/J.COR.2004.01.001