VOL. 11, NO. 10, MAY 2016 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

SCHEDULING ANALYSIS FOR FLOWSHIP USING ARTIFICAL BEE COLONY (ABC) ALGORITHM WITH VARYING ONLOOKER APPROACHES

Nur Fazlinda Binti Mohd Pauzi and Salleh Ahmad Bareduan
Faculty of Mechanical and Manufacturing Engineering, University Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, Johor, Malaysia
E-Mail: hd110165@siswa.uthm.edu.my

ABSTRACT

Artificial Bee Colony (ABC) algorithm is one of the methods used to solve the flowshop scheduling problem. In order to further investigate the ABC capabilities, we proposed a methodology with the capability of manipulating the onlooker bee approaches in ABC Algorithm for solving flowshop scheduling problem. This paper reviews the analysis of the performance of the ABC algorithm through three different onlooker approaches i.e. method 3+0+0 (three onlooker bees are dedicated to the best employed bee), method 2+1+0 (two onlooker bees are dedicated to the best employed bee and one onlooker bee is dedicated to second best employee bee) and method 1+1+1 (one onlooker bee is dedicated to each employed bee). The simulation results indicated that method 2+1+0 produces best result at low iterations of 102 and below. At high iterations of 204 and above method 3+0+0 dominates the best performance.

Keywords: artificial bee colony algorithm, flowshop scheduling problem, onlooker bee, employee bee, makespan.

INTRODUCTION

Scheduling is a decision making process used in manufacturing and services industries. One of the main objectives of scheduling applications in both industries is to solve the permutation flowshop scheduling problem through minimizing the maximum completion time or makespan. Several heuristics have been developed to solve the permutation flowshop problem such as NEH Heuristics (Nawaz *et al*. 1983), Gupta Heuristics (Gupta 1971), Palmer Heuristics (Palmer 1965) and others. Nowadays, the trend is changing to use swarm intelligent concept to solve the flowshop scheduling problem. One of the methods that can be used to solve the makespan problem in scheduling is using Artificial Bee Colony (ABC) algorithm.

ABC Algorithm is the concept proposed by Karaboga in 2005. Artificial bee colony algorithm concept is simple, easy to implement, fewer control parameters setting (Bonabeau *et al*. 1999), (Bao & Zeng 2009) and is known to be better than other algorithm for global optimization (Karaboga & Basturk 2008), (Jeya Mala *et al*. 2010), (Karaboga & Akay 2009), (Marinakis *et al*. 2009), (Liu & Liu 2013). This algorithm used the concept based on the foraging behavior of honey bee swarms. Bee swarm consists of three group i.e. employee bees, onlooker bees and scout bees. Employee bee is responsible to search the new food source. Employee bee used waggle dance to communicate with other bees. Onlooker bees will wait in the hive and receive the information from employee bee. Onlooker bee is responsible for making decision to choose the best food source. Scout bee is responsible to find the new food source randomly.

Permutation flowshop scheduling problem

The permutation flowshop scheduling will find the best permutation to minimize the maximum completion time or makespan. Solution to permutation flowshop scheduling problem is represented by the permutation of n jobs. There is a set of n jobs, $\pi = \pi_1, \pi_2, \ldots, \pi_n$. Each job will be processed on m operations. Every operation will be performed by different machine. The processing time p_{ij} for job j and using machine i is given. The best permutation for jobs $\pi^* = \{\pi_1^*, \pi_2^*, \ldots, \pi_n^*\}$ to be processed on each machine can be found using the permutation flowshop scheduling (N.A. Sidek 2014). Let, $C(\pi_j,m)$ denotes the completion time for the job π_j using machine m. The makespan for a permutation π is equal to the completion time for the last job π_n using the last machine m. The completion time for the permutation π is $c_{max}(\pi)=c(\pi_n, m)$ (Karaboga & Basturk 2007). Given the job permutation π, the completion time for the n job, m machine problem is calculated as follows.

$$c(\pi_1, 1) = \rho_{\pi 1, 1}$$

$$c(\pi_j, 1) = c(\pi_j - 1, 1) + \rho_{\pi j,1} \quad j = 2, \ldots, n$$

$$c(\pi_1, i) = c(\pi_1, i - 1) + \rho_{\pi 1,i} \quad i = 2, \ldots, m$$

$$c(\pi_j, i) = max[\, c(\pi_j - 1, i), c(\pi_j, i - 1)] + \rho_{\pi j,i}$$

$$j = 2, \ldots, n \qquad I = 2, \ldots, m$$

Artificial bee colony algorithm

There are three important components in ABC algorithm after the foraging process. Among them are food source, fitness value and bee agents. Food source represents a feasible solution in an optimization problem. Fitness value represents the profitability of a food source. In other words, it is represented as a single quantity related to the objective function of a feasible solution. The last components of ABC algorithm, Bee agents refer to a set of computational agents. The agents in ABC algorithm are categorized into three groups of bees i.e. employed bees, onlooker bees and the last group is scout bees (P. L. Wei 2011).

VOL. 11, NO. 10, MAY 2016 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

The ABC algorithm process begins by initializing the solutions corresponding to the initial food source (being *SN* the number of solutions, one of the parameters of the algorithm). These solutions are evaluated and a cycle that repeats *MCN* times starts, where *MCN* is the maximum number of cycles (Banharnsakun *et al*. 2012). This cycle starts with the employed bees assigned to the food source and they calculated new candidate solutions using $y_{i,j}=x_{i,j}+\phi\cdot(x_{i,j}-x_{k,j})$ formula. Then, greedy selection process is applied i.e. the best solution will be selected between the food source and their candidate solution. Based on the fitness of the food sources that are retained after the previous step, it is determined which solutions will be visited by onlooker bees. Onlooker bees will be visiting this solution and generate candidate solution using $y_{i,j}=x_{i,j}+\phi\cdot(x_{i,j}-x_{k,j})$ formula. After that, it will apply the greedy selection process same as the process of employed bees.

Next, it is to determine whether there is any abandoned solution. This happens when a solution is not improved or not replaced by a candidate after a number of cycles set by the variable *limit,* which the value is given by the formula (*SN* * *D*) where *SN* is the number of solutions and *D* is the number of variables of the problem. Those abandoned sources are replaced with the new source found by scout bees, which in this research, the new solution will be calculated randomly within the search space. Lastly, the best solution found during the cycle is compared with the best solution found so far, and if it has a better fitness, it will replace it. The cycle is repeated *MCN* times[10]. The summary of the ABC process is shown in Figure-1.

```
Step 1   Begin
Step 2     Initialize the solution population x_i, i=1, ...,SN
Step 3     Evaluate population
Step 4     cycle = 1
Step 5     Repeat
Step 6       Generate new solutions v_i for the employed bees using y_{i,j}=x_{i,j}+φ·(x_{i,j}−x_{k,j}) formula and
             evaluate them
Step 7       Keep the best solution between current and candidate
Step 8       Select the visited solution for onlooker bees by their fitness
Step 9       Generate new solutions v_i for the onlooker bees using y_{i,j}=x_{i,j}+φ·(x_{i,j}−x_{k,j}) formula and
             evaluate them
Step 10      Keep the best solution between current and candidate
Step 11      Determine if exist an abandoned food source and replace it using a scout bee
Step 12      Save in memory the best solution so far
Step 13      cycle = cycle + 1
Step 14    Until cycle = MCN
Step 15  End
```

Figure-1. Procedure in ABC algorithm (Karaboga & Basturk 2007).

Onlooker decision approaches for artificial bee colony system

This paper focused on the application of the ABC algorithm for a three machine flow shop environment. In executing the ABC algorithm, three employee bees and three onlooker bees were utilized. A detail observations and analysis were specially done to evaluate the ABC system effectiveness with respect to varying the approaches of the onlooker bees in making decision for food source searching.

Three onlooker decision approaches are reported in this paper i.e. method 3+0+0, method 2+1+0 and method 1+1+1. All the methods were based on the potential movement of onlooker bees in a cycle.

a) Method 3+0+0

The movement of employed bees and onlooker bees when searching the food source for this method is shown in Figure-2. Three onlooker bees will receive the information from employed bees and choose the best food source as a reference. Assuming source A is the best food source then the three onlooker bees will find new food sources within the source A area.

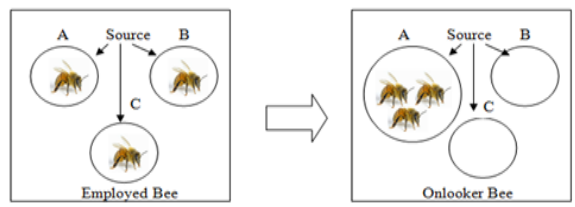

Figure-2. Movement of bee in method 3 + 0 + 0.

b) Method 2+1+0

For this method, assuming source C is the best food source, then two onlooker bees will choose source C and another bee will choose the second best food source i.e. in source A to keep searching new food source. Figure-3 shows the movement of employee bees and onlooker bees for this method.

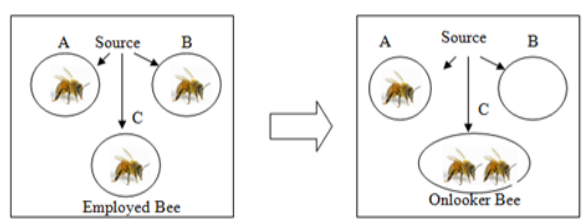

Figure-3. Movement of bee in method 2 + 1 + 0.

c) Method 1+1+1

For this method, each onlooker bees will go to single source as shown in Figure 4. Each onlooker bees go to area A, area B and area C.

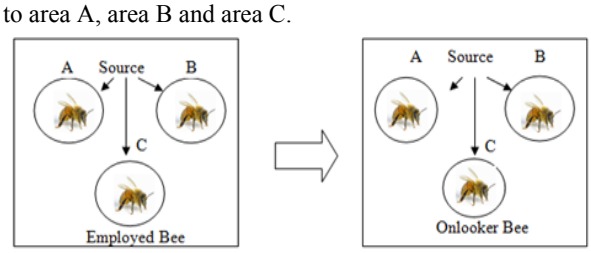

Figure-4. Movement of bee in method 1 + 1 + 1.

Element used in ABC system

There are 5 elements considered in this system. Each element has been used in all three methods developed through the research work.

- Six job and three machine

This system is set to solve the scheduling problem for six jobs and three machines flowshop.

- One hundred sets of data

The performance of the system is tested using randomly generated one hundred different set of data.

- Makespan / Total completion time

The aim for this system is to find the best makespan. This system calculates the makespan for each data set by computing the completion time of the last job at the last machine sequence.

d) Average percentage makespan difference

Optimum makespan is the best makespan in a set of data. This is obtained by finding the minimum makespan from all possible sets of sequences. The makespan obtained from the ABC system may or may not be the best makespan. The difference between optimum makespan with makespan obtained from the system is known as makespan difference. Percentage makespan difference is calculated by using the formula below.

$$MakespanDifference(\%) = \left(\frac{MakespanFromABC - OptimumMakespan}{OptimumMakespan}\right) \times 100$$

$$AveragePercentageMakespanDifference = \frac{TotalMakespanDifference(\%)}{100}$$

e) Thirty times replication

All simulations in this system were repeated thirty times. This process is also known as the thirty replicates. This process is important to ensure that the result obtain from this system is reliable. Average for all thirty replications was calculated and recorded as the result for this study.

Parameter setting

The related parameter setting used in this experiment was in Table-1.

Table-3. Parameter setting.

Parameter	Value
Cycle Limit	6, 12, 18
Iteration limit	24, 54, 102, 204, 300, 402, 504
Replications	30

The ABC process begins by assigning a random initial solution to the flow shop scheduling problem. The makespan for this initial solution is also measured. Three employed bees are released to search for better solutions within the neighborhood of the initial solution. The neighborhood is controlled by maintaining the same first job sequence for employed bees and onlooker bees corresponding to either initial solution or scout bee.

The overall ABC process utilized in this research can be summarized as the following steps:

1. Generate initial solution randomly.

Set Cycle = 0, Iteration = 0, BestMakespan = Initial solution makespan.

2. Release 3 employed bees.

Calculate each solution makespan.

3. Release 3 onlooker bees following the specific onlooker decision approach (either 3+0+0, 2+1+0 or 1+1+1). Calculate each solution makespan.
4. Iteration = Iteration + 6
   CycleBestMakespan = Minimum makespan from step 2 and 3.
5. If CycleBestMakespan < BestMakespan
   update BestMakespan
   cycle = 0.
6. If Iteration ≥ IterationLimit, stop.
   If Cycle < CycleLimit, go to step 2.

Otherwise release scout bee. Go to step 2.

EXPERIMENTAL RESULTS

The result from this simulation measures the average percentage makespan differences from the optimum solution. From this simulation, the average percentage makespan differences were analyzed to identify the best onlooker approach (method) to solve the makespan problem.

a) 24 iterations

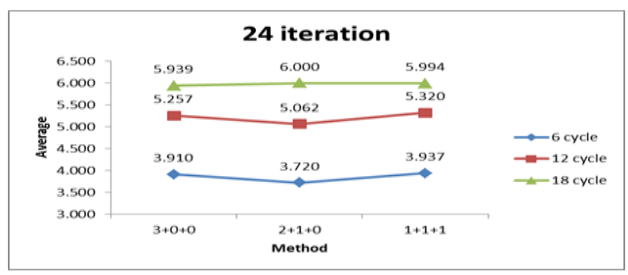

Figure-5. Average percentage makespan difference for iteration twenty four.

Figure-5 shows the average percentage makespan difference for each cycle in iteration limit of twenty four. From the result, cycle six is better than other cycles for all methods. The best average percentage makespan difference in iteration twenty four is 3.72% which is also in cycle six. The best average percentage makespan difference for this iteration is using second method i.e. 2 + 1 + 0.

b) 54 iterations

VOL. 11, NO. 10, MAY 2016 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

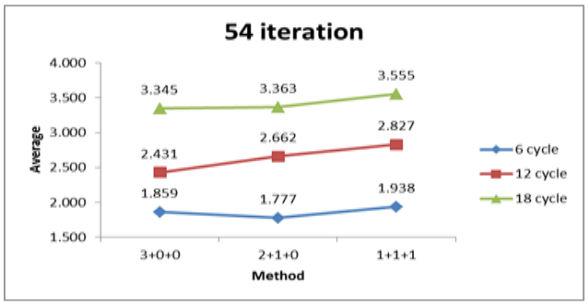

Figure-6. Average percentage makespan difference for iteration fifty four.

Figure-6 shows the average percentage makespan difference for iteration limit of fifty four. The performance of this iteration is better than iteration twenty four. The performance of average percentage makespan difference for each cycle in iteration fifty four shows improvements. Similar with iteration twenty four, the best average percentage makespan difference for iteration fifty four is also using 2+1+0 method with cycle six i.e. 1.777%.

c) 102 iterations

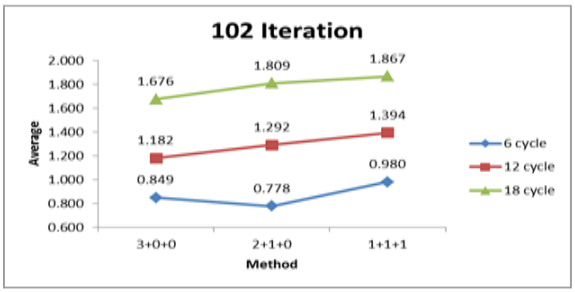

Figure-7. Average percentage makespan difference for iteration one hundred two.

Figure-7 shows the average percentage makespan difference for each cycle in iteration limit of one hundred two. From the result, cycle six is better than other cycles for all methods. The best average percentage makespan difference in this iteration is 0.778% which is also in cycle six. Similar with iteration twenty four and fifty four, the best average percentage makespan difference for this iteration is using second method i.e. 2 + 1 + 0.

d) 204 iterations

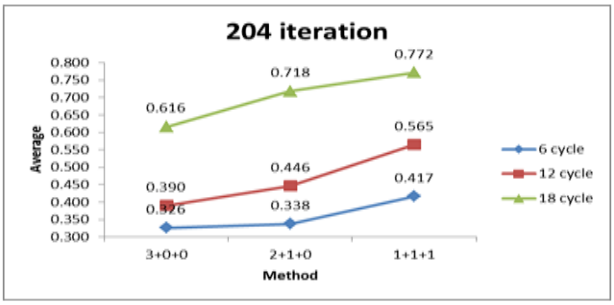

Figure-8. Average percentage makespan difference for iteration two hundred four.

Figure-8 shows the average percentage makespan difference for iteration limit of two hundred four. Starting at iteration two hundred four, the performance of cycle twelve become closer to cycle six. There is small difference between the best cycles with second best cycle for iteration two hundred four. Beginning from this iteration, the best makespan difference has changed from method 2+1+0 to 3+0+0. The best average percentage makespan difference for this iteration is 0.326% which is using method 3+0+0 with six cycles.

e) 300 iterations

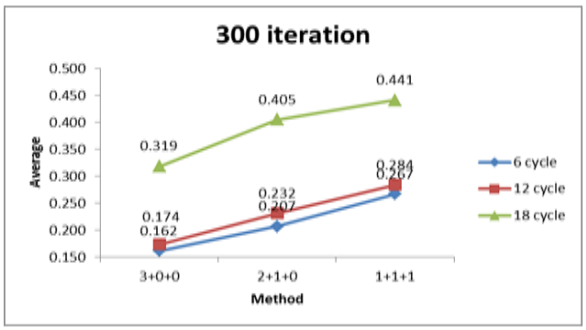

Figure-9. Average percentage makespan difference for iteration three hundred.

Figure-9 shows the average percentage makespan difference for iteration limit of three hundred. The difference of average percentage makespan difference for iteration three hundred is smaller than iteration two hundred four. The best cycle for iteration three hundred is cycle 6 using method 3+0+0.

f) 402 iterations

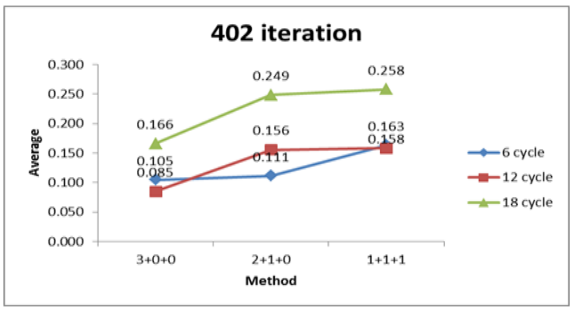

Figure-10. Average percentage makespan difference for iteration four hundred two.

Figure-10 shows the average percentage makespan difference for iteration limit of four hundred two. The graph pattern for this iteration is different with the previous iteration. The best average percentage makespan for this iteration is 0.085% with twelve cycle and using method 3+0+0.

g) 504 iterations

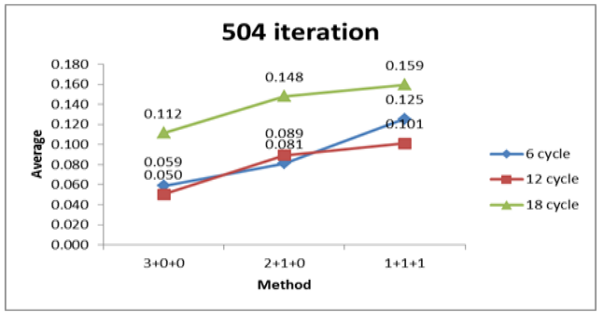

Figure-11. Average percentage makespan difference for iteration five hundred four.

Figure-11 shows the average percentage makespan difference for iteration limit of five hundred four. The graph pattern for this iteration is similar with iteration four hundred two. The performance of average percentage makespan difference for each cycle in iteration five hundred four shows improvements. The best average percentage makespan for this iteration is 0.050% with twelve cycle and using method 3+0+0.

Verification of the developed ABC system

Verification of the developed ABC system was executed using data from Chow *et al.* (2013). They developed a simulator to identify the makespan of a flowshop problem using the basic procedure in ABC. Table-2 shows the data used in Chow *et al.* to illustrate the basic concept of implementing ABC in flowshop. This data were used in this research to ensure that the new procedure in the ABC system can be used to find the minimum makespan.

Table-2. Data completion time for 6 jobs, 6 machines (Chow *et al*. 2013).

Job	P_1	P_2	P_3	P_4	P_5	P_6
A	7	8	9	4	5	6
B	4	5	6	1	2	3
C	1	2	3	7	8	9
D	7	8	9	1	2	3
E	4	5	6	7	8	6
F	1	2	3	4	5	9

The simulator from Chow *et al.* (2013) showed that the minimum makespan for data in Table-3.4 is 55 hours with the job sequence is F, C, E, A, B and D. By using the ABC system, the minimum makespan for the same data is 53 hours with the job sequence A, B, F, D, E and C. Table-3 shows the minimum makespan obtained from the same data by using two different procedures in ABC.

Table-3. Makespan obtained from two different procedures in ABC.

Method	Makespan	Sequence
Basic ABC (Chow *et al.*, 2013)	55	F-C-E-A-B-D
ABC system (300 iteration, 3+0+0 method)	53	A-B-F-D-E-C

CONCLUSIONS

In the previous section, we have analyzed the three methods (3+0+0, 2+1+0 and 1+1+1) of different onlooker bees approaches. Based on this experiment, method 2 + 1 + 0 able to find the lowest average percentage makespan difference compared with the other two methods at the early stages i.e. using twenty four iterations until one hundred two iterations. From two hundred four iterations until five hundred four iterations, the method that is able to find the lowest average percentage makespan difference is method 3 + 0 + 0. Similarly, cycle six is able to generate best solution at iterations of twenty four to three hundred. Beginning from iteration four hundred two onwards, cycle twelve produced the best result. In other words, we can conclude the findings as the followings:

If iteration is limited to only one hundred two and below, cycle six with 2+1+0 method is the best. If iteration limit is between two hundred four and three hundred, cycle six with 3+0+0 method is the best. Finally, if iteration limit is at four hundred two and above, cycle twelve with 3+0+0 is the best.

ACKNOWLEDGEMENTS

This work is supported by Research Grant Scheme VOTE 1497 under Ministry of Education, Malaysia.

REFERENCES

[1] Banharnsakun, A., Sirinaovakul, B. & Achalakul, T., 2012. Job Shop Scheduling with the Best-so-far ABC. Engineering Applications of Artificial Intelligence, 25(3), pp.583–593.

[2] Bao, L. & Zeng, J., 2009. Comparison and analysis of the selection mechanism in the artificial bee colony algorithm. In Hybrid Intelligent Systems, 2009. HIS'09. Ninth International Conference on. IEEE, pp. 411–416.

[3] Bonabeau, E., Dorigo, M. & Theraulaz, G., 1999. Swarm intelligence: from natural to artificial systems Oxford Univ Press.

[4] Chow, H.Y., Hasan, S. & Bareduan, S.A., 2013. Basic Concept of Implementing Artificial Bee Colony (ABC) System in Flow Shop Scheduling. In Applied Mechanics and Materials. Trans Tech Publ, pp. 385–388.

[5] Gupta, J.N.D., 1971. A functional heuristic algorithm for the flowshop scheduling problem. Operational Research Quarterly, pp.39–47.

[6] Jeya Mala, D., Mohan, V. & Kamalapriya, M., 2010. Automated software test optimisation framework-an artificial bee colony optimisation-based approach. Software, IET, 4(5), pp.334–348.

[7] Karaboga, D. & Akay, B., 2009. A comparative study of artificial bee colony algorithm. Applied Mathematics and Computation, 214(1), pp.108–132.

[8] Karaboga, D. & Basturk, B., 2007. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. Journal of global optimization, 39(3), pp.459–471.

[9] Karaboga, D. & Basturk, B., 2008. On the performance of artificial bee colony (ABC) algorithm. Applied soft computing, 8(1), pp.687–697.

[10] Liu, Y.-F. & Liu, S.-Y., 2013. A hybrid discrete artificial bee colony algorithm for permutation flowshop scheduling problem. Applied Soft Computing, 13(3), pp.1459–1463.

[11] Marinakis, Y., Marinaki, M. & Matsatsinis, N., 2009. A hybrid discrete artificial bee colony-GRASP algorithm for clustering. In Computers & Industrial Engineering, 2009. CIE 2009. International Conference on. IEEE, pp. 548–553.

[12] N.A. Sidek, S.A.B., 2014. Effect of Trial Counter Variation on the Performance of Artificial Bee Colony (ABC) Algorithm for Permutation Flowshop Scheduling Problem (PFSP). In Institute of Engineering Mathematics, Universiti Malaysia Perlis (UniMAP).

[13] Nawaz, M., Enscore, E.E. & Ham, I., 1983. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. Omega, 11(1), pp.91–95.

[14] P. L. Wei, T.C.W., 2011. A Novel Artificial Bee Colony Algorithm for Solving Constrained Optimization Problems. , pp.789–798.

[15] Palmer, D., 1965. Sequencing jobs through a multi-stage process in the minimum total time--a quick method of obtaining a near optimum. OR, pp.101–107.