Optimization of Makespan and Mean Flow Time for Job Shop Scheduling Problem FT06 Using ACO

Nasir Mehmood1, Muhammad Umer2, Dr. Riaz Ahmad3, Dr. Amer Farhan Rafique4

F. Author, Nasir Mehmood is with National University of Sciences & Technology (NUST), Islamabad, Pakistan at its School of Mechanical & Manufacturing Engineering (SMME) as a student.
S. Author, Mohammad Umar is with National University of Sciences & Technology (NUST), Islamabad, Pakistan at its School of Mechanical & Manufacturing Engineering (SMME).
T. Author, Riaz Ahmad is with National University of Sciences & Technology (NUST), Islamabad, Pakistan at its School of Mechanical & Manufacturing Engineering (SMME).
F. Author, Amer Farhan Rafique is with Mohammad Ali Jinnah University (MAJU), Islamabad, Pakistan
rajanasiraja@hotmail.com[1], muhammad.umer@smme.nust.edu.pk[2], dresearch@nust.edu.pk[3], afrafique@yahoo.com [4]

Abstract: Ant Colony Optimization (ACO) is inspired by the foraging behavior of the ants which are mostly blind but due to the indirect means of communication between the ants called stigmergy, they follow the shortest path between their nest and the source of the food. The swarm intelligence of the ants is then translated into the artificial intelligence by means of ant colony optimization metaheuristics. Job Shop Scheduling Problem (JSSP) is very prominent area of a manufacturing environment. These problems are called Combinatorial Optimization Problems (COP) which are NP-hard. The solution of these NP-hard problem through exact algorithm is not suitable. The best way to tackle such problems is the metaheuristic approach which finds out the optimal solution in minimum possible time. There exits many metaheuristics approaches to solve such COP like Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Simulated Annealing (SA) and ACO. In this paper COP from the manufacturing environment, JSSP FT06 will be solved through ACO based algorithm with the objective function to minimize the makespan and mean flow time. The results of the proposed algorithm will be checked against the Best Known Solution (BKS) of the benchmark problem FT06. This paper will show how the proposed algorithm has produced better results than the BKS of the FT06. Then the achieved results will be compared with the results of the other metaheuristic approaches like GA, Conventional Clonal Selection Algorithm (CSA), Positive Selection based Modified Clonal Selection Algorithm (PSMCSA), Particle Swarm Optimization (PSO) and Tabu Search (TS). This paper shall discuss the results and its analysis in detail regarding the makespan, mean flow time and the computational time of the FT06 problem. And in the end this paper shall draw conclusion on the basis of this research and shall render some recommendations as well.

[Mehmood N, Umer M, Ahmad R, Farhan A. **Optimization of Makespan and Mean Flow Time for Job Shop Scheduling Problem FT06 Using ACO.** *Life Sci J* 2013;10(4):477-484] (ISSN:1097-8135). http://www.lifesciencesite.com. 62

Keywords: Job Shop Scheduling Problems (JSSP); Ant Colony Optimization (ACO); Genetic Algorithm (GA); Artificial Immune System

1. Introduction

ACO metaheuristic is based on probabilistic approach. ACO metaheuristic calculates the probability of an ant *k* for the move from node *i* to node *j* by equation [1] given below

$$p_{ij}^{k} = \frac{[[\tau]_{ij}]^{\alpha}[[\eta]_{ij}]^{\beta}}{\sum_{l \in N_i^k}[[\tau]_{il}]^{\alpha}[[\eta]_{il}]^{\beta}} \quad ; \text{if } j \in N_i^k, \ (1)$$

Where

τ_{ij} indicates the pheromone trail amount.

η_{ij} indicates the heuristics information.

á is the parameter that controls the quantity of pheromone. If á = 0 this indicates that algorithm is only using the heuristic information which is problem specific .

â is the parameter that controls the heuristics information. If â = 0, this indicates that the algorithm is not using heuristic information and it is only dependant on pheromone quantity.

Accumulation of pheromone is avoided because it leads more towards exploitation than exploration and the algorithm may stuck in the local optima. Therefore pheromone evaporation is introduced in the algorithm. With the help of pheromone evaporation, the exploration of the global optima becomes easier. The pheromone evaporation in controlled by the following equation:

$$\mid \left(1 \mid \rho\right) \tau_{ij}$$

τ_{ij} (2)

Where

τ_{ij} is the intensity of the pheromone already deposited.

$\rho \mid$ (0,1) is a parameter that controls and regulates the pheromone trail evaporation.

2. Problem Definition

In 1963 Fisher and Thompson proposed FT06 as a classical benchmark JSSP [2]. The size of the FT06 is 6 x 6 which shows that there are six machines and six jobs. The total number of machine operations is 36. In FT06 JSSP each job will have one operation on each machine or each machine will process each job at least once. The BKS of FT06 for makespan and mean flow time are 55 and 44.17 time units respectively. Following two matrix will explain the jobs, machines and their processing time

$$M = \begin{bmatrix} j1 & 3 & 1 & 2 & 4 & 6 & \mathbf{5} \\ j2 & 2 & 3 & 5 & 6 & 1 & \mathbf{4} \\ j3 & 3 & 4 & 6 & 1 & 2 & \mathbf{5} \\ j4 & 2 & 1 & 3 & 4 & 5 & \mathbf{6} \\ j5 & 3 & 2 & 5 & 6 & 1 & \mathbf{4} \\ j6 & 2 & 4 & 6 & 1 & 5 & \mathbf{3} \end{bmatrix} \quad T = \begin{bmatrix} j1 & 3 & 1 & 2 & 4 & 6 & \mathbf{5} \\ j2 & 2 & 3 & 5 & 6 & 1 & \mathbf{4} \\ j3 & 3 & 4 & 6 & 1 & 2 & \mathbf{5} \\ j4 & 2 & 1 & 3 & 4 & 5 & \mathbf{6} \\ j5 & 3 & 2 & 5 & 6 & 1 & \mathbf{4} \\ j6 & 2 & 4 & 6 & 1 & 5 & \mathbf{3} \end{bmatrix}$$

In above matrix M, first row represent the the machines on which job 1 is to be processed. The processing time of each machine is given is the matrix T. In matrix T each element in the row represent the processing time of corresponding elements in the matrix M which represent the machines. The above problem is FT06 as it represents six machines and six jobs.

2.1Assumptions. To solve the FT06 problem following assumptions are considered in mind

1. Preemption is not allowed. Each machine will complete its operation . It will process other operation when first operation is complete.
2. Breakdown is not allowed. This means that machines work from the start of the operations till completion without any malfunction.
3. Processing time includes the setup time or it is assumed to be zero.
4. No job enters during the process rather each job is available and ready at the start of the process.
5. No priority is attached to any job and they are independent.
6. Processing time includes the inspection time or there is simply no inspection.
7. Processing time includes the transportation time of jobs from one machine to other machine or it is assumed as zero.

3. The Proposed Algorithm

The proposed algorithm is based on the ACO metaheuristic which is described by the following equation

$$p_{ij}^{k} = \frac{[[\tau]_{ij}]^{\alpha}[[\eta]_{ij}]^{\beta}}{\sum_{l \mid N_i^k} [[\tau]_{il}]^{\alpha}[[\eta]_{il}]^{\beta}} \text{ ; if } j \in N_i^k \text{ , (1)}$$

In the equation above α = 0 and β =1 .Initially the

$\mid \tau \mid [_{ij}$

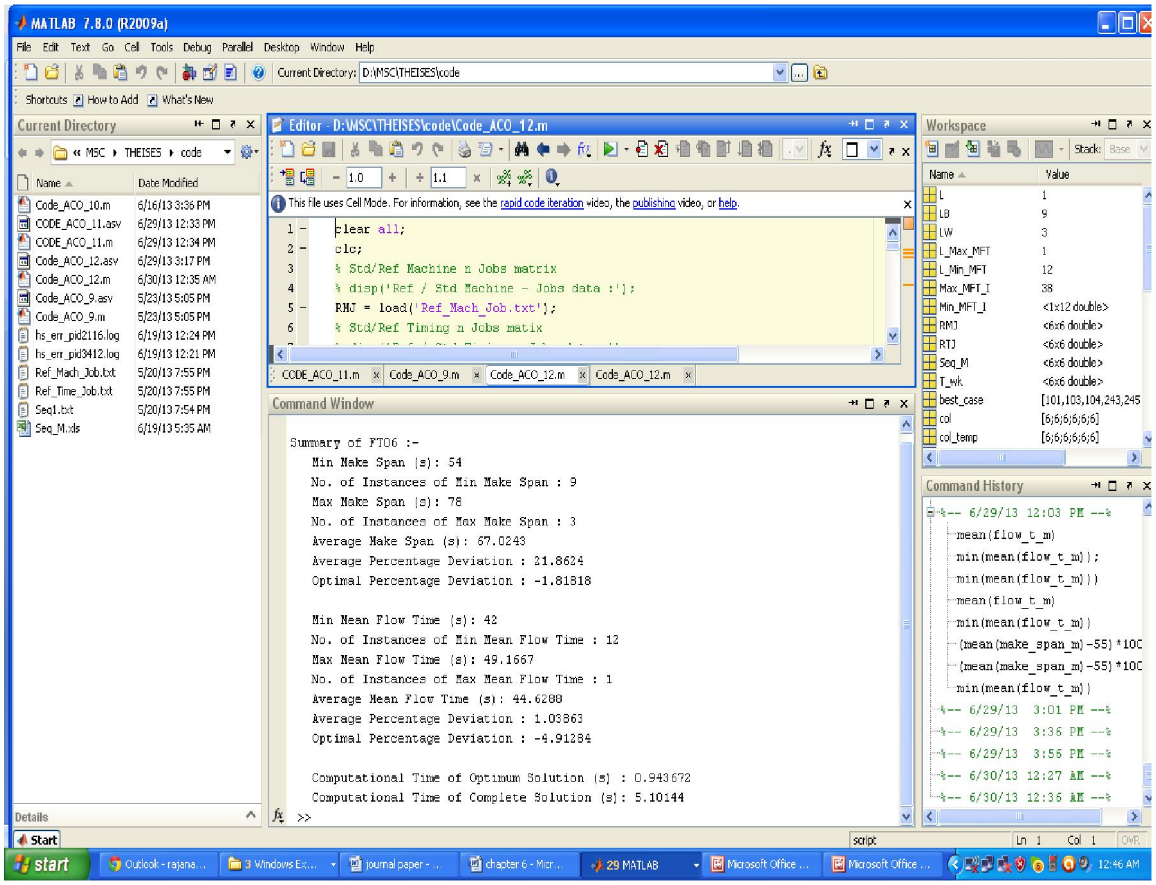
pheromone concentration) is not considered for the first iteration and the processing time is taken as the heuristic information(). After this the probabilities of each job for the sequence of machines is calculated. The higher value of probabilities will have higher chances for the visit of the ant. In this way the total sequences of machines for all six jobs comes out to be 576 which is the new reduced design space. Now for each sequence, the sequence of jobs for each machine would be different. When extensive search for all the 576 sequences is carried out then it gives the optimized makespan of the problem.

The algorithm uses the processing time as the visibility function. It first finds the sequence of operations for each job. Then it finds the sequence of jobs for each machine. The conflict that which job will be processed first is again resolved by the processing time used as heuristic information. Makespan can be found by combining the sequences for each job and each machine together. These sequences are finally plotted on the gantt chart.The algorithm uses the greedy local search which is more inclined towards the exploitation than exploration and it may get stuck in local optima rather than finding the global optima. But the proposed algorithm minimizes the makespan and finds the global optima.

4. Results Achieved

The proposed ACO algorithm works efficiently and produces optimal solution in competitive time span.

The algorithm uses the processing time as heuristics information and reduces the design space upto 576 sequences. Then the algorithm performs 576 iterations to find the optimal solutions. The best make span found is 54 and the mean flow time comes out to be 42. The summary of the results is shown below.


MATLAB 7.8.0 (R2009a)
File Edit Text Go Cell Tools Debug Parallel Desktop Window Help
Current Directory: D:\MSC\THEISES\code
Shortcuts How to Add What's New
Current Directory
Editor - D:\MSC\THEISES\code\Code_ACO_12.m
Workspace
Command Window
Command History
Start

Name	Date Modified
Code_ACO_10.m	6/16/13 3:36 PM
CODE_ACO_11.asv	6/29/13 12:33 PM
CODE_ACO_11.m	6/29/13 12:34 PM
Code_ACO_12.asv	6/29/13 3:17 PM
Code_ACO_12.m	6/30/13 12:35 AM
Code_ACO_9.asv	5/23/13 5:05 PM
Code_ACO_9.m	5/23/13 5:05 PM
hs_err_pid2116.log	6/19/13 12:24 PM
hs_err_pid3412.log	6/19/13 12:21 PM
Ref_Mach_Job.txt	5/20/13 7:55 PM
Ref_Time_Job.txt	5/20/13 7:55 PM
Seq1.txt	5/20/13 7:54 PM
Seq_M.xls	6/19/13 5:35 AM

This file uses Cell Mode. For information, see the rapid code iteration video, the publishing video, or help.

```matlab
clear all;
clc;
% Std/Ref Machine n Jobs matrix
% disp('Ref / Std Machine - Jobs data :');
RMJ = load('Ref_Mach_Job.txt');
% Std/Ref Timing n Jobs matix
```

```
Summary of FT06 :-
   Min Make Span (s): 54
   No. of Instances of Min Make Span : 9
   Max Make Span (s): 78
   No. of Instances of Max Make Span : 3
   Average Make Span (s): 67.0243
   Average Percentage Deviation : 21.8624
   Optimal Percentage Deviation : -1.81818

   Min Mean Flow Time (s): 42
   No. of Instances of Min Mean Flow Time : 12
   Max Mean Flow Time (s): 49.1667
   No. of Instances of Max Mean Flow Time : 1
   Average Mean Flow Time (s): 44.6288
   Average Percentage Deviation : 1.03863
   Optimal Percentage Deviation : -4.91284

   Computational Time of Optimum Solution (s) : 0.943672
   Computational Time of Complete Solution (s): 5.10144
>>
```

Name	Value
L	1
LB	9
LW	3
L_Max_MFT	1
L_Min_MFT	12
Max_MFT_I	38
Min_MFT_I	<1x12 double>
RMJ	<6x6 double>
RTJ	<6x6 double>
Seq_M	<6x6 double>
T_wk	<6x6 double>
best_case	[101,103,104,243,245
col	[6;6;6;6;6;6]
col_temp	[6;6;6;6;6;6]

```
%-- 6/29/13 12:03 PM --%
   mean(flow_t_m)
   min(mean(flow_t_m));
   min(mean(flow_t_m)))
   mean(flow_t_m)
   min(mean(flow_t_m))
   (mean(make_span_m)-55)*10C
   (mean(make_span_m)-55)*10C
   min(mean(flow_t_m))
%-- 6/29/13  3:01 PM --%
%-- 6/29/13  3:36 PM --%
%-- 6/29/13  3:56 PM --%
%-- 6/30/13 12:27 AM --%
%-- 6/30/13 12:36 AM --%
```

Figure 1. Snapshot of matlab showing summary of results of FT06

5. Comparison of Results

In order to evaluate the performance of the proposed algorithm, the results of the proposed algorithm are compared with algorithms of other metaheuristics approaches. The proposed algorithm has found the optimal solution in less than one second, that is just in 0.94 seconds. GA has calculated the optimal solution in one second [4].TS yielded the results in two seconds [4]. When GA and TS were hybridized then the hybrid algorithm yielded the optimal results in one second. Conventional CSA and PSMCSA produced the results in 50 and 17 seconds respectively[3]. PSO algorithm has yielded the results in seven seconds[5]. Comparison of all these results with the proposed algorithm has shown that the proposed algorithm performance is better. The same is shown in the table 1 below:

Table 1. Comparison of Results

Optimization Metaheuristic	Makespan	Computational Time (seconds)
Proposed ACO	**54**	**0.94**
GA	55	1
PSO	55	7
TS	55	2
Conventional CSA	55	50
PSMCSA	55	17

6. Analysis of Results

In this section the results achieved by the ACO algorithm will be discussed. Make span, mean flow time and the computational time for the FT06 will be discussed in detail.

6.1 Makespan

The algorithm has produced the optimum solution of makespan for nine times out of 576 iterations which has been shown below. From the graph it can be seen that the line goes nine times below 55 line which shows the BKS.

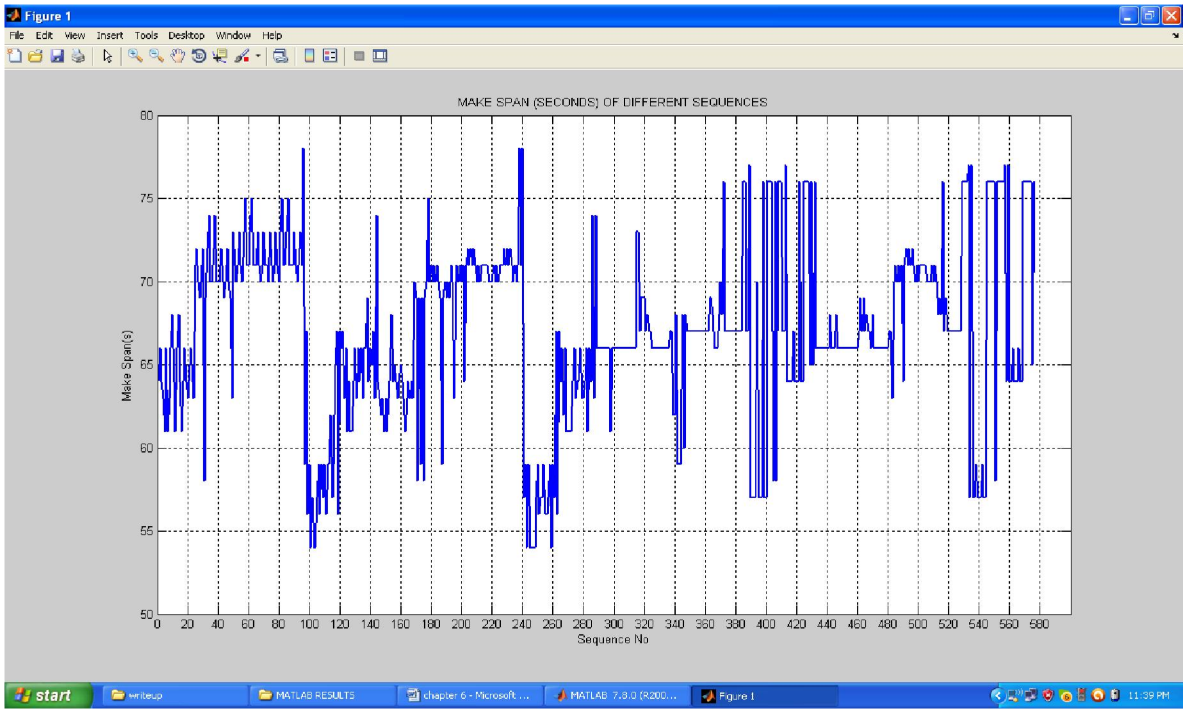


Figure 2. Snapshot of Matlab Showing Graph for Makespan

The algorithm has produced the optimal makespan of 54 against the BKS of 55. The algorithm not only calculates the optimal makespan but also it gives the gantt chart of the optimal solutions for all nine optimal solutions. One among the nine gantt chart is shown below.

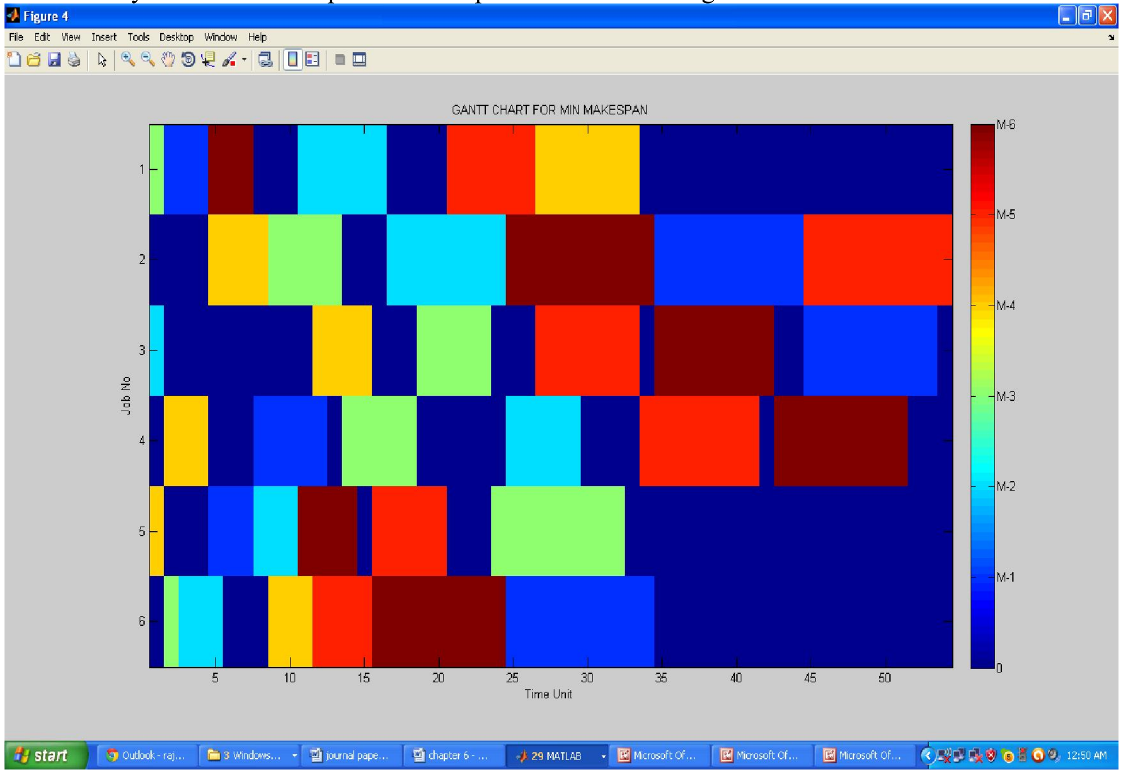


Figure 3. Snapshot of Matlab Showing Gantt Chart for Min Makespan

Maximum makespan as worst case has come out to be 78 which has been produced three times in 576 iterarions. The algorithm also gives the solution in the form of gantt chart for all three worst cases. One among three worst cases is shown below.

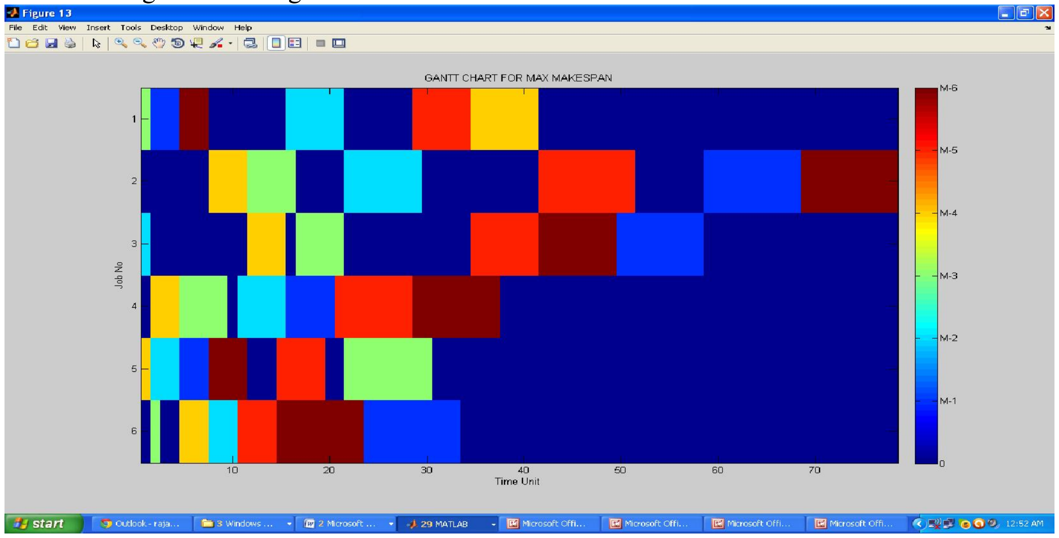


Figure 4. Snapshot of Matlab Showing Gantt Chart for Max Makespan

The algorithm also calculates the percentage deviation of the makespan. Following formula is used to calculate the percentage deviation

% deviation = (achieved results – BKS) * 100/BKS, (3)

The average makespan of all the iterations comes out to be 67.02. The percentage deviation of the average makespan from the BKS (55) comes out to be 21.86%. The percentage deviation of the optimal solution (54) from the BKS (55) comes out to be -1.8

6.2 Mean Flow Time

The algorithm has produced the optimum solution of mean flow time for twelve times out of 576 iterations which has been shown below. From the graph it can be seen that the line goes twelve times below 44.17 line which shows the BKS.

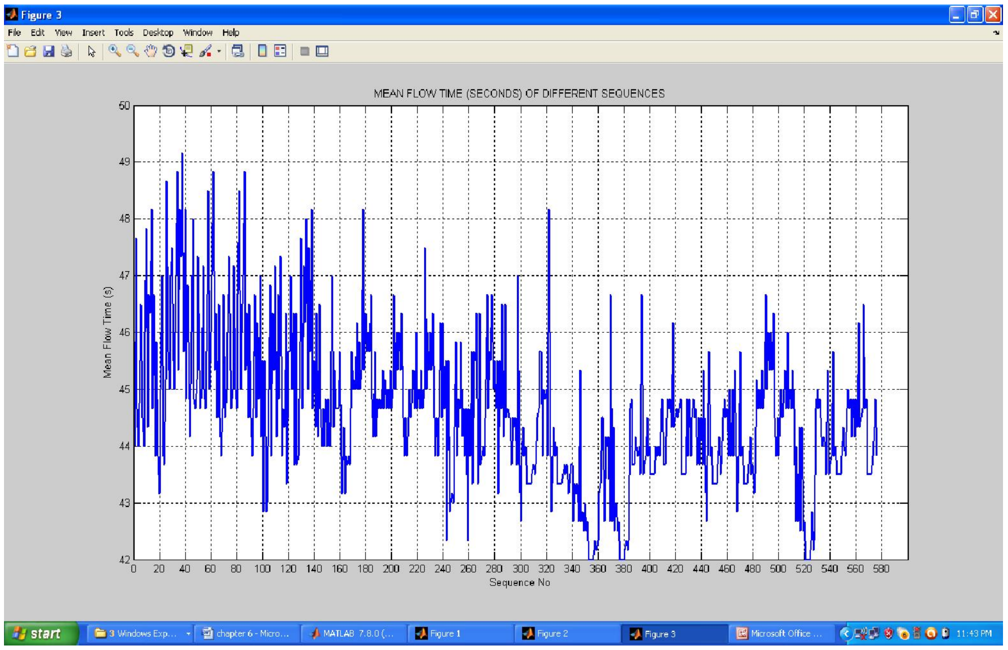


Figure 5. Snapshot of Matlab Showing Graph for Mean Flow Time

The algorithm has produced the optimal mean flow time of 42 against the BKS of 44.17. The algorithm not only calculates the optimal mean flow time but also it gives the gantt chart of the optimal solutions for all twelve optimal solutions. One among the twelve gantt chart is shown below.

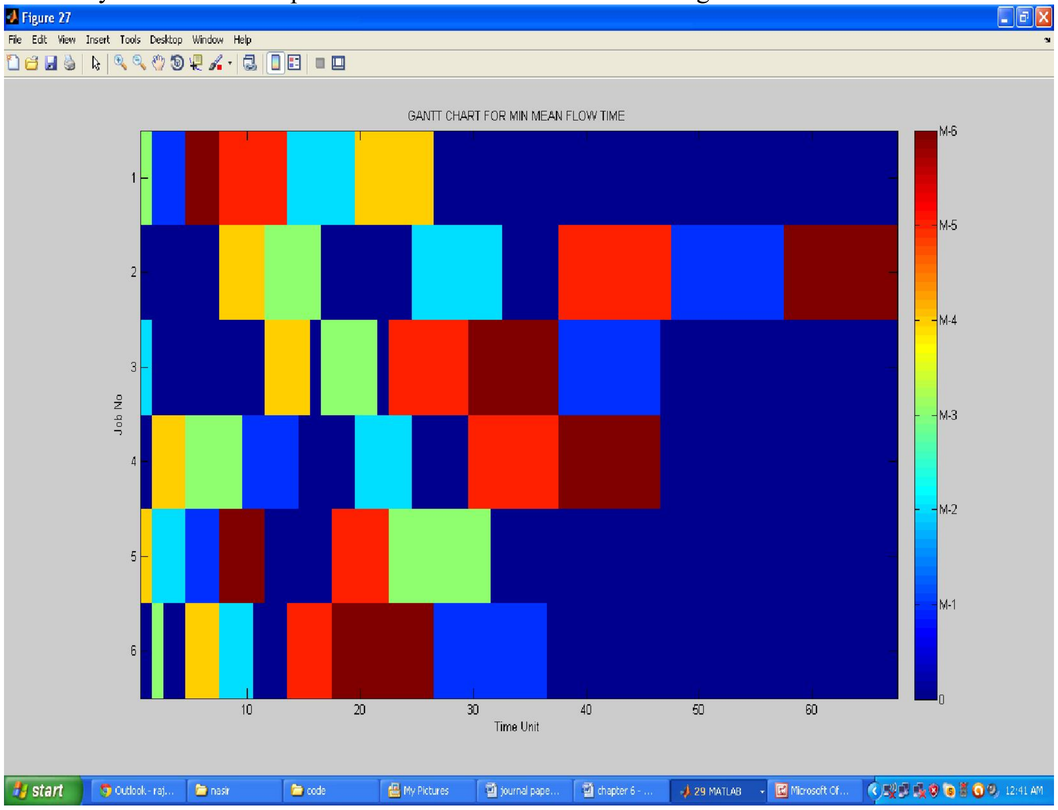


Figure 6. Snapshot of Matlab Showing Gantt Chart for Min Mean Flow Time

Maximum mean flow time as worst case has come out to be 49.16 which has been produced one times in 576 iterarions. The algorithm also gives the solution in the form of gantt chart for one worst case which is show below.

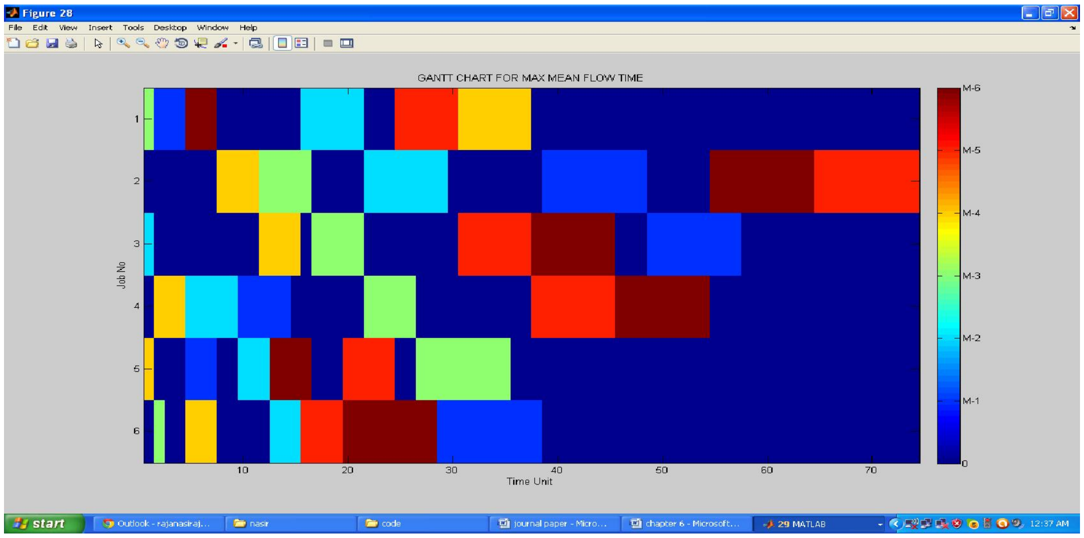


Figure 7. Snapshot of Matlab Showing Gantt Chart for Max Mean Flow Time

The algorithm also calculates the percentage deviation of the mean flow time. Following formula is used to calculate the percentage deviation

% deviation = (achieved results – BKS) * 100/BKS, (3)

The average mean flow time of all the iterations comes out to be 44.62. The percentage deviation of the average mean flow time from the BKS (44.17) comes out to be 1.03%. The percentage deviation of the optimal solution (42) from the BKS (44.17) comes out to be -4.9%.

6.3 Computational Time

The proposed algorithm is written in Matlab@9 and it is run on Intel(R) Core(TM)2 Dou CPU 17500 @ 2.20GHz processor. The computational time calculated by the algorithm for the solution of first optimum result of 54 is just less than one second and it is .94 seconds. However the algorithm gives complete results in 5.1 seconds. The graphical results of each iteration calculated by the algorithm is given below. From the graph it is evident that the average computational time of single iteration is less than .01 seconds.

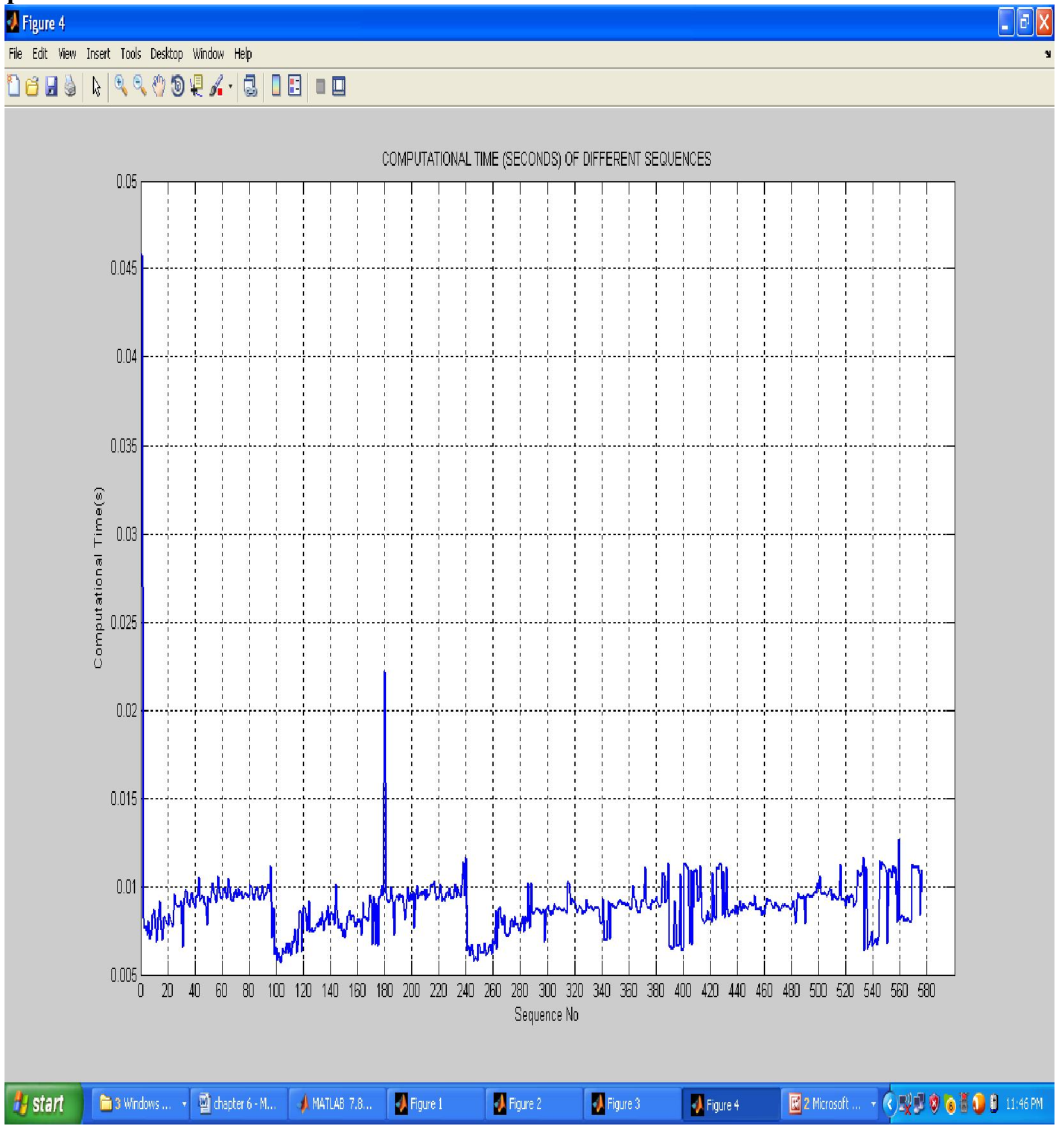


Figure 8. Snapshot of Matlab Showin Graph for Computational Time

7. BKS And Achieved Results

The BKS of the JSSP FT06 are 55 and 44.17 in case of makespan and mean flow time respectively. The proposed algorithm has produced 54 as makespan and 42 as mean flow time. This means that BKS has been improved from 55 to 54 in case of makespan and in case of mean flow time the solution has been improved from 44.17 to 42. The same has been shown in the table below.

Table 2. Comparison of achieved results with BKS of FT06

Parameters	BKS	Proposed ACO Algorithm			% Deviation(Ave)
		Best	Worst	Average	
Makespan (sec)	55	**54**	78	67.02	21.86
Mean Flow Time (sec)	44.17	**42**	49.16	44.62	1.03

8. Conclusions and Recommendations

The proposed ACO algorithm has minimized the makespan and mean flow time of the JSSP FT06 in less than one second. The optimal makespan of the benchmark problem FT06 is 55 seconds but the proposed algorithm has optimized it as 54 seconds. This means that the optimal solution has been improved one seconds by the proposed algorithm. Likewise the mean flow time of the FT06 is 44.17 seconds but it has been optimized to 42 by the proposed algorithm. So it is concluded that the proposed ACO algorithm has optimized the solution quality. On comparison the computational time with the other metaheuristic optimization approaches, it is concluded that proposed algorithm has yielded the results in better computational time.

Having been encouraged by the achieved results, it is recommended that proposed ACO algorithm shall be applied to other benchmark problems available in the operation research liberary. It is highly recommendable that the proposed ACO algorithm shall be applied to some bigger size problems like FT10 and FT20 and then the achieved results shall be compared with the best available optimization metaheuristics. It is also recommended that other than FT series, the proposed algorithm shall also be applied to Lawrence Series of problems and shall be compared with other available optimization metaheuristics approaches.

Corresponding Author:
Nasir Mehmood.
School of Mechanical and Manufacturing Engineering.
National University of Science and Technology, Islamabad, Pakistan
Email: rajanasiraja@hotmail.com

Reference

[1] Marco Dorigo, Thomas Stutzle, Ant colony optimization, 1st edition, The MIT Press, London, 2004
[2] Arindam Chaudhury, Kajal De, Job Scheduling Problem Using Rough Fuzzy Multiplier Perception Neural Networks, International Journal of Advance Engineering & application, jan.2010.
[3] R.Murugesan, Positive Selection based Modified Clonal Selection Algorithm for Solving Job Shop Scheduling Problem, Applied Mathematical Science, Vol.6, 2012, no. 46, 2255-2271.
[4] Thamilselven, R. and P. Balasubramanie, Integration of Genetic Algorithm with Tabu Search for Job Shop Scheduling with Unordered Subsequence Exchange Crossover, Journal of Computer Science, 8(5) : 681-693, 2012.
[5] Jun-qing Li, Quan-Ke Pan, Sheng-xian Xie, Yu-ting Wang, A Novel Hybrid Public Critical Block Algorithm for Solving the Job Shop Scheduling Problem.

9/22/2013