International Journal For Technological Research In Engineering
Volume 1, Issue 9, May-2014 ISSN (Online): 2347 - 4718

A MODIFIED ACTIVITY SELECTION ALGORITHM WITH NEW PARAMETER: MINIMUM MAKESPAN

Vaishali K. Patel[1] (PG Student), Prof. Mitula H. Pandya[2] (Assistant Professor)
Department of Computer Engineering
Alpha college of Engineering and Technology
Khatraj, Gujarat, India.

Abstract: The activity selection problem is mathematical optimization problem. It is concerning the selection of non-conflicting (compatible) activities to perform within a given time frame. The goal is to select maximum number of compatible activities that can be performed by a single person or machine. There are two types of technique to solve this problem: Greedy method and Dynamic programming method. The job-shop scheduling problem is the problem in which n number of jobs and m number of machines are given the main goal is to schedule the job on each machine in such a manner that makespan should be minimum. Our research focus on algorithm which gives maximum number of activities and minimum makespan.
Index Terms: Activity selection problem, Compatible activity, Makespan

I. INTRODUCTION

The main goal of activity selection problem is to select maximum number of non-conflicting (compatible) activities within a given time frame. Problem statement for activity selection can be given as follow: Given a set of activities each marked by start time (si) and finish time (fi). We have to select maximum number of activities that can be performed by a single person or machine. There will be some situation when multiple solutions exists in which number of maximum compatible activities are same and their makespan will be different. The dissertation work will try to select the solution which gives importance to both parameters: maximum number of activities and minimum makespan. Here in this work we will try to find solution using two methods. We will analyze the result of the methods and conclude best possible method.

Makespan: The time required for all operations to complete their processes is called makespan [2].

Compatible activities: Two activities are said to be compatible or non-conflicting if activity a1 & a2 has start time s1 & s2 and finish time f1 & f2 respectively. Now a1 is compatible with a2 if s2>=f1 [10].

Many criteria have been defined for comparing scheduling algorithms are as follows [6]:

- CPU utilization: keep the CPU busy for maximum time.
- Throughput: Number of processes that are completed in particular time unit.
- Turnaround Time: The time interval between submission of a process and completition of a process.
- Waiting time: Total time spend waiting in the queue.
- Response time: time interval between submissions of a process and first response from the system.

II. BASIC CONCEPTS

A. Activity Selection Problem

The steps of greedy algorithm for activity selection problem are as follows [10]:

- Sort the set of activities by finishing time (f[i])
- S = {1}
- f = f[1]
- for i=2 to n
- if s[i] ≥ f
- S = S U i
- f = f[i]
- end for

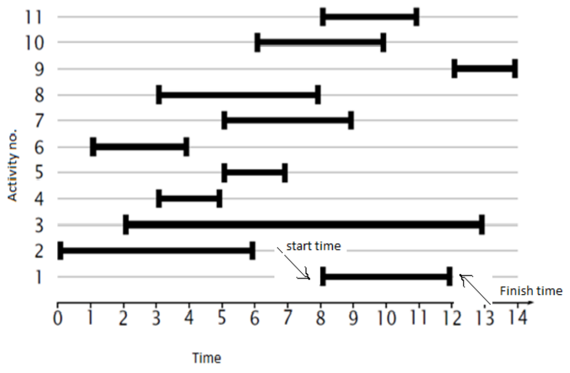
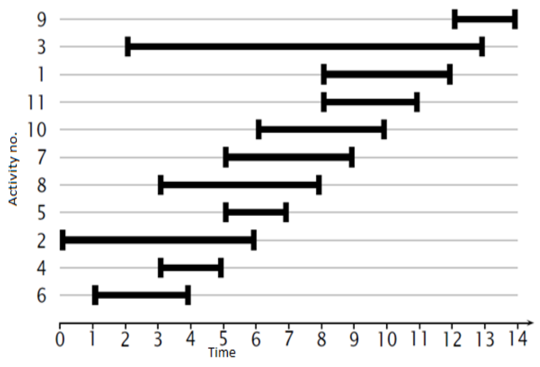


Fig.1 [Representation of activity & Time]



After sorting all activities according to finish time

Fig.2 [after sorting according to finish time]

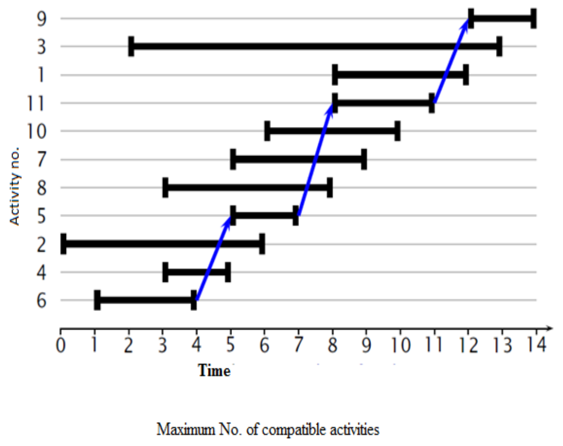


Fig.3 [Maximum compatible activities]

B. Genetic Algorithm
Genetic algorithm must have the following five components:

- There should chromosomal representation of solutions to the problem.
- There should function that evaluates the performances of solutions.
- Inhabitants of initialized solutions.
- There should be Genetic operators that evolve the population.
- The Parameters that specify the probabilities by which these genetic operators are applied.

III. PROPOSED WORK

A. Problem Description
Here 11 activities are given with its starting time and finish time. We will find a solution for activity selection problem using greedy algorithm. Then we will find solution using two proposed algorithm.

Table 1 [Activities with start time and finih time]

Activity	Start time(si)	Finish time(fi)
1	8	12
2	0	6
3	2	13
4	3	5
5	5	7
6	1	4
7	5	9
8	3	8
9	12	14
10	6	10
11	8	11

1) According To Greedy Algorithm
After sorting the activities according to their finish time the order of activities will be as follows:

Table 2 [Activities in increasing order of finish time]

Activity	**Start time(si)**	**Finish time(fi)**
1	1	4
2	3	5
3	0	6
4	5	7
5	3	8
6	5	9
7	6	10
8	8	11
9	8	12
10	2	13
11	12	14

Here solution is(1,4),(5,7),(8,11),(12,14).
Makespan for this solution is=3+2+3+2=10

2) According To Proposed Scheme
Some different possible solutions are:

- (1,4),(5,7),(8,11),(12,14)
- (3,5),(5,7),(8,12),(12,14)

Now calculate makespan for each possible solution

- Makespan(A)=3+2+3+2=10
- Makespan(B)=2+2+4+2=10

After applying cross over on two solutions

- Child1: (1,4),(5,7),(8,12),(12,14)
- Child2: (3,5),(5,7),(8,11),(12,14)
- Makespan(child1):= 3+2+4+2=11
- Makespan(child2):=2+2+3+2=9

Here solution with minimum makespan is child2 which has scheduled activities in intervals (3,5),(5,7),(8,11),(12,14).

B. Proposed Algorithm
1) Genetic Algorithm
Step-1: Enter start time and finish time for each activity.
Step-2: Find out duration for each activity.
Step-3: Initialize population based on increasing order of finish time of activity.
Step-4: Perform crossover on randomly selected two nodes based on duration as fitness function.
Step-5: Perform mutation if required.
Step-6: Perform Replacement
Step-7: Repeat step-4 and step-5 until stopping criteria satisfied.

2) Branch and Bound
Step-1: Enter start time and finish time for each activity.
Step-2: Find out duration for each activity.

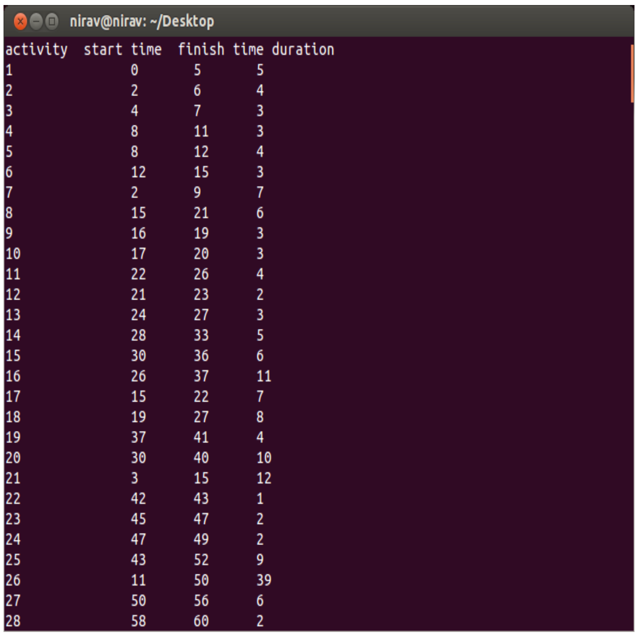
Step-3: Find out maximum number of compatible activities with particular activity.
Step-4: Include the solution found in step-3 if

- Its total duration is less than previous solution in solution set and delete previous solution in solution set.
- Otherwise discard solution found in step-3

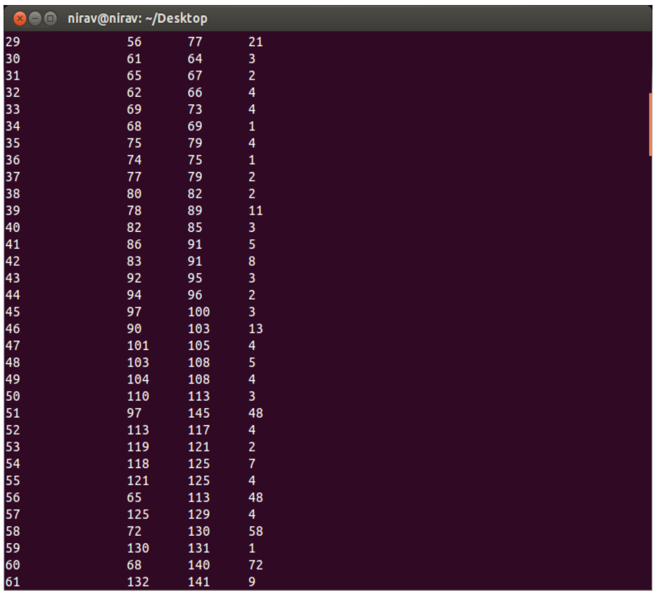
Step-5: Repeat step-3 and step-4 for all possible compatible activities with each activity.
Step-6: Find out solution from solution set with maximum number of compatible activities and minimum makespan.
Step-7: Finish

IV. IMPLEMENTATION DETAILS

- Activities with start time and finish time

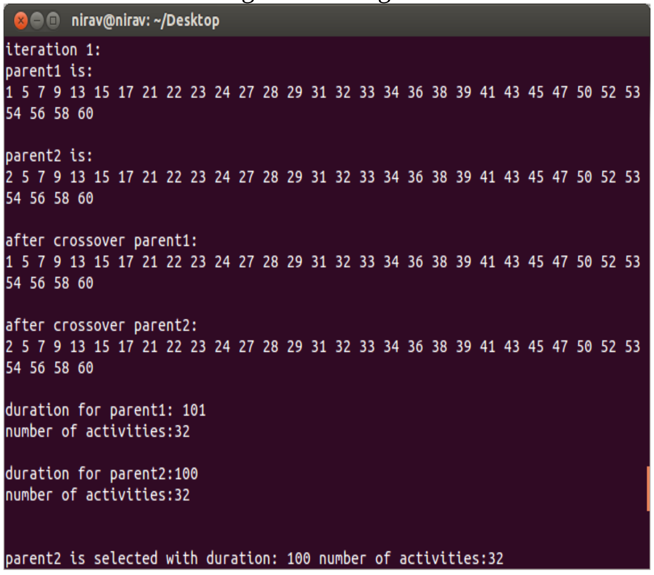


```
activity  start time  finish time duration
1                0        5        5
2                2        6        4
3                4        7        3
4                8        11       3
5                8        12       4
6                12       15       3
7                2        9        7
8                15       21       6
9                16       19       3
10               17       20       3
11               22       26       4
12               21       23       2
13               24       27       3
14               28       33       5
15               30       36       6
16               26       37       11
17               15       22       7
18               19       27       8
19               37       41       4
20               30       40       10
21               3        15       12
22               42       43       1
23               45       47       2
24               47       49       2
25               43       52       9
26               11       50       39
27               50       56       6
28               58       60       2
```

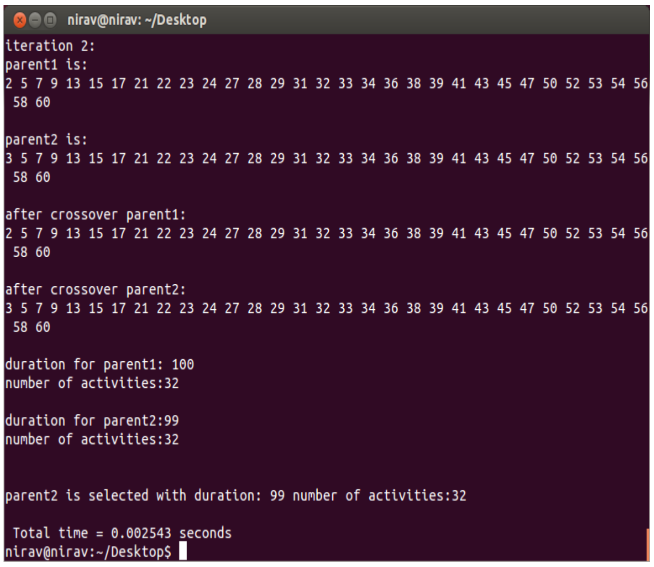


```
29               56       77       21
30               61       64       3
31               65       67       2
32               62       66       4
33               69       73       4
34               68       69       1
35               75       79       4
36               74       75       1
37               77       79       2
38               80       82       2
39               78       89       11
40               82       85       3
41               86       91       5
42               83       91       8
43               92       95       3
44               94       96       2
45               97       100      3
46               90       103      13
47               101      105      4
48               103      108      5
49               104      108      4
50               110      113      3
51               97       145      48
52               113      117      4
53               119      121      2
54               118      125      7
55               121      125      4
56               65       113      48
57               125      129      4
58               72       130      58
59               130      131      1
60               68       140      72
61               132      141      9
```

- Solution using Genetic Algorithm

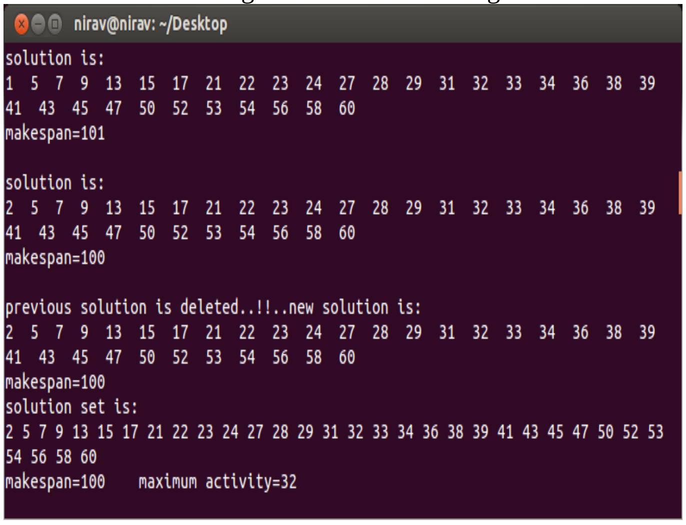


```
iteration 1:
parent1 is:
1 5 7 9 13 15 17 21 22 23 24 27 28 29 31 32 33 34 36 38 39 41 43 45 47 50 52 53
54 56 58 60

parent2 is:
2 5 7 9 13 15 17 21 22 23 24 27 28 29 31 32 33 34 36 38 39 41 43 45 47 50 52 53
54 56 58 60

after crossover parent1:
1 5 7 9 13 15 17 21 22 23 24 27 28 29 31 32 33 34 36 38 39 41 43 45 47 50 52 53
54 56 58 60

after crossover parent2:
2 5 7 9 13 15 17 21 22 23 24 27 28 29 31 32 33 34 36 38 39 41 43 45 47 50 52 53
54 56 58 60

duration for parent1: 101
number of activities:32

duration for parent2:100
number of activities:32


parent2 is selected with duration: 100 number of activities:32
```

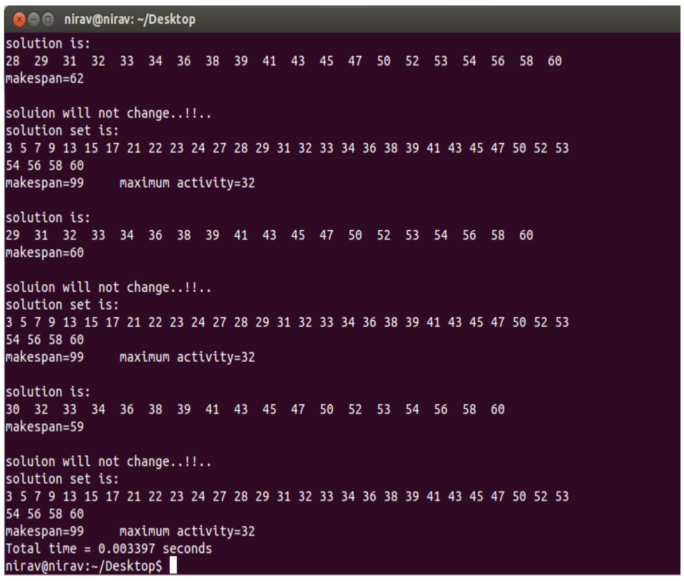


```
iteration 2:
parent1 is:
2 5 7 9 13 15 17 21 22 23 24 27 28 29 31 32 33 34 36 38 39 41 43 45 47 50 52 53 54 56
 58 60

parent2 is:
3 5 7 9 13 15 17 21 22 23 24 27 28 29 31 32 33 34 36 38 39 41 43 45 47 50 52 53 54 56
 58 60

after crossover parent1:
2 5 7 9 13 15 17 21 22 23 24 27 28 29 31 32 33 34 36 38 39 41 43 45 47 50 52 53 54 56
 58 60

after crossover parent2:
3 5 7 9 13 15 17 21 22 23 24 27 28 29 31 32 33 34 36 38 39 41 43 45 47 50 52 53 54 56
 58 60

duration for parent1: 100
number of activities:32

duration for parent2:99
number of activities:32


parent2 is selected with duration: 99 number of activities:32

 Total time = 0.002543 seconds
nirav@nirav:~/Desktop$
```

- Solution using Branch and Bound algorithm

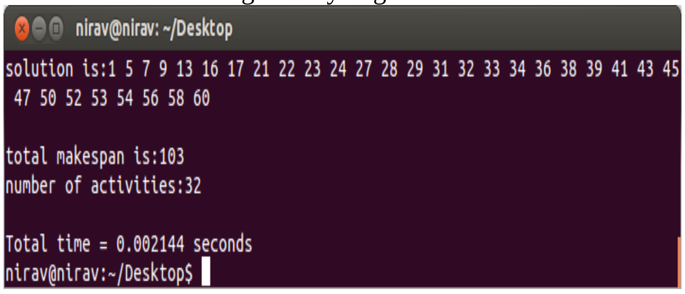


```
solution is:
1  5  7  9  13  15  17  21  22  23  24  27  28  29  31  32  33  34  36  38  39
41  43  45  47  50  52  53  54  56  58  60
makespan=101

solution is:
2  5  7  9  13  15  17  21  22  23  24  27  28  29  31  32  33  34  36  38  39
41  43  45  47  50  52  53  54  56  58  60
makespan=100

previous solution is deleted..!!..new solution is:
2  5  7  9  13  15  17  21  22  23  24  27  28  29  31  32  33  34  36  38  39
41  43  45  47  50  52  53  54  56  58  60
makespan=100
solution set is:
2 5 7 9 13 15 17 21 22 23 24 27 28 29 31 32 33 34 36 38 39 41 43 45 47 50 52 53
54 56 58 60
makespan=100    maximum activity=32
```



```
nirav@nirav: ~/Desktop
solution is:
28  29  31  32  33  34  36  38  39  41  43  45  47  50  52  53  54  56  58  60
makespan=62

soluion will not change..!!..
solution set is:
3 5 7 9 13 15 17 21 22 23 24 27 28 29 31 32 33 34 36 38 39 41 43 45 47 50 52 53
54 56 58 60
makespan=99     maximum activity=32

solution is:
29  31  32  33  34  36  38  39  41  43  45  47  50  52  53  54  56  58  60
makespan=60

soluion will not change..!!..
solution set is:
3 5 7 9 13 15 17 21 22 23 24 27 28 29 31 32 33 34 36 38 39 41 43 45 47 50 52 53
54 56 58 60
makespan=99     maximum activity=32

solution is:
30  32  33  34  36  38  39  41  43  45  47  50  52  53  54  56  58  60
makespan=59

soluion will not change..!!..
solution set is:
3 5 7 9 13 15 17 21 22 23 24 27 28 29 31 32 33 34 36 38 39 41 43 45 47 50 52 53
54 56 58 60
makespan=99     maximum activity=32
Total time = 0.003397 seconds
nirav@nirav:~/Desktop$
```

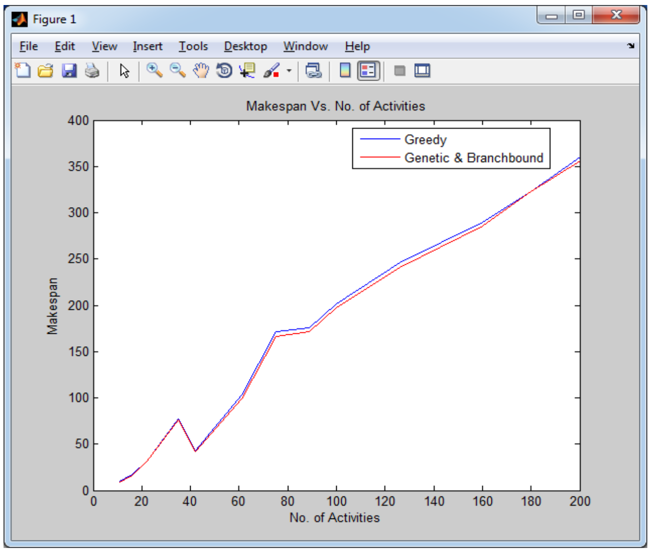
- Solution using Greedy Algorithm

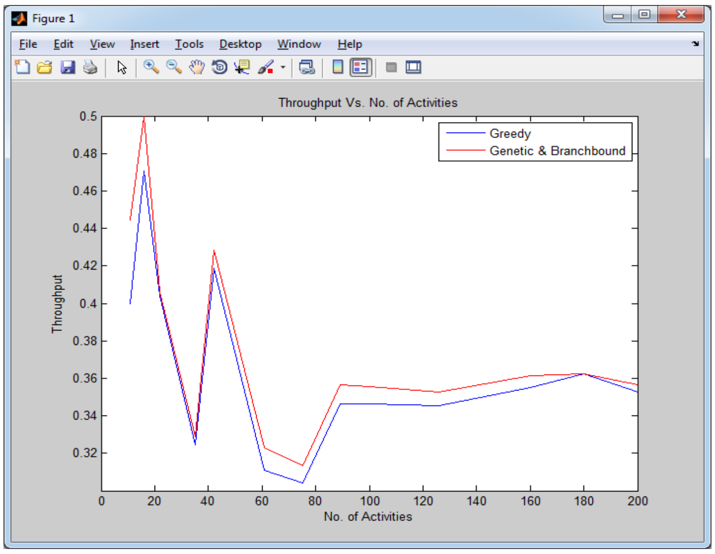


```
nirav@nirav: ~/Desktop
solution is:1 5 7 9 13 16 17 21 22 23 24 27 28 29 31 32 33 34 36 38 39 41 43 45
 47 50 52 53 54 56 58 60

total makespan is:103
number of activities:32

Total time = 0.002144 seconds
nirav@nirav:~/Desktop$
```

V. RESULTS

Here different combinations of activities with start time and finish time are taken and their makespan is evaluated using proposed algorithm (Genetic algorithm and Branch & Bound algorithm) and existing algorithm (Greedy algorithm). Proposed algorithm improves throughput for particular set of activities compared to Greedy algorithm





VI. CONCLUSION

According to results shown in graphs we can conclude that proposed algorithm gives better result than existing algorithm with parameter makespan.

REFERENCES

[1] *ITiCSE*,Active Learning of Greedy Algorithms by Means of Interactive Experimentation, J. Angel Velázquez-Iturbide, Antonio Perez-Carrasco
[2] 2006-IMT-GT conference on mathematics, science and application, A Job-Shop Scheduling Problem (JSSP) Using Genetic Algorithm (GA), Mahanim Omar, Adam Baharum, Yahya Abu Hasan
[3] A Genetic Algorithm for Job Shop Scheduling with Load Balancing, Sanja Petrovic and Carole Fayad
[4] International Journal on Computer Science and Issues,A Genetic Algorithm Approach for Solving Flexible Job Shop Scheduling Problem, Sayedmohammadreza Vaghefinezhadand Kuan Yew Wong
[5] IEEE,An Improved Genetic Algorithm for Solving Flexible Job shop Scheduling Problem, ZHOU Wei, BU Yan-ping, ZHOU Ye-qing
[6] International Journal on Computer Science and Engineering,An Improved Round Robin Scheduling Algorithm for CPU scheduling, Rakesh Kumar Yadav, Abhishek K Mishra, Navin Prakash and Himanshu Sharma
[7] International Journal on Computer Science and Engineering,Burst Round Robin as a Proportional-Share Scheduling Algorithm, Tarek Helmy and Abdelkader Dekdouk
[8] http://en.wikipedia.org/wiki/Activity_selection_problem
[9] http://en.wikipedia.org/wiki/Load_balancing_%28comp utin g%29
[10] Introduction to Algorithm, Thomas H. Coreman