

---

# SOLVING JOB-SHOP SCHEDULING PROBLEM USING TABU SEARCH

---

CSC 489



MARCH 29, 2019  
KING SAUD UNIVERSITY  
RIYADH, SAUDI ARABIA

## Values of parameters and settings

The description of the input data: “Each instance consists of a line of description, a line containing the number of jobs and the number of machines, and then one line for each job, listing the machine number and processing time for each step of the job. The machines are numbered starting with 0”. However, we edited it a little bit and write the number of jobs and the number of machines as parameters (hard coded).

Test instances link:

<http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/jobshop1.txt>

## Our tests parameters values

Number of Jobs =4

Number of Machines =4 also equals number of operations

firstPossibleTime = 0

bestFoundSolution = new int[numberOfJobs];

bestFoundFitness = 100000000;

tabuListSize = (10+numberOfJobs)/numberOfMachines;

maxNumOfIterations = 100

iterations = 10 ,used in the termination condition

numOfDiversification = 2, This attribute indicates the possible number of consecutive Diversification without improvement

countConseqDiversifNoImprovement=0, used for termination criteria

possibleIterWithoutImprovement = 4, execute the diversification procedure when \*possibleIterWithoutImprovement\* iterations is reached without any improvement.

## Result

Input:

instance 1: 3 5 2 3 4 7 1 3 2 10 3 2 4 2 1 4 1 7 2 4 3 6 4 2 3 8 4 1 2 11 1 7

Output:

RUN#	Makespan
1	37
2	37
3	37
4	37
5	37

Input:

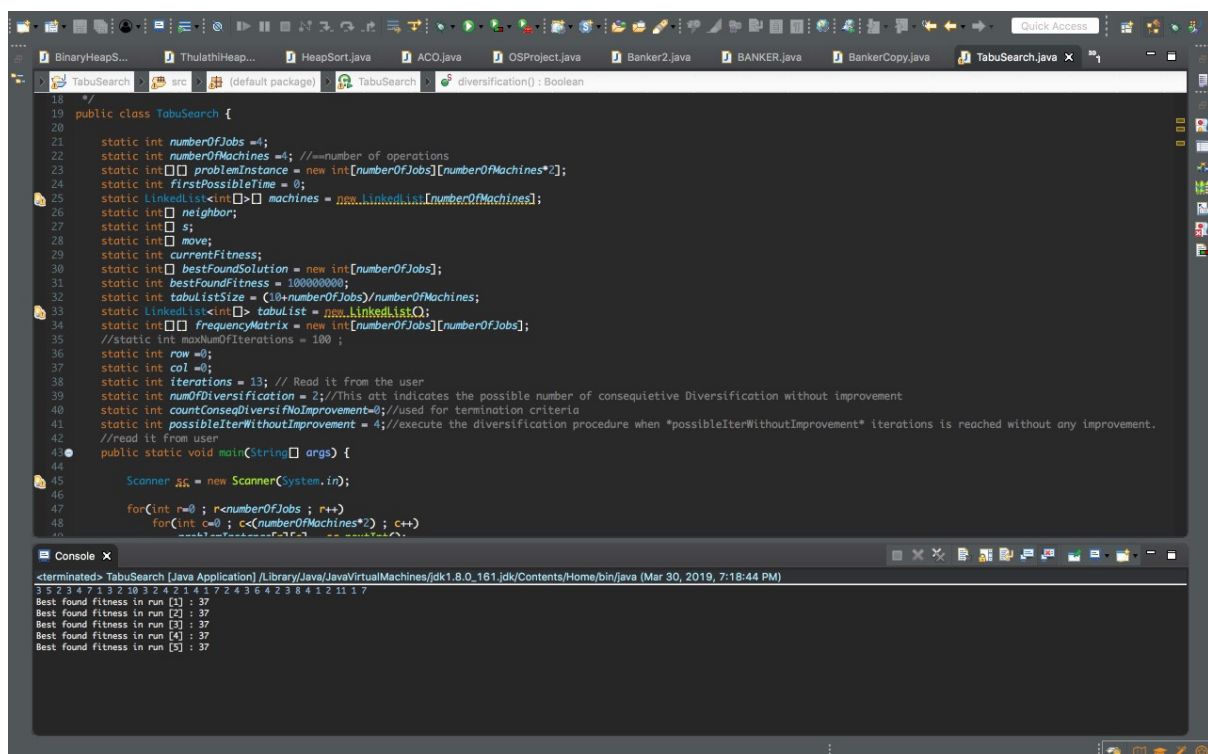
instance 6:

7 62 8 24 5 25 3 84 4 47 6 38 2 82 0 93 9 24 1 66  
5 47 2 97 8 92 9 22 1 93 4 29 7 56 3 80 0 78 6 67  
1 45 7 46 6 22 2 26 9 38 0 69 4 40 3 33 8 75 5 96  
4 85 8 76 5 68 9 88 3 36 6 75 2 56 1 35 0 77 7 85  
8 60 9 20 7 25 3 63 4 81 0 52 1 30 5 98 6 54 2 86  
3 87 9 73 5 51 2 95 4 65 1 86 6 22 8 58 0 80 7 65  
5 81 2 53 7 57 6 71 9 81 0 43 4 26 8 54 3 58 1 69  
4 20 6 86 5 21 8 79 9 62 2 34 0 27 1 81 7 30 3 46  
9 68 6 66 5 98 8 86 7 66 0 56 3 82 1 95 4 47 2 78  
0 30 3 50 7 34 2 58 1 77 5 34 8 84 4 40 9 46 6 44

Output:

RUN#	Makespan
1	950
2	1060
3	945
4	950
5	986

## Screen shots



The screenshot displays an IDE window with the `TabuSearch.java` file open. The code defines a `TabuSearch` class with various static variables and a `main` method. The console output shows the results of the program's execution, including a sequence of numbers and fitness values.

```
18 /*
19 public class TabuSearch {
20
21     static int numberOfJobs = 4;
22     static int numberOfMachines = 4; //==number of operations
23     static int[] problemInstance = new int[numberOfJobs][numberOfMachines*2];
24     static int firstPossibleTime = 0;
25     static LinkedList<int[]> machines = new LinkedList<int[]>(numberOfMachines);
26     static int[] neighbor;
27     static int[] s;
28     static int[] move;
29     static int currentFitness;
30     static int[] bestFoundSolution = new int[numberOfJobs];
31     static int bestFoundFitness = 100000000;
32     static int tabuListSize = (4+numberOfJobs)/numberOfMachines;
33     static LinkedList<int[]> tabuList = new LinkedList<int[]>();
34     static int[] frequencyMatrix = new int[numberOfJobs][numberOfJobs];
35     //static int maxNumOfIterations = 100 ;
36     static int row = 0;
37     static int col = 0;
38     static int iterations = 13; // Read it from the user
39     static int numOfDiversification = 2; //This att indicates the possible number of consecutive Diversification without improvement
40     static int countConseqDiversifNoImprovement = 0; //used for termination criteria
41     static int possibleIterWithoutImprovement = 4; //execute the diversification procedure when "possibleIterWithoutImprovement" iterations is reached without any improvement.
42     //read it from user
43     public static void main(String[] args) {
44
45         Scanner sc = new Scanner(System.in);
46
47         for(int r=0 ; r<numberOfJobs ; r++)
48             for(int c=0 ; c<(numberOfMachines*2) ; c++)
49                 problemInstance[r][c] = (int)(Math.random()*100000000);
50     }
51 }
```

Console Output:

```
<terminated> TabuSearch [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_161.jdk/Contents/Home/bin/java (Mar 30, 2019, 7:18:44 PM)
3 5 2 3 4 7 1 3 2 10 3 2 4 2 1 4 1 7 2 4 3 6 4 2 3 8 4 1 2 11 1 7
Best found fitness in run [0] : 37
Best found fitness in run [2] : 37
Best found fitness in run [3] : 37
Best found fitness in run [4] : 37
Best found fitness in run [5] : 37
```

```
BinaryHeapS... ThulathiHeap... HeapSort.java ACO.java OSProject.java Banker2.java BANKER.java BankerCopy.java TabuSearch.java x
TabuSearch src (default package) TabuSearch
31 /*
32 public class TabuSearch {
33
34     static int numberOfJobs = 10;
35     static int numberOfMachines = 10; //==number of operations
36     static int[] problemInstance = new int[numberOfJobs][numberOfMachines*2];
37     static int firstPossibleTime = 0;
38     static LinkedList<int[]> machines = new LinkedList<int[]>(numberOfMachines);
39     static int[] neighbor;
40     static int[] s;
41     static int[] move;
42     static int currentFitness;
43     static int[] bestFoundSolution = new int[numberOfJobs];
44     static int bestFoundFitness = 100000000;
45     static int tabuListSize = (10+numberOfJobs)/numberOfMachines;
46     static LinkedList<int[]> tabuList = new LinkedList<int[]>();
47     static int[] frequencyMatrix = new int[numberOfJobs][numberOfJobs];
48     //static int maxNumOfIterations = 100 ;
49     static int row = 0;
50     static int col = 0;
51     static int iterations = 10; // Read it from the user
52     static int numofDiversification = 2; //This att indicates the possible number of consecutive Diversification without improvement
53     static int countConseqDiversifNoImprovement = 0; //used for termination criteria
54     static int possibleIterWithoutImprovement = 4; //execute the diversification procedure when "possibleIterWithoutImprovement" iterations is reached without any improvement.
55     //read it from user
56     public static void main(String[] args) {
57
58         7 62 8 24 5 25 3 84 4 47 6 38 2 82 0 93 9 24 1 66
59         5 47 2 97 8 92 9 22 3 93 4 29 7 56 3 89 0 78 6 67
60         1 45 7 46 6 22 2 26 9 38 0 69 4 40 3 33 8 75 5 96
61         4 85 8 76 5 68 9 88 3 36 6 75 2 56 1 35 0 77 7 85
62         8 68 9 28 7 25 3 63 4 81 0 52 1 39 5 98 6 54 2 86
63         3 87 9 73 5 51 2 46 4 65 1 86 6 22 8 58 0 80 7 65
64         5 81 2 53 7 57 6 71 9 81 0 43 4 26 8 54 3 58 1 69
65         4 20 6 86 5 21 8 79 9 62 2 34 0 27 1 81 7 30 3 46
66         9 68 6 65 5 98 8 86 7 66 0 56 3 82 1 95 4 47 2 78
67         0 30 3 58 7 34 2 58 1 77 5 34 6 84 4 40 9 46 6 44
68
69         Best found fitness in run [1] : 950
70         Best found fitness in run [2] : 1060
71         Best found fitness in run [3] : 945
72         Best found fitness in run [4] : 950
73         Best found fitness in run [5] : 986
74
75         <terminated> TabuSearch [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_161.jdk/Contents/Home/bin/java (Mar 30, 2019, 7:22:20 PM)
76
77         Writable Smart Insert 29 : 51
```

```
BinaryHeapS... ThulathiHeap... HeapSort.java ACO.java OSProject.java Banker2.java BANKER.java BankerCopy.java TabuSearch.java x
TabuSearch src (default package) TabuSearch firstPossibleTime : int
16 0 94 6 71 3 81 7 85 1 66 2 90 4 76 5 58 6 93 9 97
17 3 50 0 59 1 82 8 67 7 56 9 96 6 58 4 81 5 59 2 96
18 /*
19 public class TabuSearch {
20
21     static int numberOfJobs = 10;
22     static int numberOfMachines = 10; //==number of operations
23     static int[] problemInstance = new int[numberOfJobs][numberOfMachines*2];
24     static int firstPossibleTime = 0;
25     static LinkedList<int[]> machines = new LinkedList<int[]>(numberOfMachines);
26     static int[] neighbor;
27     static int[] s;
28     static int[] move;
29     static int currentFitness;
30     static int[] bestFoundSolution = new int[numberOfJobs];
31     static int bestFoundFitness = 100000000;
32     static int tabuListSize = (10+numberOfJobs)/numberOfMachines;
33     static LinkedList<int[]> tabuList = new LinkedList<int[]>();
34     static int[] frequencyMatrix = new int[numberOfJobs][numberOfJobs];
35     //static int maxNumOfIterations = 100 ;
36     static int row = 0;
37     static int col = 0;
38     static int iterations = 10; // Read it from the user
39     static int numofDiversification = 2; //This att indicates the possible number of consecutive Diversification without improvement
40     static int countConseqDiversifNoImprovement = 0; //used for termination criteria
41     static int possibleIterWithoutImprovement = 4; //execute the diversification procedure when "possibleIterWithoutImprovement" iterations is reached without any improvement.
42     //read it from user
43     public static void main(String[] args) {
44
45         4 88 8 68 6 94 5 99 1 67 2 83 9 72 1 82 7 94 9 63
46         5 72 3 58 6 69 4 75 2 94 8 66 0 92 1 82 7 94 9 63
47         9 83 8 61 0 83 1 65 6 64 5 85 7 78 4 85 2 55 3 77
48         7 94 2 68 1 61 4 99 3 54 6 75 5 66 0 76 9 63 8 67
49         3 69 4 88 9 82 8 85 0 99 2 67 6 95 5 68 7 67 1 86
50         1 99 4 81 5 64 6 66 8 88 2 80 7 69 9 62 3 79 0 88
51         7 50 1 86 4 97 3 96 0 95 8 97 2 66 5 99 6 52 9 71
52         4 98 6 73 3 82 2 51 1 71 5 94 7 85 0 62 8 55 9 79
53         0 94 6 71 3 81 7 85 1 66 2 90 4 76 5 58 8 93 9 97
54         3 50 0 59 1 82 8 67 7 56 9 96 6 58 4 81 5 59 2 96
55
56         Best found fitness in run [1] : 1169
57         Best found fitness in run [2] : 1169
58         Best found fitness in run [3] : 1349
59         Best found fitness in run [4] : 1314
60         Best found fitness in run [5] : 1169
61
62         <terminated> TabuSearch [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_161.jdk/Contents/Home/bin/java (Mar 30, 2019, 7:20:45 PM)
63
64         Writable Smart Insert 24 : 35
```

