ISSN: 0067-2904

New Improved Heuristic Method for Solving Travelling Salesman Problem

Faez Hassan Ali*, Sajad Majeed Jassim
Department of Mathematics, College of Science, Al-Mustansiryah University, Baghdad, Iraq

Abstract

In this paper we will investigate some Heuristic methods to solve travelling salesman problem. The discussed methods are Minimizing Distance Method (MDM), Branch and Bound Method (BABM), Tree Type Heuristic Method (TTHM) and Greedy Method (GRM).

The weak points of MDM are manipulated in this paper. The Improved MDM (IMDM) gives better results than classical MDM, and other discussed methods, while the GRM gives best time for $5 \leq n \leq 500$, where n is the number of visited cities.

Keywords: Travelling Salesman Problem, Minimizing Distance Method, Branch and Bound Method, Tree Type Heuristic Method, Greedy Method.

طريقة تقريبية محسنة جديدة لحل مسالة البائع المتجول

سجاد مجيد جاسم ، فائز حسن علي*
قسم الرياضيات، كليه العلوم، الجامعة المستنصرية، بغداد، العراق.

خلاصة

في هذا البحث سيتم تنفيذ بعض الطرق التقريبية لحل مسالة البائع المتجول. هذه الطرق التقريبية هي: طريقة المسافة الاصغر (MDM)، طريقة التقيد والتفرع (BABM)، طريقة الشجرة التقريبية (TTHM) وطريقة الشراهة (GRM).

في هذا البحث تم معالجة نقاط الضعف بطريقة MDM. طريقة المسافة الاصغر المحسنة (IMDM) اعطت نتائج افضل بكثير من طريقة MDM التقليدية ومن بقية الطرق التقريبة الاخرى، فيما اعطت طريقة GRM افضل زمن لعدد مدن n، حيث ان $5 \leq n \leq 500$.

1. Introduction

Traveling Salesman Problem (TSP) consists of set of cities (say n), between any pair of cities there exist a way (path).The pair of cities has known distance each of these paths have specified distance or time of the travel, the cost or the trip (all considered distance). Traveling salesman starts from specific city and then travel to all other cities without passes from the same city again, finally returns to the origin city. The main objective of TSP is finding the complete path where the total distance be reduced by the seller during the trip [1].

Many mathematical problems may relate to TSP studied in the start of 18^{th} century by the two mathematicians sir W. R. Hamilton and T. P. Kirkman. Hamilton found an Icosian game in 1857 which required from a player to complete path or tour using specific connectors for 20 points. However, K. Menger and Harvard are the first who studied the general form of TSP in the late 1920's or early 1930's [2].

*Emil: faezhassan@uomustansiriyah.edu.iq

In the mid-1950's, many solution methods of TSP started to appear many papers, these papers used simple variations of the term TSP. In 1954, Dantzig, Fulkerson, and Johnson put a description of a method for solving TSP and showing the power of the proposed method by solving an instance with 49 cities. In 1972, R. M. Karp showed that the Hamiltonian cycle problem (HCP) was NP-complete, and since TSP is a HCP, then its NP-complete, (NP-hard, is a class of problems that are informally the hardest problems in NP which means no polynomial-time algorithm is known for solving TSP) [3].

The main outlines of this paper are as follows: in section 2 we discuss the digraph representation since TSP can be treated as graph. While in section 3 we will show the concept of TSP with some applications. The mathematical formulation of TSP introduced in section 4. Some heuristic methods of TSP are discussed in section 5. In section 6 we suggest some improvements for MDM to construct a new method called IMDM. Many practical examples are tested in section 7 using some discussed heuristic methods for TSP including the IMDM. Lastly some conclusions are introduced in section 8.

2. Digraph Representation

Definition (1) [4]: A **graph** (G) is a finite set of points, called **nodes or vertices (V)**, together with a finite set of **edges**, each joins a pair of vertices. A **loop** is an edge joining a vertex to itself.

Definition (2) [4]: If G is a graph that has n vertices, then the **matrix A(G) (**called the **adjacency matrix** of G**)**, whose i^{th}, j^{th} element is 1 if there is at least one edge between V_i and V_j and zero otherwise.

Definition (3) [4]: A **directed graph** or a **digraph** is a graph has a finite set of **directed edges**, each of which joins an ordered pair of distinct vertices.

Remark (1):

- A digraph G contains no loops.
- There are no multiple edges.
- The directed edge V_iV_j in general is different from V_jV_i.
- A(G) of G of a digraph may be symmetric.

Definition (4): The **tour length** is the number of edges which are connected at least two nodes.

Definition (5): The **cyclic tour** is the tour which ends with same nodes with length L, s.t $2 \leq L < n$.

Definition (6): The **complete tour** is cyclic tour with length L=n.

3. Concept of TSP and Applications

TSP is a hard problem which its solution has eluded many mathematicians for many years. Currently there is no specific solution to this problem that has satisfied mathematicians.

The most popular application of the TSP is finding a route that a travelling salesman would take to visit every geographical location in a specified list such that the minimum total distance is obtained [5].

Consider a number of machines in an assembly line. There are some machines whose main purposes are to drill different holes in a piece of material. This material may be the frame of a vehicle, a circuit board, or a piece of wood to be used as a book shelf. The drill has a specific positions specified by motors that move along tracks such that the drill could move to any position within a specific area. Find a solution to the TSP could be useful to find the optimal or best order in which the all the holes should be drilled [5].

Another application when a solution to the TSP can be useful to is mechanical or electronic connection placement. Consider the wiring of a specific circuit board, or the electrical wiring for a large building, or even the plumbing layout for a building. In many of these samples, the connections must to be laid out such that all the components are connected in a cycle [5].

4. Mathematical Formulation of TSP

TSP model is defined by the number of n cities and the distance matrix $\|d_{ij}\|$, $i,j=1,\ldots,n$ ($d_{ii}=\infty$) be the distance between city i and city j. we defined the matrix X interpreted as follows [6]:

$$x_{ij} = \begin{cases} 1, \text{if city j is reached from city i} \\ 0, \text{otherewise} \end{cases}$$

Then the Matrix $X=[X_{ij}]_{n\times n}$ is:

$$X=\begin{bmatrix} 0 & x_{12} & \dots & x_{1n} \\ x_{21} & 0 & \dots & x_{2n} \\ \vdots & \vdots & \dots & \vdots \\ x_{n1} & x_{n2} & \dots & 0 \end{bmatrix}$$

The TSP model is given as:

$$\left.\begin{aligned} &\text{Minimize } z=\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}x_{ij} \\ &\text{Subject to} \\ &\sum_{j=1}^{n} x_{ij}=1, i=1,...,n \\ &\sum_{i=1}^{n} x_{ij}=1, j=1,...,n \\ &x_{ij}=0, 1. \end{aligned}\right\} \quad \dots(1)$$

TSP can be classified into two types; symmetrical or asymmetrical. In the symmetrical one, the distance between the two nodes or cities does not depend on the distance of the travel. For instance, if we have the distance between two nodes called i and j by d_{ij} and if we have $d_{ji}=d_{ij}$ TSP is symmetric otherwise it is asymmetrical [7]. In this work we deal with the issue of asymmetry in the distance matrix because it's the general situation.

We discuss the solving of TSP according to existence of successive rules (SR), in this paper SR means precedence rules between two direct nodes exists or not. We choose a simple example and solve this example using the complete enumeration method (CEM) (or known brute force), which is to find all possible permutations of the path ((n-1)!).

To describe this situation let solve the TSP by using CEM for the following example.

Example (1): Let n=4 and for the following asymmetrical TSP:

	A	B	C	D
A	−	2	3	7
B	1	−	3	4
C	4	5	−	6
D	1	2	5	−

Remark (2): The table above can be represented by $D=[d_{ij}]$

The probable solutions are:

1. ABCDA, Z=12.
2. ABDCA, Z=15.
3. ACBDA, Z=13.
4. ACDBA, Z=12.
5. ADBCA, Z=16.
6. ADCBA, Z=18.

The optimal solution is Z=12 for the two path (1) and (4).

Now suppose we have the following SR: **(D→C)** for same example:

1. ADCBA, Z=18.
2. ABDCA, Z=15.

The optimal solution is Z=15 for the path (2).

Remark (3): In this paper the two symbols (∞) and (−) have the same meaning, where no path or rout is between the pair of nodes or it's the same node.

As special case for TSP we introduce the following proposition:

Proposition (1): For TSP, if $d_{ij}=d$ ∀i,j=1,..,n, then any sequence will be an optimal sequence.

Proof: Since TSP has n cities and n edges, then $Z = \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}x_{ij} = d\sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij} = d \cdot n$ for any sequence. ■

5. Heuristic Methods for Solving TSP

The term "**heuristics**", in general, used for algorithms to find solutions among all possible ones. However, the heuristic algorithms do not guarantee that the best solution will be found; but they usually find a solution close to the best or the optimal one and they find it fast and easily. Sometimes these heuristics can be accurate, i.e. they may find the best solution, but the algorithm is still treated as heuristic algorithm until the best solution is proven to be optimal [8]. Among available heuristics are Greedy Method (GRM), Tree Type Heuristic Method (TTHM), Branch and Bound Method (BABM) and Minimizing Distance Method (MDM).

5.1 Greedy Method (GRM)

This algorithm starts by sorting the edges by length, and always adding the shortest remaining available edge to the tour. The shortest edge is available if it is not yet added to the tour and if adding it would not create a 3-degree vertex or a cycle with edges less than n. This heuristics can be applied to run in $O(n^2\log(n))$ time [9]. The main steps of the GRM are as follows:

Greedy Method (GRM) Algorithm [10]

Step 1: Input n, $D=[d_{ij}]$, i, j=1,...,n.
Step 2: From the first row, choose the lowest number (x), and delete the column x.
Step 3: In the row x, find the least undeleted matrix cell entry (y) (except col. (1)) and identify the corresponding column (break tie randomly), y is the next city in the tour.
Step 4: Delete the column y in D.
Step 5: If (n-1) columns are deleted, goto **Step 7**, else goto **Step 6**.
Step 6: Set x = y and goto **Step 3**.
Step 7: Include the first city as the last city in the tour.
Step 8: Calculate the total cost (Z) of the obtained tour.

5.2 Tree Type Heuristic Method (TTHM) [11]

The main step in this method is the objective function evaluated at all nodes in each level of the search tree, then some of the nodes within each level (with minimum or maximum value) of the search tree are chosen from which to branch. Usually, one node is chosen with each level and stop at the first complete sequence of the nodes to be the solution. The algorithm of the TTHM is as follows:

Tree Type Heuristic Method (TTHM) Algorithm

Step 1: Input n, $D=[d_{ij}]$, $i,j=1,\ldots,n$, $C_{ij}=0$.
Step 2: Start with level i=1.
Step 3: the minimum cost (z_{ij}) is evaluated at all nodes (j) in level (i), then branch from the all least cost nodes.
Step 4: i=i+1, $C_{ij}=C_{ij}+z_{ij}$.
Step 5: If i<=n goto **Step 3**.
Step 6: Output $Z=\min(C_{ij})$.

5.3 Branch and Bound Method (BABM)

The terms "Branch and Bound" represent all the state space search methods such that all the children of E-node are generated any now nodes called line node when it became E-node. E-node is the node, which can be expended. The live-node is node generated all of whose children are not yet been expanded. A node which cannot be expanded called dead node, but this node can be useful for backtracking concept. If there are no more children to expand then we have to reach its parent and expand its children and we do so until we obtain the solution or complete tree path [12]. The algorithm of the BABM is as follows:

Branch and Bound Method (BABM) Algorithm

Step 1: Input n, $D=[d_{ij}]$, i, j=1,...,n.
Step 2: Reduce each row and column to ensure that be at least one zero on them (this done by finding the minimum value and then subtract it from each element in each row and column).
Step 3: Calculate the total expected cost of expanding root node (L_i).
Step 4: L(node)=L(parent node)+Parent(i,j)+total cost of reduction.
Step 5: Branch from minimum L(break the tie arbitrarily).

Step 6: If its remain one node goto **Step 7**, else goto **Step 2**.
Step 7: Output the complete tour and the total cost Z.

5.4 Minimizing Distance Method (MDM) [13]

This method is one of the best methods mentioned above in terms of results, we have to reduce matrix by subtracting the minimum element from each row and subtracting the minimum element from each column so that in each row and column there is zero. Then we calculate the penalties for each zero, we collect the smallest number in the row and column of this zero. Row and column are deleted. This zero is cancelled by the item with the opposite position of zero of the matrix and continued this process until the dimension of reduced matrix is 2.At the end we link the resulting paths and calculate the cost (see example (2)). The algorithm of the MDM is as follows:

Minimizing Distance Method (MDM) Algorithm

Step 1: Input n, D=[d_{ij}], i, j=1,...,n.
Step 2: Reduce the matrix D s. t. must be at least one zero occurs in each row and column by do the following: Let R_k be the minimum element in row_i, then $row_i=row_i-R_k$, i=1,…,n, $1\le k\le n$, and C_k be the minimum element in Col_j, then $Col_j=Col_j-C_k$, j=1,…,n, $1\le k\le n$.
Step 3: Calculate penalties $P_{ij}(0)=\min(row_i)+\min(col_j)$ for all 0's.
Step 4: Select the $\max(P_{ij}(0))$ (choose arbitrarily, if more than one $P_{ij}(0)$ are maximum) to obtain the single path i→j, and delete the row_i and col_j and set $d(row_j,col_i)= -$, of the matrix D.
Step 5: Check whether all the rows and columns of matrix D are deleted. If yes, go to **Step 6**; otherwise go to **Step 2**.
Step 6: Link all resulting single paths together and calculate the total cost Z.
Example (2): Lets have the following TSP:

	A	B	C	D	E	F
A	−	8	9	3	6	6
B	3	−	5	10	2	9
C	9	1	−	3	2	10
D	4	8	6	−	1	4
E	9	1	3	4	−	3
F	3	7	8	3	9	−

After applying MDM we obtain the following routs:

A→D, F→A, B→E, C→B

While the last reduce matrix is:

	C	F
D	2	0^2
E	0^2	0^0

By MDM rules we have to **D→F** and **E→C**.
Then when linking the above routs, we obtain the following cyclic route: **A→D→F→A**, thus we have a cyclic path with length less then n=6 and that is one of the weak points of MDM.

6. Improving Minimizing Distance Method (IMDM)

The method MDM is an efficient method for finding a good solution, but through this paper we discover some weak points. These weak points can summarize as follows: The general weak point of the MDM is: the cyclic in the obtained path by MDM with length L < n (Table-1)). The number of cyclic paths increased as n increased. We mentioned before that if many equal $\max(P_{ij}(0))$ occurred, we choose one of them arbitrary. The chosen one may cause cyclic path and this may be known in the end of applying MDM.

In order to manipulate this weak point of MDM we suggest an improving for MDM to increase its achievement and efficiency. The manipulate of the MDM weak points can be summarized as follows: first when more than one $P_{ij}(0)$ are maximum, we choose one arbitrary and link it with obtained path(s) previously it produces a path $\delta(L)$, $L\ge 2$. If this choice not leads to cyclic path $\delta(L)$ with L<n, we have to set $d(\delta(L+1),\delta(m))= -$, for m=1,…,L-1. If this choice leads to cyclic path we set it as $d_{ij}= -$, then re-choose another one; if all choices lead to a cyclic path we will choose direct less value than the

penalty cost of last max($P_{ij}(0)$) we stopped at. If all 0's in D leads to cyclic path with L<n, we re-reduce the matrix D. This new procedure continues until get non-cyclic path (complete path) until complete all nodes (cities) of the given matrix D (see Table-1) and example (3)). The algorithm of the improved MDM (IMDM) is as follows:

Improved Minimizing Distance (IMDM) Algorithm

Step 1: Input n, D=[d_{ij}], i, j=1,...,n, path $\delta = \phi$ with length L=0.

Step 2: Reduce the matrix D must be obtaining at least one zero in each row and column by do the following: Let R_k be the minimum element in row_i, then $row_i=row_i-R_k$, i=1,…,n, $1\le k\le n$, and C_k be the minimum element in Col_j, then $Col_j=Col_j-C_k$, j=1,…,n, $1\le k\le n$.

Step 3: Calculate penalties $P_{ij}(0)=\min(row_i)+\min(col_j)$ for all 0's.

Step 4: Select the max($P_{ij}(0)$) (choose arbitrarily, if more than one $P_{ij}(0)$ are maximum) to obtain the single path $i\rightarrow j$, $\delta=\delta+i\rightarrow j$, L=L+1.

Step 5: If δ cyclic and if L < n, set $d_{ij}=\infty$, $\delta = \delta- i\rightarrow j$, L=L-1.

Step 6: If all 0's of D imply to δ with L < n then goto **step 2**, else goto **step 4**.

Step 7: If δ cyclic and if L = n, goto **step 9**.

Step 8: If δ not cyclic then delete the row_i and col_j and $d(row_j,col_i)= -$ of the matrix D, and set $d(\delta(L+1),\delta(m))= -$, for m=1,…,L-1, goto **step 2**.

Step 9: Calculate the total cost Z(δ).

The block diagram of IMDM is shown in Figure-1.

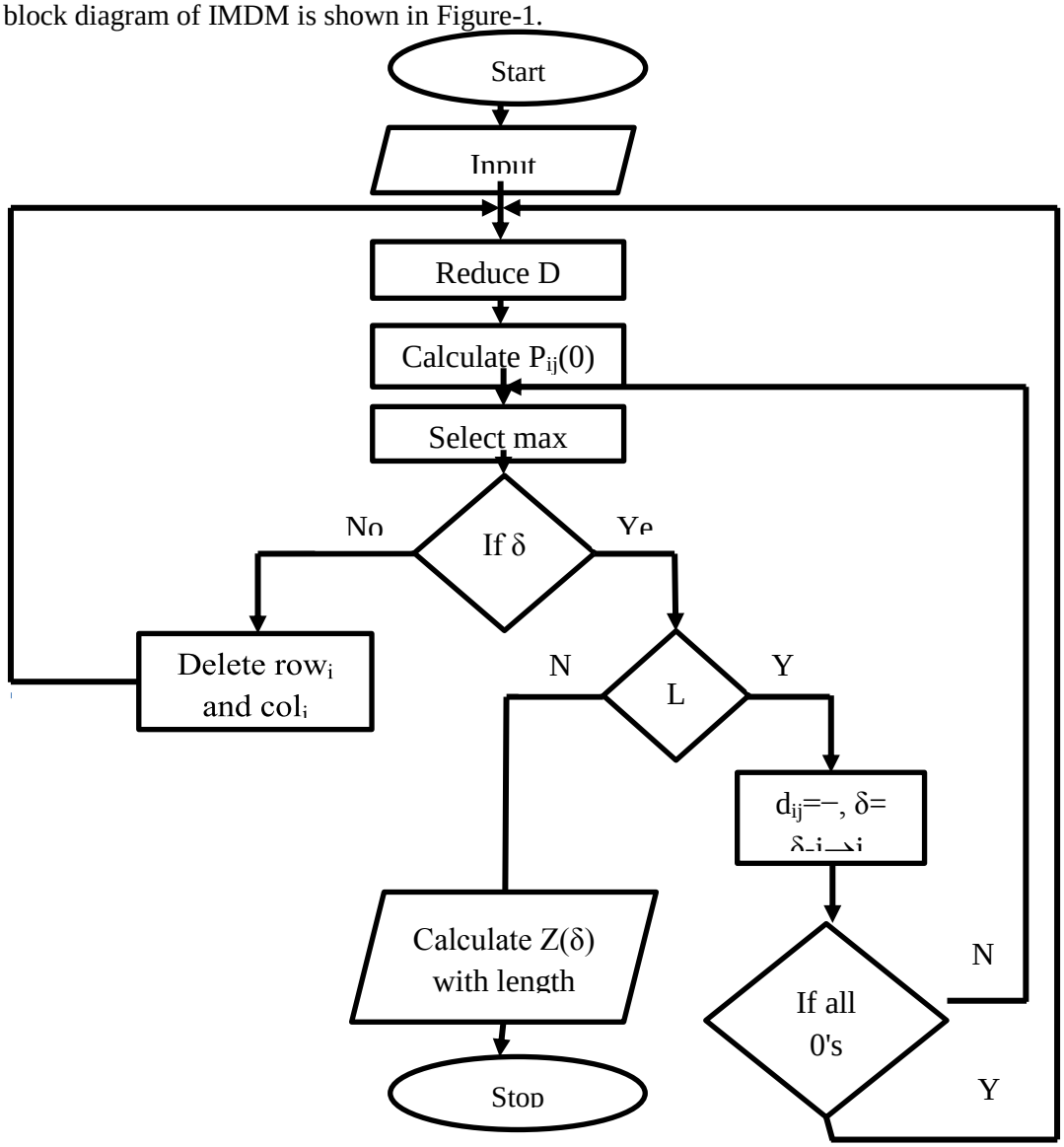

Figure 1-The block diagram of IMDM.

Example (3): Recall example (2):
1st step: we have to minimize row and column.

	A	B	C	D	E	F
A	−	5	4	0	3	1
B	1	−	1	8	0	5
C	8	0	−	2	1	7
D	3	7	3	−	0	1
E	8	0	0	3	−	0
F	0	4	3	0	6	−

2nd step: calculate penalties of all 0's ($\min(row_i)+\min(col_j)$). All rows and columns have at least one zero (choose maximum penalty if equal choose arbitrarily).

	A	B	C	D	E	F
A	−	5	4	0^1	3	1
B	1	−	1	8	0^1	5
C	8	0^1	−	2	1	7
D	3	7	3	−	0^1	1
E	8	0^0	0^1	3	−	0^1
F	0^1	4	3	0^0	6	−

Maximum penalty is 1, for more than path, we choose **A→D** arbitrary.
Cross out row A and column D. So we have the following reduced matrix (set cell D-A to (−)).

	A	B	C	E	F
B	1	−	1	0	5
C	8	0	−	1	7
D	−	7	3	0	1
E	8	0	0	−	0
F	0	4	3	6	−

Since at least one zero occurs in each row and column condition is satisfied, then finds the maximum penalty:

	A	B	C	E	F
B	1	−	1	0^1	5
C	8	0^1	−	1	7
D	−	7	3	0^1	1
E	8	0^0	0^1	−	0^1
F	0^4	4	3	6	−

Maximum penalty is 4. Choose **F→A**.

	B	C	E	F
B	−	1	0^1	5
C	0^1	−	1	7
D	7	3	0^1	1
E	0^0	0^1	−	0^1

Since at least one zero occurs in each row and column condition is satisfied, then finds the maximum penalty which is 1, we have **B→E**, (set the cell E-B to (−)), now the reduced matrix is:

	B	C	F
C	0^{13}	−	7
D	6	2	0^2
E	−	0^2	0^0

After applying IMDM steps we obtain the following reduced matrix:

	B	C	F
C	0^{13}	−	7
D	6	2	0^2
E	−	0^2	0^0

We have **C→B**. Now we will link the current path with the remaining paths that are linkable to it, **C→B→E**, So set d(E,C) = −.

	C	F
D	2	0
E	−	0

We must choose **D→C** and **E→F**.

So we have the following routs: **A→D, F→A, B→E, C→B, D→C** and **E→F**.

Then we obtain the following complete route: **A→D→C→B→E→F→A**, Z=18.

Table-1shows the comparison results between MDM and IMDM.

Table 1- Comparison results between MDM and IMDM.

n	MDM			IMDM	
	Cost	Time	Cyclic length	Cost	Time
5	37	R	-	37	R
6	49	R	-	49	R
7	63	R	-	63	R
8	61	R	-	61	R
9	-	-	7	57	R
10	39	R	-	39	R
11	-	-	4	69	R
12	40	R	-	40	R
20	-	-	17	71	R
50	-	-	12	77	R
70	-	-	45	92	1.1
100	-	-	56	116	3.3
No. of solved problem	6			12	
Ratio of solved problem	50%			100%	

7. Practical Examples

In the practical examples we will choose different n such that $5\leq n\leq 500$ with integer distance such that $d_{ij}\in[1,30]$ for $5\leq n\leq 30$, $d_{ij}\in[1,100]$ for $40\leq n\leq 90$ and $d_{ij}\in[1,150]$ for $100\leq n\leq 500$. All these examples are tested by heuristic methods (BABM, TTHM and GRM) adding to IMDM, all the tested methods are compared with CEM for $5\leq n\leq 12$. The most used important notations are:

n: number of cities

Av-C: Average of costs.

Av-T: Average of time.

AE: absolute error between best method and other methods.

Av-all: Average of all Av-C or Av-T.

Table-2 shows the comparison results between CEM from one side and IMDM, BABM, TTHM and GRM from another side for n=5…12.

Table 2-Comparison results between CEM with IMDM, BABM, TTHM and GRM for n=5…12.

n	CEM		IMDM			BABM			TTHM			GRM		
	Av-C	Av-T	Av-C	Av-T	AE	Av-C	Av-T	AE	Av-C	Av-T	AE	Av-C	Av-T	AE
5	51	R	51.7	R	0.7	51.7	R	0.7	58.7	R	7.7	58.7	R	7.7
6	46.3	R	50.3	R	4	55	R	8.7	59.7	R	13.4	59.7	R	13.4
7	53	R	53	R	0	53	R	0	67	R	14	67	R	14
8	56.7	R	56.7	R	0	68.3	R	11.6	83	R	26.3	83	R	26.3
9	43.3	R	43.3	R	0	43.3	R	0	69.3	R	26	69.3	R	26
10	52.7	3.8	52.7	R	0	53.3	R	0.6	63.3	R	10.6	61.7	R	9
11	61.7	37.1	63.7	R	2	67.3	R	5.6	78.3	R	16.6	77.7	R	16
12	41.7	409.1	43.7	R	2	52.7	R	11	70	R	28.3	73.7	R	32
Av-all	50.8	56.6	51.9	R	1.1	55.6	R	4.8	68.7	R	17.9	68.9	R	18.1

From the results of Table-1 we notice that IMDM is the best method for accuracy from other methods so it can be compared with other methods for n > 12.
Figure-2) shows the comparison results of Table-2.

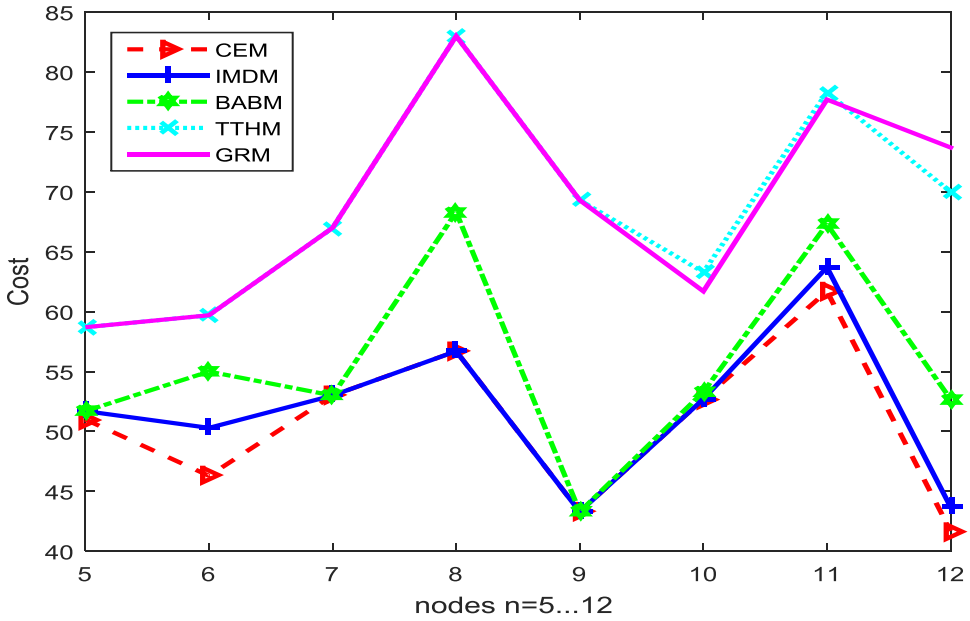

Figure 2-Comparison results between CEM with IMDM, BABM, TTHM and GRM for n=5…12.

Table-3 shows the comparison results between IMDM from one side and BABM, TTHM and GRM from another side for n=20:10:90.
Table 3-Comparison results between IMDM with BABM, TTHM and GRM for n=20:10:90.

n	IMDM		BABM			TTHM			GRM		
	Av-C	Av-T	Av-C	Av-T	AE	Av-C	Av-T	AE	Av-C	Av-T	AE
20	60	R	79	R	19	101.3	R	41.3	68.7	R	8.7
30	58	R	66.7	R	8.7	109	R	51	117.3	R	59.3
40	182	R	260	R	78	382	R	200	340	R	158
50	201.3	R	292	R	90.7	450	R	248.7	334.3	R	133
60	227	R	292.7	R	65.7	416	R	189	426.3	R	199.3
70	233.3	R	295.3	1.3	62	474.3	R	241	462.7	R	229.4
80	236.7	1.4	311	1.9	74.3	497.7	R	261	520.7	R	284
90	247	1.9	314	2.8	67	472.7	R	225.7	503.3	R	256.3
Av-all	180.7	0.8	238.8	1.1	58.2	362.9	R	182.2	346.7	R	166

Figure-3 show the comparison results of Table-3.

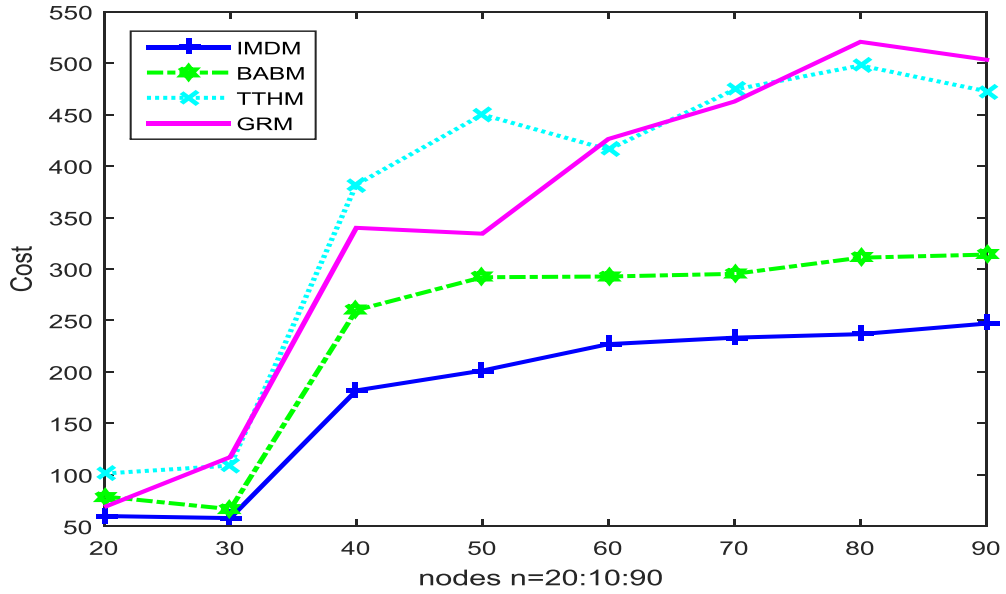

Figure 3-Comparison results between IMDM with BABM, TTHM and GRM for n=20:10:90.

Table-4 shows the comparison results of IMDM from one side with BABM and GRM from another side for n=100:100:500.

Table 4-Comparison results of IMDM with BABM and GRM for n=100:100:500.

n	IMDM		BABM			GRM		
	Av-C	Av-T	Av-C	Av-T	AE	Av-C	Av-T	AE
100	308.3	2.9	415.3	3.9	107	691.3	R	383
200	387	18.4	523.7	41.5	136.7	915.3	R	528.3
300	436.7	62.5	664	174.8	227.3	1152	1.1	715.3
400	522.7	316.8	656.3	554.4	133.6	1213.3	1.9	690.6
500	568.3	761.4	831.3	1283.5	263	1262.7	3.2	694.4
Av-all	444.6	232.4	618.1	411.6	173.5	1046.9	1.4	602.3

In this table we exclude the TTHM because it takes high CPU time to solve the experimental examples.

Figure-4 shows the comparison results of Table-4.

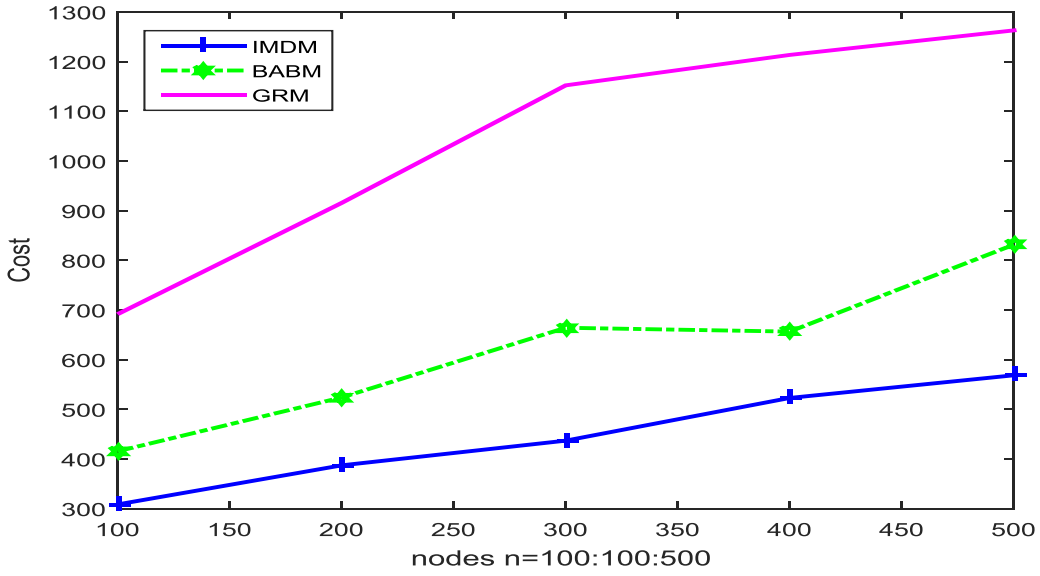

Figure 4-Comparison results of IMDM with BABM and GRM for n=100:100:500.

All the above tables are considered without successive rules (WOSR).

Table-5 describes the results of comparing of BABM, TTHM and GRM WOSR or with successive rule (WSR) for n=70:10:100, and 100:100:500 and shows the number of SR (NSR).

Table 5-Comparing of BABM, TTHM and GRM WORS or WSR for n=70:10:100, and 100:100:500.

n	NSR	BABM	WOSR	WSR	TTHM	WOSR	WSR	GRM	WOSR	WSR
		Av-C	Av-T	Av-T	Av-C	Av-T	Av-T	Av-C	Av-T	Av-T
70	14	295.3	1.3	R	474.3	R	R	462.7	R	R
80	16	311	1.9	1.1	497.7	R	R	520.7	R	R
90	18	314	2.8	1.6	472.7	R	R	503.3	R	R
100	20	415.3	3.9	2.3	672	R	R	691.3	R	R
200	40	523.7	41.5	24.3	936.3	8.8	1.7	915.3	R	R
300	60	664	174.8	105.3	-	-	-	1152	1.1	R
400	80	656.3	554.4	360.4	-	-	-	1213.3	1.9	R
500	100	831.3	1283.5	838.7	-	-	-	1262.7	3.2	R

Figure-5 shows the comparison results of Table-5 for CPU time.

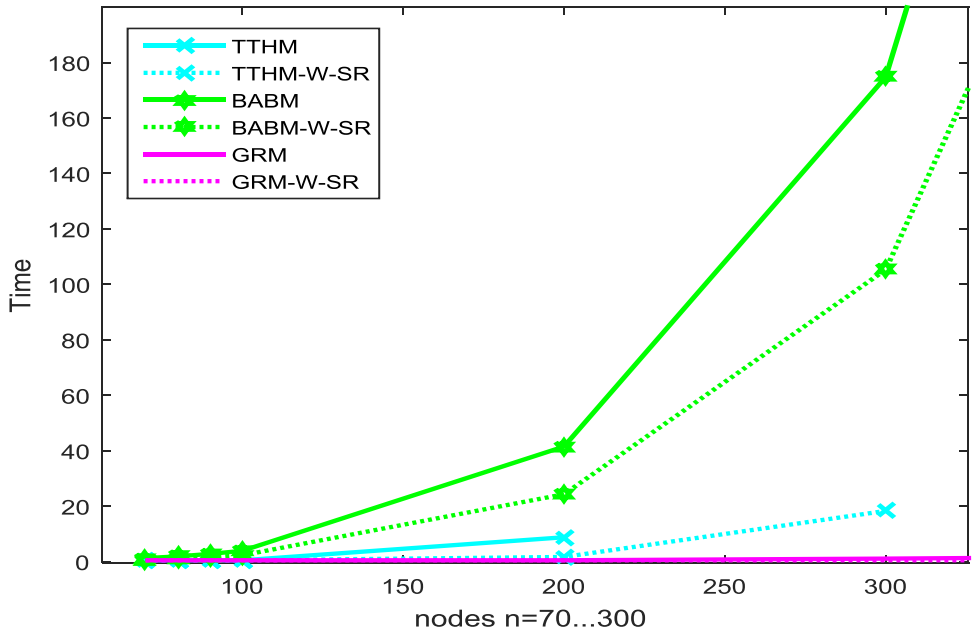

Figure 5-Comparing of BABM, TTHM and GRM with or without SR for n=70:10:100, and 100:100:300.

8. Conclusions

1. The present study manipulates some weak points, in MDM to improve the achieving and that is clear from the ratio of solved problem for MDM is 50% while for IMDM is 100% (see Table-1)).
2. The cost results of Tables-(2, 3, 4) describe the good efficiency of IMDM for different number of cities (n).
3. For the time criterion, we notice that GRM is the best method from the other approximated methods.
4. The existence of SR gives improved just in time without any effect on cost results (see Table-5).
5. As future work, we recommend to use exact methods (Branch and Bound Technique and Dynamic programing) and local search methods (Genetic Algorithm, Particle Swarm Optimization, Bees Algorithm,… etc.) to solve TSP.

References

1. Hosseinabadi A. R., Yazdanpanah M. and Rostami A.S. **2012**. "A New Search Algorithm for Solving Symmetric Traveling Salesman Problem Based on Gravity", *World Applied Sciences Journal*, **16**(10): 1387-1392.
2. Saiyed A.R. **2012**. "*The Traveling Salesman problem*", Indiana State University Terre Haute, IN 47809, USA, April 11, 2012.
3. Cook, William. **2010**. "*History of the TSP." The Traveling Salesman Problem*. Oct, 2009. Georgia Tech, 22 Jan 2010. <http://www.tsp.gatech.edu/index.html>.
4. Kolman, B. **1988**. "*Introductory Linear Algebra with Applications*", Macmillan Publishing company.
5. Leonardo Z. **2006**. "*The Traveling Salesman Problem A Comprehensive Survey*", Submitted as a project for CSE 4080, Fall.
6. Hamdy A. T. **2011**. "*Operations Research an Introduction*", Publishing as Prentice Hall, New Jersey 07458, 9th, 2011.
7. Hejazi, S.R. and Soltani, R. **2005**. "The implementation of combined ant colony and genetic algorithm for solving traveling salesman problem", in the proceeding of 4th International Industrial Engineering Conference.
8. Lawler, E. L. and Lenstra, J. K. **1985.** "*The Traveling Salesman Problem*", Wiley-Interscience series in discrete mathematics and optimization, John Wiley and Sons, New York, 1985.
9. Johnson, D.S. and McGeoch, L.A. **1997.** "The traveling salesman problem: A case study in local optimization", *Local search in combinatorial optimization*, **1**: 215–310.
10. Panneerselvam, R. **2007.** "*Design and Analysis of Algorithms*", Publishing as Prentice Hall of India, New Delhi 110001.
11. Ali, F. H. **2015.** "Improving Exact and Local Search Algorithms for Solving Some Combinatorial Optimization Problems", P.h. D. thesis College of Science, Al-Mustansiriyah University.
12. Rastogi, A., Shrivastava, A.K, Payal, N. and Singh R. **2013.** "A Proposed Solution to Travelling Salesman Problem using Branch and Bound", *International Journal of Computer Applications* (0975 – 8887) **65**(5), March.
13. https://www.youtube.com/watch?v=vNqE_LDTsa0, **2015**.