A New Approach to Solving the Fixed Destination Multi-depot Multiple Traveling Salesman Problem Using Genetic Algorithms

Kamran Kardel
Department of Industrial Engineering, University of Science and Technology Mazandaran Babol, no. 256, Keshvari st. Iran
k_kardel@ustmb.ac.ir

Nikbakhsh Javadian
Department of Industrial Engineering, University of Science and Technology Mazandaran Babol, no. 256, Keshvari st., Iran
nijavdian@ustmb.ac.ir

Fereydoun Adbesh
Department of Industrial Engineering, University of Science and Technology Mazandaran Babol, no. 256, Keshvari st. Iran
f_adbesh@ustmb.ac.ir

Abstract

The fixed destination multi depot multiple traveling salesmen problem (MmTSP) in which more than one salesmen depart from several starting cities and having returned to the starting city, form tours so that each city is visited with exactly one salesman and the tour lengths stay within certain limits. This problem is of a great complexity and few investigations have been done on it before. In this paper we propose a new GA chromosome and related operators for the MmTSP and compare the theoretical properties and computational performance of the some common operators and strategies of genetic algorithms. We also compare results in problems with small dimensions to the optimal answers obtained by solving the problems by Lingo 8.

Keywords: Genetic algorithms, multiple traveling salesmen problem, multi depot, fixed destination

1 Introduction

A generalization of the well-known Traveling Salesman Problem is the standard multiple Traveling Salesman Problem (mTSP). The problem can be defined simply as the determination of a set of routes for *m* salesmen who all start from and return to a single home city (depot). The mTSP finds applications in printing press scheduling [1], crew scheduling [2], mission planning for autonomous mobile robots [3], hot rolling scheduling [4] and interview scheduling [5].

The multidepot mTSP (MmTSP) is a generalization of the single depot mTSP, such that there is more than one depot (*d* depot) and a number of salesmen at each depot (m_k). If the problem is to determine a total of *m* tours such that the *m* salesmen must return to their original depots, this is referred to as the fixed destination MmTSP. On the other hand, if the salesmen do not have to return to their original depots but the number of salesmen at each depot should remain the same at the end as it was in the beginning, we have the nonfixed destination MmTSP [6].

In Fig. 1 (a) we illustrate the mTSP with 8 cities and 2 salesmen. Also in Fig. 1 (b) we illustrate the MmTSP with 12 cities, 4 salesmen, 2 depots and 2 salesmen in each depot.

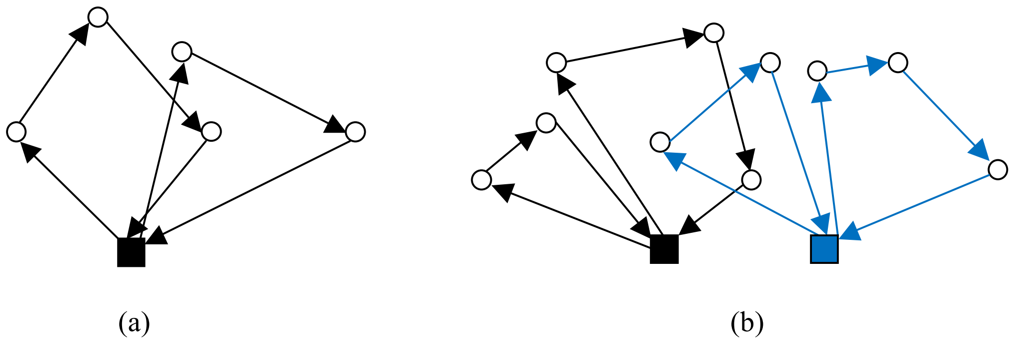

Figure 1: (a): mTSP (*n = 5, m = 2*) and (b): MmTSP(*n = 10, m = 4, d = 2,* $m_1 = 2$*,* $m_2 = 2$)

The principles of GAs are well known. Generally a GA is composed of three operators which are applied sequentially to the current population of chromosomes interactively generation after generation. These three operators are reproduction, crossover and mutation. A population of solutions is maintained and a reproductive process allows parent solutions to be selected from the population. Offspring solutions are produced which exhibit some of the characteristics of each parent. The fitness of each solution can be related to the objective function value, in this case the total distance travelled by each salesman. Analogous to biological processes, offspring with relatively good fitness levels are more likely to survive and reproduce, with the expectation that fitness levels throughout the population will improve as it evolves. In this paper we propose a new chromosome design include two distinct part, then utilize and compare four type of common permutation based crossovers for first part of it as follows: ordered crossover (OX), partially matched crossover (PMX), union crossover#2 (UX2) and proposed modified moon crossover (new MX).

This paper is organized as follows. The problem definition and formulation is described in Section 3. The instruction of genetic algorithms is described in Section 4 followed by computational results in Section 5 and by the conclusion in Section 6.

2 Fixed destination MmTSP

2.1 Problem definition and notation

Consider a complete directed graph $G = (V, A)$, where V is the set of n nodes (vertices), A is the set of arcs and $C = (c_{ij})$ is the cost (distance) matrix associated with each arc $(i, j) \in A$. The cost matrix C can be symmetric, asymmetric or Euclidean. Now let the node set be partitioned such that $V = V' \cup D$, where the first *d* nodes of V are depot set *D*. there are m_i salesmen located at depot *i* initially and the total number of salesmen is *m*. Also let $V' = \{d+1, d+2, \ldots, n\}$ be the set of customer nodes.

The MmTSP consists of finding tours for all the salesmen such that all customers are visited exactly once, the number of customers visited by a salesman lies between a interval and predetermined salesmen (m_k) depart from each depot and the total cost of all the tours is minimized.

The formulation for the fixed destination MmTSP presented in [6] is adopted here. The following binary variable is defined:

$$x_{ijk} = \begin{cases} 1, & \text{if the salesman departured from the kth node, passes the ij link} \\ 0, & \text{other wise} \end{cases}$$

Notice that x_{iik} is equal to zero.

For any traveler, u_i is the number of nodes visited on the traveler's path from the origin up to node i (i.e., the visit number of the ith node). L is the maximum number of nodes a salesman may visit; thus, $1 \le u_i \le L$ for all $i \ge 2$. In addition, let K be the minimum number of nodes a salesman must visit, i.e., if $x_{ikk} = 1$, then $K \le u_i \le L$ must be satisfied.

2.2 Formulation

The related model for the fixed destination MmTSP as follows:

Minimize $$\sum_{k\in D}\sum_{j\in V'}\left(c_{kj}x_{kjk} + c_{jk}x_{jkk}\right) + \sum_{k\in D}\sum_{i\in V'}\sum_{j\in V'} c_{ij}x_{ijk}$$

s.t. $$\sum_{j\in V'} x_{kjk} = m_k, \quad k\in D, \tag{1}$$

$$\sum_{k\in D} x_{kjk} + \sum_{k\in D}\sum_{i\in V'} x_{ijk} = 1, \quad j\in V', \tag{2}$$

$$x_{kjk} + \sum_{i\in V'} x_{ijk} - x_{jkk} - \sum_{i\in V'} x_{jik} = 0, \quad k\in D, \quad j\in V', \tag{3}$$

$$\sum_{j\in V'} x_{kjk} - \sum_{j\in V'} x_{jkk} = 0, \quad k\in D, \tag{4}$$

$$u_i + (L-2)\sum_{k\in D} x_{kik} - \sum_{k\in D} x_{ikk} \le L-1, \quad i\in V' \tag{5}$$

$$u_i + \sum_{k\in D} x_{kik} + (2-K)\sum_{k\in D} x_{ikk} \ge 2, \quad i\in V' \tag{6}$$

$$\sum_{k\in D} x_{kik} + \sum_{k\in D} x_{ikk} \le 1, \quad i\in V', \tag{7}$$

$$u_i - u_j + L\sum_{k\in D} x_{ijk} + (L-2)\sum_{k\in D} x_{jik} \le L-1, \quad i \ne j, \quad i,j\in V', \tag{8}$$

$$x_{ijk} \in \{0,1\}, \quad i,j\in V, \quad k\in D. \tag{9}$$

This formulation is valid when $2 \leq K \leq \lfloor (n-1)/m \rfloor$ and $L \geq K$. Thus one salesman cannot depart from a depot and visit only one city and return to his depot. So the number of cities each salseman must visit is at least 2 without his depot. Constraints (1) ensure that exactly m_k salesmen depart from each depot $k \in D$. Constraints (2) ensure that each customer is visited exactly once. Route continuity for customer nodes and depot nodes is represented respectively by constraints (3) and (4). Constraints (5) and (6) impose upper and lower limits to the tours, respectively. In addition, if i is the first node on a tour, these constraints oblige u_i to be equal to 1. Tours with just one customer node are prohibited with constraints (7). Finally, constraint (8) is subtour elimination constraints (SECs) in that they break all subtours between customer nodes. Subtours are closed tours made of customer nodes and with no depots as the starting or ending point which might be found in the solutions if there are no SECs considered in the model.

Observe that this problem belongs to the class of NP-hard problem. There are $O(dn^2)$ binary variables and $O(n^2)$ constraints in this formulation.

3 GA issues

3.1 Chromosome representation

A well-designed chromosome should reduce or eliminate redundancy, accurately represent a solution to the problem, and allow the GA operators to work effectively to generate better solutions as the iterative evolutionary process continues.

Fig. 2 illustrates a new chromosome design for the MmTSP (with $n = 12$, $m = 4$, $d = 2$, $m_1 = 2$, $m_2 = 2$) that has two distinct parts. The idea of using this chromosome for the MmTSP is similar to the two-part chromosome of the MTSP's GA in [7].

The first part of the chromosome is a permutation of integers from 1 to n, representing the n cities. The second part of the chromosome is of length m and represents the number of cities assigned to each of the m salesmen. The values assigned to the second part of the chromosome are constrained to be m positive integers at least 2 that must sum to the number of cities to be visited (n). Also the second part demonstrates the number of salesmen who respectively depart from each depot (i.e. the first m_1 salesmen referred to first depot and so on).

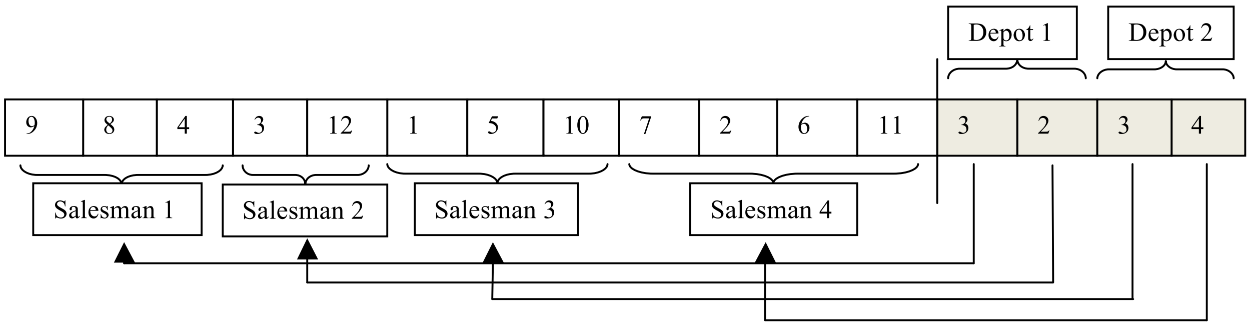

Figure 2: Example of the chromosome representation for a 12 city MmTSP with four salesmen and two depots.

In Fig. 2, salesman 1 departs from depot 1, visits cities 9, 8 and 4 (in that order) and returns to depot 1, salesman 2 also departs from depot 1 visits cities 3 and 12 (in that order) and returns to depot 1, salesman 3 departs from depot 2 visits cities 1, 5 and 10 (in that order) and returns to depot 2 and so on for salesmen 4. Using this new two-part chromosome method reduces the size of the search space due to the elimination of some (but not all) redundant solutions. Using this chromosome for the MmTSP, there are $n!$ possible permutations for the first part of the chromosome. The second part of the chromosome represents a positive vector of integers (x_1 , x_2 , …, x_m) that must equal or greater than 2 and sum to n. There are $\binom{n-m-1}{n}$ distinct positive integer-valued m-vectors that satisfy this requirement [8]. Thus, the solution space for the two-part chromosome is of size $n!\binom{n-m-1}{n}$.

3.2 Crossover

For first part of this chromosome we use four type of crossover that commonly used for TSP-type of problems in the previous researches. These crossover methods are ordered crossover (OX), partially matched crossover (PMX), union crossover#2 (UX2) and proposed modified moon crossover (new MX). A new crossover operator we are proposing is new MX. The proposed crossover is called the moon crossover because it is very similar to the change of the moon.

In this crossover first we choose an arbitrary salesman from second part of first parent (i.e. a number between 1 and the number of salesmen), for example i, then copy the first j elements from first parent to the offspring preserving the order, that j is the sum of first i elements of second part of the first parent. Finally choose the remaining values that not exist in the offspring, from the second parent and copy them to the child, preserving the order. An example of the moon crossover is illustrated in Fig. 3.

Suppose that there is two chromosomes (parents) with 12 cities, 4 salesmen, 2 depots and 2 salesmen in each depot (i.e. $n = 10, m = 4, d = 2, m_1 = 2, m_2 = 2$) that we show them in Fig. 3 as parent1 = [3 7 2 11 9 5 1 12 4 8 10 6 3, 3 4 3 2] and parent2 = [10 6 5 2 12 11 4 9 1 7 3 8, 3 2 5 2]. First, select a number between 1 and m - 1 from parent1 at random and note it i. In this example $i = 2$. i is the number of first tours (or salesmen) that remain without change from parent1 in offspring. So copy the 2 first tours to offspring without change (i.e. copy the first 7 number from parent1 to offspring). Cities 3, 7, 2, 11, 9, 5, 1 have already appeared in the offspring, so we cannot add these cities into the offspring. Finally choose the remaining values that not exist in the offspring, from parent2 and copy them to the child, preserving the order, i.e. copy 10, 6, 12, 4, 8 respectively to offspring.

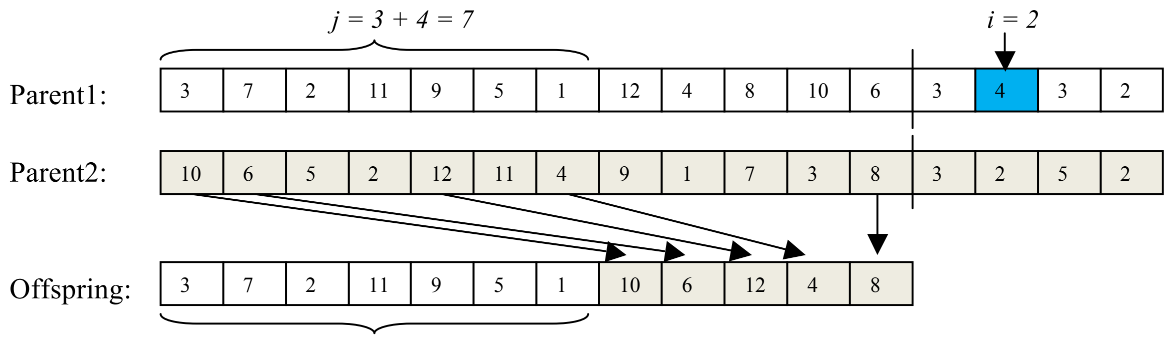

Figure 3: new MX

The second part of the two-part chromosome used single point asexual crossover method [9]. This method simply cut the second part of the chromosome into two sections and reversed the order in which the two pieces were arranged. This type of crossover ensures that the second part of the chromosome remained feasible (with the sum of the values in the chromosome equaling *n*).

3.3 Mutation

We use the operators in 1-bit and 2-opt methods randomly as the mutation with all crossover methods in the GA's generations. The performance of operators in 1-bit and 2-opt methods over tours is shown in Fig. 4. In 1-bit operator, two routes of two salesmen are randomly chosen from one parent and then a node is deleted from a route and is added to the other route. In 2-opt operator, two routes belonging to two salesmen's tours are randomly chosen from the existing from the parent and then two nodes out of the two routes are exchanged with each other. Fig. 4 depicts a profile of the two genetic mutation operators used in all solving problems for extraction the better solutions from the feasible solution space.

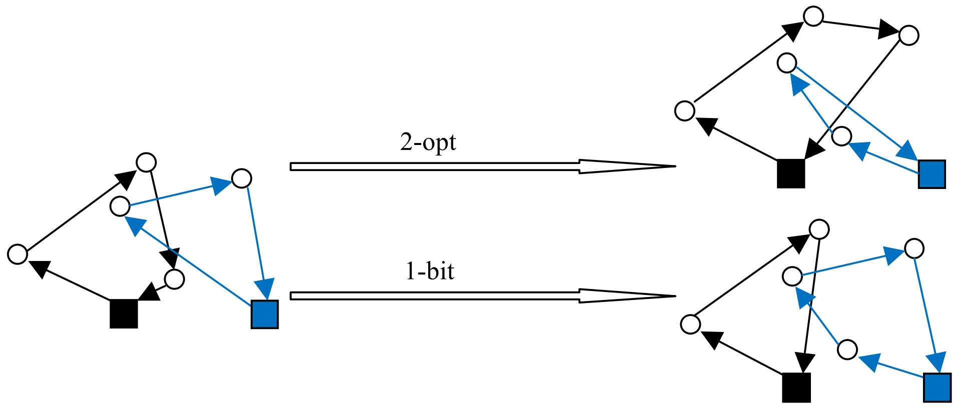

Figure 4: 1-bit and 2-opt

3.4 Initial population

Chromosomes in initial population are generated randomly with this condition that there are no similar chromosomes in the initial population pool. Because of the chromosome has two parts we use two different ways to produce each part as follow: The first part of each chromosome is a randomly generated permutation of the *n* cities. In the second part of the chromosome the sum of the *m* positive integers must be equal *n*. *m* gene values (x_i) for the second part of the chromosome are generated as a discrete uniform random number between 1 and $n - \sum_{j=1}^{i-1} x_j$, for $i = 1$ to *m* (i.e., the maximum value of each successive gene value was based on *n* and the sum of the previous values). These values were then randomly assigned to the genes in the second part of the chromosome.

3.5 Choosing parents' strategies

We use three strategies for choosing parents in each GA's generations. These strategies are elitism, roulette wheel [10] and binary tournament. In elitism strategy the chromosomes with better fitness than others in preceding generation are chosen as parents in the current generation. In roulette wheel strategy all chromosomes have chance to choose as parent, considering their fitness functions (i.e. a chromosome with lower fitness has more chance to choose). In binary tournament two chromosomes are chosen from the population at random. The one with the better fitness value is chosen as a parent. The process is repeated to obtain a second parent with this condition that there is no similar chromosomes in the parents' population pool.

4 Experimental results

In this section, we describe our experience with some computational experiments done using the genetic algorithms described in previous section. In the two following subsections, computational testing methodology and computational results on the fixed destination MmTSP are presented, respectively.

4.1 Computational testing methodology

In order to evaluate the practical benefits of the genetic algorithms described in previous sections, computational tests were conducted to compare the performance of all four crossover techniques and three choosing parents' strategies on a set of problems created for the fixed destination MmTSP.

Table 1 summarizes the experimental conditions of 14 different problem size (*n*), salesman (*m*) and depots (*d*) combinations for each type of problem. For each level of *n*, all salesmen started and ended their individual tours in the one arbitrary depot. In any type of problem the number of salesmen in each depot is the same and equal to the number of salesmen (*m*) divided by number of depots (*d*) (i.e. m/d).The fitness function minimizes the total distance traveled by all of the salesmen. For these runs, each salesman was required to visit at least two cities (other than the home city). This objective would represent a situation where there are a set number of salesmen and there are no constraints associated with the maximum number of cities visited by any one salesman. So any one salesman can visit at most $2(m-1)$ cities.

Table 1
Computational test conditions

Number of cities (n)	Number of salesmen (m)	Number of depots (d)	Number of salesmen in each depot (m_k)
10*	2	2	1
20*	2	2	1
20*	4	2	2
30*	2	2	1
30*	4	2	2
50	2	2	1
50	4	2	2
50	8	4	2
100	4	2	2
100	8	4	2
100	12	4	3
100	16	4	4
150	8	4	2
150	12	4	3
150	16	4	4
150	20	5	4

*Test problems compared with Lingo 8.0

In the next subsection we have done two comparison ways to examine the accuracy and efficiency of the algorithm. First we compare the best result obtained by the GAs and optimal solution obtained by Lingo 8.0 with small dimensions. This problems were assigned by (*) in table 1. Second comparisons are between the efficiency of four type crossovers together and also three strategies of choosing parents described in previous sections.

Each of 14 number of cities (n), salesmen (m) and depots (d) combinations were run 30 times. The tests were run on a Pentium 4 CPU 2.8 GHz 512 MB of Ram using programs written in Delphi 7.

4.2 Computational results

To the best knowledge of the authors, there were no standard test problems for the MmTSP on the accessible libraries. Therefore we produce the two dimensional cost (distance) matrix $C_{(d+n)*(d+n)}$ between all cities (depots and cities) that c_{ij} s are random integral numbers generated uniformly in the span $10 \leq c_{ij} < 100$. The cost matrix for all problems with same number of cities (n) and depots (d) is constant.

The GA programs all utilized a population size of 100, an 80% crossover, a 15% mutation and a 5% reproduction probability. When the GA created a new child chromosome using these operators, the first part of the new child chromosome was matched with each of the second parts from the two parents as well as with the newly created second part. If any new child solution of these produced offspring has better fitness than his parent(s), then the new child inserted in the current new population. The algorithm stops when the diversity of chromosomes in last generation be smaller than 0.1 (i.e. $\delta_{LastGeneration'sChromosomes} < 10$)

When minimizing the total distance of all the salesmen in an MmTSP, as the number of salesmen in each depot increases, the total distance of all of the trips also tends to increase. This results from the

fact that each salesman must start and return to the home city. So as the number of salesmen increases, the number of trips into and out of the home city increases, thereby increasing the total distance traveled.
The comparison between results for the runs using the total distance fitness function with the GA and optimal solution obtained by Lingo 8 are presented in table 2. The last column presents the GA percentage of the optimal solution. These are calculated as:

$$GA/LINGO = (1 - \frac{GA(fitness) - LINGO(fitness)}{LINGO(fitness)}) * 100$$

Table 2
Comparison between the GA result and optimal solution by LINGO 8

n	*m*	*d*	m_k	optimal solution		GA solution		GA / LINGO %
				fitness	time (s)	fitness	time (s)	
10	2	2	1	268	1	271	10	98.88
20	2	2	1	335	109	342	25	97.91
20	4	2	2	383	8	393	31	97.39
30	2	2	1	458	64	507	40	89.30
30	4	2	2	503	16	561	42	88.47

Table2 shows that in small problem size with the GA we can obtain answers very close to the optimal solution in a reasonable time. This closeness decreases with the size of problems increases.

Tables 3-5 present the comparison results between four type of crossovers and three strategies of choosing parents in the GA.

Table 3
Comparison between four type of crossovers result with elitism strategy of choosing parents

n	*m*	*d*	m_k	UX#2		PMX		OX		new MX	
				fitness	time (s)	fitness	time (s)	fitness	time (s)	fitness	time (s)
50	2	2	1	732	163	823	157	832	190	839	112
50	4	2	2	755	186	861	136	840	162	854	127
50	8	4	2	791	199	886	207	884	217	898	129
100	4	2	2	1411	493	1415	383	1482	371	1229	272
100	8	4	2	1401	497	1412	360	1519	363	1362	324
100	12	4	3	1425	489	1431	371	1496	372	1424	349
100	16	4	4	1471	519	1467	402	1490	398	1436	343
150	8	4	2	1936	739	1954	679	2016	712	2040	726
150	12	4	3	2017	773	2030	669	2053	767	2102	751
150	16	4	4	2092	841	2127	683	2212	803	2239	729
150	20	5	4	2145	847	2196	749	2151	814	2271	801

Table 4
Comparison between four type of crossovers result with roulette wheel strategy of choosing parents

n	*m*	*d*	m_k	UX#2		PMX		OX		new MX	
				fitness	time (s)	fitness	time (s)	fitness	time (s)	fitness	time (s)
50	2	2	1	745	181	849	163	864	212	872	129
50	4	2	2	781	193	821	162	875	166	871	141
50	8	4	2	794	207	823	224	873	214	883	149
100	4	2	2	1441	502	1421	363	1480	396	1246	251
100	8	4	2	1445	494	1454	372	1524	390	1365	321
100	12	4	3	1431	498	1429	384	1488	329	1467	311
100	16	4	4	1496	524	1473	417	1482	401	1439	354
150	8	4	2	1958	747	1936	684	2062	719	2082	694
150	12	4	3	2029	787	2051	683	2074	736	2134	712
150	16	4	4	2112	875	2138	692	2226	819	2268	742
150	20	5	4	2249	881	2339	842	2383	841	2246	853

Table 5
Comparison between four type of crossovers result with binary tournament strategy of choosing parents

n	*m*	*d*	m_k	UX#2		PMX		OX		new MX	
				fitness	time (s)	fitness	time (s)	fitness	time (s)	fitness	time (s)
50	2	2	1	784	211	812	153	845	196	831	91
50	4	2	2	804	232	824	123	830	140	846	97
50	8	4	2	854	241	871	202	880	215	892	100
100	4	2	2	1401	483	1407	370	1486	366	1231	296
100	8	4	2	1423	499	1429	352	1516	368	1351	312
100	12	4	3	1417	483	1427	370	1486	366	1412	333
100	16	4	4	1472	512	1463	398	1476	392	1433	335
150	8	4	2	2089	738	1914	680	2012	702	2025	665
150	12	4	3	2105	751	2018	659	2048	725	2096	683
150	16	4	4	2245	893	2112	689	2216	801	2234	718
150	20	5	4	2305	745	2145	748	2048	812	2259	732

Tables 3-5 show that in problems with 50 cities, combination of elitism choosing parents strategy with UX#2 has a better fitness but worse time than other combinations. Problems with 100 cities the binary tournament strategy with newMX works better than other in fitness and time. In big problem size with 150 cities binary tournament and PMX has a better fitness and also has average time versus other combinations.

Tables 3-5 also show that generally the newMX has shorter time than others, UX2 has better fitness than others and PMX has both average time and fitness versus others.

5 Conclusion

In this paper we present a genetic algorithm for the fixed destination multi depot multiple traveling salesmen problem. Modeling the MmTSP using the new two-part chromosome proposed in this paper has small solution space that allows it to produce solutions with better fitness values in most cases tested in this research (and all of the most difficult cases). Compared to Lingo 8.0 the experimental results demonstrate the efficiency of the algorithm which obtains answers very close to the optimal solution in small dimensions related to the exact approach used in Lingo 8.0. Also in problem with big dimensions that exact approaches (such that Lingo) are not applicable we compare four types of crossover and three types of parent selection strategies and obtain reasonable and predicable results. We addition to the MmTSP presented here; there are other kinds of the MmTSP with very few meta-heuristics designed that can be investigated.

6 References

[1] S. Gorenstein, Printing press scheduling for multi-edition periodicals, *Management Science* 16 (6) (1970) B373–B383.

[2] J.K. Lenstra, A.H.G. Rinnooy Kan, Some simple applications of the traveling salesman problem, *Operations Research Quarterly* 26 (1975) 717–733.

[3] B. Brummit, A. Stentz, Dynamic mission planning for multiple mobile robots, in: *Proceedings of the IEEE International Conference on Robotics and Automation*, April 1996.

[4] L. Tang, J. Liu, A. Rong, Z. Yang, A multiple traveling salesman problem model for hot rolling scheduling in Shangai Baoshan Iron & Steel Complex, *European Journal of Operational Research* 124 (2000) 267–282.

[5] K.C. Gilbert, R.B. Hofstra, A new multiperiod multiple traveling salesman problem with heuristic and application to a scheduling problem, *Decision Sciences* 23 (1992) 250–259.

[6] I. Kara and T. Bektas: Integer Linear Programming Formulations of Multiple Salesmen Problems and its Variations, *European Journal of Operational Research* 174 (2006) 1449–1458.

[7] Arthur E. Carter, Cliff T. Ragsdale, A new approach to solving the multiple traveling salesperson problem using genetic algorithms, *European Journal of Operational Research* 175 (2006) 246–257

[8] Ross, S., 1984. Introduction to Probability Models. *Macmillian*, New York, NY.

[9] Chatterjee, S., Carrera, C., Lynch, L., 1996. Genetic algorithms and traveling salesman problems. *European Journal of Operational Research* 93 (3), 490–510.

[10] Goldberg, D.E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. *Addison-Wesley*, Reading, MA.