International Journal of Engineering, Mathematical and Physical Sciences
ISSN: 2517-9934
Vol:2, No:2, 2008

Enhanced Traveling Salesman Problem Solving by Genetic Algorithm Technique (TSPGA)

Buthainah Fahran Al-Dulaimi, and Hamza A. Ali

***Abstract*—**The well known NP-complete problem of the Traveling Salesman Problem (TSP) is coded in genetic form. A software system is proposed to determine the optimum route for a Traveling Salesman Problem using Genetic Algorithm technique. The system starts from a matrix of the calculated Euclidean distances between the cities to be visited by the traveling salesman and a randomly chosen city order as the initial population. Then new generations are then created repeatedly until the proper path is reached upon reaching a stopping criterion. This search is guided by a solution evaluation function.

Keywords—Genetic Algorithms, Traveling Salesman Problem Solving. Optimization.

I. Introduction

GENETIC ALGORITHMS (GA's) are relatively new paradigms in artificial intelligence which are based on the principles of natural selection. The formal theory was initially developed by John Holland and his students in the 1970's [1, 2]. The continuing improvement in the price/performance value of GA's has made them attractive for many types of problem solving optimization methods. In particular, genetic algorithms work very well on mixed (continuous and discrete) combinatorial problems. They are less susceptible to getting 'stuck' at local optima than gradient search methods. However, they tend to be computationally expensive and they are probabilistic algorithms. An initial population called genome (or chromosome) is randomly generated then successive populations, or generations, are derived by applying genetic operators such as selection, crossover and mutation to evolve the solutions in order to find the best one(s). Fig. 1 depicts an overview of a simple GA.

The selection operator chooses two members of the present generation in order to participate in the next operations: crossover and mutation. The crossover operator intermixes the alleles of the two parents to obtain an offspring. The mutation operator occurs a short period of time after crossover and as in nature it exchanges alleles randomly. The three most important aspects of using genetic algorithms are: (1) definition of the objective function, (2) definition and implementation of the genetic representation, and (3) definition and implementation of the genetic operators. Once these three have been defined, the generic genetic algorithm should work fairly well.

Buthainah Fahran Al-Dulaimi and Hamza A. Ali are with Software Engineering Department, Al-Isra Private University, Jordan (phone: +962 79 590 6054, e-mails: buthynna@yahoo.com, alsewadi@hotmail.com).

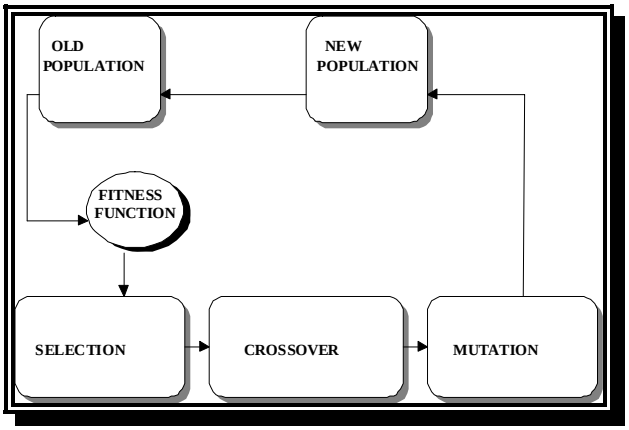


Fig. 1 Genetic Algorithm Cycle

II. Traveling Salesman Problem

The Traveling Salesman Problem (TSP) is one of the important subjects which have been widely addressed extensively by mathematicians and computer scientists. Its importance stems from the fact there is a plethora of fields in which it finds potential applications such as DNA fragment assembly and VLSI design. Formally, the TSP may be defined as follows [3]

It is a permutation problem with the objective of finding the path of the shortest length (or the minimum cost) on an undirected graph that represents cities or node to be visited. The traveling salesman starts at one node, visits all other nodes successively only one time each, and finally returns to the starting node. i.e., given n cities, named {c1, c2, ..., cn}, and permutations, σ1, ..., σn!, the objective is to choose σi such that the sum of all Euclidean distances between each node and its successor is minimized. The successor of the last node in the permutation is the first one. The Euclidean distance d, between any two cities with coordinate (x1, y1) and (x2, y2) is calculated by

$$d = \sqrt{(|x_1 - x_2)^2 + (|y_1 - y_2|)^2} \quad (1)$$

Recent Works: In 2004, the traveling salesman problem of visiting all 24,978 cities in Sweden was solved. A tour of length 855,597 kilometers was found and it was reported that no shorter tour exists. This is currently the largest solved TSP instance, surpassing the previous record of 15112 cities through out Germany set in 2001.

Literature reviews: The TSP is well-studied problem [4]. Many approaches have been tried. Among others are simulated annealing with local search heuristics [5], genetic algorithms [6, 7] and neural networks [8]. Historically, researchers have suggested a multitude of heuristic algorithms, including genetic algorithms (GA's) [9] for solving TSP [10]. Furthermore, So many research activities and achievements have reported recently of which are the followings: knowledge based multiple inversion and knowledge based neighborhood swapping GA algorithms [11], hybridized GA for integrated TSP and quadratic assignment problem [12], two-level GA for clustered TSP and large scale TSP [13], parallel GA program implementation on multicomputer cluster [14]. Moreover, Oracle data base was implemented on GA's for TSP [15]. Tracking moving targets in non-stationary TSP is also reported [16]. A random-key genetic algorithm for the generalized TSP [17] and an Immune Inspired-based Optimization Algorithm: Application to the Traveling Salesman Problem [18].

After this brief introduction and definition in section 1 and 2, section 3 outlines the proposed TSPGA system. Section 4 includes the implementation of the system, and then results and discussion were given in section 5. Finally section 6 concludes the paper.

III. The Proposed GA Scheme for TSA

The work in this paper aims to develop a computer software system for solving the Traveling Salesman Problem based on Genetic Algorithm (TSPGA). The block diagram of the proposed system is shown in Fig. 2. The number of cities involved in the search is entered and named as the letters {a, b, c, ….}, then Euclidean distances between the cities are calculated or entered. An initial solution is generated as a sequence of letters. Then a fitness function is used and two solutions are selected leading to the emergence of a new born solution which replaces one of the parent solutions. This process continues till final solution is achieved, i.e. the system finds the shortest path between n cities. Details of the system are shown in the following sections.

A. GA Components

The GA system whose flow diagram is given in Fig. 2 starts with randomly selected initial population. The successive generations are derived by applying the selection, crossover and mutation operators to the previous tour population. A current generation is acted upon by the three operators to produce the successive generation, i.e. selection, crossover and mutation operation. Before performing these operations, a fitness function evaluation is being implemented.

B. The Fitness Function

A fitness function evaluation is incorporated to assigns a value to each organism, noted as f_i. This f_i value is a figure of merit which is calculated by using any domain knowledge that applies. In principle, this is the only point in the algorithm that domain knowledge is necessary. Organisms are chosen using the fitness value as a guide, where those with higher fitness values are chosen more often. Selecting organisms based on fitness value is a major factor in the strength of GAs as search algorithms. The method employed here was to calculate the total Euclidean distance D_i for each organism first, then compute f_i by using the following equation

$$f_i = D_{\max} - D_i \tag{2}$$

where D_{max} is the longest Euclidean distance over organisms in the population.

Table I below illustrates an example for these fitness concepts for a population of 5 cities.

C. Selection Operations

The selection operator chooses two members of the present generation to participate in the next operations of crossover and mutation. There are two popular approaches for implementing selection, the first called roulette selection, is to

TABLE I
An Example for Path Fitness Function

i	Organism	D_i	f_i	P_i	S_i
1	BEFCAD	122	0	0.01	1
2	FEBACD	101	21	0.10	2
3	CDAFBE	80	42	0.19	2
4	DAFCBE	50	72	0.31	2
5	BDAEFC	30	92	0.40	3

assign a probability to each organism i, computed as the proportion using the following equation

$$P_i = f_i \div \sum f_j \tag{3}$$

(Where: j = 1, 2, …, n!).

A parent is randomly selected based on this probability P_i, see the example of Table I.

The second approach, called deterministic sampling, assigns to each organism i, a value S_i evaluated by the relation

$$S_i = ROUND\,(P_i * POPSIZE\,) + 1 \tag{4}$$

(Where: *ROUND* means rounding off to integer and *POPSIZE* means the population size).

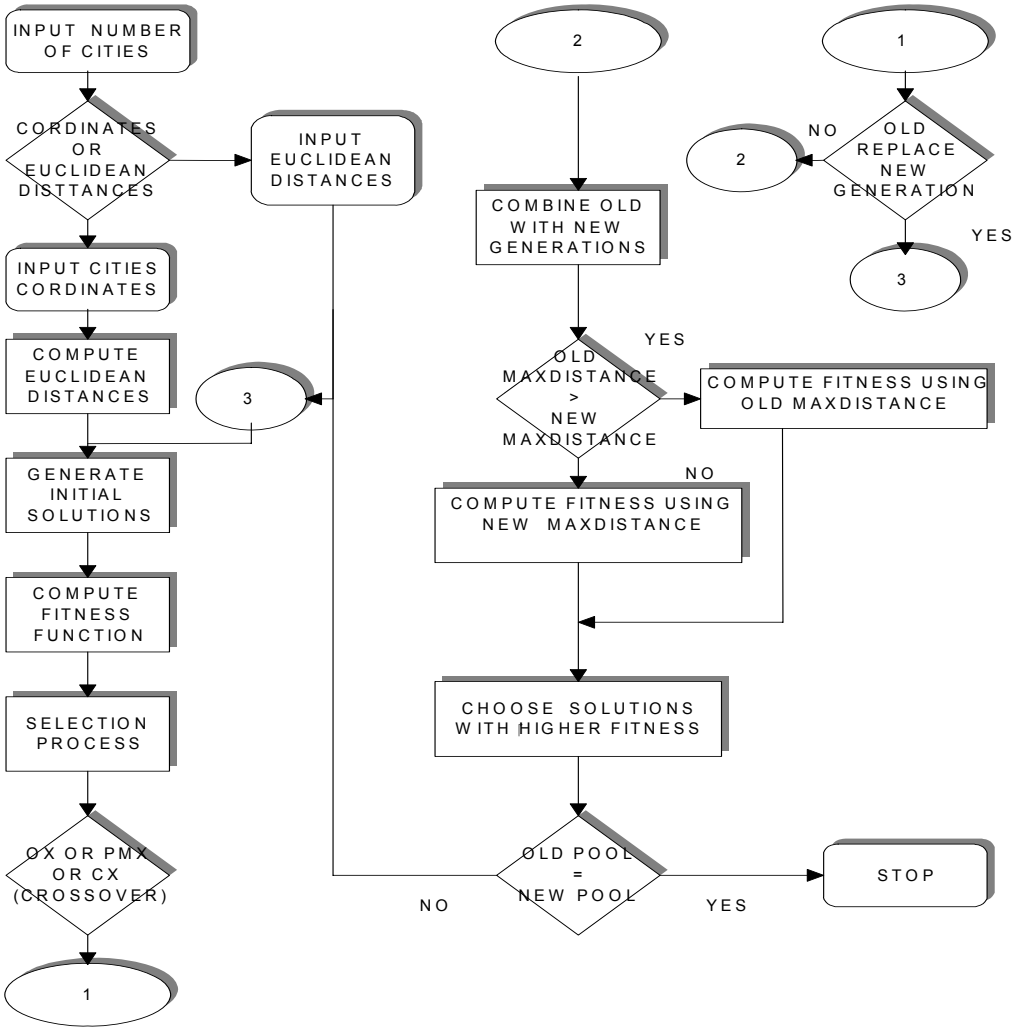


Fig. 2 The Flow Diagram of TSPGA System

The selection operator then assures that each organism participates as parent exactly S_i times [19] see the example of Table I.

The system performance using the above two methods was poor; in the sense that it takes long time to diverge. This result implied to adopt the following newly developed method; which is a combination of the above two methods with some modification, so that relevant organisms with higher fitness are selected and survived. Algorithm 1 outlines the steps of the proposed selection method.

Selection raises the issue of fitness function for TSP. A requirement of the above selection methods is that f_i be higher for organisms that represent better tours. However tours physically comprise lower Euclidean distances. Neither the longest nor shortest is known in advance.

D. Crossover and Mutation Operations

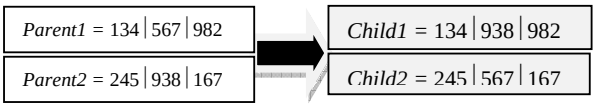
Problems such as those involving the optimization of a certain order of parameters, are naturally coded permutation organisms such as (i, b, g, ..., e). The use of crossover and mutation operators on these organisms is more subtle, as will be shown in the following:

Algorithm 1. The Selection Process
begin
for each organism compute:
Euclidean distance; determine maximum Euclidean distance; fitness function; fitness summation; roulette selection probability; deterministic sampling probability
endfor
sort population on fitness value in a descending order;
duplicate each organism exactly as deterministic sampling probability in mating pool;
counter=number of organisms in a mating pool;
parent1_index = random(counter – 1)
parent2_index = random(counter – 1)
/ on condition that not selected before /
start crossover process;
end

1. Specialized Crossover Operator

Two points crossover operator would have to be modified to work with such problems. Exchanging parts of two solutions will usually produce an invalid solution. Below an example that shows two parents of length 9. Select two random points in parents; say 3 and 6, then the alleles between those points are swapped to generate two children solutions. Both of the resulting children solutions are invalid [20].

Example

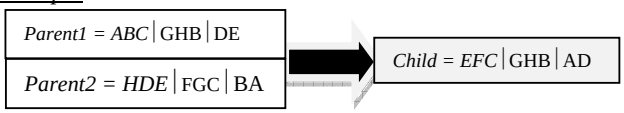


Three types of special crossover operators reported for permutation problems are selected to be examined and used in the proposed TSPGA system. They are: order crossover (OX), partially matched crossover (PMX), and cycle crossover (CX), as described briefly below.

♦ *Order Crossover (OX):*

To apply OX, two random cross points are selected. Alleles from parent1 that fall between the two cross points are copied into the same positions of the offspring. The remaining allele order is determined by parent2. Nonduplicative alleles are copied from parent2 to the offspring beginning at the position following the second cross point. Both the parent2 and the offspring are traversed circularly from that point. A copy of the parent′s next nonduplicative allele is placed in the next available child position [19]. An Example of OX is given below with two random cross points; 3 and 6, the alleles in the crossing sites from parent1 (GHB) are copied into the same positions of the child. The alleles after second cross point in parent2 (BA), B is skipped since it already exists in child, therefore only A is copied to child at position 7. Traverse parent2 circularly, the alleles (HDE), skip H to D which is copied to child at position 8. Traversed Child circularly, the alleles from parent2 (EF) are copied to child at positions 1 and 2. Finally, skip G to C and copy it to child at position 3.

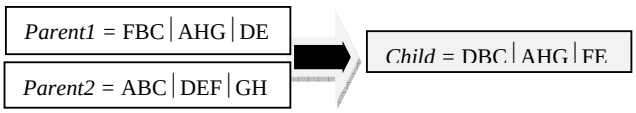
Example



♦ *Partially Matched Crossover:*

PMX proceeds just as OX. Alleles from parent1 that fall between two randomly selected crossing sites are copied into the same positions of the offspring. The remaining alleles positions are determined by parent2 during a two step process. First, alleles in parent2 not within crossing sites are copied to the corresponding positions within the offspring. Next each allele of parent2 within the crossing sites is placed in the offspring at the position occupied in parent2 by the allele from parent1 that displaced it [19], see the example below. The random crossing points are 3 and 6, the alleles in the crossing site of parent1 (AHG) displace in the child (DEF), and then the alleles (BC) in parent2 which are not in crossing site are copied into the same positions of the child. D goes to position 1 which is the position with respect to parent2, displacing allele A. Finally E and F are placed in the positions of H and G, respectively.

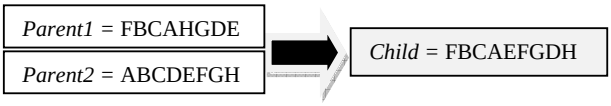
Example



♦ ***Cycle Crossover:***

CX performs recombination under the constraint that each allele value comes from one parent or the other [21]. CX does not use crossing sites, but a cycle is defined in a manner similar to an algebraic permutation group [19], see the example below, Comparing the strings of parent1 with parent2, F displaces A, A displaces D, D displaces G, and G displaces F. This forms the cycle FADG. The remaining alleles are filled from parent2 in the corresponding positions, BC, E, and H.

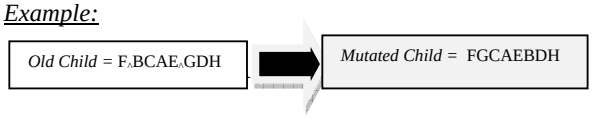
Example



2. Specialized Mutation Operator (SMX)

The mutation operator is a permutation problem arbitrarily changing single allele value that does not preserve allele uniqueness. The method frequently used for permutation problems is to interchange two randomly selected positions, thus preserving allele uniqueness [19]. The example below uses one of the previous child solutions. Let the crossing points be 2 and 6 selected randomly. The alleles B and G are swapped positions in the mutated child solution.

Example:



The system is designed to run in one of two options; default and customized as requested by the user. This is clearly shown in the flow diagram of Fig. 2.

A. Default System

The user enters the required data to produce solutions only, but he/she has neither influence nor choice to control the process. The selection of the crossover, mutation operations and mating pool strategies are chosen based on the experimental results, rather than user choice. It is executed according to the following steps.

1- The system prompts to enter number of cities.
2- The system prompts to enter coordinates of cities, then it computes Euclidean distances between cities or they may be entered directly depends on user desire.
3- After distance matrix becomes available, the system generates randomly initial solutions and computes longest Euclidean distance.
4- Finally the GA system continues processing till a solution is reached.

B. Customized System

This option operates the system according to user choices. The system prompts to enter the required data as in default option, then prepares distance matrix. The flow diagram of the TSPGA system is shown in Fig. 2. The user has the selection of the followings:

1- The version of crossover operator, i.e. OX, PMX, CX.
2- The strategy of maintaining the mating pool can be one of the following:
Strategy1: *the new generation replaces old generation.*
Strategy2: *the combination of old and new generations.*

Choosing *strategy1* of the above have given noisy results and took long implementation time. Therefore, *strategy2* was adopted, which determines the longest (maximum) Euclidean distance over old and new generations, then this maximum Euclidean distance is used to compute the fitness of both generations. The resulted solutions were sorted in descending order according to fitness value then the solution with higher fitness was chosen. Finally the GA system continues the cycle as shown in Fig. 2.

IV. Results and Discussion

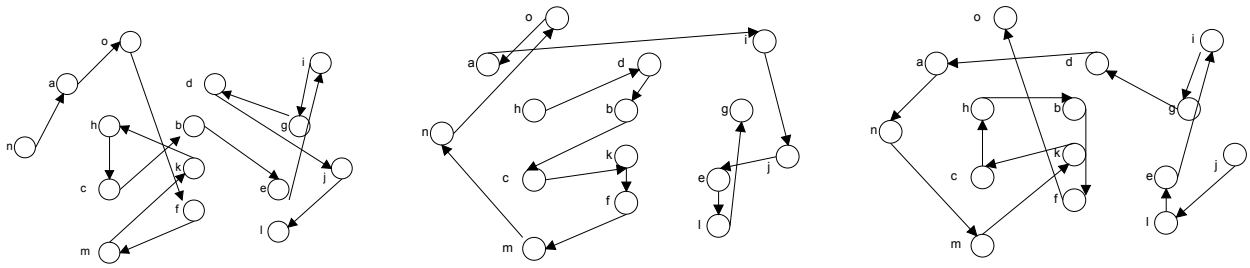
A. Experimental Results

The TSPGA system is tested for various parameter variations, such as crossover, mutation and mating pool. Variable setups of number of cities and distances combinations were considered as an example. The data for the system run for 15 cities is shown in Table II and the resulting best fit sketch using OX, PMX and CX crossover with 100% mutation is illustrated diagrammatically in Fig. 3.

TABLE II
The Data for TSPGA System for 15 Cities

Cities names	Coordinates
a, b, c, d, e, f, g, h, I, j, k, l, m, n, o	(110,54), (236,110), (153,151), (227,49), (307,176), (220,211), (341,90), (149,91), (335,40), (371,150), (218,161), (334,239), (148,227), (49,128), (183,39)

Table III includes experimental results giving the best fit tour Dmin for the developed TSPGA system runs for various operator selections for the purpose of comparison. For these variations strategy 2 mating pool that covers different values of crossover methods and mutations.



(a) Order Crossover, OX. (b) Partially Matched Crossover, PMX. (c) Cycle Crossover, CX.

Fig. 3 The Best Fit Tour for the TSPGA System of 15 Cities Using Strategy 2 and 100% Mutation

TABLE III
COMPARISON OF TSPGA SYSTEM OPERATOR VARIATIONS

Mutation	Order Crossover, OX		Partially Matched Crossover, PMX		Cycle Crossover, CX	
	Sequence	D_{min}	Sequence	D_{min}	Sequence	D_{min}
1 %	lgjibhkfmnaodce	1398	bcdjlegihkfmnoa	1486	ijlfmkcnhaodebg	1298
30 %	higjelmnkfcdoab	1465	hancejlkbgdiomf	1519	dielgjmncfkboah	1419
70 %	higjelmnkcadofb	1554	bcdoahmnikfeljg	1507	limfkhbdoancejg	1446
100 %	naofmkhcbeigdjl	1439	hdbckfmnoaijelg	1429	jleigdanmkchbfo	1412

B. Result Analysis

Intensive test of the proposed TSPGA system briefly shown in the above tables, the following remarks may be concluded:

1- The order crossover is very sensitive to mutation levels. The difference in performance is hint that there is a steady improvement in performance as mutation is decreased and this is in large and small populations. Performance improves slightly as population size increases.
2- The partially matched crossover is sensitive to the mutation level. The best mutation level is not as obvious as that with order crossover, it depends on the population size, the performance improved when population size increased.
3- The cycle crossover is less sensitive to mutation levels. It depends on population size; performance improves as population size decreased.
4- OX does better than the PMX, and better than the CX. The TSP is a problem where adjacency of particular elements (cities) is important. Thus we expect that the OX would perform best. The less sensitive behavior of the PMX and CX, especially the CX, can be thought of as a consequence of a built in higher level of mutation present in these operators.

V. CONCLUSION

The TSPGA system was developed to assist the user to obtain optimal tour to visit n cities. It has proposed the use of advanced operators of Genetic Algorithms in order to enhance the rate of divergence, and achieved tours with reasonable time. It has used an interactive user interface enabling users to handle most system features. The developed selection algorithm used helps to select a suitable strategy for selecting a pair of parents for crossover operation. Besides, newly developed fitness function is adopted and has produced reasonable results.

Furthermore, the use of man-machine interaction to guide the algorithmic procedure in a way different from that used in the proposed systems, as it allows the user to intervene at each step in the system executions to choose the relevant path or schedule from the current generation to produce a satisfactory results.

REFERENCES

[1] J. H. Holland et. Al., “Induction: Processes of Inference, Learning, and Discovery", MIT Press, 1989.
[2] M. Melanie, "An Introduction to Genetic Algorithms", MIT Press, 1996.
[3] S. Sur-Kolay, S. Banerjee, and C. A. Murthy, “Flavours of Traveling Salesman Problem in VLSI Design”, 1st Indian International Conference on Artificial Intelligence, 2003.
[4] E.L. Lawler, J.K. Lenstra, A.H.G Rinnooy Kan, and D. B. Shmoys, “The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization”, Wiley address Interscience, Chichester, 1985.
[5] Moscato, P. and M.G. Norman, “A `Memetic', Approach for the Traveling Salesman Problem. Implementation of a Computational Ecology for Combinatorial Optimization on Message-Passing a Systems, Parallel Computing and ransputer Applications”, edited by M. Valero, E. Onate, M. Jane, J.L. Larriba and B. Suarez, Ed. IOS Press, Amsterdam, 1992. pp: 187 –194.
[6] M. Lalena, “Traveling Salesman Problem Using Genetic Algorithms. http://www.lalena.com/ai/tsp/, 2003.
[7] D. Whitley and K. Mathias, “Genetic Operators, the Fitness Landscape and the Traveling Salesman Problem”, Parallel Problem Solving from Nature-PPSN 2. R. Manner and B. Mandrake North Holland-Elsevier, eds., 1992, pp: 219 – 228.
[8] S. Kedar, Naphade & Dilek Tuzun, “Initializing the Hopfield-Tank Network for the TSP using a convex hull: A Computational Study. Proceedings of the Artificial Neural Networks in Engineering (ANNIE'95) Conference, 1995, PP399 – 404.
[9] D. E. Goldberg, “Genetic Algorithm in Search, Optimization and Machine Learning”, Machine Learning. Addison-Wesley, New York, 1989.
[10] P. Larranaga, C. M. H. Kuijpers, R. H. Murga, I. Inza and S. Dizdarevic, “Genetic Algorithms for the Travelling Salesman Problem: A Review of Representations and Operators”, Artificial Intelligence Review. 13, 1999, PP129-170.
[11] S. R. Shubhra, B. Sanghamitra and K. Pal. Sankar, “New Operators of Genetic Algorithms for Traveling Salesman Problem”, IEEE, 0-7695-2128-2/04, 2004.
[12] Ji. Ping and Ho. William, “The Traveling Salesman and the Quadratic Assignment Proble: Integration, Modeling and Genetic Algorithm”, Int. Symposium on OR and its Applications, 2005, PP198-205.
[13] C. Ding, Ye. Cheng and M. He, “Two-Level Genetic Algorithm for Clustered Traveling Salesman Problem with Application in Large-Scale TSPs”, Tsinghua Science and Technology, Vol.12, No.4, 2007, PP459-465.
[14] P. Borovska, T. Ivanova and H. Salem, “Efficient Parallel Computation of the Traveling Salesman Problem on Multicomputer Platform”, Proceedings of the International Scientific Conference ‘Computer Science’ 2004, Sofia, Bulgaria, Dec. 2004, PP74-79.
[15] R. Gremlich, A. Hamfelt, H. de Pereda and V. Valkovsky, “Genetic Algorithms with Oracle for the Traveling Salesman Problem”, proceedings of world academy of science, Engineering and Technology, Vol. , Aug 2005, PP1307-6884.
[16] Q. Jiang, R. Sarker and H. Abbass, “Tracking moving targets in the non-stationary travelling salesman problem,” Complexity International, Vol.11, 2005, PP171-179.
[17] V. Lawrence, A. Snyder and M. S. Daskin, “A random-key genetic algorithm for the generalized traveling salesman problem”, Available online since 17 May 2005, www.sciencedirect.com

[18] M. Bakhouya and J. Gaber, “An Immune Inspired-based Optimization Algorithm: Application to the Traveling Salesman Problem, AMO”, Advanced Modeling and Optimization, Vol. 9, No. 1, 2007.
[19] B. P. Buckles, P. E. Petry and R. I. Kuester, “Schema Survival Rates and Heuristic Search in Genetic Algorithms”, IEEE, 1990, PP 86-91.
[20] D. T. Phillips, A. Rarindran and J. J. Solberg, “Operations Research: Principles and Practice”, John Wiely and Sons Inc, 1976.
[21] D. E. Goldberg, “Genetic Algorithms in Search, Optimization, and Machine Learning”, Addison Wesley Publishing Company Inc. 1989.

Buthainah Fahran Al-Dulaimi is an assistance professor at the Faculty of Science and Information Technology at Isra Private University (IPU), Jordan. She got her B.Sc.in 1981 from AL-Mustansryah University/Iraq, M.Sc. in 1999 from University of Baghdad/ Iraq and Ph.D in 2003 from Institute for Post Graduate Studies in Informatic/ Iraqi Commission for Computers and Informatic (ICCI)/Iraq. Before joining IPU, she worked in the National Computer Center (NCC)/ Iraq as chief programmer/analyzer, executive director of training management and teacher in the the same institute. Her research interests include Artificial Intelligence and Software Engineering (model designs).

Hamza Abbass Ali is an associate professor at the Faculty of Science and Information Technology at Isra Private University (IPU), Jordan. He got his B.Sc.in 1968 from Basrah University/Iraq, M.Sc. and Ph.D. in 1973 and 1977 respectively, from The University of London, UK. Before joining IPU, he worked as associate professor at Zarqa Private University (Jordan), visiting professor at University of Aizu (Japan) and associate professor at Basrah University (Iraq). His research interests include Cryptography, Information and Computer Network Security, Artificial Intelligence, Neural Networks and character recognition.