*Proceedings of the 2015 International Conference on Industrial Engineering and Operations Management*
*Dubai, United Arab Emirates (UAE), March 3 – 5, 2015*

# A Genetic and Iterative Local Search Algorithm for solving Subgraph Isomorphism Problem

Mina Mazraeh Farahani, Seyed Kamal Chaharsoughi*
Department of Industrial Engineering
Faculty of Engineering, Tarbiat Modares University
Tehran, Iran
mina.m.farahani@gmail.com, skch@modares.ac.ir

***Abstract*—Finding the subgraph of a graph that is isomorphic to a given graph has practicalapplications in several fields, from cheminformatics to image understanding. Since subgraph isomorphismproblem is NP-Hard, meta-heuristics are of especial use and importance in solving it. In this paper a hybrid meta-heuristic algorithm for subgraph isomorphism is proposed. The main idea of the new algorithm is to use the iterative local search (ILS) to improve the genetic operations and to better guide the search process in the genetic algorithm. The concept of the permutation matrix was used for the matching process. By applying permutation matrix, we could also define an efficient fitness function for the optimization version of the Subgraph Isomorphism problem which made possible the generation of instance problems of subgraph isomorphism for the first time. A landscape analysis was alsoconducted to justify the application of a population-based algorithm (namely GA) for this problem.**

***The results of the hybrid GA and ILS are computed for four groups of instance problems and are compared to the solutions of the genetic algorithm which show the improvement in performance of the GA, both in value and time.***

***Keywords: Subgraph Isomorphism Problem, Optimization, Meta-heuristics, GA, Iterative Local Search***

## I. INTRODUCTION

Graphs are one the most powerful tools in describing data structures useful in various fields of science and engineering. In this representation, nodes represent relations between primitives or structured objects and branches between nodes represent relations between them. One way to recognize the structure of an unknown pattern/object is to transform it into a graph, then match it with other graphs representing of prototype patterns.

This process of matching is called graph isomorphism. Formally, two graphs G and G'are said to be isomorphic (to each other) if there is a one-to-one correspondence between their vertices and between their edges such that incidence relationship is preserved [10].

If the isomorphism is encountered between a graph and a subgraph of another larger graph, then the problem is called subgraph isomorphism or graph monomorphism [13].

In theoretical computer science, the subgraph isomorphism problem is a computational task in which two graphs G and H are given as input, and one must determine whether G contains a subgraph that is isomorphic to H. Subgraph isomorphism is a generalization of both the maximum clique problem and the problem of testing whether a graph contains a Hamiltonian cycle, and is therefore NP-complete [2]. However certain other cases of subgraph isomorphism may be solved in polynomial time [3]. Graph isomorphism problem has a wide range of applications in pattern recognition, image processing, genetics, bioinformatics and scene analysis, which involves finding whether a given chemical compound is a subcompound of further specified compounds [12]. With the development of subgraph isomorphism, it is an important form of exact pattern recognition [3]. The algorithm has been also applied to integrated circuit testing [1], microprogrammed controller optimization [5], robot motion planning [7] and other problems.

## II. SUBGRAPH ISOMORPHISM

If a graph G consists of a finite nonempty set V of p elements that are called points, together with a set E of q distinct unordered pairs of distinct points that belong to V. A pair of points that belongs to E is a line. A subgraph of G is a graph whose points and lines all belong to G. A graph Gais isomorphism to a subgraph of a graph Gb if and only if there is a 1:1 correspondence between the point sets of this subgraph and of Gathat preserves adjacency. Figure 1 is a simple example of subgraph isomorphism. G1 is isomorphism to subgraph of G2. The correspondences between G1 and G2 are {a,b,c,d} in G1 to {2,1,3,4} in G2 or {a,b,c,d} in G1 to {2,5,4,6}. There are many mappings in G1and G2 for subgraph isomorphism. Therefore subgraph isomorphism will not be a unique correspondence.

Much research has been conducted on the algorithms to improve the efficiency of matching search and to reduce computational complexity. Refinement enumeration algorithm was introduced by [3]. Nonlinear optimization [11], deterministic matching [8] techniques are used in the recognition or evaluation of subgraph isomorphism. Also a genetic algorithm based solving method for assignment of RTDSs[1] was developed in [14].

[1]Real Time Digital Simulators

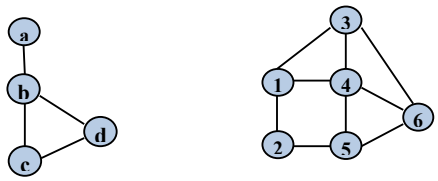


Fig. 1. Example of subgraph isomorphism

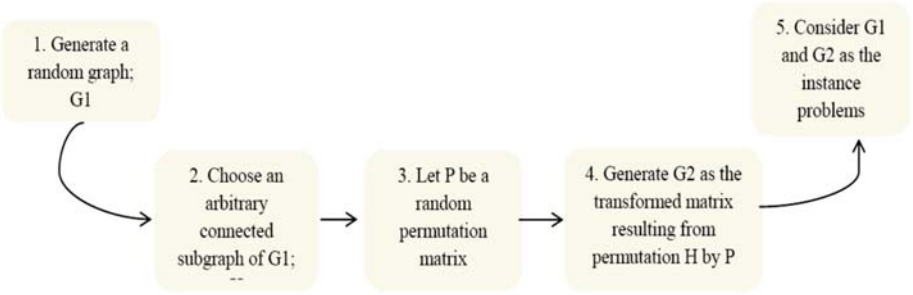


Fig. 2. The process of generating instance problems

## III. Fitness Landscape Analysis of SIP[2]

### A. Problem Modelling

- Matrix Representation and Encoding of SIP

Adjacency matrix is a very effective approach to describe a graph. In the matrix, the ones indicate there are connections between the nodes, while the zeros indicate there are no edges connecting two nodes. With the adjacency matrix, the graph can be operated mathematically. If H is a subgraph of G1, then the adjacency matrices of H and G1 satisfy:

$$M_H \subseteq M_{G1} \quad (1)$$

where $M_H$ and $M_{G1}$ are the adjacency matrices of H and G1 respectively. If G1 is isomorphism to G2, there must exist a P (permutation or transform matrix) satisfying the following equation:

$$P^* M_H * P^T = M_{G2} \quad (2)$$

P represents the rows and columns conversion of $M_H$. A permutation matrix is a square binary matrix that has exactly one entry 1 in each row and each column and 0s elsewhere. suppose that n is the number of nodes in subgraph G2, and a value i at position t=1,…,n indicating a node number, is an integer less than or equal to m, where m is the number of nodes in graph G1.Suppose that the problem is to find subgraph H of G1(with m nodes) which can be matched to G2 (with n nodes, such that m≥n). If H is the subgraph isomorph to G2, then it has an adjacency matrix of size n. If C is the matrix satisfying the following equation:

$$C = P^* M_H * P^T * M_{G2} \quad (3)$$

where P is the transform matrix, P will make the C a full ones matrix.

The objective function or the fitness function can be defined as the number of ones in the expended incidence matrix C. If C is a matrix of all ones, the objective function will be at a maximum, which is $n^2$. The objective function can also be defined as subtract of $n^2$ the number of the ones in the matrix C. The objective is minimal when zero. The fitness function is given as:

Fitness_Func=min($n^2$-number of ones in matrix(C))

- Generating instance problems

The SIP is at first the problem of deciding whether there exists a subgraph of G1 isomorphic to G2 and so it is a decision problem until we make sure of the existence of some subgraph of G2, e.g. H.

Because of the unavailability of the instance problems in the literature, we have generated some instances which can be regarded as optimization versions of SIP. The process was designed in a way which insures the existence of an isomorphic subgraph. This process is exhibited in Figure 2.

In this paper for the sake of generality, four classes of problems were generated for which the number of nodes belongs to the following uniform distributions (m is the number of nodes):

[2]Subgraph Isomorphism Problem

C1: class1: m~uni[7,12]
C2: class2: m~uni[13,18]
C3: class3: m~uni[19,24]
C4: class4: m~uni[25,30]

- Encoding

A solution for SIP is represented by a permutation vector without repetitions. Thus, P is encoded as a vector of length n(Fig.3) where n is the number of nodes in subgraph G2, and a value i at position t=1,…,n indicating a node number, is an integer less than or equal to m, where m is thenumber of nodes in graph G1.The problem is defined as the problem of finding a subgraph of G1 which can be matched to G2.


Fig. 3. Example of a permutation vector

### B. Fitness-Distance-Correlation

The Fitness-Distance-Correlation (FDC) [6] is a measure for problem difficulty for evolutionary algorithms. The FDC measures the correlation between the fitness differences of a solution and the global best solution and their distances in the search space. Thus, with a sample of solutions, the FDC coefficient δ is defined as

$$\delta = \frac{cov(f,d_{opt})}{\sigma(f)\sigma(d_{opt})} (5)$$

**Input:** initial solution x
**Output:** best solution found
**Initialize** with random permutation p, d=0
**Evaluate(x)**
**for** i := 1 to N do
N ← SelectNeighborhoodOperator(ρ)
x′← N(x)
Evaluate(x′)
**if** x′.fitness ≤ x.fitness**then**
x ← x′
d←d+1
**end if**
**i ← i + 1**
**end for**

Fig. 4. Outline of ILS

With$d_{opt}$being a solutions distance to the nearest optimal solution and *cov*(f, $d_{opt}$) the covariance of f and $d_{opt}$. A value of δ = 1 indicates a perfect correlation between fitness and distance to the optimum. The more both values are correlated, the easier the resulting problem is for selection based algorithms as a path of solutions with increasing fitness values leads to the optimum [9].

In order to analyse the landscape of the SIP, an iterative local search was conducted and the correlation of the fitness values with a distance measure of local solutions was calculated for the above mentioned four classes of instances. the distance measure which has been used is the hamming distance of two solutions. the iterative local search pseudo code is stated in Fig. 4.

### C. Results of Landscape analysis

We have determined 100 local optima for each class of instance problems, by applying the ILS algorithm using the respective 2-swap operator which starts from a random solution and stops if a local optimum is reached. The global optimum for each instance is known, which clearly equals 0, regarding the formulation of fitness function (equation 4).

Applying the hamming distance as the distance measure of solutions, FDC coefficients are calculated for the distance to the local optimum. Table I presents the respective results. the scatter plot of the change in fitness values in terms of the walk length (distances) to local optimum of every instance for two special case of the graphs with m~uni[13,18] and m~uni[19,25] number of nodes is exhibited in *Fig. 5*.

## IV. Genethic Algorithm and ILS for SIP

### A. Solution encoding and genetic operators

SIP faces the challenge of a tremendously expanded search space when there is an increase in the number of nodes in a graph. Though this can be mitigated by increasing the population size to provide a larger variety of building blocks for the search, this increase is limited, however, by constraints in computational resources. To address this problem, it should be noted that previous work has shown that genetic algorithm (GA) could be rather efficient for heuristic searches in a large search space with a relatively small population.

**Table I. Fitness-Distance-Correlation**

| Problem Class (number of nodes) | Correlation Coefficient (δ) | Average of Fitness Values |
|---|---|---|
| Uni[7,12] | 0.51 | 32.17 |
| Uni[13,19] | 0.31 | 128.14 |
| Uni[20,25] | 0.2 | 328.82 |
| Uni[25,20] | 0.33 | 499.67 |

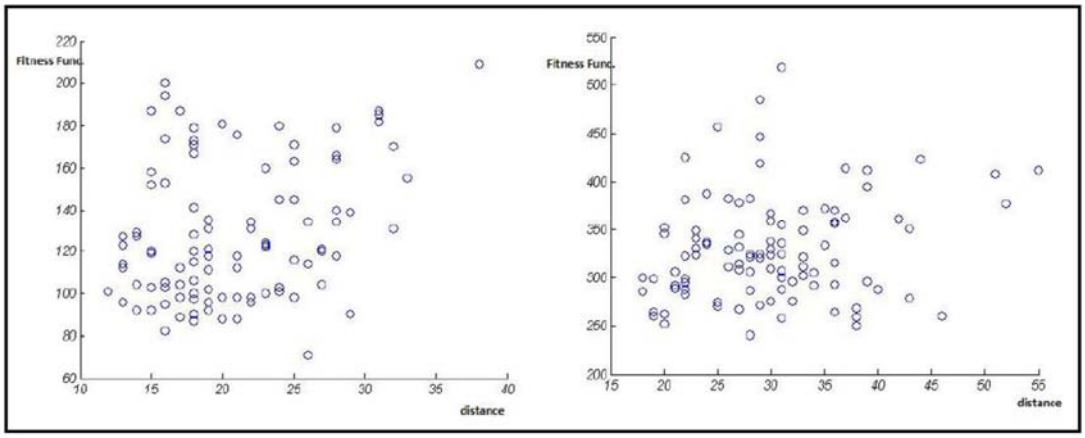


Fig. 5. Fitness Values vs. solution distances to local optimum

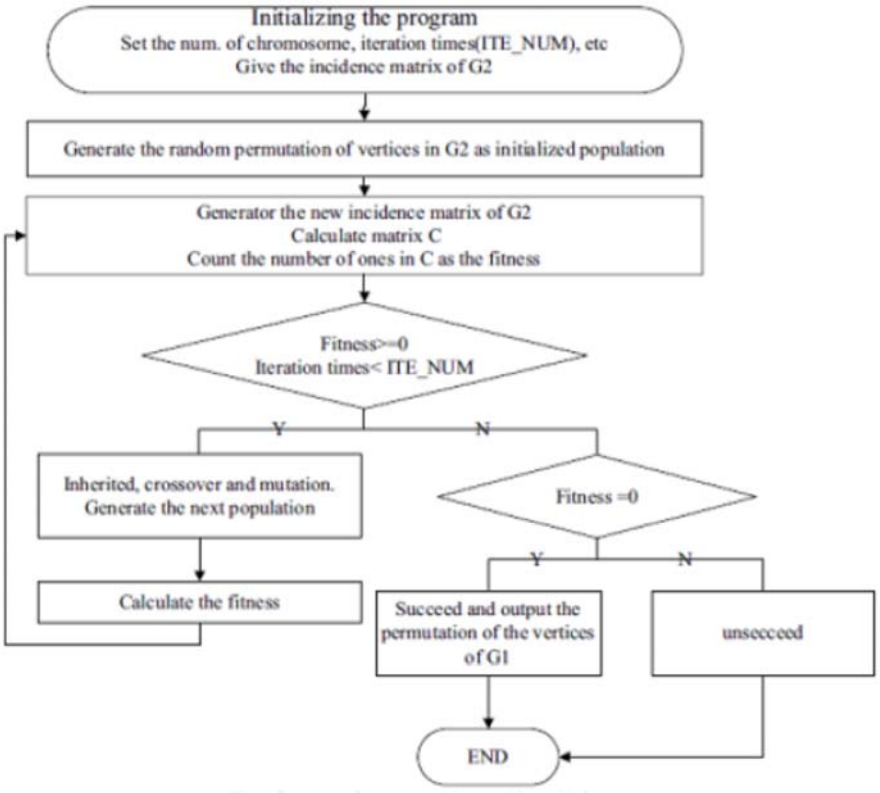


Fig. 6. Outline of Genetic Algorithms

- Encoding

The coding of GA can use the modulo-N vector {v1, v2,…,vn}. Vi is the number of vertices in G2=(V2,E2) where |V2|=n, 1≤vi≤m and G2 is a random subgraph generated by applying the procedure described in Fig.2. The vector is taken as an individual in a population. Every individual represent a subgraph of G1, where G1=(V1, E1), |V1|=m. The fitness functions of individuals can be calculated. By applying selection, crossover and mutation operators, the next population can be generated. The different patterns of subgraph are being searched. If the object function reaches minimal which is zero, the match pattern is found, otherwise the search will continue to the next iteration. Finally an

iterative local search was conducted to intensify the search procedure. The flowchart of GA is given in the *Fig. 6.*

- Selection

During the operations of a GA process, a new generation is born by selecting some particular parents and some children. The problem is how to select good chromosomes from the population. There are many existing methods in the literature, for instance roulette wheel selection, Boltzmann selection, rank selection and some others. While we have used the sequential parents selection procedure, we have maintained the good solutions of the population by an elitism approach. In every generation the first and second chromosomes with minimum fitness values will be preserved to be transferred to the next generation.

- Cross over

After deciding the encoding representation of a solution, two parents are selected according to the selection method,to create a new offspring. The crossover operation tries to swap parts of two parents in the population to generate new offspring. The crossover is made in hope that an offspring will inherit good parts of old chromosome. There are many ways to do crossover, more the crossover is specific to the problem more the GA is performant. Our crossover operator is applied on the permutation vector P.

We use a two point crossover by randomly choosing two crossover points *cp* and *dp* uniformly selected from 1 to m. An offspring is then obtained by appending the beginning (resp. the end) of the first parent to the end (resp. the beginning) of the second parent. The same procedure will be done for generating the second child, by switching first and second parents. This will maintain the population size constant. For clarity let us consider an example with n=5 , m=8 and crossover points *cp*=2 and *dp*=4 as depicted in *Fig. 7.*

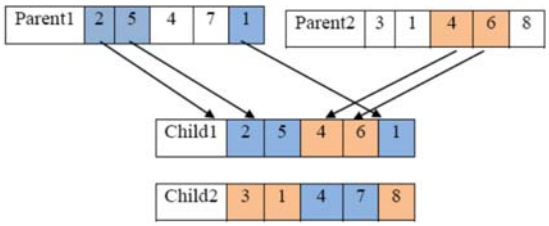


Fig. 7. Crossover operator for GA

- Mutation operators

When designing a GA, mutation appears to be among the most important operations to escape local optima since it preserves the diversification in the population. The mutation step takes place immediately after the crossover is performed. In our approach, we randomly modify the permutation vector by swapping a vertex to another one belonging to the set of potential vertices not have been chosen so far. This gives rise to a possibility of considering new subgraphs of G1, allows us to explore new solutions and thus to diversify the search.

- Fitness Function

Comparing the quality of two individuals plays an important role in any GA. In our work, the fitness function has been defined as previously stated in equation (4).

### *B. The ILS method*

This section describes the ILS heuristic we have developed for our hybrid LRP approach. As stated previously, the ILS is used to mainly improve the genetic operations and to better guide the search process. The main steps of an ILS, namely, local search and perturbation were first given in [4]. The general framework of an ILS is depicted in *Fig. 8* below. Let x0 be the initial solution for the ILS process. A local search using some neighbourhoods structures is applied to x0 to find a better solution x'. Once a better solution is found, perturbation is performed on x' to obtain a new solution x. After that, a local search is applied again to x to obtain a new local optimum x''. The solution x'' replaces x' only if x'' is better than x' is verified and the ILS process continues until a stopping criterion is met.

```
1 let x0 be an initial solution;
2 x'← LOCAL SEARCH (x0)
3 repeat
4 x←PERTURBATION (x');
5 x''←LOCAL SEARCH (x);
6 if FEVAL(x') <FEVAL(x'') then
7 until Termination condition is met
```

Fig. 8. Outline of local search algorithm

## V. COMPUTATIONAL RESULTS

The proposed heuristic described in the previous section was coded in MATLAB. The experiments are performed on a desktop PC Intel Pentium IV, windows XP, 3.2 GHz processor and 1 GB memory.

### *A. Test instances*

We tested our algorithm on randomly generated instances as described in *Fig. 2*. There are four classes of problems: C1, C2, C3, C4, which determine the number of nodes in the basic graph G1.

### *B. Parameter setting*

We use the following parameters in our experimentation: the genetic parameters are a population size M = 30; additionally, our mutation operations are parameterized with the probability of 0.9. The algorithm was terminated whenever it has reached the optimal or after1000 generations. The algorithm was tested ten times for every class of iterations.

### *C. Comparative study*

In this section, a computational study is carried out to compare the performance of our approach with the optimal solution. The results of separately running the GA are compared to that of GA*ILS. According to the computational experiments, the GA*ILS algorithm outperforms the results of GA, both in terms of average fitness value and time. We use the following notations: %gap, the average deviation of solution value to the optimal solution (in percents), the standard deviation of fitness values for every class of

problems and Time, the average running time over ten instances. Table II summarizes the results of GA and GA*ILS.

**Table II. Percent deviations to optimum and CPU time**

| | GA | | | GA*ILS | | |
|---|---|---|---|---|---|---|
| | Ave.FF | Std. FF | Time | Ave. FF | Std. FF | Time |
| Uniform[7,12] | 0.11% | 0.11% | 2.95 | 0.09% | 0.09% | 2.18 |
| Uniform[13,18] | 0.85% | 0.97% | 5.70 | 0.71% | 0.67% | 3.96 |
| Uniform[19,24] | 2.53% | 2.38% | 9.50 | 2.06% | 2.04% | 7.74 |
| Uniform[25,30] | 5% | 5% | 17.60 | 4.50% | 4.87% | 13.31 |

## VI. CONCLUSION

In this paper a new improved genetic algorithm for subgraph optimal isomorphism was introduced. In order to justify the application of GA the landscape analysis of the SIP, was conducted with an iterative local search and the correlation of the fitness values with a distance measure of local solutions was calculated for four classes of instances.

The concept of the permutation matrix was used for the matching process. By applying permutation matrix, we could also define an efficient fitness function for the optimization version of the Subgraph Isomorphism problem which made possible the generation of instance problems of subgraph isomorphism for the first time.

A hybrid approach that combines a GA with an ILS to solve the SIP efficiently was developed for solving the SIP. Our hybridization is based on an ILS using a neighborhood structure and allows us to improve each of the generations outputted by a GA.

The proposed GA&ILS algorithm was tested on four randomly generated problem sets. The computational results that we have conducted show that the hybrid approach outperforms the GA.

## REFERENCES


[1] Brown, D., Thomas. P.R., Coal-oriented subgraph isomorphism technique for IC device recognition.IEE Proceeding I (Solid-state and Electron Devices) 1998, 135: 141-150.

[2] Cook, S. A. (1971), "The complexity of theorem-proving procedures", Proc. 3rd ACM Symposium on Theory of Computing, pp. 151–158, doi:10.1145/800157.805047.

[3] Eppstein. D., Subgraph Isomorphism in Planar Graphs and Related Problems. Journal of Graph Algorithms and Application, 1999, 3(3): 1-27.

[4] Glover, F., &Kochenberger, G. (2002).Handbook of metaheuristics. Boston: Kluwer [Chap. Iterated Local Search, pp. 321–353].

[5] Guha. A., Optimizing codes for concurrent fault detection in microprogrammed controllers. Proc. Int. Conf. Computer Design: VSLI in Computers and Processors (ICCD'87). pp. 486-489, IEE, 1987.

[6] Jones T., and Forrest, S., "Fitness distance correlation as a measure of problem difficulty for genetic algorithms," in Proceedings of the 6th International Conference on Genetic Algorithms, 1995, pp. 184–192.

[7] Lang, S.Y.T, Wong. A.K.C., A sensor model registration technique for mobile robot localization.IEE Proc. Int. Symp. Intelligent Control, 1991, pp. 298-305

[8] Luigi P. Cordella, Pasquale Foggia, Carlo Sansone, Mario Vento. A (Sub)Graph Isomorphism Algorithm for Matching Large Graphs.IEEE Trans. on Pattern Analysis and Machine Intelligence, 2004, 26(10): 1367-1372.

[9] Merz, P., Freisleben, B., "Fitness Landscape Analysis and Memetic Algorithms for the Quadratic Assignment Problem," IEEE Transactions On Evolutionary Computation, vol. 4, no. 4, pp. 337–352, 2000.

[10] Deo, Narsingh, Graph Theory with Applications to Engineering and Computer Science, Printice-Hall, Englewood Cliffs, New Jersey (1995).

[11] Gold, Steven, Rangarajan.,Anand. A Graduated Assignment Algorithm for Graph Matching. IEEE Trans. on Pattern Analysis and Machine Intelligence, 1996, 18(4): 377-388.

[12] Ullmann. J.R., An Algorithm for Subgraph Isomorphism. Journal of the Association for Computing Machinery, 1976, 23(1): 31-42.

[13] Wong, K. C. , You, M., and Chan, S. C., An algorithm for graph optimal monomorphism, IEEE Trans. Systems,ManCybernet. 20(3), 628-636 (1990).

[14] Zhong, Q., Wu, Z., Lin, L., Zhang, J., Zhang, Y., Computing Resources Assignment in RTDS Simulators with Subgraph Isomorphism based on Genetic Algorithm,4th International Conference on Electric Utility Deregulation and Restructuring and Power Technologies (DRPT), 2011. 1144 – 1149


## BIOGRAPHY

**Dr. Seyed Kamal Chaharsooghi** is Associate Professor of Industrial Engineering at the Dept. of Industrial Engineering, Faculty of Engineering, Tarbiat Modares University, Tehran, Iran. Dr. Chaharsooghi's research interests include: manufacturing systems, supply chain management, information systems, strategic management, international marketing strategy and systems theory. Dr. Chaharsooghi's papers have appeared in European Journal of Operational Research, International Journal of Advanced Manufacturing Technology, Scientia Iranica, Modares Journal of Engineering, Amirkabir Journal of Science and Technology, International Journal of Engineering Science. Dr. Chaharsooghi obtained his PhD from Hull University, England.

**Mina Mazraeh Farahani** is crrently Ph.D candidate of Industrial Engineering at the Dept. of Industrial Engineering, Faculty of Engineering, Tarbiat Modares University, Tehran, Iran. Ms. Farahani is also a lecturer at Payam-E-Noor Univerity of Iran . She was an analyst at the Department of Planning and Organization, Iran Air Company for five years. Ms. Farahani holds a Bachelor of Science degree in Pure Mathematics from Amirkabir University of Technology, Tehran, Iran and a Master of Scince in Industrial Engineering from Iran University of Scince and Technology, Tehran.