DE GRUYTER OPEN

JAISCR, 2015, Vol. 5, No. 4, pp. 239– 245
10.1515/jaiscr-2015-0032

OPTIMIZATION OF TRAVELING SALESMAN PROBLEM USING AFFINITY PROPAGATION CLUSTERING AND GENETIC ALGORITHM

Ahmad Fouad El-Samak[1] and Wesam Ashour[2]

[1]*Computer Engineering Department, Islamic University of Gaza, Gaza, Palestine*
af.samak@alaqsa.edu.ps

[2]*Computer Engineering Department, Islamic University of Gaza, Gaza, Palestine*
washour@iugaza.edu.ps

Abstract

Combinatorial optimization problems, such as travel salesman problem, are usually NP-hard and the solution space of this problem is very large. Therefore the set of feasible solutions cannot be evaluated one by one. The simple genetic algorithm is one of the most used evolutionary computation algorithms, that give a good solution for TSP, however, it takes much computational time. In this paper, Affinity Propagation Clustering Technique (AP) is used to optimize the performance of the Genetic Algorithm (GA) for solving TSP. The core idea, which is clustering cities into smaller clusters and solving each cluster using GA separately, thus the access to the optimal solution will be in less computational time. Numerical experiments show that the proposed algorithm can give a good results for TSP problem more than the simple GA.

1 Introduction

Traveling Salesman Problem (TSP) is a well-known and extensively studied NP-hard combinatorial optimization problem and the aim of TSP is to find the shortest tour that visits every city once for a given list of cities and back to the starting city [1]. This problem has been attracting the attention of many researchers and remains an active research area, due to a large number of real-world problems that can be modeled by TSP. For example, planning, scheduling and searching in scientific and engineering fields, such as vehicle routing, manufacturing, and computer operations. This problem is hardly solvable through finding the exact solution directly. Thus, many heuristics and approximation algorithms have been proposed to produce the useable solutions for TSP, such as neural networks [2, 3], simulated annealing [4], genetic algorithm [5, 6], and ant colony optimization [7, 8, 9]. These methods can produce the good solutions of TSP, but they still use a lot of computational time in the large TSP.

The clustered travelling salesman problem (CTSP) is a variation of the usual travelling salesman problem in which a set of cities of the tour is divided into some clusters. CTSP attempts to compute the shortest Hamiltonian tour that visits all the cities, in which the cities of each cluster are visited consecutively [10]. The objective is to improve the computational time. Since the nodes in the large TSP are clustered, the TSP problem should be smaller and the computational time might be reduced.

The rest of the paper is organized as follows; in Section II, some related works to solving CTSP problem will be discussed. The basic concepts of affinity propagation clustering and genetic algorithm are briefly reviewed in Section III. Section IV presents the proposed method used in this paper. The experimental results are shown in Section V and the conclusion is presented in Section VI.

2 Related Works

There are some researches on CTSP, such as in [11], Anily, Bramel, and Hertz present a 5/3-approximation algorithm for the ordered CTSP, which runs in O(n^3) time. It is an adaptation of Christodes' Heuristic for the TSP.

Ding Chao el al. [12] developed a two-level genetic algorithm (TLGA) for the clustered traveling salesman problem (CTSP). In the lower level, a genetic algorithm (GA) is used to find the shortest Hamiltonian cycle rather than the shortest Hamiltonian path for each cluster. In the higher level, a modified genetic algorithm is designed to determine which edge will be deleted from the shortest Hamiltonian cycle for each cluster, and the visiting sequence of all the clusters with the objective of shortest traveling tour for the whole problem. However, the genetic algorithm technique for ordering the CTSP used a lot of computational time. Moreover, this technique did not always generate a good result.

Tanasanee [13] proposed the use of clustering technique, i.e. Gaussian mixture model and *k*-means, to improve the performance of the evolutionary computation algorithm, i.e. genetic algorithm and ant colony optimization, on the traveling salesman problem. In the first step, nodes in the TSP are grouped by a clustering technique in order to group the nearest nodes in the problems. Then, the genetic algorithm and the ant colony optimization are used to find the optimal path. In the final step, a simple method for choosing clusters and nodes is presented to connect all clusters in the TSP. The path of all clusters is connected by considering the centroids of clusters and the marginal nodes.

3 Background

3.1 Affinity Propagation Clustering

Clustering data by identifying data centers or exemplars, are traditionally found by randomly choosing an initial subset of data points and then iteratively refining it, but this only works well if that initial choice is close to a good solution. Affinity propagation is a new algorithm that takes as input measures of similarity between pairs of data points and simultaneously considers *all* data points as potential exemplars. Real-valued messages are exchanged between data points until a high-quality set of exemplars and corresponding clusters gradually emerges [14]. Unlike clustering algorithms such as *k*-means or *k*-medoids, AP does not require the number of clusters to be determined or estimated before running the algorithm.

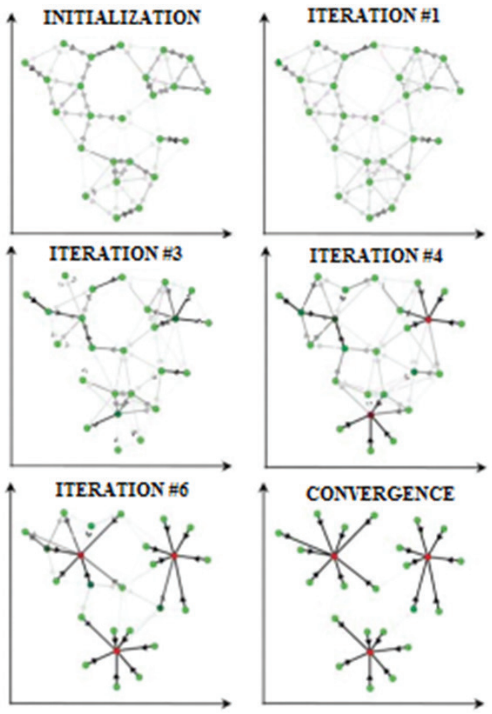

Figure 1. Affinity propagation for 2-dimensional data points

Affinity propagation has been used to solve a variety of clustering problems, for example, to cluster images of faces, detects genes in microarray data, clustering in Vehicular Ad hoc Networks [16], solving large immobile location–allocation problem [17] and to identify cities that are efficiently accessed by airline travel. Affinity propagation found clusters with much lower error than other methods, and it did so in less than one-hundredth the amount of time.

Figure 1 illustrate affinity propagation for two-dimensional data points [14], where negative Euclidean distance (squared error) was used to measure similarity. Each point is colored according to the current evidence that it is a cluster center (exemplar). The darkness of the arrow directed from point i to point k corresponds to the strength of the transmitted message that point i belongs to exemplar point k.

The algorithm proceeds by alternating two message passing steps, to update two matrices which are responsibility and availability matrices. Figure 2 illustrate responsibility $r(i, k)$ [14], which are sent from data points to candidate exemplars and indicate how strongly each data point favors the candidate exemplar over other candidate exemplars.

In addition, availability $a(i, k)$ which are sent from candidate exemplars to data points and indicate to what degree each candidate exemplar is available as a cluster center for the data point.

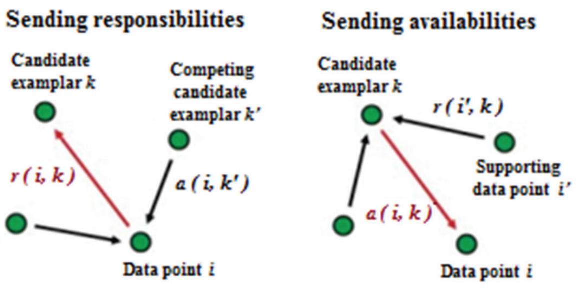

Figure 2. Illustration of responsibilities and availabilities massages

3.2 Genetic Algorithm

The genetic algorithm involves the encoding of solutions as chromosomes. The length of chromosome is the number of cities in the problem. Each gene of a chromosome takes a label of city such that no city can appear twice in the same chromosome. The fitness of these chromosomes is evaluated by computing cost (distance) of each complete tour.

The first step in genetic algorithm is to create initial population. The population is generated randomly. The creation of a new population is performed by the genetic operators, i.e. fitness proportionate selection, crossover, and mutation. As shown in Figure 3, these new chromosomes are selected based on their fitness. The best chromosome should survive and become the original breed for the next generation. In this algorithm, we use the cycle crossover (CX) [15] and three different mutation operators, i.e. Reverse Sequence Mutation (Flip), Swap, and Slide mutation, with the intent to make modifications that are more likely to improve the best solutions.

The old population is replaced by the new one. These steps are repeated for a number of generations. In the end, the best chromosome is decoded to obtain a solution.

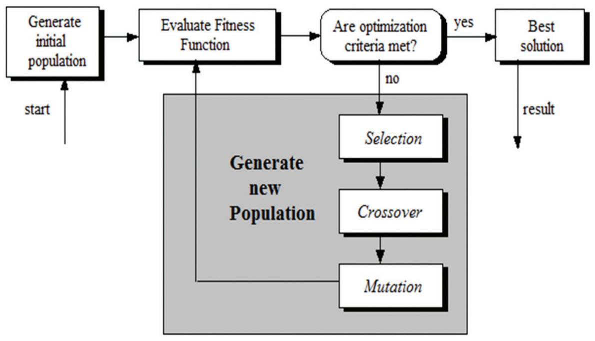
Generate initial population
start
Evaluate Fitness Function
Are optimization criteria met?
yes
no
Best solution
result
Generate new Population
Selection
Crossover
Mutation

Figure 3. The operation of genetic algorithm

4 Proposed Work

The main idea of this paper is to use affinity propagation clustering technique, to optimize the performance of the genetic algorithm, on the traveling salesman problem. This section presents the steps of solving the TSP. The flow of clustering genetic algorithm is shown in Figure 4.

In the simple genetic algorithm, the first step is to create initial population. The population is generated randomly. However, in this approach, instead of generating the initial population randomly, a new procedure is used to create the initial population. This procedure is explained below.

1. Figure 5 explains TSP problem with 70 cities before clustering. These cities are clustered using affinity propagation into undetermined number of clusters (k). In this case, TSP problem was clustered into six clusters as shown in Figure 6.

2. After clustering, apply GA on each generated cluster (k_1,k_2,..., k_L) separately, and compute the minimum path length for each cluster (p_1, p_2,...,p_L). Figure 7 shows the minimum path length for each cluster after applying GA on each.

3. After that, generate new chromosome from the paths of all clusters, by selecting the first path

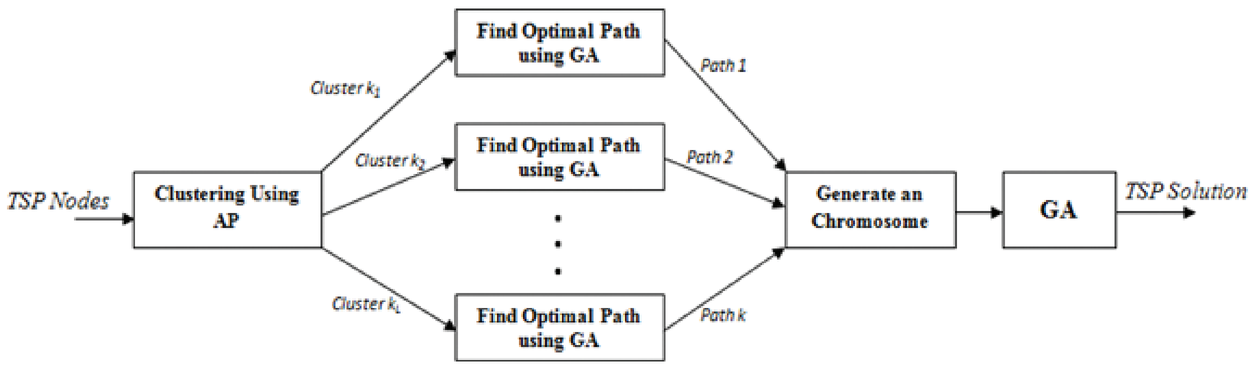

Figure 4. Clustering genetic algorithm steps

p_1 from cluster k_1, the next path p_2 is selected from cluster k_2, and so on until the selection of last path p_L from cluster k_L. The process of generating new chromosome is depicted in Figure 8.

4. Then, initialize all chromosomes of the population with the new chromosome.

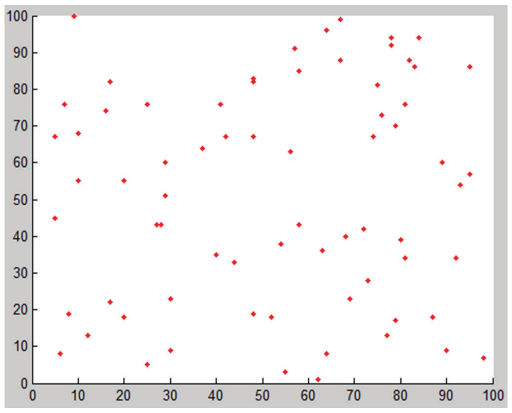

Figure 5. TSP problem before AP clustering

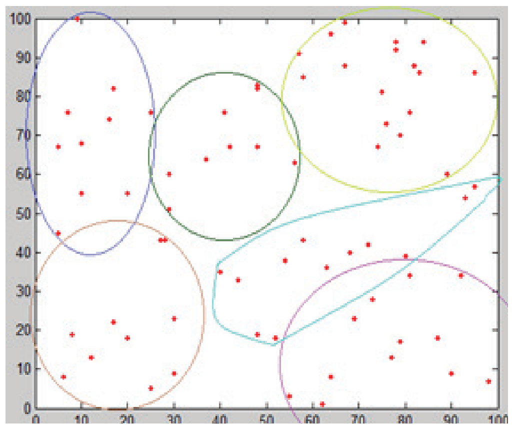

Figure 6. TSP problem after AP clustering

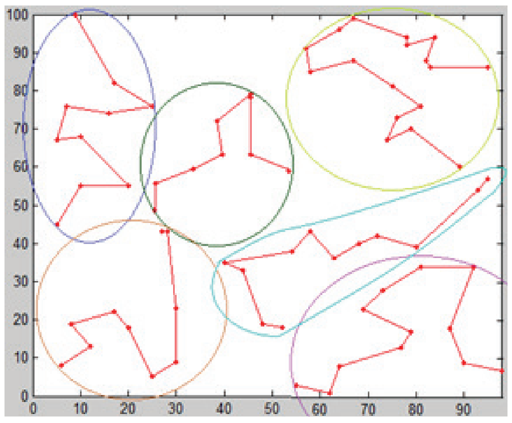

Figure 7. The minimum path length for each cluster

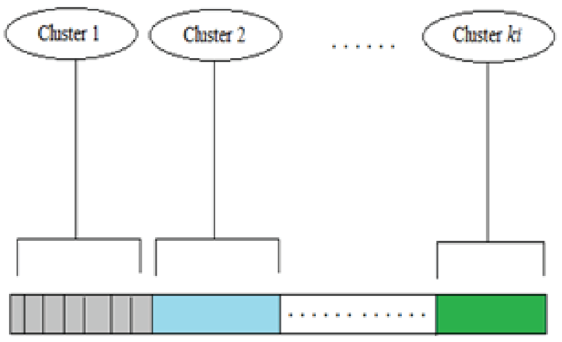

Figure 8. Generate new chromosome from the paths of all clusters

Finally, after generating the initial population using the above procedure, the remaining steps of a simple GA can be applied on this initial population.

Figure 9 shows a complete TSP solution after applying GA on the new initial population. The idea is that the process of clustering cities into smaller clusters and solving each cluster separately, make it easily to access to the optimal solution in a very short time.

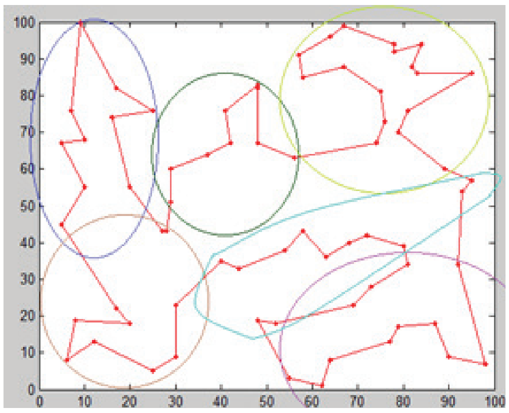

Figure 9. A complete TSP solution

Table 1. Datasets for testing

No.	Dataset	#Cities	Optimal Length
1	eil51	51	426
2	berline52	52	7,542
3	st70	70	675
4	ch150	150	6,528
5	kroB200	200	29,437
6	lin318	318	42,029
7	att532	532	27,686
8	d657	657	48,912

5 Experimental Results

The proposed method is tested on 8 benchmark problems which are taken from the TSPLIB test problems [18]. These datasets are symmetric TSPs in the two-dimensional Euclidean distance. Experiments are performed on a desktop with 2GHz Core 2 Duo CPU and 3GB RAM, under Windows 8 OS. The program was written in the MATLAB programming language. The number of cities and the optimal tour length of these datasets are shown in Table 1.

The experimental results of the simple genetic algorithm and genetic algorithm based on affinity propagation clustering are compared in Table 2. The minimum tour length and the computational time are illustrated. For the parameter settings, population size determines how many chromosomes and thereafter, how much genetic material is available for use during the search. If there is too little, the search has no chance to adequately cover the space. If there is too much, the GA wastes time evaluating chromosomes. For this reason, the choice of population size is between 160 and 240. The maximum number of iterations is 10^6.

Table 2 shows that the clustering genetic algorithm is better than the simple genetic algorithm, since it gives minimum tour lengths and less computational times than simple GA. For example, on att532, the minimum tour length of the clustering genetic algorithms are significantly better than those on the genetic algorithm, and mostly it do in half of its computational time. The differences in the minimum tour lengths are compared by the chart in Figure 10.

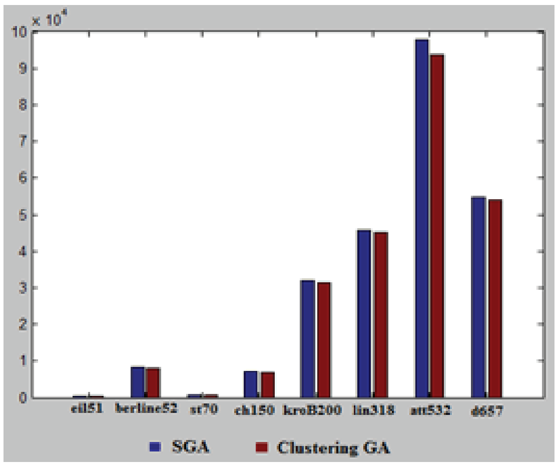

Figure 10. The minimum tour length

For the computational time, affinity propagation clustering can improve the computational time of the simple genetic algorithm. Chart in Figure 11 illustrates that the computational times of the simple GA and clustering GA.

Table 2. The results of clustering by genetic algorithm

No.	Datasets	Pop size	Minimum Tour Length			Computational Time (Sec.)	
			Simple GA	**GA with AP**		**Simple GA**	**GA with AP**
1	eil51	160	440	437		192	178.9
2	berline52	160	8,223	7,974		421	70
3	st70	160	713	702		109	72
4	ch150	160	7,098	6,762		1,242	166
5	kroB200	240	32,360	31,431		310	87
6	lin318	240	45,844	45,226		930	192.5
7	att532	240	97,704	93,633		1,214	542
8	d657	240	54,889	53,981		1,682	1,027

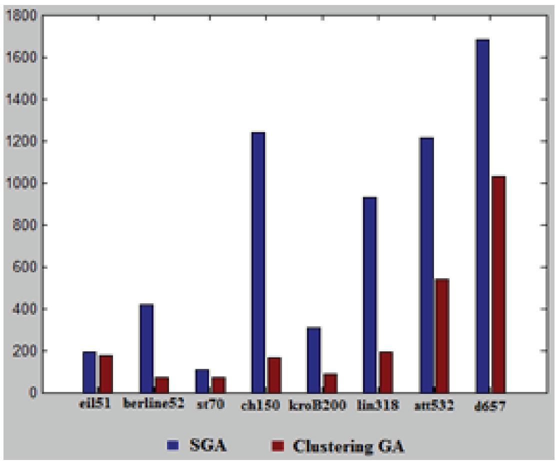

Figure 11. The computational time

6 Conclusion

In conclusion, a new method to optimize the performance of the genetic algorithm on TSN has been developed in this paper.

Clustering the cities using affinity propagation and then initialize all the population with the chromosome that composed minimum path lengths from each cluster is an effective method for solving TSP using the genetic algorithm. This idea might be useful for other problems solved by the genetic algorithm.

References

[1] Liao Y.-F., Yau D.-H., Chen C.-L., Evolutionary algorithm to traveling salesman problems, Computers & Mathematics with Applications, Vol. 64, Issue 5, pages 788–797, 2012.

[2] Leung, K.S., Jin H.D., & Xu, Z.B., An expanding self-organizing neural network for the traveling salesman problem, Neurocomputing, 62, 267–292. Masutti, T.A.S., & de Castro, L.N. (2004).

[3] Masutti, T.A.S., & de Castro, L.N., A self-organizing neural network using ideas from the immune system to solve the traveling salesman problem, Information Sciences, Vol. 179, Issue 10, pages 1454–1468, April 2009.

[4] Lo, C.C., & Hsu, C.C., Annealing framework with learning memory, IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans, Vol. 28, Issue 5, pages 648-661, 1998.

[5] Jayalakshmi, G.A., & Sathiamoorthy, A hybrid genetic algorithm – A new approach to solve traveling salesman problem, International Journal of Computational Engineering Science, Vol. 2, Issue 2, pages 339–355, 2001.

[6] Tsai, H.K., Yang, J.M., Tsai, Y.F., & Kao, C.Y., Heterogeneous election genetic algorithms for traveling salesman problems, Engineering Optimization, Vol. 35, Issue 3, pages 297–311, 2003.

[7] Dorigo, M., & Gambardella, L.M., Ant colony system: A cooperative learning approach to the traveling salesman problem, IEEE Transactions on Evolutionary Computation, Vol. 1, Issue 1, pages 53-66, 1997.

[8] Liu, J.L., Rank-based ant colony optimization applied to dynamic traveling salesman problems, Engineering Optimization, Vol. 37, Issue 8, pages 831–847, 2005.

[9] Puris, A., Bello, R., & Herrera, Analysis of the efficacy of a two-stage methodology for ant colony optimization: Case of study with TSP and QAP, Expert Systems with Applications, Vol. 37, Issue 7, pages 5443–5453, 2010.

[10] Laporte, G., & Palekar U, Some applications of the clustered travelling salesman problem, The Journal of the Operational Research Society, Vol. 53, Issue 9, pages 972–976, 2002.

[11] Anily, S., Bramel, J., & Hertz, A., A 5/3-approximation algorithm for the clustered traveling salesman tour and path problems, Operations Research Letters, Vol. 24, Issue1, pages 29-35, 1999.

[12] Ding Chao, Cheng Ye, He Miao, Two-Level Genetic Algorithm for Clustered Traveling Salesman Problem with Application in Large-Scale TSPs, Tsinghua Science & Technology, Vol. 12, Issue 4, Pages 459–465, August 2007.

[13] Tanasanee Phienthrakul, Clustering Evolutionary Computation for Solving Travelling Salesman Problems, International Journal of Advanced Computer Science and Information Technology (IJACSIT), Vol. 3, Issue 3, Page: 243-262, 2014.

[14] B. J. Frey and D. Dueck. Response to Comment on Clustering by Passing Messages Between Data Points. Science, Vol. 319, Issue 5864, pages 726, 8 February 2008.

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

[16] B. Hassanabadi, C.Shea, L.Zhang, S.Valaee, Clustering in Vehicular Ad Hoc Networks using Affinity Propagation, Original Research Article Ad Hoc Networks, Vol. 13, Part B, Feb. 2014, Pages 535–548.

[17] F. Torrent-Fontbona, V. Muoz, B. Lpez Solving large immobile location–allocation by affinity propagation and simulated annealing, Expert Systems with Applications, Vol. 40, Issue 11, 1 Sep. 2013, Pages 4593-4599.

[18] http://www2.iwr.uniheidelberg.de/groups/comopt/software/TSPLIB9 5/tsp/

Ahmad Fouad El-Samak received the B.Eng. degree in Computer System Engineering from Palestine Technical Collage, Deir El-Balah City, Palestine, in 2009. He is currently a Network Engineer at Al-aqsa University, Gaza. His research interests are in the areas of communication systems, network design and maintenance.

Wesam Ashour received B.Sc. in Electrical and Computer Engineering from Islamic University of Gaza, in 2000. He has worked at IUG for 3 years as a teaching assistant. He has finished his M.Sc. in Multimedia with Distinction in 2004 from the University of Birmingham, UK. After that, he has returned back to Gaza and he has joined the staff of Electrical and Computer Engineering for one year. In 2005, he has got a scholarship from the University of the West of Scotland (UWS), UK, for his PhD. During his PhD study, he has worked in UWS as a teaching assistant and lab demonstrator for some modules. Currently, Dr Ashour is a head of the Computer Engineering Department at IUG.