An Application of Imperialist Competitive Algorithm to Solve the Quadratic Assignment Problem

Ali Safari Mamaghani
Computer Engineering Department
Islamic Azad University, Bonab Branch
Bonab, Iran
Ali.Safari.m@gmail.com

Mohammad Reza Meybodi
Computer Engineering and Information Technology Department
AmirKabir University of Technology
Tehran, Iran
mmeybodi@aut.ac.ir

Abstract— **Imperialist Competitive Algorithm (ICA) is a new socio-politically motivated global search strategy that has recently been introduced for dealing with different optimization problems. In this paper, we adopt ICA to solve the quadratic assignment problem which is a NP-Complete problem and is one of the most interesting and most challenging combinatorial optimization problems in existence. We test our algorithm on some of the benchmark instances of QAPLIB, a well-known library of QAP instances. This algorithm is compared with two meta heuristic strategies. These methods are based on simulated annealing approach and genetic algorithm. In most of instances, the proposed method outperforms other approaches. Experimental results illustrate the effectiveness of ICA approach on the quadratic assignment problem.**

Keywords- Quadratic Assignment Problem; Imperialistic Competitive Algorithm; NP-Complete problem; meta heuristic algorithms.

I. Introduction

The Quadratic Assignment Problem (QAP) is one of the classical combinatorial optimization Problems and is widely regarded as one of the most difficult problem in this class. Given a set $N = \{1,2,3,...,n\}$ and $n \times n$ matrices $F = \{f_{ij}\}$ and $D = \{d_{ij}\}$, and $C = \{c_{ij}\}$, the QAP is to find a permutation ϕ of the set N which minimizes

$$z = \sum_{i=1}^{n}\sum_{j=1}^{n} f_{ij} d_{\phi(i)\phi(j)} + \sum_{i=1}^{n} c_{i\phi(i)}$$

As an application of the QAP, consider the following campus planning problem. On a campus, new facilities are to be erected and the objective is to minimize the total walking distances for students and staffs. Suppose there are *n* available sites and *n* facilities to locate. Let d_{kl} denotes the walking distance between the two sites *k* and *l* where the new facilities will be erected. Further, let $f_{i,j}$ denotes the number of people per week who travel between the facilities *i* and *j*. Then, the decision problem is to assign facilities to sites so that the walking distance of people is minimized. Each assignment can be mathematically described by a permutation ϕ of $N = \{1,2,3,...,n\}$ such that $\phi(i) = k$ means that the facility *i* is assigned to site *k*. The product $f_{ij} d_{\phi(i)\phi(j)}$ describes the weekly walking distance of people who travel between facilities i and j. Consequently, the problem of minimizing the total walking distance reduces to identifying a permutation ϕ that minimizes the function *z* defined above. This is an application of the QAP with each $c_{ik} = 0$. In this application, we have assumed that the cost of erecting a facility does not depend upon the site. In case it does, we will denote by $c_{i,k}$ the cost of erecting facility *i* at site *k*, and these costs will also play a role in determining the optimal assignment of facilities to sites [1].

Additional applications of QAP include the allocation of plants to candidate locations, layout of plants, backboard wiring problem, design of control panels and typewriter keyboards, balancing turbine runners, ordering of interrelated data on a magnetic tape, processor-to-processor assignment in a distributed processing environment, placement problem in VLSI design, analyzing chemical reactions for organic compounds, and ranking of archaeological data.

On account of its diverse applications and the intrinsic difficulty of the problem, the QAP has been investigated extensively by the research community. The QAP has been proved to be an NP-complete problem and a variety of exact and heuristic algorithms have been proposed.

Exact algorithms for QAP include approaches based on dynamic programming [2], cutting planes [3], and branch and bound [4, 5]. Among these, only branch and bound algorithms are guaranteed to obtain the optimum solution, but they are generally unable to solve problems of size larger than n=20.

Since many applications of QAP give rise to problems of size far greater than 20, there is a special need for good heuristics for QAP that can solve large-size problems. Known heuristics for QAP can be classified into the following categories: construction methods [6, 7], limited enumeration methods [8, 9], local improvement methods [10], simulated annealing methods [11], tabu search methods [12, 13] and genetic algorithms [12, 14]. Among these, the tabu search method due to Skorin-Kapov [12] and the (randomized) local improvement method due to Li et al. [10] and Pardalos et al. [15], which they named as Greedy Randomized Adaptive Search Procedure (GRASP), are the two most accurate heuristic algorithms to solve the QAP.Other approaches were used to solve the QAP problem are ant colony algorithm [16, 17], Path Relinking method [18], hybrid GRASP approach with Tabu search[19], hybrid Ant Colony approach with Genetic and Hill Climbing[20] and Parallel Tabu Search[21].

The structure of the reminder of this paper is as follows: Section 2 is an introduction to ICA algorithm. In section 3, we describe the proposed algorithm based on ICA to solve the problem. Section 4 and 5 are dedicated to describe experimental results and paper conclusion respectively.

II. Introduction to Imperialist Competitive Algorithm

Imperialist Competitive Algorithm is a new evolutionary optimization method which is inspired by imperialistic competition [22]. Like other evolutionary algorithms, it starts with an initial population which is called country and is divided into two types of colonies and imperialists which together form empires. Imperialistic competition among these empires forms the proposed evolutionary algorithm. During this competition, weak empires collapse and powerful ones take possession of their colonies. Imperialistic competition converges to a state in which there exists only one empire and colonies have the same cost function value as the imperialist. The pseudo code of Imperialist competitive algorithm is as follows:

```
1) Select some random points on the function and initialize the empires.
2) Move the colonies toward their relevant imperialist (Assimilation).
3) Randomly change the position of some colonies (Revolution).
4) If there is a colony in an empire which has lower cost than the imperialist, exchange the positions of that colony and the imperialist.
5) Unite the similar empires.
6) Compute the total cost of all empires.
7) Pick the weakest colony (colonies) from the weakest empires and give it (them) to one of the empires (Imperialistic competition).
8) Eliminate the powerless empires.
9) If stop conditions satisfied, stop, if not go to 2.
```

After dividing all colonies among imperialists and creating the initial empires, these colonies start moving toward their relevant imperialist state which is based on assimilation policy. Figure 1 shows the movement of a colony towards the imperialist.

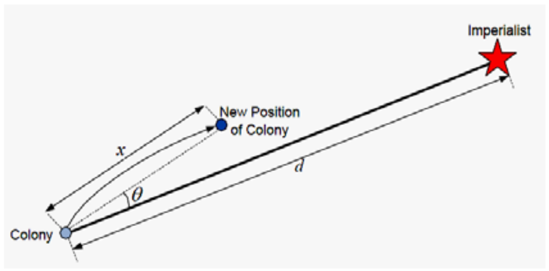


Figure 1. Moving colonies toward their relevant imperialist

In this movement, θ and x are random numbers with uniform distribution as illustrated in formula (1) and d is the distance between colony and the imperialist.

$$x \sim U(0, \beta \times d),\ \theta \sim U(-\gamma, \gamma) \qquad (1)$$

Where β and γ are parameters that modify the area that colonies randomly search around the imperialist. In our Implementation β and γ are considered as 2 and 0.5 (Radian) respectively.

In ICA, revolution causes a country to suddenly change its socio-political characteristics. That is, instead of being assimilated by an imperialist, the colony randomly changes its position in the socio-political axis. The revolution increases the exploration of the algorithm and prevents the early convergence of countries to local minimums.

The total power of an empire depends on both the power of the imperialist country and the power of its colonies which is shown in formula (2).

$$T.C._n = Cost(imperialist_n) + \xi\, mean\{Cost(colonies\ of\ empire_n)\} \qquad (2)$$

In imperialistic competition, all empires try to take possession of colonies of other empires and control them. This competition gradually brings about a decrease in the power of weaker empires and an increase in the power of more powerful ones. This is modeled by just picking some of the weakest colonies of the weakest empires and making a competition among all empires to possess these colonies. Figure 2 shows a big picture of the modeled imperialistic competition. Based on their total power, in this competition, each of the empires will have a likelihood of taking possession of the mentioned colonies. The more powerful an empire, the more likely it will possess the colonies. In other words these colonies will not be certainly possessed by the most powerful empires, but these empires will be more likely to possess them. Any empire that is not able to succeed in imperialist competition and can not increase its power (or at least prevent decreasing its power) will be eliminated.

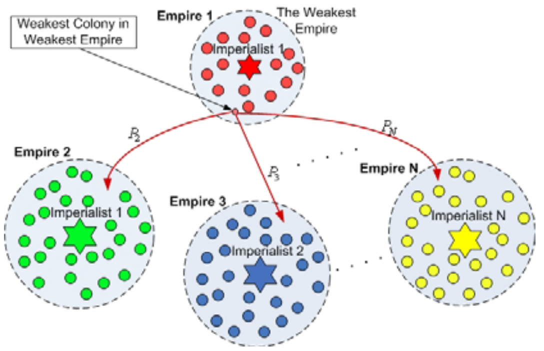


Figure 2. Imperialistic competition

ICA as a new evolutionary method which is used in several applications, such as designing PID controller, characterizing materials properties, error rate beam forming, designing vehicle fuzzy controller, etc. In this paper, we have applied this algorithm for solve the quadratic assignment problem.

III. An Application of ICA to Solve the Quadratic Assignment Problem

In this section we adapt ICA to solve the QAP problem. We describe details of basic steps of the proposed ICA algorithm which used to solve the problem.

Representation of the Countries: We form an array of variable values to be optimized. In the GA, this array is called chromosome, but in ICA the term country is used for this array. In this paper representation is employed, which is illustrated in Figure 3. We form an $1 \times n$ array as a country. Suppose there are *n* available sites and *n* facilities to locate in sites.

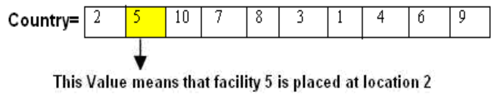


Figure 3. Representation of the Countries for QAP problem

In the above representation, the value of every cell in array represents the facility that is assigned to corresponding location. In the country shown in Fig. 3, there are 10 facilities to be placed at 10 locations. For example, the second cell in the chromosome means that facility 5 is placed at location 2. At first, the algorithm starts with initial population which is formed by randomly generated countries.

The Cost Function: The cost of each country can be built according to the specific problem. The cost function is to evaluate condition of each solution. In this case, we use C_m to evaluate each solution Where, C_m is the cost of county m. Each cost of county can be calculated by:

$$C_m = Cost(country_m) = \sum_{i=1}^{n}\sum_{j=1}^{n} f_{ij} d_{\phi(i)\phi(j)}$$

Assimilating Operator: The original version of Imperialist competitive algorithm operates on real values. But QAP problem is a discrete problem. So we use new assimilating and revolution operator which are suitable for the problem. An example of this operator is shown in figure 4. This operator is as below:

1) Choose some cells (about 50% of cells) randomly in imperialist array (cells {1, 6, 2, 8 and 7})

2) Copy selected cells into new_colony array.

3) Scan whole of colony array from first to end. If there is any cell which is not in new_colony, copy it in to first empty cell (cells {5, 10, 3, 4 and 9}).

Imperialist	1	5	6	2	9	10	3	8	4	7
Colony	2	5	10	7	8	3	1	4	6	9
New_Colony	1	5	6	2	10	3	4	8	9	7

Figure 4. An example of Assimilating Operator for QAP problem

Revolution operator: For revolution operator, we choose two different cells randomly in colony array and then exchange them. This operator is illustrated in figure 5.

Colony	2	5	10	7	8	3	1	4	6	9
New_Colony	2	5	6	7	8	3	1	4	10	9

Figure 5. An example of Revolution Operator for QAP problem

IV. Experimental Results

In this section the results of the implementation of the new algorithm is described. Experimental results and analysis are divided into two sections. First, in section A, we evaluate the effect of parameters on ICA algorithm.

In section B, we present the computational results of the ICA algorithm when applied to some instances of the QAPLIB compiled by Burkard et al. [23]. These groups can be downloaded from http://www.seas.upenn.edu/qaplib. Our algorithm was programmed in the C#.Net 2010 programming language and was implemented on a Intel Core Due 2000 G. HZ computer.

A. Evaluate the effect of parameters on ICA algorithm

It is clear that the performance of the approximated algorithms is affected by parameter tuning. So, at first we do tuning process, to obtain good values for the key parameters.

We used bur26f instance of QAPLIB to test the effect of ICA parameters. This instance contains 26 facilities. we use it and increase the number of iterations of algorithm ranging from 100 to 500. The experiment produces Figure 6. The first point which is showed that increasing the number of iterations generates solutions with low cost. The Second notification focuses on parameter tuning. As shown in the figure, the cost decreased rapidly up to 300 iterations. After that there is small decrease of the cost. So, to balance the quality of the results and the running time, we set the maximum number of iterations to be 300 for following experiments.

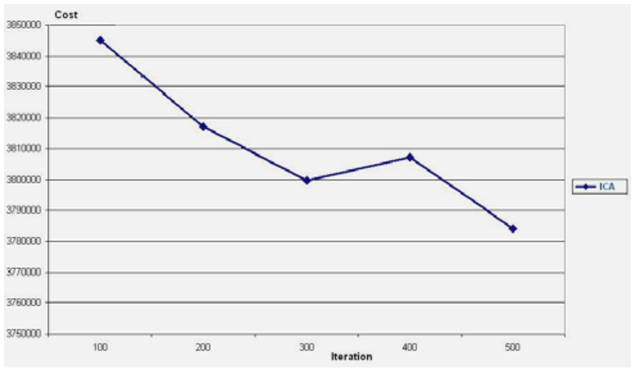


Figure 6. The impact of increasing the ICA's number of iterations on generated solution

To test the effect of the population size (the number of countries) we use ICA and select number of countries from the set {50, 80, 100, 120 and 150}. The figure 7 shows the results.

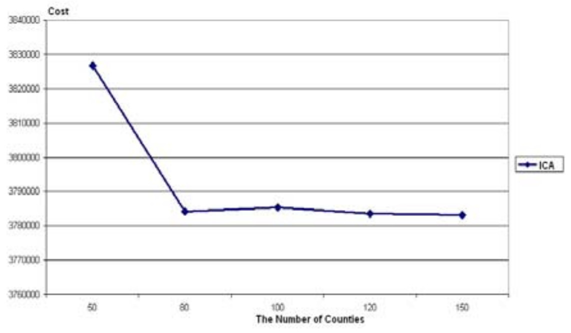


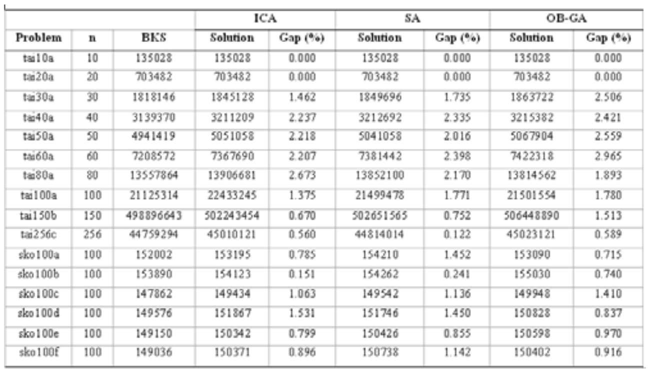
Figure 7. The impact of increasing the ICA's number of countries on generated solution.

As you see increasing the number of countries makes to generate solutions with low cost. Second notification is the cost decreased rapidly for up to 120 countries. So we've used 120 in our experiments.

B. *Performance of Algorithm*

We compared our Algorithm with simulated annealing method and a kind of genetic algorithm is called GA_OB. A set of test cases with different size are used. Results are shown in table1. In table 1, the first column contains the QAP instance and second contains the size of problem. The third column contains the Best-Known Solution (BKS). The Solution and gap Column are respectively the solutions produced by each algorithm and the deviation in percentage from BKS.

TABLE I. PERFORMANCE COMPARISON ACCORDING TO PROBLEM SIZE



			ICA		SA		OB-GA	
Problem	n	BKS	Solution	Gap (%)	Solution	Gap (%)	Solution	Gap (%)
tai10a	10	135028	135028	0.000	135028	0.000	135028	0.000
tai20a	20	703482	703482	0.000	703482	0.000	703482	0.000
tai30a	30	1818146	1845128	1.462	1849696	1.735	1863722	2.506
tai40a	40	3139370	3211209	2.237	3212692	2.335	3215382	2.421
tai50a	50	4941419	5051058	2.218	5041058	2.016	5067904	2.559
tai60a	60	7208572	7367690	2.207	7381442	2.398	7422318	2.965
tai80a	80	13557864	13906681	2.673	13852100	2.170	13814562	1.893
tai100a	100	21125314	22433245	1.375	21499478	1.771	21501554	1.780
tai150b	150	498896643	502243454	0.670	502651565	0.752	506448890	1.513
tai256c	256	44759294	45010121	0.560	44814014	0.122	45023121	0.589
sko100a	100	152002	153195	0.785	154210	1.452	153090	0.715
sko100b	100	153890	154123	0.151	154262	0.241	155030	0.740
sko100c	100	147862	149434	1.063	149542	1.136	149948	1.410
sko100d	100	149576	151867	1.531	151746	1.450	150828	0.837
sko100e	100	149150	150342	0.799	150426	0.855	150598	0.970
sko100f	100	149036	150371	0.896	150738	1.142	150402	0.916

We find out the results obtained by new approach is better than other two approach. So, it generates solutions which have low cost. In most cases the results are as same as BKS. So we can use ICA approach as an effective method to solve the QAP problem.

V. CONCLUSIONS

The quadratic assignment problem is a very difficult combinatorial optimization problem. In order to obtain satisfactory results in a reasonable time, heuristic algorithms are to be applied. In this paper, we developed a newly developed algorithm which is called ICA for solving QAP and presented computational results of the algorithm on a set of standard problem instances in literature. This algorithm is a meta heuristic approach which is inspired by imperialistic competition.

By comparison the result produced by ICA with the result produced by SA and GA, we find that ICA do better and find the solutions have low cost. These results are as same as Best-Known Solutions. So this results shows the ICA is an effective algorithm to solve the Quadratic Assignment Problem.

REFERENCES

[1] R.K. Ahuja, J.B. Orlin, and A. Tiwari," A greedy genetic algorithm for the quadratic assignment problem ", Computers and Operations Research 27, 917-934. 2000.

[2] Christo"des N, Benavent E. An exact algorithm for the quadratic assignment problem. Oper. Res. 1989;37:760-8.

[3] Bazara MS, Sherali MD. Benders' partitioning scheme applied to a new formulation of the quadratic assignment problem. Naval Res. Logist. Quart. 1980;27:29-41.

[4] Lawler EL. The quadratic assignment problem. Manag. Sci. 1963;9:586-99.

[5] Pardalos PM, Crouse J. A parallel algorithm for the quadratic assignment problem. Proceedings of the Supercomputing 1989 Conference, ACM Press. New York, 1989, pp. 351-60.

[6] Armour GC, Bu!a ES. Heuristic algorithm and simulation approach to relative location of facilities. Manag. Sci.1963;9:294 -309.

[7] Bula ES, Armour GC, Vollmann TE. Allocating facilities with CRAFT. Harvard Business Rev. 1962;42:136-58.

[8] West DH. Algorithm 608: approximate solution of the quadratic assignment problem.ACMTrans. Math. Software1983;9:461-6.

[9] Burkard RE, Bonniger T. A heuristic for quadratic boolean programs with applications to quadratic assignment problems. European J. Oper. Res. 1983;13:374-86.

[10] Li T, Pardalos PM, Resende MGC. A greedy randomized adaptive search procedure for the quadratic assignment problem. In: Pardalos PM, Wolkowicz H (Eds.), Quadratic Assignment and Related Problems. DIMACS Series in Discrete Mathematics and Theoretical Computer Science. Providence. RI: Americal Mathematical Society, 1994, pp. 237-261.

[11] Wilhelm MR, Ward TL. Solving quadratic assignment problems by simulated annealing. IEEE Trans.1987;19:107-19.

[12] Skorin-Kapov J. Tabu search applied to the quadratic assignment problem. ORSA J. Comput. 1990;2:33-45. R.K. Ahuja et al. Computers & Operations Research 27 (2000) 917-934.

[13] Taillard E. Robust tabu search for the quadratic assignment problem. Parallel Comput. 1991;17:443-55.

[14] Fleurent C, Ferland JA. Genetic hybrids for the quadratic assignment problem. DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 16, Providence. RI: American Mathematical Society, 1994, pp. 173-87.

[15] Resende MGC, Pardalos PM, Li Y. Fortran subroutines for approximate solution of dense quadratic assignment problems using GRASP. ACM Trans. Math. Software 1994;22:104-18.

[16] Stutzle, T., Dorigo, M., 1999. ACO algorithms for the quadratic assignment problem. In: Corne, D., Dorigo, M., Glover, F. (Eds.), New Ideas for Optimization. McGraw-Hill, pp. 33–50.

[17] Gambardella, L., Taillard, E., Dorigo, M., 1997. Ant Colonies for the QAP, Tech. Report IDSIA-4-97, IDSIA, Lugano, Switzerland.

[18] James, T., Rego, C., Glover, F., 2005. Sequential and parallel pathrelinking algorithms for the quadratic assignment problem. IEEE Intelligent Systems 20 (4), 58–65.

[19] Oliveira, C.A., Pardalos, P.M., Resende, M.G.C., 2004. GRASP with pathrelinking for the quadratic assignment problem, Efficient and Experimental Algorithms. In: Ribeiro, C.C., Martins, S.L. (Eds.), . In: Lecture Notes in Computer Science, vol. 3059. Springer-Verlag, pp. 356–368.

[20] Tseng, L., Rego, C., Glover, F.,2009. A cooperative parallel tabu search algorithm for the quadratic assignment problem. European Journal of Operational Research 195, 810–826.

[21] James, T., Liang, S., 2005. A hybrid metaheuristic for the quadratic assignment problem. Computational Optimization and Applications 34, 85–113.

[22] Atashpaz-Gargari, E., Lucas, C., 2007. "Imperialist Competitive Algorithm: An algorithm for optimization inspired by imperialistic competition". IEEE Congress on Evolutionary Computation, pp.4661–4667.

[23] Burkard RE, Karisch SE, Rendl F. QAPLIB - A quadratic assignment program library. J. Global Optim.1997;10:391-403.