Quadratic Assignment Problem

Hossein Shahbazi, Ali Eghbali Ghahyazi, Farhad Zeinali

Department of Industrial Engineering, Islamic Azad University, South Tehran Branch
shahbazi.hossein1368@gmail.com

Abstract: One of the well-known combinatorial optimization problems is the quadratic assignment problem (QAP). There are three directions of research in the QAP: proposing linearization method for the QAP formulation, developing heuristics/meta-heuristics to find near optimal solution and finding tight lower bound for optimal solution of the QAP. In this paper, we review these directions and next propose three algorithms to find near optimal solutions. The efficiency of our best proposed algorithm is tested in 59 instances of QAPLIB and our numerical results confirmed that our algorithm performs better than four algorithms existed in the literature of the QAP. [Hossein Shahbazi, Ali Eghbal Ghahyazi, Farhad Zeinali. **Quadratic Assignment Problem.** *J Am Sci* 2013;9(8):178-184]. (ISSN: 1545-1003). http://www.jofamericanscience.org. 26

Keywords: Quadratic assignment problem; meta-heuristics; Single-period facility layout problem; Mixed integer programming.

1. Introduction

One of the oldest combinatorial optimization problems in facility planning is the quadratic assignment problem (QAP) that it has been shown as an NP-Hard problem (Sahni and Gonzalez, 1976). The QAP was first proposed by Koopmans and Beckmann (1957) and there are many researches focused on the QAP so far. Referring to Burkard et al. (1998), the QAP can be defined as the problem of allocating a set of facilities to a set of locations, with the cost being a function of the distance and flow between the facilities, plus costs associated with a facility being placed at a certain location. The objective is to assign each facility to a location such that the total cost is minimized. The QAP can be formulated as follows:

$$Min \sum_{q=1}^{n}\sum_{p=1}^{n}\sum_{j=1}^{n}\sum_{i=1}^{n} d_{pq} f_{ij} y_{ipjq} + \sum_{p=1}^{n}\sum_{i=1}^{n} c_{ip} x_{ip} \quad (1.1)$$

$$\sum_{i=1}^{n} x_{ip} = 1 \qquad \forall p \quad (1.2)$$

$$\sum_{p=1}^{p} x_{ip} = 1 \qquad \forall i \quad (1.3)$$

$$x_{ip} \in \{0,1\}, \qquad \forall i,p \quad (1.4)$$

n: number of facilities,

c_{ip}: Cost of installing facility i to location j,

d_{pq}: Distance between locations p and q,

f_{ij}: Material flow from facility i to facility j,

$$x_{ip} = \begin{cases} 1, & \text{If facility } i \text{ is located in location } p \\ 0, & \text{Otherwise} \end{cases}$$

Objective function is to minimize material handling cost and cost of installing facilities. Constraints (1-2) and (1-3) state that each location must be occupied by only one facility and each facility must be located in only one location. Since the linear term of (1.1) is easy to solve, most authors ignored it, moreover the non-linear term of (1.1) can be reformulated as

$$x_{ip}x_{jq} = \frac{1}{2}\left(\left(x_{ip} + x_{jq}\right)^2 - x_{ip} - x_{jq}\right)$$

where it is implied that the QAP can be formulated as a convex binary problem. Several researches have been implemented in the QAP in three directions: proposing linearization method for problem (1), developing heuristics/meta-heuristics to find near optimal solution and finding tight lower bound for optimal solution of the QAP. A useful survey was written by Loiola et al. (2006) classified 368 references in these directions. Moreover, the QAP library (QAPLIB) has provided sets of instances with their best known solution and lower bound. In this paper, we study the QAP and particularly, the single-period facility layout problem (SFLP). The contributions of this paper are mentioned as follows:

- We develop an efficient algorithm to solve the QAP near optimally.
- We explain some directions for the future research's direction and explain their relation with the QAP.

Remainder of paper is organized as follows: In section 2, directions of research in QAP are reviewed. In section 3, we develop four meta-heuristics including, simulated annealing (SA) algorithm, Tabu search (TS) algorithm and a hybrid algorithm of TS algorithm and SA algorithm called as TABUSA for the QAP and their efficiency are next evaluated in 59 instances of QAPLIB. In section 4, conclusions are explained and some suggestions for future research are described.

2. Research's directions of QAP

Almost all the researches in the field of QAP can be divided by three subjects: proposing efficient linearization for the problem (1), finding tight upper-bound and lower bound for optimal solution of the QAP. Here, we provide a short description of each direction as follows.

2.1. Linearization of the QAP

In this type of research, the goal is to find an efficient way to linearize the problem (1). The simplest linearization is

$$Min \sum_{q=1}^{n}\sum_{p=1}^{n}\sum_{j=1}^{n}\sum_{i=1}^{n} d_{pq} f_{ij} y_{ipjq} + \sum_{p=1}^{n}\sum_{i=1}^{n} C_{ip} x_{ip} \quad (2.1)$$

$$\sum_{i=1}^{n} x_{ip} = 1 \quad \forall p \quad (2.2)$$

$$\sum_{p=1}^{p} x_{ip} = 1 \quad \forall i \quad (2.3)$$

$$y_{ipjq} \le x_{ip} \quad \forall i,j,p,q \quad (2.4)$$

$$y_{ipjq} \le x_{jq} \quad \forall i,j,p,q \quad (2.5)$$

$$y_{ipjq} \ge x_{ip} + x_{jq} - 1 \quad \forall i,j,p,q \quad (2.6)$$

$$x_{ip} \in \{0,1\}, y_{ipjq} \in \{0,1\} \quad \forall i,j,p,q \quad (2.7)$$

It can be shown that it is not necessary to impose variable y_{ipjq} as binary variables and because of objective function, we can reduced constraints by removing (2.4) and (2.5). Therefore, the linearization of the QAP can be reformulated as

$$Min \sum_{q=1}^{n}\sum_{p=1}^{n}\sum_{j=1}^{n}\sum_{i=1}^{n} d_{pq} f_{ij} y_{ipjq} + \sum_{p=1}^{n}\sum_{i=1}^{n} C_{ip} x_{ip} \quad (3.1)$$

$$\sum_{i=1}^{n} x_{ip} = 1 \quad \forall p \quad (3.2)$$

$$\sum_{p=1}^{p} x_{ip} = 1 \quad \forall i \quad (3.3)$$

$$y_{ipjq} \ge x_{ip} + x_{jq} - 1 \quad \forall i,j,p,q \quad (3.4)$$

$$x_{ip} \in \{0,1\}, \quad (3.5)$$

Although formulation (3) reduces the computational time of formulation (2), however it is not the most efficient linearization of the QAP. It is probably because of this fact that the non-linearity is linearized based on inequality constraint. Adams and Jhonson (1994) proposed a linearization method that it is the most well-known method in the literature based on some equality constraints:

$$Min \sum_{q=1}^{n}\sum_{p=1}^{n}\sum_{j=1}^{n}\sum_{i=1}^{n} d_{pq} f_{ij} y_{ipjq} + \sum_{p=1}^{n}\sum_{i=1}^{n} C_{ip} x_{ip} \quad (4.1)$$

$$\sum_{i=1}^{n} x_{ip} = 1 \quad \forall p \quad (4.2)$$

$$\sum_{i=1}^{p} x_{ip} = 1 \quad \forall i \quad (4.3)$$

$$\sum_{q \ne p} y_{ipjq} = x_{ip} \quad \forall i \ne j, p \quad (4.4)$$

$$\sum_{j \ne i} y_{ipjq} = x_{ip} \quad \forall i, p \ne q \quad (4.5)$$

$$y_{ipjq} = y_{jqip} \quad \forall i \ne j, p \ne q \quad (4.6)$$

$$x_{ip} \in \{0,1\}, y_{ipjq} \in \{0,1\} \quad \forall i,j,p,q \quad (4.7)$$

Recently, Zhang et al. (2010) improved Adams and Johnson (1994) linearization as

$$Min \sum_{q=1}^{n}\sum_{p=1}^{n}\sum_{j=1}^{n}\sum_{i=1}^{n} d_{pq} f_{ij} y_{ipjq} + \sum_{p=1}^{n}\sum_{i=1}^{n} C_{ip} x_{ip} \quad (5.1)$$

$$\sum_{i=1}^{n} x_{ip} = 1 \quad \forall p \quad (5.2)$$

$$\sum_{i=1}^{p} x_{ip} = 1 \quad \forall i \quad (5.3)$$

$$\sum_{q \ne p} y_{ipjq} = x_{ip} \quad \forall i > j, p \quad (5.4)$$

$$\sum_{j > i} y_{ipjq} \le x_{ip} \quad \forall i, p \ne q \quad (5.5)$$

$$y_{ipjq} = y_{jqip} \quad \forall i \ne j, p \ne q \quad (5.6)$$

$$x_{ip} \in \{0,1\}, y_{ipjq} \in \{0,1\} \quad \forall i,j,p,q \quad (5.7)$$

Their linearization was also improved by removing zero flow variables. Here we report a brief numerical implementation as Table 1. comparing the efficiency of the formulations (3), (4) and (5) in 5 instances of QAPLIB as follows. It is shown that computational time of problem depends on structure of objective coefficients as well the size of problems. Moreover, it is shown that the formulation (5) is more efficient linearization than the other linearization.

Table 1. Computational time of different linearization				
	CPU Time			
Instance	(2)	(3)	(4)	(5)
Chr12a	>2hrs	28.67 sec	9.5 sec	6.29sec
Chr12b	>2hrs	6.9 sec	9.89 sec	1.09sec
Chr12c	>2hrs	47.02 sec	21.31 sec	14.15sec
Had12	>2hrs	>2hrs	>2hrs	1.5hrs
Nug12	>2hrs	>2hrs	>2hrs	>2hrs

2.2. Upper bound

Another direction of researches focused on QAP is to find near optimal solution for QAP. Several heuristics and meta-heuristics have been proposed and tested in set of instances existed in QAPLIB. QAPLIB provide several data sets containing 135 instances where best known solution (BKS) and lower bound of each instance are reported. Researchers have focused on improving the best known solution of each instance of QAPLIB as Drezner et al. (2013) or solving all instances of QAPLIB with the minimum gap respect to best known solution as James et al. (2009).

2.4.Lower bound

The last direction of QAP is to find a lower bound for optimal solution. The simplest way of finding lower bound is to sort element of matrix f from smallest to largest, $\hat{f}$, and sort element of matrix D from largest to smallest, $\hat{D}$. The value of $\hat{f}^T\hat{D}$ can be considered as a lower bound for optimal solution of QAP. Another lower bound was proposed by Gilmore (1962) as Gilmore-Lawler lower bound. They find the lowest cost of each facility in each location, $\underline{f}_{ip}$, and convert the QAP to assignment problem as

$$Min \sum_{p=1}^{n}\sum_{i=1}^{n}\underline{f}_{ip}x_{ip} + \sum_{p=1}^{n}\sum_{i=1}^{n}C_{ip}x_{ip} \quad (6.1)$$

$$\sum_{i=1}^{n}x_{ip} = 1 \quad \forall p \quad (6.2)$$

$$\sum_{i=1}^{p}x_{ip} = 1 \quad \forall i \quad (6.3)$$

Problem (6) can be solved in polynomial time with $O(n^4)$ using Hungarian method. Another lower bound can be found by LP- relaxation of problem (5), however the most efficient method proposed in the literature is based on semi-definite programming method (SDP). SDP relaxation can be applied for the QAP as

$$Min \sum_{q=1}^{n}\sum_{p=1}^{n}\sum_{j=1}^{n}\sum_{i=1}^{n}d_{pq}f_{ij}y_{ipjq} + \sum_{p=1}^{n}\sum_{i=1}^{n}C_{ip}x_{ip}$$

$$\sum_{i=1}^{n}x_{ip} = 1 \quad \forall p \quad (7.1)$$

$$\sum_{i=1}^{p}x_{ip} = 1 \quad \forall i \quad (7.2)$$

$$Q = \begin{bmatrix} y & x \\ x & 1 \end{bmatrix} \text{ is positive semi-definite} \quad (7.3)$$

where Q is $SDP \Leftrightarrow x, y \geq 0, x^2 \geq y$ and Q is a positive semi-definite matrix if $x^TQx \geq 0, \forall x \geq 0$.

3. Algorithms

In this section, four algorithms, that are, SA, TS and TABUSA are proposed for the QAP. The steps of each algorithm are explained as Figure 1. We compare the efficiency of all algorithms in 59 instances of QAPLIB shown in Table 2. The bold number shows that the algorithm finds BKS of instance and underline shows the best solution between four algorithms. It is shown that GA is the most efficient algorithm and SA is the worst algorithm. We next compare the best proposed algorithm, that is TABUSA, with four algorithms existed in the literature of the QAP, Ramkumar et al. (2008, 2009a&b) and Goldberg and Goldberg (2012). The solution, the gap (respect to BKS) and the computational time of each method are reported in each instance in Table 3.

We evaluate the efficiency of all algorithms in 59 instances (containing burxxx, elsxxx, escxxx, hadxxx, nugxxx, rouxxx, scrxxx, stexxx and thoxxx) of QAPLIB shown in Table 2. The bold number shows that the algorithm finds BKS of instance and underline shows the best solution between four algorithms. We next compare the best proposed algorithm, that is GA, with four algorithms existed in the literature of the QAP, Ramkumar et al. (2008, 2009a, 2009b) and Goldberg and Goldberg (2012) shown in Table 3. It is also shown in Table 4 that our proposed GA performs better than the other algorithm in all instances such that it can solve the most instances with zero gap (53 instances), it is also the best algorithm in terms of largest gap (1.56%) and dominates other algorithms in all instances.

```
Begin
  Input data and algorithm's parameters
      elmax = n × 10^4; %Total number of movement
      β = 0.9997; %cooling rate
      NEIT=n; %movement in each temperature
      T_in = 20; %Initial temperature
      K_0 = 10; %Coefficient of accepting unfavorable solution
      f_best = inf % Best value of the algorithm
  Generate initial solution randomly (c_main) and evaluate its efficiency (f_main).
  Consider the best solution, c_best := c_main.
  Set count = 1; move = 1; T = T_in
  While count ≤ elmax
      For move = 1 to NEIT
        If elmax − count ≥ 10^4 then replace location of two randomly selected facilities
        Elseif replace location of two pair of randomly selected facilities Endif
        Evaluate the efficiency of neighbor solution (f_local)
        Define Δ_f = f_local − f_main,
        If Δ_f < 0 then update c_main and f_main count := count − 1; T = β^-1 T,
        Elseif rand ≤ exp(−Del_f / f_main (T_0 − T)) then update c_main and f_main Endif.
        Update c_best, f_best, count := count + 1;
      Endfor
      T := βT.
  Endwhile
End.
```

Figure 1.a. SA Algorithm pseudo-code

```
Begin
  Input data and algorithm's parameters
      elmax = 3n × 10^3; %Total number of movement
      Tabulist = 2n; %Size of tabulist
      Tabu = []; %Initial Tabulist
      NEIT = n;  %movement before each local search
      f_best = inf % Best value of the algorithm
  Generate initial solution randomly (c_main) and evaluate its efficiency (f_main).
  Consider the best solution, c_best := c_main.
  Set count = 1; move = 1; Tabusize = 1; Tabu = Tabu ∪ c_main
  While count ≤ elmax
      For move = 1 to NEIT
        If elmax − count ≥ elmax/2 then replace location of two randomly selected facilities
        Elseif replace location of two pair of randomly selected facilities Endif
        If solution of neighbor, c_local ∉ Tabulist
          Tabu = Tabu ∪ c_main; Tabusize = Tabusize + 1;
          If Tabusize > Tabulist then remove half of solution of Tabulist, Tabusize = Tabulist/2 Endif
          If Δ_f < 0 then update c_main and f_main,
          Update c_best, f_best, count := count + 1;
      Endfor
      Implement localsearch %find the optimal permutation of three successive facilities
  Endwhile
```

End.

Figure 1.b. TS Algorithm pseudo-code

```
Begin
  Input data and algorithm's parameters
        elmax = n × 10^3; %Total number of movement
        Tabulist = 2n; %Size of tabulist
        Tabu = []; %Initial Tabulist
        NEIT = n;  %movement before each local search
        β = 0.9997; %cooling rate
        T_in = 20; %Initial temperature
        K_0 = 10; %Coefficient of accepting unfavorable solution
        f_best = inf % Best value of the algorithm
  Generate initial solution randomly (c_main) and evaluate its efficiency (f_main).
  Consider the best solution, c_best := c_main.
  Set count = 1; move = 1; Tabusize = 1; Tabu = Tabu ∪ c_main
  While count ≤ elmax
        For move = 1 to NEIT
          If elmax − count ≥ 10^3 then replace location of two randomly selected facilities
          Elseif replace location of two pair of randomly selected facilities Endif
          If solution of neighbor, c_local ∉ Tabulist
            Tabu = Tabu ∪ c_main; Tabusize = Tabusize + 1;
          If Δ_f < 0 then update c_main and f_main count := count − 1; T = β^-1 T,
          Elseif rand ≤ exp(−Del_f / f_main (T_0 − T)) then update c_main and f_main Endif.
          Update c_best, f_best, count := count + 1;
        Endfor
        Implement 2-opt local search
        T := βT.
  Endwhile
End.
```

Figure 1.c. TABUSA Algorithm pseudo-code

Table 4. Optimality gap comparison of algorithms

Interval	TABUSA	Algorithms of literature			
		2008	2009a	2009b	2012
zero gap	42	46	44	49	3
≤0.1	50	48	46	50	12
≤1	54	53	52	53	24
≤2	54	56	55	56	26
Dominating	73%	81%	75%	51	11%
Total instances	59	57	59	57	28
Largest gap	4.12	2.25	3.66	2.48	2.85

6. Conclusions

In this paper, three directions of research in the QAP are reviewed. We then developed four meta-heuristics, (SA, TS, TABUSA and GA) and compared with four existed algorithms in the literature (Ramkumar et al. 2008, 2009 a,b and Golberg and Golberg 2012). The numerical results show that our proposed GA performs is better than proposed algorithm in the literature. Finally, we explain some problems addition as future research's direction as follows:

- SFLP: This problem is a special case of the QAP where parameter d is defined as departmental distance. Proposed algorithms for the QAP can be applied easily for the SFLP.
- Dynamic facility layout problem (DFLP): DFLP is an extension of the QAP such that there are several periods with different departmental material flows and it's objective is to minimize material handling cost of all periods plus relayout cost of departments. Developed algorithms can be tested in a benchmark proposed by Balakrishnan et al. (2000).
- Traveling salesman problem (TSP): TSP is a special case of the QAP that is a special case of the QAP where facilities will be located over a circle.

Developed algorithms can be tested in TSP library benchmark proposed by Reinelt (2001).

- Sequencing problem: A set of jobs will be worked in several machines. There are several types of sequencing problem as single machine scheduling and parallel machine scheduling. In this problem, the solution can be represented as permutation encoding. There is a lot of instance for each type of this problem.
- Stochastic QAP: It is an interesting area to assume uncertain parameters in the QAP. In each instance of QAPLIB, we can consider an interval for each parameter where uncertain parameters deviated from their nominal value in these intervals.

Table 2. Comparison of proposed algorithm

			SA			TS			TABUSA		
n	Name	BKS	CPU time	Solution	RPD	CPU time	Solution	RPD	CPU time	Solution	RPD
(a) Solution qualities with CPU time of burxxx problems											
26	Bur26a	**5426670**	32.9	5547753	2.23	30.4	5434116	0.14	50.2	5433083	0.12
26	Bur26b	**3817852**	33.4	3904762	2.28	30.1	3824830	0.18	49.7	3818041	0.00
26	Bur26c	**5426795**	34.0	5532348	1.95	31.8	5428558	0.03	47.2	5427703	0.02
26	Bur26d	**3821225**	33.8	3902153	2.12	31.0	3831262	0.26	46.5	3821525	0.01
26	Bur26e	**5386879**	34.0	5509541	2.28	29.3	5401444	0.27	48.3	5387503	0.01
26	Bur26f	**3782044**	32.8	3881039	2.62	29.4	3794964	0.34	45.4	3782219	0.00
26	Bur26g	**10117172**	34.9	10314185	1.95	31.2	10120332	0.03	48.9	10119767	0.03
26	Bur26h	**7098658**	33.9	7234980	1.92	29.7	7099875	0.02	50.9	7098905	0.00
(b) Solution qualities with CPU time of elsxxx problems											
19	Els19	**17212548**	22.4	21868218	27.05	20.8	22324310	29.70	25.9	17260346	0.28
(c) Solution qualities with CPU time of escxxx problems											
16	esc16a	**68**	15.6	72	5.88	16.3	**68**	0.00	19.9	**68**	0.00
16	esc16b	**292**	13.7	**292**	0.00	17.6	**292**	0.00	20.2	**292**	0.00
16	esc16c	**160**	19.0	162	1.25	20.6	**160**	0.00	21.1	**160**	0.00
16	esc16d	**16**	15.5	18	12.50	22.4	**16**	0.00	19.4	**16**	0.00
16	esc16e	**28**	16.3	**28**	0.00	16.2	**28**	0.00	17.5	**28**	0.00
16	esc16f	**0**	10.7	**0**	0.00	23.2	**0**	0.00	33.9	**0**	0.00
16	esc16g	**26**	15.3	**26**	0.00	21.1	**26**	0.00	22.3	**26**	0.00
16	esc16h	**996**	17.1	**996**	0.00	21.4	**996**	0.00	17.2	**996**	0.00
16	esc16i	**14**	14.7	**14**	0.00	16.7	**14**	0.00	21.7	**14**	0.00
16	esc16j	**8**	13.8	**8**	0.00	16.8	**8**	0.00	18.1	**8**	0.00
32	esc32a	**130**	95.3	254	95.38	23.0	160	23.08	72.9	**130**	0.00
32	esc32b	**168**	90.4	320	90.48	38.2	196	16.67	79.2	**168**	0.00
32	esc32c	**642**	37.5	704	9.66	40.2	**642**	0.00	85.3	**642**	0.00
32	esc32d	**200**	38.7	244	22.00	42.0	214	7.00	78.1	**200**	0.00
32	esc32e	**2**	31.6	**2**	0.00	37.9	**2**	0.00	78.0	**2**	0.00
32	esc32g	**6**	33.1	**6**	0.00	41.4	**6**	0.00	81.2	**6**	0.00
32	esc32h	**438**	40.6	518	18.26	39.6	466	6.39	75.0	**438**	0.00
64	esc64a	**116**	85.7	154	32.76	124.2	**116**	0.00	445.0	**116**	0.00
128	esc128a	**64**	244.7	210	228.13	530.0	68	6.25	2187.0	**64**	0.00
(d) Solution qualities with CPU time of hadxxx problems											
12	Had12	**1652**	12.1	1676	1.45	13.6	1676	1.45	11.1	**1652**	0.00
14	Had14	**2724**	15.1	2780	2.06	14.7	**2724**	0.00	14.1	**2724**	0.00
16	Had16	**3720**	17.1	3820	2.69	15.6	**3720**	0.00	17.8	**3720**	0.00
18	Had18	**5358**	20.1	5522	3.06	18.9	5420	1.16	22.0	**5358**	0.00
20	Had20	**6922**	23.1	7158	3.41	20.2	7034	1.62	27.7	**6922**	0.00
(g) Solution qualities with CPU time of nugxxx problems											
12	nug12	**578**	11.9	600	3.81	13.4	586	1.38	11.6	**578**	0.00
14	nug14	**1014**	15.1	1084	6.90	15.1	1016	0.20	14.1	**1014**	0.00
15	nug15	**1150**	16.0	1222	6.26	15.7	1152	0.17	15.1	**1150**	0.00
16	nug16a	**1610**	17.3	1752	8.82	16.6	1634	1.49	16.9	**1610**	0.00
16	nug16b	**1240**	17.7	1388	11.94	17.4	1290	4.03	18.7	**1240**	0.00
17	nug17	**1732**	19.2	1884	8.78	18.4	1764	1.85	20.6	**1732**	0.00
18	nug18	**1930**	21.0	2126	10.16	20.6	1970	2.07	22.9	**1930**	0.00
20	nug20	**2570**	23.6	2864	11.44	22.3	2632	2.41	28.5	**2570**	0.00
(h) Solution qualities with CPU time of rouxxx problems											
12	rou12	**235528**	11.7	255982	8.68	12.7	240038	1.91	11.1	**235528**	0.00
15	rou15	**354210**	15.9	384298	8.49	16.3	363786	2.70	16.1	**354210**	0.00
20	rou20	**725522**	23.1	804298	10.86	20.5	729774	0.59	26.6	725582	0.01
(i) Solution qualities with CPU time of scrxxx problems											
12	scr12	**31410**	12.2	34462	9.72	12.7	**31410**	0.00	12.1	**31410**	0.00
15	scr15	**51140**	15.8	61808	20.86	15.0	54002	5.60	16.3	**51140**	0.00
20	scr20	**110030**	24.4	143882	30.77	22.5	116022	5.45	29.8	**110030**	0.00
(k) Solution qualities with CPU time of stexxx problems											
12	tai12a	**224416**	12.8	244438	8.92	13.8	235554	4.96	11.5	**224416**	0.00
15	tai15a	**388214**	16.6	419472	8.05	16.3	399756	2.97	17.0	**388214**	0.00
15	tai15b	**51765268**	16.7	52247647	0.93	16.2	51881760	0.23	16.5	**51765268**	0.00
17	tai17a	**491812**	19.0	546896	11.20	18.5	504676	2.62	20.6	**491812**	0.00
20	tai20a	**703482**	24.6	794058	12.88	21.9	731042	3.92	28.5	705622	0.30
20	tai20b	**122455319**	22.8	146972858	20.02	21.5	123190475	0.60	34.2	**122455319**	0.00
25	tai25a	**1167256**	32.3	1309950	12.22	28.4	1230588	5.43	44.8	1199284	2.74
30	tai30a	**1818146**	40.6	2034712	11.91	35.1	1902096	4.62	67.6	1867454	2.71
40	tai40a	**3139370**	65.0	3559698	13.39	57.3	3288442	4.75	130.7	3259350	3.82
50	tai50a	**4938796**	81.2	5643164	14.26	74.2	5139074	4.06	200.0	5142270	4.12
(m) Solution qualities with CPU time of thoxxx problems											
30	tho30	**149936**	45.8	184932	23.34	35.0	158644	5.81	64.3	150994	0.71
40	tho40	**240516**	76.1	295656	22.93	52.8	255248	6.13	121.0	245646	2.13

Table 3. Comparison of our best algorithm with algorithms in the literature of the QAP

			Ramkumar et al. (2008)			Ramkumar et al. (2009,a)			Ramkumar et al. (2009,b)		Goldberg & Goldberg (2012)		TABUSA		
n	Name	BKS	CPU time	Solution	RPD	CPU time	Solution	RPD	Solution	RPD	CPU Time	RPD	CPU time	Solution	RPD
(a) Solution qualities with CPU time of burxxx problems															
26	Bur26a	**5426670**	61.3	**5426670**	0.00	39.1	5431255	0.08	**5426670**	0.00	0.15	0.01	50.2	5433083	0.12
26	Bur26b	**3817852**	60.3	**3817852**	0.00	41.1	3824315	0.17	**3817852**	0.00			49.7	3818041	0.00
26	Bur26c	**5426795**	57.8	**5426795**	0.00	39.1	**5426795**	0.00	**5426795**	0.00			47.2	5427703	0.02
26	Bur26d	**3821225**	61.3	**3821225**	0.00	40.1	**3821225**	0.00	**3821225**	0.00			46.5	3821525	0.01
26	Bur26e	**5386879**	57.8	**5386879**	0.00	38.4	**5386879**	0.00	**5386879**	0.00			48.3	5387503	0.01
26	Bur26f	**3782044**	59.2	**3782044**	0.00	38.6	**3782044**	0.00	**3782044**	0.00			45.4	3782219	0.00
26	Bur26g	**10117172**	57.7	**10117172**	0.00	37.1	**10117172**	0.00	**10117172**	0.00			48.9	10119767	0.03
26	Bur26h	**7098658**	57.5	**7098658**	0.00	37.4	**7098658**	0.00	**7098658**	0.00			50.9	7098905	0.00
(b) Solution qualities with CPU time of elsxxx problems															
19	Els19	**17212548**	-	-	-	8.9	**17212548**	0.00	**17212548**	0.00	-	-	25.9	17260346	0.28
(c) Solution qualities with CPU time of escxxx problems															
16	esc16a	**68**	3.2	**68**	0.00	2.2	**68**	0.00	**68**	0.00	–	–	19.9	**68**	0.00
16	esc16b	**292**	2.8	**292**	0.00	1.9	**292**	0.00	**292**	0.00	–	–	20.2	**292**	0.00
16	esc16c	**160**	4.0	**160**	0.00	2.8	**160**	0.00	**160**	0.00	–	–	21.1	**160**	0.00
16	esc16d	**16**	4.0	**16**	0.00	2.8	**16**	0.00	**16**	0.00	–	–	19.4	**16**	0.00
16	esc16e	**28**	2.3	**28**	0.00	1.6	**28**	0.00	**28**	0.00	–	–	17.5	**28**	0.00
16	esc16f	**0**	1.1	**0**	0.00	0.8	**0**	0.00	**0**	0.00	–	–	33.9	**0**	0.00
16	esc16g	**26**	2.8	**26**	0.00	1.9	**26**	0.00	**26**	0.00	–	–	22.3	**26**	0.00
16	esc16h	**996**	2.1	**996**	0.00	1.5	**996**	0.00	**996**	0.00	–	–	17.2	**996**	0.00
16	esc16i	**14**	2.0	**14**	0.00	1.4	**14**	0.00	**14**	0.00	–	–	21.7	**14**	0.00
16	esc16j	**8**	2.9	**8**	0.00	2.0	**8**	0.00	**8**	0.00	–	–	18.1	**8**	0.00
32	esc32a	**130**	136.8	**130**	0.00	89.5	134	3.08	**130**	0.00	5	0.57	72.9	**130**	0.00

32	esc32b	**168**		110.4	**168**	0.00		72.9	**168**	0.00		**168**	0.00					79.2	**168**	0.00
32	esc32c	**642**		54.7	**642**	0.00		34.6	**642**	0.00		**642**	0.00					85.3	**642**	0.00
32	esc32d	**200**		74.3	**200**	0.00		49.0	**200**	0.00		**200**	0.00					78.1	**200**	0.00
32	esc32e	**2**		46.1	**2**	0.00		29.4	**2**	0.00		**2**	0.00					78.0	**2**	0.00
32	esc32g	**6**		28.4	**6**	0.00		18.9	**6**	0.00		**6**	0.00					81.2	**6**	0.00
32	esc32h	**438**		85.8	**438**	0.00		54.9	**438**	0.00		**438**	0.00					75.0	**438**	0.00
64	esc64a	**116**		1521.7	**116**	0.00		1059.9	**116**	0.00		**116**	0.00		15	**0.00**		445.0	**116**	0.00
128	esc128a	**64**		-	-	-		23370.1	**64**	0.00		**64**	0.00		20	0.88		2187.0	**64**	0.00
(d) Solution qualities with CPU time of hadxxx problems									-			-							-	
12	Had12	**1652**		1.0	**1652**	0.00		1.0	**1652**	0.00		**1652**	0.00		–	–		11.1	**1652**	0.00
14	Had14	**2724**		2.0	**2724**	0.00		2.0	2744	0.73		**2724**	0.00		–	–		14.1	**2724**	0.00
16	Had16	**3720**		3.6	3722	0.05		3.6	3722	0.05		**3720**	0.00		–	–		17.8	**3720**	0.00
18	Had18	**5358**		6.5	**5358**	0.00		6.4	**5358**	0.00		**5358**	0.00		–	–		22.0	**5358**	0.00
20	Had20	**6922**		10.6	**6922**	0.00		10.2	**6922**	0.00		**6922**	0.00		3.0	**0.00**		27.7	**6922**	0.00
(g) Solution qualities with CPU time of nugxxx problems																				
12	nug12	**578**		1.4	**578**	0.00		1.0	**578**	0.00		**578**	0.00		–	–		11.6	**578**	0.00
14	nug14	**1014**		3.1	1018	0.39		2.1	**1014**	0.00		1016	0.20		–	–		14.1	**1014**	0.00
15	nug15	**1150**		4.0	**1150**	0.00		2.9	**1150**	0.00		**1150**	0.00		–	–		15.1	**1150**	0.00
16	nug16a	**1610**		5.6	**1610**	0.00		3.9	**1610**	0.00		**1610**	0.00		–	–		16.9	**1610**	0.00
16	nug16b	**1240**		5.6	**1240**	0.00		3.9	**1240**	0.00		**1240**	0.00		–	–		18.7	**1240**	0.00
17	nug17	**1732**		7.3	**1732**	0.00		5.2	1734	0.12		**1732**	0.00		–	–		20.6	**1732**	0.00
18	nug18	**1930**		9.6	**1930**	0.00		6.7	**1930**	0.00		**1930**	0.00		–	–		22.9	**1930**	0.00
20	nug20	**2570**		16.1	**2570**	0.00		11.2	**2570**	0.00		**2570**	0.00		3.0	**0.00**		28.5	**2570**	0.00
(h) Solution qualities with CPU time of rouxxx problems									-			-							-	
12	rou12	**235528**		1.1	**235528**	0.00		1.1	**235528**	0.00		**235528**	0.00		–	–		11.1	**235528**	0.00
15	rou15	**354210**		3.0	**354210**	0.00		2.9	**354210**	0.00		**354210**	0.00		–	–		16.1	**354210**	0.00
20	rou20	**725522**		11.7	725662	0.02		11.4	**725522**	0.00		**725522**	0.00		3.0	0.16		26.6	725582	0.01
(i) Solution qualities with CPU time of scrxxx problems									-			-							-	
12	scr12	**31410**		1.1	**31410**	0.00		11.1	**31410**	0.00		**31410**	0.00		–	–		12.1	**31410**	0.00
15	scr15	**51140**		3.1	**51140**	0.00		3.1	**51140**	0.00		**51140**	0.00		–	–		16.3	**51140**	0.00
20	scr20	**110030**		12.7	**110030**	0.00		12.3	**110030**	0.00		**110030**	0.00		3.0	0.01		29.8	**110030**	0.00
(k) Solution qualities with CPU time of stexxx problems									-			-							-	
12	tai12a	**224416**		1.1	**224416**	0.00		1.1	**224416**	0.00		**224416**	0.00		-	-		11.5	**224416**	0.00
15	tai15a	**388214**		3.0	**388214**	0.00		3.0	388870	0.17		388250	0.01		-	-		17.0	**388214**	0.00
15	tai15b	**51765268**		3.1	**51765268**	0.00		3.1	**51765268**	0.00		**51765268**	0.00		-	-		16.5	**51765268**	0.00
17	tai17a	**491812**		5.6	493662	0.38		5.5	**491812**	0.00		494550	0.56		-	-		20.6	**491812**	0.00
20	tai20a	**703482**		11.4	706786	0.47		1.4	729204	3.66		709080	0.80		3.0	0.52		28.5	705622	0.30
20	tai20b	**122455319**		12.5	**122455319**	0.00		12.4	125286585	2.31		**122455319**	0.00		-	-		34.2	**122455319**	0.00
25	tai25a	**1167256**		33.0	1190574	2.00		32.9	1181072	1.18		1185587	1.57		6.0	1.29		44.8	1199284	2.74
30	tai30a	**1818146**		83.1	1838380	1.11		82.8	1845582	1.51		1842986	1.37		10.0	1.52		67.6	1867454	2.71
40	tai40a	**3139370**		354.4	3197384	1.85		346.9	3198114	1.87		3192668	1.70		25.0	2.22		130.7	3259350	3.82
50	tai50a	**4938796**		1104.1	5049794	2.25		1076.1	5043946	2.13		5061236	2.48		50.0	2.85		200.0	5142270	4.12
(m) Solution qualities with CPU time of thoxxx problems									-			-	-						-	
30	tho30	**149936**		118.9	150378	0.29		78.1	150454	0.35		-	-		10.0	0.21		64.3	150994	0.71
40	tho40	**240516**		501.8	241782	0.53		351.3	242888	0.99		-	-		25.0	0.45		121.0	245646	2.13

Corresponding Author:
Hossein Shahbazi
Department of Industrial Engineering
Azad University, South Tehran Branch,
Tehran, Iran
E-mail: shahbazi.hossein1368@gmail.com

References

1. Adams, WARREN P, and TERRI A Johnson. "Improved Linear Programming-Based Lower Bounds for the Quadratic Assignment Problem." DIMACS series in discrete mathematics and theoretical computer science 16 (1994): 43-77.
2. Balakrishnan, Jaydeep. "Dplp Balakrishnan Dataset". http://dspace.ucalgary.ca/handle/1880/46715.
3. R.E. Burkard, E. Çela, P.M. Pardalos And L. Pitsoulis. The quadratic assignment problem. In P.P. Pardalos and M.G.C. Resende, editors, Handbook of Combinatorial Optimization, 1998. Kluwer Academic Publishers, Dordrecht, pp. 241-238.
4. Drezner, Zvi, and Alfonsas MisevičIus. "Enhancing the Performance of Hybrid Genetic Algorithms by Differential Improvement." Computers & Operations Research (2012).
5. Goldbarg, Elizabeth Ferreira Gouvêa, and Marco Cesar Goldbarg. "An Experimental Study of Variable Depth Search Algorithms for the Quadratic Assignment Problem." Pesquisa Operacional 32 (2012): 165-96.
6. Koopmans, Tjalling C, and Martin Beckmann. "Assignment Problems and the Location of Economic Activities." Econometrica: Journal of the Econometric Society (1957): 53-76.
7. Loiola, E. M., N. M. M. de Abreu, P. O. Boaventura-Netto, P. Hahn, and T. Querido. "A Survey for the Quadratic Assignment Problem." [In English]. European Journal of Operational Research 176, no. 2 (Jan 16 2007): 657-90.
8. Ramkumar, A. S., S. G. Ponnambalam, and N. Jawahar. "A Population-Based Hybrid Ant System for Quadratic Assignment Formulations in Facility Layout Design". International Journal of Advanced Manufacturing Technology 44, no. 5-6 (Sep 2009): 548-58.
9. Ramkumar, A. S., S. G. Ponnambalam, N. Jawahar, and R. K. Suresh. "Iterated Fast Local Search Algorithm for Solving Quadratic Assignment Problems." [In English]. Robotics and Computer-Integrated Manufacturing 24, no. 3 (Jun 2008): 392-401.
10. Reinelt, Gerhard. "Tsplib—a Traveling Salesman Problem Library." ORSA journal on computing 3, no. 4 (1991): 376-84.
11. Sahni, Sartaj, and Teofilo Gonzalez. "P-Complete Approximation Problems." Journal of the ACM (JACM) 23, no. 3 (1976): 555-65.

12. Zhang, Huizhen, Cesar Beltran-Royo, and Miguel Constantino. "Effective Formulation Reductions for the Quadratic Assignment Problem." Computers & Operations Research 37, no. 11 (2010): 2007-16.

5/29/2013