Extension of PSO and ACO-PSO algorithms for solving Quadratic Assignment Problems

K. Rameshkumar

1Department of Mechanical Engineering, Amrita University, Bangalore, India.

Abstract. In this paper, PSO and Ant Colony Optimization inspired PSO (ACO-PSO) algorithms were adopted to solve the Quadratic Assignment Problems. A hybrid approach is adopted in this paper by combining assignment construction with local-search. In the PSO algorithm, solution construction has been carried out by assigning weights to current, particle’s best and global best solutions associated with assignment of resources. Velocities which are used to construct the assignments in this approach are similar to the trail intensities considered in the ant colony algorithms. The proposed algorithms have been applied to a set of benchmark problems and the performance of the algorithm is evaluated by testing the obtained results with the results published in the literature. The computational results show that good quality solutions are obtained using the PSO and ACO inspired PSO algorithm.

1. Introduction

The quadratic assignment problem (QAP) is an ‘NP-hard’ optimization problem [1] and it is one of the most hard optimization problems to resolve optimally. The QAP can best be depicted as the problem of assigning items to locations. The distances between the locations and flows between the items will be provided as an input to the algorithm to find the optimal or near optimal assignment. The goal in QAP is to position the items in the locations such a way that the sum of the product between flows and distances is minimal. Many practical problems like blackboard wiring [2], campus and hospital layout [3], typewriter keyboard design [4], scheduling [1] and many others [5-7] formulated as QAP’s.

For the given ‘n’ items and ‘n’ locations, two ‘n × n’ matrices ‘A’ and ‘B’ are given as an input to the algorithm. Distance between the locations ‘i’ and ‘j’ are indicated by ‘a_{ij}’. The flow between items ‘r’ and ‘s’ is ‘b_{rs}’. In QAP items are assigned to locations such that every item is assigned exactly at one location. It is also considered that no location is assigned more than one item. In QAP, the number of items, ‘n’ is the same as the number of locations ‘n’. A typical assignment in QAP corresponds to a permutation of the integers from ‘1’ to ‘n’. The objective for the QAP is formulated as an optimization function as shown in the equation (1).

\[
\min_{\phi \in S_n} \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ij} a_{\phi_i \phi_j} \tag{1}
\]

It is considered that ‘S_n’ is the set of all permutations of \{1, 2, n\}, and ‘F_i’ gives the location of item ‘i’ in a solution ‘\phi \in S_n’. In practice, the only feasible way to resolve large QAP instances is to apply heuristic algorithms. In general, heuristic algorithms find good quality solutions in short computational time. Theory, algorithms and heuristic approaches to solve QAP is presented in [8]. The heuristic approaches include local search algorithms like iterative improvement and stochastic local search [9-12] simulated annealing [13-15], Tabu search [16-19], Genetic algorithms [20-22], and Ant Colony Optimization algorithms [23-26] were proposed over the year to solve the QAPs. Off late, particle
Swarm optimization algorithms have been applied to a variety of optimization problems in including continuous function optimization, sequencing and scheduling and QAP [27-29]. In this paper, two approaches were used to solve the QAP instances available in the literature. First, a discrete PSO algorithm which was used to solve discrete optimization problem for scheduling [30-31] is adapted to solve QAP. Next, in the Ant inspired PSO, assignment construction procedure employed in this study is based on procedures developed by [32-35].

2. Structure of the proposed PSO and ACO based PSO algorithm
For the QAP, permutation that yields minimum cost for allocating items to the various locations is to be determined. A sequence is a permutation order of the items assigned to various locations. A sequence is considered as particle’s location in the ‘d’ dimensional space. A particle ‘k’ in PSO is considered as a sequence of ‘i’ items. In the QAP problems no. of items (‘i’) assigned are equal to the number of locations (‘j’). The structure of the PSO and ACO based PSO algorithm is described as follows:

**Step1:** Generate the initial solution \([X_{kd}]\) randomly according to the swarm size.
**Step2:** Apply improvement schemes for the initial population
**Step3:** For each particle ‘k’ compute the cost value of the assignment
**Step4:** Initialize \([P_{kd}]\) and \([G_d]\); \([P_{kd}]\) is the particle’s best assignment which provide minimum assignment cost and \([G_d]\) is the overall global best assignment among all the particles in the swarm which provides over all minimum assignment cost.
**Step5:** While (until the termination criterion is met with)
   Do (for all the particle ‘k’)
   {  
   Calculate velocity of the particle
   Construct new solution (PSO / ACO-PSO approach)
   Improve the solution using improvement schemes
   }
   **Step6:** Update \([P_{kd}]\) and \([G_d]\) and evaluate the assignment cost
   **Step7:** Return the assignment and minimum assignment cost

3. Assignment initialization of PSO and ACO-PSO algorithm
   Initial assignments were generated randomly as per the size of the swarm. In this work, swarm size of ‘5’ has been chosen. It means ‘5’ solutions are generated randomly irrespective of the size of the problem. The initial solutions were subjected to improvement schemes namely ‘adjacent-pair wise-interchange’ [36], ‘job index based insertion scheme’, ‘job index based swap scheme’ [37] sequentially. These schemes will improve the solution quality there by convergence of the algorithm is improved. From the improved assignment, for each particle current location \([X_{kd}]\) and particle best location \([P_{kd}]\) is identified. For the first iteration it is set that \([X_{kd}] = [P_{kd}]\). The best particle among all the particles in the swarm is also identifies and called as \([G_d]\). Using \([X_{kd}], [P_{kd}]\) and \([G_d]\) new solutions are constructed in the subsequent iterations. It is to be noted that in every iteration swarm size of 20 is maintained. To avoid premature convergence, in every iteration, assignments in \([X_{kd}], [P_{kd}]\) and \([G_d]\) is verified. If \([X_{kd}] = [P_{kd}] = [G_d]\), generate one solution randomly and improvement schemes are applied. The generated assignment is called as \([X’_{kd}]\). Compare the solution quality of \([X’_{kd}]\) with the other sequence. The better one will be called as \([P_{kd}]\) and the rest will be \([X_{kd}]\). Update the \([G_d]\) if applicable by comparing with \([X_{kd}]\) and \([P_{kd}]\).

4. Assignment construction in PSOA
   A new position of a particle ‘k’ is constructed from the existing location of the particle, particle’s best location ever reached and best location of any particle in the swarm. This procedure used in this study is adopted from the solution construction procedure employed in the shop scheduling problems [30]. Location means that the position of the particle indicating its objective functions value. Weights \(w_1, w_2\) and \(w_3\) are generated in such a way that \(w_1 + w_2 + w_3 = 1\) . The weights, \(w_1\), \(w_2\) and \(w_3\) play a role similar to the constriction coefficients ‘c1’ and ‘c2’ used in the generic PSO velocity updating
equation. The velocity and position updating equations are given in the equation (2) and (3). \( V_{kd} \) is the velocity obtained in the previous iteration; \( u_1 \) and \( u_2 \) are the two uniformly distributed random numbers. \( X_{kd} \) is the current location of the particle, \( P_{kd} \) is the best location ever reached by the particle and \( G_d \) is the best location reached by any of the particle in the swarm. In every iteration \( i \) the particle \( k \) travels in the solution space of dimension, \( d \). This velocity definition is more applicable for solving continuous functions. The problem considered in this study is of discrete in nature. The velocity construction procedure is modified to suit the discrete problem considered in this study.

\[
\text{New Velocity, } V_{kd} = V_{kd} + c_1 \times \left[ u_1 \times (P_{kd} - X_{kd}) \right] + c_2 \times \left[ u_2 \times (G_d - X_{kd}) \right] \quad \text{(2)}
\]

\[
\text{New location, } X_{kd}^{new} = X_{kd}^{new} + V_{kd} \quad \text{(3)}
\]

In this approach, a uniformly distributed random number \( u \) in the range ‘0’ and ‘1’ is generated and compared with the weights assigned to the \([X_{kd}], [P_{kd}]\) and \([G_d]\) for selecting a job in the sequence. If the sampled random number; \( u \leq w_1 \), first unassigned item in the \( X_{kd} \) is chosen and updated in \( X_{kd}^{new} \). If the sampled random number; \( 0.2 < u \leq 0.5 \), first unassigned item from the \( P_{kd} \) is chosen and updated in \( X_{kd}^{new} \). If the random number; \( 0.5 < u \leq 1 \), the first unassigned item from \( G_d \) is chosen and updated in \( X_{kd}^{new} \). This procedure is repeated until all the items are assigned to \( X_{kd}^{new} \). For the newly constructed assignment, improvement schemes are applied and the resultant assignment is called \([X_{kd}], [P_{kd}]\) be the current location and corresponding \( P_{kd} \) and \( G_d \) locations are updated. The procedure continues until the termination criterion is met with.

The weights ‘\( w_1 \)’, ‘\( w_2 \)’ and ‘\( w_3 \)’ used in this study are 0.2, 0.3 and 0.5 respectively. For example, \([X_{kd}] = [4 \ 1 \ 3 \ 5 \ 2]; \ [P_{kd}] = [5 \ 3 \ 2 \ 1 \ 4]; \ [G_d] = [3 \ 5 \ 4 \ 1 \ 2]; \ w_1=0.2; \ w_2=0.3; \ w_3=0.5, \ r_1=0.34; \ r_2=0.78; \ r_3=0.12; \ r_4=0.02; \) and \( r_5=0.97. \) The \([X_{kd}^{new}] = [5 \ 3 \ 4 \ 1 \ 2]. \) The solution represents the order of allocation of items ‘i’ to locations ‘n’. It means 5th item is allocated to location 1, 3rd item is assigned to location 2, 4th item is allocated to location 3, 1st item is located to location 4 and item 2 is located to the location 5. The solution construction methodology is shown in Figure 1. The corresponding cost function is calculated for every solution generated during the iterative process.

![Figure 1. Solution construction](image)

5. Assignment construction in ACO based PSOA

5.1 Velocity calculation in ACO based PSOA

Velocity and assignment construction procedure employed in this study is based on procedures developed in [32-35]. In this proposed procedure, velocities are computed for updating the particles from one position to other by calculating trail intensities. Trial intensities computed are similar to velocities used in PSO algorithm.

In the beginning, velocities trails are updated using the equation, \( V_{ij}^{new} = w_1 \times V_{ij}^e + w_2 \times V_{ij}^p + w_3 \times V_{ij}^g. \)

Weights, ‘\( w_1 \)’, ‘\( w_2 \)’ and ‘\( w_3 \)’ are relative importance given to the \([X_{kd}], [P_{kd}]\) and \([G_d]. \) For all ‘i’ (item), ‘j’ (location) and ‘k’ (particles), velocity components are set as shown in the equations (4), (5) and (6).
\[
V_{ij}^s = \frac{1}{\left(\text{Position of item } 'i' \text{ in the assignment } [X_{kd}] - j + 1\right)^{4/3}} \quad \ldots (4)
\]
\[
V_{ij}^p = \frac{1}{\left(\text{Position of item } 'i' \text{ in the assignment } [P_{kd}] - j + 1\right)^{4/3}} \quad \ldots (5)
\]
\[
V_{ij}^g = \frac{1}{\left(\text{Position of item } 'i' \text{ in the assignment } [G_{kd}] - j + 1\right)^{4/3}} \quad \ldots (6)
\]

The suitable values of weights used in this study are 'w_1'=0.2, 'w_2'=0.3 and 'w_3'=0.5. The coefficients, 'c_1'=3; 'c_2'=2; 'c_1'=3 are arrived based on pilot studies.

5.2 Assignment construction in the ACO based PSO algorithm
Assignment of an item (i) to a location (j) is based on current location [X_{kd}] and velocity trail, 'V_{ij}^{new}'. Velocity trail is analogous to trail-intensity used in ant algorithm, which indicates desire of assigning an item 'i' in the location 'j' of the assignment, where 'j'=1,2,...D and 'i'=1,2,...D. Total no. of locations / items is indicated by 'D'. For assigning items, a parameter called '\gamma_{ij}' is set, which indicates sum of velocities of swarm particles for item, 'i' up to the location 'j'.

\[
\gamma_{ij} = \sum_{k=1}^{D} V_{ij}^{new} \quad \ldots (7)
\]

For each 'i', a random number 'r' is generated. If 'r' is less than or equal to 0.4, an unassigned item from the [G_{kd}] is assigned to [s]. If the sampled 'r' of between 0.4 and 0.8, the unassigned item with maximum value of 'r_{ij}' is assigned to [s]. If the sampled 'r' is above 0.8, an item is assigned probabilistically to [s]. The probability of assigning item 'i' in the location 'j' is given by

\[
P_{ij} = \frac{\gamma_{ij}}{\sum_{i \text{ unscheduled } \text{jobs in } [X_{kd}]} \gamma_{ij}} \quad \ldots (8)
\]

Construction of an assignment is shown with the following example:

Locations (j)
\[
V_{ij}^{new} = \begin{pmatrix}
0.4 & 0.2 & 0.6 \\
0.3 & 0.2 & 0.7 \\
0.5 & 0.3 & 0.1
\end{pmatrix}
\]

Items (i)

Let [X_k]=[3 1 2] and [G_k]=[2 1 3] and Set \[ \gamma_{ij} = \sum_{k=1}^{D} V_{ij}^{new} \]

Location (j)
\[
\gamma_{ij} = \begin{pmatrix}
0.4 & 0.6 & 1.2 \\
0.3 & 0.5 & 1.2 \\
0.5 & 0.8 & 0.9
\end{pmatrix}
\]

Item (i)

Assignment of items starts with a null set, [\emptyset]. Initially set [s] = [\emptyset]. Three uniformly distributed random numbers be 0.35, 0.96, and 0.71. For assigning an item in the first position, j=1; r \leq 0.4; select unassigned job from [G_{kd}]. Select item '2' from [G_k] and append to [s]; i.e. [s]=[2]. For the second position, j=2; the sampled random number is between 0.8 and 1.0; select the unassigned item probabilistically. The probability, 'P_{ij}' of placing a item 'i' in the location 'j' is computed, i.e, 'P_{12}'=0.375 and 'P_{22}'=0.571. For assigning an item for position '2', a uniformly distributed random number 'r' is generated. If r \leq P_{12}, assign item '1' to position '2'. Otherwise select item '3' to position '2'. In this case, the generated 'r' = 0.96. Since r \geq P_{12}, assign item '3' to the location '2' and update [s]. The updated assignment [s]=[2 3]. For the third position, the sampled random number is 0.71,
which is between 0.4 and 0.8. The position ‘3’ is selected based on the unassigned item with maximum value of \( c_{ij} \). The unassigned item is ‘1’. The updated assignment \([s]\) is \([2 \ 3 \ 1]\). The assignment is subjected to improvement scheme and the resultant is \([X_{i\ell}]\). Assignment cost is evaluated for all the particles. \([P_{d\ell}]\) and \([G_{d\ell}]\) is updated and iteration continues.

As like ant colony optimization algorithm, velocity trails are updated at the end of every iteration. For every particle in the swarm, \( V^k_{\ell} \) is set as \( \rho \times V^k_{\ell} \). The \( \rho \) value considered in this study is 0.75 for all iterations. The velocity is updated using the equation

\[
V^k_{\ell} = V^k_{\ell} + \frac{1}{|\text{Position of item 'i' in the assignment }[X_{i\ell}] - j + 1|} \quad \cdots \quad (8)
\]

6. Performance analysis

6.1 Benchmark problems

To evaluate the PSO and ACO-PSO, QAP instances available in the QAPLIB [38-39] has been utilised. The updated optimum / best known solutions for well-known QAP instances are available at https://www.opt.math.tugraz.at/qaplib/. Eight problem instances of Bur26X, 15 instances of NugXX, 20 instances of EscXXX, 13 instances of SkoXX, 14 instances of ChrXXX, 3 instances of KraXXX, 5 instances of HadXXX, 3 instances of RouXX, 3 instances of ScrXX, 9 instances of TaiXX problems and 2 instances of SteXXX were solved to evaluate the performances of the algorithms. A total of 95 problems of size varying from 12 to 100 were solved to evaluate the performance of the algorithm.

6.2 Results and discussions

The performances of algorithms are evaluated using the best-known upper bound values (https://www.opt.math.tugraz.at/qaplib/) . PSO and ACO-PSO algorithms were allowed to run for a maximum of 2000 × n² functional evaluations by fixing the swarm size as 5. The proposed algorithms were coded in C++ language and are implemented in Intel centrino 2 processor running at 2.0 GHz processor with 4.0 GB RAM. The algorithms were allowed to run only once. Taillard random number generator is used to generate the unbiased uniformly distributed random number. The same seed is used in computing the objective function value for all QAP instances considered. The assignment cost yielded by the PSO and ACO-PSO algorithms and the relative percentage deviation over the best known values for each of the benchmark instances are shown in Table 1. The relative percentage increase in assignment cost over the best-known values is computed using the equation (9).

\[
\text{Relative % increase in assignment cost} = \frac{((\text{Solution: PSO / ACO - PSO}) - BKS)}{((\text{Solution: PSO / ACO}) - \text{PSO})} \times 100 \cdots (9)
\]

The results shows that out of 95 problem instances considered, PSO produces optimum / best known solutions for 61 problem instances and ACO-PSO algorithm produces optimum / best-known solutions for 57 problem instances.

Table 1. Performance evaluation of PSO and ACO-PSO algorithm for BurXXX problems

| Prob. No. | Problem instances | Problem size | Best known cost PSO | ACO-PSO | Relative percentage increase in cost PSO | Relative percentage increase in cost ACO-PSO |
|----------|------------------|--------------|---------------------|---------|-----------------------------------------|------------------------------------------|
| 1        | Bur26a           | 26           | 5426670             | 5426670 | 0.00                                    | 0.00                                     |
| 2        | Bur26b           | 26           | 3817852             | 3817852 | 0.00                                    | 0.00                                     |
| 3        | Bur26c           | 26           | 5426795             | 5426795 | 0.00                                    | 0.00                                     |
| 4        | Bur26d           | 26           | 3821225             | 3821225 | 0.00                                    | 0.00                                     |
| 5        | Bur26e           | 26           | 5386879             | 5386879 | 0.00                                    | 0.00                                     |
| 6        | Bur26f           | 26           | 10117172            | 10262690| 1.42                                    | 2.01                                     |
| 7        | Bur26g           | 26           | 7098658             | 7098658 | 0.00                                    | 0.00                                     |
Table 2. Performance evaluation of PSO and ACO-PSO algorithm for NugXXX problems

| Prob. No. | Problem instances | Problem size | Best known cost | PSO | ACO-PSO | Relative percentage increase in cost |
|-----------|------------------|--------------|-----------------|-----|---------|-------------------------------------|
| 1         | Nug12            | 12           | 578             | 578 | 578     | 0.00                                 |
| 2         | Nug14            | 14           | 1014            | 1014| 1014    | 0.00                                 |
| 3         | Nug15            | 15           | 1150            | 1150| 1150    | 0.00                                 |
| 4         | Nug16a           | 16           | 1610            | 1610| 1610    | 0.00                                 |
| 5         | Nug16b           | 16           | 1240            | 1240| 1240    | 0.00                                 |
| 6         | Nug17            | 17           | 1732            | 1732| 1732    | 0.00                                 |
| 7         | Nug18            | 18           | 1930            | 1930| 1930    | 0.00                                 |
| 8         | Nug20            | 20           | 2570            | 2570| 2570    | 0.00                                 |
| 9         | Nug21            | 21           | 2438            | 2438| 2438    | 0.00                                 |
| 10        | Nug22            | 22           | 3596            | 3596| 3596    | 0.00                                 |
| 11        | Nug24            | 24           | 3488            | 3488| 3488    | 0.00                                 |
| 12        | Nug25            | 25           | 3744            | 3744| 3744    | 0.00                                 |
| 13        | Nug27            | 27           | 5234            | 5234| 5234    | 0.00                                 |
| 14        | Nug28            | 28           | 5166            | 5166| 5170    | 0.08                                 |
| 15        | Nug30            | 30           | 6124            | 6124| 6148    | 0.16                                 |

Table 3. Performance evaluation of PSO and ACO-PSO algorithm for EscXXX problems

| Prob. No. | Problem instances | Problem size | Best known cost | PSO | ACO-PSO | Relative percentage increase in cost |
|-----------|------------------|--------------|-----------------|-----|---------|-------------------------------------|
| 1         | Esc16a           | 16           | 68              | 68  | 68      | 0.00                                 |
| 2         | Esc16b           | 16           | 292             | 292 | 292     | 0.00                                 |
| 3         | Esc16c           | 16           | 160             | 160 | 160     | 0.00                                 |
| 4         | Esc16d           | 16           | 16              | 16  | 16      | 0.00                                 |
| 5         | Esc16e           | 16           | 28              | 28  | 28      | 0.00                                 |
| 6         | Esc16f           | 16           | 0               | 0   | 0       | 0.00                                 |
| 7         | Esc16g           | 16           | 26              | 26  | 26      | 0.00                                 |
| 8         | Esc16h           | 16           | 996             | 996 | 996     | 0.00                                 |
| 9         | Esc16i           | 16           | 14              | 14  | 14      | 0.00                                 |
| 10        | Esc16j           | 16           | 8               | 8   | 8       | 0.00                                 |
| 11        | Esc32a           | 32           | 130             | 130 | 130     | 0.00                                 |
| 12        | Esc32b           | 32           | 168             | 168 | 168     | 0.00                                 |
| 13        | Esc32c           | 32           | 642             | 642 | 642     | 0.00                                 |
| 14        | Esc32d           | 32           | 200             | 200 | 200     | 0.00                                 |
| 15        | Esc32e           | 32           | 2               | 2   | 2       | 0.00                                 |
| 16        | Esc32f           | 32           | 2               | 2   | 2       | 0.00                                 |
| 17        | Esc32g           | 32           | 6               | 6   | 6       | 0.00                                 |
| 18        | Esc32h           | 32           | 438             | 438 | 438     | 0.00                                 |
| 19        | Esc64a           | 64           | 116             | 116 | 116     | 0.00                                 |
| 20        | Esc128           | 128          | 64              | 64  | 64      | 0.00                                 |

Table 4. Performance evaluation of PSO and ACO-PSO algorithm for SteXXX problems

| Prob. No. | Problem instances | Problem size | Best known cost | PSO | ACO-PSO | Relative percentage increase in cost |
|-----------|------------------|--------------|-----------------|-----|---------|-------------------------------------|
| 1         | Ste36a           | 36           | 9526            | 9526| 9526    | 0.00                                 |
| 2         | Ste36b           | 36           | 8653            | 8653| 8653    | 0.00                                 |
Table 5. Performance evaluation of PSO and ACO-PSO algorithm for SkoXXX problems

| Prob. No. | Problem instances | Problem size | Best known cost | PSO Best cost | ACO-PSO Best cost | Relative percentage increase in cost |
|-----------|-------------------|--------------|-----------------|---------------|-------------------|-------------------------------------|
| 1         | Sko42             | 42           | 15812           | 15944         | 15836             | 0.83                                |
| 2         | Sko49             | 49           | 23386           | 23530         | 23514             | 0.61                                |
| 3         | Sko56             | 56           | 34458           | 34924         | 34554             | 1.33                                |
| 4         | Sko64             | 64           | 48498           | 48928         | 48684             | 0.88                                |
| 5         | Sko72             | 72           | 66256           | 66934         | 66740             | 1.01                                |
| 6         | Sko81             | 81           | 90998           | 91830         | 91556             | 0.91                                |
| 7         | Sko90             | 90           | 115534          | 116898        | 116302            | 1.17                                |
| 8         | Sko100a           | 100          | 152002          | 153822        | 153254            | 1.18                                |
| 9         | Sko100b           | 100          | 153890          | 155956        | 155496            | 1.32                                |
| 10        | Sko100c           | 100          | 147862          | 150204        | 149426            | 1.56                                |
| 11        | Sko100d           | 100          | 149576          | 151592        | 151126            | 1.33                                |
| 12        | Sko100e           | 100          | 149150          | 151494        | 150736            | 1.55                                |
| 13        | Sko100f           | 100          | 149036          | 150734        | 150664            | 1.13                                |

Table 6. Performance evaluation of PSO and ACO-PSO algorithm for ChrXXX problems

| Prob. No. | Problem instances | Problem size | Best known cost | PSO Best cost | ACO-PSO Best cost | Relative percentage increase in cost |
|-----------|-------------------|--------------|-----------------|---------------|-------------------|-------------------------------------|
| 1         | Chr12a            | 12           | 9552            | 9552          | 10096             | 0.00                                |
| 2         | Chr12b            | 12           | 9742            | 9742          | 9742              | 0.00                                |
| 3         | Chr12c            | 12           | 11156           | 11186         | 11141             | 0.27                                |
| 4         | Chr15a            | 15           | 9896            | 9936          | 9978              | 0.40                                |
| 5         | Chr15b            | 15           | 7990            | 7990          | 8210              | 0.00                                |
| 6         | Chr15c            | 15           | 9504            | 9540          | 9504              | 0.38                                |
| 7         | Chr18a            | 18           | 11098           | 11462         | 11610             | 3.18                                |
| 8         | Chr18b            | 18           | 1534            | 1534          | 1534              | 0.00                                |
| 9         | Chr20a            | 20           | 2192            | 2232          | 2292              | 1.79                                |
| 10        | Chr20b            | 20           | 2298            | 2298          | 2298              | 0.00                                |
| 11        | Chr20c            | 20           | 14142           | 14996         | 14142             | 5.69                                |
| 12        | Chr22a            | 22           | 6156            | 6176          | 6156              | 0.32                                |
| 13        | Chr22b            | 22           | 6194            | 6194          | 6194              | 0.00                                |
| 14        | Chr25a            | 25           | 3796            | 3998          | 4230              | 5.05                                |

Table 7. Performance evaluation of PSO and ACO-PSO algorithm for TaiXXX problems

| Prob. No. | Problem instances | Problem size | Best known cost | PSO Best cost | ACO-PSO Best cost | Relative percentage increase in cost |
|-----------|-------------------|--------------|-----------------|---------------|-------------------|-------------------------------------|
| 1         | Tai12a            | 12           | 224416          | 224416        | 224416            | 0.00                                |
| 2         | Tai15a            | 15           | 388214          | 389718        | 389718            | 0.39                                |
| 3         | Tai17a            | 17           | 491812          | 494550        | 500966            | 0.55                                |
| 4         | Tai20a            | 20           | 703482          | 709642        | 706478            | 0.87                                |
| 5         | Tai25a            | 25           | 1167256         | 1196340       | 1179188           | 2.43                                |
| 6         | Tai30a            | 30           | 1818146         | 1846190       | 1854720           | 1.52                                |
| 7         | Tai35a            | 35           | 2422002         | 2496036       | 2475146           | 2.97                                |
| 8         | Tai40a            | 40           | 3139370         | 3227924       | 3233532           | 2.74                                |
| 9         | Tai50a            | 50           | 4941410         | 5109982       | 5085874           | 3.30                                |
### Table 8. Performance evaluation of PSO and ACO-PSO algorithm for HadXX problems

| Prob. No. | Problem instances | Problem size | Best known cost | PSO | ACO-PSO | Relative percentage increase in cost PSO | Relative percentage increase in cost ACO-PSO |
|-----------|-------------------|--------------|-----------------|-----|---------|----------------------------------------|------------------------------------------|
| 1         | Had12             | 12           | 1652            | 1652| 1660    | 0.00                                   | 0.48                                     |
| 2         | Had14             | 14           | 2724            | 2724| 2724    | 0.00                                   | 0.00                                     |
| 3         | Had16             | 16           | 3720            | 3720| 3720    | 0.00                                   | 0.00                                     |
| 4         | Had18             | 18           | 5358            | 5358| 5358    | 0.00                                   | 0.00                                     |
| 5         | Had20             | 20           | 6922            | 6922| 6934    | 0.00                                   | 0.17                                     |

### Table 9. Performance evaluation of PSO and ACO-PSO algorithm for KraXXX problems

| Prob. No. | Problem instances | Problem size | Best known cost | PSO | ACO-PSO | Relative percentage increase in cost PSO | Relative percentage increase in cost ACO-PSO |
|-----------|-------------------|--------------|-----------------|-----|---------|----------------------------------------|------------------------------------------|
| 1         | Kra30a            | 30           | 88900           | 90460| 88900   | 1.72                                   | 0.00                                     |
| 2         | Kra30b            | 30           | 91420           | 91710| 91490   | 0.32                                   | 0.08                                     |
| 3         | Kra32             | 32           | 88900           | 88900| 88900   | 0.00                                   | 0.00                                     |

### Table 10. Performance evaluation of PSO and ACO-PSO algorithm for RouXX problems

| Prob. No. | Problem instances | Problem size | Best known cost | PSO | ACO-PSO | Relative percentage increase in cost PSO | Relative percentage increase in cost ACO-PSO |
|-----------|-------------------|--------------|-----------------|-----|---------|----------------------------------------|------------------------------------------|
| 1         | Rou12             | 12           | 235528          | 235528| 240038  | 0.00                                   | 1.88                                     |
| 2         | Rou15             | 15           | 354210          | 354210| 368356  | 0.00                                   | 3.84                                     |
| 3         | Rou20             | 20           | 725522          | 725522| 726920  | 0.00                                   | 0.19                                     |

### Table 11. Performance evaluation of PSO and ACO-PSO algorithm for ScrXX problems

| Prob. No. | Problem instances | Problem size | Best known cost | PSO | ACO-PSO | Relative percentage increase in cost PSO | Relative percentage increase in cost ACO-PSO |
|-----------|-------------------|--------------|-----------------|-----|---------|----------------------------------------|------------------------------------------|
| 1         | Scr12             | 12           | 31410           | 31410| 31410   | 0.00                                   | 0.00                                     |
| 2         | Scr15             | 15           | 51140           | 51140| 51140   | 0.00                                   | 0.00                                     |
| 3         | Scr20             | 20           | 110030          | 110030| 110030  | 0.00                                   | 0.00                                     |

### 7. Conclusions

In this paper, PSO and ACO-PSO algorithms were adopted to solve the QAP problems. Ninety five well-known benchmark problems available in the literature were solved. Results shows that out of 95 problem instances, PSO produces optimum / best known solution for 61 problem instances. ACO-PSO algorithm produces optimum / best known solutions for 57 problem instances. In summary, it is noteworthy that the PSO and ACO-PSO algorithms were converging to good quality solution due the solution construction procedures adopted in this study. The solution construction procedures used in QAP were adopted from the shop schedule construction exist in the literature. In PSO, assignments were generated by assigning weights for current, particle’s best and global best assignments. In ACO-PSO, velocities for solution construction were arrived analogous to computing velocity trails in ant colony algorithms. The use of improvement schemes were also helped for attaining faster convergence. Further, algorithm has to be studied for solving real-world QAP problems.

### References

[1] Garey M R, and Johnson D S 1979 Computers and Intractability: a Guide to Theory of NP – Completeness *Freeman*. San Francisco.
Steinberg L 1961 The backboard wiring problem: a placement algorithm SIAM Review. 3 37-50.

Elshafei A N 1977. Hospital layout as quadratic assignment problem Operational Res. Quarterly. 28 167-179.

Burkard R E, and Oﬀermann J 1977 Entwurf von Schreibmaschinentastaturen mittels quadratischer Zuordnungsprobleme Mathematical Methods of Operations Research 21(4) B121-B132.

Eiselt H A, and Laporte G 1991 A Combinatorial optimization problem arising in Dartboard design Journal of the Operational Research Society 42 B121-118.

Laporte G, and Mercure H 1988 Balancing hydraulic turbine runners: A quadratic assignment problem European Journal of Operational Research 35 378-381.

Abdelkafi O, Idoumghar L, and Lepagnot J 2016 A survey on the metaheuristics applied to QAP for the graphics processing units Parallel Processing Letters. 26(03) 1650013.

Burkard R E 2013 Quadratic assignment problems Handbook of combinatorial optimization 2741-2814.

Stützle T 2006 Iterated local search for the quadratic assignment problem European Journal of Operational Research. 174(3) 1519-1539.

Rameshkumar K, Rajendran C, and Mohanasundaram K M 2011 Discrete particle swarm optimisation algorithms a review. MIT Press.

Bonyadi M R, and Michalewicz Z 2017 Particle swarm optimization for single objective optimization problem Computers & Operations Research 79 1-21.

Jordehi A R, and Jasni J 2015 Particle swarm optimisation for discrete optimisation problems. European Journal of Operational Research 241(1) 85-96.

Skorin-Kapov J 1994 Extensions of a tabu search adaptation to the quadratic assignment problem Computers & Operations Research 21(8) 855-865.

Misevicius A 2005 A tabu search algorithm for the quadratic assignment problem Computational Optimization and Applications, 30(1) 95-111.

Fleurent C, and Ferland J A 1994 Genetic hybrids for the quadratic assignment problem. Quadratic assignment and related problems, 16 173-187.

Tate D M, and Smith A E 1995 A genetic approach to the quadratic assignment problem. Computers & Operations Research, 22(1) 73-83.

Merz P, and Freisleben B 1997 A genetic local search approach to the quadratic assignment problem. Proceedings of the 7th international conference on genetic algorithms.

Gambardella L M, Taillard É D, and Dorigo M 1999 Ant colonies for the quadratic assignment problem Journal of the operational research society, 50(2) 167-176.

Stützle T 1997 MAX-MIN ant system for quadratic assignment problems. Germany: Intelletek Group, Department of Computer Science, Darmstadt University of Technology (Report No. AIDA-97-04).

Maniezzo V, and Colotti A 1999 The ant system applied to the quadratic assignment problem IEEE Transactions on knowledge and data engineering 11(5) 769-778.

Montero A R, and López A S 2015 Ant Colony Optimization for Solving the Quadratic Assignment Problem. In Artificial Intelligence (MICAI) Fourteenth Mexican International Conference, IEEE.

Jordehi A R, and Jasni J 2015 Particle swarm optimisation for discrete optimisation problems: a review. Artificial Intelligence Review, 43(2) 243-258.

Hafiz F, and Abdenour A 2016 Particle Swarm Algorithm variants for the Quadratic Assignment Problem-A probabilistic learning approach Expert Systems with Applications, 44 413-431.

Bonyadi M R, and Michalewicz Z 2017 Particle swarm optimisation for single objective continuous space problems: a review. MIT Press.

Rameshkumar K, Rajendran C, and Mohanasundaram K M 2011 Discrete particle swarm optimisation algorithms for minimising the completion-time variance of jobs in flowshops International Journal of Industrial and Systems Engineering. 7(3) 317-340.

31 Rameshkumar K, and Rajendran C 2017 A novel discrete PSO algorithm for solving job-shop scheduling problem to minimize makespan Second International Conference on Advances in Materials and Manufacturing Applications (IconAMMA)

Stützle T 1998 An ant approach to the flow shop problem Proceedings of the 6th European Congress on Intelligent Techniques and Soft Computing 3 1560-1564.

Rajendran C, and Ziegler H 2004 Ant-colony algorithms for permutation flowshop scheduling to minimize makespan/total flowtime of jobs European Journal of Operational Research, 155(2) 426-438.

Rajendran C, and Ziegler H 2005 Two ant-colony algorithms for minimizing total flowtime in permutation flowshops Computers & Industrial Engineering, 48(4) 789-797.

Rameshkumar K, Rajendran C, and Mohanasundaram K M 2012 A novel particle swarm optimisation algorithm for continuous function optimisation International Journal of Operational Research. 13(1) 1-21.
[36] Miyazaki S, Nishiyama N, and Hashimoto F 1978 An adjacent pairwise approach to the mean flow-time scheduling problem. *Journal of the Operations Research Society of Japan, 21*(2) 287-301.

[37] Varadharajan T K, and Rajendran C 2005 A multi-objective simulated-annealing algorithm for scheduling in flowshops to minimize the makespan and total flowtime of jobs *European Journal of Operational Research, 167*(3) 772-795.

[38] Burkard R E, Karisch S, and Rendl F 1991 QAPLIB-A quadratic assignment problem library *European Journal of Operational Research, 55*(1) 115-119.