A Fuzzy Particle Swarm Approach to Multiobjective Quadratic Assignment Problems

Mingyan Zhao[1], **Ajith Abraham**[2], **Crina Grosan**[3] **and Hongbo Liu**[1,4]

[1]School of Software, Dalian University of Technology, 116620 Dalian, China.

[2]Centre for Quantifiable Quality of Service in Communication Systems,
Norwegian University of Science and Technology, Trondheim, Norway.
ajith.abraham@ieee.org

[3]Department of Computer Science, Faculty of Mathematics and Computer Science
Babes-Bolyai University, Cluj-Napoca, Romania

[4]School of Computer Science and Engineering, Dalian Maritime University, 116026 Dalian, China.
lhb@dlut.edu.cn

Abstract

The multiobjective Quadratic Assignment Problem (mQAP) is considered as one of the hardest optimization problems but with many real-world applications. Since it may not be possible to simply weight the importance of each flow for the mQAP, it is best to use Pareto optimization to obtain the Pareto front or an approximation of it. Although Particle Swarm Optimization (PSO) algorithm has exhibited good performance across a wide range of application problems, research on mQAP has not much been investigated. This paper introduces a fuzzy particle swarm algorithm to handle the Multiobjective Quadratic Assignment Problem (mQAP). In the fuzzy scheme, the representations of the position and velocity of the particles in the conventional PSO is extended from the real vectors to fuzzy matrices. A new mapping is introduced between the particles in the swarm and the problem space in an efficient way. We evaluated the performance of the proposed approach. Empirical results illustrate that the approach can be applied for solving mQAP's very effectively.

1 Introduction

The scalar quadratic assignment problem (QAP) was introduced in 1957 by Koopmans and Beckmann [1]. In 2002, Knowles and Corne extended the QAP to be multiobjective and it became the multiobjective quadratic assignment problem [2]. Since then, it has several good practical applications in diverse areas, such as some layout problems, network design problems, and communication problems [3, 4, 5]. The QAP is a NP-hard optimization problem and instances of size larger than 20 are considered intractable. A literature review reveals the use of stochastic local search, genetic algorithms (GA) and ant colony optimization to find some good solutions within a reasonable amount of time.

Particle swarm optimization (PSO) incorporates swarming behaviors observed in flocks of birds, schools of fish, or swarms of bees, and even human social behavior, from which the swarm intelligence paradigm has emerged [6, 8]. The main strength of PSO is its fast convergence, which compares favorably with many global optimization algorithms [9]. However, research for solving the mQAPs using PSO approach has not much been investigated. In this paper, we make an attempt to solve the problem using a discrete particle swarm optimization approach based on a fuzzy scheme and a kind of matrix mapping, which help us to deal with the constraints in the problems and provide more diverse solutions for the problems.

The rest of the paper is organized as follows. We formulate the quadratic assignment problem and multiobjective quadratic assignment problem in Section 2. The proposed fuzzy particle swarm approach for mQAP is presented in Section 3. Experiment, results and discussions are given in Section 4 followed by some conclusions in the last Section.

2 QAP and mQAP

The QAP can be described as the problem of assigning a set of facilities to a set of locations with given distances between the locations and given flows between the facilities. The goal then is to place the facilities on locations in such a way that the sum of the product between flows and distances is minimal [10]. More formally, given n facilities $\{F_1, F_2, \cdots, F_n\}$ and n locations $\{L_1, L_2, \cdots, L_n\}$, two $n \times n$ matrices $FM = [f_{ij}]$ and $DM = [d_{rs}]$, where f_{ij} is the flow between facilities F_i and F_j and d_{rs} is the distance between locations L_r and L_s, the QAP can be stated as follows:

$$\min_{\pi \in P(n)} C(\pi) = \sum_{i=1}^{n} \sum_{j=1}^{n} f_{ij} d_{\pi_i \pi_j} \tag{1}$$

where $P(n)$ is the set of all permutations (corresponding to the assignment solutions) of the set of integers $\{1, 2, \cdots, n\}$, and π_i gives the location of facility F_i in the current solution $\pi \in P(n)$. Here $f_{ij} d_{\pi_i \pi_j}$ describes the cost contribution of simultaneously assigning facility F_i to location π_i and facility F_j to location π_j. It is to be noted that the number of facilities (n) is assumed to be the same as the number of locations. In other words, one facility could be assigned to only one location, and one location could be assigned to only one facility in a feasible assignment solution.

The term *quadratic* stems from the formulation of the QAP as an integer optimization problem with a quadratic objective function [10]. Let b_{ij} be a binary variable which takes value 1 if facility F_i is assigned to location L_j and 0 otherwise. Then the problem can be re-formulated as:

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{r=1}^{n} \sum_{s=1}^{n} f_{ij} d_{rs} b_{ir} b_{js} \tag{2}$$

s.t.

$$b_{ij} \in \{0, 1\}, \quad i = 1, 2, \cdots, n, \quad j = 1, 2, \cdots, n; \tag{3}$$

$$\sum_{i=1}^{n} b_{ij} = 1, \quad j = 1, 2, \cdots, n; \tag{4}$$

$$\sum_{j=1}^{n} b_{ij} = 1, \quad i = 1, 2, \cdots, n. \tag{5}$$

Knowles and Corne [2] presented another QAP variation (mQAP) considering several flow and distance matrices. For the allocation of facilities in hospitals, it is desired to minimize the products of the flows by the distances between doctors and patients, and between nurses and medical equipment simultaneously. Each of the matrices describing these flows is very different and it may not be possible to simply weight the importance of each flow. It can be modeled using QAP but as a multiobjective problem. Formally, it can be defined mathematically as:

$$\min_{\pi \in P(n)} \vec{C}(\pi) = \{C^1(\pi), C^2(\pi), \cdots, C^m(\pi)\} \tag{6}$$

where

$$C^k(\pi) = \sum_{i=1}^{n} \sum_{j=1}^{n} f_{ij}^k d_{\pi_i \pi_j}, \quad 1 \le k \le m. \tag{7}$$

In this last constraint, f_{ij}^k denotes the kth flow between i- and j-facilities, others are similar to the above definition in the QAP.

Since it may not be possible to simply weigh the importance of each flow for the mQAP, the solution usually must be non-dominated and it is impossible to improve one component of the solution without worsening the value of at least one other component of the solution [2]. We must deal with generating the set of non-dominated solutions for the problems having more than one objective. So the Pareto optimization is used to obtain the Pareto front or an approximation of it.

3 A Fuzzy Particle Swarm Approach for mQAP

The classical particle swarm model consists of a swarm of particles, which are initialized with a population of random candidate solutions. They move iteratively through the d-dimension problem space to search new solutions, where the fitness f can be calculated as the certain quality measure. Each particle has a position represented by a position-vector $\vec{x}_i$ (i is the index of the particle), and a velocity represented by a velocity-vector $\vec{v}_i$. Each particle remembers its own best position so far in a vector $\vec{x}_i^{\#}$, and its j-th dimensional value is $x_{ij}^{\#}$. The best position-vector among the swarm so far is then stored in a vector $\vec{x}^*$, and its j-th dimensional value is x_j^*. During the iteration time t, the update of the velocity from the previous velocity to the new velocity is determined by Eq.(8). The new position is then determined by the sum of the previous position and the new velocity by Eq.(9).

$$\begin{aligned} v_{ij}(t+1) = w v_{ij}(t) + c_1 r_1 (x_{ij}^{\#}(t) - x_{ij}(t)) \\ + c_2 r_2 (x_j^*(t) - x_{ij}(t)) \end{aligned} \tag{8}$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \tag{9}$$

Where r_1 and r_2 are the random numbers in the interval [0,1]. c_1 is a positive constant, called as coefficient of the self-recognition component, c_2 is a positive constant, called as coefficient of the social component. The variable w is

called as the inertia factor, which value is typically setup to vary linearly from 1 to near 0 during the iterated processing. From Eq.(8), a particle decides where to move next, considering its current state, its own experience, which is the memory of its best past position, and the experience of its most successful particle in the swarm.

For applying the particle swarm algorithm successfully for a problem, one of the key issues is how to map the problem solution to the particle space, which directly affects its feasibility and performance. In a "crisp" particle swarm model for the assignment problem, it would trend to assign many facilities to the same location or assign many locations to the same facility. This kind of assignment would be unfeasible. In this section, a fuzzy matrix is introduced to represent the quadratic assignment problem [11]. Then a PSO approach to the problem space mapping is depicted for the multiobjective quadratic assignment problem. Suppose $F = \{F_1, F_2, \cdots, F_n\}$, $L = \{L_1, L_2, \cdots, L_n\}$, then the fuzzy assignment relation from F to L is expressed as follows:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

Here a_{ij} represents the degree of membership of the j-th element F_j in domain F and the i-th element L_i in domain L to relation A. In the fuzzy relation matrix A between F and L, the elements are subject to the following constraints:

$$a_{ij} = \mu_R(F_j, L_i), \quad i = 1, 2, \cdots, n, \quad j = 1, 2, \cdots, n. \tag{10}$$

μ_R is the membership function, the value of a_{ij} means the degree of membership that the facility F_j would be assigned to the location L_i in the feasible assignment solution. In the quadratic assignment problem, the elements of the solution must satisfy the following conditions:

$$a_{ij} \in [0, 1], \quad i = 1, 2, \cdots, n, \quad j = 1, 2, \cdots, n; \tag{11}$$

$$\sum_{i=1}^{n} a_{ij} = 1, \quad j = 1, 2, \cdots, n; \tag{12}$$

$$\sum_{j=1}^{n} a_{ij} = 1, \quad i = 1, 2, \cdots, n. \tag{13}$$

According to fuzzy matrix representation of the quadratic assignment problem, the position X and velocity V in the particle swarm are re-defined as follows:

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nn} \end{bmatrix}$$

$$V = \begin{bmatrix} v_{11} & v_{12} & \cdots & v_{1n} \\ v_{21} & v_{22} & \cdots & v_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ v_{n1} & v_{n2} & \cdots & v_{nn} \end{bmatrix}$$

The elements in the matrix X above have the same property as Eq.(10). Accordingly, the elements of the matrix X must satisfy the following conditions:

$$x_{ij} \in [0, 1], \quad i = 1, 2, \cdots, n, \quad j = 1, 2, \cdots, n; \tag{14}$$

$$\sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, 2, \cdots, n; \tag{15}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, 2, \cdots, n. \tag{16}$$

Since the position and velocity in the new fuzzy particle swarm model have been transformed to the form of matrices, they are updated by the new Eqs.(17) and (18) by applying matrix operations.

$$\begin{aligned} V(t+1) = w \otimes V(t) \oplus (c_1 r_1) \otimes (X^{\#}(t) \ominus X(t)) \\ \oplus (c_2 r_2) \otimes (X^{*}(t) \ominus X(t)) \end{aligned} \tag{17}$$

$$X(t+1) = X(t) \oplus V(t+1) \tag{18}$$

The position matrix may violate the constraints (14), (15) and (16) after some iterations, thus it is necessary to normalize the position matrix. First we make all the negative elements in the matrix to become zero. If all elements in a column of the matrix are zero, they need be re-evaluated using a series of random numbers within the interval [0,1]. Then the matrix undergoes normalization, which makes all the rows and the columns sum up to 1.

Since the position matrix indicates the potential assignment solution, we can "defuzzify" it to the feasible solution. We choose the element, which has the maximum value in the column, then tag it as "1", and other numbers in the column and row are set as "0" in the assigning matrix. After all the columns and rows have been processed, we get the assignment solution without violating the constraints (14), (15) and (16), and then calculate the assignment cost of the solution.

In order to apply the PSO strategy for solving multiobjective quadratic assignment problems, the original scheme has to be modified. The algorithm need to search a set of different solutions (the so-called Pareto optimal set) instead of a single solution (as in global optimization) [13]. Based on the fuzzy particle swarm mapping, we apply multiobjective particle swarm optimization algorithm [12] to search toward the true Pareto front (non-dominated solutions) or approximate the Pareto optimal set for the mQAPs. Instead of the single objective particle swarm optimization,

our algorithm has a solution pool to store the non-dominated solutions found by searching up to time t. Any of the solutions in the pool can be used as the global best particle to guide other particles in the swarm during the iterated process. The plot of the objective functions whose non-dominated solutions are in the solution pool would make up of the Pareto front. The pseudo-code for the multi-objective particle swarm optimization algorithm is illustrated in Algorithm 1.

Algorithm 1 Multi-Objective Particle Swarm Algorithm

```
01. Begin
02.   Parameter settings and initialize swarm
03.   Evaluation
04.   Archive the top best into leader pool
05.   g = 1
06.   While (the end criterion is not met) do
07.       For each particle
08.           Select leader in the archiving pool
09.           Update velocity
10.           Update position
11.           Mutation periodically
12.           Evaluation
13.           Update pbest
14.       EndFor
15.       Crowding the leaders
16.       Update the top best into leader pool
17.   g++
18.   End While
19. End
```

4 Experiment and Results

To illustrate the effectiveness and performance of the particle swarm optimization algorithm, we consider the instances from Knowles's datasets[1]. In our experiments, specific parameter settings for the algorithm are described in Table 1, where D is the dimension of search space. The instances and associated Pareto fronts are provided in Table 2 and Figures 1 to 10. It is noted that all the Pareto fronts are well-distributed, since our fuzzy scheme and matrix mapping provide higher probability of generating very diverse solutions for the problems during the defuzzification process from the particles' positions to the solutions.

5 Conclusions

In this paper, we presented a fuzzy particle swarm algorithm to handle the mQAP. In the fuzzy scheme, the representations of the position and velocity of the particles in

[1] http://dbkgroup.org/knowles/mQAP

Table 1. Parameter settings for the algorithm.

Parameter name	Parameter value
Swarm size	$int(10 + 2sqrt(D))$
Self coefficient c_1	$0.5 + log(2)$
Social coefficient c_2	$0.5 + log(2)$
Inertia weight w	0.91
Maximum Iteration	$2D$

Table 2. Test Suite used.

Instance name	Instance category	Loc. no.	Flow mat. no.	Pareto front
KC10-2fl-1rl	Real-like	10	2	Figure 1
KC10-2fl-3rl	Real-like	10	2	Figure 2
KC10-2fl-5rl	Real-like	10	2	Figure 3
KC20-2fl-2rl	Real-like	20	2	Figure 4
KC20-2fl-3rl	Real-like	20	2	Figure 5
KC20-2fl-3uni	Uniform	20	2	Figure 6
KC20-2fl-4rl	Real-like	20	2	Figure 7
KC20-2fl-5rl	Real-like	20	2	Figure 8
KC30-3fl-1rl	Real-like	30	3	Figure 9
KC30-3fl-2rl	Real-like	30	3	Figure 10

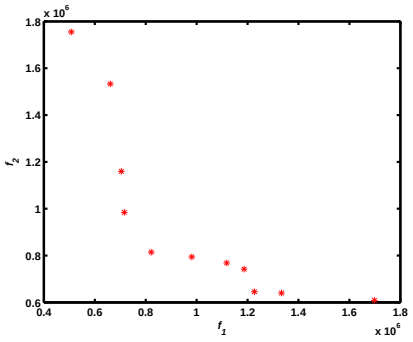

Figure 1. Pareto front for KC10-2fl-1rl.

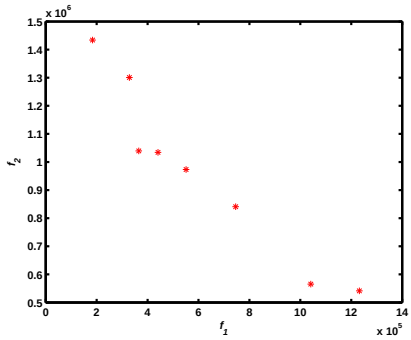

Figure 2. Pareto front for KC10-2fl-3rl.

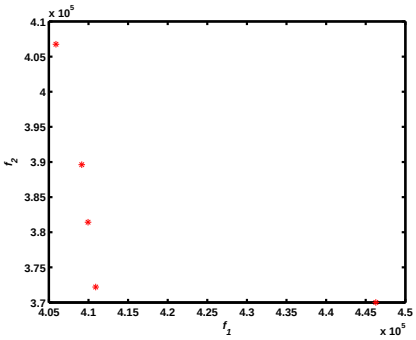

Figure 6. Pareto front for KC20-2fl-3uni.

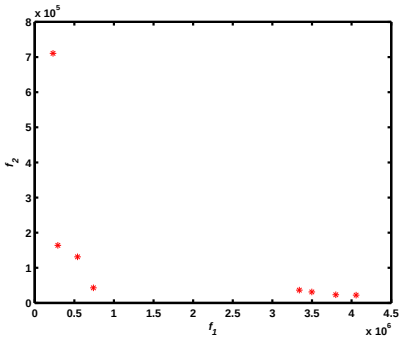

Figure 3. Pareto front for KC10-2fl-5rl.

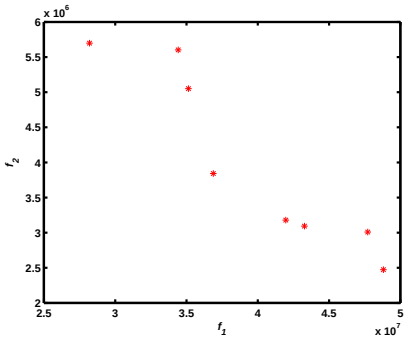

Figure 7. Pareto front for KC20-2fl-4rl.

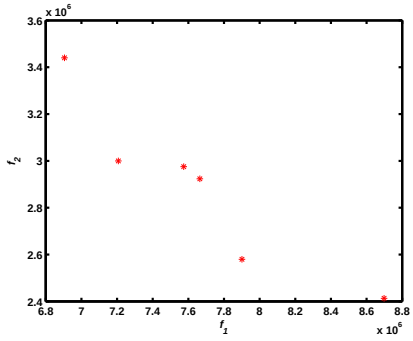

Figure 4. Pareto front for KC20-2fl-2rl.

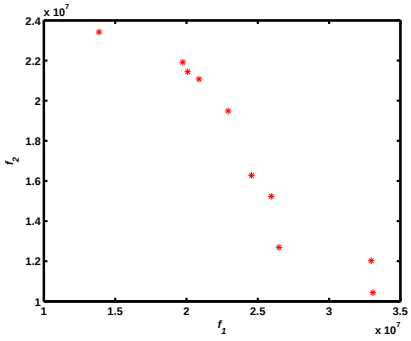

Figure 8. Pareto front for KC20-2fl-5rl.

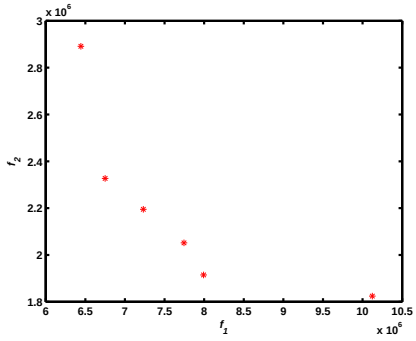

Figure 5. Pareto front for KC20-2fl-3rl.

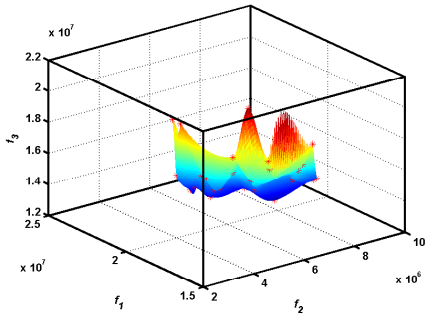

Figure 9. Pareto front for KC30-3fl-1rl.

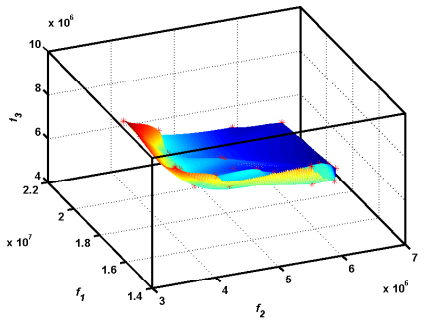

Figure 10. Pareto front for KC30-3fl-2rl.

the conventional PSO is extended from the real vectors to fuzzy matrices. We illustrated our approach by solving four benchmark mQAP instances and their Pareto fronts were provided. The Pareto fronts are well-distributed. Empirical results illustrate that the approach can be applied for solving mQAPs very effectively.

Our future work is targeted to generate more mQAP instances and involve more heuristics including hybrid approaches [14], [15].

Acknowledgments

This work is supported partly by NSFC (60573087) and MOST (2005CB321904).

References

[1] T. Koopmans and M. Beckman. Assignment Problems And The Location of Economic Activities. IEEE Econometrica, 25, pp. 53-76, 1957.

[2] J. Knowles and D. Corne. Towards Landscape Analyses To Inform The Design of A Hybrid Local Search for The Multiobjective Quadratic Assignment Problem. Soft Computing Systems: Design, Management and Applications, pp. 271-279. IOS Press, Amsterdam, 2002.

[3] R. Day and G. Lamont. Multiobjective Quadratic Assignment Problem Solved by an Explicit Building Block Search Algorithm – MOMGA-IIa. Lecture Notes in Computer Science (EvoCOP 2005), 3448, pp. 91-100, 2005.

[4] E. Loiola, N.M. Maia de Abreu, P.O. Boaventura-Netto, and T. Querido. A Survey For The Quadratic Assignment Problem. European Journal of Operational Research, 176, pp. 657-690, 2007.

[5] J. Knowles and D. Corne. Particle Swarm Optimization Versus Genetic Algorithms For phased Array Synthesis. In Proceedings of the Second International Conference Evolutionary Multi-Criterion Optimization, pp. 295-310, 2003.

[6] J. Kennedy and R. Eberhart. Swarm Intelligence, Morgan Kaufmann, CA, 2001.

[7] C. Grosan and A. Abraham, Stigmergic Optimization: Inspiration, Technologies and Perspectives, Studies in Computational Intelligence, Springer Verlag, Germany, pp. 1-24, 2006.

[8] H. Liu, A. Abraham, and M. Clerc. Chaotic Dynamic Characteristics in Swarm Intelligence. Applied Soft Computing Journal, 7, pp. 1019-1026, 2007.

[9] A. Abraham, H. Guo, and H. Liu. Swarm Intelligence: Foundations, Perspectives and Applications. In Studies in Computational Intelligence, N. Nedjah and L. Mourelle (Eds.), pp. 3-25. Springer, 2006.

[10] T. Stützle. Iterated Local Search for the Quadratic Assignment Problem. European Journal of Operational Research, 174, pp. 1519–1539, 2006.

[11] W. Pang, K. Wang, C. Zhou, and L. Dong. Fuzzy Discrete Particle Swarm Optimization For Solving Traveling Salesman Problem. In Proceedings of the Fourth International Conference on Computer and Information Technology, pp. 796-800, IEEE CS Press, 2004.

[12] C. Coello, G. Pulido, and M. Salazar. Handling Multiobjectives With Particle Swarm Optimization. IEEE Transactions on Evolutionary Computation, 8, pp. 256–279, 2004.

[13] A. Abraham, L. Jain and R. Goldberg (Eds.), Evolutionary Multiobjective Optimization: Theoretical Advances and Applications, Springer Verlag, London, ISBN 1852337877, 12 Chapters, 2005.

[14] C. Grosan, A. Abraham and M. Nicoara, Search Optimization Using Hybrid Particle Sub-Swarms and Evolutionary Algorithms, International Journal of Simulation Systems, Science and Technology, UK, Volume 6, Nos. 10 and 11, pp. 60-79, 2005.

[15] C. Grosan, Improving the performances of evolutionary algorithms for the multiobjective 0/1 knapsack problem using epsilon -dominance. Congress on Evolutionary Computation (CEC), G. Greenwood et al. (Eds.), IEEE Press, pp. 1958-1963, Portland, USA, 2004.