



Facility layout problem with QAP formulation under scenario-based uncertainty

Mert Şahinkoç & Ümit Bilge

To cite this article: Mert Şahinkoç & Ümit Bilge (2018): Facility layout problem with QAP formulation under scenario-based uncertainty, INFOR: Information Systems and Operational Research, DOI: [10.1080/03155986.2018.1424445](https://doi.org/10.1080/03155986.2018.1424445)

To link to this article: <https://doi.org/10.1080/03155986.2018.1424445>



Published online: 24 Mar 2018.



Submit your article to this journal [↗](#)



View related articles [↗](#)



View Crossmark data [↗](#)



Facility layout problem with QAP formulation under scenario-based uncertainty

Mert Şahinkoç  and Ümit Bilge

Industrial Engineering Department, Bogazici University, Istanbul, Turkey

ABSTRACT

The facility layout problem is usually treated as a deterministic problem and uncertainty regarding problem parameters has seldom been addressed. This study aims to investigate different ways of dealing with uncertainty to design a facility layout which attains robust and efficient performance under a finite number of possible scenarios. For this purpose, several mathematical models based on the quadratic assignment problem (QAP) formulation are developed. These formulations cover alternative approaches in stochastic programming and robust optimization literature such as: minimizing expected cost, maximum cost and maximum regret. Proposed models are solved using genetic algorithms incorporating operators and schemes that are specially selected and adapted for the models. Finally, a novel approach, where the optimization problem under scenario-based uncertainty is transformed into a multi-objective optimization problem by considering each scenario as a separate objective, is proposed. By solving the multi-objective counterpart of scenario-based QAP (*m*QAP), optimal solutions with respect to different robust performance measures can be obtained simultaneously in a Pareto optimal set. A multi-objective evolutionary algorithm is developed to solve the *m*QAP. Extensive numerical analysis enables comparison of the performance of these approaches and provides important insights about dealing with uncertainty in the facility layout problem.

ARTICLE HISTORY

Received 28 February 2017
Accepted 3 January 2018

KEYWORDS

Scenario-based uncertainty;
robust optimization;
stochastic programming;
genetic algorithm; multi-
objective optimization;
quadratic assignment
problem

1. Introduction

Facility layout decisions are strategically important, costly and usually difficult to change or adapt. On the other hand, the market environments in which the facilities operate are typically subject to uncertainties. Uncertainty and variability often arise in the product mix, part routings and the production amounts which are all represented in the flow matrix of the layout problem. In spite of this inherent uncertainty, the facility layout problem has usually been treated as a static deterministic problem. In this study, different ways of addressing uncertainty while designing an equal-area facility layout problem formulated as a quadratic assignment problem (QAP) are investigated.

Uncertainty is often described by a set of alternative scenarios. Thus, a finite number of sampled instances of uncertainty is considered and for a given scenario realization, the problem reduces to a deterministic one. It is argued in the literature that the scenario-based approach generally results in more tractable optimization models (Snyder 2006) and it can successfully reflect the characteristics of system uncertainties (Wu et al. 2012). For instance, this approach has the advantage of allowing parameters to be statistically dependent, which is not the case when parameters are governed by continuous probability distributions. Dependence is usually necessary to build realistic models, i.e. demands for different product types produced in a facility and production amounts of these items and their components are usually correlated.

Scenario-based uncertainty is usually handled as either stochastic programming or robust optimization. In the stochastic programming, probabilities associated with the occurrence of scenarios are known in advance by the decision-maker and the goal is usually optimizing the expected behaviour. A thorough discussion of stochastic programming theory and models can be found in Shapiro et al. (2009) among others. In the robust optimization, on the other hand, these probability distributions are not necessary, and the common attempt is to optimize the worst-case performance of the system. One of the most common robustness measures in robust optimization methodology is minimizing the cost of the scenario where the maximum cost occurs (minimax cost). Robustness measures that involve the regret concept are also very common. The regret for a scenario is the absolute or percentage deviation of the objective value of a solution from the objective value of the optimal solution for that scenario. Models try to minimize the maximum absolute (or relative) regret across all scenarios. The main attraction of such robustness measures is that they do not require the decision-maker to estimate scenario probabilities. Kouvelis and Yu (1997) provide a framework for the robust discrete optimization. Comprehensive surveys on robust optimization are presented by Ben-Tal and Nemirovski (2002) and Beyer and Sendhoff (2007). Ben-Tal et al. (2009) focus on modelling the robust counterparts of linear programming problems and extend their findings to more general optimization problems. Snyder (2006) presents a comprehensive survey on the uncertainty in facility location problems in which almost every stochastic and robust performance measure is stated and explained briefly. Also, examples from more general logistic problems are covered to illustrate different types of stochastic programming and robust optimization approaches. Aissi et al. (2009) present a survey on the topic of discrete minimax cost and minimax regret versions of the combinatorial optimization problems. p -Robustness introduced by Snyder and Daskin (2006) put constraints on the regret of each scenario. Other approaches include γ -robustness, α -reliability and conditional value at risk.

As we find several performance measures proposed in the literature to deal with uncertainty in optimization problems, it becomes clear that choosing an appropriate performance measure is a significant part of the modelling process. This is a critical issue since the definition for 'good performance under the set of all scenarios' depends on the decision-maker as well as the given case. Furthermore, in some cases, it may be difficult for the decision-maker to choose a performance measure before seeing the trade-offs.

In fact, the techniques driven from both stochastic programming and robust optimization methodologies offer different ways of converting the distinct objective values, e.g. cost of a layout, given by a feasible solution under each scenario realization into a single

performance measure, e.g. max regret. Any performance measure in robust optimization or stochastic programming is a particular preset function of scenario objectives and leads into a compromising decision.

At this point, a novel approach is to transform the optimization problem under scenario-based uncertainty into a multi-objective optimization problem by considering each scenario as a separate objective. By solving the multi-objective counterpart of the scenario-based optimization problem, optimal solutions with respect to different performance measures can be obtained simultaneously within a Pareto optimal set. A few authors have discussed that this is a valid approach (we are aware of Aissi et al. 2009 and Klamroth et al. 2013) but the approach has not been implemented before in the robust optimization or stochastic programming literature to test its viability.

As a result, this paper contributes by:

- providing QAP models (and their solutions) under scenario-based uncertainty for several performance measures existing in stochastic programming and robust optimization literature; such as minimizing expected cost, maximum cost, and maximum absolute and relative regret; and
- providing the multi-objective counterpart of scenario-based QAP (*mQAP*) to dwell on the fact that robust solutions should be in its Pareto front.

Proposed QAP-based formulations are solved using genetic algorithms (GAs) to search the optimal or Pareto optimal solutions for our formulations. Genetic operators and local improvement schemes are selected and adapted for our formulations. Also, following the remark by Iancu and Trichakis (2013) that worst-case minimization in robust optimization might lead to ‘un-optimized’ decisions for the non-worst-case scenarios, we take this into account in our solution procedures. For the *mQAP* formulation, we proposed a multi-objective genetic algorithm to approximate an efficient frontier in reasonable computation times. After investigating the approaches in multi-objective evolutionary algorithm (MOEA) literature, non-dominated sorting genetic algorithm-II (NSGA-II) proposed by Deb et al. (2002) is adopted with several modifications to address the specific characteristics of the problem.

Extensive numerical analysis enables us to compare the performance of these approaches in terms of obtaining robust solutions as well as gaining insights about the relationship among robust optimization and multi-objective optimization.

The rest of the paper is organized as follows. In Section 2, a brief literature survey on QAP is presented. In Section 3, we introduce five formulations for QAP-based facility layout problems under uncertainty and in Section 4, we present our proposed solution methods. Section 5 covers the numerical experiments and comparisons of the proposed methods. Finally, Section 6 concludes this paper and future working directions are discussed.

2. A brief review of the QAP literature

QAP is one of the most difficult problems in the NP-hard class (Sahni and Gonzales 1976). In Loiola et al. (2007), a comprehensive survey about QAP is presented stating that many real-life problems in several areas such as; facilities location, combinatorial data

analysis are modelled as QAP. Some of the most important QAP formulations are stated and classified and a detailed discussion on the exact and heuristic solution techniques, including metaheuristic strategies is provided. In addition, the main research trends and tendencies in the QAP literature are identified to guide future researches and these materials constructed a foundation for our QAP investigation.

In Zhang et al. (2013), general purpose mixed integer linear programming solvers are addressed to solve QAP. Different types of formulations are obtained by using linearization techniques to find tight lower bounds and efficient performance to solve test problems in reasonable times. Other examples that apply mixed integer linear programming reformulation techniques are Xia and Yuan (2006) and Nyberg and Westerlund (2012). Kaufman and Broeckx (1978) apply Bender's decomposition whereas in Resende et al. (1995) an interior point algorithm for linear programming is developed to compute lower bounds for the QAP in order to serve branch-and-bound techniques to come up with optimal solutions with a convenient effort. Some other exact methods include reformulation-linearization techniques that are used in Hahn et al. (2012), and Rostami and Malucelli (2014). However, all these techniques are proved to be impractical for all but small-sized problems. The computational difficulty in solving QAP motivated the development of many heuristic algorithms. While a comprehensive review of the literature for QAP is beyond the scope of the current paper, the following is a brief overview of the studies that served in constructing our approaches.

During the recent decades, almost every heuristic search technique has been adopted to QAP. Pardalos and Resende (1994) develop a greedy randomized adaptive search, called GRASP. They claim that their solution technique is capable of quickly producing good quality solutions not only for QAP but also a wide variety of combinatorial optimization problems.

Taillard (1995) compares the performances of three different tabu search algorithms and a hybrid GA, indicating that all these solution methods are efficient heuristics and none of among them outperforms the other. Tate and Smith (1995) are among the pioneers that employ GAs to solve QAPs. All the main stages of the GAs including the tuning for the parameters are investigated and they draw the conclusion that GA is a very powerful tool for QAP.

Misevicius (2003) proposes a GA hybridized with an improvement procedure called ruin and recreate procedure. In Misevicius (2004), a hybrid GA using some elements of tabu search at its local improvement stage, is proposed. Drezner (2005) proposes a compounded GA consisting of two phases, where the second phase evolves good solutions of the first phase. His algorithm is hybridized with tabu search and successful results are obtained for the test problems in QAPLIB. Later on, in Drezner (2008) some modifications concerning mainly diversification are proposed and improved results are obtained. In Tosun et al. (2013) and Ahmed (2016), some crossover operator alternatives are explored.

Ahuja et al. (2000) propose a greedy GA including many greedy principles for both intensification and diversification. They present several alternatives for the crossover operator, concepts of immigration and tournament and a detailed fine tuning for the algorithm parameters. The methodology and the operator schemes of their paper serve as a basis for the GAs developed here.

Knowles and Corne (2002) present a multi-objective version of QAP (*mQAP*) considering several flow and distance matrices. The authors state that *mQAP* can be useful for some layout problems, such as hospital layout where different types of flows, i.e. doctors, patients, and nurses lead to different objectives. They investigate landscape analysis issues for approximating Pareto front using *mQAP* as a benchmark. In their paper, a hybrid local search algorithm is presented to approximate the Pareto front and in Knowles and Corne (2003), they formulate some instance generators and test suites for *mQAP*.

In Paquete and Stützle (2006), a two-phased local search procedure is used whereas in López-Ibáñez et al. (2004) and Özkale and Fiğlalı (2013) ant colony optimization meta-heuristic is employed for the bi-objective QAP. There are some works applying MOEAs for *mQAP*; such as Kleeman et al. (2004) and Day and Lamont (2005).

Studies that address uncertainty in the facility design problems, particularly QAP are sparse. To the best of our knowledge, robustness notion is first mentioned in the context of a facility design problem by Rosenblatt and Lee (1987). Although a formulation is not provided, the facility layout model is described as a QAP. The authors enumerate the feasible solutions of a small example and measure the robustness of a solution by the frequency it lies within a pre-specified percentage of an optimal solution for the scenario set. Kouvelis et al. (1992) also propose a similar definition that leads to the *p*-robustness approach for the single and multiple period QAP formulation. Benjaafar and Sheikhzadeh (2000) propose stochastic programming approach for the generalized QAP formulation where flow volumes assigned among departments are also decision variables. They also include a robustness constraint in their formulation. More recently, QAP under uncertainty is studied in Feizollahi and Modarres (2012) with uncertain department locations, in Feizollahi and Averbakh (2014), and Feizollahi and Feyzollahi (2015) with uncertain flows. These studies use interval uncertainty parameters and robust optimization metrics such as minimax cost and minimax regret are covered by applying solution techniques such as: Bender's decomposition, local search and tabu search. Zhao and Wallace (2014) integrate the QAP with flow assignment problem in a under demand uncertainty represented in discrete levels though no specific solution method is suggested to solve QAP.

3. Proposed models

Five different mathematical programming formulations for QAP-based facility layout problems under uncertainty are provided in this section to investigate different stochastic programming and robust optimization approaches. The integer nonlinear programming formulation below is the basis of these proposed models and will be referred as the *classical QAP*.

$$\begin{aligned} &\text{Classical QAP :} \\ &\min \sum_{i,j,k,l} f_{ij} d_{kl} x_{ik} x_{jl} \end{aligned} \quad (1)$$

$$\begin{aligned} &\text{s.t.} \\ &\sum_i x_{ik} = 1 \quad \forall k \end{aligned} \quad (2)$$

$$\sum_k x_{ik} = 1 \quad \forall i \quad (3)$$

$$x_{ik} \in \{0, 1\} \quad \forall i, k \quad (4)$$

The indices i and j are used to denote departments in the set I , while k and l denote the locations in the set K . Flow between departments i and j is denoted by f_{ij} and the distance between locations k and l is denoted by d_{kl} . The binary decision variable x_{ik} takes value 1 if department i is located at location k , and 0 otherwise.

The classical QAP model is valid when both the flow and the distance matrices are assumed to be known deterministically. However, proposed formulations use a finite set of discrete scenarios to model the uncertainty displayed by the input parameters. Let us define S as the set of the scenarios. For each scenario s , where $s \in S$, there is a different deterministic minimization problem having a specific flow matrix. In this paper, only the flows between the departments are defined distinctively for each possible scenario and denoted by f_{ij}^s . The reason behind is that the product mixes and the quantities of each product type produced can easily change due to various reasons such as changes in demand or changes in the production preferences. On the other hand, all scenarios share the same distance matrix. In other words, distance matrix is not affected by the scenario-based uncertainty. However, this assumption can easily be dropped and the solution procedures and results can be extended for the cases having also specific distance matrices for each scenario.

While the notation given above is common in all the rest of the formulations, some additional notation will be introduced specifically for some of the formulations.

3.1. Minimizing expected cost

In the first model, the objective is to minimize the expected cost of the assignment of departments to locations under all possible scenarios in the scenario set. This model includes stochastic component in the objective function and constraints are the same as in the classical QAP:

$$\begin{aligned} &\text{Model 1: expQAP} \\ &\min \sum_{i,j,k,l,s} p_s f_{ij}^s d_{kl} x_{ik} x_{jl} \\ &\text{s.t.} \\ &\quad (2), (3) \text{ and } (4) \end{aligned} \quad (5)$$

The parameter p_s , the probability that scenario s occurs, is used specifically in this formulation. Stochastic programming requires some degree of probability information that is sometimes provided as probability mass/density functions or in our case distinct probability values assigned for each scenario. Obviously, as all probable scenarios are represented in the scenario set S , the sum of their probabilities must be equal to one. In our case, all p_s values are assumed to be equal without loss of generality meaning that each scenario is equally likely to happen. Solving this formulation is identical to solving the classical QAP. The only difference is that instead of a single set of flow parameters, we now

have $|S|$ of them. By taking a weighted average with respect to the p_s values, we can fit all flow parameters into a single matrix and reduce the problem into the classical QAP.

3.2. Minimizing maximum cost

As the first of the three formulations belonging to the robust optimization methodology approach, the minimax cost solution minimizes the maximum cost across all scenarios and thus gives emphasis to the worst case. This minimax structure might make robust optimization problems more difficult to solve than stochastic programming. The primary attraction of minimax measures is that they do not require decision-maker or planner to estimate scenario probabilities:

$$\begin{aligned} &\text{Model 2: maxQAP} \\ &\min C_{\max} \end{aligned} \tag{6}$$

$$\begin{aligned} &\text{s.t.} \\ &\sum_{i,j,k,l} f_{ij}^s d_{kl} x_{ik} x_{jl} \leq C_{\max} \quad s \in S \\ &(2), (3) \text{ and } (4) \end{aligned} \tag{7}$$

The variable C_{\max} is defined as the maximum travel distance (or, cost) across all scenarios and the objective is to minimize this parameter. Solving this model is again similar to solving the classical QAP when a GA is employed. The only difference is in the calculation of the fitness values of the generated solutions. To calculate the fitness of a solution, total travel distance computation is made for each scenario and maximum of those values is taken as fitness of the solution.

3.3. Minimizing maximum absolute and relative regret

The term regret used in robust optimization terminology is defined as the difference between the cost of a solution in a given scenario and the cost of the optimal solution for that scenario. The minimax regret solution minimizes the regret across all scenarios. There is no need to develop models for minimizing expected (average) regret since this is equivalent to minimizing expected cost (Snyder 2006).

The regret can be computed as either absolute or relative difference and both of these two measures can be transformed into each other. When the difference between the cost of a solution in a given scenario and the cost of the optimal solution for that scenario is defined in terms of nominal units, this is called the absolute regret of a solution.

The parameter c_s^* is the cost of the optimal solution for the scenario s and it is an input to the regret model. The c_s^* values are assumed to be computed beforehand by solving $|S|$ separate and deterministic QAPs. The variable R_{\max}^{abs} is defined as the maximum regret across all scenarios and the objective is to minimize this parameter:

$$\begin{aligned} &\text{Model 3: absQAP} \\ &\min R_{\max}^{\text{abs}} \end{aligned} \tag{8}$$

$$\begin{aligned}
& \text{s.t.} \\
& \sum_{i,j,k,l} f_{ij}^s d_{kl} x_{ik} x_{jl} - c_s^* \leq R_{\max}^{\text{abs}} \quad s \in S \\
& (2), (3) \text{ and } (4)
\end{aligned} \tag{9}$$

When the difference between the cost of a solution in a given scenario and the cost of the optimal solution for that scenario is defined in terms of the percentage value of the optimal solution for that scenario, the objective becomes minimizing the maximum relative regret, R_{\max}^{rel} :

$$\begin{aligned}
& \text{Model4 : relQAP} \\
& \min R_{\max}^{\text{rel}}
\end{aligned} \tag{10}$$

$$\begin{aligned}
& \text{s.t.} \\
& \frac{\sum_{i,j,k,l} f_{ij}^s d_{kl} x_{ik} x_{jl} - c_s^*}{c_s^*} \leq R_{\max}^{\text{rel}} \quad s \in S \\
& (2), (3) \text{ and } (4)
\end{aligned} \tag{11}$$

3.4. The multi-objective QAP (mQAP)

In the last model, the objective function becomes a vector function whose elements represent the objectives of the solutions in each scenario separately. In *mQAP* introduced by Knowles and Corne (2002), different flow types are considered as multiple objectives. In our case, each scenario becomes a separate objective so the number of objective functions is equal to the number of scenarios:

$$\begin{aligned}
& \text{Model 5: } m\text{QAP} \\
& \min_x \vec{C}(x) = \{c^1(x), c^2(x), \dots, c^{|S|}(x)\}, \\
& \text{where } c^s(x) = \sum_{i,j,k,l} f_{ij}^s d_{kl} x_{ik} x_{jl} \quad s \in S \\
& \text{s.t.} \\
& (2), (3) \text{ and } (4)
\end{aligned} \tag{12}$$

The motivation behind developing the *mQAP* formulation is the fact that robust optimization problems can be seen as a multi-objective optimization problems with objectives corresponding to uncertainty scenarios. The optimum solutions for all stochastic and robustness measures covered must fall within the Pareto front obtained from the *mQAP* formulation. This phenomenon, called *Pareto robust optimal solutions* by Iancu and Trichakis (2013), briefly explains that for every robustness measure one of the robust optimal solutions should also be a Pareto optimal solution. This indicates that if there is only one robust optimal solution, then it must be a member of the Pareto front.

Mathematical propositions for specific robustness measures are presented in Aissi et al. (2009) stating that in a combinatorial minimization problem at least one optimal solution of minimax cost and minimax regret versions of the problem must be a Pareto optimal

solution. It means that among all members of Pareto front, one solution must have a minimum maximum cost or minimum maximum regret value. The strong relationship between the robustness measures and the multi-objective version of an optimization problem enables finding the optimal solutions in terms of all robustness measures if the true Pareto front can be obtained.

A similar proposition can also be offered for the expected cost measure. A solution that is optimal in terms of the expected cost criterion must also be a Pareto optimal solution for our $mQAP$ formulation. The difference is that in the case of multiple optimal solutions for the expected cost criterion, not only at least one of the optimal solutions but all of them must also be Pareto optimal solutions.

Proposition 3.1: *The optimal solution for the expected cost criterion must be a non-dominated solution.*

Proof: Let us define $x \in X$ with the objective function vector, $u = (u_1, u_2, \dots, u_{|S|})$ as a dominated solution and we want to show that it is an optimal solution for the expected cost criterion. By the definition of Pareto dominance if x is a dominated solution, there must be at least one solution $y \in X$ with the objective function vector, $v = (v_1, v_2, \dots, v_{|S|})$ satisfies that for $\forall s \in S, u_s \geq v_s \cap \exists s \in S, u_s > v_s$. However, if this is the case then the expected cost of solution y is strictly less than the expected cost of solution x meaning that solution x cannot be an optimal solution for the expected cost criterion. Hence, by proof with contradiction, the optimal solution for the expected cost criterion must be a non-dominated solution. \square

By using $mQAP$ model, we can approximate the Pareto front in a single algorithm dedicated to $mQAP$ formulation and select the appropriate member based on our chosen performance measure, instead of evaluating each performance measure separately using the first four formulations. The degree of success, of course, depends on the success in approximating the true Pareto front.

4. Solution methods

As QAP is computationally NP-hard, large problem instances can require a great amount of time to be solved optimally by exact methods. GA is one of the most popular and successful heuristic algorithms to solve optimization problems. Moreover, multi-objective evolutionary approaches, which work satisfactorily in multi-objective optimization domains, are suitable for our purpose of approximating the Pareto front for $mQAP$ formulation. Therefore, to remain compatible and comparable, GA becomes a natural choice to use across all five formulations.

Basically, GA imitates the process of natural evolution over a population of individuals each representing a feasible solution of an optimization problem. The evolution process is achieved by using several strategies like crossover and mutation that can be adapted to the specifications of problem structure. Next, the details for the single objective GA used for first four formulations and the multi-objective GA for the $mQAP$ formulation are explained.

4.1. Single objective GA

In our GA design, an individual is encoded by a chromosome composed of n genes each representing a department. Each allele represents the location index where the

corresponding department is assigned in that particular individual. To determine the fitness value of each individual, first the scenario objectives for the corresponding solution are computed. Fitness is calculated based on the objective function of the particular QAP formulation using these scenario objective values. An important remark is related with the tie-breaking rule used during sorting and any type of comparison of individuals throughout the GA. In case of a tie, an efficient individual that is, the solution which performs at least as good as the other under all scenarios, will be favoured. In this manner, the pitfall mentioned in Iancu and Trichakis (2013) that worst-case minimization in robust optimization might lead to ‘un-optimized’ decisions for the non-worst-case scenarios, is avoided.

When generating the initial population, feasible individuals are produced in a random manner ensuring that departments are not assigned same locations. To this random population, optimal solutions obtained by solving $|S|$ separate classical QAPs are included as seeds. Since the scenario optimal solutions are already obtained for the regret formulations, it makes sense to make this information available to all cases. Furthermore, our preliminary analysis supports that the quality of our GA improves by including these individuals in the initial population.

Binary tournament is implemented for parent selection, where the individual with a better fitness value is favoured among two individuals randomly selected from the population. The comparison between the candidate individuals is made with respect to their fitness values and ties are broken randomly.

As the crossover operator, we employ the path crossover proposed by Ahuja et al. (2000) as it performs significantly better than other alternatives such as one-order crossover in our preliminary analysis. In path crossover, the *path* connecting the parents is formed by creating new solutions through move operations (i.e. swap) that make the parents look more alike at each step.

In the path swap operator scheme, starting from a random position of the chromosomes, the alleles of the two parents at that position are examined. If the corresponding alleles of two parents are different, a swap operation is performed among the parents. Two resulting solutions are inspected and whichever solution is fitter, a move is made by the corresponding parent, thus forming a node on the path. In the next iteration, this new node is considered as the parent and compared with the other parent. In each swap operation, two parents look more similar and swap moves continue until all alleles of two parents become the same and the full path is formed. Once the path is built, the nodes are considered as candidate children. The fittest member of the path is selected as the first child. In addition to that, one of the original parents is chosen as the second child based on fitness value comparison between two parents. If the generated child is fitter than both parents, the parent which is less similar to the first child (having less number of alleles in common) is chosen as second child.

To impose diversity on the children formed in this way, immigration is used instead of mutation. The children population is sorted and some pre-specified per cent of its poorest members is removed. Newly introduced solutions, called immigrants, which are significantly different from the individuals created so far replace these members. For the generation stage of immigrants, Ahuja et al. (2000) propose a process that makes use of the historical frequency information similar to the long-term memory in tabu search (Glover and Laguna 1999). In this approach, the number of times each department has been assigned to each location is stored in a $n \times n$ matrix called *population history*. The

immigrants are created by choosing departments in a random sequence and assigning them to the available location with lowest value in the population history matrix.

However, by generating immigrants with this methodology, there is a high chance that the fitness of immigrants will be below than other members already existing in the current population. The outcome is that they rarely qualify as parent individuals and contribute to create new offspring. In addition, it is likely for them to be eliminated at the very next iterations. Therefore, all immigrant individuals are subjected to the local improvement operation right after their creation. In this way, the unexplored regions of the solution space are intensively investigated and individuals with hopefully fair fitness values are incorporated into the population leading to improvement in the overall quality. As for the local improvement operator, the 2-exchange neighbourhood local search is used.

Also, some pre-specified percentage of individuals in the resulting population (excluding the immigrants) enters local improvement operation in each iteration. By adjusting the values of immigration rate and local improvement rate parameters properly, one can ensure the balance of diversification and intensification in the GA.

4.2. Proposed multi-objective GA

The MOEA techniques are classified into different categories based on the sequence of two main stages of the algorithm (Deb and Gupta 2005). These two stages are searching the solution space with respect to objective functions and deciding on what kind of trade-offs between the priorities of the objectives are convenient from the decision-maker's perspective. Some techniques make decision about the priority ranks of the objectives before searching the solution space, whereas some do the reverse.

The choice in our research is to find as many solutions as possible in the Pareto front and then leave the rest of the multi-criteria decision-making process to a decision-maker. This is called a posteriori technique. In Zitzler et al. (2000), some of the most applied MOEAs with different a posteriori approaches are compared and evaluated with respect to the closeness of their approximations of the true Pareto front. The performance also depends on diversity which is not only about avoiding local optimums but also covering the entire Pareto front. Based on their findings, NSGA-II which has been developed by Deb et al. (2002) is adopted for solving our multi-objective formulation.

In NSGA-II, the sorting of individuals requires the calculation of two attributes for each individual: *non-domination rank* and *crowding distance*. In the calculation stage of non-domination rank, all solutions are examined in regard to whether they are dominated by any other solution or not. Then, all the non-dominated solutions are ranked as one. After excluding these non-dominated solutions, the same is done for the solutions in the remaining set and the non-dominated solutions are ranked as two and this procedure is implemented until all individuals in the entire population are ranked.

After the assignment of non-domination ranks, a front for each rank is formed and crowding distances are assigned. The goal in crowding distance assignment is to ensure that every region of a front is represented by enough individuals (i.e. the individuals are not crowded into certain regions). After sorting the individuals in the same front with respect to each of the objective criteria (scenarios in our case), a solution's distance to its closest neighbour on a sorted list is calculated as the absolute normalized difference of their corresponding objective values. The crowding distance of an individual is computed

as the sum of these distances over the objective criteria. As a result, individuals who take place in the scarce regions of their fronts will have high crowding distance values and they will be preferred for reproduction in GA.

After the assignment of two attributes, solutions are sorted primarily based on their non-domination ranks and solutions in the same front are sorted based on their crowding distance assignment. In NSGA-II, the current population and the newly generated children population are combined and sorted based on the non-dominated sorting criteria in each iteration. Since all previous and current members are included in the combined population, elitism is ensured. Then, the set of best individuals with a size equal to population size parameter is kept and declared as the new generation of population.

Evidently, some modifications will be necessary on NSGA-II to address the specific characteristics of our problem. When generating the initial population, optimal solutions achieved from solving $|S|$ separate classical QAPs are included in the initial population. In this way, genetic material from the extreme ends of the objective function space is introduced in the population. We observed that this greatly enhances the capability of the search in representing the whole range of the Pareto front. Another enhancement that provides the search with valuable genetic information is achieved by making us of the final populations of the $|S|$ separate classical QAPs. All final populations are combined and members of the first front of this combined population are added to the initial population. The remaining members of the initial population are generated randomly.

Preliminary results of our algorithm hinted at the need for additional diversity. It is observed that the crowding distance assignment is inadequate to prevent premature convergence. An attribute called duplication factor is introduced to each individual. In duplication factor assignment, every unique solution is ranked as one and their replicas are ranked with the number of times the same solution is observed in the current population. Thus, it is preferred to have small values for the duplication factor and in our modification, solutions are sorted primarily based on their duplication factor values and then sorted on the attributes proposed by NSGA-II.

Finally, the components of our GA are revised and necessary adaptations are made for the multi-objective case. Path crossover, binary tournament and local improvement procedures require some adaptations since they involve comparison of individuals. In the path crossover, after the swap moves the candidate that dominates the other is selected and included on the path. After the path is completed, candidate children created on the path are sorted based on non-dominated sorting criteria. Similarly in binary tournament, non-dominated sorting criteria are used for parent selection. In local improvement, we move to a neighbour solution only if it dominates the current solution. Immigration is implemented exactly in the same way as in our single objective GA implementation. Our GA for the multi-objective case is presented in Algorithm 1.

Algorithm 1 Pseudo code of multi-objective GA

- 1: Initialize *populationSize*, *maxIteration*, *positionHistory*
- 2: Initialize *immigrationRate*, *localImprovementRate*
- 3: Initialize *iterationCount* \leftarrow 1
- 4: Generate initial *population*: P
- 5: Update *bestSolution* and *positionHistory*

```

6: repeat
7:   Initialize parentPopulation:  $P_{par} \leftarrow \emptyset$ , childPopulation:  $P_{chi} \leftarrow \emptyset$ 
8:   Select  $P_{par}$  from population
9:   Generate  $P_{chi}$  from  $P_{par}$  with Crossover
10:  Apply non-dominated sorting on  $P_{chi}$ 
11:  Update bestSolution and positionHistory
12:  Apply Immigration to  $P_{chi}$ 
13:  Apply Local Improvement to  $P_{chi}$ 
14:  Update bestSolution and positionHistory
15:  Combine  $P$  and  $P_{chi}$  as combinedPopulation:  $P_{com}$ 
16:  Apply non-dominated sorting on  $P_{com}$ 
17:   $P \leftarrow$  best populationSize solutions in  $P_{com}$ 
18:  Update bestSolution and positionHistory
19:  iterationCount  $\leftarrow$  iterationCount + 1
20: until iterationCount < maxIteration
    
```

5. Numerical results and performance of the proposed solution methods

Our algorithms are coded in C# programming language in Microsoft Visual Studio 2015. All experiments are carried out on a PC with 3.60 GHz Intel® Core™ i7-3820 CPU and 16 GB RAM, running under 64-bit Windows 10 operating system. In order to improve the solution quality and reduce the effect of the randomness, the results are replicated with 10 different seeds.

An initial set of experiments is carried out to set GA strategies and parameters. The final GA is established with binary tournament selection and path swap crossover. The immigration rate is 0.4 and local improvement rate is 0.2. The population size is determined as 100 and the algorithm terminates after 100 generations.

This final form of the single objective GA is validated using some classical QAP instances with different sizes and characteristics that are taken from QAPLIB (<http://anjios.mgi.polymtl.ca/qaplib/inst.html>). In our validation phase, 25 well-studied problem instances referenced by numerous researches are selected. Among these, 12 problems with distance matrices satisfying triangular inequality are used further in our numerical experimentation phase for the uncertainty cases. More detailed information on the problem instances is available in Burkard et al. (1997).

The validation of our GA is made by comparing the best objective function value found through 10 replications with the optimal objective value provided in QAPLIB. For the last four problems, the comparison is made with the best upper bound values available in QAPLIB. The results are presented in Table 1. The percentage gap is equal to zero in almost all of the problems. Only for the problems with size larger than 36, the gap is larger than zero and even so, the gap takes values about only 1%. As a result, it can be claimed that with this operator setting and parameter tuning, our single objective GA is successful in solving the QAP effectively in reasonable times.

For the numerical studies concerning different QAP formulations under scenario-based uncertainty, problems with three scenarios are used. To obtain three scenarios, two additional flow matrices are generated by randomly disrupting the existing flow matrices of the 12 test problems that are chosen from QAPLIB. The disruptions are made by

Table 1. Validation of the proposed GA (computation times are cumulative of 10 replications).

Problem name	Problem size	Optimal value	Best found by GA	%Gap	Time (sec)
chr12a	12	9552	9552	0.0	84
tai12a	12	224,416	224,416	0.0	65
scr15	15	51,140	51,140	0.0	108
esc16a	16	68	68	0.0	108
had16	16	3720	3720	0.0	123
els19	19	17,212,548	17,212,548	0.0	581
had20	20	6922	6922	0.0	339
nug20	20	2570	2570	0.0	261
rou20	20	725,522	725,522	0.0	352
scr20	20	110,030	110,030	0.0	521
chr25a	25	3796	3796	0.0	1199
nug25	25	3744	3744	0.0	853
bur26a	26	5,426,670	5,426,670	0.0	1979
kra30a	30	88,900	88,900	0.0	2648
kra30b	30	91,420	91,420	0.0	2711
nug30	30	6124	6124	0.0	2424
tai30b	30	637,117,113	637,117,113	0.0	5245
tho30	30	149,936	149,936	0.0	2627
esc32a	32	130	130	0.0	2012
kra32	32	88,700	88,700	0.0	3469
ste36a	36	9526	9526	0.0	8169
tai40a	40	3,139,370 ^a	3,178,108	1.2	4237
sko72	72	66,256 ^a	66,346	0.1	13,082
wil100	100	273,038 ^a	273,786	0.3	26,470
tho150	150	8,133,398 ^a	8,216,038	1.0	23,840

^aBest feasible objective values provided in QAPLIB.

randomly switching the numbers in the existing flow matrix and multiplying them by uniformly distributed random variables. Uniform distribution between $[0, 1]$ is used for the second scenario and uniform distribution between $[1, 2]$ is used for the third scenario. In this manner, the second scenario represents the case where the demand is low and third scenario represents the high demand case. Three scenarios seem to be adequate to represent most real-life applications, e.g. modelling the demand in three levels as in our case. Additionally, since three objectives allow visualization, it is easier to inspect the behaviour of different approaches. In all the problem instances, both distance and flow matrices are defined as symmetrical and the diagonal entities are always equal to zero. All the test problems used in our numerical studies are available from <http://www.bufram.boun.edu.tr/sceQAP.zip>.

In this phase of experimentation, our algorithm is organized to solve each of the five models introduced in Section 3. As a result, five different GAs are constituted each focusing on a different performance measure. The names of the algorithms are given in accordance with the performance measure that they are dealing with: expQAP, maxQAP, absQAP, relQAP and *m*QAP. The only difference between the first four algorithms is in the way the fitness calculation is made, whereas in *m*QAP algorithm, the modifications discussed in Section 4.2 are made and our GA is transformed into a MOEA which is based on the NSGA-II. It should be kept in mind that the classical QAPs corresponding to each scenario in our problem instances are solved beforehand since c^*_s values are necessary for the formulations containing the regret criteria and the solutions found are used in the initial populations of all five algorithms.

To validate scenario-based approaches, test instances with sizes 8, 10 and 12 are generated and the true Pareto fronts for these test instances are obtained by enumerating all

Table 2. Optimum solutions for small-sized problems used for verification of proposed algorithms.

Problem name	Expected cost	Max. cost	Max. absolute regret	Max. relative regret
had8_3sce	642.67	928	10	0.022
nug8_3sce	390.00	544	52	0.160
scr8_3sce	31,726.67	42,782	5536	0.205
had10_3sce	1150.67	1806	124	0.223
nug10_3sce	589.33	858	62	0.120
scr10_3sce	39,140.00	53,042	6092	0.171
had12_3sce	1810.67	2794	54	0.037
nug12_3sce	1035.33	1512	108	0.130
scr12_3sce	44,794.00	61,208	6,262	0.161

permutations. We must recall that the total number of solutions in the feasible space of a QAP of size n is $n!$ and the true Pareto front can only be obtained for the problems with small sizes. We record the best solution in the front corresponding to each criterion and compare it to the solution provided by the corresponding GA for verification. The optimum solutions for these problems used for verification are provided in Table 2.

Each of our proposed algorithms finds the optimal solution for the corresponding performance metric in all of these small-sized problems while our m QAP algorithm is capable of finding the optimal solutions for all of the performance metrics. On the other hand, the performance of our algorithms should also be evaluated through larger problem instances. For those instances, a cross validation is implemented in the absence of a true Pareto front.

The performances of five models are evaluated and compared with one another in terms of their success in generating good solutions for minimizing the criteria: *expected cost*, *maximum cost*, *maximum absolute regret* and *maximum relative regret*. The results are provided in Tables 3–6 in a respective order for each performance criterion. For each problem instance, the best results over 10 replications are shown in bold. For each criterion, the solution in the final population of each algorithm with the best objective value with respect to the corresponding criterion is provided. In that way, our analysis tries to capture whether each of these algorithms is justifiable on their own right and whether they are distinct in behaviour and leading us to different solutions where robustness is interpreted in a different way. In other words, we have targeted to find an answer to the question whether it were possible to find good solutions for all of these performance criteria in the final population of one of these procedures.

Table 3. Comparison of the proposed GA models (criterion: expected cost).

Problem name	expQAP	maxQAP	absQAP	relQAP	m QAP
scr15_3sce	61,714.67	64,913.33	61,822.00	62,283.33	61,714.67
had16_3sce	3953.33	3984.00	3974.67	3983.33	3953.33
els19_3sce	18,679,666.67	20,036,724.67	19,193,050.00	21,023,503.33	18,679,666.67
had20_3sce	7266.67	7375.33	7272.00	7290.67	7266.67
nug20_3sce	2928.67	3029.33	2946.00	2960.67	2928.67
scr20_3sce	134,568.00	142,548.00	136,284.67	139,763.33	134,568.00
nug25_3sce	4260.67	4460.00	4302.67	4295.33	4260.67
kra30a_3sce	108,613.33	113,826.67	109,796.67	109,646.67	109,380.00
kra30b_3sce	110,433.33	114,733.33	111,163.33	112,946.67	110,730.00
nug30_3sce	6946.00	7144.67	6989.33	7034.00	6946.00
tho30_3sce	159,854.67	163,214.00	162,412.00	165,139.33	160,353.33
ste36a_3sce	12,524.00	13,648.00	12,766.67	12,853.33	12,714.00

Note: Best results for each problem are shown in bold.

Table 4. Comparison of the proposed GA models (criterion: maximum cost).

Problem name	expQAP	maxQAP	absQAP	relQAP	mQAP
scr15_3sce	77,574	69,710	77,234	81,358	69,710
had16_3sce	5824	5724	5894	5972	5724
els19_3sce	25,571,140	24,211,326	27,113,016	32,167,018	24,211,326
had20_3sce	10,510	10,076	10,404	10,570	10,076
nug20_3sce	4064	3736	4016	4162	3736
scr20_3sce	172,974	152,892	170,814	183,086	157,122
nug25_3sce	6042	5648	6010	6154	5670
kra30a_3sce	141,950	136,430	148,650	154,430	136,780
kra30b_3sce	146,210	139,550	150,570	158,160	140,050
nug30_3sce	9566	9364	9894	10,140	9364
tho30_3sce	235,650	225,308	241,200	254,108	226,076
ste36a_3sce	17,006	16,148	18,286	19,872	16,058

Note: Best results for each problem are shown in bold.

The results reveal that each of the first four formulations leads to distinct solutions. They are good at providing solutions for the criterion they originally address but not for the other criteria. For instance, we observe in Table 3 that in order to obtain solutions with the least expected cost values, the formulation that is dedicated to find expected cost optimum must be employed and other single objective formulations can obtain the best results in only a few problem instances. Similar conclusions can be made for the rest of the performance criteria in Tables 4–6 about the single objective formulations.

It should be noted that some exceptions can be observed where an algorithm cannot obtain the best result corresponding to its own criterion. This occurs for some of the problem instances with large sizes (for instance in Table 5). As the problem size increases, the number of feasible solutions and the number of non-dominated solutions also increase. As a result, such exceptions can be observed. Furthermore, in some specific instances, good solutions might have similar genotype. In this case, the search might lead to similar regions under different fitness measures. However, it should also be noted that mQAP is the winner in most of these cases and when it is excluded from the analysis, the number of such exceptions diminishes. As a result, it can be concluded that none of single criterion algorithms can substitute the other. The decision-maker needs to determine which robustness measure to optimize a priori and then employ its corresponding algorithm.

On the other hand, the case is very different when the results of the mQAP algorithm are examined in these tables. Among all members of the approximated Pareto front

Table 5. Comparison of the proposed GA models (criterion: maximum absolute regret).

Problem name	expQAP	maxQAP	absQAP	relQAP	mQAP
scr15_3sce	9762	17,460	806	12,534	8606
had16_3sce	222	326	170	248	172
els19_3sce	4,799,058	6,838,840	3,739,512	8,181,322	3,758,216
had20_3sce	434	714	336	494	338
nug20_3sce	328	654	294	426	294
scr20_3sce	29,698	50,994	30,608	39,192	29,164
nug25_3sce	394	784	362	506	352
kra30a_3sce	12,120	25,020	12,460	17,820	12,480
kra30b_3sce	12,700	25,500	11,820	18,510	11,450
nug30_3sce	548	1118	548	776	524
tho30_3sce	20,300	27,542	15,892	28,800	15,278
ste36a_3sce	2516	5936	2606	4186	2690

Note: Best results for each problem are shown in bold.

Table 6. Comparison of the proposed GA models (criterion: maximum relative regret)

Problem name	expQAP	maxQAP	absQAP	relQAP	mQAP
scr15_3sce	0.191	0.404	0.197	0.182	0.182
had16_3sce	0.060	0.099	0.081	0.048	0.048
els19_3sce	0.794	0.883	0.608	0.347	0.347
had20_3sce	0.068	0.186	0.087	0.051	0.051
nug20_3sce	0.145	0.390	0.173	0.116	0.116
scr20_3sce	0.349	0.659	0.388	0.292	0.275
nug25_3sce	0.131	0.289	0.133	0.090	0.085
kra30a_3sce	0.156	0.362	0.151	0.130	0.131
kra30b_3sce	0.156	0.376	0.174	0.135	0.125
nug30_3sce	0.136	0.246	0.141	0.093	0.092
tho30_3sce	0.312	0.424	0.233	0.128	0.131
ste36a_3sce	0.390	0.706	0.484	0.267	0.272

Note: Best results for each problem are shown in bold.

provided by the *mQAP* algorithm, the one with the best objective value of with respect to the performance criterion under investigation is reported in the last column of the tables. The results indicate that the Pareto front approximation proposed by the *mQAP* algorithm includes such solutions that perform well in terms of different performance measures.

The improvement gained by our algorithms in different performance measures is also tested against a base case as follows. In addition to the solutions of each scenario, we find the solution for the average scenario \bar{s} where $f_{ij\bar{s}} = \frac{\sum_{s \in S} f_{ij}^s}{|S|}$. Then, the best of these solutions (for \bar{s} and all scenarios $s \in S$) are evaluated for each criterion except for the expected cost criterion (note that the solution for the average scenario s is identical to the solution *expQAP*). The results indicate that our algorithms bring a significant improvement, especially for the robust measures. Also it must be noted that the computational effort for this base case is the same with the other approaches since it also requires $|S| + 1$ GA runs.

The quality of our *mQAP* algorithm can also be observed in a more compact way in Table 7. Each column displays results for a different performance measure. The values are the ratios of the objective values provided by the *mQAP* algorithm over of the objective values suggested by one of the other four algorithms that is dedicated to that performance criterion. The ratios that are less than or equal to one are shown in bold. A value less than one indicates that the *mQAP* objective algorithm is capable of finding a member in its

Table 7. Ratio analysis for the *mQAP* model.

Problem name	Expected cost	Maximum cost	Maximum absolute regret	Maximum relative regret
scr15_3sce	1.000	1.000	1.000	1.000
had16_3sce	1.000	1.000	1.012	1.000
els19_3sce	1.000	1.000	1.005	1.000
had20_3sce	1.000	1.000	1.006	1.000
nug20_3sce	1.000	1.000	1.000	1.000
scr20_3sce	1.000	1.028	0.953	0.940
nug25_3sce	1.000	1.004	0.972	0.954
kra30a_3sce	1.007	1.003	1.002	1.006
kra30b_3sce	1.003	1.004	0.969	0.926
nug30_3sce	1.000	1.000	0.956	0.993
tho30_3sce	1.003	1.003	0.961	1.026
ste36a_3sce	1.015	0.994	1,032	1.018

Note: Ratios that are less than or equal to one are shown in bold.

Table 8. Ratio analysis for the *m*QAP model under six scenarios.

Problem name	Maximum cost	Maximum absolute regret	Maximum relative regret
scr15_6sce	1.002	1.029	1.042
had16_6sce	1.036	1.132	1.140
els19_6sce	1.099	1.111	1.173
had20_6sce	1.020	1.059	1.028
nug20_6sce	1.098	1.067	1.024
scr20_6sce	1.177	1.047	1.247
nug25_6sce	1.226	1.224	1.210
kra30a_6sce	1.215	1.041	1.162
kra30b_6sce	1.041	1.141	1.144
nug30_6sce	1.266	1.131	1.132
tho30_6sce	1.022	1.306	1.014
ste36a_6sce	1.207	1.319	1.184

Pareto front such that the member performs better than the solutions proposed by other algorithms. When the ratio equals to one, it means that the algorithms lead up to the same solution. It is observed that the ratios which are greater than one are quite close to one. As a final remark, it must be noted that the time requirement for all five algorithms is very close to each other. It means that instead of using different algorithms each for a different performance measure, we can use the *m*QAP algorithm and obtain the Pareto front which is composed of good compromises between the scenario objectives and includes the successful solutions for every performance measure.

5.1. Larger number of scenarios

Although three scenarios can be adequate to model uncertainty in many real-life applications, there may be several applications where larger number of scenarios are required. When the number of objectives is more than three, the performances of the MOEAs deteriorate and they face scalability issues (Ishibuchi et al. 2015). In this subsection, we test our multi-objective GA under six scenarios. We increase the number of scenarios in our test problems to six by randomly generating three more flow matrices representing different demand levels between the low and the high demand cases. The ratios of the objective values for the robust measures under six scenarios are presented in Table 8.

It can be observed from Table 8 that our multi-objective GA based on NSGA-II starts facing scalability issues and the ratios can rise up to 1.319. This suggests that using *many-objective* evolutionary algorithms can be more appropriate when the number of scenarios increases (Jaimes and Coello 2015). Additional work is required to develop a many-objective evolutionary algorithm that solves QAP instances with large number of scenarios effectively.

6. Conclusion

Although uncertainty in the input parameters of a facility layout problem is quite common, this is often overlooked during design. This study addresses scenario-based uncertainty in a facility layout problem formulated as a QAP and investigates different ways of dealing with uncertainty to design a facility layout which attains robust and efficient performance under all possible scenarios. For this purpose, four mathematical formulations each covering a different stochastic or robust performance measure are developed and

solved by an efficient GA. We observe that the proposed GA procedure is capable of finding very good solutions to the problem under a given target performance measure. Moreover, our GA is designed to prefer efficient solutions in case there are alternative solutions with the same worst-case performance.

Given the difficulty of solving a QAP-based formulation by any method, the main effort in this study has been put into finding ways of decreasing the necessity to set the robustness measure a priori, or obtaining the solutions for several measures simultaneously. However, the idea of exploiting the population-based parallelism of the GA approach did not help much in this respect in the case of the single objective GA procedures, i.e. the members of the final populations did not perform too well for any measure other than the targeted one. To this end, we propose a novel approach, where the optimization problem under scenario-based uncertainty is transformed into a multi-objective optimization problem by considering each scenario as a separate objective. The motivation behind this transformation comes from the strong relationship between robustness measures and the multi-objective formulations as pointed out by Aissi et al. (2009) and Klamroth et al. (2013). Instead of evaluating each performance measure separately we can approximate the Pareto front in a single run of a MOEA and let the decision-maker select the appropriate member.

For this purpose, we developed a multi-objective GA adapted from NSGA-II introduced by Deb et al. (2002). Our numerical results show that the approximated Pareto front generated by this approach provides plenty of good quality solutions that include the very good solutions for all robustness measures. This is a very promising result which was not previously investigated in the literature.

Furthermore, the observation obtained about the potential of the multi-objective version for the QAP may be generalized to other combinatorial optimization problems under scenario-based uncertainty. It is hoped that this study will stimulate further investigation in this topic and bring a new and natural application area into multi-objective optimization field.

Finally, it is clear that additional work is required for the proposed MOEA to solve problem instances with large number of scenarios effectively. Scalability issues bring many-objective evolutionary algorithms into attention when the number of scenarios is high. Many-objective optimization methodologies which are still in the development stage may be adapted for handling scenario-based uncertainty successfully. This is the focus of our on-going research.


Acknowledgments

This research is supported by Boğaziçi University Research Fund [grant number 13843].

Disclosure statement

No potential conflict of interest was reported by the authors.

ORCID

Mert Şahinkoç  <http://orcid.org/0000-0001-8054-7767>

References

- Ahmed ZH. 2016. Experimental analysis of crossover and mutation operators on the quadratic assignment problem. *Ann Oper Res.* 247(2):833–851.
- Ahuja RK, Orlin JB, Tiwari A. 2000. A greedy genetic algorithm for the quadratic assignment problem. *Comput Oper Res.* 27(10):917–934.
- Aissi H, Bazgan C, Vanderpooten D. 2009. Min–max and min–max regret versions of combinatorial optimization problems: a survey. *Eur J Oper Res.* 197(2):427–438.
- Benjaafar S, Sheikhzadeh M. 2000. Design of flexible plant layouts. *IIE Trans.* 32(4):309–322.
- Ben-Tal A, El Ghaoui L, Nemirovski A. 2009. *Robust optimization*. New Jersey: Princeton University Press.
- Ben-Tal A, Nemirovski A. 2002. Robust optimization-methodology and applications. *Math Program.* 92(3):453–480.
- Beyer H, Sendhoff B. 2007. Robust optimization – a comprehensive survey. *Comput Methods Appl Mech Eng.* 196(33):3190–3218.
- Burkard RE, Karisch SE, Rendl F. 1997. QAPLIB – a quadratic assignment problem library. *J Global Optim.* 10(4):391–403.
- Day RO, Lamont GB. 2005. Multiobjective quadratic assignment problem solved by an explicit building block search algorithm – MOMGAlIa. In: Raidl GR, Gottlieb J, editors. *European Conference on Evolutionary Computation in Combinatorial Optimization*. Berlin Heidelberg: Springer-Verlag; Lecture Notes in Computer Science; p. 91–100.
- Deb K, Gupta H. 2005. Searching for robust pareto-optimal solutions in multi-objective optimization. *Lect Notes Comput Sci.* 34(10):150–164.
- Deb K, Pratap A, Agarwal S, Meyarivan T. 2002. Fast and elitist multi-objective genetic algorithm: NSGA-II. *Evolutionary computation*. *IEEE Trans.* 6(2):182–197.
- Drezner Z. 2005. Compounded genetic algorithms for the quadratic assignment problem. *Oper Res Lett.* 33(5):475–480.
- Drezner Z. 2008. Extensive experiments with hybrid genetic algorithms for the solution of the quadratic assignment problem. *Comput Oper Res.* 35(3):717–736.
- Feizollahi MJ, Averbakh I. 2014. The robust (minmax regret) quadratic assignment problem with interval flows. *INFORMS J Comput.* 26(2):321–335.
- Feizollahi MJ, Feyzollahi H. 2015. Robust quadratic assignment problem with budgeted uncertain flows. *Oper Res Perspect.* 2:114–123.
- Feizollahi MJ, Modarres M. 2012. Robust quadratic assignment problem with uncertain locations. *Iran J Oper Res.* 3(2):46–65.
- Glover F, Laguna M. 1999. *Tabu search*. New York (NY): Springer.
- Hahn PM, Zhu YR, Guignard M, Hightower WL, Saltzman MJ. 2012. A level-3 reformulation-linearization technique-based bound for the quadratic assignment problem. *INFORMS J Comput.* 24(2):202–209.
- Iancu DA, Trichakis N. 2013. Pareto efficiency in robust optimization. *Manage Sci.* 60(1):130–147.
- Ishibuchi H, Akedo N, Nojima Y. 2015. Behavior of multiobjective evolutionary algorithms on many-objective knapsack problems. *IEEE Trans Evol Comput.* 19(2):264–283.
- Jaimes AL, Coello CAC. 2015. Many-objective problems: challenges and methods. Kacprzyk J, Pedrycz W, editors. *Springer handbook of computational intelligence*. Berlin Heidelberg: Springer; p. 1033–1046.
- Kaufman L, Broeckx F. 1978. An algorithm for the quadratic assignment problem using Bender's decomposition. *Eur J Oper Res.* 2(3):207–211.
- Klamroth K, Köbis E, Schöbel A, Tammer C. 2013. A unified approach for different concepts of robustness and stochastic programming via non-linear scalarizing functionals. *Optimization.* 62(5):649–671.
- Kleeman MP, Day RO, Lamont GB. 2004. Analysis of a parallel MOEA solving the multi-objective quadratic assignment problem. In: Deb K, et al., editors. *Genetic and evolutionary computation – GECCO 2004. Proceedings of the Genetic and Evolutionary Computation Conference.*

- Vol. 3103, Part II. Seattle (WA): Springer-Verlag; Lecture Notes in Computer Science; p. 402–403.
- Knowles JD, Corne D. [2002](#). Towards landscape analyses to inform the design of a hybrid local search for the multiobjective quadratic assignment problem. In: Abraham A, Ruiz-del-Solar J, Koppen M, editors. *Soft computing systems: design, management and applications*. Amsterdam: IOS Press; p. 271–279.
- Knowles JD, Corne D. [2003](#). Instance generators and test suites for the multiobjective quadratic assignment problem. In: Fonseca CM, Fleming P, Zitzler E, Deb K, Thiele L, editors. *Evolutionary multi-criterion optimization (EMO 2003)*, Vol. 2632 of Lecture Notes in Computer Science. Berlin Heidelberg: Springer-Verlag; p. 295–310.
- Kouvelis P, Kurawarwala AA, Gutierrez GJ. [1992](#). Algorithms for robust single and multiple period layout planning for manufacturing systems. *Eur J Oper Res*. 63(2):287–303.
- Kouvelis P, Yu G. [1997](#). *Robust discrete optimization and its applications*. New York (NY): Springer.
- Loiola EM, de Abreu NMM, Boaventura-Netto PO, Hahn P, Querido T. [2007](#). A survey for the quadratic assignment problem. *Eur J Oper Res*. 176(2):657–690.
- López-Ibáñez M, Paquete L, Stützle T. [2004](#). On the design of ACO for the bi-objective quadratic assignment problem. *Ant colony optimization and swarm intelligence*. New York (NY): Springer.
- Misevicius A. [2003](#). Genetic algorithm hybridized with ruin and recreate procedure: application to the quadratic assignment Problem. *Knowl-Based Syst*. 16(5):261–268.
- Misevicius A. [2004](#). An improved hybrid genetic algorithm: new results for the quadratic assignment problem. *Knowl-Based Syst*. 17(2):65–73.
- Nyberg A, Westerlund T. [2012](#). A new exact discrete linear reformulation of the quadratic assignment problem. *Eur J Oper Res*. 220(2):314–319.
- Özkale C, Fiğlalı A. [2013](#). Evaluation of the multiobjective ant colony algorithm performances on biobjective quadratic assignment problems. *Appl Math Model*. 37(14):7822–7838.
- Paquete L, Stützle T. [2006](#). A study of stochastic local search algorithms for the bi-objective QAP with correlated flow matrices. *Eur J Oper Res*. 169(3):943–959.
- Pardalos L, Resende M. [1994](#). A greedy randomized adaptive search procedure for the quadratic assignment problem. *Discrete Math Theor Comput Sci*. 16:237–261.
- Resende MG, Ramakrishnan K, Drezner Z. [1995](#). Computing lower bounds for the quadratic assignment problem with an interior point algorithm for linear programming. *Oper Res*. 43(5):781–791.
- Rosenblatt MJ, Lee HL. [1987](#). A robustness approach to facilities design. *Int J Prod Res*. 25(4):479–486.
- Rostami B, Malucelli F. [2014](#). A revised reformulation-linearization technique for the quadratic assignment problem. *Discrete Optim*. 14:97–103.
- Sahni S, Gonzalez T. [1976](#). P-complete approximation problems. *J ACM (JACM)*. 23(3):555–565.
- Shapiro A, Dentcheva , Ruszczyński A. [2009](#). *Lectures on stochastic programming: modeling and theory*. Philadelphia (PA): SIAM.
- Snyder LV. [2006](#). Facility location under uncertainty: a review. *IIE Trans*. 38(7):547–564.
- Snyder LV, Daskin MS. [2006](#). Stochastic p -robust location problems. *IIE Trans*. 38(11):971–985.
- Taillard ED. [1995](#). Comparison of iterative searches for the quadratic assignment problem. *Locat Sci*. 3(2):87–105.
- Tate DM, Smith AE. [1995](#). A genetic approach to the quadratic assignment problem. *Comput Oper Res*. 22(1):73–83.
- Tosun U, Dokeroglu T, Cosar A. [2013](#). A robust island parallel genetic algorithm for the quadratic assignment problem. *Int J Prod Res*. 51(14):4117–4133.
- Wu L, Shahidehpour M, Li Z. [2012](#). Comparison of scenario-based and interval optimization approaches to stochastic SCUC. *IEEE Trans Power Syst*. 27(2):913–921.
- Xia Y, Yuan YX. [2006](#). A new linearization method for quadratic assignment problems. *Optim Method Softw*. 21(5):805–818.

- Zhang H, Beltran-Royo C, Ma L. 2013. Solving the quadratic assignment problem by means of general purpose mixed integer linear programming solvers. *Ann Oper Res.* 207(1):261–278.
- Zhao Y, Wallace SW. 2014. Integrated facility layout design and flow assignment problem under uncertainty. *INFORMS J Comput.* 26(4):798–808.
- Zitzler E, Deb K, Thiele L. 2000. Comparison of multi-objective evolutionary algorithms: empirical results. *Evol Comput.* 8(2):173–195.