A Hybrid Metaheuristic for Multiobjective Unconstrained Binary Quadratic Programming

Arnaud Liefooghe[*,a,b], Sébastien Verel[c], Jin-Kao Hao[d]

[a]*LIFL, Université Lille 1, UMR CNRS 8022, Cité scientifique, Bât. M3, 59655 Villeneuve d'Ascq cedex, France*
[b]*Inria Lille-Nord Europe, Parc Scientifique de la Haute Borne, 40 av. Halley, 59650 Villeneuve d'Ascq, France*
[c]*LISIC, Université du Littoral Côte d'Opale, 50 rue F. Buisson, 62228 Calais cedex, France*
[d]*LERIA, Université d'Angers, 2 bd. Lavoisier, 49045 Angers, France*

Abstract

The conventional Unconstrained Binary Quadratic Programming (UBQP) problem is known to be a unified modeling and solution framework for many combinatorial optimization problems. This paper extends the single-objective UBQP to the multiobjective case (mUBQP) where multiple objectives are to be optimized simultaneously. We propose a hybrid metaheuristic which combines an elitist evolutionary multiobjective optimization algorithm and a state-of-the-art single-objective tabu search procedure by using an achievement scalarizing function. Finally, we define a formal model to generate mUBQP instances and validate the performance of the proposed approach in obtaining competitive results on large-size mUBQP instances with two and three objectives.

Key words: Unconstrained binary quadratic programming, Multiobjective combinatorial optimization, Hybrid Metaheuristic, Evolutionary Multiobjective Optimization, Tabu search, Scalarizing function

1. Introduction

Given a collection of n items such that each pair of items is associated with a profit value that can be positive, negative or zero, unconstrained binary quadratic programming (UBQP) seeks a subset of items that maximizes the sum of their paired values. The value of a pair is accumulated in the sum only if the two corresponding items are selected. A feasible solution to a UBQP instance can be specified by a binary string of size n, such that each variable indicates whether the corresponding item is included in the selection or not. More formally, the conventional and single-objective UBQP problem is to maximize the following objective function.

$$f(x) = x'Qx = \sum_{i=1}^{n}\sum_{j=1}^{n} q_{ij}x_i x_j \qquad (1)$$

where $Q = (q_{ij})$ is an n by n matrix of constant values, x is a vector of n binary (zero-one) variables, *i.e.*, $x_i \in \{0, 1\}$, $i \in \{1, \ldots, n\}$, and x' is the transpose of x.

The UBQP is known to be a general model able to represent a wide range of important problems, including those from financial analysis [1], social psychology [2], computer aided design [3] and cellular radio channel allocation [4]. Moreover, a number of NP-hard problems can be conveniently transformed into the UBQP, such as graph coloring, maxcut, set packing, set partitioning, maximum clique, and so on [5, 6]. As a consequence, the UBQP itself is clearly a NP-hard problem [7]. During the past few decades, a large number of algorithms and approaches have been proposed for the single-objective UBQP in the literature. This includes several exact methods based on branch and bound or branch and cut [8, 9, 10] and a number of heuristic and metaheuristic methods like simulated annealing [11], tabu search [12, 13, 14, 15, 16], path-relinking [17], evolutionary and memetic algorithms [18, 19, 20, 21].

In this paper, we extend this conventional single-objective UBQP problem to the multiobjective case, denoted by mUBQP, where multiple objectives are to be optimized simultaneously. Such an extension naturally increases the expressive ability of the UBQP and provides a convenient formulation to fit situations which the single-objective UBQP cannot accommodate. For instance, UBQP can recast the vertex coloring problem (of determining the chromatic number of a graph) [5] and the sum coloring problem (of determining the chromatic sum of a graph) [22]. Still, UBQP is not convenient to formulate the bi-objective coloring problem which requires to determine a legal vertex coloring of a graph while minimizing simultaneously the number of colors used and the sum of colors. For this bi-objective coloring problem, the mUBQP formulation can be employed in a straightforward way.

In addition of introducing the mUBQP problem, the paper has two contributions. First, given that the single-objective UBQP is NP-hard, its generalized mUBQP formulation is also a difficult problem to solve in the general case. For the purpose of approximating the Pareto set of a given mUBQP instance, heuristic approaches are appealing. Following the studies on memetic algorithms for the UBQP and many other problems, we adopt as our solution approach the memetic framework and propose a hybrid metaheuristic which combines an elitist evolutionary multiobjective optimization algorithm with

[*]Corresponding author, Tel.: +33 3 59 35 86 30.
Email addresses: arnaud.liefooghe@univ-lille1.fr (Arnaud Liefooghe), verel@lisic.univ-littoral.fr (Sébastien Verel), hao@info.univ-angers.fr (Jin-Kao Hao)

a state-of-the-art single-objective tabu search procedure based on an achievement scalarizing function. The last contribution of this work is to define a formal and flexible model to generate hard mUBQP instances. An experimental analysis validates the effectiveness of the proposed hybrid metaheuristic by achieving a clear improvement over non-hybrid and conventional algorithms on large-size mUBQP instances with two and three objectives.

The paper is organized as follows. Section 2 introduces the multiobjective formulation of the UBQP problem (mUBQP). Section 3 presents the hybrid metaheuristic (HM) proposed for the mUBQP problem and its main ingredients, including the scalarizing evaluation function, the tabu search procedure, the initialization phase and the variation operators. Section 4 gives an experimental analysis of the HM algorithm on a large set of mUBQP instances of different structure and size. The last section concludes the paper and suggests further research lines.

2. Multiobjective Unconstrained Binary Quadratic Programming

This section first introduces the multiobjective unconstrained binary quadratic programming problem. Some definitions related to multiobjective combinatorial optimization are then recalled, followed by problem complexity-related properties and a link with similar problem formulations. Last, the construction of problem instances, together with an experimental study on the correlation of objective values and the cardinality of the Pareto set, are presented.

2.1. Problem Formulation

The multiobjective unconstrained binary quadratic programming (mUBQP) problem can be stated as follows.

$$\begin{aligned} \max \ \ f_k(x) &= \sum_{i=1}^{n}\sum_{j=1}^{n} q_{ij}^k x_i x_j \qquad && k \in \{1,\ldots,m\} \\ \text{subject to } x_i &\in \{0,1\} && i \in \{1,\ldots,n\} \end{aligned} \tag{2}$$

where $f = (f_1, f_2, \ldots, f_m)$ is an objective function vector with $m \geqslant 2$, n is the problem size, and we have m matrices $Q^k = (q_{ij}^k)$ of size n by n with constant positive, negative or zero values, $k \in \{1,\ldots,m\}$. The solution space X is defined on binary strings of size n.

2.2. Definitions

Let $X = \{0,1\}^n$ be the set of feasible solutions in the *solution space* of Problem (2). We denote by $Z \subseteq \mathbb{R}^m$ the feasible region in the *objective space*, *i.e.*, the image of feasible solutions when using the maximizing function vector f. The Pareto dominance relation is defined as follows. A solution $x \in X$ is dominated by a solution $x' \in X$, denoted by $x \prec x'$, if $f_k(x) \leqslant f_k(x')$ for all $k \in \{1,\ldots,m\}$, with at least one strict inequality. If neither $x \nprec x'$ nor $x' \nprec x$ holds, then both solutions are *mutually non-dominated*. A solution $x \in X$ is *Pareto optimal* (or efficient, non-dominated) if there does not exist any other solution $x' \in X$ such that x' dominates x. The set of all Pareto optimal solutions is called the *Pareto set*, denoted by X_{PS}, and its mapping in the objective space is called the *Pareto front*. One of the most challenging issues in multiobjective combinatorial optimization is to identify a minimal complete Pareto set, *i.e.*, one Pareto optimal solution mapping to each point from the Pareto front. Note that such a set may not be unique, since multiple solutions can map to the same non-dominated vector.

2.3. Properties

For many multiobjective combinatorial optimization problems, computing the Pareto set is computationally prohibitive for two main reasons. First, the question of deciding if a candidate solution is dominated is known to be NP-hard for numerous multiobjective combinatorial optimization problems [23, 24]. This is also the case for the mUBQP problem since its single-objective counterpart is NP-hard [7]. Second, the cardinality of the Pareto front typically grows exponentially with the size of the problem instance [24]. In that sense, most multiobjective combinatorial optimization problems are said to be *intractable*. In the following, we prove that the mUBQP problem is intractable.

Proposition 1. *The multiobjective unconstrained binary quadratic programming problem (2) is intractable, even for* $m = 2$.

Proof. Consider the following bi-objective mUBQP instance.

$$q_{ij}^1 = \begin{cases} 2^{n(i-1)-\frac{i(i-1)}{2}+j-1} & \text{if } i \geqslant j \\ 0 & \text{if } i < j \end{cases} \qquad i,j \in \{1,\ldots,n\}$$

Let $q_{ij}^2 = -q_{ij}^1$ for all $i, j \in \{1,\ldots,n\}$. It is obvious that all solutions are mutually non-dominated, and that each solution maps to a different vector in the objective space. Therefore, $|Z_{PF}| = |X_{PS}| = |X| = 2^n$. □

The bi-objective mUBQP instance used in the proof is illustrated in Figure 1 for $n = 3$.

In order to cope with NP-hard and intractable multiobjective combinatorial optimization problems, researchers have developed approximate algorithms that identify a *Pareto set approximation* having both good convergence and distribution properties [25, 26]. To this end, metaheuristics in general, and evolutionary algorithms in particular, have received a growing interest since the late eighties [27].

2.4. Links with Existing Problem Formulations

The single-objective UBQP problem is of high interest in practice, since many existing combinatorial optimization problems can be formalized in terms of UBQP [5]. As a consequence, multiobjective versions of such problems can potentially be defined in terms of mUBQP. However, to the best of our knowledge, the UBQP problem has never been explicitly defined in the multiobjective formulation given in Eq. (2).

Q^1-matrix

q^1_{ij}	1	2	3
1	2^0	2^1	2^2
2	0	2^3	2^4
3	0	0	2^5

Feasible solutions

x	$(f_1$,	$f_2)$
000	(0 ,	0)
100	(1 ,	-1)
010	(8 ,	-8)
001	(32 ,	-32)
110	(11 ,	-11)
101	(37 ,	-37)
011	(56 ,	-56)
111	(63 ,	-63)

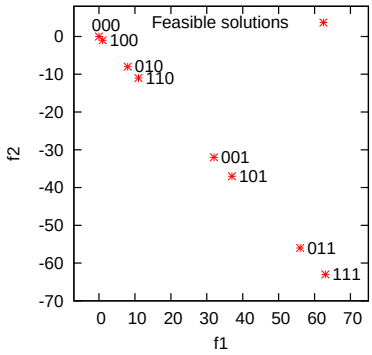


Figure 1: Enumeration of all feasible solutions for the mUBQP problem instance considered in the proof of Proposition 1: The input data of the Q^1-matrix (left), the enumeration of feasible solutions (middle), and their representation in the objective space (right). The problem size is $n = 3$.

Existing multiobjective formulations of classical combinatorial optimization problems with binary variables include multiobjective linear assignment problems [24, 28], multiobjective knapsack problems [29, 30], multiobjective maxcut problems [31], or multiobjective set covering and partitioning problems [28], just to mention a few. Nevertheless, the objective functions of such formulations are linear, and not quadratic as in mUBQP. Still, they often contain additional constraints; typically the unimodularity of the constraint matrix for linear assignment, or the capacity constraint for knapsack. This means that many existing binary multiobjective combinatorial optimization problems can be formalized in terms of mUBQP by adapting and generalizing the techniques from [5] to the multiobjective case, whereas the opposite does not hold in general due to the quadratic nature of mUBQP. The mUBQP problem is also different from the multiobjective quadratic assignment problem (mQAP) [32, 33], which seeks an assignment of n objects to n locations under multiple flow matrices. The solution representation is then usually based on a permutation for mQAP whereas it is based on a binary string for mUBQP.

2.5. Problem Instances

We propose to construct correlated mUBQP problem instances as follows. Each objective function is defined by means of a matrix Q^k, $k \in \{1, \ldots, m\}$. Based on the single-objective UBQP instances available in the OR-lib [34], non-zero matrix integer values are randomly generated according to a uniform distribution in $[-100, +100]$. As in the single-objective case, the density d gives the expected proportion of non-zero numbers in the matrix. In order to define matrices of a given density d, we set $q^k_{ij} = 0$ for all $k \in \{1, \ldots, m\}$ at the same time, following a Bernoulli distribution of parameter d.

Moreover, we define a correlation between the data contained in the m matrices Q^k, $k \in \{1, \ldots, m\}$. The positive (respectively negative) data correlation decreases (respectively increases) the degree of conflict between the objective function values. For simplicity, we use the same correlation between all pairs of objective functions, given by a correlation coefficient $\rho > \frac{-1}{m-1}$. The generation of correlated data follows a multivariate uniform law of dimension m [35]. In order to validate the behavior of the objective correlation coefficient experimentally, we conduct an empirical study for $n = 18$ in order to enumerate the feasible set $\{0, 1\}^n$ exhaustively. Figure 2 reports the average value of the Spearman correlation coefficient over 30 different and independent instances for different combinations of ρ, m, and d. Clearly, the correlation coefficient ρ tunes the objective correlation with a high accuracy.

To summarize, the four parameters used to define a mUBQP instance are (*i*) the problem size n, (*ii*) the matrix density d, (*iii*) the number of objective functions m, and (*iv*) the objective correlation coefficient ρ. The mUBQP problem instances used in the paper and an instance generator are available at the following URL: `http://mocobench.sf.net/`.

2.6. Cardinality of the Pareto Set

In this section, we analyze the impact of the mUBQP problem instance features (in particular, d, m and ρ) on the number of Pareto optimal solutions. The Pareto set cardinality plays an important role on the problem complexity (in terms of intractability), and then on the behavior and the performance of solution approaches. Indeed, the higher the number of Pareto optimal solutions, the more computational resources are required to identify a minimal complete Pareto set.

We set $n = 18$ in order to enumerate the feasible set $\{0, 1\}^n$ exhaustively. We report the average values over 30 different and independent mUBQP instances of same structure. Figure 3 gives the proportion of Pareto optimal solutions. Unsurprisingly, the matrix density d has a low influence on the results. However, the number of objective functions m and the objective correlation ρ both modify the proportion of Pareto optimal solutions to several orders of magnitude. Indeed, this proportion decreases from 10^{-4} for $\rho = -0.9$ to 10^{-5} for $\rho = +0.9$ for two- and three-objective mUBQP problem instances. As well, for a negative objective correlation $\rho = -0.2$, this proportion goes from 10^{-4} up to 10^{-1}, whereas it goes from 10^{-5} up to 10^{-3} for a positive objective correlation $\rho = +0.9$, for $m = 2$ and $m = 5$, respectively. Figure 4 shows three examples of mUBQP problem instances represented in a two-objective space. When the objective correlation is negative, the objective functions are in conflict, and the Pareto front is large (left). When the objective correlation is zero, the image of the feasible set in the objective

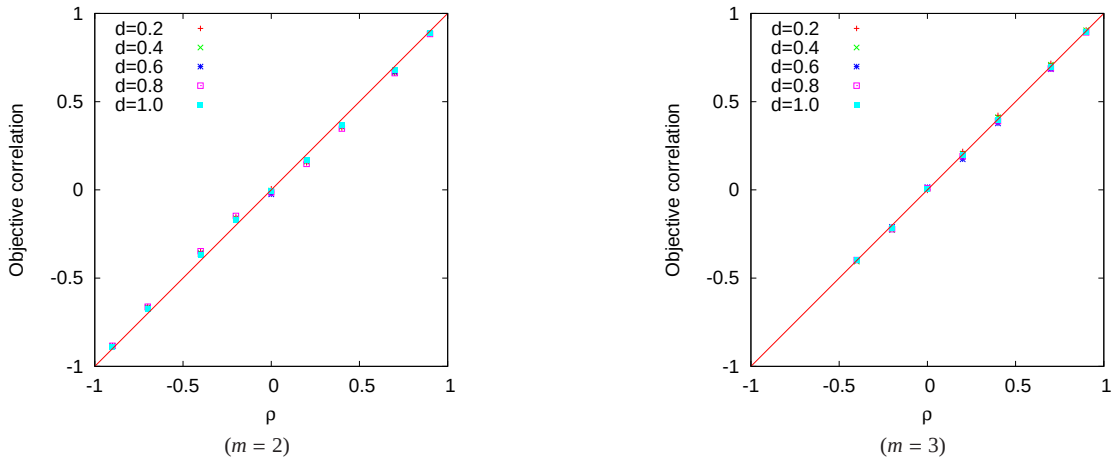


Figure 2: Average value of the Spearman correlation coefficient between the objective function values and the correlation coefficient ρ. The feasible set is enumerated exhaustively for $n = 18$ on a set of 30 independent random instances. The number of objectives is $m = 2$ (left) and $m = 3$ (right).

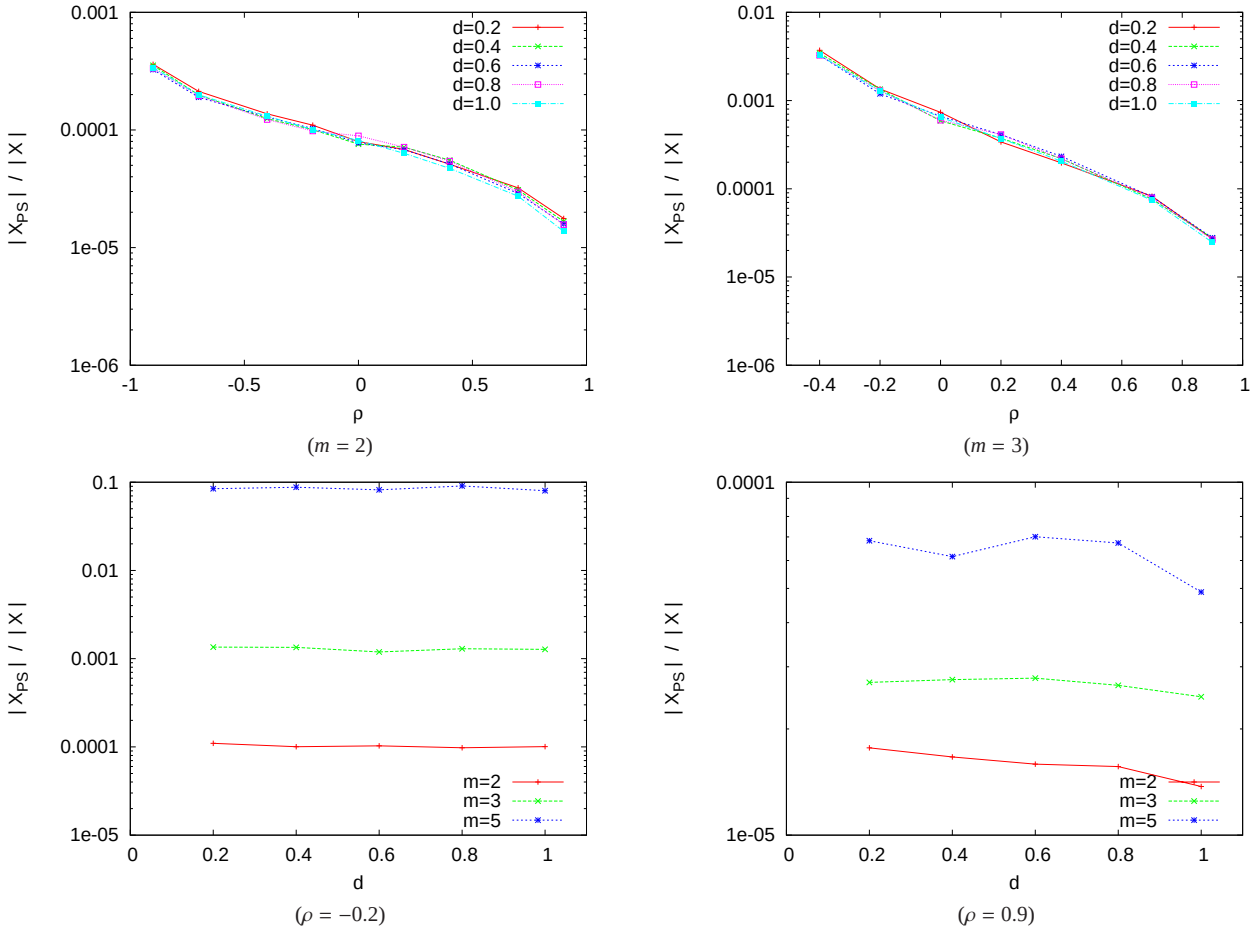


Figure 3: Average ratio of the minimal complete Pareto set cardinality ($|X_{PS}|$) to the solution space size ($|X| = 2^{18}$) according to the objective correlation ρ (top left $m = 2$, right $m = 3$), and according to the Q-matrix density d (bottom left $\rho = -0.2$, right $\rho = 0.9$). The problem size is $n = 18$. Notice the log-scale.

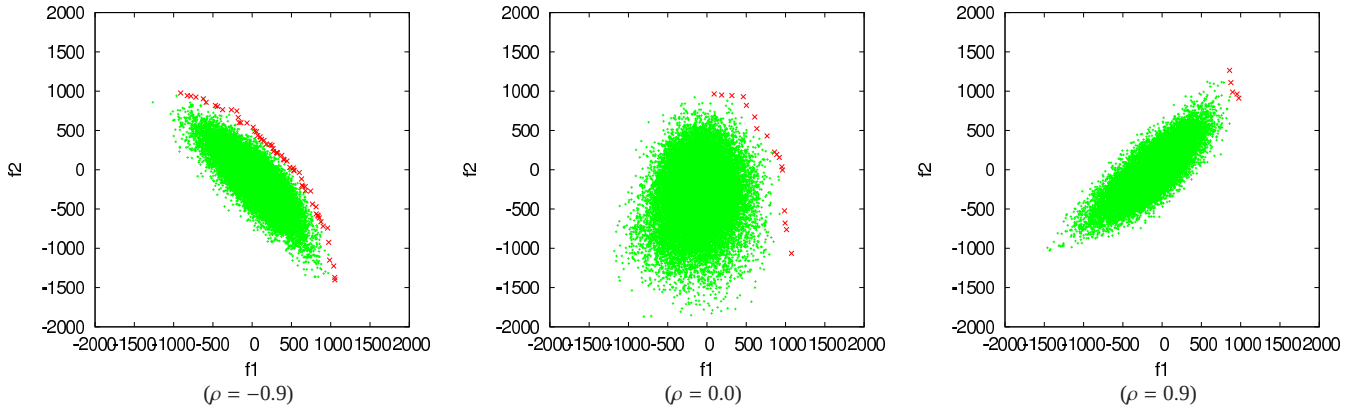


Figure 4: Representation of feasible solutions of a mUBQP problem instance in a two-objective space. The problem size is $n = 18$, the Q-matrix density is $d = 0.8$, the number of objective functions is $m = 2$, and the objective correlation is $\rho = -0.9$ (left), $\rho = 0.0$ (middle) and $\rho = 0.9$ (right). The objective vectors of (random) dominated solutions (10% of the solution space size) are represented by a + while (all) non-dominated objective vectors are represented by a $\times$.

space can be embedded in a multidimensional ball (middle). Last, when the objective correlation is positive, there exist few solutions in the Pareto front (right).

3. A Hybrid Metaheuristic for mUBQP

The hybrid metaheuristic proposed for the mUBQP problem is based on a memetic algorithm framework [36], which is known to be an effective approach for discrete optimization [37, 38]. Our approach uses one of the best performing local search algorithm for single-objective UBQP as one of its main components [12, 13].

3.1. General Principles

Memetic algorithms are hybrid metaheuristics combining an evolutionary algorithm and a local search algorithm. Multiobjective memetic algorithms [39] seek an approximation of the Pareto set (not only a subpart of it). A simple elitist multiobjective population-based evolutionary algorithm operates as the main metaheuristic, whereas an advanced single solution-based local search is used as an improvement operator in place of the mutation step. Keeping the exploration *vs.* exploitation trade-off in mind, the idea behind such an approach is that the evolutionary algorithm will offer more facilities for diversification, while the local search algorithm will provide more capabilities for intensification.

The search space is composed of all binary vectors of size n. The size of the search space is then equal to 2^n. The evaluation function is the canonical objective function given in Eq. (2). An unbounded *archive* of mutually non-dominated solutions found so far is maintained with respect to the Pareto dominance relation defined in Section 2.2. Throughout the search process, solutions are discarded as soon as they are detected to be equivalent to, or dominated by, at least one other solution from the archive. At each iteration, two parents are selected at random from the archive and recombined to produce a single offspring solution (Section 3.5). The offspring solution is further improved by means of a tabu search algorithm (Section 3.3). The evaluation function used by tabu search is based on a scalarizing technique of the initial objective function values (Section 3.2). The corresponding achievement scalarizing function is defined in such a way that the tabu search procedure focuses its search within the objective space area enclosed by the positions of parent solutions. Another crucial component of the HM algorithm appears at the initial phase (Section 3.4), where a computational effort is made in order to identify high-quality solutions for each individual objective function, independently of the remaining objective functions. The algorithm is iterated until a user-given stopping condition is satisfied. An outline of the hybrid metaheuristic (HM) is given in Algorithm 1. The main components of the HM algorithm are detailed below.

Algorithm 1 Pseudo-code of the hybrid metaheuristic (HM) for mUBQP

Input: matrix Q (dimension $m \times n \times n$)
Output: Pareto set approximation A

initialize the archive A /* see Section 3.4 */
repeat
  randomly select two individuals x_i, x_j from A
  $x \leftarrow$ recombine(x_i, x_j) /* see Section 3.5 */
  $x^\star \leftarrow$ tabu_search(x) /* see Section 3.3 */
  $A \leftarrow$ non-dominated solutions from ($A \cup \{x^\star\}$)
until a stopping condition is satisfied

3.2. Achievement Scalarizing Function

The tabu search procedure, that will be presented later in the paper, is known to be well-performing for solving single-objective UBQP instances of different structures and sizes [12, 13, 14, 15, 16]. Of course, given that it manipulates a single solution only, a scalarization of the multiple objective functions is required due to the multiobjective nature of the mUBQP. The goal is to *temporarily* transform the mUBQP problem into a single-objective one so that the tabu search algorithm can be used in a straightforward way. Many general-purpose scalariz-

ing functions have been proposed for multiobjective optimization [40], generally with the aim of incorporating preference information coming from a decision-maker. The matter is here different since we are interested in approximating the whole Pareto set. Hence, the parameters required by the scalarizing function are *dynamically* set according to the current state of the search process. This will be discussed in Section 3.5.

In multiobjective memetic algorithms, the most popular scalarizing function is the weighted sum aggregation [39, 41], where a weighting coefficient vector represents the relative importance of each objective function. However, this approach cannot identify a number of Pareto optimal solutions, whose corresponding non-dominated objective vectors are located in the *interior* of the convex hull of the feasible set in the objective space [24, 40]. Another example is the *achievement scalarizing function*, proposed by Wierzbicki [42]. This technique is based on a reference point. A reference point gives desirable or acceptable values for each objective function. These objective values are called *aspiration levels* and the resulting objective vector is called a *reference point* and can be defined either in the feasible or in the infeasible region of the objective space. One of the families of achievement scalarizing functions can be stated as follows. Let us recall that the maximization of the objective functions is assumed.

$$\begin{aligned} \sigma_{(z^r,\lambda,\epsilon)}(x) &= \max_{k\in\{1,\ldots,m\}} \left\{\lambda_k(z^r_k - f_k(x))\right\} \\ &\quad + \epsilon \sum_{k=1}^{m} \lambda_k(z^r_k - f_k(x)) \end{aligned} \tag{3}$$

where σ is a function from X to $\mathbb{R}$, $x \in X$ is a feasible solution, $z^r \in \mathbb{R}^m$ is a reference point, $\lambda \in \mathbb{R}^m$ is a weighting coefficient vector, and ϵ is an arbitrary small positive number ($0 < \epsilon \ll 1$). We keep the ϵ parameter constant throughout the search process. The following achievement scalarizing optimization problem can be formalized.

$$\begin{aligned} \min \quad & \sigma_{(z^r,\lambda,\epsilon)}(x) \\ \text{subject to} \quad & x \in X \end{aligned} \tag{4}$$

Interestingly, two properties are ensured [43]:

(*i*) if $x^\star = \arg\min_{x\in X} \sigma_{(z^r,\lambda,\epsilon)}(x)$, then $x^\star$ is a Pareto optimal solution;

(*ii*) if $x^\star$ is a Pareto optimal solution, then there exists a function $\sigma_{(z^r,\lambda,\epsilon)}$ such that $x^\star$ is a (global) optimum of Problem (4).

This makes the achievement scalarizing function attractive. Indeed, as noticed earlier, only a subset of Pareto optimal solutions, known as supported solutions [24], can be found with a weighted sum aggregation function, since the second property (*ii*) is not satisfied. Those solutions are known as supported Pareto optimal solutions, and their corresponding non-dominated objective vectors are located on the boundary of the convex hull of the Pareto front. On the contrary, the achievement scalarizing function potentially enables the identification of both supported and non-supported Pareto optimal solutions [40]. Successful integrations of the achievement scalarizing function into evolutionary multiobjective optimization algorithms can be found elsewhere [44, 45, 46]. However, in existing approaches, the parameters of the achievement scalarizing function are usually kept static or randomly chosen throughout the search process, whereas they are adapted to appropriate values according to the current state of the search process in the HM proposed in the paper, as will be detailed in Section 3.5.

3.3. Tabu Search

The following tabu search algorithm, used as a subroutine of the HM, is reported to be one of the best-performing approaches for the single-objective UBQP problem [13]. In order to extend it to the multiobjective case, we use the achievement scalarizing function, so that the initial objective vector values are transformed into a single scalar value. Notice, however, that the evaluation function considered in the paper has a different structure than the classical evaluation function of single-objective UBQP. We describe the main principles of the tabu search below.

The neighborhood structure is based on the 1-flip operator. Two feasible solutions are neighbors if they differ exactly on one variable. In other words, a given neighbor can be reached by changing the value of a binary variable to its complement from the current solution. The size of the 1-flip neighborhood structure is linear with the problem size n. As in the single-objective UBQP, each mUBQP objective function can be evaluated incrementally. We follow the fast incremental evaluation procedure proposed by Glover and Hao [47] to calculate the move gain of a given neighboring solution. For a given objective function, the whole set of neighbors can be evaluated in linear time. As a consequence, the objective values of all neighboring solutions are evaluated in $O(m \cdot n)$ in the multiobjective case. Once the objective values of a given neighboring solution have been (incrementally) evaluated, we compute its scalar fitness value with respect to Eq. (3).

As a short-term memory, we maintain the tabu list as follows. Revisiting solutions is avoided within a certain number of iterations, called the tabu tenure. The *tabu tenure* of a given variable x_i is denoted by $tenure(i)$. Hence, variable x_i will *not* be flipped again for a number of $tenure(i)$ iterations. Following Lü *et al.* [20], we set the tabu tenure of a given variable x_i after it has been flipped as follows.

$$tenure(i) = tt + rand(10) \tag{5}$$

where tt is a user-given parameter and $rand(10)$ gives a random integer value between 1 and 10. From the set of neighboring solutions produced by all non-tabu moves, we select the one with the best (smallest) fitness value according to Eq. (3). Indeed, let us recall at this point that the aim of the tabu search algorithm is to find a good approximate solution for Problem (4), for a given definition of z^r and λ. However, all neighboring solutions are always evaluated, and a tabu move can still be selected if it produces a better solution than the current global best. This is called an *aspiration criterion* in tabu search. The stopping condition of the tabu search algorithm is met when no improvement

has been performed within a given number of moves α. The parameter α is called the *improvement cutoff*. For more details on the tabu search algorithm for the single-objective UBQP, the reader is referred to Glover *et al*. [13, 20].

3.4. Initial Phase

The goal of the initial phase is to identify good-quality solutions with respect to each individual objective function of the mUBQP. This set of solutions initializes the search process in order to ensure that the HM provides a good covering of the Pareto front. To this end, we define the following achievement scalarizing function parameter setting. We set the reference point $z^r = \{z_1^{max}, \ldots, z_m^{max}\}$ such that z_k^{max} is higher than any possible f_k-value. This (unfeasible) objective vector is an utopia point [24, 40]. Now, let us consider a particular objective function $k \in \{1, \ldots, m\}$. We set $\lambda_k = 1$, and $\lambda_l = 0$ for all $l \in \{1, \ldots, m\} \setminus \{k\}$.

The tabu search algorithm, seeded with a random solution, is then considered within the corresponding achievement scalarizing function as an evaluation function. Those initial solutions have a high impact on the performance of the HM, particularly in terms of diversification. As a consequence, we perform γ independent restarts of the tabu search *per* objective function in order to increase the chance of getting high-performing solutions with respect to each individual objective function. This process is iterated for every objective function of the mUBQP problem instance under consideration.

3.5. Variation Operator

At each iteration of the HM algorithm, a single offspring solution is created by a recombination operator. First, we select two mutually non-dominated parent solutions at random from the current archive $x^{(i)}, x^{(j)} \in A$ such that $x^{(i)} \neq x^{(j)}$. Then, an offspring solution is created with uniform crossover. Common variables between both parents are thus assigned to the offspring solution, while the remaining ones are assigned at random. The offspring solution is further improved by means of the tabu search procedure presented in Section 3.3. We aim at obtaining a new solution in an unexplored region of the Pareto front by defining the parameters of the achievement scalarizing function properly. The procedure attempts to find a non-dominated point that "fills the gap" between the objective vectors associated with $x^{(i)}$ and $x^{(j)}$. The region of the objective space where the tabu search algorithm operates is then delimited by the position of parent solutions, given by the following definition of the achievement scalarizing function.

$$z_k^r = \max\{f_k(x^{(i)}), f_k(x^{(j)})\} \qquad k \in \{1, \ldots, m\} \tag{6}$$

$$\lambda_k = \frac{1}{\left|f_k(x^{(i)}) - f_k(x^{(j)})\right|} \qquad k \in \{1, \ldots, m\} \tag{7}$$

This procedure allows the HM to improve, at each iteration, a particular part of the Pareto front approximation, dynamically chosen with respect to the pair of parent solutions under selection. The overall variation procedure is illustrated in Figure 5.

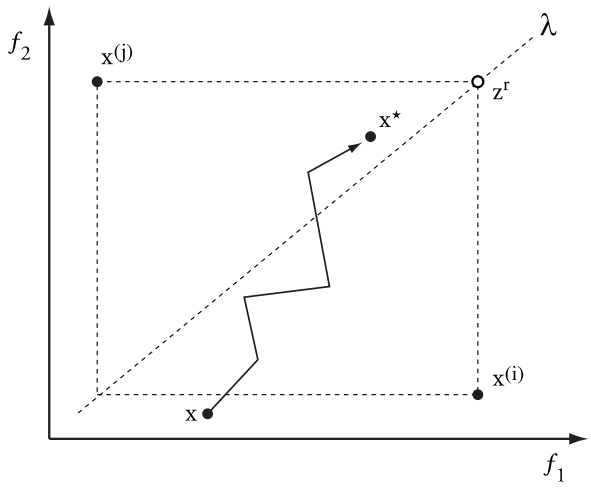


Figure 5: Graphical representation of the improvement phase in a two-objective space, where $x^{(i)}$ and $x^{(j)}$ are the parent solutions, x is the offspring solution and $x^\star$ is the solution improved by means of the tabu search procedure through the achievement scalarizing evaluation function defined by the reference point z^r and the weighting coefficient vector λ.

4. Experimental Analysis

This section presents an experimental analysis of the proposed approach on a broad range of mUBQP problem instances.

4.1. Experimental Design

We conduct an experimental study on the influence of the problem size (n), the number of objectives (m), and the objective correlation (ρ) of the mUBQP problem on the performance of the HM algorithm proposed in the paper. In particular, we investigate the following parameter setting: $n \in \{1000, 2000, 3000, 4000, 5000\}$, $m \in \{2, 3\}$, and $\rho \in \{-0.5, -0.2, 0.0, +0.2, +0.5\}$. The density of the matrices is set to $d = 0.8$. One instance, generated at random, is considered *per* parameter combination. This leads to a total of 50 problem instances.

We compare the performance of our algorithm against a steady-state evolutionary algorithm that follows the same structure as the HM, but where the tabu search is replaced by a random mutation. This allows us to appreciate the impact of the tabu search and scalarizing procedure. The same initialization phase as in the HM is applied. Then, at each iteration, an offspring solution is created by uniform crossover and an independent bit-flip operator is applied, *i.e.*, each variable is randomly flipped with a probability $1/n$. We refer to this algorithm as SS-EA, for steady-state evolutionary algorithm. We also compare the results of the algorithms to a baseline algorithm, the well-known NSGA-II [48]. NSGA-II maintains a population of constant size, initialized at random, and produces the same number of offspring solutions at every iteration. Selection for reproduction and replacement is based on dominance-depth ranking first, and on crowding distance at second-level. At each iteration, non-dominated solutions from the current population are first

Table 1: Parameter setting for the experimental analysis.

Description	Parameter	Value(s)
Instances		
Problem size	n	$\{1000, 2000, 3000, 4000, 5000\}$
Matrix density	d	0.8
Number of objectives	m	$\{2, 3\}$
Objective correlation	ρ	$\{-0.5, -0.2, 0.0, +0.2, +0.5\}$
Algorithms		
Crossover rate		1.0
Mutation rate (SS-EA, NSGA-II)		$1.0/n$
Population size (NSGA-II)		100
Tabu tenure	tt	$n/150$
Tabu improvement cutoff	α	$5n$
Number of restarts (initialization)	γ	5
reference point	z^r	adaptively set; see Section 3.5
weighting coefficient vector	λ	adaptively set; see Section 3.5
ϵ-parameter (achievement function)	ϵ	10^{-8}
Stopping condition (CPU time)		$(n \cdot m \cdot 10^{-3})$ minutes

assigned a rank of 1 and are discarded from consideration, non-dominated solutions from the remaining solutions of the population are then assigned a rank of 2 and are discarded from consideration, and so on. This process is iterated until the set of solutions with no rank is empty. The crowding distance estimates the density around a particular objective vector. The crowding value is computed among solutions with the same rank. A solution is said to be better than another solution if the former has a better rank, or in the case of equality, if it is less crowded. A binary tournament is used for selection, and an elitist strategy is used for replacement. The same crossover and mutation operators as for SS-EA are considered. In other words, the main differences between SS-EA and NSGA-II are: (*i*) SS-EA uses an unbounded population whereas NSGA-II maintains a fixed-size population, (*ii*) selection for reproduction is performed at random within SS-EA whereas it is based on dominance-depth and crowding distance within NSGA-II, and (*iii*) the archive is initialized as detailed in Section 3.4 for SS-EA whereas the NSGA-II initial population is generated at random. However, an external unbounded archive has been added to the canonical NSGA-II in order to prevent the loss of non-dominated solutions. We did not experience any memory issues by maintaining the whole set of non-dominated solutions found during the search process with any of the competing algorithms.

All the algorithms stop after $(n \cdot m \cdot 10^{-3})$ minutes of CPU time, *i.e.*, from 2 minutes *per* run for smaller instances up to 15 minutes for large-size instances. Since neighboring solutions are evaluated incrementally within HM during the tabu search phases, a maximum number of evaluations cannot be used as a stopping condition. Following [20], the tabu tenure constant is set to $tt = n/150$, and the improvement cutoff to $\alpha = 5n$. During the initialization phase, the number of random restarts *per* objective function is set to $\gamma = 5$. Last, the ϵ-parameter of the achievement scalarizing function is set to $\epsilon = 10^{-8}$. The population size of NSGA-II is set to 100 solutions. A summary of all the parameters is given in Table 1.

HM, SS-EA and NSGA-II have been implemented within the ParadisEO software framework [49, 50]. All the algorithms have been executed under comparable conditions and share the same base components for a fair comparison. The experiments have been conducted on an Intel Core 2 quad-core processor (2.40 GHZ, 4GB RAM) running under Ubuntu 10.04. All codes were compiled with g++ 4.4.3 using the `-O3` compilation option.

4.2. Performance Assessment

A set of 30 runs *per* instance has been performed for each algorithm. In order to evaluate the quality of the approximations found for each instance, we follow the performance assessment protocol proposed by Knowles *et al.* [26]. Such a way of comparing multiple stochastic multiobjective optimizers is a common practice in the specialized literature. Let us consider a given mUBQP problem instance. Let Z^{all} be the set of objective vectors from all the Pareto set approximations we obtained during all our experiments. Note that Z^{all} may contain both dominated and non-dominated objective vectors, since a given approximation may contain points dominating the ones of another approximation, and *vice versa*. We define $z^{min} = (z_1^{min}, \ldots, z_m^{min})$ and $z^{max} = (z_1^{max}, \ldots, z_m^{max})$, where z_k^{min} (respectively z_k^{max}) denotes the smallest (respectively largest) value of the k^{th} objective for all the points contained in Z^{all}, $\forall k \in \{1, \ldots, m\}$. In order to give a roughly equal range to the objective functions, values are normalized between 1 and 2 with respect to z^{min} and z^{max}. Then, we compute a reference set $Z^\star$ containing the non-dominated points of Z^{all}. In order to compare the quality of Pareto front approximations, we firstly use the Pareto dominance relation extended to sets, illustrated in Figure 6. The Pareto dominance relation over sets can be defined as follows. A given Pareto front approximation A^1 is dominated by another

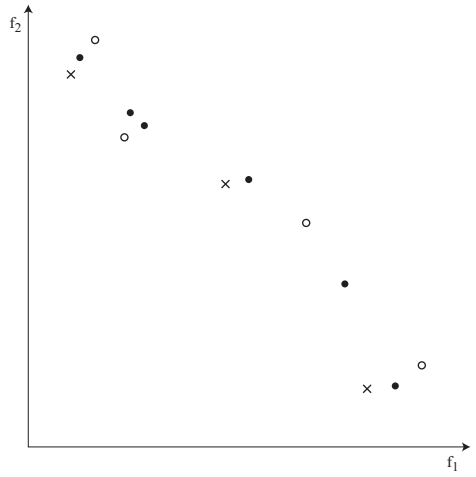


Figure 6: Illustration of the Pareto dominance relation over Pareto front approximations: (*i*) the approximation (•) dominates the approximation (×), (*ii*) the approximations (•) and (○) are incomparable, and (*iii*) the approximations (×) and (○) are incomparable.

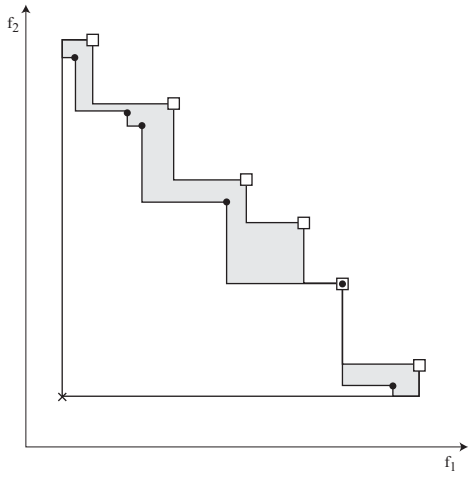


Figure 7: Illustration of the hypervolume difference quality indicator (I_H^-). The reference set is represented by boxes (□), the Pareto front approximation by bullets (•) and the reference point z^I by a cross (×). The shaded area represents the hypervolume difference $I_H^-(\bullet, \square)$.

approximation A^2, if for all objective vectors $z^1 \in A^1$, there exists an objective vector $z^2 \in A^2$ such that z^1 is dominated by z^2. However, in the case of incomparability with respect to the Pareto dominance relation, we use the hypervolume difference indicator (I_H^-) [25], illustrated in Figure 7, as a second criterion. The I_H^--indicator value of a given approximation A gives the portion of the objective space that is dominated by $Z^\star$ and not by A, $z^I = (0.9, \ldots, 0.9)$ being the reference point. Note that I_H^--values are to be minimized. This indicator allows us to obtain a total order between approximation sets. The experimental results report average I_H^--values and a Wilcoxon signed rank statistical test with a p-value of 0.05. This procedure has been achieved using the performance assessment tools provided in PISA [26].

4.3. Computational Results and Discussion

Computational results are presented in Table 2. Let us start with an example. The left part of the first line corresponds to the following mUBQP problem instance: $n = 1000$, $\rho = -0.5$ and $m = 2$. The average I_H^--value obtained by HM, NSGA-II and SS-EA over the 30 executions is 0.042, 0.325 and 0.085, respectively. According to the I_H^- indicator, the ranking deduced from the statistical test is as follows: (*i*) HM, (*ii*) SS-EA, and (*iii*) NSGA-II. The Pareto set approximations obtained by NSGA-II are reported to be statistically outperformed by the ones from HM in terms of Pareto dominance. Similarly, SS-EA is outperformed by HM in terms of hypervolume indicator-values.

First, compared against NSGA-II, the HM algorithm clearly performs better. Indeed, the Pareto set approximation found by NSGA-II is always dominated by the one obtained by HM. That is, every solution found by NSGA-II is dominated by at least one solution found by HM for all the runs over all the instances. The only cases where this does not happen is for $m = 3$ and $\rho = -0.5$ as well as the following instance: $n = 1000$, $m = 3$ and $\rho = -0.2$. Still, HM outperforms NSGA-II in terms of hypervolume for the corresponding instances.

With respect to SS-EA, the hypervolume indicator is always required to differentiate approximation sets. For all the instances with $n \leqslant 3000$, HM gives better results, except for $m = 3$ and $\rho = -0.5$. However, for large-size instances ($n \geqslant 4000$), HM seems to have more trouble in finding a better approximation set than SS-EA in some cases, particularly when the objective functions are in conflict. Indeed, HM performs better than SS-EA on nine out of the twenty largest instances while the reverse holds for eight cases. For such problem instances, the number of non-dominated solutions can become very large, such that there is probably a lack of diversity for the HM algorithm compared to its non-hybrid counterpart.

Overall, we can conclude that the HM algorithm gives significantly better results on most mUBQP problem instances. It clearly outperforms the conventional NSGA-II algorithm on the whole set of instances, whereas it is outperformed by SS-EA on only ten out of fifty mUBQP instances.

5. Conclusions

The contributions of the paper are three-fold. First, the unconstrained binary quadratic programming (UBQP) problem has been extended to the multiobjective case (mUBQP) which involves an arbitrary number of UBQP objective functions to be maximized simultaneously over the same feasible solution set of binary strings of size n. In the single-objective case, the UBQP problem is one of the most challenging problems from combinatorial optimization, and is known to enable the formulation of a large number of practical applications in many areas. The multiobjective UBQP problem introduced in this paper will allow more practical applications to be formulated and solved.

Second, multiobjective UBQP problem instances and an instance generator have been made available at the following

Table 2: Comparison of the proposed HM against NSGA-II and SS-EA. The symbol '$\succ$' (resp. '$\prec$') means that HM significantly outperforms (resp. is significantly outperformed by) the algorithm under consideration with respect to the set-based Pareto dominance relation. The symbol '$\succeq$' (resp. '$\preceq$') means that HM significantly outperforms (resp. is significantly outperformed by) the algorithm under consideration with respect to the difference hypervolume indicator (I_H^-). The symbol '$\equiv$' means that no algorithm outperforms the other in terms of either Pareto dominance or I_H^- values. The average I_H^--value is reported in brackets for HM, NSGA-II and SS-EA, respectively (lower is better).

		$m = 2$					$m = 3$				
n	ρ	HM		NSGA-II		SS-EA	HM		NSGA-II		SS-EA
1000	−0.5	(0.042)	$\succ$	(0.325)	$\succeq$	(0.085)	(0.104)	$\succeq$	(0.273)	$\succeq$	(0.113)
	−0.2	(0.052)	$\succ$	(0.336)	$\succeq$	(0.094)	(0.120)	$\succeq$	(0.410)	$\succeq$	(0.339)
	0.0	(0.037)	$\succ$	(0.336)	$\succeq$	(0.109)	(0.127)	$\succ$	(0.449)	$\succeq$	(0.405)
	+0.2	(0.037)	$\succ$	(0.348)	$\succeq$	(0.120)	(0.096)	$\succ$	(0.471)	$\succeq$	(0.420)
	+0.5	(0.032)	$\succ$	(0.385)	$\succeq$	(0.132)	(0.092)	$\succ$	(0.508)	$\succeq$	(0.409)
2000	−0.5	(0.099)	$\succ$	(0.416)	$\succeq$	(0.176)	(0.140)	$\succeq$	(0.248)	$\preceq$	(0.080)
	−0.2	(0.112)	$\succ$	(0.473)	$\succeq$	(0.188)	(0.221)	$\succ$	(0.434)	$\succeq$	(0.335)
	0.0	(0.070)	$\succ$	(0.520)	$\succeq$	(0.177)	(0.208)	$\succ$	(0.518)	$\succeq$	(0.427)
	+0.2	(0.097)	$\succ$	(0.587)	$\succeq$	(0.215)	(0.193)	$\succ$	(0.577)	$\succeq$	(0.477)
	+0.5	(0.054)	$\succ$	(0.757)	$\succeq$	(0.229)	(0.171)	$\succ$	(0.738)	$\succeq$	(0.556)
3000	−0.5	(0.136)	$\succ$	(0.471)	$\succeq$	(0.153)	(0.159)	$\succeq$	(0.239)	$\preceq$	(0.071)
	−0.2	(0.125)	$\succ$	(0.566)	$\succeq$	(0.192)	(0.262)	$\succ$	(0.417)	$\succeq$	(0.288)
	0.0	(0.111)	$\succ$	(0.640)	$\succeq$	(0.223)	(0.321)	$\succ$	(0.529)	$\succeq$	(0.394)
	+0.2	(0.177)	$\succ$	(0.728)	$\succeq$	(0.303)	(0.282)	$\succ$	(0.639)	$\succeq$	(0.470)
	+0.5	(0.131)	$\succ$	(0.931)	$\succeq$	(0.341)	(0.254)	$\succ$	(0.845)	$\succeq$	(0.572)
4000	−0.5	(0.216)	$\succ$	(0.497)	$\preceq$	(0.178)	(0.188)	$\succeq$	(0.235)	$\preceq$	(0.051)
	−0.2	(0.195)	$\succ$	(0.607)	$\succeq$	(0.238)	(0.311)	$\succ$	(0.405)	$\preceq$	(0.267)
	0.0	(0.157)	$\succ$	(0.687)	$\succeq$	(0.233)	(0.325)	$\succ$	(0.441)	$\preceq$	(0.280)
	+0.2	(0.147)	$\succ$	(0.813)	$\succeq$	(0.271)	(0.349)	$\succ$	(0.647)	$\succeq$	(0.450)
	+0.5	(0.089)	$\succ$	(1.001)	$\succeq$	(0.263)	(0.299)	$\succ$	(0.860)	$\succeq$	(0.568)
5000	−0.5	(0.267)	$\succ$	(0.500)	$\preceq$	(0.153)	(0.201)	$\succeq$	(0.231)	$\preceq$	(0.056)
	−0.2	(0.250)	$\succ$	(0.624)	$\equiv$	(0.204)	(0.283)	$\succ$	(0.319)	$\preceq$	(0.156)
	0.0	(0.219)	$\succ$	(0.725)	$\equiv$	(0.235)	(0.305)	$\succ$	(0.403)	$\preceq$	(0.238)
	+0.2	(0.192)	$\succ$	(0.802)	$\succeq$	(0.253)	(0.359)	$\succ$	(0.576)	$\equiv$	(0.393)
	+0.5	(0.125)	$\succ$	(1.023)	$\succeq$	(0.236)	(0.359)	$\succ$	(0.859)	$\succeq$	(0.518)

URL: `http://mocobench.sf.net`. These problem instances are characterized by a problem size, a matrix density, a number of objective functions, and a correlation coefficient between the objective values. In particular, the objective correlation can be tuned precisely, allowing one to study the impact of this feature on the size of the Pareto front, and on the performance of solution approaches. These instances are useful for performance assessment and comparison of new algorithms for the general mUBQP problem.

Third, we have presented a hybrid evolutionary-tabu search algorithm for the multiobjective UBQP. The proposed approach integrates a state-of-the-art tabu search algorithm for the single-objective UBQP, with Pareto-based evolutionary optimization principles. Based on the achievement scalarizing function, the proposed algorithm is able to generate both supported and unsupported solutions, with the aim of finding a well-converged and well-diversified Pareto set approximation. We have showed that this hybrid metaheuristic obtains significantly better results than two conventional evolutionary multiobjective optimization techniques for large-size multiobjective UBQP problem instances of different structure and size.

A better understanding of the main problem characteristics would allow us to improve the design of heuristic search algorithms by incorporating a deeper problem knowledge. To this end, we plan to study the correlation between the main problem features and the algorithm performance through fitness landscape analysis in multiobjective combinatorial optimization [35, 51]. Last, we hope that the challenge proposed by multiobjective UBQP will gain the attention of other researchers. In particular, a stronger link is required between multiobjective UBQP formulations and existing combinatorial optimization problems like the multiobjective variants of assignment, covering, partitioning, packing and quadratic knapsack problems. This would enable the identification of a Pareto front approximation for many problems from multiobjective combinatorial optimization under a unified modeling, either as a standalone methodology, or to provide a fast computation of a lower bound set for improving the performance of exact approaches.

Acknowledgements. The authors would like to acknowledge the reviewers for their valuable feedback that highly contributed to improve the quality of the paper. We are also grateful to Prof. Fred Glover and Prof. Gary Kochenberger for fruitful discussions related to the subject of this work.

References

[1] R. D. McBride, J. S. Yormark, An implicit enumeration algorithm for quadratic integer programming, Management Science 26 (3) (1980) 282–296.

[2] F. Harary, On the notion of balance of a signed graph, Michigan Mathematical Journal 2 (2) (1953) 143–146.

[3] J. Krarup, P. M. Pruzan, Computer-aided layout design, in: Mathematical Programming in Use, Vol. 9 of Mathematical Programming Studies, Springer, 1978, Ch. 6, pp. 75–94.

[4] P. Chardaire, A. Sutter, A decomposition method for quadratic zero-one programming, Management Science 41 (4) (1994) 704–712.

[5] G. Kochenberger, F. Glover, B. Alidaee, C. Rego, A unified modeling and solution framework for combinatorial optimization problems, OR Spectrum 26 (2) (2004) 237–250.

[6] M. Lewis, G. Kochenberger, B. Alidaee, A new modeling and solution approach for the set-partitioning problem, Computers & Operations Research 35 (3) (2008) 807–813.

[7] M. R. Garey, D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman & Co Ltd, 1979.

[8] E. Boros, P. L. Hammer, R. Sun, G. Tavares, A max-flow approach to improved lower bounds for quadratic 0-1 minimization, Discrete Optimization 5 (2) (2008) 501–529.

[9] C. Helmberg, F. Rendl, Solving quadratic (0,1)-problem by semidefinite programs and cutting planes, Mathematical Programming 82 (3) (1998) 291–315.

[10] P. M. Pardalos, G. P. Rodgers, Computational aspects of a branch and bound algorithm for quadratic zero-one programming, Computing 45 (2) (1990) 131–144.

[11] K. Katayama, H. Narihisa, Performance of simulated annealing-based heuristic for the unconstrained binary quadratic programming problem, European Journal of Operational Research 134 (1) (2001) 103–119.

[12] F. Glover, G. A. Kochenberger, B. Alidaee, Adaptive memory tabu search for binary quadratic programs, Management Science 44 (3) (1998) 336–345.

[13] F. Glover, Z. Lü, J.-K. Hao, Diversification-driven tabu search for unconstrained binary quadratic problems, 4OR: A Quarterly Journal of Operations Research 8 (3) (2010) 239–253.

[14] G. Palubeckis, Multistart tabu search strategies for the unconstrained binary quadratic optimization problem, Annals of Operations Research 131 (1) (2004) 259–282.

[15] Y. Wang, Z. Lü, F. Glover, J. Hao, Backbone guided tabu search for solving the UBQP problem, Journal of Heuristics 19 (4) (2013) 679–695.

[16] Y. Wang, Z. Lü, F. Glover, J. Hao, Probabilistic GRASP-tabu search algorithms for the UBQP problem, Computers & Operations Research 40 (12) (2013) 3100–3107.

[17] Y. Wang, Z. Lü, F. Glover, J. Hao, Path relinking for unconstrained binary quadratic programming, European Journal of Operational Research 223 (3) (2012) 595–604.

[18] I. Borgulya, An evolutionary algorithm for the unconstrained binary quadratic problems, in: Computational Intelligence, Theory and Applications, Vol. 33 of Advances in Soft Computing, Springer, 2005, Ch. 1, pp. 3–16.

[19] A. Lodi, K. Allemand, T. M. Liebling, An evolutionary heuristic for quadratic 0-1 programming, European Journal of Operational Research 119 (3) (1999) 662–670.

[20] Z. Lü, F. Glover, J.-K. Hao, A hybrid metaheuristic approach to solving the UBQP problem, European Journal of Operational Research 207 (3) (2010) 1254–1262.

[21] P. Merz, K. Katayama, Memetic algorithms for the unconstrained binary quadratic programming problem, Biosystems 78 (1-3) (2004) 99–118.

[22] Y. Wang, Z. Lü, F. Glover, J. Hao, Solving the minimum sum coloring problem via binary quadratic programming, arXiv:1304.5876v1 [cs.DS].

[23] P. Serafini, Some considerations about computational complexity for multiobjective combinatorial problems, in: Recent advances and historical development of vector optimization, Vol. 294 of Lecture Notes in Economics and Mathematical Systems, Springer, 1987, pp. 222–232.

[24] M. Ehrgott, Multicriteria optimization, 2nd Edition, Springer, 2005.

[25] E. Zitzler, L. Thiele, M. Laumanns, C. M. Foneseca, V. Grunert da Fonseca, Performance assessment of multiobjective optimizers: An analysis and review, IEEE Transactions on Evolutionary Computation 7 (2) (2003) 117–132.

[26] J. Knowles, L. Thiele, E. Zitzler, A tutorial on the performance assessment of stochastic multiobjective optimizers, TIK Report 214, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Zurich, Switzerland (2006).

[27] C. A. Coello Coello, G. B. Lamont, D. A. Van Veldhuizen, Evolutionary Algorithms for Solving Multi-Objective Problems, 2nd Edition, Springer, New York, USA, 2007.

[28] M. Ehrgott, X. Gandibleux, Multiobjective combinatorial optimization — theory, methodology, and applications, in: Multiple Criteria Optimization: State of the Art Annotated Bibliographic Surveys, Vol. 52 of International Series in Operations Research & Management Science, Springer, 2003, pp. 369–444.

[29] C. Bazgan, H. Hugot, D. Vanderpooten, Solving efficiently the 0–1 multi-objective knapsack problem, Computers & Operations Research 36 (1) (2009) 260–279.

[30] R. Kumar, P. Singh, Assessing solution quality of biobjective 0-1 knapsack problem using evolutionary and heuristic algorithms, Applied Soft Computing 10 (3) (2010) 711–718.

[31] E. Angel, E. Bampis, L. Gourvès, Approximation algorithms for the bi-criteria weighted max-cut problem, Discrete Applied Mathematics 154 (12) (2006) 1685–1692.

[32] J. Knowles, D. Corne, Instance generators and test suites for the multiobjective quadratic assignment problem, in: 2nd International Conference on Evolutionary Multi-Criterion Optimization (EMO 2003), Vol. 2632 of Lecture Notes in Computer Science, Springer, Faro, Portugal, 2003, pp. 295–310.

[33] L. Paquete, T. Stützle, A study of stochastic local search algorithms for the biobjective QAP with correlated flow matrices, European Journal of Operational Research 169 (3) (2006) 943–959.

[34] J. E. Beasley, OR-library: Distributing test problems by electronic mail, Journal of the Operational Research Society 41 (11) (1990) 1069–1072.

[35] S. Verel, A. Liefooghe, L. Jourdan, C. Dhaenens, On the structure of multiobjective combinatorial search space: MNK-landscapes with correlated objectives, European Journal of Operational Research 227 (2) (2013) 331–342.

[36] F. Neri, C. Cotta, P. Moscato (Eds.), Handbook of Memetic Algorithms, Vol. 379 of Studies in Computational Intelligence, Springer, 2011.

[37] C. Blum, J. Puchinger, G. R. Raidl, A. Roli, Hybrid metaheuristics in combinatorial optimization: A survey, Applied Soft Computing 11 (6) (2011) 4135–4151.

[38] J.-K. Hao, Memetic algorithms for discrete optimization, in: Handbook of Memetic Algorithms, Vol. 379 of Studies in Computational Intelligence, Springer, 2012, Ch. 6, pp. 73–94.

[39] J. Knowles, D. Corne, Memetic algorithms for multiobjective optimization: Issues, methods and prospects, in: Recent Advances in Memetic Algorithms, Vol. 166 of Studies in Fuzziness and Soft Computing, Springer, 2005, pp. 313–352.

[40] K. Miettinen, Nonlinear Multiobjective Optimization, Vol. 12 of International Series in Operations Research and Management Science, Kluwer Academic Publishers, Boston, MA, USA, 1999.

[41] P. Chitra, R. Rajaram, P. Venkatesh, Application and comparison of hybrid evolutionary multiobjective optimization algorithms for solving task scheduling problem on heterogeneous systems, Applied Soft Computing 11 (2) (2011) 2725–2734.

[42] A. Wierzbicki, The use of reference objectives in multiobjective optimization, in: Multiple Objective Decision Making, Theory and Application, Vol. 177 of Lecture Notes in Economics and Mathematical Systems, Springer, 1980, pp. 468–486.

[43] R. E. Steuer, Multiple Criteria Optimization: Theory, Computation and Application, John Wiley & Sons, Chichester, UK, 1986.

[44] M. Szczepański, A. Wierzbicki, Application of multiple criteria evolutionary algorithms to vector optimisation, decision support and reference point approaches, Journal of Telecommunications and Information Technology 3 (2003) 16–33.

[45] L. Thiele, K. Miettinen, P. J. Korhonen, J. Molina, A preference-based evolutionary algorithm for multi-objective optimization, Evolutionary Computation 17 (3) (2009) 411–436.

[46] J. R. Figueira, A. Liefooghe, E.-G. Talbi, A. P. Wierzbicki, A parallel multiple reference point approach for multi-objective optimization, European Journal of Operational Research 205 (2) (2010) 390–400.

[47] F. Glover, J.-K. Hao, Efficient evaluations for solving large 0-1 uncon-

strained quadratic optimisation problems, International Journal of Metaheuristics 1 (1) (2010) 3–10.

[48] K. Deb, S. Agrawal, A. Pratap, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation 6 (2) (2002) 182–197.

[49] J. Humeau, A. Liefooghe, E.-G. Talbi, S. Verel, ParadisEO-MO: From fitness landscape analysis to efficient local search algorithms, Journal of Heuristics (in press). doi:10.1007/s10732-013-9228-8.

[50] A. Liefooghe, L. Jourdan, E.-G. Talbi, A software framework based on a conceptual unified model for evolutionary multiobjective optimization: ParadisEO-MOEO, European Journal of Operational Research 209 (2) (2011) 104–112.

[51] S. Verel, A. Liefooghe, C. Dhaenens, Set-based multiobjective fitness landscapes: a preliminary study, in: 13th conference on Genetic and Evolutionary Computation Conference (GECCO 2011), ACM, Dublin, Ireland, 2011, pp. 769–776.