Available online at www.sciencedirect.com
ScienceDirect
European Journal of Operational Research 187 (2008) 1494–1503
EUROPEAN JOURNAL OF OPERATIONAL RESEARCH
www.elsevier.com/locate/ejor

Constrained 0–1 quadratic programming: Basic approaches and extensions

Alberto Caprara *

DEIS, University of Bologna, Viale Risorgimento 2, 40136 Bologna, Italy

Available online 15 November 2006

Abstract

We describe the simplest technique to tackle 0–1 Quadratic Programs with linear constraints among those that turn out to be successful in practice. This method is due to and familiar to the Quadratic Assignment experts, even if it took some time to realize that most approaches to the problem could be interpreted in these terms, whereas it does not appear to be widely known outside this community. Since the technique is completely general and is by far the most successful one in several other cases, such as Quadratic Knapsack, we illustrate it in its full generality, pointing out its relationship to Lagrangian and linear programming relaxations and discussing further extensions. We believe that this method should be in the background of every practitioner in Combinatorial Optimization.

Keywords: Combinatorial optimization; Quadratic 0–1 programming; Bounding procedures; Lagrangian relaxation; Reformulation

1. Introduction

In this survey, we illustrate a fairly simple technique to derive bounds for 0–1 Quadratic Programs with linear constraints, which on the one hand is very easy to understand and implement, and on the other turns out to be the best tool available for a wide set of specific problems, especially when the corresponding instances are not "hard". As is the case for most bounding procedures, the method can be used as the main block not only of exact branch-and-bound algorithms but also of heuristics that derive near optimal solutions (proving near optimality at the same time).

The main ideas of the method were developed for Quadratic Assignment by various authors starting from the 60s, but the first attempts to unify the apparently different techniques into a common framework were made only in the 90s. The same techniques were also rediscovered much later for Quadratic Knapsack and then applied to a number of other problems. If one looks at the method from the right perspective, the unified treatment for general 0–1 Quadratic Programs poses no difficulty. However, to the best of our knowledge such a treatment cannot be found in the literature, and this is the main motivation for the present survey.

The paper is organized as follows. In Section 2 we define 0–1 Quadratic Programming and its linearizations, illustrate a number of relevant special cases, and introduce the notion of monotonicity, which is a property shared by all cases for which

* Tel.: +39 051 20 93029; fax: +39 051 20 93073.
E-mail address: acaprara@deis.unibo.it

0377-2217/$ - see front matter
doi:10.1016/j.ejor.2006.09.028

the methodology of this paper has been successful. This methodology is illustrated in Section 3, while extensions that should produce significantly tighter bounds (with a significantly larger computational effort) are discussed in Section 4.

2. 0–1 Quadratic Programming

In this paper, we consider the 0–1 *Quadratic Program* (QP)

$$\max \sum_{i\in N}\sum_{j\in N} c_{ij}x_ix_j, \tag{1}$$

$$x \in F, \tag{2}$$

where $N := \{1,\dots,n\}$, c is an $n\times n$ *cost matrix* and $F \subseteq \{0,1\}^n$ denotes the *feasible region*. Sometimes we will denote QP by QP(c,F) to recall that an instance is defined by c and F. Note that, for $i,j \in N$, $i<j$, the overall cost associated with the product x_ix_j is $c_{ij}+c_{ji}$. Accordingly, even if c_{ij} and c_{ji} are changed keeping their sum unchanged, the problem remains the same. The explicit introduction of two separate terms $c_{ij}x_ix_j$ and $c_{ji}x_jx_i$ is aimed at illustrating the method discussed in this paper, as will be clear in Section 3.

Note that there is always a formulation in which F is described by linear constraints

$$F := \left\{ x \in \{0,1\}^n : \sum_{i\in N} a_{hi}x_i \leqslant b_h,\ h \in M \right\}, \tag{3}$$

where $M := \{1,\dots,m\}$, although this description is not strictly needed by the technique we describe in this paper. The simplest example of (3) arises for a single cardinality constraint, i.e.

$$F := \left\{ x \in \{0,1\}^n : \sum_{i\in N} x_i = p \right\}, \tag{4}$$

in which case the problem is already strongly NP-hard even for non-negative costs, as shown in the next section. Our suggestion for the reader of this paper is to keep this special case in mind as a reference example when trying to understand the description of the methodology, presented for the general case.

The standard *linearization* of QP amounts to replacing each product x_ix_i by x_i, $i \in N$ (which is of course legal), as well as each product x_ix_j, $i,j \in N$, $i<j$, by a new binary variable y_{ij}. For convenience, we will sometimes write y_{ii} for x_i. By introducing constraints that force y_{ij} to be 1 if and only if both x_i and x_j are 1, we get the equivalent statement of QP

$$\max \sum_{i\in N} c_{ii}x_i + \sum_{i\in N}\sum_{j\in N:j>i} (c_{ij}+c_{ji})y_{ij}, \tag{5}$$

$$x \in F, \tag{6}$$

$$y_{ij} \leqslant x_i, \quad i,j \in N,\ i<j, \tag{7}$$

$$y_{ij} \leqslant x_j, \quad i,j \in N,\ i<j, \tag{8}$$

$$y_{ij} \geqslant x_i + x_j - 1, \quad i,j \in N,\ i<j, \tag{9}$$

$$y_{ij} \in \{0,1\}, \quad i,j \in N,\ i<j. \tag{10}$$

Often in the literature one can find approaches to QP that solve the linear programming relaxation of (5)–(10) by a general-purpose solver (given the linear representation of F). Such a choice can only be motivated by the null implementation effort (just plug the model into the solver), since the method described below produces (often much) better upper bound values with an (often tremendously) smaller computational effort.

The basis of the method we present is a different (and stronger) linearization. First of all, we introduce *separate* variables y_{ij} and y_{ji} for $i,j \in N$, $i<j$, constraining them to be equal. Then, for $i \in N$, note that $x_i = 1$ implies $y_{ij} = x_j$ for $j \in N\backslash\{i\}$. Accordingly, if $x_i = 1$ we can impose $(y_{i1},\dots,y_{in}) \in F$ (recall that $y_{ii} \equiv x_i$). Define

$$F(x_i) := \begin{cases} \{0^n\} & \text{if } x_i = 0, \\ F & \text{if } x_i = 1, \end{cases}$$

where 0^n denotes the null n-dimensional vector, and note that if F is described by linear constraints as in (3), we have

$$F(x_i) = \left\{ (y_{i1},\dots,y_{in}) \in \{0,x_i\}^n : \sum_{j\in N\backslash\{i\}} a_{hj}y_{ij} \leqslant (b_h - a_{hi})x_i,\ h \in M \right\}. \tag{11}$$

The new linearization is

$$\max \sum_{i\in N} c_{ii}x_i + \sum_{i\in N}\sum_{j\in N\backslash\{i\}} c_{ij}y_{ij}, \tag{12}$$

$$x \in F, \tag{13}$$

$$(y_{i1},\dots,y_{in}) \in F(x_i), \quad i \in N, \tag{14}$$

$$y_{ij} \leqslant x_i, \quad i \in N,\ j \in N\setminus\{i\}, \tag{15}$$

$$y_{ij} = y_{ji}, \quad i,j \in N,\ i<j, \tag{16}$$

$$y_{ij} \geqslant x_i + x_j - 1, \quad i \in N,\ j \in N\setminus\{i\}. \tag{17}$$

The linear constraints in (11) are part of the constraints that can be obtained from (3) by applying a general procedure proposed by [2] and further studied in [3,23].

2.1. Special cases and applications

For the case $F = \{0,1\}^n$, called *Unconstrained* QP, the problem is already strongly NP-hard, being equivalent to *Max Cut* [11,17] (see Proposition 8 in the Appendix), whereas it is trivial if the costs are non-negative and polynomially solvable if all quadratic costs are non-negative, i.e. $c_{ij} + c_{ji} \geqslant 0$ for $i,j \in N$, $i < j$ (see Proposition 7 in the Appendix).

In the following, we list some relevant cases of QP in which not all vertices of the unit cube are feasible solutions. The methodology described in the next section is particularly suited for these cases, especially if the optimization of a *linear* objective function over F is easy. The cases are listed by increasing practical difficulty of this latter problem, which more or less corresponds to increasing difficulty of the quadratic problem itself.

As already mentioned, the simplest case is the one in which F is defined by (4). If all costs are non-negative, the resulting problem is known as p-Heaviest Subgraph or p-Dispersion. This problem, given a complete weighted undirected graph G with node set N and an integer $p \leqslant n$, calls for a subgraph of p nodes such that the sum of the costs of the edges internal to the subgraph is maximized, and can be formulated as the QP with F given by (4) and the cost of each edge (i,j) by $c_{ij} + c_{ji}$. This problem is discussed in [25].

Another case of QP in which F is defined by a single linear constraint is *Quadratic Knapsack*, arising for

$$F := \left\{ x \in \{0,1\}^n : \sum_{i \in N} w_i x_i \leqslant c \right\},$$

where w_i is the positive *weight* of each item $i \in N$ and c is the knapsack *capacity*. A wide set of references on Quadratic Knapsack can be found in [7].

A special case of QP that was proposed recently is *Contact Map*, which is a model for protein structure alignment in computational biology; we refer to [6,21] for a detailed background. In this case, the input is given by two *proteins*, each corresponding to a linear sequence of so-called *residues* in the three-dimensional space, for which we know the list of so-called *contacts*, that are residue pairs whose spatial distance is below a certain threshold. This information is represented by an undirected graph $G = (V,E)$ in which nodes correspond to residues and edges to contacts, and there is a *total order* of the nodes representing the sequence of residues in the protein, corresponding to the numbering of the nodes. Namely, letting $V = \{1,\ldots,m\}$, node $i \in V$ precedes node $j \in V$ in the total order if and only if $i < j$. Given the two graphs $G_1 = (V_1,E_1)$ and $G_2 = (V_2,E_2)$ associated with the given proteins, Contact Map calls for a *(partial) map* $f(\cdot)$ of the nodes in V_1 into those in V_2, which associates with each node $i \in V_1$ either nothing or a node $f(i) \in V_2$, guaranteeing that $f(i) < f(j)$ if $i < j$ (according to the total order mentioned above). We say that two edges $(i,j) \in V_1$ and $(h,k) \in V_2$, $i < j$, $h < k$, are *shared* by the map if $f(i) = h$ and $f(j) = k$. The objective is the maximization of the number of edges shared by the map. The resulting problem can be formulated as the QP where $N = V_1 \times V_2$ represents all possible mappings of one node in V_1 into a node in V_2, each quadratic cost is equal to 1 if it corresponds to two shared edges and 0 otherwise, and F defines the set of feasible maps and can be described by an exponentially large set of constraints, see [21].

Last but most important, we have the most studied case of constrained QP, namely *Quadratic Assignment*, in which we have a groundset M such that $N = M \times M$ and, noting that $i \in N$ corresponds to a pair (h,k), with $h,k \in M$,

$$F := \left\{ x \in \{0,1\}^n : \sum_{h \in M} x_{(h,k)} = 1,\ k \in M,\ \sum_{k \in M} x_{(h,k)} = 1,\ h \in M \right\}.$$

For a survey on the solution methods for Quadratic Assignment, see [9].

The methodology illustrated in Section 3 was first proposed for Quadratic Assignment (see [1,8,20] for detailed surveys), then applied to Quadratic Knapsack [7], p-Dispersion [25], and Contact Map [6]. As to *Unconstrained QP* (or Max Cut), the extensive computational results carried out on the problem suggest that the linear programming relaxation underlying our methodology is fairly weak (see also [18]), which motivates the use of more sophisticated methods.

2.2. Monotone QP

To the best of our knowledge, the technique described in this paper has been applied only to

cases in which constraints (17) turn out to be redundant. It is therefore natural to classify this class of problems under an appropriate category.

We will call a QP a *Monotone QP* (MQP) if, for every $\bar{x} \in F$, the optimal solution value of (12)–(17) with the addition of $x_i = \bar{x}_i$ for $i \in N$ does not change if (17) are removed.

Monotonicity is in general hard to test for a specific QP. For the sake of completeness, we report a detailed proof of this fact. Given an undirected graph $G = (V, E)$, a *vertex cover* is a subset $S \subseteq V$ such that for each edge $(i,j) \in E$, $S \cap \{i,j\} \neq 0$. The associated optimization problem, *Vertex Cover*, calls for a vertex cover of minimum cardinality.

Proposition 1. *Given a cost matrix c and a feasible region F, testing if $QP(c,F)$ is monotone is NP-complete.*

Proof. Given an undirected graph $G = (V, E)$ and a positive integer p, let $n := |V| + 1$ (letting $V = \{1, \ldots, n-1\}$), $c_{ii} := n$ for $i \in N (= V \cup \{n\})$, $c_{ij} + c_{ji} := 0$ for $i,j \in V$, $i < j$, and $c_{in} + c_{ni} := -1$ for $i \in V$. Moreover, let

$$F := \left\{ x \in \{0,1\}^n : \sum_{i \in V} x_i - (n-p)x_n \leqslant p, \quad x_i + x_j + x_n \geqslant 1, \ (i,j) \in E \right\}.$$

It is easy to see that the vectors in F are given by all vectors $x \in \{0,1\}^n$ with $x_n = 1$, and by all vectors $x \in \{0,1\}^n$ with $x_n = 0$ and $(x_1, \ldots, x_{n-1})$ incidence vector of a vertex cover of G of cardinality at most p, if any exists. Accordingly, if no vertex cover of G has cardinality at most p, the QP defined is monotone, since for each $\bar{x} \in F$ and $i \in V$ such that $\bar{x}_i = 1$, all vectors $(y_{i1}, \ldots, y_{in}) \in F(\bar{x}_i) \equiv F$ have $y_{in} = 1$, and therefore the cost is the same for all feasible solutions with $x = \bar{x}$, even if (17) are removed. On the other hand, we show below that this QP is not monotone if G contains a vertex cover of cardinality p. This yields the proof, since Vertex Cover is NP-hard (see e.g. [14]).

Consider $\bar{x}$ defined by $\bar{x}_i := 1$ for $i \in N$. Forcing $x_i = \bar{x}_i$ for $i \in N$ yields a QP solution of value $n(n-1) - (n-1)$. However, if G contains a vertex cover $S \subseteq V$ of cardinality p, assuming for simplicity $S = \{1, \ldots, p\}$, the following is a feasible solution of (12)–(17) after the removal of (17) and addition of $x_i = \bar{x}_i (= 1)$ for $i \in N$: $y_{ij} = 1$ for $i \in S$, $j \in S \backslash \{i\}$, $y_{in} = y_{ni} = 1$ for $i \in V \backslash S$, $y_{ij} = 0$ for the remaining i,j pairs. The value of this solution is $n(n-1) - (n-p-1)$, strictly larger than the previous one, showing that the QP is not monotone. □

However, there are many cases of QP for which monotonicity is guaranteed, that can be subdivided into two main classes.

Proposition 2. *Given a cost matrix c, $QP(c,F)$ is monotone for every feasible region F if and only if*

$$c_{ij} + c_{ji} \geqslant 0, \quad i,j \in N, \ i < j. \tag{18}$$

Proof. Consider $\bar{x} \in F$, assuming for convenience (and without loss of generality) $\bar{x}_1 = \cdots = \bar{x}_s = 1$ and $\bar{x}_{s+1} = \cdots = \bar{x}_n = 0$. Imposing $x = \bar{x}$, (15) imply $y_{ij} = y_{ji} = 0$ whenever $\max\{i,j\} > s$. Moreover, the feasible solution $y_{ij} = y_{ji} = 1$ for $i,j \in \{1, \ldots, s\}$, $i < j$ is optimal, regardless of the presence of (17), since $c_{ij} + c_{ji} \geqslant 0$ for $i,j \in \{1, \ldots, s\}$, $i < j$ and all other feasible solutions are obtained by decreasing some entries of this solution. This proves the if part.

The only if part is shown by an easy example. Suppose $c_{ij} + c_{ji} < 0$ for some $i \in N$, $j \in N \backslash \{i\}$. Considering $F := \{x \in \{0,1\}^n : x_k = 0, \ k \in N \backslash \{i,j\}\}$ and taking $\bar{x}_i := \bar{x}_j := 1$ (and $\bar{x}_k := 0$ for $k \in N \backslash \{i,j\}$) yields $\bar{x} \in F$ for which the corresponding best value of y_{ij} and y_{ji} is 0 in case (17) are not imposed. □

Proposition 3. *Given a feasible region F, $QP(c,F)$ is monotone for every cost matrix c if and only if, for each $\bar{x} \in F$, letting $\{i_1, \ldots, i_s\} = \{i : \bar{x}_i = 1\}$, there exists no $s \times s$ matrix Z such that:*

(i) $z_{kl} = z_{lk}$ *for* $k,l = 1, \ldots, s$, $k < l$;
(ii) $z_{kk} = 1$ *for* $k = 1, \ldots, s$;
(iii) $z_{kl} = 0$ *for some* $k,l \in \{1, \ldots, s\}$;
(iv) $x^k \in F$ *for* $k = 1, \ldots, s$, *where* $x^k_{i_l} := z_k$ *for* $l = 1, \ldots, s$ *and* $x^k_i := 0$ *for* $i \in N \backslash \{i_1, \ldots, i_s\}$.

Proof. The proof for the if part is analogous to Proposition 2. The difference is that, in this case, $y_{ij} = y_{ji} = 1$ for $i,j \in \{1, \ldots, s\}$, $i < j$ is the *unique* feasible solution after having imposed $x = \bar{x}$.

For the only if part, suppose there exists $\bar{x} \in F$, for convenience $\bar{x}_1 = \cdots = \bar{x}_s = 1$ and $\bar{x}_{s+1} = \cdots = \bar{x}_n = 0$, along with a matrix Z as in the statement of the proposition (noting $\{i_1, \ldots, i_s\} = \{1, \ldots, s\}$). Let $c_{ii} := n$ for $i \in N$ and $c_{ij} + c_{ji} := -1$ for $i \in N$, $j \in N \backslash \{i\}$. If $x_i = \bar{x}_i$ for $i \in N$ is imposed and (17) are not imposed, the solution $y_{ij} = z_{ij}$ for

$i,j = 1,\ldots,s$ (and $y_{ij} = 0$ for $\max\{i,j\} > s$) is feasible and better than $y_{ij} = 1$ for $i,j = 1,\ldots,s$. □

Note that the requirement of Proposition 3 is satisfied whenever there is no dominance relation among elements in F, i.e. for each $x^1 \in F$ there is no $x^2 \in F\backslash\{x^1\}$ such that $x^1_i \leqslant x^2_i$ for $i \in N$. A further special case that is particularly relevant is the one in which the cardinality of each element in F is the same, i.e.

$$\sum_{i\in N} x_i = k, \quad x \in F \tag{19}$$

for some constant k.

Unfortunately, testing any of the above conditions, including (19), for a set F described by a set of linear constraints is easily seen to be NP-complete. The situation would be even worse if F was given by a membership oracle, since in this case verifying (19) might require exponential time even if P = NP. On the other hand, for all cases of practical interest, one knows in advance whether (19) holds.

Note that (18) holds for p-Dispersion, Quadratic Knapsack and Contact Map, whereas (19) holds for Quadratic Assignment. Accordingly, all these cases are MQPs. On the other hand, Unconstrained QP (and Max Cut) are non-monotone.

3. An upper bounding procedure for QP

When the quadratic costs are non-negative, since QP is easy if $F = \{0,1\}^n$ (see Proposition 7), a possible approach to derive upper bounds for general F given in form (3) is to relax the linear constraints in a Lagrangian way, getting a Lagrangian relaxed problem with unchanged quadratic costs. This approach, discussed in [12], does not appear to be competitive with the methodology illustrated in this paper, that can be applied to general quadratic costs.

3.1. A simple upper bound

We will present an intuitive interpretation of the procedure, starting from the original formulation (1) and (2). Expressing the objective function (1) as

$$\max \sum_{i\in N} \left(\sum_{j\in N} c_{ij}x_j \right) x_i,$$

it is immediate to observe that, if $x_i = 1$, the term $\sum_{j\in N} c_{ij}x_j$ cannot exceed p_i, which is the optimal solution value of the following problem:

$$p_i := \max \sum_{j\in N} c_{ij}x_j, \tag{20}$$

$$x \in F, \tag{21}$$

$$x_i = 1, \tag{22}$$

calling for the maximization of a *linear* objective function over F, with x_i fixed to 1. For several cases of interest, this can be done either in polynomial time, as is the case for all problems mentioned in the previous section with the exception of Quadratic Knapsack, or anyway within short time in practice, as for Quadratic Knapsack (see below).

Once p_i has been computed for $i \in N$, a valid upper bound on the optimal value of (1) and (2), denoted by L_0, is given by the solution of

$$L_0 := \max \sum_{i\in N} p_i x_i, \tag{23}$$

$$x \in F, \tag{24}$$

which is again the maximization of a linear objective function over F. We let L_0 denote the corresponding upper bound. This methodology is related with the concept of *upper plane*, introduced by [13,18], which is a linear function $g(x)$ such that $g(x) \geqslant \sum_{i\in N}\sum_{j\in N} c_{ij}x_ix_j$ for all $x \in F$. In this case, the upper plane is $\sum_{i\in N} p_i x_i$.

For completeness, we report below the complexity of the maximization of a linear objective function over F for the cases discussed.

For Unconstrained QP, one simply has to set to 1 all variables with positive coefficient in the objective function.

For p-Dispersion, one simply needs to find the p smallest entries in a vector of n elements, that can be done in $O(n)$ time by median-finding techniques; see e.g. [25].

For Quadratic Knapsack, one has to solve a classical *Knapsack*, which can be done in time $O(nc)$ (where c is the right-hand-side of the unique linear constraint defining F, assuming all coefficients are integer) by dynamic programming. Of course a valid upper bound is obtained also by solving the continuous relaxation of Knapsack, which can be done in $O(n)$ time again by median-finding techniques; see e.g. [7,13].

For Contact Map, the problem to be solved is a so-called *Alignment* between two strings, which is the same as a *(Weighted) Bipartite Non-crossing Matching*, solvable in $O(n)$ time by dynamic programming, recalling that n is the product of the lengths of the two strings in this case; see e.g. [6].

For Quadratic Assignment, one has a (Linear) Assignment, solvable in $O(n^{3/2})$ time by classical methods (recall that n is the number of entries in the Assignment cost matrix); see e.g. [20]. The corresponding bound is known as the *Gilmore–Lawler bound* [15,22] (typically presented as a lower bound since Quadratic Assignment is mainly formulated in minimization form).

3.2. Improving the bound

Note that, although the subdivision of the cost gained if $x_i x_j = 1$ among c_{ij} and c_{ji} does not affect the optimal value of QP, it *does* affect L_0. In view of this, one would be interested in finding the subdivision that *minimizes* this upper bound. The key observation in this direction is the following:

Proposition 4. *L_0 is the optimal solution value of the relaxation of (12)–(17) obtained by removing constraints (16) and (17).*

Proof. If constraints (16) and (17) are removed, each variable y_{ij} appears only in constraints (14) and (15) associated with i. Accordingly, for $i \in N$, the optimal cost associated with $y_{i1}, \ldots, y_{in}$ is 0 if $x_i = 0$, whereas if $x_i = 1$ it is given by

$$\max \sum_{j \in N} c_{ij} y_{ij}, \tag{25}$$

$$(y_{i1}, \ldots, y_{in}) \in F, \tag{26}$$

$$y_{ii} = 1, \tag{27}$$

whose optimal solution value is clearly equal to p_i. □

Of course, an alternative to removing the above constraints is to relax them in a Lagrangian way. In the following, we will first concentrate on the Lagrangian relaxation of (16), while permanently removing (17).

3.2.1. The MQP case

We associate a Lagrangian multiplier λ_{ij} with each constraint (16). Noting that λ_{ij} is defined for $i < j$ and may be negative, we define for convenience $\lambda_{ji} := -\lambda_{ij}$ for $i,j \in N$, $i < j$. The resulting objective function reads

$$L(\lambda) := \max \sum_{i \in N} c_{ii} x_i + \sum_{i \in N} \sum_{j \in N \setminus \{i\}} (c_{ij} + \lambda_{ij}) y_{ij} \tag{28}$$

and the resulting Lagrangian relaxation calls for the maximization of (28) subject to (13)–(15). This relaxation can clearly be solved in the same way as in the case in which (16) are removed (i.e. $\lambda_{ij} = 0$ for $i,j \in N$, $i \neq j$). Note that the optimal x_is are given by the solution of (23) and (24), whereas we have

$$y_{ij} := \bar{y}_{ij} \cdot x_i, \quad i \in N, \ \ j \in N \setminus \{i\},$$

where $\bar{y}_{ij}$ define an optimal solution of (25)–(27) (with the Lagrangian costs), i.e. y_{ij} has value 1 in the solution of the relaxation if it has value 1 in the solution of the ith problem (25)–(27) *and* x_i has value 1 in the solution of (23) and (24).

As already mentioned, (28) is a *reformulation* of (12), since the overall profit achieved if $y_{ij} = y_{ji} = 1$ is $(c_{ij} + \lambda_{ij}) + (c_{ji} + \lambda_{ji}) = c_{ij} + c_{ji}$. In other words:

Proposition 5. *Finding optimal Lagrangian multipliers for (28) subject to (13)–(15) is equivalent to splitting the costs between symmetrical entries of c so as to minimize L_0.*

Let λ^* be an optimal set of multipliers and $L(\lambda^*)$ the corresponding upper bound. Taking into account the relationship between Lagrangian relaxation and linear programming (see e.g. [24]), we have the following result. Let C be the optimal value of the *continuous* relaxation of (12)–(16), obtained by replacing (13) and (14) by

$$x \in \mathrm{conv}(F), \tag{29}$$

$$(y_{i1}, \ldots, y_{in}) \in \mathrm{conv}(F(x_i)), \quad i \in N, \tag{30}$$

where $\mathrm{conv}(S)$ denote the convex hull of a set $S \subseteq \{0,1\}^n$.

Proposition 6

$$L(\lambda^*) = C.$$

In [27] the authors discuss the direct computation of C for Quadratic Assignment by a general-purpose linear programming solver. However, computation of near-optimal Lagrangian multipliers yields similar bounds within much smaller computing times.

As a final remark, note that, although for the monotone case constraints (17) are redundant as long as the variables are binary, the inclusion of these constraints may yield a better upper bound in the continuous relaxation. This is exploited for Quadratic Knapsack in [4].

3.2.2. Extension to non-monotone QP

Of course, for non-monotone QP we may proceed analogously, relaxing also (17) in a Lagrangian

way. Since these are inequality constraints, we will not get a reformulation in this case. Letting $\mu_{ij} \geqslant 0$ for $i \in N$, $j \in N\backslash\{i\}$ be the multipliers associated with (17) (and keeping the multipliers λ_{ij} associated with (16)), the Lagrangian objective function reads

$$\sum_{i\in N, j\in N\backslash\{i\}} \mu_{ij} + \max \sum_{i\in N} \left(c_{ii} - \sum_{j\in N\backslash\{i\}} \mu_{ij} \right) x_i + \sum_{i\in N} \sum_{j\in N\backslash\{i\}} (c_{ij} + \lambda_{ij} + \mu_{ij}) y_{ij}. \quad (31)$$

We expect that this straightforward extension to non-monotone QP could be effective in many cases, even if we are not aware of any reference dealing with it, and presentation of new computational results is out of the scope of this survey.

3.2.3. Finding good multipliers

Near-optimal Lagrangian multipliers can be found by iterative procedures such as the classical *subgradient* [19] or more sophisticated methods with better convergence properties. A few specific issues for iterative procedures in the case of Quadratic Assignment have been proposed in the literature and are surveyed in [20].

The general structure of an iterative procedure, described for simplicity for the MQP case, is the following: for given multipliers λ_{ij} the relaxed problem is solved, finding the optimal y_{ij} values. Then, for $i,j \in N$, $i<j$, if $y_{ij} = y_{ji}$ then λ_{ij} is unchanged, if $y_{ij} = 1$ and $y_{ji} = 0$ then λ_{ij} is decreased, and if $y_{ij} = 0$ and $y_{ji} = 1$ then λ_{ij} is increased, always decreasing the cost of the variable at 1 and increasing that of the variable at 0.

As a general rule, the choice of the iterative procedure should depend on the time required to solve the relaxation, that amounts in our case to $n+1$ optimizations of linear functions over F. If this time is very small, as is the case for p-Dispersion or Quadratic Knapsack (especially if the *continuous relaxation* of the resulting Knapsacks is solved) many iterations are not a problem and plain subgradient tends to work well, as discussed in [7,25].

For the non-monotone case, also constraints (17) can be relaxed, and one should increase μ_{ij} if $x_i = x_j = 1$ and $y_{ij} = 0$ in the relaxed solution, and decrease it if $x_i = x_j (= y_{ij}) = 0$. Although this has never been tried in practice, we would expect a subgradient procedure to optimize both λ_{ij} and μ_{ij} at the same time to work decently in practice (but this can be confirmed only by experimental evidence).

3.2.4. Heuristics

In order to produce good heuristic solutions within the iterative procedure, note first of all that at each iteration a solution of (23) and (24) is found, which is feasible for the original problem. In some cases this heuristic "for free" produces acceptable results. Moreover, one can define several specific heuristics based on the reformulation. A natural one is based on fixing to 1 the value of the variable x_i with highest p_i, solving again the relaxation, and iterating.

Another heuristic which is more time consuming but generally produces better solutions fixes one (or more) variables when the iterative procedure is terminated, and then re-applies the iterative procedure to re-optimize the Lagrangian multipliers for the reduced problem, according to the framework presented e.g. in [5,10].

3.2.5. Branch-and-bound

Of course, the procedure above can be embedded within a branch-and-bound algorithm that recursively generates two subproblems from the given one by fixing the value of a suitably-chosen variable to 1 and 0, respectively. Computational experience on some specific problems such as Quadratic Knapsack [7] seems to suggest that the best thing to do after fixing is *not to re-optimize* the Lagrangian multipliers after fixing. In other words, it is apparently better to find a near-optimal reformulation at the root node of the branch-and-bound tree and then stick to this reformulation at the other nodes, whose processing (per node) is then very fast. This probably reflects the limitations of the re-optimization techniques for Lagrangian relaxation with respect to the effectiveness of re-optimization techniques for linear programming.

4. 3- (and more-) index formulations

One of the possible interpretations of constraints (14) in the linearization that we considered is the one given in [2,3,23]: consider the linear constraints defining conv(F) and multiply them by x_i for each $i \in N$. The resulting linear constraints after the introduction of the y_{ij} variables give a tighter linear programming relaxation.

Clearly, one may iterate the process by multiplying the linear constraints defining $F(x_i)$ ($i \in N$) by x_j for $j \in N\backslash\{i\}$. This introduces terms of the form $y_{ik}x_j$, that can be linearized by introducing binary

variables z_{ijk}, $i \in N, j \in N\backslash\{i\}, k \in N\backslash\{i,j\}$. Proceeding as for the y_{ij}, we let

$$F(y_{ij}) := \begin{cases} \{0^n\} & \text{if } y_{ij} = 0, \\ F & \text{if } y_{ij} = 1. \end{cases}$$

If F is described by (3), we have

$$F(y_{ij}) = \left\{ (z_{ij1}, \ldots, z_{ijn}) \in \{0, y_{ij}\}^n : \sum_{k \in N\backslash\{i,j\}} a_{hk} z_{ijk} \leqslant (b_h - a_{hi} - a_{hj}) y_{ij}, \ h \in M \right\}, \tag{32}$$

where $z_{iji} \equiv z_{ijj} \equiv y_{ij}$. We obtain the following linearization:

$$\max \sum_{i \in N} c_{ii} x_i + \sum_{i \in N} \sum_{j \in N\backslash\{i\}} c_{ij} y_{ij}, \tag{33}$$

$$x \in F, \tag{34}$$

$$(y_{i1}, \ldots, y_{in}) \in F(x_i), \quad i \in N, \tag{35}$$

$$y_{ij} \leqslant x_i, \quad i \in N, \ j \in N \setminus \{i\}, \tag{36}$$

$$y_{ij} = y_{ji}, \quad i, j \in N, \ i < j, \tag{37}$$

$$y_{ij} \geqslant x_i + x_j - 1, \quad i \in N, \ j \in N \setminus \{i\}, \tag{38}$$

$$(z_{ij1}, \ldots, z_{ijn}) \in F(y_{ij}), \quad i \in N, \ j \in N \setminus \{i\}, \tag{39}$$

$$z_{ijk} \leqslant y_{ij}, \quad i \in N, \ j \in N \setminus \{i\}, \ k \in N \setminus \{i,j\}, \tag{40}$$

$$z_{ijk} = z_{ikj} = z_{jik} = z_{jki} = z_{kij} = z_{kji}, \quad i, j, k \in N, \ i < j < k, \tag{41}$$

$$z_{ijk} \geqslant y_{ij} + x_k - 1, \quad i \in N, \ j \in N \setminus \{i\}, \ k \in N \setminus \{i,j\}. \tag{42}$$

The associated continuous relaxation is tighter than that of (12)–(17). Proceeding as in the previous section, assuming the original QP is monotone so we can remove constraints (38) and (42), and relaxing both (37) and (41) in a Lagrangian way, we get the following reformulation of the objective function:

$$\max \sum_{i \in N} c_{ii} x_i + \sum_{i \in N} \sum_{j \in N\backslash\{i\}} (c_{ij} + \lambda_{ij}) y_{ij} + \sum_{i \in N} \sum_{j \in N\backslash\{i\}} \sum_{k \in N\backslash\{i,j\}} u_{ijk} z_{ijk}, \tag{43}$$

where u_{ijk} denotes the sum of the *five* Lagrangian multipliers associated with the five constraints (41) in which z_{ijk} appears, noting that for $i,j,k \in N$, $i < j < k$

$$u_{ijk} + u_{ikj} + u_{jik} + u_{jki} + u_{kij} + u_{kji} = 0.$$

The solution of the Lagrangian relaxation is conceptually identical to the previous case, even if now it requires $O(n^2)$ optimizations of a linear objective function over F. When this optimization can be done quickly, as for instance for p-Dispersion, the associated computational effort is limited. Actually, the bottleneck of the method is the fact that one should store the $O(n^3)$ values u_{ijk}, which may be impractical for large values of n.

Since the z_{ijk} variables have three indices, the above model goes under the name of 3-*index formulation*, letting the 2-*index formulation* be the original linearization. Such a formulation would arise naturally starting from a 0–1 *Cubic Program*. However, it may turn out to be useful even for QP although it has been exploited to a very limited extent so far. Of course, *k-index formulations* are possible for any $k \geqslant 4$, but they are at present impractical for all instances of interest.

In [26], the authors try to solve directly the linear programming relaxation of (33)–(42) for Quadratic Assignment (with (38) and (42) removed), succeeding for very small values of n. On the other hand, following the approach illustrated here, [16] obtains similar bounds for larger values of n, showing that the bounds computed with this 3-index reformulation tend to be much tighter than the one obtained by the classical 2-index reformulation.

5. Conclusions

We believe that the technique illustrated in this paper should always be taken into account when tackling a 0–1 problem in which the objective function is quadratic and optimization of a linear objective function over the feasible region is easy in practice, and hope that the present survey will stimulate research in this direction. Only if the method performs poorly (essentially because the underlying linear programming relaxation is weak), one should switch to more sophisticated tools. At present, these tools amount essentially to stronger linear programming and/or semidefinite programming relaxations, that potentially produce tighter (but also much slower to compute) bounds.

Appendix

In the appendix we prove the properties of Unconstrained QP mentioned in Section 2.1.

Proposition 7. *If* $c_{ij} + c_{ji} \geqslant 0$ *for* $i,j \in N$, $i < j$, *Unconstrained QP can be solved as a minimum s-t cut in a directed network.*

Proof. By possibly redefining $c_{ij} := c_{ji} := (c_{ij} + c_{ji})/2$, assume without loss of generality $c_{ij} \geqslant 0$ for $i \in N, j \in N\backslash\{i\}$. Moreover, observe that, if $c_{ii} \geqslant 0$, x_i can be fixed to 1, which amounts to removing x_i and redefining $c_{jj} := c_{jj} + c_{ij} + c_{ji}$ for $j \in N\backslash\{i\}$. Then, assuming $c_{ii} < 0$ for $i \in N$, construct the *directed network* with node set $\{s\} \cup \{t\} \cup N$, where s is the *source* and t the *sink*, and arc set

$$\{(s,i) : i \in N\} \cup \{(i,t) : i \in N\} \cup \{(i,j) : i \in N, j \in N \setminus \{i\}\},$$

with *capacities* q defined by

$$q_{si} := \sum_{j \in N\backslash\{i\}} c_{ij}, \quad i \in N, \quad q_{it} := -c_{ii}, \quad i \in N,$$
$$q_{ij} := c_{ij}, \quad i \in N, \quad j \in N \setminus \{i\}.$$

The *value* of an *s-t* cut associated with node set S, $s \in S$, is given by

$$v(S) := \sum_{i \notin S} \left(\sum_{j \in N\backslash\{i\}} c_{ij} \right) + \sum_{i \in S} \sum_{j \notin S} c_{ij} - \sum_{i \in S} c_{ii}.$$

Therefore, setting $x_i := 1$ if $i \in S$ and $x_i := 0$ if $i \notin S$, the value of objective function (1) is equal to

$$\sum_{i \in N} \sum_{j \in N\backslash\{i\}} c_{ij} - v(S),$$

i.e. QP can be solved by finding a minimum *s-t* cut in the network, by maximum flow techniques. □

Given a complete weighted undirected graph with node set V, Max Cut requires finding a subset $S \subseteq V$ such that $u(S) := \sum_{i \in S} \sum_{j \notin S} w_{ij}$ is maximum, where w_{ij} is the weight of each edge (i,j).

Proposition 8. *Unconstrained QP and Max Cut are equivalent.*

Proof. Letting x_i be a binary variable equal to 1 if and only if $i \in S$, we can formulate Max Cut as

$$\max \sum_{i \in V} \sum_{j \in V : j > i} w_{ij}[x_i(1 - x_j) + x_j(1 - x_i)]$$

subject to $x_i \in \{0,1\}$, $i \in V$, which is a special case of Unconstrained QP. Vice versa, given an instance of Unconstrained QP, define an undirected graph with node set $N \cup \{s\}$ and edge weights

$$w_{ij} := \frac{c_{ij} + c_{ji}}{2}, \quad i,j \in N, \quad i < j,$$
$$w_{is} := c_{ii} + \sum_{j \in N\backslash\{i\}} \frac{c_{ij} + c_{ji}}{2}, \quad i \in N.$$

The value of a cut $S \subseteq V$, assuming without loss of generality $s \in S$, is given by

$$u(S) = \sum_{i \notin S} \left(c_{ii} + \sum_{j \in N\backslash\{i\}} \frac{c_{ij} + c_{ji}}{2} \right) + \sum_{i \in S} \sum_{j \notin S} \frac{c_{ij} + c_{ji}}{2}$$

and setting $x_i := 1$ if $i \in S$ and $x_i := 0$ if $i \notin S$ we have that the value of objective function (1) is equal to

$$\sum_{i \in N} \sum_{j \in N} c_{ij} - u(S). \qquad \square$$

References

[1] W.P. Adams, T. Johnson, Improved linear programming-based lower bounds for the quadratic assignment problem, in: P. Pardalos, H. Wolkowicz (Eds.), Quadratic Assignment and Related Problems, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 16, 1994, pp. 43–76.

[2] W.P. Adams, H.D. Sherali, A tight linearization and an algorithm for zero-one quadratic programming problems, Management Science 32 (1986) 1274–1290.

[3] E. Balas, S. Ceria, G. Cornuéjols, A lift-and-project cutting plane algorithm for mixed 0–1 programs, Mathematical Programming 58 (1993) 295–324.

[4] A. Billionnet, F. Calmels, Linear programming for the 0–1 quadratic Knapsack problem, European Journal of Operational Research 92 (1996) 310–325.

[5] A. Caprara, M. Fischetti, P. Toth, A heuristic method for the set covering problem, Operations Research 47 (1999) 730–743.

[6] A. Caprara, G. Lancia, Structural alignment of large-size proteins via Lagrangian relaxation, in: Proceedings of the Sixth Annual International Conference on Computational Molecular Biology (RECOMB'02), ACM Press, 2002, pp. 100–109.

[7] A. Caprara, D. Pisinger, P. Toth, Exact solution of the quadratic knapsack problem, INFORMS Journal on Computing 11 (1999) 125–137.

[8] P. Carraresi, F. Malucelli, A reformulation scheme and new lower bounds for the QAP, in: P. Pardalos, H. Wolkowicz (Eds.), Quadratic Assignment and Related Problems, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 16, 1994, pp. 147–160.

[9] E. Çela, The Quadratic Assignment Problem: Theory and Algorithms, Kluwer Academic Publishers, 1998.

[10] S. Ceria, P. Nobili, A. Sassano, A Lagrangian-based heuristic for large-scale set covering problems, Mathematical Programming 81 (1998) 215–228.

[11] C. De Simone, The cut polytope and the boolean quadric polytope, Discrete Mathematics 79 (1989) 71–75.

[12] G. Gallo, M.D. Grigoriadis, R.E. Tarjan, A fast parametric maximum flow algorithm and applications, SIAM Journal on Computing 18 (1989) 30–55.
[13] G. Gallo, P.L. Hammer, B. Simeone, Quadratic knapsack problems, Mathematical Programming 12 (1980) 132–149.
[14] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman, 1979.
[15] P.C. Gilmore, Optimal and suboptimal algorithms for the quadratic assignment problem, SIAM Journal on Applied Mathematics 10 (1962) 305–313.
[16] W.P. Adams, M. Guignard, P.M. Hahn, W.L. Hightower, A level-2 reformulation-linearization technique bound for the quadratic assignment problem, European Journal of Operational Research, in press, doi:10.1016/j.ejor.2006.03.051.
[17] P. Hammer, Some network flow problems solved with pseudo-boolean programming, Operations Research 13 (1965) 388–399.
[18] P.L. Hammer, P. Hansen, B. Simeone, Roof, duality, complementation and persistency in quadratic 0–1 optimization, Mathematical Programming 28 (1984) 121–155.
[19] M. Held, R.M. Karp, The traveling salesman problem and minimum spanning trees: Part II, Mathematical Programming 1 (1971) 6–25.
[20] S.E. Karisch, E. Çela, J. Clausen, T. Espersen, A dual framework for lower bounds of the quadratic assignment problem based on linearization, Computing 63 (1999) 351–403.
[21] G. Lancia, R. Carr, B. Walenz, S. Istrail, 101 optimal PDB structure alignments: A branch-and-cut algorithm for the maximum contact map overlap problem, in: Proceedings of the Fifth Annual International Conference on Computational Molecular Biology (RECOMB'01), ACM Press, 2001, pp. 100–109.
[22] E.L. Lawler, The quadratic assignment problem, Management Science 9 (1963) 586–599.
[23] L. Lovász, A. Schrijver, Cones of matrices and set-functions and 0–1 optimization, SIAM Journal on Optimization 1 (1991) 166–190.
[24] G.L. Nemhauser, L.A. Wolsey, Integer and Combinatorial Optimization, Wiley, 1988.
[25] D. Pisinger, Upper bounds and exact algorithms for p-dispersion problems, Computers and Operations Research 33 (2006) 1380–1398.
[26] K.G. Ramakrishnan, M.G.C. Resende, B. Ramachandran, J.F. Pekny, Tight QAP bounds via linear programming, in: P.M. Pardalos, A. Migdalas, R.E. Burkard (Eds.), Combinatorial and Global Optimization, World Scientific Publishing, 2002, pp. 297–303.
[27] M.G.C. Resende, K.G. Ramakrishnan, Z. Drezner, Computing lower bounds for the quadratic assignment problem with an interior point algorithm for linear programming, Operations Research 43 (1995) 781–791.