ELSEVIER

European Journal of Operational Research 112 (1999) 664–672

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

Theory and Methodology

A new upper bound for the 0-1 quadratic knapsack problem

Alain Billionnet [*], Alain Faye, Éric Soutif

CEDRIC, Institut d'Informatique d'Entreprise, 18 allée Jean Rostand, 91025 Évry Cedex, France

Received 27 January 1997; accepted 3 November 1997

Abstract

The 0-1 quadratic knapsack problem (QKP) consists in maximizing a positive quadratic pseudo-Boolean function subject to a linear capacity constraint. We present in this paper a new method, based on Lagrangian decomposition, for computing an upper bound of QKP. We report computational experiments which demonstrate the sharpness of the bound (relative error very often less than 1%) for large size instances (up to 500 variables). © 1999 Elsevier Science B.V. All rights reserved.

Keywords: Quadratic programming; 0-1 Quadratic knapsack problem; Lagrangian decomposition; Bounds; Computational experiments

1. Introduction

A quadratic pseudo-Boolean function $f : \{0,1\}^n \rightarrow \mathbb{R}$ is defined as $f(x) = \sum_{i=1}^{n} c_i x_i + \sum_{1 \leqslant i<j \leqslant n} c_{ij} x_i x_j$ where c_i $(i = 1, \dots, n)$ and c_{ij} $(1 \leqslant i < j \leqslant n)$ are real coefficients. The 0-1 quadratic knapsack problem (QKP) consists in maximizing a positive quadratic pseudo-Boolean function subject to a linear capacity constraint. This problem was introduced by Gallo et al. (1980) and can be expressed as follows:

$$\text{(QKP)} \quad \max \quad f(x) = \sum_{i=1}^{n} c_i x_i + \sum_{1 \leqslant i<j \leqslant n} c_{ij} x_i x_j$$

$$\text{s.t.} \quad \sum_{i=1}^{n} a_i x_i \leqslant b,$$

$$x_i \in \{0,1\} \quad (i = 1, \dots, n),$$

where the coefficients c_i, a_i $(i = 1,\dots, n)$ and c_{ij} $(1 \leqslant i < j \leqslant n)$ are non-negative integers and b is an integer such as $0 < b < \sum_{i=1}^{n} a_i$.

[*] Corresponding author.

QKP may represent many practical situations (Laughhunn, 1970; Witzzgall, 1975; Gallo et al., 1980; Billionnet and Calmels, 1996). Several classical graph problems can be formulated as QKP. Consider, for example, the graph bipartitioning problem: given a graph $G=(V, E)$ of n vertices 1, 2,…, n, where V is the set of vertices and E the set of edges, and given a positive integer $l_{i,j}$ for each edge $[i,j] \in E$, find a partition of V into two disjoint sets V_1 and V_2 such that $|V_1| = k, (k \in \{2,\ldots,n-2\})$ and such that if $E' \subset E$ is the set of edges that have their two endpoints in two different sets, then $\sum_{[i,j]\in E'} l_{i,j}$ is minimal. We may formulate the bipartitioning problem as the following constrained quadratic 0-1 program in which $x_i = 1$ if and only if vertex i belongs to V_1:

$$\begin{aligned}
(\text{Pb1}) \quad & \min \quad \sum_{i=1}^{n} q_i x_i + \sum_{l \leqslant i<j \leqslant n} q_{ij} x_i x_j \\
& \text{s.t.} \quad \sum_{i=1}^{n} x_i = k, \\
& \qquad\;\; x \in \{0,1\}^n,
\end{aligned}$$

where $q_i = \sum_{j|[i,j]\in E} l_{i,j}$ and q_{ij} is equal to zero if $[i,j] \notin E$ and to $-2l_{i,j}$ if $[i,j] \in E$.

Noting that $x_i = k - \sum_{j\in\{1,\ldots,n\}\setminus\{i\}} x_j$ and since $\arg\min_{x\in D_f} f(x) = \arg\max_{x\in D_f} -f(x)$ ($D_f \subseteq$ definition domain of f) , the optimal solutions of this problem are exactly the ones of the following problem (Pb2) which sature the capacity constraint

$$\begin{aligned}
(\text{Pb2}) \quad & \max \quad cst + \sum_{i=1}^{n} c_i x_i + \sum_{l \leqslant i<j \leqslant n} c_{ij} x_i x_j \\
& \text{s.t.} \quad \sum_{i=1}^{n} x_i \leqslant k, \\
& \qquad\;\; x \in \{0,1\}^n,
\end{aligned}$$

where $cst = k\left(\sum_{i=1}^{n} q_i\right), c_i = \sum_{j\in\{1,\ldots,n\}/\{i\}} q_j(\geqslant 0), c_{ij} = -q_{ij}(\geqslant 0)$.

Note that it is always possible to construct from any optimal solution of (Pb2) an optimal solution saturing the constraint. We recognize (Pb2) as a QKP.

See another example of graph problem which can be formulated as a particular QKP: consider a graph $G=(V, E)$ where V is the set of vertices and E the set of edges. Find a subgraph of G with k vertices which maximizes the number of edges. This problem is called k-clustering and is NP-complete in general (Corneil and Perl, 1984). QKP is also interesting because it can be viewed as the simplest constrained integer non-linear programming problem.

A few authors have proposed algorithms for QKP. The algorithm proposed by Gallo et al., 1980 is of branch and bound type. The upper bound is obtained by replacing the objective function f by a linear function g which dominates f in all feasible points and by solving a linear knapsack problem. This upper bound has been improved by Körner, 1986. Chaillou et al., 1986 have also proposed a branch and bound algorithm for QKP. The computation of an upper bound is based on a Lagrangian relaxation. This computation has been improved by Gallo et al., 1989. Valid inequalities and facets for a linear formulation of QKP are described by Johnson et al., 1993. De Farias et al., 1993 describe three Lagrangian decompositions and present numerical results for problems up to 25 variables. Michelon and Veuilleux, 1996 also use Lagrangian decomposition and a heuristic method for solving the dual problem. They propose a branch and bound algorithm based on the obtained bound. Note that several authors try to fix a maximum of variables before the branch and bound by using the upper bound. Billionnet and Calmels, 1996 compute an upper bound by a cut generation algorithm associated with a linear formulation of QKP, try to fix some variables and use the obtained results to find the optimal solution. Helmberg et al., 1996 propose a semi-definite programming approach to QKP and discuss possible strengthenings of linear relaxations by polyhedral cutting plane approaches. Hammer and Rader, 1997 have proposed approximate and exact algorithms for QKP. They use in every node of the branch and bound tree a three-step procedure. The aim of this procedure is to fix the values of some of the variables by the use of order relations, Lagrangian techniques and constraint pairing. In two recent research reports, Rader (1997a, b) introduces a new class of valid inequalities and facets of the QKP-polytope and also characterizes several classes of facets of this polytope. His study does not mention computational results.

We present in this paper a new method for computing an upper bound of QKP which is based on the following points:

A feasible solution is determined by the algorithm of Billionnet and Calmels, 1996. This computation is composed of two stages. The first stage uses the greedy heuristic presented by Chaillou et al., 1986; the second stage consists of trying to improve the obtained feasible solution by a fill-up and exchange procedure as proposed by Gallo et al., 1980. The computational experiments show that this algorithm always gives a feasible solution very close to the optimum. This solution is quickly obtained (less than 1 s of CPU time is required on a HP 9000 workstation for large size instances).

An upper bound is computed by a new Lagrangian decomposition method inspired by the decomposition method proposed by Chardaire and Sutter, 1995 for minimizing an unconstrained quadratic pseudo-Boolean function. Computational experiments are promising. The proposed method allows us to find approximate solutions with a relative error less than 1%. For instances involving up to 500 variables we find such approximate solutions within 60 min of CPU running time on a HP 9000 workstation.

2. Computation of an upper bound

The method that we propose for QKP is inspired by the decomposition method proposed by Chardaire and Sutter, 1995 for minimizing an unconstrained quadratic pseudo-Boolean function. This decomposition has also been adapted and completed by Faye, 1994 for a task allocation problem in distributed computing systems.

Consider the following definitions:

- $X = \{x_1, x_2, \ldots, x_n\}$ is the set of the variables of the problem,
- $\{X_1, \ldots, X_p\}$ is a partition of X $(1 \leqslant p \leqslant n)$, X_k is called cluster k,
- $Y_k = X - X_k$,
- I_k (respectively J_k) is the index set of the variables of X_k (respectively Y_k),
- $cl(i)$ is the index of the cluster involving the variable x_i,
- x_{I_k} is the vector of the variables x_i, $i \in I_k$ (called vector of the "complicating" variables)

and, given a partition (I_k, J_k) of $\{1,\ldots,n\}$, let us consider the functions $f_k(x_{I_k}, x_{J_k}), k = 1, \ldots, p$, defined as follows:

$$f_k(x_{I_k}, x_{J_k}) = \sum_{i \in I_k} c_i x_i + \sum_{i \in I_k} \sum_{\substack{i' \in I_k \\ i' \neq i}} \frac{1}{2} c_{ii'} x_i x_{i'} + \sum_{i \in I_k} \sum_{j \in J_k} \frac{1}{2} c_{ij} x_i x_j,$$

where $c_{ij} = c_{ji}, 1 \leqslant i < j \leqslant n$.

Noting that

$$\forall x \in \{0,1\}^n, \quad \sum_{k=1}^{p} f_k(x_{I_k}, x_{J_k}) = \sum_{i=1}^{n} c_i x_i + \sum_{1 \leqslant i < j \leqslant n} c_{ij} x_i x_j = f(x),$$

where f is the objective function and x the vector $(x_1, x_2, \ldots, x_n)$, then QKP can be written as

$$\max\left\{\sum_{k=1}^{p} f_k(x_{I_k}, x_{J_k}):\ \sum_{i=1}^{n} a_i x_i \leqslant b, x \in \{0,1\}^n\right\}.$$

Let us examine the following problem

$$\max\left\{f_k(x_{I_k}, x_{J_k}):\ \sum_{i=1}^{n} a_i x_i \leqslant b, x \in \{0,1\}^n\right\}.$$

We may note that for each function f_k ($k = 1,\ldots,p$), if the components of the vector x_{I_k} are fixed, that is to say if the variables of the cluster k have a fixed value, then the function f_k has just linear terms. Thus we obtain a linear knapsack problem whose continuous relaxation can easily be computed.

This writing of the QKP suggests making use of the following Lagrangian decomposition: we create the copy variables y_j^k such that

$$y_j^k = x_j (j \in J_k, k = 1,\ldots,p). \tag{1}$$

We notice that the problem initially including n variables now includes pn ones: with each cluster (that includes its own variables x) is associated a set of copy variables that "take the place" of the variables which do not belong to the cluster.

Permuting the components of the vector a, we introduce these copy variables in the capacity constraint $ax \leqslant b$ and obtain the new constraints

$$a_k \begin{pmatrix} x_{I_k} \\ y_{J_k} \end{pmatrix} \leqslant b \quad (k = 1,\ldots,p).$$

We also introduce redundant copy constraints concerning the quadratic terms. This yields the following problem, equivalent to the initial problem:

$$(\mathrm{QKP}')\quad \max\quad \bar{f}(x,y) = \sum_{k=1}^{p}\left(\sum_{i\in I_k} c_i x_i + \sum_{i\in I_k}\sum_{\substack{i'\in I_k\\ i'\neq i}} \tfrac{1}{2} c_{ii'} x_i x_{i'} + \sum_{i\in I_k}\sum_{j\in J_k} \tfrac{1}{2} c_{ij} x_i y_j^k\right)$$

$$\text{s.t.}\quad x_j = y_j^k \quad (k = 1,\ldots,p; j \in J_k), \tag{c1}$$

$$x_i y_j^{cl(i)} = x_j y_i^{cl(j)} \quad (i = 1,\ldots,n-1; j = i+1,\ldots,n; cl(j) \neq cl(i)), \tag{c2}$$

$$x_i \in \{0,1\}^n \quad (i \in I_k) \quad (k = 1,\ldots,p), \tag{c3}$$

$$y_j^k \in \{0,1\}^n \quad (j \in J_k) \quad (k = 1,\ldots,p), \tag{c4}$$

$$a_k \begin{pmatrix} x_{I_k} \\ y_{J_k} \end{pmatrix} \leqslant b \quad (k = 1,\ldots,p). \tag{c5}$$

In order to determine an upper bound of QKP we now study the Lagrangian dual problem of QKP′. We first consider the relaxation of the constraints (c1) and (c2) involving, respectively, the Lagrangian multipliers $\lambda = (\lambda_j^k)_{\substack{1\leqslant k\leqslant p\\ j\in J_k}}$ and $\mu = (\mu_{ij})_{\substack{1\leqslant i<j\leqslant n\\ cl(i)\neq cl(j)}}$ such that $\lambda_j^k \in \mathbb{R}$ and $\mu_{ij} \in \mathbb{R}$. Let $L(x,y,\lambda,\mu)$ be the classical Lagrangian function and $w(\lambda,\mu)$ the dual function, defined by $w(\lambda,\mu) = \max_{\substack{x,y \text{ s.t}\\ (c3),(c4),(c5)}} L(x,y,\lambda,\mu)$. We have the following property.

Property 1. *Consider the Lagrangian dual problem of QKP′ obtained by dualizing constraints (c1) and (c2). The dual function $w(\lambda,\mu)$ can be written as*

$$w(\lambda,\mu) = \sum_{k=1}^{p} \max_{\substack{x_{I_k}, y_{J_k}^k\\ \text{s.t.}(c3),(c4),(c5)}} L_k(x_{I_k}, y_{J_k}^k, \lambda, \mu),$$

where

$$L_k\left(x_{I_k}, y^k_{J_k}, \lambda, \mu\right) = \sum_{i\in I_k}\left(c_i + \sum_{h\neq k}\lambda^h_i\right)x_i - \sum_{j\in J_k}\lambda^k_j y^k_j + \sum_{i\in I_k}\sum_{\substack{i'\in I_k\\ i'\neq i}}\frac{1}{2}c_{ii'}x_i x_{i'}$$
$$+ \sum_{i\in I_k}\sum_{\substack{j\in J_k\\ j>i}}\left(\frac{1}{2}c_{ij} + \mu_{ij}\right)x_i y^k_j + \sum_{i\in I_k}\sum_{\substack{j\in J_k\\ j<i}}\left(\frac{1}{2}c_{ij} - \mu_{ji}\right)x_i y^k_j.$$

Proof. Since L is the Lagrangian function associated with the transformed problem, we have

$$L(x,y,\lambda,\mu) = \overline{f}(x,y) + \sum_{k=1}^{p}\sum_{j\in J_k}\lambda^k_j(x_j - y^k_j) + \sum_{i=1}^{n-1}\sum_{\substack{j=i+1\\ cl(i)\neq cl(j)}}^{n}\mu_{ij}\left(x_i y_j^{cl(i)} - x_j y_i^{cl(j)}\right)$$

$$= \sum_{k=1}^{p}\left(\begin{array}{l}\sum\limits_{i\in I_k}\left(c_i + \sum\limits_{h\neq k}\lambda^h_i\right)x_i - \sum\limits_{j\in J_k}\lambda^k_j y^k_j + \sum\limits_{i\in I_k}\sum\limits_{\substack{i'\in I_k\\ i'\neq i}}\frac{1}{2}c_{ii'}x_i x_{i'} + \\ \sum\limits_{i\in I_k}\sum\limits_{\substack{j\in J_k\\ j>i}}\left(\frac{1}{2}c_{ij} + \mu_{ij}\right)x_i y^k_j + \sum\limits_{i\in I_k}\sum\limits_{\substack{j\in J_k\\ j<i}}\left(\frac{1}{2}c_{ij} - \mu_{ji}\right)x_i y^k_j\end{array}\right) \quad \text{(by reorganization of the terms).}$$

If L_k is defined as follows:

$$L_k(x_{I_k}, y^k_{J_k}, \lambda, \mu) = \sum_{i\in I_k}\left(c_i + \sum_{h\neq k}\lambda^h_i\right)x_i - \sum_{j\in J_k}\lambda^k_j y^k_j + \sum_{i\in I_k}\sum_{\substack{i'\in I_k\\ i'\neq i}}\frac{1}{2}c_{ii'}x_i x_{i'} + \sum_{i\in I_k}\sum_{\substack{j\in J_k\\ j>i}}\left(\frac{1}{2}c_{ij} + \mu_{ij}\right)x_i y^k_j$$
$$+ \sum_{i\in I_k}\sum_{\substack{j\in J_k\\ j<i}}\left(\frac{1}{2}c_{ij} - \mu_{ji}\right)x_i y^k_j,$$

then the Lagrangian function can be written as $L(x,y,\lambda,\mu) = \sum_{k=1}^{p} L_k(x_{I_k}, y^k_{J_k}, \lambda, \mu)$.

Since the clusters form a partition of the set of the variables and since the variables from $y^k_{J_k}$ are peculiar to each cluster, the vectors x_{I_k} and $y^k_{J_k}$ are independent for each k ($1 \leqslant k \leqslant p$). It follows that

$$w(\lambda,\mu) = \max_{\substack{x,y\\ \text{s.t.(c3),(c4),(c5)}}} L(x,y,\lambda,\mu) = \sum_{k=1}^{p}\max_{\substack{x_{I_k}, y^k_{J_k}\\ \text{s.t.(c3),(c4),(c5)}}} L_k(x_{I_k}, y^k_{J_k}, \lambda, \mu),$$

implying the result. □

We see that the computation of the dual function $w(\lambda,\mu)$ can be decomposed into reduced problems. It is well known that the minimum of this dual function yields an upper bound of the optimal value of the initial problem (QKP). Then the dual Lagrangian problem is now

$$\min_{\lambda,\mu} w(\lambda,\mu). \qquad \text{(D)}$$

In order to make the computation of $w(\lambda,\mu)$ easier, we consider the continuous relaxation of the constraint (c4), which becomes the constraint (c4′)

$$y^k_j \in [0;1] \quad (k = 1,\ldots,p) \ (j \in J_k). \qquad \text{(c4′)}$$

For notational expedience, we do not rename w and the functions L_k: it is obvious that the problem (D) in which the constraint (c4′) replaces the constraint (c4) still yields an upper bound of QKP.

2.1. Computation of the dual function

Assume that λ and μ are fixed. Then, the computation of the dual function $w(\lambda, \mu)$ comes down to compute for each cluster k:

$$\max_{\substack{x_{I_k}, y^k_{J_k} \\ \text{s.t.}(c3),(c4'),(c5)}} L_k(x_{I_k}, y^k_{J_k}, \lambda, \mu).$$

To compute this maximum, we proceed by enumeration on the possible values (0 or 1) of the variables of x_{I_k}, called complicating variables. This enumeration, of complexity $O(2^{|I_k|})$, presupposes that the size of the clusters is reasonable. In practice, we have chosen to limit the size of the clusters to five variables, which turned out to be a good compromise between the running time and the quality of the bound (intuitively, the bigger the clusters are, the better is the obtained bound!).

If the values of the variables of x_{I_k} are fixed, then the function $L_k(x_{I_k}, y^k_{J_k}, \lambda, \mu)$ is just a function of the variables of $y^k_{J_k}$. Thus we have to solve for each cluster k the $2^{|I_k|}$ following linear problems:

$$\begin{aligned} (\mathrm{LKP})_{x_{I_k}} \quad & \max \quad L_k(x_{I_k}, y_{J_k}, \lambda, \mu) = \text{const} + \sum_{j \in J_k} p^k_j y^k_j \\ & \text{s.t.} \quad \sum_{j \in J_k} a_r y^k_j \leqslant b' \\ & \qquad\ y^k_j \in [0;1] \quad \forall j \in J_k, \end{aligned}$$

$\forall x_{I_k} \in \{0,1\}^{|1_k|}$, where const and p^k_j $(\forall j \in J_k)$ are integers whose value depends on x_{I_k} and $b' = b - \sum_{i \in I_k} a_i x_i$.

We can recognize a continuous and linear knapsack problem. The problem may even be simplified for we can fix to 0 each variable y^k_j whose coefficient is negative in the objective function since such a variable damages the function (maximization). Moreover it is well known that an optimal solution of this problem can be computed very quickly by sorting the variables in order of the ratio p^k_j / a_j. We use the method proposed by Plateau, 1987 to solve these linear knapsack problems. This method allows us to significantly reduce the running time of the resolution of these problems in comparison with the use of the classical quicksort method.

As a conclusion, given λ and μ, $w(\lambda, \mu)$ can be computed quickly. Then the solution of the dual problem (D), that is to say

$$\min_{\lambda, \mu} w(\lambda, \mu)$$

is computed by the subgradient method of Held et al., 1974. This method yields the searched upper bound:

$$UB = w(\lambda^{(\text{End})}, \mu^{(\text{End})}),$$

where $\lambda^{(\text{End})}$ and $\mu^{(\text{End})}$ indicate the value of the multipliers λ and μ at the last step of the subgradient algorithm.

3. Computational results and conclusion

We have implemented in C language on an HP 9000 workstation the algorithm presented in Section 1 for finding a feasible solution and the algorithm of Section 2 for computing an upper bound. Computational experiments have been performed on randomly generated test problems. As Gallo et al., 1980, Chaillou et al., 1986, Michelon and Veuilleux, 1996, Billionnet and Calmels, 1996 and Hammer and Rader, 1997, the coefficients c_i and c_{ij} of the objective function are integers uniformly distributed between

0 and 100, the coefficients a_i of the capacity constraint are integers uniformly distributed between 1 and 50, while b is an integer randomly chosen between 50 and $\max\left(50, \sum_{i=1}^{n} a_i\right)$. The first experiments we have performed led us to fix the size of the clusters to 5. Computational results about upper and lower bounds for problems involving 100, 300 and 500 variables with a density of the objective function (number of non-zero coefficients divided by $n(n+1)/2$) equal to 25%, 50%, 75% and 100% are reported in Table 1. For a fixed number of variables and a fixed density the tests have been performed on 20 instances and the average results and the corresponding standard deviation are reported. This table shows the tightness of the bounds since the average value of the relative gap between the upper and lower bounds is almost always less than 1%. Moreover we can note that this relative gap decreases with the size of the problem. The CPU times are reasonable. They are less than half an hour for the 500 variables problems with density 25%, 75% and 100% and they are less than an hour for the 500 variables problems with density 50%.

All CPU times are given in seconds and each column from 3 to 7 is divided into two subcolumns: the first one (Av.) concerns average results and the second one (St. d.) concerns the corresponding standard deviation. The description of Table 1 follows:

- Column 3: the ratio $b/\sum a_i$ indicates if the problem is very constrained (ratio close to 0) or not (ratio close to 1).
- Column 4: $(\mathrm{UB}-\mathrm{LB})/\mathrm{LB}$ is the relative gap between the upper bound and the value of the feasible solution.
- Column 5: CPU time for UB and LB is the time spent to compute the feasible solution and the upper bound. Note that the CPU time required for computing LB is negligible.
- Column 6: CPU time required to compute an upper bound such that $(\mathrm{UB}-\mathrm{LB})/\mathrm{LB} \leqslant 10\%$.
- Column 7: CPU time required to compute an upper bound such that $(\mathrm{UB}-\mathrm{LB})/\mathrm{LB} \leqslant 2\%$.

Gallo et al., proposed in 1980 (see Section 1) an exact algorithm for QKP. They reported numerical results concerning their upper bound for problems up to 70 variables. Chaillou et al., 1986 solved problems up to 50 variables but did not indicate the quality of their bound. The semi-definite approach of Helmberg et al., 1996 is illustrated by 12 compiler design problems with a number of variables ranging from 30 to 61. Their upper bound appears to be of very good quality: the relative error gap rarely exceeds 1% and is often negligible. But some of the 12 investigated problems require about 1800 s to compute the bound on a Sun Sparcstation 10: the high computational cost involved seems to be the main practical difficulty with

Table 1
Average computational results for 240 problems

Number of variables	Density (%)	$b/\sum a_i$		(UB − LB)/LB		CPU time (s) for UB and LB		CPU time (s) 10%		CPU time (s) 2%	
		Av.	St. d.	Av. (%)	St. d. (%)	Av.	St. d.	Av.	St. d.	Av.	St. d.
100	25	0.55	0.29	1.13	1.74	85.12	26.72	6.98	6.49	13.01	11.67
	50	0.47	0.24	0.57	0.47	67.94	21.33	3.20	1.61	8.61	4.53
	75	0.49	0.29	0.67	0.63	61.43	14.39	2.01	0.96	5.31	2.69
	100	0.56	0.28	0.59	0.60	51.17	8.78	1.37	0.61	3.36	1.26
300	25	0.46	0.28	1.55	2.14	1102.58	193.13	135.07	90.09	222.58	125.81
	50	0.51	0.28	0.29	0.30	783.86	208.07	37.10	15.38	95.02	40.59
	75	0.55	0.32	0.24	0.22	610.49	135.85	20.08	9.91	49.35	20.76
	100	0.46	0.26	0.31	0.19	415.14	74.63	15.20	3.38	30.21	7.53
500	25	0.47	0.27	0.83	1.24	1142.62	778.37	586.69	296.22	1262.17	522.18
	50	0.55	0.29	0.17	0.23	3197.98	844.56	164.68	141.53	414.14	280.92
	75	0.42	0.27	0.12	0.12	1490.98	407.71	59.06	20.69	111.98	52.89
	100	0.54	0.27	0.16	0.14	1237.34	201.53	49.36	15.89	81.09	20.62

Table 2
Upper bound and computation time – Comparison with existing methods for problems of 40 variables

			Gallo et al. (1980)	Helmberg et al. (1996) Number of variables: 45 [a]	Billionnet and Calmels (1996)	Michelon and Veuilleux (1996)	Our approach
Workstation			IBM 370/168	Sun sparcstation 10	HP 9000	Sun 4.0	HP 9000
Density	5%	(UB − LB)/LB	39.9%	0.5%	–	–	2.3% (2.3%)
		CPU time (s) for UB	5.4 s	508 s	–	–	4.7 s (2.0)
	25%	(UB − LB)/LB	31.4%	–	–	3.5%	1.5% (0.9%)
		CPU time (s) for UB	74.5s	–	–	–	6.5 s (1.4)
	50%	(UB − LB)/LB	37.5%	–	–	5.7%	1.1% (0.5%)
		CPU time (s) for UB	88.5 s	–	–	–	7.1 s (1.2)
	75%	(UB − LB)/LB	–	–	–	4.8%	1.9% (2.0%)
		CPU time (s) for UB	–	–	–	–	6.8 s (1.7)
	100%	(UB − LB)/LB	13.5%	–	1.3% (1.1%)	4.3%	1.8% (0.9%)
		CPU time (s) for UB	14.4 s	–	1082 s (944)	–	7.2 s (1.6)

[a] This column reports average results for problems of 45 variables. The CPU time is reported for only 2 instances.
–Data not available in the reference.

semidefinite approaches. The algorithm proposed by Billionnet and Calmels, 1996 and the one proposed by Michelon and Veuilleux, 1996 both provide tight upper bounds, whose relative mean error gap is less than 3% for problems ranging from 10 to 40 variables. The method of Billionnet and Calmels seems to be faster than the one of Michelon and Veuilleux. Using the first of these two methods, about 1000 s are required to compute the upper bound of problems of 40 variables (density 100%). Note that this is twice as long as the time we need to compute an as tight upper bound for problems of 300 variables on the same workstation.

Table 2 compares our approach with the previous methods in terms of tightness of the bound and of CPU running time for computing this bound. Each column of the table reports average results for several problems of 40 variables. We chose to let a division empty when we could not find the corresponding result in a paper. Whenever it is known, the standard deviation is mentioned in parentheses. As the previous one, this table indicates that our bound is tight and can be very quickly computed.

However the experimental results reported in the previously mentioned papers indicate neither the quality of the bound nor the CPU time for problems involving more than 70 variables. Table 3 compares the results of Hammer and Rader, 1997 for problems of 100 variables, performed on a Sparc server 1000, and our approach. Unfortunately, the authors do not mention in their paper the quality of their upper bound, so we present hereafter a comparison in terms of running time.

Table 3
Comparison with Hammer and Rader's Branch and Bound method for problems of 100 variables

Density (%)	[Hammer and Rader] CPU time (s) for finding the optimum	Our approach	
		CPU time (s) for computing UB	(UB − LB)/LB (%)
25	133	85	1.1
50	109	68	0.6
75	469	61	0.7
100	1428	51	0.6

We also can note that, to the best of our knowledge, no paper of the literature reports computational experiments for instances with more than 100 variables.

References

Billionnet, A., Calmels, F., 1996. Linear programming for the 0-1 quadratic knapsack problem. European Journal of Operational Research 92, 310–325.

Chaillou, P., Hansen, P., Mahieu, Y., 1986. Best network flow bound for the quadratic knapsack problem. Lecture Notes in Mathematics 1403, 226–235.

Chardaire, P., Sutter, A., 1995. A decomposition method for quadratic zero-one programming. Management Science 41 (4), 704–712.

Corneil, D.G., Perl, Y., 1984. Clustering and domination in perfect graphs. Discrete Applied Mathematics 9, 27–39.

De Farias Jr., I.R., Maculan, N., Michelon, P., 1993. Branch and Bound Schemes using Lagrangean Decomposition Techniques for 0-1 Quadratic Knapsack Problems, Logique et Analyse, to appear.

Faye, A., 1994. Programmation quadratique en variables bivalentes sous contraintes linéaires. Application au placement de tâches dans les systèmes distribués et à la partition de graphes, CNAM - Thèse de doctorat.

Gallo, G., Hammer, P.L., Simeone, B., 1980. Quadratic knapsack problems. Mathematical Programming Study 12, 132–149.

Gallo, G., Grigoriadis, M., Tarjan, R., 1989. A fast parametric maximum flow algorithm and applications. SIAM Journal on Computing 18, 30–35.

Hammer, P.L., Rader, D.J., Jr., 1997. Efficient methods for solving quadratic 0-1 knapsack problems. INFOR 35 (3), 170–182.

Held, M., Wolfe, P., Crowder, H.P., 1974. Validation of subgradient optimization. Mathematical Programming 6, 62–88.

Helmberg, C., Rendl, F., Weismantel, R., 1996. A semidefinite programming approach to the quadratic knapsack problem, ZIB, Preprint SC 96-10, 15 p.

Johnson, E.L., Mehrotra, A., Nemhauser, G.L., 1993. Min-cut clustering. Mathematical Programming 62, 133–151.

Körner, F., 1986. A new bound for the quadratic knapsack problem and its use in a branch and bound algorithm. Optimization 17, 643–648.

Laughhunn, D.J., 1970. Quadratic binary programming with applications to capital budgeting problems. Operations Research 18, 454–461.

Michelon, P., Veuilleux, L., 1996. Lagrangean methods for the 0-1 quadratic knapsack problem. European Journal of Operational Research 92, 326–341.

Plateau, G., 1987. Résolution du problème de sac-à-dos linéaire, personal communication.

Rader Jr., D.J., 1997a. Valid inequalities and Facets of the Quadratic 0-1 Knapsack Polytope. Rutcor Research Report 16-97, 11 p.

Rader Jr., D.J., 1997b. Lifting Results for the Quadratic 0-1 Knapsack Polytope, Rutcor Research Report 17-97, 27 p.

Witzzgall, C., 1975. Mathematical methods of site selection for Electronic Message System (EMS), NBS International Report.