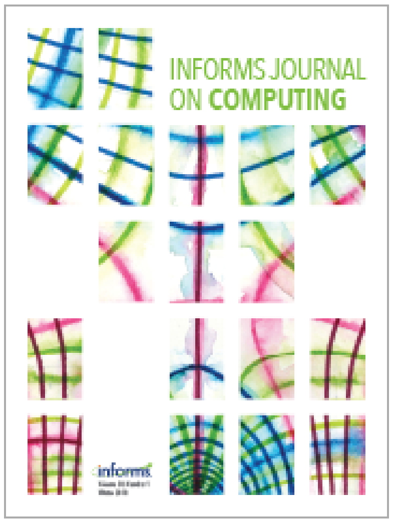

Using a Mixed Integer Programming Tool for Solving the 0-1 Quadratic Knapsack Problem

Alain Billionnet, Éric Soutif,

INFORMS Journal on Computing
Vol. 16, No. 2, Spring 2004, pp. 188–197
ISSN 0899-1499 | EISSN 1526-5528 | 04 | 1602 | 0188

DOI 10.1287/ijoc.1030.0029
© 2004 INFORMS

Using a Mixed Integer Programming Tool for Solving the 0–1 Quadratic Knapsack Problem

Alain Billionnet
CEDRIC—IIE, 18 Allée Jean Rostand, 91025 Evry Cedex, France, alain.billionnet@iie.cnam.fr

Éric Soutif
CERMSEM UMR CNRS 8095, Université Paris 1 Panthéon Sorbonne,
106-112, Boulevard de l'Hôpital, 75647 Paris Cedex 13, France, eric.soutif@univ-paris1.fr

In this paper we will consider the 0–1 quadratic knapsack problem (QKP). Our purpose is to show that using a linear reformulation of this problem and a standard mixed integer programming tool, it is possible to solve the QKP efficiently in terms of computation time and the size of problems considered, in comparison to existing methods. Considering a problem involving n variables, the linearization technique we propose has the advantage of adding only $(n-1)$ real variables and $2(n-1)$ constraints. We present extensive computational results on randomly generated instances and on structured problems coming from applications. For example, the method allows us to solve randomly generated QKP instances exactly with up to 140 variables.

Key words: programming; integer; nonlinear; 0–1 quadratic knapsack; linearization; branch and bound; experiments
History: Accepted by John W. Chinneck; received July 1999; revised February 2002, July 2002, December 2002; accepted December 2002.

1. Introduction

Mixed integer programming is a classical tool in operations research. It can be applied to many problems and in particular to 0–1 quadratic problems in which we are interested. The technique is known and well-tried but must be carefully implemented since some (clumsy) formulations may require a prohibitive computation time (see, for example, Beale 1988 and Salkin and Mathur 1989). If a particular formulation of a problem fails, another one whose continuous relaxation may yield a less sharp bound may appear more efficient in practice. This may be seen when considering the 0–1 quadratic knapsack problem (QKP). We propose two 0–1 mixed integer formulations of QKP, denoted by LIN1 and LIN2, in order to show that it is possible to solve large QKP instances using only mixed integer programming and standard, commercially available software.

In §2 we will present QKP, indicate its classical quadratic formulation, and briefly mention the state of the art. In §3 we will present a first mixed integer linear reformulation of QKP, LIN1, its origin, its specificity, and we will compare it, from a theoretical point of view, with the classical 0–1 linearization. In §4 we will present LIN2, a potential improvement over the LIN1 formulation. In §5, we will present computational results using the proposed formulations and a mixed integer programming tool, CPLEX 6.0 (ILOG 1998), for solving randomly generated instances of QKP. We also provide results concerning instances that appear in compiler design. We will compare these results to those in the literature. In §6 we present our conclusions.

2. Presentation of QKP

The QKP was introduced in Gallo et al. (1980) and it may be formulated as follows:

Problem QKP

$$\max \quad f(x) = \sum_{i=1}^{n} c_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} x_i x_j \tag{1}$$

$$\text{subject to} \quad \sum_{i=1}^{n} a_i x_i \le b \tag{2}$$

$$x_i \in \{0, 1\} \quad \forall i \in N \tag{3}$$

where the coefficients c_i, a_i $(i = 1, \ldots, n)$ and c_{ij} $(1 \le i < j \le n)$ are non-negative integers, b is an integer such that $0 < b < \sum_{i=1}^{n} a_i$, and $N = \{1, \ldots, n\}$. Constraint (2) is the capacity constraint. We will not refer back to the results obtained by various methods for solving QKP here; consult Billionnet et al. (1999), in which the state of the art with regard to QKP is presented. However, we must now update Billionnet et al. (1999) by including two recent algorithms that represent real progress in the solution of QKP: Billionnet et al. (1997), Caprara et al. (1999), who developed an exact method based on Lagrangian relaxation. Their highly efficient algorithm enabled them to optimize QKP instances with up to 120 variables for low-density problems (the density is the proportion of non-zero coefficients in the objective function) and up to 400 variables for high-density problems. However, the best methods previously published relating to this problem do not offer any solution to problems involving more than 100 variables. An exact method is also proposed in Billionnet et al. (1997) using an upper bound obtained by a specific Lagrangian decomposition. The two approaches (Billionnet et al. 1997, Caprara et al. 1999) complement one another since the method proposed in Billionnet et al. (1997) appears highly efficient with regard to problems with low and medium density with up to 300 variables. The latter method also solves high-density problems with up to 150 variables.

3. Presentation of the Linearization

In this section we will present a linearization technique that transforms QKP into an equivalent 0–1 mixed integer problem. This linearization is a variant of that proposed in Glover (1975) for resolution of the general problem of optimizing a 0–1 quadratic function subject to linear constraints adapted to the particular case of QKP. The idea of this linearization consists of the rewriting of the quadratic part of the objective function as a sum of terms $x_i\varphi_i(x)$, where $\varphi_i(x)$ represents part of the gain brought to the objective function by fixing x_i to 1:

$$f(x) = \sum_{i=1}^{n} c_i x_i + \sum_{i=1}^{n-1} x_i \varphi_i(x)$$

where

$$\varphi_i(x) = \sum_{j=i+1}^{n} c_{ij} x_j.$$

Let us consider the following formulation of the initial problem:

Formulation LIN1

$$\begin{aligned} \max \quad & \sum_{i=1}^{n} c_i x_i + \sum_{i=1}^{n-1} z_i \\ \text{subject to} \quad & \sum_{i=1}^{n} a_i x_i \le b \\ & z_i \le \overline{\varphi}_i x_i \quad \forall i \in \{1, \ldots, n-1\} && (4) \\ & z_i \le \varphi_i(x) \quad \forall i \in \{1, \ldots, n-1\} && (5) \\ & x_i \in \{0, 1\} \quad \forall i \in N && (6) \end{aligned}$$

with $\forall i \in \{1, \ldots, n-1\}$, $\overline{\varphi}_i = \text{opt}(\text{KP}_i)$, the optimal value of KP_i, where KP_i is the following 0–1 knapsack problem (when x is feasible and $x_i = 1$, $\overline{\varphi}_i$ is an overestimation of $\varphi_i(x)$):

Problem KP_i

$$\begin{aligned} \max \quad & \varphi_i(x) \\ \text{subject to} \quad & \sum_{j=i+1}^{n} a_j x_j \le b - a_i \\ & x_j \in \{0, 1\} \quad \forall j \in \{i+1, \ldots, n\} && (7) \end{aligned}$$

The main interest of this formulation lies in the low number of added variables and linearization constraints in comparison with the initial formulation. In fact, this formulation contains the following additional variables and constraints:

- $(n-1)$ real variables: the variables z_i. Each variable z_i exactly corresponds to the term $x_i\varphi_i(x)$ at the optimum of the problem. Since LIN1 is a maximization problem where all the coefficients of z_i are (strictly) positive and since each variable z_i appears in the set of the constraints once in (4) and once in (5), we have that at the optimum, $z_i = \min(\overline{\varphi}_i x_i, \varphi_i(x)) = x_i\varphi_i(x)$;
- $2(n-1)$ constraints: (4) and (5). Constraints (4) require a variable z_i to be equal to 0 when the corresponding variable x_i equals 0. The $\overline{\varphi}_i$ coefficients are chosen to be large enough to allow the variable z_i to have the value $\varphi_i(x) = \sum_{j=i+1}^{n} c_{ij} x_j$ when x_i equals 1.

Proposition 1. *Problems* QKP *and* LIN1 *are equivalent in the following sense. Given any optimal solution* $x = (x_1, \ldots, x_n)$ *in* QKP, *there exists a vector* $z = (z_1, \ldots, z_n)$ *such that* (x, z) *is feasible in* LIN1, *with the same objective value. Conversely, given any optimal solution* (x, z) *in* LIN1, *the corresponding solution* x *is feasible in* QKP, *with the same objective value.*

The proof is obvious since we know that at the optimum of LIN1 $z_i = x_i\varphi_i(x)$.

We will now compare the 0–1 mixed integer linearization LIN1 to the classical 0–1 linearization CLIN described below. This consists of replacing each product of 0–1 variables $x_i x_j$ by a variable w_{ij} in the quadratic formulation and adding a set of inequalities that require the variable w_{ij} to be equal to the product $x_i x_j$ at the optimum:

Formulation CLIN

$$\begin{aligned} \max \quad & \sum_{i=1}^{n} c_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} w_{ij} \\ \text{subject to} \quad & \sum_{i=1}^{n} a_i x_i \le b \\ & w_{ij} \le x_i \quad \forall (i,j) \in N^2 \text{ s.t. } i<j \text{ and } c_{ij} > 0 && (8) \end{aligned}$$

$$w_{ij} \le x_j \quad \forall (i,j) \in N^2 \text{ s.t. } i<j \text{ and } c_{ij}>0 \quad (9)$$

$$x_i \in \{0,1\} \quad \forall i \in N \quad (10)$$

Let us note that the 0–1 linearization CLIN involves at most (and exactly for 100% density problems) $n(n-1)/2$ additional real variables and $n(n-1)$ additional constraints, compared to the initial formulation of QKP. We should also note that solving an instance of QKP using the CLIN formulation necessitates the solution of one 0–1 mixed integer program, whereas the LIN1 formulation requires a preprocessing algorithm that solves $(n-1)$ 0–1 linear knapsack problems.

In order to compare CLIN and LIN1 (that are both equivalent to QKP), we will consider $\overline{\text{CLIN}}$ and $\overline{\text{LIN1}}$, the continuous relaxations of CLIN and LIN1 obtained by replacing constraints (6) and (10) with the constraint $0 \le x_i \le 1$ $\forall i \in N$, in both formulations.

Proposition 2. *$\overline{\text{CLIN}}$ and $\overline{\text{LIN1}}$ are not comparable in the sense that some* QKP *instances are such that $\overline{\text{LIN1}}$ yields a sharper upper bound (of lower value) than the one given by $\overline{\text{CLIN}}$, while other instances may yield the inverse result.*

Proof. Consider the two following instances of QKP:

Instance QKP_1

$$\begin{aligned} \max \quad & (35x_1+5x_2+91x_3+44x_1x_2 \\ & +55x_1x_3+23x_2x_3) \\ \text{subject to} \quad & 45x_1+26x_2+41x_3 \le 57 \\ & x_i \in \{0,1\} \quad \forall i \in \{1,2,3\} \end{aligned}$$

and

Instance QKP_2

$$\begin{aligned} \max \quad & (43x_3+22x_4+11x_1x_2+57x_1x_4+95x_1x_5 \\ & +23x_2x_5+83x_3x_4+5x_4x_5) \\ \text{subject to} \quad & 4x_1+7x_2+13x_3+32x_4+38x_5 \le 85 \\ & x_i \in \{0,1\} \quad \forall i \in \{1,2,3,4,5\} \end{aligned}$$

For QKP_1, $\text{opt}(\overline{\text{LIN1}}_1) \simeq 103.4 < \text{opt}(\overline{\text{CLIN}}_1) \simeq 128.8$ and for QKP_2, $\text{opt}(\overline{\text{LIN1}}_2) \simeq 313.7 > \text{opt}(\overline{\text{CLIN}}_2) \simeq 312.2$. □

Although the continuous relaxations of CLIN and LIN1 cannot be compared, it is possible to show the determining role that the coefficients $\overline{\varphi}_i$ (constraints (4)) play in the quality of the bound obtained by continuous relaxation of LIN1: Let $\overline{\text{LIN0}}$ be the linearization obtained by replacing $\overline{\varphi}_i$ by $\sum_{j=i+1}^n c_{ij}$ (which is another overestimation of $\varphi_i(x)$ but less sharp than $\overline{\varphi}_i$) in the formulation $\overline{\text{LIN1}}$, constraints (4) becoming constraints (4′). $\overline{\text{LIN0}}$ is less sharp than $\overline{\text{CLIN}}$:

Proposition 3.
(i) $\text{opt}(\overline{\text{LIN0}}) \ge \text{opt}(\overline{\text{CLIN}})$;
(ii) *there are some instances in which this inequality is strict.*

Proof.
(i) Given any optimal solution (x,w) in $\overline{\text{CLIN}}$ we can define the solution (x,z), which is feasible in $\overline{\text{LIN0}}$ with the same objective value by writing $z_i = \sum_{j=i+1}^n c_{ij} w_{ij}$, $\forall i \in \{1,\ldots,n-1\}$:

$$\begin{aligned} &(x,z) \text{ satisfies } (4'){:}\ \forall (i,j) \in N^2 \text{ s.t. } i<j \text{ and } c_{ij}>0, \\ &\qquad (w_{ij} \le x_i,\ (8),\ \text{and } c_{ij} \ge 0) \\ &\qquad \Rightarrow \Big(z_i = \textstyle\sum_{j=i+1}^n c_{ij} w_{ij} \le \Big(\sum_{j=i+1}^n c_{ij}\Big) x_i\Big) \\ &(x,z) \text{ satisfies } (5){:}\ \forall (i,j) \in N^2 \text{ s.t. } i<j \text{ and } c_{ij}>0, \\ &\qquad (w_{ij} \le x_j,\ (9),\ \text{and } c_{ij} \ge 0) \\ &\qquad \Rightarrow \Big(z_i = \textstyle\sum_{j=i+1}^n c_{ij} w_{ij} \\ &\qquad\qquad \le \textstyle\sum_{j=i+1}^n c_{ij} x_j = \varphi_i(x)\Big). \end{aligned}$$

(ii) Let us consider the following QKP instance:

Instance QKP_3

$$\begin{aligned} \max \quad & (5x_1x_2+4x_1x_3+5x_2x_3+7x_2x_4) \\ \text{subject to} \quad & 3x_1+2x_2+2x_3+2x_4 \le 5 \\ & x_i \in \{0,1\} \quad \forall i \in \{1,2,3,4\} \end{aligned}$$

For this instance, $\text{opt}(\overline{\text{LIN0}}_3) \simeq 11.97 > \text{opt}(\overline{\text{CLIN}}_3) \simeq 11.66$. □

As expected, the bound provided by LIN0 is weaker than the one provided by CLIN. LIN1 yields a better bound than LIN0, but the computational results (see §5) show that despite the preprocessing required (involving a NP-complete subproblem) the bound provided by LIN1 often remains weaker than the one provided by CLIN. However, the small number of variables and constraints in LIN1 appears to be an important asset for solving large instances of QKP.

4. Improvement of the Proposed Linearization

As stated in §3, the choice of coefficients $\overline{\varphi}_i$ with regard to the quality of the bound given by $\overline{\text{LIN1}}$ is important. We saw that it is preferable to select these coefficients as precisely as possible. Therefore, we suggest the reformulation LIN2 of QKP, whose continuous relaxation yields a better upper bound than the one provided by $\overline{\text{LIN1}}$, as we will see in Proposition 5. Unlike LIN1, the formulation LIN2 also requires finding a solution $\tilde{x}$ that is feasible in QKP and that is as accurate as possible. A good heuristic algorithm very often yielding the optimum solution to the problem is proposed in Billionnet and Calmels (1996). It is also used in Billionnet et al. (1999) and

Caprara et al. (1999). Let $\alpha = f(\widetilde{x})$ be the value of this feasible solution.

Formulation LIN2

$$\max \quad \sum_{i=1}^{n} c_i x_i + \sum_{i=1}^{n-1} z_i + \sum_{i=1}^{n-1} \underline{\varphi}_i x_i$$

$$\text{subject to} \quad \sum_{i=1}^{n} a_i x_i \le b \tag{11}$$

$$z_i \le \left(\overline{\varphi}_i - \underline{\varphi}_i\right) x_i \quad \forall i \in \{1,\dots,n-1\} \tag{12}$$

$$z_i \le \varphi_i(x) - \underline{\varphi}_i \quad \forall i \in \{1,\dots,n-1\} \tag{13}$$

$$x_i \in \{0,1\} \quad \forall i \in N \tag{14}$$

with $\forall i \in \{1,\dots,n-1\}$,

$$\underline{\varphi}_i = \begin{cases} \text{opt}(\text{LP}_i) & \text{if } \widetilde{x}_i = 1 \\ \text{opt}(\text{LP}'_i) & \text{if } \widetilde{x}_i = 0. \end{cases}$$

LP_i corresponds to the following linear problem:

Problem LP$_i$

$$\min \quad \varphi_i(x)$$

$$\text{subject to} \quad \sum_{j=1}^{n} a_j x_j \le b \tag{15}$$

$$z_k \le \overline{\varphi}_k x_k \quad \forall k \in \{1,\dots,n-1\} \tag{16}$$

$$z_k \le \varphi_k(x) \quad \forall k \in \{1,\dots,n-1\} \tag{17}$$

$$\sum_{j=1}^{n} c_j x_j + \sum_{k=1}^{n-1} z_k \ge \alpha \tag{18}$$

$$x_j \in [0,1] \quad \forall j \in N \tag{19}$$

LP'_i is the linear problem obtained by adding the following constraint to LP_i:

$$x_i = 0 \tag{20}$$

Recall that $\overline{\varphi}_i = \text{opt}(\text{KP}_i)$ (see §3).

Proposition 4. *Problems* QKP *and* LIN2 *are equivalent in the following sense. Given any optimal solution* $x = (x_1,\dots,x_n)$ *in* QKP, *there exists a vector* $z = (z_1,\dots,z_n)$ *such that* (x,z) *is feasible in* LIN2, *with the same objective value. Conversely, given any optimal solution* (x,z) *in* LIN2, *the corresponding solution* x *is feasible in* QKP, *with the same objective value.*

Proof. First, notice that LP_i and LP'_i admit at least one feasible solution since $(\widetilde{x},\widetilde{z})$, where $\widetilde{z}_k = \widetilde{x}_k \varphi_k(\widetilde{x})$ for all $k \in \{1,\dots,n-1\}$, is a feasible solution of LP_i if $\widetilde{x}_i = 1$ and is a feasible solution of LP'_i if $\widetilde{x}_i = 0$.

1. opt(LIN2) ≥ opt(QKP). Let x^* be an optimal solution of QKP and (x^*,z^*) be such that $z_i^* = x_i^*(\varphi_i(x^*) - \underline{\varphi}_i)$ for all $i \in \{1,\dots,n-1\}$. We must prove that (x^*,z^*) is a feasible solution of LIN2 with objective value equal to $f(x^*)$. The objective value of LIN2 for (x^*,z^*) is

$$\sum_{i=1}^{n} c_i x_i^* + \sum_{i=1}^{n-1} x_i^*(\varphi_i(x^*) - \underline{\varphi}_i) + \sum_{i=1}^{n-1} \underline{\varphi}_i x_i^*$$
$$= \sum_{i=1}^{n} c_i x_i^* + \sum_{i=1}^{n-1} x_i^* \varphi_i(x^*) = f(x^*).$$

Obviously, x^* satisfies (11) and (14). Now we must prove that (x^*,z^*) also satisfies (12) and (13).

(a) *Case* 1. $x_i^* = 1$
$x_i^* = 1 \Rightarrow z_i^* = \varphi_i(x^*) - \underline{\varphi}_i$ and (13) is satisfied. ($x_i^* = 1$ and x^* satisfies (11)) $\Rightarrow 0 \le \varphi_i(x^*) \le \overline{\varphi}_i$ (by definition of $\overline{\varphi}_i$). Therefore, $z_i^* = \varphi_i(x^*) - \underline{\varphi}_i \le \overline{\varphi}_i - \underline{\varphi}_i = (\overline{\varphi}_i - \underline{\varphi}_i)x_i^*$ and (12) is satisfied.

(b) *Case* 2. $x_i^* = 0$
$x_i^* = 0 \Rightarrow z_i^* = 0$ and (12) is satisfied. Let $(x^*,\underline{z})$ be such that $\underline{z}_k = x_k^* \varphi_k(x^*)$ for all $k \in \{1,\dots,n-1\}$. We must prove that $(x^*,\underline{z})$ is a feasible solution of LP_i or LP'_i of value $\varphi_i(x^*)$. $(x^*,\underline{z})$ satisfies (15); $x_k^* \varphi_k(x^*) \le x_k^* \overline{\varphi}_k$ since by definition $\varphi_k(x) \le \overline{\varphi}_k$ if $x_k = 1$, which implies that (16) is satisfied; $x_k^* \varphi_k(x^*) \le \varphi_k(x^*) \Rightarrow$ (17) is satisfied; the left-hand side of inequality (18) is equal to $f(x^*)$, which is greater than or equal to α, and (18) is satisfied; last the constraint (20) is satisfied since we suppose here that $x_i^* = 0$. Finally, since $(x^*,\underline{z})$ is a feasible solution of LP_i or LP'_i, we obtain $\underline{\varphi}_i \le \varphi_i(x^*)$ and (13) is satisfied.

2. opt(QKP) ≥ opt(LIN2). Let (x^*,z^*) be an optimal solution of LIN2. Obviously, x^* is a feasible solution of QKP. Now we must prove that $f(x^*) \ge \sum_{i=1}^{n} c_i x_i^* + \sum_{i=1}^{n-1} z_i^* + \sum_{i=1}^{n-1} \underline{\varphi}_i x_i^*$. To show this inequality we will prove that $x_i^* \varphi_i(x^*) \ge z_i^* - \underline{\varphi}_i x_i^*$ for all $i \in \{1,\dots,n-1\}$. $x_i^* = 0 \Rightarrow z_i^* \le 0$ (by (12)) $\Rightarrow x_i^* \varphi_i(x^*) \ge z_i^* - \underline{\varphi}_i x_i^*$; $x_i^* = 1 \Rightarrow z_i^* \le \varphi_i(x^*) - \underline{\varphi}_i$ (by (13)) $\Rightarrow x_i^* \varphi_i(x^*) \ge z_i^* + \underline{\varphi}_i x_i^* \ge z_i^* - \underline{\varphi}_i x_i^*$ since $\underline{\varphi}_i \ge 0$ (remember that $\varphi_i(x) \ge 0$ for all $x \in \{0,1\}^n$). □

The reformulation LIN2 involves the same number of variables and constraints as LIN1 and also requires a preprocessing algorithm, namely, the computation of coefficients $\overline{\varphi}_i$ and $\underline{\varphi}_i$, $\forall i \in \{1,\dots,n-1\}$. Coefficients $\overline{\varphi}_i$ are the same as those used in LIN1, so they are obtained by solving $(n-1)$ 0–1 knapsack problems, i.e., the problems KP_i. The computation of coefficients $\underline{\varphi}_i$ is carried out by solving $(n-1)$ linear problems, the problems LP_i or LP'_i, involving $(2n-1)$ variables, $2n$ or $(2n+1)$ constraints, and n bounds concerning the variables. These problems always have at least one solution $(x = \widetilde{x})$.

Whereas each coefficient $\overline{\varphi}_i$ represents an overestimation of the term $\varphi_i(x)$ (when x satisfies the capacity constraint (11) and $x_i = 1$), each coefficient $\underline{\varphi}_i$ represents an underestimation of $\varphi_i(x)$ that takes the knowledge of the feasible solution $\widetilde{x}$ into account (thanks to constraints (18)).

Proposition 5. $\overline{\text{LIN2}}$, *obtained by continuous relaxation of* LIN2, *is at least as sharp as* $\overline{\text{LIN1}}$.

Proof. Given any optimal solution (x^*, z^*) in $\overline{\text{LIN2}}$, we are able to define a solution (x^*, z) that is feasible in $\overline{\text{LIN1}}$ and with the same objective value, by writing $z_i = z_i^* + \underline{\varphi}_i x_i^*$ $\forall i \in \{1, \ldots, n-1\}$. □

The relations between the different linearizations are summarized as:

$$\text{opt}(\overline{\text{LIN2}}) \leq \text{opt}(\overline{\text{LIN1}}) \leq \text{opt}(\overline{\text{LIN0}}) \quad \text{and}$$

$$\text{opt}(\overline{\text{CLIN}}) \leq \text{opt}(\overline{\text{LIN0}}).$$

Proposition 6. $\overline{\text{CLIN}}$ *and* $\overline{\text{LIN2}}$ *are not comparable in the sense that some* QKP *instances are such that* $\overline{\text{LIN2}}$ *yields a sharper upper bound (of lower value) than that given by* $\overline{\text{CLIN}}$, *while other instances may yield the inverse result.*

Proof. Consider the following QKP instance:

Instance QKP_4

$$\begin{aligned} \max \quad & (x_1 + 7x_2 + 5x_3 + 3x_1x_2 + 8x_1x_3 + 8x_2x_3) \\ \text{subject to} \quad & 5x_1 + 5x_2 + 10x_3 \leq 15 \\ & x_i \in \{0,1\} \quad \forall i \in \{1,2,3\} \end{aligned}$$

For QKP_4, $\text{opt}(\overline{\text{CLIN}_4}) = 24 > \text{opt}(\overline{\text{LIN2}_4}) \simeq 22.18$ ($\overline{\varphi}_1 = 8$, $\overline{\varphi}_2 = 8$ and, for $\tilde{x} = (1,0,1)$, $\underline{\varphi}_1 \simeq 4.14$ and $\underline{\varphi}_2 = 8$). □

It is more difficult to find a small instance of QKP such as $\text{opt}(\overline{\text{CLIN}}) < \text{opt}(\overline{\text{LIN2}})$ but the results of Table 1 in §5 prove that there are such instances. For example, for 80 variables and a density of 5%, the average value of $(\text{opt}(\overline{\text{CLIN}}) - \text{LB})/\text{LB}$ is equal to 1.47%, whereas the average value of $(\text{opt}(\overline{\text{LIN2}}) - \text{LB})/\text{LB}$ is equal to 4.95%. This shows that there is at least one instance of QKP such that $\text{opt}(\overline{\text{CLIN}}) < \text{opt}(\overline{\text{LIN2}})$. □

5. Computational Experiments

All the mathematical programs CLIN, LIN1, and LIN2, in addition to programs generated during the preprocessing of LIN1 and LIN2, i.e., programs KP_i, LP_i, and LP'_i, were solved using CPLEX 6.0 (ILOG 1998) on a Pentium II 300 MHz computer. In §5.1 we will present computational results concerning randomly generated QKP instances and in §5.2, computational results concerning structured problems arising from compiler design.

5.1. Randomly Generated Instances

These experiments were performed on randomly generated problems. Some of these problems are available at http://cermsem.univ-paris1.fr/soutif/QKP. As in Gallo et al. (1980), Chaillou et al. (1986), Michelon and Veuilleux (1996), Billionnet et al. (1997, 1999), and Caprara et al. (1999), the coefficients c_i and c_{ij} of the objective function are integers that are uniformly distributed between 0 and 100. The coefficients a_i of the capacity constraint are integers that are uniformly distributed between 1 and 50, and b is an integer randomly chosen between 50 and $\max(50, \sum_{i=1}^{n} a_i)$. Preliminary experiments showed that it was worthwhile to sort the variables; the influence of the order will be discussed more precisely in §5.1.1. For both linearizations LIN1 and LIN2, finding an accurate feasible solution is an important aspect. This solution may be found, for example, by using the heuristic algorithm proposed in Billionnet and Calmels (1996). The corresponding lower bound is used not only in calculating coefficients $\underline{\varphi}_i$ but also to reduce the size of the search tree. To achieve this, the lower bound is used as supplementary data to the mixed integer programming software. Note that the computation time of this feasible solution is negligible (always less than 1 second, even for large problems).

In Table 1 we compare the three linearizations CLIN, LIN1, and LIN2 on instances with $n = 80$, 100, 120, and 140 for five values of the density (5%, 25%, 50%, 75%, and 100%). Each line of this table gives the average results obtained by the three methods for 20 randomly generated instances. When LIN1 or LIN2 is used to solve an instance, the variables are sorted by increasing weight in every case, except for the 140-variable instances having a 5% density, where the inverse order is preferable (see §5.1.1). When the 20 instances could not be solved exactly within the time limit of 40,000 seconds ($\simeq$11 hours), the results only concern the solved instances. For each type of problems in Table 1 we present:

- the sharpness of the upper bound obtained by solving continuous relaxations of the linearizations (rel. gap),
- the running time required by the preprocessing, i.e., for solving problems KP_i, LP_i, and LP'_i,
- the running time corresponding to the upper bound (T. UB),
- the number of nodes in the search tree associated with the branch-and-bound procedure (# nodes),
- the total CPU time required to solve the instances considered (total time),
- the number of instances (among the twenty considered) that may be solved exactly within the permitted running time (SI).

We measured the degree of difficulty of the randomly generated problems in the following way: For each considered instance, we built ten feasible solutions by randomly filling the knapsack (to the maximum), we retained the best value, and we calculated the relative gap between this value and the optimum value. In all the instances considered, this relative gap is on average between 29% and 59%. This indicates that a randomly generated feasible solution is generally a poor one. From this point of view, finding

Table 1 **Numerical Comparison Between CLIN, LIN1, and LIN2**

Data		CLIN					LIN1						LIN2					
# var. (*n*)	D. (%)	Rel. gap (%)	T. UB (s)	# Nodes	Total time (s)	SI	Rel. gap (%)	T. UB (s)	T. PP1 (s)	# Nodes	Total time (s)	SI	Rel. gap (%)	T. UB (s)	T. PP2 (s)	# Nodes	Total time (s)	SI
80	5	1.47	0.16	237	14.45	20	5.34	0.08	0.03	2,801	4.99	20	4.95	2.21	0.03	2,633	6.96	20
	25	3.68	0.75	719	88.78	20	8.20	0.34	0.05	18,630	45.20	20	7.04	4.10	0.07	17,216	43.22	20
	50	4.52	2.71	3,617	670.07	20	7.16	0.86	0.06	33,291	101.78	20	5.22	5.08	0.06	14,356	43.24	20
	75	2.54	5.98	8,202	3,556.43	19	4.33	1.10	0.06	53,839	118.10	20	2.36	6.09	0.06	5,001	17.06	20
	100	3.75	9.54	23,511	6,553.75	17	5.14	2.81	0.07	39,923	107.19	20	2.88	7.95	0.07	8,454	37.20	20
100	5	2.87	0.28	252	29.31	20	12.10	0.24	0.05	398,541	894.59	20	11.85	3.99	0.05	398,169	921.28	20
	25	2.30	1.88	2,418	479.19	20	7.61	0.64	0.09	165,650	595.55	20	6.67	7.83	0.11	115,427	374.45	20
	50	2.91	7.55	6,208	2,086.25	19	6.82	1.55	0.10	107,090	424.18	20	4.40	11.29	0.12	22,047	92.44	20
	75	5.98	16.82	15,135	8,046.07	17	6.01	2.66	0.13	86,432	347.29	20	3.39	12.69	0.11	28,574	90.52	20
	100	1.11	20.66	11,249	7,197.80	11	2.78	3.73	0.11	49,324	143.33	20	1.22	13.84	0.09	3,796	27.58	20
120	5	1.39	0.43	423	71.64	20	7.66	0.24	0.08	903,638	2,918.62	20	7.44	6.85	0.07	884,084	2,773.84	20
	25	1.92	4.11	4,949	1,707.97	20	5.57	0.71	0.13	421,800	2,096.72	17	4.46	14.55	0.19	315,208	1,548.96	17
	50	2.24	17.54	18,695	11,033.78	16	4.81	1.39	0.16	814,752	4,607.90	18	3.21	19.94	0.19	237,234	1,249.08	20
	75	1.13	25.72	2,709	3,256.62	12	3.98	4.26	0.16	546,122	2,969.34	19	2.44	22.70	0.16	119,781	623.51	20
	100	10.33	68.17	11,851	11,989.39	7	4.01	6.19	0.16	635,518	2,477.52	20	1.81	24.70	0.13	47,823	190.55	20
140	5	0.86	0.65	616	169.41	20	7.31	0.19	0.15	97,420	550.34	20	6.96	15.47	0.11	95,852	539.73	20
	25	1.22	6.32	3,508	2,171.18	20	7.54	1.09	0.29	620,612	3,800.42	17	6.27	23.64	0.24	672,920	3,959.34	18
	50	2.24	30.97	14,825	15,059.24	10	5.34	2.26	0.30	1,074,721	7,785.54	16	3.30	31.24	0.24	292,461	1,636.62	16
	75	1.28	68.02	9,180	12,892.30	9	4.71	6.08	0.27	185,530	989.22	16	3.05	35.83	0.23	560,925	3,758.89	20
	100	1.63	109.19	5,859	17,399.07	6	2.78	7.44	0.23	1,272,492	3,854.74	18	1.48	38.42	0.18	77,035	882.71	19

Notes. D. (%), density (proportion of non-zero linear and quadratic coefficients in the objective function); rel. gap (%), relative gap between the lower bound (LB), i.e., the value of an accurate feasible solution and the upper bound (UB): relative gap $= (\text{UB} - \text{LB})/\text{LB}$; T. PP1 (s), CPU time (in seconds) required to compute the coefficients $\overline{\varphi}_i$ used in LIN1; T. PP2 (s), CPU time (in seconds) required to compute the coefficients $\overline{\varphi}_i$ and $\underline{\varphi}_i$ used in LIN2; T. UB (s), CPU time (in seconds) required by the computation of the upper bound obtained by continuous relaxation (evaluation at the root of the branching tree); # nodes, number of nodes investigated during the branch-and-bound algorithm; total time (s), CPU time (in seconds) corresponding to the exact integer solution, including the preprocessing and the computation of an initial feasible solution. SI, number of instances (over 20) solved within the fixed time limit of 40,000 seconds.

an optimal solution for a randomly generated QKP instance is not a simple problem to resolve.

In Table 1 the results indicate that LIN1 and LIN2 are almost always considerably more efficient than CLIN with regard to the CPU time, although the corresponding upper bounds are often less accurate than that corresponding to CLIN. The number of variables and constraints in LIN1 and LIN2 compensates for this disadvantage: CLIN contains $O(n^2)$ variables and $O(n^2)$ constraints, whereas LIN1 and LIN2 contain only $O(n)$ variables and $O(n)$ constraints. Therefore, for a 100-variable instance of 100% density, there are 5,050 variables and 9,901 constraints in CLIN, and only 199 variables and 199 constraints in LIN2. Table 1 also shows that LIN1 and LIN2, contrary to CLIN, enable us to solve high-density instances with 140 variables. The results in Table 1 also show that CLIN is a good method for solving low-density QKP instances. This 0–1 linearization of the problem enables us to solve all of the considered instances of up to 140 variables when the density equals 5% or 25%. CLIN is somewhat more efficient than LIN1 and LIN2 for these densities where fewer linearization variables w_{ij} are required. When using LIN1 the preprocessing time is on average equal to 0.6% of the total time, and that of LIN2 is equal to 2%.

Table 2 summarizes the numerical comparison between the three linearizations, with regard to CPU time. In spite of the fact that LIN1 does not appear in Table 2, it remains a satisfactory compromise between CLIN and LIN2 if one wishes to solve large QKP instances and is even simpler to implement than LIN2. In fact, implementing LIN1 requires only computation of coefficients $\overline{\varphi}_i$ during preprocessing, whereas LIN2 also requires computation of an accurate feasible solution and computation of the coefficients $\underline{\varphi}_i$.

5.1.1. Initial Feasible Solution and Sorting the Variables. Experiments showed that the knowledge of a good initial feasible solution is an important aspect in solving QKP using our approaches. The influence of the accuracy of this solution in solving instances with 140 variables using LIN2 is indicated in Table 3. We measured the impact of this solution in the following way: We considered a randomly generated feasible solution obtained by filling the knapsack as long as the capacity constraint was satisfied, substituting it for the heuristic solution found by using the method of Billionnet and Calmels (1996). The number of instances that can be solved within the time limit decreases, and the CPU time required for solving them increases. This result can be explained by the deterioration (of about 30%) of the quality of the upper bound, since LIN2 exploits the knowledge of a good initial solution, and also by the fact that the less accurate solution stops the solver from cutting as many branches.

Table 2 **Mixed Integer Formulations Recommended**

	Density	
n	5%, 25%	50%, 75%, 100%
80	LIN2	LIN2
100	CLIN	LIN2
120	CLIN	LIN2
140	CLIN	LIN2

The results in Table 3 also show the influence of the order of the variables on the solution of QKP by LIN2. We can additionally note that when i increases, problems KP_i become progressively less constrained: The number of variables involved in constraint (7) progressively decreases, so that the last coefficients $\overline{\varphi}_i$ are almost always equal to $\sum_{j=i+1}^{n} c_{ij}$ (see §3). By sorting the variables according to their increasing weight (a_i), we can slow down this phenomenon, so that more coefficients $\overline{\varphi}_i$ get a value lower than $\sum_{j=i+1}^{n} c_{ij}$, and as we have seen in §3, it was crucial to decrease the coefficients $\overline{\varphi}_i$ as much as possible. This observation led us to consider both the formulations LIN1 and LIN2 rather than LIN0. For example, in Figure 1, for a 140-variable instance of density 100%, we presented the value of the ratio $(\sum_{j=i+1}^{n} c_{ij} - \overline{\varphi}_i)/\sum_{j=i+1}^{n} c_{ij}$ for $i = 1, \ldots, 140$ for both decreasing and increasing orders. We notice that the value of $\overline{\varphi}_i$ is much more significant compared to $\sum_{j=i+1}^{n} c_{ij}$ for the latter. We also carried out tests for solving QKP instances with 140 variables through the formulation LIN2 and using the three following orders of the variables:

- the natural order (numbering the variables by randomly generating the instances),
- decreasing order of their weight,
- increasing order of their weight.

The results in Table 3 show that the increasing order is the most efficient, except for the 5% low-density problems. Compared to the natural order of the variables, their preliminary sorting by increasing order of their weight significantly improves the upper bound (by about 40% on average, taking all densities into account; see columns (UB−opt/opt)). As a consequence, this preliminary sorting also speeds up the solution of LIN2 for all densities over 5%. This sorting enables the average time required for solving the 140-variable instances (by taking all densities greater than 5% into account) to go from 5,732 to 2,446 seconds, an improvement of 57%. Moreover, the increasing order enables many more instances to be solved. Conversely, sorting the variables by decreasing weight is particularly useful for 5% density problems. In this case the coefficients $\overline{\varphi}_i$ and $\underline{\varphi}_i$ are rarely significant, i.e., strictly lower than $\sum_{j=i+1}^{n} c_{ij}$ for $\overline{\varphi}_i$ and positive for $\underline{\varphi}_i$.

Table 3 How Important It Is to Order the Variables Correctly and Start with a Good Initial Solution

Data		LIN2-increasing order initial solution of poor quality					LIN2-natural order heuristic (Billionnet and Calmels 1996)				
# Var. (n)	Dens. (%)	(UB − LB)/LB (%)	(UB − opt)/opt (%)	# Nodes	Total time (s)	SI	(UB − LB)/LB (%)	(UB − opt)/opt (%)	# Nodes	Total time (s)	SI
140	5	67.0	2.3	551,809	2,466.66	15	8.7	8.1	1,939,874	1,783.48	20
	25	378.5	7.2	241,441	1,629.31	16	13.5	13.1	2,735,010	5,442.07	13
	50	279.2	4.0	288,840	1,833.23	12	7.9	7.9	1,629,715	8405.99	10
	75	186.5	4.2	443,978	3,033.88	18	6.7	6.6	1,308,564	4,819.50	10
	100	128.8	2.5	352,081	1,873.56	19	3.7	3.5	1,359,977	8,208.87	15

Data		LIN2-decreasing order heuristic (Billionnet and Calmels 1996)					LIN2-increasing order heuristic (Billionnet and Calmels 1996)				
# Var. (n)	Dens. (%)	(UB − LB)/LB (%)	(UB − opt)/opt (%)	# Nodes	Total time (s)	SI	(UB − LB)/LB (%)	(UB − opt)/opt (%)	# Nodes	Total time (s)	SI
140	5	7.0	6.4	95,852	539.73	20	2.2	1.9	454,043	1,992.41	15
	25	8.0	7.6	569,151	4,256.64	17	6.3	5.9	672,920	3,959.34	18
	50	4.9	4.7	226,537	1,554.19	13	3.3	3.1	292,461	1,636.62	16
	75	3.8	3.6	435,320	2,896.35	20	3.0	2.9	560,925	3,758.89	20
	100	1.4	1.2	433,482	2,214.56	19	1.5	1.3	77,035	882.71	19

Notes. Dens. (%), density (proportion of non-zero linear and quadratic coefficients in the objective function); (UB − LB)/LB (%), average relative gap between the lower bound (LB) and the upper bound (UB); (UB − opt)/opt (%), average relative gap between the upper bound (UB) and the optimum (opt); SI, number of exactly solved instances among the 20.

5.1.2. Comparison to Other Methods. In Table 4 we present a comparison between the running times and the number of nodes in the search tree that are required for solving QKP using the methods presented in Billionnet et al. (1997), Hammer and Rader (1997), Caprara et al. (1999), and those required by our own approach. The running times required by Hammer and Rader's algorithm, on a Sparc Server 1000 with 2 processors, in order to solve 100-variable instances, range from 133 to 1,428 seconds and are comparable to ours. The methods proposed in Caprara et al. (1999) and Billionnet et al. (1997) enable us to solve larger instances, up to 400 variables for high-density problems compared to Caprara et al.

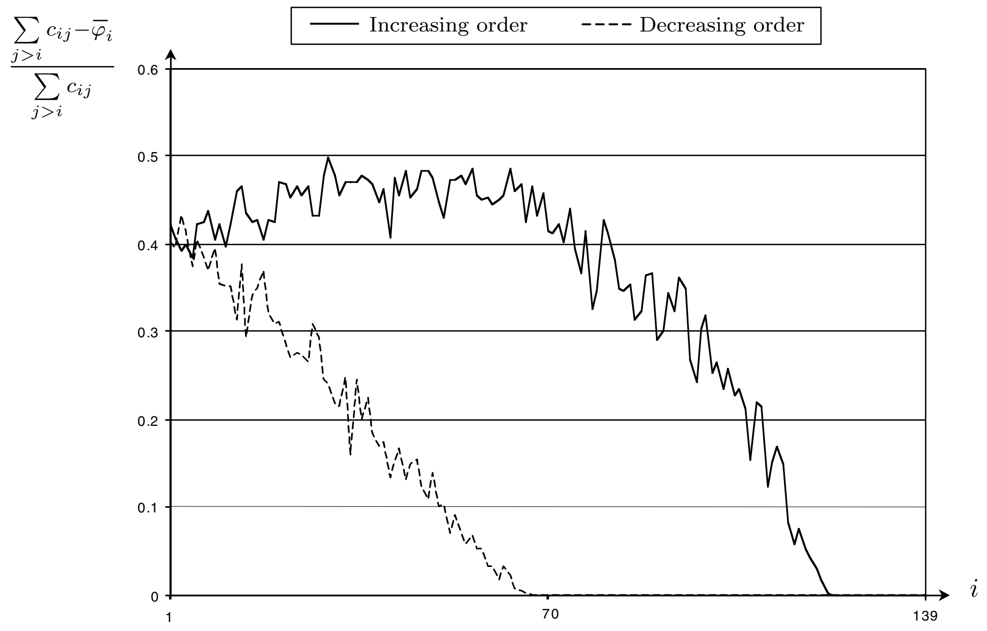

Figure 1 Measure of the Significance of the Coefficients $\overline{\varphi}_i$ When Using Increasing and Decreasing Order During the Preprocessing
Note. Instance of 140 variables, density 100%.

Table 4 **Numerical Comparison Between Four Methods for Solving QKP**

Data		Hammer and Rader (1997) Sparc server 1000 with 2 processors		Caprara et al. (1999) HP9000/735		Billionnet et al. (1997) Pentium II 300 MHz		Our approach: Best of CLIN or LIN2 Pentium II 300 MHz	
# Var. (*n*)	Density (%)	CPU time (s)	# Nodes B&B	CPU time (s)	# Nodes B&B	CPU time (s)	# Nodes B&B	CPU time (s)	# Nodes B&B
100	25	133	26	217	1,605,951	117	77	374	115,427
	50	169	60	29	149,888	82	26	92	22,047
	75	469	204	7	1,312	120	112	91	28,574
	100	1,428	159	6	656	190	272	28	3,796
120	25			20	9,191	240	50	1,708	4,949
	50			17	28,172	329	185	1,250	237,234
	75			8	1,548	234	143	624	119,781
	100			11	31,133	1,321	3,099	191	47,823
140	25					645	330	2,171	3,508
	50			285	1,354,323	315	54	1,637	292,461
	75			20	13,097	320	116	3,759	560,925
	100			24	44,208	3,688	3,278	882	77,035

(1999) and up to 300 variables for low- and medium-density problems compared to Billionnet et al. (1997). Their running times are comparable to LIN1 and LIN2 for instances with up to 140 variables. However, the non-classical Lagrangian methods that they carry out are particularly elaborate and difficult to implement. On the contrary, the main interest of our approach is its ease of implementation. Indeed, solving LIN1 and LIN2 is a straightforward task with mixed integer programming software.

5.2. Application of the Method to Designing Compilers

In this section we study the behavior of LIN2 when this linearization is used to solve the compiler-design problems presented in Helmberg et al. (1996). Using LIN2 we compared the results to those obtained by Helmberg et al. (1996) and by Caprara et al. (1999). The results in Table 5 show that the upper bound obtained by our approach is much less accurate than that obtained in Helmberg et al. (1996) and also less

Table 5 **Numerical Comparison Between CLIN, LIN2, and Two Others Methods Concerning 12 Compiler Design Instances**

Data				Helmberg et al. (1996) Sun Sparcstation 10			Caprara et al. (1999) HP9000/735				(CLIN) Pentium II 300 MHz				(LIN2) Pentium II 300 MHz			
# var (*n*)	D. (%)	RHS	Opt.	Feas. sol.	Rel. gap (%)	T. UB (s)	Rel. gap (%)	T. UB (s)	# Nodes	Total time (s)	Rel. gap (%)	T. UB (s)	# Nodes	Total time (s)	Rel. gap (%)	T. UB (s)	# Nodes	Total time (s)
30	12.0	450	1,580	1,580	0	24	11.15	0.45	15	0.45	63	0.0	65	0.4	52	0.0	100	0.4
	12.0	512	1,802	1,802	0	1,570	6.04	0.65	1	0.65	31	0.0	93	0.5	22	0.0	66	0.3
	12.0	600	2,326	2,326	0	91	0.00	0.39	1	0.39	31	0.0	204	1.1	29	0.0	111	0.4
45	9.5	450	2,840	2,840	0	823	3.21	0.77	17	0.77	10	0.1	304	3.9	14	0.0	100	0.9
	9.5	512	3,154	3,154	1.58	Unknown	5.57	0.93	83	0.93	19	0.1	171	2.2	23	0.0	253	1.0
	9.5	600	3,840	3,840	0	182	0.28	0.78	1	0.78	17	0.1	115	1.6	22	0.0	240	1.0
47	8.8	450	1,732	1,732	0	1,870	3.64	0.77	1	0.77	18	0.1	286	4.3	14	0.0	70	1.1
	8.8	512	1,932	1,932	0	500	1.73	1.01	1	1.01	29	0.1	379	5.3	26	0.0	649	3.2
	8.8	600	2,186	2,186	4.09	Unknown	13.91	1.88	46	1.88	23	0.0	310	4.6	24	0.0	4,322	14.8
61	9.9	450	26,996	26,996	0	203	0.97	1.30	1	1.30	4	0.1	22	1.4	5	0.0	42	2.3
	9.9	512	29,492	29,492	0.02	Unknown	1.69	1.24	3	1.24	2	0.1	75	4.1	3	0.0	34	2.3
	9.9	600	32,552	32,552	0.33	Unknown	1.54	1.38	29	1.38	2	0.1	64	3.6	3	0.0	57	2.4

Notes. D. (%), density (proportion of non-zero linear and quadratic coefficients in the objective function); RHS, right-hand-side value of the instance; Opt. optimum value of the instance; feas. sol., value of the best feasible solution found by the used method; rel. gap (%), relative gap between the lower bound (LB), i.e., the value of an accurate feasible solution and the upper bound (UB): relative gap = (UB − LB)/LB; T. UB (s), CPU time (in seconds) required by the computation of the upper bound (evaluation at the root of the branching tree); # nodes, number of nodes investigated during the branch-and-bound algorithm; total time (s), CPU time (in seconds) corresponding to the exact integer solution.

accurate than that obtained in Caprara et al. (1999). However, the formulation LIN2, and to a lesser extent the formulation CLIN, make it possible to solve these structured instances quickly. With regard to LIN2, the weakness of the bound is compensated for by the number of variables ($O(n)$) involved in the formulation. With regard to CLIN, we previously noted that this formulation was well adapted to low-density problems, which is the case in these instances.

6. Conclusion

In this paper we have presented methods for solving the 0–1 quadratic knapsack problem by using mixed integer programming software. The linearizations we have proposed differ from the classical one in their number of variables and constraints: $O(n)$ for our own linearizations and $O(n^2)$ for the classical one. The classical linearization and both of our linearizations present three main advantages. First, their implementation is particularly easy with mixed integer programming software. On the contrary, the best methods available are very elaborate and difficult to implement without the software the authors developed themselves. Moreover, mixed integer formulations enable us to benefit from the reliability of mixed integer programming software. Second, this approach by linearization is competitive (in terms of size of solved instances) with the best methods. Third, the linearization approach enables us easily to add supplementary linear constraints to the original quadratic knapsack problem. Finally, the linearization technique we have offered may be applied to several other classical quadratic 0–1 programs, such as the task-assignment problem, which is a particular case of the semi-assignment problem (Billionnet and Elloumi 2001).

References

Beale E. M. L. 1988. *Introduction to Optimization*. Wiley, New York.

Billionnet, A., F. Calmels. 1996. Linear programming for the 0–1 quadratic knapsack problem. *Eur. J. Oper. Res.* **92** 310–325.

Billionnet, A., S. Elloumi. 2001. Best reduction of the quadratic semi-assignment problem. *Discrete Appl. Math.* **109** 197–213.

Billionnet, A., A. Faye, E. Soutif. 1997. An exact algorithm for the 0–1 quadratic knapsack problem. *ISMP97, Lausanne, Switzerland, August 1997*.

Billionnet, A., A. Faye, E. Soutif. 1999. A new upper bound for the 0–1 quadratic knapsack problem. *Eur. J. Oper. Res.* **112** 664–672.

Caprara, A., D. Pisinger, P. Toth. 1999. Exact solution of the quadratic knapsack problem. *INFORMS J. Comput.* **11** 125–137.

Chaillou, P., P. Hansen, Y. Mahieu. 1986. Best network flow bound for the quadratic knapsack problem. *Lecture Notes in Mathematics*, No. 1403, 226–235.

Ilog, Inc., CPLEX Division. 1998. Using the CPLEX callable library, Version 6.0. Incline Village, NV, USA.

Gallo, G., P. L. Hammer, B. Simeone. 1980. Quadratic knapsack problems. *Math. Programming Stud.* **12** 132–149.

Glover, F. 1975. Improved linear integer programming formulations of nonlinear integer problems. *Management Sci.* **22** 455–460.

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

Helmberg, C., F. Rendl, R. Weismantel. 1996. Quadratic knapsack relaxation using cutting planes and semidefinite programming. W. H. Cunningham, S. T. McCormick, M. Queyranne, eds. *Proc. Fifth IPCO Conf., Lecture Notes Comput. Sci.*, No. 1084. Springer Verlag, 175–189.

Michelon, P., L. Veuilleux. 1996. Lagrangean methods for the 0–1 quadratic knapsack problem. *Eur. J. Oper. Res.* **92** 326–341.

Salkin, H. M., K. Mathur. 1989. *Foundations of Integer Programming*. North-Holland, Amsterdam, The Netherlands.