Constraint Programming and Linear Programming for Quadratic 0-1 Problems

Serigne Gueye, Philippe Michelon

Constraints Programming and Linear Programming for Quadratic 0-1 Problems

S.Gueye (*) and P.Michelon(**)

(*) Université du Havre,
Laboratoire de Mathématiques Appliquées du Havre,
25 rue Philippe Lebon, BP 540, 76058 Le Havre cedex.
email : serigne.gueye@univ-lehavre.fr

(**) Université d'Avignon,
Laboratoire d'Informatique d'Avignon,
UPRES 931, 339 chemin des Meinajariès,
Agroparc BP 1228, 84911, Avignon cedex 9, France.
email : philippe.michelon@lia.univ-avignon.fr

January 21, 2007

Abstract

In order to solve more easily combinatorial optimization problems, one way is to find theoretically a set of necessary or/and sufficient conditions that the optimal solution of the problem has to verify. For instance, in linear programming, weak and strong duality conditions can be easily derived. And in convex quadratic programming, Karush-Kuhn-Tucker conditions gives necessary and sufficient conditions. Despite that, in general, such conditions doesn't exist for integer programming, some necessary conditions can be derived from Karush-Kuhn-Tucker conditions for the unconstrained quadratic 0-1 problem. In this paper, we present these conditions. We show how they can be used with constraints programming techniques to fix directly some variables of the problem. Hence, reducing strongly the size of the problem. For example, for low density problems of size lower than 50, those conditions combined with constraints programming may be sufficient to solve completely the problem, without branching. In general, for quadratic 0-1 problems with linear constraints, we propose a new method combining these conditions with constraints and linear programming. Some numerical results, with the instances of the OR-Library, will be presented.

1 Introduction

We are interested in this paper to solve exactly unconstrained quadratic 0-1 problems. The mathematical form of the problem is the following

$$\begin{array}{lll} \text{(QP)} & \text{Min} & < c, x > + x^\top Q x \\ & & \\ & \text{s-t :} & (1)\ x \in \{0,1\}^n \end{array}$$

where $c \in \mathbb{R}^n$, Q is a real $n \times n$ upper triangular matrix.

This problem involves (with additional constraints) many academical problems as the graph bipartitoning problem, the quadratic assignment problem, the max-cut problem or the maximum independent set problem. All of these problems are known to be NP-hard (Garey and Johnson [12]) and modelize real-world applications in computer sciences, transports and logistics.

The difficulty of quadratic problems is mainly due to the non-linearity of the function and the integrity of the variables. That's why most of the techniques proposed to solve it consist first to break the non-linearity of the functions and to relax the integrity constraints. For instance, it can be found in the litterature linearizations of the objective function [2] [7] [13], semidefinite relaxations [17] [18] [23], and lagrangean relaxations [6] [20]. For all of these techniques, the size of the instances that be can be solved exactly in a reasonable amount of time is limited : no more than 150 variables for all densities for the unsconstrained quadratic 0-1 problem, and no more than 32 variables for the quadratic assignment problems. It must be noticed that low density problems are easier to solve that high density ones.
Since the resolution techniques are based on relaxations of the problem, to improve the numerical results we have to strengthen the relaxation by adding some linear cuts (ideally some facets) in the case of linear relaxation, or some semidefinite cuts in the case of semidefinite relaxation [17] [18]. Another way is to try first to reduce the size of the realisable domain of the problem, by characterizing theoretically the optimal solution of the problem. In integer linear programming, this phase is usually known as the preprocessing phase. It allows to fix some variables and add some new constraints. This is precisely one purpose of this paper.

Our contribution may be viewed as an extension of the first result below of P. Hansen [16]. It allows to fix some variables under certain conditions.

Lemma 1.1 . *Let us notice* $q_{ij}^- = Min\{0, q_{ij}\}$ *and* $q_{ij}^+ = Max\{0, q_{ij}\}$ *(*$1 \leq i < j \leq n$*).*

- *If* $c_i + \sum_{j=1}^{i-1} q_{ij}^- + \sum_{j=1}^{i-1} q_{ij}^- < 0$ *then* $x_i = 0$

- *If* $c_i + \sum_{j=1}^{i-1} q_{ij}^+ + \sum_{j=1}^{i-1} q_{ij}^+ < 0$ *then* $x_i = 1$

Depending on the structure of the problem, these rules may be succesfull to fix some variables. This is the case for the quadratic problem benchmarks for which the values of c_i and q_{ij} are chosen randomly between $[-100, 100]$. At the opposed, in the max-cut problem the lemma has absolutely no effect because of the structure of the objective function.

We have observed and proved that, even when we cannot fix a variable, it is always possible to fix products of any number of variables as $x_i x_j = 0$ or $x_i x_j x_k = 0$, or $(1 - x_i) x_j x_k x_l = 0$, etc. We call these equalities **fixations**.

To present the fixations, we need to introduce some notations and definitions. If x_i $(i = 1, 2, .., n)$ denote a boolean variable, the complement of x_i (i.e $1 - x_i$) will be noticed by $\overline{x}_i$.

Definition 1.2 *A litteral is a boolean variable of the set* $L = \{x_1, x_2, ..., x_n, \overline{x}_1, , .., \overline{x}_n\}$.

This is a definition from the theory of posiforms (see [4]). Hence, a fixation is in fact a product of litterals. For instance the third one above corresponds also to $\overline{x}_i x_j x_k x_l = 0$.

Definition 1.3 The cardinality *of a fixation is the number of litterals of the fixation.*

It can be noticed that lower is the cardinality and better is the fixation for the simplification of the problem. For instance, $x_i = 0$ and $x_i x_j = 0$ are two fixations of cardinality 1 and 2. With the first one, the number of variables of the problem is decreased by one, but with the second we can only eliminate a quadratic term : $x_i = 0$ is more interesting than $x_i x_j = 0$.

The interests of the fixations are the following :

- Reducing the density of the matrix Q : for instance the fixation $x_i x_j = 0$ eliminates the term $q_{ij} x_i x_j$ hence, simplifying the resolution of the problem.

- Reducing the realisable domain : for instance, the fixation $x_i x_j x_k = 0$ reduce the realisable domain of the problem.

Moreover, in all cases, these fixations can be linearized as $x_i + x_j \leq 1$, $x_i + x_j + x_k \leq 2$ and introduced in a linear or a semidefinite relaxation. Therefore, they can be viewed as new cuts strengthening the relaxations.

These fixations are derived from necessary optimal conditions for the unconstrained quadratic 0-1 problem. We present these conditions in section 2 and show how numerous fixations can be generated from these conditions.

To find in practice these new fixations, we have to solve a series of minimal cover problems. The link between minimal cover problems and our fixations are shown in section 3 and a heuristic is proposed to solve it in the same section. The iterations of the heuristics produce a series of fixations, for which it can be viewed that some deductions can produced (in certain cases) other fixations with lower cardinalities. These deductions are obtained by using constraint programming techniques, specially propagation schemes. We discuss this aspect in section 4.

By combining the heuristics and constraint programming, we propose in the same section a preprocessing algorithm for unconstrained quadratic 0-1 problems and some numerical tests in section 5. The algorithm allows, for most cases, to solve completely the problem (i.e to fix all variables) for low density problem of size lower than 50.

For more difficult problem, we propose to combine the algorithm with linear or semidefinite relaxations. In section 6, this approach is presented with a linear relaxation and is experimented in section 7 .

Some perspectives of this framework are discussed in section 8.

2 Fixations for the unconstrained quadratic 0-1 problem

It is well known (see Rosenberg [24]) that solving the problem (QP)

$$\text{(QP)} \quad \text{Min} \quad < c, x > + \tfrac{1}{2} x^\top Q x$$

$$(1)\ x \in \{0,1\}^n$$

is equivalent to solve its continuous relaxation $(\overline{QP})$

$$(\overline{QP}) \quad \text{Min} \quad < c, x > + x^\top Q x$$

$$(2)\ x \in [0,1]^n$$

In other words, there exists at least one integer point $x \in \{0,1\}^n$ for which the value of the objective function corresponds to the optimal value of the problem $(\overline{QP})$. Given that x is optimal for $(\overline{QP})$, x has to verify Karush-Kuhn-Tucker conditions. By expressing these conditions, the following necessary optimality conditions may be deduced for (QP).

Proposition 2.1 *Let x be the optimal solution of $(\overline{QP})$. The following two inequalities are necessary verified.*

- $c_i x_i + \sum\limits_{\substack{j=1 \\ j\neq i}}^{n} q_{ij} x_i x_j \leq 0$, $i = 1, 2, ..., n$.
- $c_i(1 - x_i) + \sum\limits_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}(1 - x_i)x_j \geq 0$, $i = 1, 2, ..., n$.

Proof Let us first consider a more explicit formulation of $(\overline{QP})$

$$\text{(QP)} \quad \text{Min} \quad \sum_{i=1}^{n} c_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} q_{ij} x_i x_j$$

$$(2)\ 0 \leq x_i \leq 1 \qquad\qquad i = 1, 2, ..., n$$

Let μ_i (resp. α_i) be the lagrangean multipliers associated to the constraints $x_i \leq 1$ (resp. $-x_i \leq 0$).

We know that the Karush-Kuhn-Tucker conditions applied to the problem yield the following equalities and inequalities :

1. $c_i + \sum\limits_{\substack{j=1 \\ j\neq i}}^{n} q_{ij} x_j + \mu_i - \alpha_i = 0$, $1 \leq i \leq n$
2. $\mu_i(1 - x_i) = 0$, $1 \leq i \leq n$
3. $\alpha_i x_i = 0$, $1 \leq i \leq n$
4. $\mu_i \geq 0$, $\alpha_i \geq 0$, $1 \leq i \leq n$
5. $x_i \geq 0$, $x_i \leq 1$, $1 \leq i \leq n$

By multiplying with x_i the equality (1), we obtain :

$$c_i x_i + \sum_{\substack{j=1 \\ j \neq i}}^{n} q_{ij} x_i x_j + \mu_i x_i - \alpha_i x_i = 0$$

$$\Rightarrow \quad c_i x_i + \sum_{\substack{j=1 \\ j \neq i}}^{n} q_{ij} x_i x_j + \mu_i x_i = 0, \text{ because of the relation (3)}$$

$$\Rightarrow \quad c_i x_i + \sum_{\substack{j=1 \\ j \neq i}}^{n} q_{ij} x_i x_j = -\mu_i x_i,$$

$$\Rightarrow \quad c_i x_i + \sum_{\substack{j=1 \\ j \neq i}}^{n} q_{ij} x_i x_j \leq 0, \text{ because of the relation (4)}$$

The first inequality of the theorem is verified.

By multiplying the relation (1) by $(1 - x_i)$ and perform the same deduction, the second inequality of the theorem can be proved.

□

Using a result of E. Boros and P.L. Hammer [5], the same theorem can be obtained more easily. In this paper, the authors consider more general function called pseudo-boolean functions for which a representation is

$$f(x_1, x_2, ..., x_n) = \sum_{S \subseteq \{1,2,...,n\}} q_S \prod_{j \in S} x_j$$

Notice that quadratic boolean functions in problem (QP) are particular cases of pseudo-boolean functions, since they can be obtained by taking subset S of cardinality 2 and $q_{ii} = c_i$ $(1 \leq i \leq n)$. Now, let us consider the following quantities

$$\Delta_i(x) = \sum_{S \subseteq \{1,2,...,n\} \setminus i} q_{S \cup \{i\}} \prod_{j \in S} x_j, \; i = 1, 2, ..., n$$

With these notations, E.Boros and P.L. hammer prove the following result.

Proposition 2.2 *Given the pseudo-boolean function f, a binary vector $x \in \{0,1\}^n$ is a local minimzer of f if and only if*

$$x_i = \begin{cases} 1 & if \; \Delta_i(x) < 0 \\ \\ 0 & if \; \Delta_i(x) > 0 \end{cases}, \quad i = 1, 2, ..., n$$

Applying this proposition to the problem (QP) gives our first proposition.

Remark 2.3 *It is intersting to remark that Hansen result in the introduction can be obtained with the proposition 2.1.*

Indeed, we know that

$$c_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}^{-} \leq c_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}x_j \text{ where } q_{ij}^{-} = Min\{0, q_{ij}\}.$$

Since $x_i[c_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}x_j] \leq 0$ *and* $x_i \geq 0$ *then*

$x_i[c_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}^{-}] \leq 0$ *is necessary verifiy by the optimal solution.*

Therefore, if $c_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}^{-} > 0$ *then* $x_i = 0$*, and the first result of Hansen is obtained.*

Just like above, since $c_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}^{+} \geq c_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}x_j$ *where* $q_{ij}^{+} = Max\{0, q_{ij}\}$*, we have*

$$(1 - x_i)[c_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}^{+}] \geq 0.$$

Thus, if $c_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}^{+} < 0$ *then* $x_i = 1$.

□

It can be also observed that each inequality of the proposition 2.1 is a deduction obtained by multiplying the expression $c_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}x_j$ by x_i or $1 - x_i$.

Since the mutiplication by one variable yields the hansen fixation rules for that variable, it comes that the multiplication by a product of 2 variables will produced another rule for this product. The proposition below is based on this idea.

Proposition 2.4 *Let* i *and* $k \in \{1, 2, ..., n\}$ *with* $i \neq k$*, and* x *the optimal solution of the problem (UQP).*

(i) • *If* $c_i + q_{ik} + \sum_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij}^{-} x_j > 0$ *then* $x_i x_k = 0.$

(ii) • *If* $c_i + q_{ik} + \sum_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij}^{+} x_j < 0$ *then* $(1 - x_i)x_k = 0.$

(iii) • *If* $c_i + \sum_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij}^{-} x_j > 0$ *then* $x_i(1 - x_k) = 0.$

(iv) • *If* $c_i + \sum_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij}^{+} x_j < 0$ *then* $(1 - x_i)(1 - x_k) = 0.$

Proof We know that $c_i x_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij} x_i x_j \leq 0.$

By multiplying with x_k $(i \neq k)$ we obtain $c_i x_i x_k + q_{ik} x_i x_k + \sum_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij} x_i x_j x_k \leq 0.$

Thus $x_i x_k [c_i + q_{ik} + \sum_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij} x_j] \leq 0.$

Since $c_i + q_{ik} + \sum_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij}^{-} x_j \leq c_i + q_{ik} + \sum_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij} x_j \Rightarrow x_i x_k [c_i + q_{ik} + \sum_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij}^{-} x_j] \leq 0.$

Thus, if $c_i + q_{ik} + \sum_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij}^{-} x_j > 0$ then $x_i x_k = 0$. The point (i) is obtained.

By analogy, we also know that $c_i(1 - x_i) + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}(1 - x_i) x_j \geq 0.$

By multiplying with x_k $(i \neq k)$ we obtain $x_k(1 - x_i)[c_i + q_{ik} + \sum_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij}^{+} x_j] \geq 0.$

Therefore, if $c_i + q_{ik} + \sum_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij}^{+} x_j < 0$ then $(1 - x_i)x_k = 0$. And the point (ii) is obtained.

The last two results are deduced in the same way. □

The results of the propostion 2.4 are called fixations of cardinality 2. They allow to discard or to linearize quadratic terms. Hence, to decrease the density of the

matrix Q. Indeed, the equality $x_i x_k = 0$, for instance, can be used to eliminate the term $q_{ik} x_i x_k$, but corresponds also to the valid inequality $x_i + x_k \leq 1$. This cut can be introduced in any relaxation (linearization, semidefinite relaxation) of the problem (QP).

As we derive fixations for product of 2 variables, many others fixations of all cardinalities can be deduced. The proposition below is a generalization of the propostion 2.4.

Proposition 2.5 *Let $i \in \{1, 2..., n\}$, $S \subseteq \{1, 2..., n\}$ and y_j $(j \in S)$ some litterals. Let us consider the following three sets :*

$S^1 = \{j \in S | y_j = x_j\}$ $S^0 = \{j \in S | y_j = \overline{x}_j\}$ $\overline{S} = \{1, 2..., n\} \backslash S$

- *If* $c_i + \sum_{j \in S^1} q_{ij} + \sum_{j \in \overline{S}} q_{ij}^- > 0$ *then* $x_i \prod_{j \in S} y_j = 0$
- *If* $c_i + \sum_{j \in S^1} q_{ij} + \sum_{j \in \overline{S}} q_{ij}^+ < 0$ *then* $(1 - x_i) \prod_{j \in S} y_j = 0$

Proof Let us assume that variables y_j $(j \in S^1 \cup S^0$ have been fixed to 1 (or equivalently that x_j for $j \in S^1$, have been fixed to 1 while x_j for $j \in S^0$ have been fixed to 0). We can then apply Lemma 1.1 to the resulting reduced problem. We then have

$$c_i + \sum_{j \in S^1} q_{ij} + \sum_{j \in \overline{S}} q_{ij}^- \geq 0 \Rightarrow x_i = 0$$

Therefore, if the left hand side of the above implication is verified, it is impossible to have

$$y_j = 1 \ \forall j \in S^1 \cup S_0, \text{ and } x_i = 1$$

in the initial problem. Thus $x_i \prod_{j \in S^1 \cup S^0} y_j = x_i \prod_{j \in S} y_j = 0$.

The second point is obtained like above.

□

Remark 2.6 .

- *When $|S| = 1$, we obtain exactly the results of proposition 2.4.*

- *When $|S| > 1$, we cannot fix or eliminate quadratic terms, but new valid*

inequalities can be introduced.

Indeed, the fixation $x_i \prod_{j \in S} y_j = 0$ *is equivalent to the linear inequality*

$$x_i + \sum_{j \in S} y_j \leq |S|,$$

and the fixation $(1 - x_i) \prod_{j \in S} y_j = 0$ *is equivalent to*

$$(1 - x_i) + \sum_{j \in S} y_j \leq |S|.$$

□

In order to illustrate how these results can help to solve (QP), we present the following example.

Example 2.7 *Let us consider the following uncosntrained quadratic 0-1 problem of* 7 *variables*

$$\begin{array}{lllllll}
Min & 85x_1 & +100x_2 & -88x_3 & -16x_4 & -37x_5 & +14x_6 & +80x_7 \\
 & & +42x_1x_2 & -68x_1x_3 & -6x_1x_4 & -10x_1x_5 & -23x_1x_6 & -87x_1x_7 \\
 & & & +60x_2x_3 & -76x_2x_4 & -58x_2x_5 & +82x_2x_6 & +85x_2x_7 \\
 & & & & & +19x_3x_5 & -10x_3x_6 & +60x_3x_7 \\
 & & & & & +25x_4x_5 & +98x_4x_6 & -2x_4x_7 \\
 & & & & & & +70x_5x_6 & -57x_5x_7 \\
 & & & & & & & -57x_5x_7 \\
s-t & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \in \{0,1\}
\end{array}$$

The optimal value of this problem is -109 *corresponding to the optimal solution* $x = [1, 0, 1, 0, 1, 1, 1]$.

Iteration 1

Let us first observe that the proposition 1.1 of Hansen is inactive for this problem. No variables can be fixed to 0 *or* 1.

Neverthelees, by applying our proposition 2.1, we deduce that the optimal solution has to verify :

For $i = 2$ *and* $k = 3$*, we have, with point (i),*
$c_i + q_{ik} + \sum_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij}^{-} = 100 + 60 - 76 - 58 = 26 > 0 \Rightarrow x_2 x_3 = 0$ *(a).*

For $i = 3$ *and* $k = 2$*, we have, with point (iv),*
$c_i + \sum_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij}^{+} = -88 + 19 + 60 = -9 < 0 \Rightarrow (1 - x_3)(1 - x_2) = 0$ *(b).*

For $i = 3$ *and* $k = 7$*, we have, with point (iv),*
$c_i + \sum_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij}^{+} = -88 + 19 + 60 = -9 < 0 \Rightarrow (1 - x_3)(1 - x_7) = 0$ *(c).*

For $i = 2$ *and* $k = 7$*, we have, with point (iv),*
$c_i + q_{ik} + \sum_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij}^{-} = 100 + 85 - 76 - 58 = 51 > 0 \Rightarrow x_2 x_7 = 0$ *(d).*

Thus, quadratics terms x_2x_3*,* x_3x_7*,* x_2x_7 *can be eliminated from the objective function of the problem.*

However, it is more interesting to observe that the four fixations above imply that $x_2 = 0$ *and* $x_3 = 1$.

Indeed, fixations (a) and (b) give $x_2 + x_3 = 1$*, thus* $x_2 = 1 - x_3$*. And, because of fixations (c) and (d), we can deduced that* $x_2 = 0$ *and hence* $x_3 = 1$*. Variables* x_2 *and* x_3 *can then be eliminated to the problem. It follows the new update problem.*

$$
\begin{array}{llllllll}
Min & -88 & +17x_1 & & -16x_4 & -18x_5 & +4x_6 & +140x_7 \\
 & & & & -6x_1x_4 & -10x_1x_5 & -23x_1x_6 & -87x_1x_7 \\
 & & & & & +25x_4x_5 & +98x_4x_6 & -2x_4x_7 \\
 & & & & & & +70x_5x_6 & -57x_5x_7 \\
 & & & & & & & -57x_5x_7 \\
s-t & x_1 & & x_4 & x_5 & x_6 & x_7 \in \{0,1\} & \\
\end{array}
$$

Iteration 2

Once again, no fixation rules of Hansen proposition 1.1 are verified. But the following fixations of cardinality 2 *and* 3 *can be deduced :*

For $i = 7$ *and* $k = 1$*, we have, with point (iii),*
$c_i + \sum_{\substack{j=1 \\ j \neq i,k}}^{n} q_{ij}^- = 140 - 2 - 57 - 57 = 24 > 0 \Rightarrow x_7(1 - x_1) = 0$ *(e).*

For $i = 7$*,* $S = S^1 = \{1, 6\}$*, we have, with proposition 2.5,*
$c_i + \sum_{j \in S^1} q_{ij} + \sum_{j \in \overline{S}} q_{ij}^+ = 140 - 87 - 57 + 0 = -4 < 0 \Rightarrow (1 - x_7)x_1x_6 = 0$ *(f).*

For $i = 1$*,* $S = S^0 = \{6, 7\}$*, we have, with proposition 2.5,*
$c_i + \sum_{j \in \overline{S}} q_{ij}^- = 17 - 6 - 10 = 1 > 0 \Rightarrow x_1(1 - x_6)(1 - x_7) = 0$ *(g).*

As in the first iteration above, it may be seen analytically that fixations (e), (f) and (g) give $x_1 = x_7$*.*

Thus one of the variable $x_1 = x_7$*, for instance* x_7*, can be eliminated. We obtain the new problem*

$$
\begin{array}{llllllll}
Min & -88 & +70x_1 & & -16x_4 & -18x_5 & +4x_6 \\
 & & & & -8x_1x_4 & -67x_1x_5 & -80x_1x_6 \\
 & & & & & +25x_4x_5 & +98x_4x_6 \\
 & & & & & & +70x_5x_6 \\
s-t & x_1 & & x_4 & x_5 & x_6 & \in \{0,1\}
\end{array}
$$

Iteration 3

Proposition 1.1 is still inactive. Applying propostion 2.4 gives :

For $i = 1$ *and* $k = 6$*, we have, with point (ii),*
$c_i + q_{ik} + \sum_{\substack{j=1 \\ j \neq i,k}}^{n} q_{ij}^+ = 70 - 80 + 0 = -10 < 0 \Rightarrow (1 - x_1)x_6 = 0$ *(h).*

For $i = 6$ *and* $k = 1$*, we have, with point (iii),*
$c_i + \sum_{\substack{j=1 \\ j \neq i,k}}^{n} q_{ij}^- = 4 - 0 = 4 > 0 \Rightarrow x_6(1 - x_1) = 0$ *(i).*

Thus, (h) and (i) yields $x_1 = x_6$. And the update problem is

$$\begin{array}{llllll} Min & -88 & -6x_1 & -16x_4 & -18x_5 & \\ & & & +90x_1x_4 & +3x_1x_5 & \\ & & & & +25x_4x_5 & \\ s-t & x_1 & x_4 & x_5 & & \in \{0,1\} \end{array}$$

Iteration 4

Applying propostion 2.4 in this new problem gives :

For $i = 1$ and $k = 4$, we have, with point (i),
$c_i + q_{ik} + \sum\limits_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij}^{-} = -6 + 90 = 84 > 0 \Rightarrow x_1x_4 = 0$ *(j).*

For $i = 1$ and $k = 4$, we have, with point (iv),
$c_i + \sum\limits_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij}^{+} = -6 + 3 = -3 < 0 \Rightarrow (1 - x_1)(1 - x_4) = 0$ *(k).*

For $i = 4$ and $k = 5$, we have, with point (i),
$c_i + q_{ik} + \sum\limits_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij}^{-} = -16 + 25 + 0 = 9 > 0 \Rightarrow x_4x_5 = 0$ *(l).*

For $i = 5$ and $k = 4$, we have, with point (iv),
$c_i + \sum\limits_{\substack{j=1 \\ j\neq i,k}}^{n} q_{ij}^{+} = -18 + 3 = 15 < 0 \Rightarrow (1 - x_5)(1 - x_4) = 0$ *(m).*

With (j), (k), (l) and (m), it can be seen analytically that $x_1 = x_5$ and $x_4 = 1 - x_1$. The update problem is

$$\begin{array}{llll} Min & -104 & -5x_1 & \\ s-t & x_1 & & \in \{0,1\} \end{array}$$

The solution of this last problem is trivial : $x_1 = 1$.

Therefore, by using in reverse order the chain of deduction found at each iteration, the optimal solution can be deduced as below :

- Iteration 4 : $x_1 = x_5 = 1$, $x_4 = 1 - x_1 = 0$,

- *Iteration 3 :* $x_1 = x_6 = 1$.

- *Iteration 2 :* $x_1 = x_7 = 1$.

- *Iteration 1 :* $x_2 = 0$, $x_3 = 1$.

Thus we find $x = [1, 0, 1, 0, 1, 1, 1]$ *corresponding to the optimal solution.*
□

The problem has been completely solve exactly by applying two strategies : application of proposition 2.5 (generalizing proposition 2.4 and proposition 1.1) and performing some deductions on the fixations found at each iteration.
In practice, to implement the method, we need procedures for each strategies. In section 3, an heuristic generating some fixations of any order will be presented, and in section 4, we show how constraints programming can be used to make deductions.

3 An heuristic for generating fixations

Let $i \in \{1, 2, ..., n\}$ and let us consider the quantity $c_i x_i + \sum_{\substack{j=1 \\ j \neq i}} q_{ij} x_i x_j$.

For each i, it is always possible by complementing (if necessary) the variables x_j to change the expression in an equivalent one where all q_{ij} are negative.

This is done by the following operations.

Let $V = \{1, 2, ..., n\} \backslash \{i\}$, $P = \{j \in V | q_{ij} > 0\}$ and $N = \{j \in V | q_{ij} \leq 0\}$.

Thus $c_i x_i + \sum_{j \in P} q_{ij} x_i x_j + \sum_{j \in N} q_{ij} x_i x_j$.

Since $x_j = 1 - \overline{x}_j$, we have $c_i x_i + \sum_{j \in P} q_{ij} x_i (1 - \overline{x}_j) + \sum_{j \in N} q_{ij} x_i x_j$.

$(c_i + \sum_{j \in P} q_{ij}) x_i - \sum_{j \in P} q_{ij} x_i \overline{x}_j + \sum_{j \in N} q_{ij} x_i x_j$.

So, with the notations $q'_{ij} = \begin{cases} -q_{ij} & j \in P \\ q_{ij} & j \in N \end{cases}$ *and* $y_j = \begin{cases} \overline{x}_j & j \in P \\ x_j & j \in N \end{cases}$

the expression may be written $(c_i + \sum_{j \in P} q_{ij}) x_i + \sum_{j \in V} q'_{ij} x_i y_j$ where all q'_{ij}, are non positive.

Hence for eah i, we will consider that in the expression of the form $c_i x_i + \sum_{\substack{j=1 \\ j\neq i}} q_{ij} x_i y_j$, all value q_{ij} are non positive.

Let us apply Hansen proposition 1.1 or our propositions 2.4 and 2.5 on the expression. The following cases can occured :

Case 1. If $c_i < 0$ Hansen proposition 1.1 imply that $x_i = 1$.

Case 2. If $c_i = 0$, point (ii) of proposition 2.4 imply that $(1 - x_i)y_j = 0$ $(\forall\ j \mid q_{ij} < 0)$.

Case 3. If $c_i > 0$ and :

Case 3.1. $c_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij} > 0$, then Hansen proposition 1.1 gives $x_i = 0$.

Case 3.2. $c_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij} \leq 0$, then proposition 2.5 may be applied as below.

For the case 3.2, since we consider that all q_{ij} are non positive, there are two possibilities for generating some fixations by applying proposition 2.5.

Possibility 1 • Finding sets S for which $S^1 = \emptyset$ and $c_i + \sum_{j\in\overline{S}} q_{ij}^- > 0$.

In this case, we have the fixation $x_i \prod_{j\in S} \overline{y}_j = 0$.

Possibility 2 • Finding sets $S = S^1$ for which $c_i + \sum_{j\in S^1} q_{ij} + \sum_{j\in\overline{S}} q_{ij}^+ < 0$.

In this case, we have the fixation $\overline{x}_i \prod_{j\in S} y_j = 0$.

Notice that in the possibility 2, we have $c_i + \sum_{j\in S^1} q_{ij} + \sum_{j\in\overline{S}} q_{ij}^+ = c_i + \sum_{j\in S^1} q_{ij}$, since we consider that all q_{ij} are non positive.

Finding sets S for each possibility is, in fact, equivalent to find some minimal covers for a knapsack inequality.

Let us consider the possibility 1. Since we consider that each value q_{ij} are non positive, we can write

$$c_i + \sum_{j\in\overline{S}} q_{ij}^- = c_i + \sum_{j\in\overline{S}} q_{ij} = c_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij} - \sum_{j\in S} q_{ij}$$

Let us introduce the binary variable u_j verifiying $u_j = \begin{cases} 1 & j \in S \\ 0 & else \end{cases}$

Thus $c_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij} - \sum_{j\in S} q_{ij} = c_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij} - \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}u_j$.

Finding S verifying $c_i + \sum_{j\in\overline{S}} q_{ij}^{-} > 0$ is therefore equivalent to find u verifying $-\sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}u_j > -c_i - \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}$.

Let us notice $a_j = -q_{ij}$ and $b = -c_i - \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}$.

Thus, we have to find u verifying $\sum_{\substack{j=1 \\ j\neq i}}^{n} a_j u_j > b$.

In the litterature on mixed integer linear programming, specially in knapsack problems, the set defined by u is called a cover. Thus, the set S giving a fixation corresponds to a cover of the knapsack inequality $\sum_{\substack{j=1 \\ j\neq i}}^{n} a_j u_j \leq b$.

This equivalence between fixations and covers gives in the same time a way to identify the "best" fixations. First, let us recall the definition of minimal covers.

Definition 3.1 *Let* $\sum_{j=1}^{n} a_j u_j \leq b$ *be a knapsack inequality.*

$S \subseteq \{1, 2, ..., n\}$ *is a minimal cover if and only if*

$$\sum_{j\in S}^{n} a_j u_j > b \text{ and } \sum_{j\in S'}^{n} a_j u_j \leq b \ \forall \ S' \subseteq S$$

A binary vector verifying $\sum_{\substack{j=1 \\ j\neq i}}^{n} a_j u_j > b$ corresponds to a subset S for which $\sum_{j\in S}^{n} a_j > b$ and for which the fixation $x_i \prod_{j\in S} \overline{y}_j = 0$ can be introduced in the problem.

However, if $S \subseteq S'$, we also have $\sum_{j\in S'}^{n} a_j > b$ and $x_i \prod_{j\in S'} \overline{y}_j = 0$.

$x_i \prod_{j\in S'} \overline{x}_j = 0$ and $x_i \prod_{j\in S} \overline{x}_j = 0$ are redundant. $x_i \prod_{j\in S} \overline{y}_j = 0$ is sufficient.

This observation shows that to avoid redundancy, we have to generate only minimal covers of the knapsack inequality $\sum_{\substack{j=1 \\ j\neq i}}^{n} a_j u_j \leq b$.

For the possibility 2, we obtain a similar equivalence.

This is summarized in the following proposition.

Proposition 3.2 *Let $i \in \{1, 2, ..., n\}$ and let us consider , by complementing some variables if necessary, that all value q_{ij} are non positive.*

Let us notice $a_j = -q_{ij}$, $b_1 = -c_i - \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}$ and $b_2 = c_i$.

- *If S is a minimal cover of the knapsack inequality (1) $\sum_{\substack{j=1 \\ j\neq i}}^{n} a_j u_j \leq b_1$*

  then $x_i \prod_{j\in S} \overline{y}_j = 0$.

- *If S is a minimal cover of the knapsack inequality (2) $\sum_{\substack{j=1 \\ j\neq i}}^{n} a_j u_j \leq b_2$*

  then $(1 - x_i) \prod_{j\in S} y_j = 0$.

To find fixations of any order, we thus have to perform the following procedure.

- **Finding Fixations**

for $i = 1$ to n
{
. *Generate Knapsack Inequality (1)*
. *Find minimal covers for Knapsack Inequality (1)*

. *Generate Knapsack Inequality (2)*
. *Find minimal covers for Knapsack Inequality (2)*
}

There is no difficulty to generate the different knapsack inequalities. It is sufficient to complement the adequate variables and to follow the notations of

proposition 3.2.

To find some covers and hence some fixations, a enumerative procedure has been implemented. This procedure is a recursive enumerative tree for which at each iteration an item a_j is chosen. Since the number of minimal covers may be exponential, it is not reasonable to enumerate all possible covers. That is the reason why, before applying the recursive procedure, we sort the vector a of the knapsack inequality in decreasing order and introduce a parameter to fix the maximal number of covers (i.e the maximal number of iterations of the algorithm). The whole algorithm is the following.

- **Init**

- Sort vector a in decreasing order
- max = maximal number of covers
- count = 0
- i = 1
- S = 0

- **Find minimal covers for Knapsack Inequality**

MinimalCovers(a,b,n,S,count,max,i)
{
. *if(S > b)*
. {
. *STOP*
. *Return the corresponding cover*
. }
. *else*
. *if(count == max)*
. *STOP*
. *else*
. {
. $S = S + a_i$ *(i.e choose item* a_i*)*
. *MinimalCovers(a,b,n,S,count+1,max,i+1)*
. $S = S - a_i$ *(i.e don't choose item* a_i*)*
. *MinimalCovers(a,b,n,S,count+1,max,i+1)*
. }
}

This algorithm can be used to find minimal covers for knapsack inequalities (1) and (2). Application of this procedure yields in fact many redundancies.

Indeed, many fixations from knapsack inequality (1) appear also in fixations from knapsack inequality (2). In order to avoid this effect, we propose another procedure below in which only one type of knapsack inequalities ((1) our (2)) is ocnsidered. The procedure is based on the following idea.

Since the difference between the two knapsack inequalities is only on the value of the right-hand-side b_1 or b_2, it is reasonable to think that most of covers will be found from the knapsack inequalities in which the right-hand-side has the lower value. Hence, a second strategy to generate fixations is the following.

- **Finding Fixations**

for $i = 1$ *to* n
{
. *Generate Knapsack Inequality (1)*
. *Generate Knapsack Inequality (2)*

. *if*($b_1 < b_2$)
. *Find minimal covers for Knapsack Inequality (1)*
. *else*
. *Find minimal covers for Knapsack Inequality (2)*
}

We know how to generate fixations by solving minimal cover problems. We have seen in the examples that some deductions can be obtained from a set of fixations. These deductions are analytically simple, nevertheless finding an algorithm giving the same analytical results is not clear. In the section below, we propose a constraint programming approach for this task.

4 Using constraints programming techniques

Let us consider k subsets $S_i \subseteq \{1, 2, ..., n\}$, and let us denote by F_i $(i = 1, .., k)$ a fixation involving $|S_i|$ litterals y_j $(j \in S_i)$. These fixations are found by "Finding Fixations" above.

Example 4.1 *For instance, in iteration 1 of example 2.7,* $S = \{1, 2, ..., 7\}$ *and we have* 4 *fixations :*

$$\begin{array}{lll} F_1 = & x_2x_3 = 0, & S_1 = \{2, 3\} \\ F_2 = & \overline{x}_2\overline{x}_3 = 0, & S_2 = \{2, 3\} \\ F_3 = & \overline{x}_3\overline{x}_7 = 0, & S_3 = \{3, 7\} \\ F_4 = & x_2x_7 = 0, & S_2 = \{2, 7\} \end{array}$$

□

Let $F = \{F_i, i = 1, 2, ..., k\}$. Our objectif is to derived, iteratively, from F others fixations with lower cardinalities.

Let F_{k+1} be a fixation not in F. To verify that $F_{k+1} = 0$, it suffices to solve the following constraints satisfaction problem

$$\begin{array}{lrl} \text{(CSP)} & F_{k+1} = & 1, \\ & F_1 = & 0, \\ & F_2 = & 0, \\ & . = & . \\ & F_k = & 0, \end{array}$$

If the problem has a solution, nothing can be deduced. However, if it has no solution it can be concluded that $F_{k+1} = 0$. F_{k+1} may thus be added to F, and F has to be updated. This is the main idea of our procedure.

Two questions arise to implement such procedure :

1. How to solve the constraints satisfaction problems ?
2. What new fixations have to be tested ?

4.1 Solving the constraint satisfaction problem

Constraint programming gives many techniques to solve a satisfaction problem (see [11]) : constraint propagation, searching tree.

Let us illustrate constraint propagation in the following example.

Example 4.2 *Let us consider the* 4 *fixations above :*

$$\begin{array}{l} (a)\ x_2x_3 = 0 \\ (b)\ \overline{x}_2\overline{x}_3 = 0 \\ (c)\ \overline{x}_3\overline{x}_7 = 0 \\ (d)\ x_2x_7 = 0 \end{array}$$

and the constraints satisfaction problem

$$\begin{array}{l} x_2 = 1 \\ (a)\ x_2x_3 = 0 \\ (b)\ \overline{x}_2\overline{x}_3 = 0 \\ (c)\ \overline{x}_3\overline{x}_7 = 0 \\ (d)\ x_2x_7 = 0 \end{array}$$

Initially, the domain of the variables is $\{0,1\}$. *For each propagation, domains of the variables are updated if necessary (i.e a value in the domain is eliminated). Different propogation are possible depending on the exploration type of the searching tree. A depth-first (resp. breadth-first) searching tree corresponds to a depth-first (resp. breadth-first) search propagation.*
In depth-first constraint propagation, a change in the domain of a variable is propagated in a constraint where the variable appears. If this propagation has an effect on another variable, another propogation in constraints involving this variable is immediately perfomed and so on,...

For instance

- In constraint (a), $x_2 = 1$ *imply that the domain of* x_3 *is now* $\{0\}$.

This new change is propagated again in constraints where x_3 *appears.*

- In constraint (b), $x_3 = 0$ *imply that* $x_2 = 1$. *Thus domain of* x_2 $\{1\}$ *is the same.*

In constraint (c), $x_3 = 0$ *imply that the the domain of* x_7 *is now* $\{1\}$

This new change is propagated in constraints where x_7 *appears.*

- In constraint (c), $x_7 = 1$ *imply that* $x_2x_7 = 1 = 0$ *! (Impossible).*

Hence, it can be concluded that the constraint satisfaction problem is infeasible.

□

Many software implementing these techniques are available. Gnu-prolog developped in INRIA, prolog IV implemented by Prologia or Ilog Solver are some examples. For our numerical experiments, we use Ilog Solver 6.1.

4.2 Testing fixations

Since a fixation is a product, one possibility for the question 2 could be to consider all product of possible litterals. This is obviously not possible because the number of such product is exponnential. Another more reasonable possibility is to consider only product involving in existing fixations in F.
This choice reduces the number of fixations considered, but also make easier the resolution of the associated constraints satisfaction problem. Let us illustrate this idea.

Example 4.3 *Let us consider the two fixations found at the iteration 2 of the example 2.7.*

$$(a)\ (1-x_7)x_1x_6 = 0$$
$$(b)\ x_1(1-x_6)(1-x_7) = 0$$

Other fixations of lower cardinality that can be tested are, for instance, $x_1 = 1$ *or* $x_2x_7 = 1$.

None of these fixations yield infeasible constraint satisfaction problem.

At the opposed, introducing the constraint $(1-x_7)x_1 = 1$ *(i.e.* $x_7 = 0$, $x_1 = 1$*) imply direct propagation.*

Indeed, from (a) we have $x_6 = 0$ *and from (b) we have* $x_1(1-x_6)(1-x_7) = 1 = 0$ *! (Impossible).*

This imply $(1-x_7)x_1 = 0$.

As above, the constraint $(1-x_7)x_6 = 1$ *may also be tested.*

This imply direclty from (a), that $x_1 = 1$. *And from (b), we obtain* $x_1(1-x_6)(1-x_7) = 0 = 0$. *Thus a solution has been found, and nothing can be concluded for the fixation* $(1-x_7)x_6 = 0$ □

This technique has been generalized in the following procedure

- **Init**

- $F = \{F_i, i = 1, 2, ..., k\}$ *: a set of fixations*
- *sort* F *in increasing order of the cardinality of fixations* F_i
- *max = Maximal number of iteration*
- *end = false*
- *count = 0*
- *i = 1*
- *buffer*

• **Testing fixations**

while(end = false and count $\leq$ *max)*
{
. *end = true*
. *buffer = vide*

. *for all* $j \in |S_i|$
. {
. $F_{k+1} = \prod_{j \in S_i \setminus \{j\}} y_j$
. *result = solve the constraint satisfaction problem associated with* $F \cup \{F_{k+1}\}$
. *if (result = infeasible)*
. {
. $buffer = buffer \cup \{F_{k+1}\}$
. *end = false*
. }
. }

. *if(end = false)*
. {
. *for j = 1 to* $|buffer|$
. {
. $F = F \cup \{buffer_j\}$
. *Update F (i.e. eliminate fixations in which* $buffer_j$ *are included)*
. }
. *i = 1*
. }
. *else*
. *i = i + 1*

. *count = count + 1*
}

4.3 Testing equality

At the end of the algorithm above, other fixations deriving from the initial set F could be introduced in the problem. Nevertheless, many other type of relations between variables cannot be found with the procedure. For example, equality of two variables, as in the examples 2.7 and **??**, are such relations.

In order to be able to deduce also these relations, we propose to solve $n(n-1)$ constraint satisfaction problem of the following type :

$$
\begin{array}{llll}
(CSP_1) & x_i = x_j, & (CSP_2) & x_i = (1 - x_j), \\
& F_1 = 0, & & F_1 = 0, \\
& F_2 = 0, & & F_2 = 0, \\
& . = . & & . = . \\
& F_k = 0 & & F_k = 0
\end{array}
$$

where $1 \leq i < j \leq n$ and $F = \{F_1, F_2, ..., F_k\}$ are the initial set of fixations.

This idea is implemented in the following algorithm.

- **Init**

- $F = \{F_i, i = 1, 2, ..., k\}$: a set of fixations
- equalities : list of equalities found
- buffer

- **Testing equality**

for $i = 1$ to n
for $j = i + 1$ to n
{
. *result = solve the constraint satisfaction problem associated with $F \cup \{x_i = x_j\}$*
. *if (result = infeasible)*
. $buffer = buffer \cup \{x_i \neq x_j\}$
. *result = solve the constraint satisfaction problem associated with $F \cup \{x_i \neq x_j\}$*
. *if (result = infeasible)*
. $buffer = buffer \cup \{x_i = x_j\}$
}

$equalities = equalities \cup \{buffer\}$

While $buffer \neq \emptyset$
{
. *Update F with equality $buffer_1$ (i.e. the first equality of the list)*
. *Update the problem (QP) with equality $buffer_1$*
. *Update $buffer_i$ $(2 \leq i \leq |buffer|)$ with $buffer_1$*
. *Delete $buffer_1$*
}

4.4 Global algorithm : Testing equality and testing fixation

The equalities found in the procedure above allow to update the set of fixations F. The update set F may then be the initial set of others iterations of the algo-

rithm "Testing fixations". In turn, new fixations found by "Testing fixations" can be used as initial set for "Testing equality" and so on,... Thus, "Testing equality" and "Testing fixations" can be used successively in a more general procedure "Deductions" coupling the two algorithm.

- **Init**

- $F = \{F_i, i = 1, 2, ..., k\}$ *: a set of fixations*
- *max: maximal number of iterations*
- *newequalities : list of new equalities found by Testing equality*

- **Deductions**

while (end = false and ount $\leq$ *max)*
{
. *Testing fixations*
. *newequalities = Testing equality*
. *if newequalities* $= \emptyset$
. *end = true*
. *count = count + 1*
}

This algorithm has been used in numerical experiments. Results obtained are presented in the next section.

5 Numerical Results for the constraint programming approach

The algorithm "Deductions" presented above has been implemented and tested with different instances of the unconstrained quadratic 0-1 problem. The objectives of the numerical experiments is to evaluate the number of variables that can be fixed, and the maximal size and densities of problems that can be solved exactly by this approach.
We used two type of instances : ORLIB instances and our own randomly generated problems.

ORLIB Benchmark

We first consider problems of the OR-Library (see [3]) of size 50 and of densities of approximately 10%. The benchmarks of the OR-Library has been solved by heuristics (simulated annealing, Tabou, etc.). That is the reason why the results report the best known solutions. To our knowledge, there is no exact method giving the optimal values for these instances.

The best known solutions are reported in table 1 (see [14]). Since we have

p	n	d(%)	BEST
50.1	50	8	2098
50.2	50	9	3702
50.3	50	10	4626
50.4	50	9	3544
50.5	50	10	4012
50.6	50	8	3693
50.7	50	10	4520
50.8	50	11	4216
50.9	50	9	3780
50.10	50	8	3507

Table 1: OR-Library : Best Known Solutions

best known solutions, these instances allow us to verify the results found by our algorithms : "Finding Fixations" and "Dedutions". The numerical results obtained are presented in table 3. The meanings of the column of the table are explained below (table 2).

In the results table 3, for each instance in which we fix all the variables

p	Identifier of the problem
n	Number of Variables
d(%)	Density
Fixed	Number of Fixed Variables
$\overline{Fixed}$	Average Number of Fixed Variables
Fixation	Number of fixations
$\overline{Fixation}$	Average Number of Fixations
OPT	Optimal value of the problem
BEST	Best Known value
Time	Process Time of the algorithms "Fiinding Fixations" and "Deduction"
$\overline{Time}$	Average Process Time of the algorithms "Fiinding Fixations" and "Deduction"

Table 2: Caption

we report the optimal value found and compare it to the best known in the litterature. When it remains less than $k \leq 5$ variables to fix, we also find the optimal solution by enumerate the 2^k solutions. Otherwise, we consider that the constraint programming approach failed to find the optimal solution and

write "-" in the column OPT. For the 10 instances, only two of them cannot

p	n	d(%)	Fixed	OPT	Time (sec.)
50.1	50	8	3	-	12.6
50.2	50	9	45	3702	60
50.3	50	10	50	4626	17.7
50.4	50	9	48	3544	15.8
50.5	50	10	49	4012	19.8
50.6	50	8	36	-	17.9
50.7	50	10	50	4520	17.9
50.8	50	11	50	4216	18.0
50.9	50	9	47	3780	22.8
50.10	50	8	29	3507	14.6

Table 3: Application of the algorithm "Deductions"

be solved at optimality by the constraint programming approach. All optimal values found correspond exactly to the best known solution of the litterature.

Randomly generated instances

To have more extensive experiments, we generate our own instances of size 10, 20, 30, 40 of 10% of all densities. For each size and density, 10 instances have been generated. The table 4 below report the average number of fixed variables, fixations and time for the instances.

For instances of size 10 for all densities, the value of $\overline{Fixed}$ are between 8 and 10. This signify that, except a little number of them, all variables of the instances has been fixed by the constraint programming algorithms in few times less than 1sec. For instances of size 20 (resp. 30 and 40), the algorithm works perfectly until instances of density 40% (resp. 40, 10).

Let us noticed that in the columun $\overline{Fixations}$, it has been reported the number of fixations after the constraint programming algorithms. These fixations will be exploited later to solve exactly instances for which the constraints programming approach failed to fix all variables.

It can be seen that the approach gives good results for low density problems. To have an idea of the limits of the method, we generate additional instances of size 50, 60, 70, 80 and density 10%. For these instances, we limit the computation time to 300 sec. The sign "-" signify that the computation time exceed 300 sec. The results are in table 5

Once again, the process is very efficient for these instances. For problems of size up to 70, the average number of fixed variables is closed to the number

n	d(%)	$\overline{Fixed}$	$\overline{Fixations}$	$\overline{Time}$ (sec.)
10	10	9	0	0.0
10	20	10	0	0.0
10	40	10	0	0.0
10	60	10	1	0.0
10	80	8	32	0.1
10	100	9	9	0.2
20	10	19	0	0.0
20	20	20	1	0.1
20	40	17	19	1.3
20	60	7	767	7.5
20	80	0	1716	5.9
20	100	0	1868	7.3
30	10	29	1	0.2
30	20	29	6	1.1
30	40	14	1124	56.4
30	60	0	2822	22.1
30	80	0	2984	32.1
30	100	0	2989	43.1
40	10	38	6	1.8
40	20	29	316	38.5
40	40	0	3447	67.4
40	60	0	3986	75.9
40	80	0	3997	109.0
40	100	0	4000	158.5

Table 4: Size 10 to 40

n	d(%)	$\overline{VarFixed}$	$\overline{Fixations}$	$\overline{Time}$ (sec.)
50	10	45	31	2.2
60	10	55	30	22.6
70	10	56	184	74.5
80	10	64	182	312.9

Table 5: $50, 60, 70, 80$ variables

of variables. This indicates that for most of the instances, the corresponding problem has been solved completely. This number decreases for size 80. For problems of greater sizes, the processing time exceed the maximal time. This is why the results are not reported here.

We have observed also that the most consuming-time algorithm is the algorithm "Testing equality" in which $n(n-1)$ constraint satisfaction problem has to be solved. This algorithm gives very good result for low density problem but for higher density and size problems, no equality can be found. That is the reason why in the next section adressing problem of minimal size 50 for all densities, this algorithm will be swicth off.

In conclusion, the constraint programming algorithms are very efficient for low density problems (less than 10%) but are insufficient to solve completely greater problems with high densities. In all cases, the procedures yields a huge number of fixations that can be used to reduce the realisable domain of (QP) and help other techniques. The purpose of the section below is to use these fixations in a linearization of the problem (QP).

6 Constraints programming and linear programming

In order to solve exactly more difficult instances of the unconstrained quadratic 0-1 problem, the constraint programming approach is not enough. Nevertheless, the algorithms proposed above yield a huge number of fixations implying several variables that can help any relaxation to find more quickly the optimal value. Since fixations found can be linearized, it is natural to combine the approach with linear programming.

Let us highlight, that the purpose of combining constraints programming and linear programming is not to claim that this approach numerically outperform all previous known techniques for quadratic 0-1 programming. The goal is to show that fixations found improve any linearization of the problem (QP).

We assume that the scheme proposed below have to be done after the constraints programming approach phase, where some variables may be fixed, implying a new updated problem.

Let us recall the unconstrained quadratic 0-1 problem.

$$\begin{array}{lll} \text{(QP)} & \text{Min} & < c, x > + \frac{1}{2} x^\top Q x \\ & \text{s-t :} & (1)\ x \in \{0,1\}^n \end{array}$$

Linearization of (QP) is a technique of reformulation of into a constrained 0/1 linear program. They are many linear reformulations (see [15]). The most natural linearization is the so-called "classical" linearization due to Fortet[9][10]. It consists in replacing each product term $x_i x_j$ by an additionnal variable z_{ij}. This transformation yields the following formulation

$$
\begin{array}{lll}
\text{(CL)} \quad \text{Min} & \sum_{i=1}^{n} \sum_{j=(i+1)}^{n} q_{ij} z_{ij} + \sum_{i=1}^{n} c_i x_i & \\
\text{s-t :} & (1)\ x \in \{0,1\}^n & \\
& (2)\ z_{ij} \leq x_i & 1 \leq i, j \leq n \\
& (3)\ z_{ij} = z_{ji} & 1 \leq i < j \leq n \\
& (4)\ z_{ij} \geq x_i + x_j - 1 & 1 \leq i < j \leq n \\
& (5)\ z_{ij} \geq 0 & 1 \leq i < j \leq n
\end{array}
$$

The convex hull of (CL) feasible set is contained in the "Boolean Quadric Polytope"(BQP). BQP has been fully studied by Padberg[22], and an exponential number of other inequalities for the polytope of (CL) can be deduced from facets of the so-called "Boolean Cut Polytope" (Deza and Laurent[8], De Simone[25]). Boolean Quadric Polytope facets contribute to give a very good lower bound, specially for low density problems, corresponding to the optimal value of the continuous relaxation of (CL).

It has been observed in [15] that the classical linearization is a particular case of a huge set of linearizations. More accurately, a linearization of (QP) (in [15]) corresponds to a particular decomposition of the matrix Q. Thus, the set of linearization corresponds to the set of decomposition of Q. Classical linearization is only an instance of this set.

The best numerical results for the linearizations explored in [15] has been obtained for a linearization called the "Cliques-Edges Linearization" (see [15]). That is the reason why we will use it in this section. However, it is important to notice that the scheme that we propose can be apply (an can improve) to any linearizations.

Let us present briefly the linearization that we used.

Cliques-Edges Linearization

$$(CLEF) \quad \text{Min} \quad \sum_{i=1}^{n} c_i x_i + \sum_{i=1}^{n} \sum_{j=(i+1)}^{n} q_{ij}^{-} z_{ij} + \sum_{i=1}^{p} M_{\triangle_i} t_{\triangle_i}$$

$$\text{s-t} \quad (x,z) \in Co\{(x,z_{ij}) \in \mathbb{R}^{n+1} \mid x \in \{0,1\}^n,\ z_{ij} = x_i x_j\ ,\ 1 \leq i < j \leq n\}$$

$$(x, t_{\triangle_i}) \in QP_n^{\triangle_i}\ ,\ 1 \leq i \leq p$$

$$x \in [0,1]^n$$

In this formulation :

• ${q^{-}}_{ij} = min\{0, q_{ij}\}\ 1 \leq i < j \leq n.$

• $\triangle_1$, $\triangle_2$,...,$\triangle_p$ are cliques that cover the support graph G^+ of Q^+ (i.e positive value of Q).

• $M_{\triangle_i} = min\{{Q^{+}}_{kl} \ / \ k \in \triangle_i, l \in \triangle_i\}\ (i = 1, 2, ..., p).$

• $QP_n^{\triangle_i} = \{(t,x) \in \mathbb{R}^{n+1} \mid t \geq -\frac{k(k-1)}{2} + (k-1) \sum_{i=1}^{|\triangle_i|} x_i\ ,\ x \in [0,1]^n\ ,\ 2 \leq k \leq |\triangle_i|\ ,\ k \in \mathbb{N}\}$

To obtain a good lower bound, it is important to find as many facets (or valid inequalities) as possible of the feasible domain of (CLEF). We show below how to use fixation constraints to find valid inequalities.

6.1 Lifting

It has been seen in propostion 2.1 that the following two inequalities are necessary verified by the optimal solution of (QP) .

- $c_i x_i + \sum_{\substack{j=1 \\ j \neq i}}^{n} q_{ij} x_i x_j \leq 0\ ,\ i = 1, 2, ..., n.$
- $c_i(1 - x_i) + \sum_{\substack{j=1 \\ j \neq i}}^{n} q_{ij}(1 - x_i) x_j \geq 0\ ,\ i = 1, 2, ..., n.$

Thus, if $x_i = 1$ (resp. $x_i = 0$) the following linear inequalities is valid

$$c_i + \sum_{\substack{j=1 \\ j \neq i}}^{n} q_{ij} x_j \leq 0 \text{ (resp. } c_i + \sum_{\substack{j=1 \\ j \neq i}}^{n} q_{ij} x_j \geq 0)$$

To obtain a valid linear inequalities even when the variable x_i is not fixed to 1 (resp. 0)a lifting technique (see [19] [1] [21]) can be used.

Let us consider the case where $x_i = 1$, the lifting scheme will be the same for $x_i = 0$. We have the following valid inequalities

$$c_i + \sum_{\substack{j=1 \\ j \neq i}}^{n} q_{ij} x_j \leq 0$$

To generate a valid linear inequality for (QP) we have to lift the inequality above : that is to say to find a constant M_1 verifying the following inequality

$$M_1(1 - x_i) + c_i + \sum_{\substack{j=1 \\ j \neq i}}^{n} q_{ij} x_j \leq 0$$

Thus $M_1 \leq -c_i - \sum_{\substack{j=1 \\ j \neq i}}^{n} q_{ij} x_j \ \forall x \in \{0,1\}$. Hence the best value is

$$\begin{aligned} M_1 = \quad & \text{Min} \quad -c_i - \sum_{\substack{j=1 \\ j \neq i}}^{n} q_{ij} x_j \\ & \text{s-t :} \quad x \in \{0,1\}^n \end{aligned}$$

Finding the optimal value of this problem can be done in linear time since the optimal value is $M_1 = -c_i - \sum_{\substack{j=1 \\ j \neq i}}^{n} q_{ij}^{+} x_j$.

The resulting cut unfortunately leads to poor lower bound for the linearization (CL).
But this value can be improved by using fixations found with the heuristics "Finding fixation" , "Testing fixation" and "Testing equality" present in section 2, section 3 and section 4.

Indeed, let $S = \{1, 2, ..., n\}$, in proposition 2.5, it has been seen that all fixations have the form $x_i \prod_{j \in S} y_j = 0$ or $(1 - x_i) \prod_{j \in S} y_j = 0$.

In remark 2.6 we also see that these fixations correspond respectively to linear inequalities $x_i + \sum_{j \in S} \leq |S|$, and $(1 - x_i) + \sum_{j \in S} \leq |S|$.

Thus all the fixations can be written in a matrix form as $Ax \leq b$ where $A \in \{-1, 0, 1\}^{m \times n}$ and $b \in \mathbb{Z}^m$. Here, m is the number of fixations and n the number of variables. Therefore, the constraints $Ax \leq b$ can be introduced to find a better value for M_1. The resulting problem is

$$
\begin{array}{rll}
M_1 = & \text{Min} & -c_i - \sum\limits_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}x_j \\
& \text{s-t :} & x \in \{0,1\}^n \\
& & Ax \leq b
\end{array}
$$

At the opposed of the first problem, solving exactly this problem is not reasonable since it is a mixed integer linear problem, for which the matrix A is not in general totally unimodular. A realisable approach is to solve the continuous relaxation. The resulting lower bound can be still used.

We summarize the result above in the proposition below.

Proposition 6.1 (lifting cuts) *Let $\overline{M}_1$ and $\overline{M}_2$ defined by*

$$
\begin{array}{rll}
\overline{M}_1 = & Min & -c_i - \sum\limits_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}x_j \\
& s\text{-}t : & x \in [0,1]^n \\
& & Ax \leq b
\end{array}
\qquad\qquad
\begin{array}{rll}
\overline{M}_2 = & Max & -c_i - \sum\limits_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}x_j \\
& s\text{-}t : & x \in [0,1]^n \\
& & Ax \leq b
\end{array}
$$

we have the following valid inequalities, called "lifting cuts" :

- $\overline{M}_1(1-x_i) + c_i + \sum\limits_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}x_j \leq 0,\ i = 1,2,...,n.$
- $\overline{M}_2 x_i + c_i + \sum\limits_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}x_j \geq 0,\ i = 1,2,...,n.$

6.2 Linking inequalities

To improve the formulation (CLEF), it has been seen in the subsection above how to derive some valid inequalities by a lifting procedure. In this subsection, these cuts will be exploited to find other valid inequalities. Indeed, since these cuts are linear, let us consider that the general form of these inequalities is

$\sum_{i=1}^{n} a_i x_i \leq b$. By multiplying by x_k and $(1 - x_k)$ the inequalities the following new cuts can be found.

Proposition 6.2 *Let :*

- $S = \{1, 2, ..., n\}$, $k \in S$

- $T(i) = \{j \in S \mid (i, j) \in \Delta_j\}$, $i = 1, 2, ..., n$

- $T^- = \{i \in S \mid T(i) \neq \emptyset \text{ and } a_i < 0\}$

- $T^+ = \{i \in S \mid T(i) \neq \emptyset \text{ and } a_i > 0\}$

- $Z = \{i \in S \mid x_i x_k = z_{ik}\}$

we have the following valid inequalities, called "linking cuts" :

- $(a_k - b)x_k + \sum_{i \in Z} a_i z_{ik} + \sum_{i \in T^-} a_i t_{\Delta_j} \leq 0 \ \forall \ j \in T(i) \ \forall \ i \in T$
- $bx_k + \sum_{i \in S \setminus k} a_i x_i - \sum_{i \in Z} a_i z_{ik} + \sum_{i \in T^+} a_i t_{\Delta_j} \leq b \ \forall \ j \in T(i) \ \forall \ i \in T$

Proof □

With "lifting cuts" and "Linking cuts" the enhanced model ($CLEF^{++}$) that we have to solve is then

Cliques-Edges Linearization

$$(CLEF^{++}) \quad \text{Min} \quad \sum_{i=1}^{n} c_i x_i + \sum_{i=1}^{n} \sum_{j=(i+1)}^{n} q_{ij}^{-} z_{ij} + \sum_{i=1}^{p} M_{\triangle_i} t_{\triangle_i}$$

$$\text{s-t} \quad (x, z) \in Co\{(x, z_{ij}) \in \mathbb{R}^{n+1} \mid x \in \{0,1\}^n, \ z_{ij} = x_i x_j \ , \ 1 \leq i < j \leq n\}$$

$$(x, t_{\triangle_i}) \in QP_n^{\triangle_i} \ , \ 1 \leq i \leq p$$

$$x \in \{0,1\}^n$$

Lifting cuts

Linking cuts

The cuts are generated iteratively with a cutting plane algorithm.

6.3 Branching rules

The optimal value of the continuous relaxation of $(CLEF)$ gives a lower bound for the problem (QP). This bound is used in a branch bound tree to solve exactly the problem. It is well known that to reduce the number of nodes of the tree, good branching strategies are necessary. In the light of the inequalities found in proposition 2.1, we propose some branching rules.

Recall that these inequalities are

$$\text{(e)} \bullet \quad c_i x_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij} x_i x_j \leq 0 \ , \ i = 1, 2, ..., n.$$

$$\text{(f)} \bullet \quad c_i(1 - x_i) + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}(1 - x_i)x_j \geq 0 \ , \ i = 1, 2, ..., n.$$

To reduce the numbers of nodes of the branch-and-bound tree, our objective is to cuttof a branch as soon as possible. The idea is to branch to a variable x_i, for which either the quantity $c_i x_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij} x_i x_j$ is as big as possible or the quantity is $c_i(1 - x_i) + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}(1 - x_i)x_j \geq 0$ is as little as possible.
So, let x^* be the optimal solution of the continuous realxation of (CL) in a node of the branch and boudn tree, we associate to each variable the following quantity

$$\Delta_i = Max\{0, c_i x_i^* + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij} x_i^* x_j^*\} - Min\{c_i(1 - x_i^*) + \sum_{\substack{j=1 \\ j\neq i}}^{n} q_{ij}(1 - x_i^*)x_j^*\}$$

Δ_i gives a measure of the two quantities (e) and (f) for one variable. Indeed, big value of (e) and little value of (f) correspond to big value of Δ_i. Thus, to cut off a branch as soon as possible, we branch on the variable maximizing Δ_i.

In our numerical experiments with ILOG cplex, this branching rules has been implemented in a cplex branch callback.

7 Numerical Results for the constraints and linear programming approach

To evaluate the quality of the approach, numerical experiments have been performed with some instances of the OR-Library benchmark [3]. We use the instances of Glover, Kochenberger and Alidae [14]. It consists in 40 unconstrained quadratic 0-1 problems of size up to 200 and density up to 100%. In table **??** the size and densities are written. The experiments have been done

with ILOG Cplex 9.1 on a Personnal Computer (AMD Celeron 800 Mhz). The meaning of each column is explained in the table 6 below. Since the goal is

num	identifer of the problem
n	number of variables of the problem
d	density of the matrix Q
Nvar	number of variables of the linearization
Fixed	Number of variable fixed by the constraint programming approach
Fixation	Number of fixations
Lb	Lower bound at the root of the branch and bound tree
Opt	optimal value
Ub	upper bound at the root of the branch and bound tree (found by a greedy heuristic)
Nodes	Number of nodes of the branch-and-bound tree
Time	Processing time

Table 6: Caption

to show that the fixations may improve the linearization (CLEF), two series of tests have been done. The first one (table 7) with the formulation ($CLEF^{++}$) with lifting cuts, linking cuts, and the branching rules. The second exprimentations (table 8) with the formulation ($CLEF$)(that is to say without any lifting and linking cuts, and without the branching rules). For the two table, we report results on instances solved in processing time less than $5000sec.$ (maximal time processing).

In table 7, for the 40 instances, only the first 28 has been solved within the maximal time. The last cannot be solved in this time. It can be seen that the lower bound is always good, and few nodes are necessary. The 12 instances that cannot be solved witihin the maximal time are instances of size 100 (resp. 200) of densities strictly greater to 28% (resp. 9%).

Let us now compare the results of table 7 to the results of table 8. First, let us observe that the instances 27 and 28 are not reported on table 8. This signify that the corresponding problems cannot be solved with the formulation (CLEF). Therefore, this result shows that for these two instance, the fixations allow to improve the formulation. Instances 1 to 26 seems to be easy in comparison to instances 27 and 28. All of these instances can be solved by the two approaches. Formulation ($CLEF^{++}$) need more time but fewer number nodes for these instances, and at the opposed formulation ($CLEF$) need more nodes but fewer time.

In the light of these results, it can be seen the advantage of fixations is significant for difficult problem as 27 and 28. Thus, to have more experiments we generate randomly 5 instances of size 100 and density 30%. The same type of tests with ($CLEF$) and ($CLEF^{++}$) has been performed with these instances.

The results are reported in table 9 and in table 10.

It may be observed that two instances 30 and 33 (not reported in table 10) cannot not be solved with the formulation (CLEF). It shows once again that cuts (derived from fixations) are of interest to find more easily the optimal solution of the problems. Moreover, except the instance number 32, the time to solve optimally the problem is always better for the formulation ($CLEF^{++}$), and in all cases the number of necessary nodes decreases significantly in the formulation ($CLEF^{++}$).

num	n	d	Nvar	Fixed	Fixation	Lb	Opt	Ub	Nodes	Time
1	50	8	129	16	139	-3414.0	-3414.0	-3414.0	0	0.3
2	60	9	191	18	334	-6063.0	-6063.0	-6063.0	0	0.8
3	70	9	280	8	1016	-6196.5	-6037.0	-6037.0	4	3.1
4	80	9	359	5	2038	-8598.0	-8598.0	-8598.0	0	10.4
5	50	18	273	3	2191	-6043.0	-5737.0	-5737.0	14	6.3
6	30	40	202	0	2200	-4082.3	-3980.0	-3980.0	1	3.8
7	30	48	240	0	2526	-4670.5	-4541.0	-4541.0	4	4.4
8	100	6	374	52	206	-11109.0	-11109.0	-11109.0	0	3.5
9	20	98	191	0	261	-133.0	-133.0	-98.0	0	0.2
10	30	98	398	0	678	-132.0	-121.0	-106.0	3	1.1
11	40	98	692	0	1208	-122.6	-118.0	-90.0	4	4.1
12	50	98	1052	0	1881	-134.2	-118.0	-105.0	7	10.8
13	60	98	1489	0	2440	-152.0	-150.0	-137.0	2	19.8
14	70	98	1961	0	3252	-155.9	-146.0	-112.0	4	46.0
15	80	98	2479	0	4023	-172.7	-160.0	-160.0	4	76.8
16	90	98	3111	0	4718	-176.0	-145.0	-123.0	2	114.1
17	100	99	3748	0	5719	-168.9	-135.0	-119.0	4	184.9
18	125	98	5666	0	7358	-201.0	-154.0	-154.0	36	492.4
19	40	80	652	0	4000	-5561.5	-5058.0	-5058.0	22	11.6
20	50	62	797	0	5000	-7186.0	-6213.0	-6213.0	52	20.6
21	60	39	753	0	5950	-7258.5	-6665.0	-6665.0	34	20.4
22	70	29	785	0	6247	-8138.0	-7398.0	-7398.0	38	28.1
23	80	20	710	0	6242	-8002.0	-7362.0	-7362.0	46	41.5
24	90	9	462	5	2620	-5824.0	-5824.0	-5823.0	0	19.5
25	100	10	549	20	2338	-7225.0	-7225.0	-7225.0	0	18.8
26	100	9	580	4	4150	-6385.5	-6333.0	-6333.0	6	41.1
27	100	20	1104	0	8645	-7473.3	-6579.0	-6510.0	1224	1598.5
28	100	28	1510	0	9751	-10014.0	-9261.0	-9213.0	394	3360.0

Table 7: ($CLEF^{++}$) on Glover, Kochenberger and Alidae benchmark

num	n	d	Nvar	Fixed	Fixation	Lb	Opt	Ub	Nodes	Time
1	50	8	156	0	0	-3414.0	-3414.0	-3414.0	0	0.3
2	60	9	223	0	0	-6063.0	-6063.0	-6063.0	0	0.5
3	70	9	292	0	0	-6196.5	-6037.0	-6037.0	4	0.9
4	80	9	384	0	0	-8598.0	-8598.0	-8598.0	0	1.4
5	50	18	281	0	0	-6032.0	-5737.0	-5737.0	5	0.3
6	30	40	204	0	0	-4089.0	-3980.0	-3980.0	1	0.1
7	30	48	240	0	0	-4670.5	-4541.0	-4541.0	1	0.1
8	100	6	404	0	0	-11109.0	-11109.0	-11109.0	0	3.3
9	20	98	191	0	0	-133.0	-133.0	-98.0	0	0.1
10	30	98	398	0	0	-132.0	-121.0	-106.0	9	0.3
11	40	98	692	0	0	-122.6	-118.0	-90.0	6	0.9
12	50	98	1052	0	0	-134.2	-129.0	-105.0	12	2.4
13	60	98	1489	0	0	-152.0	-150.0	-137.0	7	3.3
14	70	98	1961	0	0	-155.9	-146.0	-112.0	4	5.9
15	80	98	2479	0	0	-172.7	-160.0	-160.0	16	13.4
16	90	98	3111	0	0	-176.0	-145.0	-123.0	10	22.2
17	100	99	3748	0	0	-168.9	-120.0	-119.0	15	40.5
18	125	98	5666	0	0	-201.0	-154.0	-154.0	79	225.1
19	40	80	652	0	0	-5561.5	-5058.0	-5058.0	14	0.6
20	50	62	797	0	0	-7186.0	-6213.0	-6213.0	89	2.1
21	60	39	753	0	0	-7258.5	-6665.0	-6665.0	38	1.2
22	70	29	785	0	0	-8138.0	-7398.0	-7398.0	65	1.8
23	80	20	716	0	0	-8052.5	-7362.0	-7362.0	37	1.9
24	90	9	490	0	0	-5824.0	-5824.0	-5823.0	0	2.2
25	100	10	595	0	0	-7225.0	-7225.0	-7225.0	0	3.5
26	100	9	594	0	0	-6387.0	-6333.0	-6333.0	1	3.5

Table 8: ($CLEF$) on Glover, Kochenberger and Alidae benchmark

num	n	d	Nvar	Fixed	Fixation	Lb	Opt	Ub	Nodes	Time
29	100	30	1568	0	9744	-14412.0	-11094.0	-11094.0	938	494.8
30	100	30	1564	0	9740	-14400.0	-9630.0	-9612.0	13540	4525.8
31	100	30	1565	0	9721	-14620.0	-11236.0	-11236.0	1407	578.5
32	100	30	1566	0	9688	-13522.0	-9945.0	-9945.0	1272	652.3
33	100	30	1567	0	9608	-13902.5	-9416.0	-9416.0	11308	4417.8

Table 9: ($CLEF^{++}$) on randomly generated instances

num	n	d	Nvar	Fixed	Fixation	Lb	Opt	Ub	Nodes	Time
29	100	30	1568	0	0	-14412.0	-11094.0	-11094.0	106943	2060.4
31	100	30	1565	0	0	-14620.0	-11236.0	-11236.0	105555	2024.1
32	100	30	1566	0	0	-13522.0	-9945.0	-9945.0	26519	556.4

Table 10: ($CLEF$) on randomly generated instances

8 Conclusions and Perspectives

In this article, local optimality conditions have been used in order to fix as many variables as possible or derive, as many constraints as possible between variables. The fixations of one variable, or a product of several corresponds, to the definition of "fixations" introduced in section 2. To find these fixations, a constraint programming approach has been proposed. With the algorithms proposed, low densities problems (less than 10%) of size up to 80 can be solved completely without any branching shemes.

For more difficult problem, a new scheme exploiting the fixations and linearization has been proposed. The fixations allow to derive new valid inequalities, obtained by a lifting procedure, and new branching rules. Full densities instances of size up to 125 may be solved.

We highlight that the preprocessing phase proposed and a linearization of (QP) are independent. Any kind of linearization can be used, and the preprocessing algorithms can be plugged on any resolution schemes. Therefore, any algorithm solving an unconstrained quadratic 0-1 problem can be improved by using fixations and associated algorithms.

To experiment how much a linearization can be improved, two type of numerical tests has been performed : with the formulation ($CLEF$) and with the formulation ($CLEF^{++}$). For difficult problems, the formulation ($CLEF^{++}$) gives valuable results with less time and number of nodes.

Since the fixations can improved any resolution scheme, a perspective of this paper is to extend this work to semidefinite programming relaxation of (QP) or other kind of linearizations. Moreover, it is well known that a quadratic 0-1 problem with linear equality constraints can always be transformed into an unconstrained one by a suitable penalization of the equality constraints. Thus, another interesting extension is to apply the scheme proposed in this paper to quadratic problem with equality constraints like the Graph Bipartitioning Problem or the Quadratic Assignment Problem.

References

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

[2] E. Balas and J.B. Mazzola. Non-linear 0-1 programming i : Linearization techniques. *Mathematical Programming*, 30:1–21, 1984.

[3] J.E. Beasley. Heuristic algorithms for the unconstrained binary quadratic programming problem. Technical report, Department of Mathematics, Imperial College of Science and Technology, London, England, 1998.

[4] A. Billionnet and A. Sutter. Minimization of a quadratic pseudo-boolean function. *European Journal of Operational Research*, 78:106–115, 1994.

[5] E. Boros and P. Hammer. Pseudo-boolean optimization. Rutcor Research Report RRR 48-2001, Rutgers University, Piscataway, New Jersey, 2001.

[6] A. Caprara, D. Pisinger, and P. Toth. Exact solution of the quadratic knapsack problem. *INFORMS Journal on Computing*, 11:125–137, 1999.

[7] C.T. Chang and C.C. Chang. A linearization method for mixed 0-1 polynomial programs. *Computers and Operations Research*, 27:1005–1016, 2000.

[8] M. Deza and M. Laurent. Facets for the complete cut cone. Research memorandum rmi, Department of Mathematical Engineering, University of Tokyo, Japan, 1988.

[9] R. Fortet. L'algbre de boole et ses applications en recherche oprationnelle. *Cahier du Centre d'Etudes de Recherche Oprationnelle*, 1:5–36, 1959.

[10] R. Fortet. Application de l'algbre de boole en recherche oprationnelle. *Revue Franaise de Recherche Oprationnelle*, 4:17–26, 1960.

[11] T. Fruhwirth and S. Abdennadher. *Essentials of Constraint Programming*. Springer, 2003.

[12] M. Garey and D. Johnson. *Computers and Intractibility: A Guide to the Theory of NP-Completeness*. W.H. Freeman & Company, 1979.

[13] F. Glover. Improved linear integer programming formulations of nonlinear integer problems. *Management Science*, 22(4):455–460, 1975.

[14] F. Glover, B. Alidaee, C. Rego, and G. Kochenberger. One-pass heuristics for large scale unconstrained binary quadratic problems. Technical Report HCES-09-00, Hearin Center for Enterprise Science, 2000.

[15] S. Gueye. *Linarisation et relaxation lagrangienne pour problmes quadratiques en variables binaires*. Thse de doctorat, Universit d'Avignon, 2002.

[16] P. Hansen. Method of non-linear 0-1 programming. *Annals of Discrete Mathematics*, 5:53–70, 1979.

[17] C. Helmberg and F. Rendl. Solving quadratic (0,1)-problems by semidefinite programs and cutting planes. Preprint sc-95-35, Konrad-ZuseZentrum Berlin, 1995.

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

[19] H. Marchand, A. Martin, R. Weismantel, and L. Wolsey. Cutting plane in integer and mixed integer programming. Technical Report CORE Discussion Paper 9953, Université Catholique de Louvain, Louvain-la-Neuve, Belgique, 1999.

[20] P. Michelon and L. Veilleux. Lagrangean methods for the 0-1 quadratic knapsack problem. *European Journal of Operational Research*, 92:326–341, 1996.

[21] G.L. Nemhauser and L.A. Wolsey. *Integer and combinatorial optimization.* Wiley, 1988.

[22] M. Padberg. The boolean quadric polytope: Some characteristics, facets and relatives. *Mathematical Programming*, 45:139–172, 1989.

[23] S. Poljak, F. Rendl, and H. Wolkowicz. A recipe for best semidefinite relaxation for (0,1)-quadratic programming. Technical Report CORR Report 94-7, University of Waterloo, 1994.

[24] I. Rosenberg. 0-1 optimization and nonlinear programming. *Revue Franaise d'Automatique, Informatique et Recherche Oprationnelle*, 6:95–97, 1972.

[25] C. De Simone. The cut polytope and boolean quadric polytope. *Discrete Mathematics*, 79:71–75, 1989.