Using constraint techniques for a safe and fast implementation of optimality-based reduction

Yahia Lebbah
Département d’Informatique
Université d’Oran Es-Senia
B.P . 1524 EL-M’Naouar
31000 Oran, Algeria
and
Laboratoire I3S, UNSA-CNRS
2000, route des lucioles
BP.121
06903 Sophia Antipolis -
Cedex France
ylebbah@gmail.com

Claude Michel
Laboratoire I3S, UNSA-CNRS
2000, route des lucioles
BP.121
06903 Sophia Antipolis -
Cedex France
cpjm@polytech.unice.fr

Michel Rueher
Laboratoire I3S, UNSA-CNRS
2000, route des lucioles
BP.121
06903 Sophia Antipolis -
Cedex France
rueher@polytech.unice.fr

ABSTRACT
Optimality-based reduction attempts to take advantage of the known bounds of the objective function to reduce the domain of the variables, and thus to speed up the search of a global optimum. However, the basic algorithm is unsafe, and thus, the overall process may no longer be complete and may not reach the actual global optimum. Recently, Kearfott has proposed a safe implementation of optimality-based reduction. Unfortunately, his method suffers from some limitations and is rather slow. In this paper, we show how constraint programming filtering techniques can be used to implement optimality-based reduction in a safe and efficient way.

Categories and Subject Descriptors
G.1.6 [Optimization]: Global optimization; F.4.1 [Mathematical Logic]: Logic and constraint programming

General Terms
Reliability Algorithms

Keywords
Global optimization, Constraint programming, Reliable computing

1. INTRODUCTION
Global optimization is a critical issue in numerous application of AI ranging from robot arm design [17], safety verification problems [2] to chemical equilibrium problems [9] and scheduling [2]. Many famous hard optimization problems, such as the traveling salesman problem or the protein folding problem, are global optimization problems. We consider here the global optimization problem \( P \) to minimize an objective function under nonlinear equalities and inequalities,

\[
\begin{align*}
\text{minimize} \quad & f(x) \\
\text{subject to} \quad & g_i(x) = 0, \quad i = 1..k \\
& g_j(x) \leq 0, \quad j = k + 1..m
\end{align*}
\]

with \( x \in \mathbb{R}^n \), \( f : \mathbb{R}^n \to \mathbb{R} \) and \( g_j : \mathbb{R}^n \to \mathbb{R} \); Functions \( f \) and \( g_j \) are continuously differentiable on some vector \( x \) of intervals of \( \mathbb{R} \).

The difficulties in such global optimization problems come mainly from the fact that there are generally many local minimizers but only few of them are global minimizers, and that the feasible region may be disconnected [23]. That is why it is often hard to solve continuous global optimization problems. Thus, different techniques have been proposed to improve the standard branch and bound methods. Optimality-based reduction is one of the most interesting of these techniques.

Optimality-based reduction (OBR) has been introduced by Ryoo and Sahinidis in [26] to take advantage of the known bound of the objective function to reduce the size of the domains of the variables. This technique uses a well known property of the saddle point to compute new bounds for the domain of a variable taking into account the known bounds of the objective function\(^1\). However, the basic OBR algorithm is unsafe, and thus, the overall process may no longer be complete and may not reach the actual global optimum.

Before going into further details, let us illustrate how OBR

\(^1\)Actually, Ryoo and Sahinidis have introduced in [26] four tests separated in two different techniques: optimality-based reduction and probing. While optimality-based reduction appears as an effective technique, probing, in its attempt to reduce the bounds of a variable, is, according to their authors, less efficient.
is working on the small following optimization problem:  

\[
\begin{align*}
\text{minimize} & \quad -x_1 + x_1 x_2 - x_2 \\
\text{subject to} & \quad c_1 : -2x_1 + 2x_2 \leq 1 \\
& \quad c_2 : 3x_1 - x_2 \leq 3; \\
& \quad x_1, x_2 \in [0, 5]. 
\end{align*}
\]

There is only one nonlinear term \(x_1 x_2\) which is linearized with bilinear relaxation. We obtain the following linear relaxation:

\[
\begin{align*}
\text{minimize} & \quad -x_1 - x_2 + x_3 \\
\text{subject to} & \quad c_1 : -2x_1 + 2x_2 \leq 1 \\
& \quad c_2 : 3x_1 - x_2 \leq 3; \\
& \quad c_3 : 3x_1 x_2 - x_3 \leq 3; \\
& \quad c_4 : 2x_1 x_2 + 3x_1 - x_3 \leq 2x_1 x_2; \\
& \quad x_1, x_2 \in [0, 5]; x_3 \in [0, 2.5].
\end{align*}
\]

where \(x_3\) is a linearization variable which stands for the bilinear term \(x_1 x_2\). \(\pi_i\) denotes the upper bound of variable \(x_i\) whereas \(\lambda_i\) denotes its lower bound.

Solving relaxation (3) with the simplex algorithm yields the following dual multipliers \(\lambda_1^* = 0\) and \(\lambda_2^* = 0.1666\), and the lower bound \(L = -1.08333\). Solving locally the original problem (2) with the local solver Minos [22] yields the upper bound \(U = -1.005\) on the feasible point \(x^* = (0.917, 1.062)\).

The OBR method (see section 4.1) uses relation \(-2x_1 + 2x_2 - 1 \geq -(U - L)/\lambda_1^*\) on the active constraint \(c_1\) to generate relation \(x_1 \leq 4.57\) which enables to remove interval \([4.57, 5]\) from the domain of \(x_1\). The critical issue is the safe computation of the dual solution of \(c_1\). Recently, Kearfott [13, 12] has proposed a safe implementation of OBR which is based on a valid bounding of the dual solution. Unfortunately, his method suffers from strong limitations and is rather slow.

We show in this paper that constraint programming techniques can be used in a very simple way to safely refine the potential non-solution boxes identified by the OBR method. Roughly speaking, filtering techniques are used to reduce these boxes to empty boxes, and thus, to prove that they do not contain any feasible point. These constraint programming techniques do not suffer from the same limitations as Kearfott’s method. The first experiments show that they are also much more efficient.

The rest of this paper is organized as follows. The first section recalls basics of continuous filtering techniques required in the rest of paper. We describe also the techniques of the \(\text{quad}\) algorithm, which plays an essential role in our framework. Section 2 provides the overall schema of a safe branch and bound process for global optimization. The next section describes the OBR method and introduces our safe implementation based on constraint techniques. Finally, preliminary experimental results are discussed.

2. BASICS ON FILTERING TECHNIQUES OVER CONTINUOUS DOMAINS

The first sub-section recalls basics of continuous filtering techniques required in the rest of paper. The second subsection recalls the intuition behind the \(\text{quad}\)-filtering technique.

---

\[\text{This problem derives from a problem found in [26] with a small variation on the constraint } c_1.\]
The algorithm converges and terminates if \( \epsilon \) is greater than zero.

A safe procedure is a key issue in a process that involves linear relaxations. Thanks to the work of Neumaier and Schichorina [24], a safe underestimation of a minimizer of a linear program is available. However, the computed minimizer is safe if and only if the problem itself is a conservative approximation of the initial problem. Michel et al. [20] have introduced corrections of the linear relaxations to ensure that the linear program is a conservative approximation of the initial program. This work has been extended by Bornardaille and Van Hentenryck [6] to cover multivariate relaxations.

Therefore, the quad-filtering algorithm offers an efficient and safe framework to prune the domains of the variables. It has shown its capability to achieve a more efficient pruning than 2B-consistency or box-consistency. Moreover, the quad-framework provides all the basics required to implement an efficient and safe global optimization solver based on linear relaxations.

Now, let us recall the overall branch and bound schema.

3. **THE BRANCH AND BOUND SCHEMA**

The well known branch and bound schema provides a very general framework to find a global minimizer. This framework is complete and can be implemented as a safe and rigorous platform [15]. We introduce directly the latter framework since the focus of this paper is on safe and rigorous approaches for continuous global optimization.

The branch and bound algorithm we use here combines interval analysis techniques and constraint programming techniques within the well known branch and bound schema described by Horst and Tuy in [11]. Interval analysis techniques enable to introduce safeguards that ensure rigorous and safe computations whereas constraint programming techniques improve the reduction of the feasible space.

The solver (see algorithm 1) computes enclosers for minimizers and safe bounds of the global minimum value within an initial box \( x \). The algorithm maintains two lists : a list \( \mathcal{L} \) of boxes to be processed and a list \( \mathcal{S} \) of proven feasible boxes. It provides a rigorous encloser \([L, U]\) of the global optimum with respect to a given tolerance \( \epsilon \).

The algorithm selects the box with the lowest lower bound of the objective function. The Prune function applies filtering techniques to reduce the size of the box \( x' \). In the framework we have implemented, Prune just uses a 2B-filtering algorithm. Then, LowerBound \( x' \) computes a rigorous lower bound of the objective within the box \( x' \) using a linear programming relaxation of the initial problem. Actually, function LowerBound is based on the linearization techniques of the quad framework. The safe LP solver computes \( f_{x'} \).

UpperBox \( x \) computes a feasible box. A local search method helps to quickly find an approximate feasible point. Interval techniques are used to check the feasibility of the provided box\(^3\). If UpperBox succeeds to prove feasibility then the box \( x_p \) that contains this proven feasible point is added to the list \( \mathcal{S} \). At this stage, if the box \( x' \) is empty then, either it does not contain any feasible point or its lower bound \( f_{x'} \) is greater than the current upper bound \( U \). In both cases, we say that the box is fathomed. If \( x' \) is not empty, the box is split along one of the problem variables\(^4\). At each box selection and processing, the algorithm maintains the lower lower bound \( L \) of the remaining boxes \( \mathcal{L} \) and the lowest upper bound \( U \) of proven feasible boxes. The algorithm terminates when the space between \( U \) and \( L \) becomes smaller than the given tolerance \( \epsilon \). Of course a proven optimum cannot always be found, and thus, algorithm 1 has to be stopped in some cases to get the feasible boxes which may have been found.

The next section is devoted to OBR techniques. We first recall the basic definitions, then we describe the method proposed by Kearfott, and finally we introduce our safe algorithm for computing OBR.

4. **OPTIMALITY-BASED REDUCTION**

4.1 Basics of optimality-based reduction

Optimality-based reduction has been introduced by Ryoo and Sahinidis in [26]. It takes advantage of a property of the saddle point (see for instance Chap. 5 of [21]) to reduce the domains of the variables of the problem to optimize. Optimality-based reduction relies on the following two theorems to improve the bounds of the domain of one variable:

**Theorem 1.** Let \( U \) be a known upper bound of the original problem \( P \), let \( L \) be a known lower bound of a convex relaxation \( R \) of \( P \), and assume that the constraint \( x_i - \tau_i \leq 0 \) is active at the optimal solution of \( R \) and has a corresponding multiplier \( \lambda_i > 0 \). Then

\[
\begin{align*}
    x_i & \geq x_i' \quad \text{with} \quad x_i' = \tau_i - \frac{U - L}{\lambda_i}.
\end{align*}
\]

Thus, if \( x_i' > \tau_i \), the domain of \( x_i \) can be set to \([x_i', \tau_i]\) without loss of any global optima.

\( \lambda_i \) denotes the dual solution of \( R \). Roughly speaking, a constraint \( A_i x_i \leq b_i \) is active if \( A_i x_i = b_i \). This equality may be difficult to be checked over the floating-point numbers.

\(^3\)We rely on the techniques introduced by Hansen in [10] to handle under-determined systems

\(^4\)Various heuristics are used to select the variable the domain of which has to be split.
Theorem 2. Let $U$ be a known upper bound of the original problem $P$, let $L$ be a known lower bound of a convex relaxation $R$ of $P$, and assume that the constraint $x_i - x_{i-1} \leq 0$ is active at the optimal solution of $R$ and has a corresponding multiplier $\lambda_i > 0$. Then

$$x_i \leq x_i'' = x_i + \frac{U - L}{\lambda_i}.$$  

Thus, if $x_i'' < x_i$, the domain of $x_i$ can be set to $[x_i, x_i'']$ without loss of any global optimum.

The first theorem provides a test to improve the lower bound of the domain of a variable while the second theorem provides a test to improve the upper bound of the domain of a variable.

Moreover, these valid inequalities have been generalized to the other constraints. The following theorem is the most general one:

Theorem 3. Let $U$ be a known upper bound of the original problem $P$, let $L$ be a known lower bound of a convex relaxation $R$ of $P$, and assume that the constraint $g_i(x) = g_i(x_{i-1}) \leq 0$ is active at the optimal solution of $R$ and has a corresponding multiplier $\lambda_i > 0$. Then

$$g_i(x) \geq -\frac{U - L}{\lambda_i}.$$  

This last theorem enables to enforce some constraints, and thus to reduce the domains of the variables.

All these theorems are more detailed and proved in [26].

4.2 Kearfott’s approach: safe OBR based on bounding rigorously dual variables

The critical issue in optimality-based reduction formulae (4)(5)(6) comes from the unsafe dual solution provided by the Simplex method. Note that the techniques suggested by Neumaier et al [24] and used in the Quad framework cannot be applied to compute a safe solution of the dual problem. So, the validity of dual solutions must be proved to keep the branch and bound process rigorous. This approach has been investigated by Kearfott [12] in the specific case of linear relaxations. For the sake of simplicity, we consider the following linear relaxation formulated with lower inequalities:

$$\min \quad d^T x$$

$$\text{s.t.} \quad A x \leq b$$  

The dual of (7) is the following LP

$$\max \quad b^T y$$

$$\text{s.t.} \quad A^T \lambda = d$$  

where $\lambda$ denotes the dual variables required in OBR formulae (4)(5)(6). The Kuhn-Tucker system (KT)\(^1\) is used to provide validated lower and upper bounds on the system (7) and (8).

$$\text{(KT)} \begin{cases} A^T \lambda - d = 0 \\ \lambda_i(A_i x - b_i) = 0, 1 \leq i \leq m \end{cases}$$  

where $A_i$ is $i$-th row of $A$.

It is well known that if the $i$-th constraint $A_i : x \leq b_i$ is inactive at a solution $-\text{i.e., strict inequality } x^* < b_i$ holds for $x^*$, the solution of the primal– then the corresponding dual variable (also called Lagrange multiplier) $y_i$ is equal to zero. Thus, inactive constraints must be identified to make the whole system (9) linear. In this approach there are two main critical issues:

1. Some of the constraints of (9) are redundant due to the fact that each equality constraint is replaced by two inequality constraints. So, the Newton methods cannot be used for the validation process.

2. Inactive constraints are identified with the approximate dual solution provided by the Simplex method (i.e., inactive constraints have a null dual solution).

Kearfott handles these issues by weakening the relaxation. Consequently, the final validation method based on the Newton method applied to (9) does not always succeed and, when it succeeds, the bounds could be wide due to the fact that the relaxation has been weakened.

4.3 A safe implementation of OBR based on constraint filtering techniques

As said before, the critical issue in the OBR method comes from the unsafe dual solution provided by the simplex algorithm. In other words, due to the rounding errors, we may lose the global optima when we use formula 4 to shrink the domain of some variable $x_i$.

The essential observation is that we can use filtering techniques to prove that no feasible point exists when the domain of $x_i$ is reduced to $[x_i, x_i']$. Indeed, if the constraint system

$$f(x) \leq U$$

$$g_i(x) = 0, \quad i = 1, k$$

$$g_j(x) \leq 0, \quad j = k + 1, m$$

does not have any solution when the domain of $x_i$ is set to $[x_i, x_i']$, then the domain reduction computed by the OBR method is valid; if the filtering cannot prove that no solution exits inside the considered box, we have just to add this box to $\mathcal{E}$, the list of boxes to be processed (See algorithms 1 and 2).

The same reasoning holds for the reduction of the domain of $f$, i.e., when algorithm 2 attempts to reduce the lower bound of the objective function $f$ by means of formula 6.

Algorithm 2 details the new process of the computation of the lower bound. Note that algorithm 1 remains almost unchanged: we have just to replace the call $f_{\text{LB}}(\text{Lower Bound}(x'))$ by $f_{\text{LB}}(x', \mathcal{L})$.

5. EXPERIMENTAL RESULTS

To evaluate the contribution of the constraint-based safety proof we have compared its performance with the method proposed by Kearfott [12] on 78 benchmarks. 17 of them, the ‘chem’ problems, are described in [25]. 6 of them come from pages 140 to 147 of Aslind thesis [1]. The 55 other problems come from COCONUT Library 1 benchmarks\(^8\). All benchmarks are: alkyl, chance, circle, dispatch, ex14_1_1, ex14_1_2, ex14_1_3, ex14_1_4, ex14_1_5, ex14_1_6, ex14_1_8, ex14_1_9, ex14_2_2, ex14_2_5, ex2_1_1, ex14_1_1, ex14_1_2, ex14_1_3, ex14_1_4, ex14_1_5, ex14_1_6, ex14_1_8, ex14_1_9, ex14_2_2, ex14_2_5, ex2_1_1, ex14_1_1, ex14_1_2, ex14_1_3, ex14_1_4, ex14_1_5, ex14_1_6, ex14_1_8, ex14_1_9, ex14_2_2, ex14_2_5, ex2_1_1.

\(^1\)http://www.mat.univie.ac.at/~neum/glopt/coconut/Benchmark/Library1_new_v1.html

\(^8\)The tested benchmarks are: COCONUT Library 1 benchmarks.
Algorithm 2 Computation of a safe lower bound with OBR

Function LowerBound($\mathbf{x}, L, U, \mathcal{L}, \mathcal{OBT}(\mathbf{f}, \mathcal{L})$)
\begin{itemize}
  \item $\mathcal{L}_r \leftarrow \emptyset \% \mathcal{L}_r$: set of potential non-solution boxes
  \item Compute $\mathbf{f}$ with Quad in $\mathbf{x}$
  \item for each variable $x_i$ do
    \begin{itemize}
      \item Apply formula 4 of OBR and add the generated potential non-solution boxes to $\mathcal{L}_r$
      \item for each box $B_i$ in $\mathcal{L}_r$ do
        \begin{itemize}
          \item $B'_i \leftarrow 2B$-filtering($B_i$)
          \item if $B'_i = \emptyset$ then reduce the domain of $x_i$
          \item else $B''_i \leftarrow$ Quad-filtering($B'_i$)
          \item if $B''_i = \emptyset$ then reduce the domain of $x_i$
          \item else add $B_i$ to $\mathcal{L}$; endif
        \end{itemize}
    \end{itemize}
  \item endif
  \item Apply formula 6 of OBR to reduce the lower bound of the objective function $\mathbf{f}$
  \item Use 2B-filtering and Quad-filtering to validate the reduction
\end{itemize}

these problems have less than 100 variables and 100 constraints.

The branch and bound algorithm has been implemented within the ICOS solver\(^7\). The linear relaxations for lower bounding are solved by means of the linear programming solver Coin/CLP\(^8\). The upper-bounding step uses the nonlinear programming solver Coin/IPOPT\(^9\) based on the interior point method. All experiments have been done on PC-Notebook/1Ghz.

We have performed the following experiments:

- a branch and bound without optimality-based reduction (labelled “no OBR” in the tables);
- a branch and bound with unsafe optimality-based reduction (labelled “unsafe OBR” in the tables);
- a branch and bound with safe optimality-based reduction based on Kearfott’s approach (labelled “safe OBR Kearfott” in the tables);
- a branch and bound with safe optimality-based reduction on our approach (labelled “safe OBR CP” in the tables);

To refute the boxes rejected by OBR, a combination of a 2B-filtering and Quad-filtering has been used. More precisely, Quad-filtering is only used when 2B-filtering fails to reject the boxes identified by OBR.

Table 1 is a synthesis of the results obtained by the different methods with a timeout of 500s. The second column gives the total amount of time (in second) required to compute all the branches (with a timeout of 500s). The last column gives the percentage of time saved using one of the

<table>
<thead>
<tr>
<th>name</th>
<th>safe OBR CP</th>
<th>safe OBR Kearfott</th>
<th>% saving</th>
</tr>
</thead>
<tbody>
<tr>
<td>himmel11</td>
<td>2.4</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ex5_2_2_case3</td>
<td>4.49</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>c-chem7</td>
<td>10.5</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ex2_1_5</td>
<td>3.52</td>
<td>15.64</td>
<td>77.49%</td>
</tr>
<tr>
<td>ex7_2_5</td>
<td>2.19</td>
<td>7.14</td>
<td>69.32%</td>
</tr>
<tr>
<td>ex14_2_5</td>
<td>2.48</td>
<td>7.19</td>
<td>65.50%</td>
</tr>
<tr>
<td>ex7_2_10</td>
<td>0.16</td>
<td>0.41</td>
<td>60.97%</td>
</tr>
<tr>
<td>ex8_1_6</td>
<td>0.73</td>
<td>1.54</td>
<td>52.59%</td>
</tr>
<tr>
<td>ex14_2_2</td>
<td>3.05</td>
<td>6.38</td>
<td>52.19%</td>
</tr>
<tr>
<td>ex7_3_1</td>
<td>3.38</td>
<td>5.95</td>
<td>43.19%</td>
</tr>
<tr>
<td>ex2_1_1</td>
<td>0.25</td>
<td>0.21</td>
<td>-19.04%</td>
</tr>
<tr>
<td>ex9_1_9</td>
<td>0.11</td>
<td>0.09</td>
<td>-22.22%</td>
</tr>
<tr>
<td>ex2_1_4</td>
<td>1.25</td>
<td>1.01</td>
<td>-23.76%</td>
</tr>
<tr>
<td>ex9_2_8</td>
<td>0.05</td>
<td>0.04</td>
<td>-25%</td>
</tr>
<tr>
<td>c-chem18</td>
<td>4.83</td>
<td>3.57</td>
<td>-35.29%</td>
</tr>
<tr>
<td>ex3_1_2</td>
<td>0.19</td>
<td>0.14</td>
<td>-35.71%</td>
</tr>
<tr>
<td>c-aude147</td>
<td>0.61</td>
<td>0.44</td>
<td>-38.63%</td>
</tr>
<tr>
<td>c-aude140b</td>
<td>0.13</td>
<td>0.09</td>
<td>-44.44%</td>
</tr>
<tr>
<td>alkyl</td>
<td>32.45</td>
<td>20.55</td>
<td>-57.90%</td>
</tr>
<tr>
<td>ex5_2_2_case1</td>
<td>7.75</td>
<td>2.88</td>
<td>-169.09%</td>
</tr>
</tbody>
</table>

Table 1: Synthesis of the results (with a timeout of 500s)

Table 2: Safe OBR CP vs safe OBR Kearfott

optimality-based method instead of the branch and bound alone.

In the first line, the OBR is used in an unsafe way. Thus, the result provided by the branch and bound process might be wrong. However, these experiments underline the potential benefit of the OBR in a branch and bound process. As a matter of fact, OBR might dramatically improve the speed of the search for a global optima.

Unfortunately, most of this benefit is lost by Kearfott’s safe approach. A contrario, the constraint-based approach still increases this improvement. An essential observation is that the constraint-based approach managed to handle all these problems in less than 50s whereas Kearfott’s method is stopped by a timeout of 500s for 3 of them.

The reader may be surprised by the fact that safe constraint-based OBR is even more efficient than the unsafe OBR. A careful analysis of the benchmark was necessary to understand this point. Actually, this is due to the fact that wrong domains reductions achieved by the unsafe OBR prevent the upper-bounding process from improving the current upper bound.

The constraint-based approach introduced here is about five time faster than Kearfott’s approach. In fact, our approach introduces a negligible overhead since the proof process mostly relies on a 2B-consistency which is an effective
technique here\textsuperscript{10}. This is outlined by table 2 which compares the results of the constraint-based approach with the result of Kearfott's method. The 10 first benchmarks are the ones where constraint-based OBR has the best behavior compared to Kearfott's method. The 10 last benchmarks are the ones where constraint-based OBR has the worst behavior compared to Kearfott's method. The first column contains the name of the benchmark. The second column gives the amount of time (in seconds) required to solve the benchmark with the constraint-based approach. The third column gives the amount of time (in seconds) required to solve the benchmark with Kearfott's method. The last column gives the percentage of time saved using the constraint-based approach instead of Kearfott's method.

6. CONCLUSION

In this paper, we show how constraint programming filtering techniques can be used to implement optimality-based reduction in a safe and efficient way. Thanks to constraint programming, the branch and bound algorithm can take advantage of the OBR through a simple but efficient refutation process.

Preliminary experiments have shown that our procedure compares well to the Kearfott's procedure. Using constraint-based refutation, OBR is up to five times faster than with Kearfott's procedure. As a result, constraint-based OBR can significantly improve the branch and bound process.

Constraint-based refutation has allowed a safe embedding of the OBR in a very simple way. This approach seems to be general enough to be applied to other unsafe methods. Our next work is thus to test this approach on other unsafe methods in order to improve the branch and bound process.

7. REFERENCES