HAL open science

Revisiting the upper bounding process in a safe Branch and Bound algorithm

Alexandre Goldsztejn, Yahia Lebbah, Claude Michel, Michel Rueher

To cite this version:

Alexandre Goldsztejn, Yahia Lebbah, Claude Michel, Michel Rueher. Revisiting the upper bounding process in a safe Branch and Bound algorithm. 14th International Conference on Principles and Practice of Constraint Programming, Sep 2008, Sydney, Australia. pp.598-602. hal-00297086

HAL Id: hal-00297086

https://hal.science/hal-00297086

Submitted on 15 Jul 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Revisiting the upper bounding process in a safe Branch and Bound algorithm*

Alexandre Goldsztejn[1], Yahia Lebbah[2,3], Claude Michel[3], and Michel Rueher[3]

[1] CNRS / Université de Nantes 2, rue de la Houssinière, 44322 Nantes, France
alexandre.goldsztejn@univ-nantes.fr
[2] Université d'Oran Es-Senia B.P. 1524 EL-M'Naouar, 31000 Oran, Algeria
ylebbah@gmail.com
[3] Université de Nice-Sophia Antipolis, I3S-CNRS, 06903 Sophia Antipolis, France
{cpjm, rueher}@polytech.unice.fr

Abstract. Finding feasible points for which the proof succeeds is a critical issue in safe Branch and Bound algorithms which handle continuous problems. In this paper, we introduce a new strategy to compute very accurate approximations of feasible points. This strategy takes advantage of the Newton method for under-constrained systems of equations and inequalities. More precisely, it exploits the optimal solution of a linear relaxation of the problem to compute efficiently a promising upper bound. First experiments on the Coconuts benchmarks demonstrate that this approach is very effective.

Introduction

Optimization problems are a challenge for CP in finite domains; they are also a big challenge for CP on continuous domains. The point is that CP solvers are much slower than classical (non-safe) mathematical methods on nonlinear constraint problems as soon as we consider optimization problems. The techniques introduced in this paper try to boost constraints techniques on these problems and thus, to reduce the gap between efficient but unsafe systems like BARON[1], and the slow but safe constraint based approaches. We consider here the global optimization problem $\mathcal{P}$ to minimize an objective function under nonlinear equalities and inequalities,

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & g_i(x) = 0,\ i \in \{1,..,k\} \\ & h_j(x) \leq 0,\ j \in \{1,..,m\} \end{array} \tag{1}$$

with $x \in \mathbf{x}$, $f : \mathbb{R}^n \to \mathbb{R}$, $g_i : \mathbb{R}^n \to \mathbb{R}$ and $h_j : \mathbb{R}^n \to \mathbb{R}$; Functions f , g_i and h_j are nonlinear and continuously differentiable on some vector $\mathbf{x}$ of intervals of $\mathbb{R}$. For convenience, in the sequel, $g(x)$ (resp. $h(x)$) will denote the vector of $g_i(x)$ (resp. $h_j(x)$) functions.

* An extented version of this paper is available at:
http://www.i3s.unice.fr/%7Emh/RR/2008/RR-08.11-A.GOLDSZTEJN.pdf

[1] See http://www.andrew.cmu.edu/user/ns1b/baron/baron.html

Algorithm 1 Branch and Bound algorithm

Function BB(IN $\mathbf{x}$, ϵ; OUT $\mathcal{S}$, $[L, U]$)

% $\mathcal{S}$: set of proven feasible points
% $\mathbf{f_x}$ denotes the set of possible values for f in $\mathbf{x}$
% $nbStarts$: number of starting points in the first upper-bounding
$\mathcal{L} \leftarrow \{\mathbf{x}\}$; $(L, U) \leftarrow (-\infty, +\infty)$; $\mathcal{S} \leftarrow UpperBounding(\mathbf{x}', nbStarts)$;
while $w([L, U]) > \epsilon$ **do**
  $\mathbf{x}' \leftarrow \mathbf{x}''$ such that $\underline{\mathbf{f}}_{\mathbf{x}''} = min\{\underline{\mathbf{f}}_{\mathbf{x}''} : \mathbf{x}'' \in \mathcal{L}\}$; $\mathcal{L} \leftarrow \mathcal{L} \backslash \mathbf{x}'$; $\overline{\mathbf{f}}_{\mathbf{x}'} \leftarrow min(\overline{\mathbf{f}}_{\mathbf{x}'}, U)$;
  $\mathbf{x}' \leftarrow Prune(\mathbf{x}')$; $\underline{\mathbf{f}}_{\mathbf{x}'} \leftarrow LowerBound(\mathbf{x}')$; $\mathcal{S} \leftarrow \mathcal{S} \cup UpperBounding(\mathbf{x}', 1)$;
  if $\mathbf{x}' \neq \emptyset$ **then** $(\mathbf{x}'_1, \mathbf{x}'_2) \leftarrow Split(\mathbf{x}')$; $\mathcal{L} \leftarrow \mathcal{L} \cup \{\mathbf{x}'_1, \mathbf{x}'_2\}$;
  if $\mathcal{L} = \emptyset$ **then** $(L, U) \leftarrow (+\infty, -\infty)$
    else $(L, U) \leftarrow (min\{\underline{\mathbf{f}}_{\mathbf{x}''} : \mathbf{x}'' \in \mathcal{L}\}, min\{\overline{\mathbf{f}}_{\mathbf{x}''} : \mathbf{x}'' \in \mathcal{S}\})$
endwhile

The difficulties in such global optimization problems come mainly from the fact that many local minimizers may exist but only few of them are global minimizers [3]. Moreover, the feasible region may be disconnected. Thus, finding feasible points is a critical issue in safe Branch and Bound algorithms for continuous global optimization. Standard strategies use local search techniques to provide a reasonable approximation of an upper bound and try to prove that a feasible solution actually exists within the box around the guessed global optimum. Practically, finding a guessed point for which the proof succeeds is often a very costly process.

In this paper, we introduce a new strategy to compute very accurate approximations of feasible points. This strategy takes advantage of the Newton method for under-constrained systems of equations and inequalities. More precisely, this procedure exploits the optimal solution of a linear relaxation of the problem to compute efficiently a promising upper bound. First experiments on the Coconuts benchmarks demonstrate that the combination of this procedure with a safe Branch and Bound algorithm drastically improves the performances.

The Branch and Bound schema

The algorithm (see Algorithm 1) we describe here is derived from the well known Branch and Bound schema introduced by Horst and Tuy for finding a global minimizer. Interval analysis techniques are used to ensure rigorous and safe computations whereas constraint programming techniques are used to improve the reduction of the feasible space.

Algorithm 1 computes enclosers for minimizers and safe bounds of the global minimum value within an initial box $\mathbf{x}$. Algorithm 1 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 tolerance ϵ.

Algorithm 1 starts with $UpperBounding(\mathbf{x}, nbStarts)$ which computes a set of feasible boxes by calling a local search with $nbStarts$ starting points and a proof procedure.

The box around the local solution is added to $\mathcal{S}$ if it is proved to contain a feasible point. At this stage, if the box $\mathbf{x}'$ is empty then, either it does not contain any feasible point or its lower bound $\underline{\mathbf{f}}_{\mathbf{x}'}$ is greater than the current upper bound U. If $\mathbf{x}'$ is not empty, the box is split along one of the variables[2] of the problem.

In the main loop, algorithm 1 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 $\mathbf{x}'$. In the framework we have implemented, *Prune* just uses a 2B-filtering algorithm [2]. Then, $LowerBound(\mathbf{x}')$ computes a rigorous lower bound $\underline{\mathbf{f}}_{\mathbf{x}'}$ using a linear programming relaxation of the initial problem. Actually, function *LowerBound* is based on the linearization techniques of the `Quad`-framework [1]. *LowerBound* computes a safe minimizer $\underline{\mathbf{f}}_{\mathbf{x}'}$ thanks to the techniques introduced by Neumaier et al.

Algorithm 1 maintains the lowest 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 ϵ.

The Upper-bounding step (see Algorithm 2) performs a multistart strategy where a set of *nbStarts* starting points are provided to a local optimization solver. The solutions computed by the local solver are then given to a function $InflateAndProve$ which uses an existence proof procedure based on the Borsuk test. However, the proof procedure may fail to prove the existence of a feasible point within box $\mathbf{x}_p$. The most common source of failure is that the "guess" provided by the local search lies too far from the feasible region.

A new upper bounding strategy

The upper bounding procedure described in the previous section relies on a local search to provide a "guessed" feasible point lying in the neighborhood of a local optima. However, the effects of floating point computation on the provided local optima are hard to predict. As a result, the local optima might lie outside the feasible region and the proof procedure might fail to build a proven box around this point.

We propose here a new upper bounding strategy which attempts to take advantage of the solution of a linear outer approximation of the problem. The lower bound process uses such an approximation to compute a safe lower bound of $\mathcal{P}$. When the LP is solved, a solution x_{LP} is always computed and, thus, available for free. This solution being an optimal solution of an outer approximation of $\mathcal{P}$, it lies outside the feasible region. Thus, x_{LP} is not a feasible point. Nevertheless, x_{LP} may be a good starting point to consider for the following reasons:

- At each iteration, the branch and bound process splits the domain of the variables. The smaller the box is, the nearest x_{LP} is from the actual optima of $\mathcal{P}$.
- The proof process inflates a box around the initial guess. This process may compensate the effect of the distance of x_{LP} from the feasible region.

[2] Various heuristics are used to select the variable the domain of which has to be split.

Algorithm 2 Upper bounding build from the LP optimal solution x^*_{LP}

Function `UpperBounding`(`IN` $\mathbf{x}$, x^*_{LP}, $nbStarts$; `OUT` $\mathcal{S}'$)

% $\mathcal{S}'$: list of proven feasible boxes; $nbStarts$: number of starting points
% x^*_{LP}: the optimal solution of the LP relaxation of $\mathcal{P}(\mathbf{x})$
$\mathcal{S}' \leftarrow \emptyset$; $x^*_{corr} \leftarrow$FeasibilityCorrection(x^*_{LP}); $\mathbf{x}_p \leftarrow$InflateAndProve(x^*_{corr}, $\mathbf{x}$);
if $\mathbf{x}_p \neq \emptyset$ **then** $\mathcal{S}' \leftarrow \mathcal{S}' \cup \mathbf{x}_p$
return $\mathcal{S}'$

However, while x_{LP} converges to a feasible point, the process might be quite slow. To speed up the upper bounding process, we have introduced a light weight, though efficient, procedure which compute a feasible point from a point lying in the neighborhood of the feasible region. This procedure which is called *FeasibilityCorrection* will be detailed in the next subsection.

Algorithm 2 describes how an upper bound may be build from the solution of the linear problem used in the lower bounding procedure.

Computing pseudo-feasible points

This section introduces an adaptation of the Newton method to under-constrained systems of equations and inequalities which provides very accurate approximations of feasible points at a low computational cost. When the system of equations $g(x) = 0$ is under-constrained there is a manifold of solutions. $l(x)$, the linear approximation is still valid in this situation, but the linear system of equations $l(x) = 0$ is now under-constrained, and has therefore an affine space of solutions. So we have to choose a solution $x^{(1)}$ of the linearized equation $l(x) = 0$ among the affine space of solutions. As $x^{(0)}$ is supposed to be an approximate solution of $g(x) = 0$, the best choice is certainly the solution of $l(x) = 0$ which is the closest to $x^{(0)}$. This solution can easily be computed with the Moore-Penrose inverse: $x^{(1)} = x^{(0)} - A_g^+(x^{(0)})g(x^{(0)})$, where $A_g^+ \in I\!R^{n\times m}$ is the Moore-Penrose inverse of $A_g \in I\!R^{m\times n}$, the solution of the linearized equation which minimizes $||x^{(1)} - x^{(0)}||$. Applying previous relation recursively leads to a sequence of vectors which converges to a solution close to the initial approximation, provided that this latter is accurate enough.

The Moore-Penrose inverse can be computed in several ways: a singular value decomposition can be used, or in the case where A_g has full row rank (which is the case for $A_g(x^{(0)})$ if $x^{(0)}$ is non-singular) the Moore-Penrose inverse can be computed using $A_g^+ = A_g^T(A_g A_g^T)^{-1}$.

Inequality constraints are changed to equalities by introducing slack variables: $h_j(x) \leq 0 \iff h_j(x) = -s_i^2$. So the Newton method for under-constrained systems of equations can be applied.

Experiments

In this Section, we comment the results of the experiments with our new upper bounding strategy on a significant set of benchmarks. Detailled results can be found in the resarch report ISRN I3S/RR-2008-11-FR[3]). All the benchmarks come from the collection of benchmarks of the Coconuts project[4]. We have selected 35 benchmarks where Icos succeeds to find the global minimum while relying on an unsafe local search. We did compare our new upper bounding strategy with the following upper bounding strategies:

S1: This strategy directly uses the guess from the local search, i.e. this strategy uses a simplified version of algorithm 1 where the proof procedure has been dropped. As a consequence, it does not suffer from the difficulty to prove the existence of a feasible point. However, this strategy is unsafe and may produce wrong results.

S2: This strategy attempts to prove the existence of a feasible point within a box build around the local search guess. Here, all provided solutions are safe and the solving process is rigorous.

S3: Our upper bounding strategy where the upper bounding relies on the optimal solution of the problem linear relaxation to build a box proved to hold a feasible point. A call to the correction procedure attempts to compensate the effect of the outer approximation.

S4: This strategy applies the correction procedure to the output of the local search (to improve the approximate solution given by a local search).

S5: This strategy mainly differs from S3 by the fact that it does not call the correction procedure

S3, our new upper bounding strategy is the best strategy: 31 benchmarks are now solved within the 30s time out; moreover, almost all benchmarks are solved in much less time and with a greater amount of proven solutions. This new strategy improves drastically the performance of the upper bounding procedure and competes well with a local search.

Current work aims at improving and generalizing this framework and its implementation.

References

1. Yahia Lebbah, Claude Michel, Michel Rueher, David Daney, and Jean-Pierre Merlet. Efficient and safe global constraints for handling numerical constraint systems. *SIAM Journal on Numerical Analysis*, 42(5):2076–2097, 2004.
2. Olivier Lhomme. Consistency techniques for numeric CSPs. In *Proceedings of IJCAI'93*, pages 232–238, Chambéry(France), 1993.
3. Arnold Neumaier. Complete search in continuous global optimization and constraint satisfaction. *Acta Numerica*, 2004.

[3] http://www.i3s.unice.fr/%7Emh/RR/2008/RR-08.11-A.GOLDSZTEJN.pdf

[4] See http://www.mat.univie.ac.at/~neum/glopt/coconut/Benchmark/Benchmark.html.