TOWARD A RIGOROUS AND EFFICIENT GLOBAL OPTIMIZER

Michel RUEHER

(Joint work with Yahia LEBBAH and Claude MICHEL)

CEP Project I3S–CNRS
University of Nice – Sophia Antipolis

December 2006

OUTLINE

- Motivations
- **QUAD-SOLVER**: a CP framework based on safe linear relaxations
- **QUAD-OPT**: a safe global optimisation framework

MOTIVATIONS

THE PROBLEM

We consider the continuous global optimisation problem $\mathcal{P}$

$$\mathcal{P} \equiv \begin{cases} \min & f(x) \\ \text{s.c.} & g_j(x) = 0, \ j = 1..k \\ & g_j(x) \leq 0, \ j = k+1..m \\ & \underline{\mathbf{b}} \leq x \leq \overline{\mathbf{b}} \end{cases} \tag{1}$$

with

- $\mathbf{b} = [\underline{\mathbf{b}}, \overline{\mathbf{b}}]$: a vector of intervals of $\mathbb{R}$
- $f : \mathbb{R}^n \rightarrow \mathbb{R}$ et $g_j : \mathbb{R}^n \rightarrow \mathbb{R}$
- Functions f and g_j: are continuously differentiable on $\mathbf{b}$

TRENDS IN GLOBAL OPTIMISATION

- **Performance**

  Most successful systems (Baron, αBB, ...) use linear relaxations
  $\rightarrow$ complete methods, but **not rigorous**

- **Rigour**

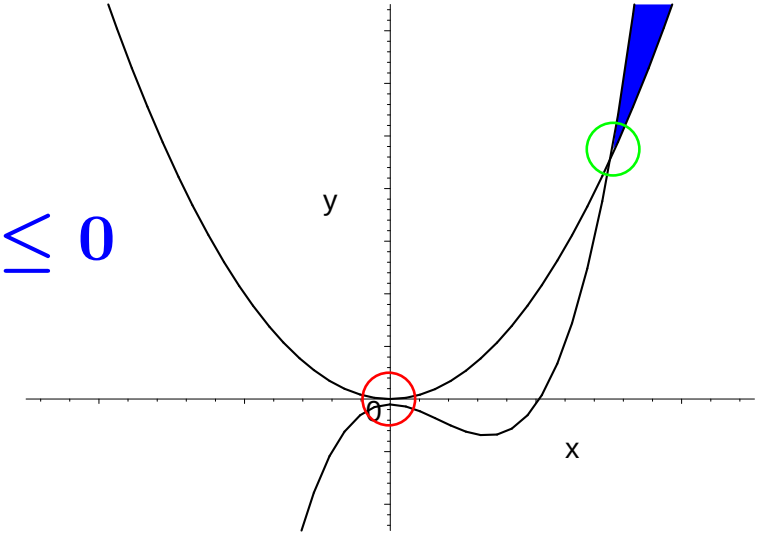
  Mainly rely on interval computation
  ... available systems (e.g., Globsol) are **quite slow**

➢ **Challenge:** to combine the advantages of both approaches in an efficient and **rigorous** global optimisation framework

EXAMPLE OF FLAW DUE TO A LACK OF RIGOUR

Consider the following optimisation problem:

$$
\begin{array}{ll}
\min & x \\
\text{s. t.} & y - x^2 \geq 0 \\
& y - x^2 * (x - 2) + 10^{-5} \leq 0 \\
& x, y \in [-10, +10]
\end{array}
$$



Baron 6.0 and Baron 7.2 find 0 as the minimum . . .

A CP FRAMEWORK BASED ON SAFE LINEAR RELAXATIONS

A SAFE CONSTRAINT PROGRAMMING FRAMEWORK

Numeric CSP $(\mathcal{X}, \mathcal{D}, \mathcal{C})$

- $\mathcal{X} = \{x_1, \ldots, x_n\}$ is a set of variables
- $\mathcal{D} = \{D_{x_1}, \ldots, D_{x_n}\}$ is a set of domains
  (D_{x_i} contains all acceptable values for variable x_i)
- $\mathcal{C} = \{c_1, \ldots, c_m\}$ is a set of constraints

CP : OVERALL SCHEME

A Branch & Prune schema :

1. **Pruning the search space**
2. **Making a choice to generate two (or more) sub-problems**

- The pruning step $\longrightarrow$ **shrinks an interval** when it can prove that the upper bound or the lower bound does not satisfy some constraint
- The branching step $\longrightarrow$ **splits the interval** associated to some variable in two or more intervals

LOCAL CONSISTENCIES (1)

A constraint system C satisfies a partial consistency property if **a relaxation of C is consistent**

Consider $X = [\underline{x}, \overline{x}]$ and $C(x, x_1, \ldots, x_n) \in \mathcal{C}$:
if $C(x, x_1, \ldots, x_n)$ does not hold for any values $a \in [\underline{x}, x']$, then X may be shrinked to $X = [x', \overline{x}]$

LOCAL CONSISTENCIES (2)

→ **2B–consistency** (or hull consistency) only requires to check the Arc–Consistency property **for each bound** of the intervals

→ **Box–consistency** is a coarser relaxation of Arc–Consistency than 2B–consistency ... **but Box–consistency algorithms actually achieve a stronger filtering than 2B–consistency**

2B–CONSISTENCY

Variable x is 2B–consistent for constraint $f(x, x_1, \ldots, x_n) = 0$ if the lower (resp. upper) bound of the domain X is the smallest (resp. largest) solution of $f(x, x_1, \ldots, x_n)$

A CSP is 2B–consistency iff all its constraints are 2B–consistent

2B–CONSISTENCY FILTERING

Algorithms that achieve 2B–consistency filtering are based upon projection functions

- Analytic functions always exist when the variable to express in terms of the others appears only once in the constraint
  $\longrightarrow$ considers that **each occurrence is a different new variable**
  $\longrightarrow$ initial constraints are decomposed into "primitive" constraints

- Decomposition does not change the semantics of the initial constraints system but **it amplifies the dependency problem**

BOX–CONSISTENCY

Variable x is Box–Consistent for constraint $f(x, x_1, \ldots, x_n) = 0$ if the bounds of the domain of x correspond to the **leftmost and the rightmost zero** of $F(X, X_1, \ldots, X_n)$ the **optimal interval extension** of $f(x, x_1, \ldots, x_n)$

C is Box–Consistent if, for all X_i the following relations hold :

1. $C(X_1, \ldots, X_{i-1}, [\underline{X_i}, \underline{X_i}^+), X_{i+1}, \ldots, X_k)$

2. $C(X_1, \ldots, X_{i-1}, (\overline{\overline{X_i}}^-, \overline{X_i}], X_{i+1}, \ldots, X_k)$

BOX–CONSISTENCY FILTERING

Transformation of the constraint $C_j(x_{j_1}, ...x_{j_k})$ into k mono-variable constraints $C_{j,l}$ by substituting their intervals for all variables but one

$\longrightarrow$ The two extremal zeros of $C_{j,l}$ can be found by a **dichotomy algorithm combined with a mono-variable version of the interval Newton method**

LIMITS OF LOCAL CONSISTENCIES

➢ A constraint is handled as a **black-box** by local consistencies (2B–filtering,BOX–filtering)

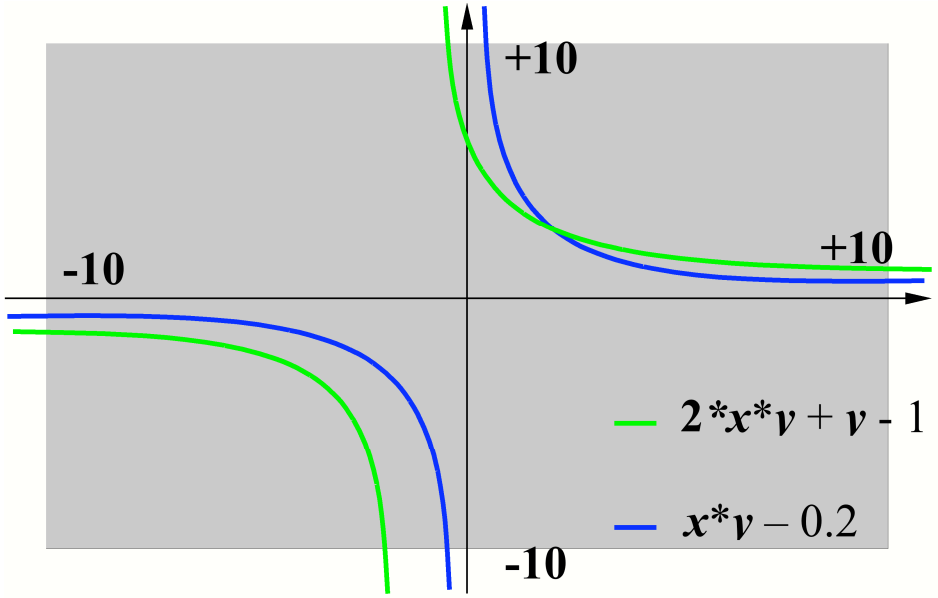
- No way to catch the dependencies between constraints
- Splitting is behind the success for small dimensions

➢ Higher consistencies (KB–filtering,Bound–filtering)
→ visiting numerous combinations

LIMITS OF LOCAL CONSISTENCIES(2)

- $2x*y + y = 1$
- $x*y = 0.2$

  x, y *in [-10, +10]*



- ***Box-consistency : no reduction***

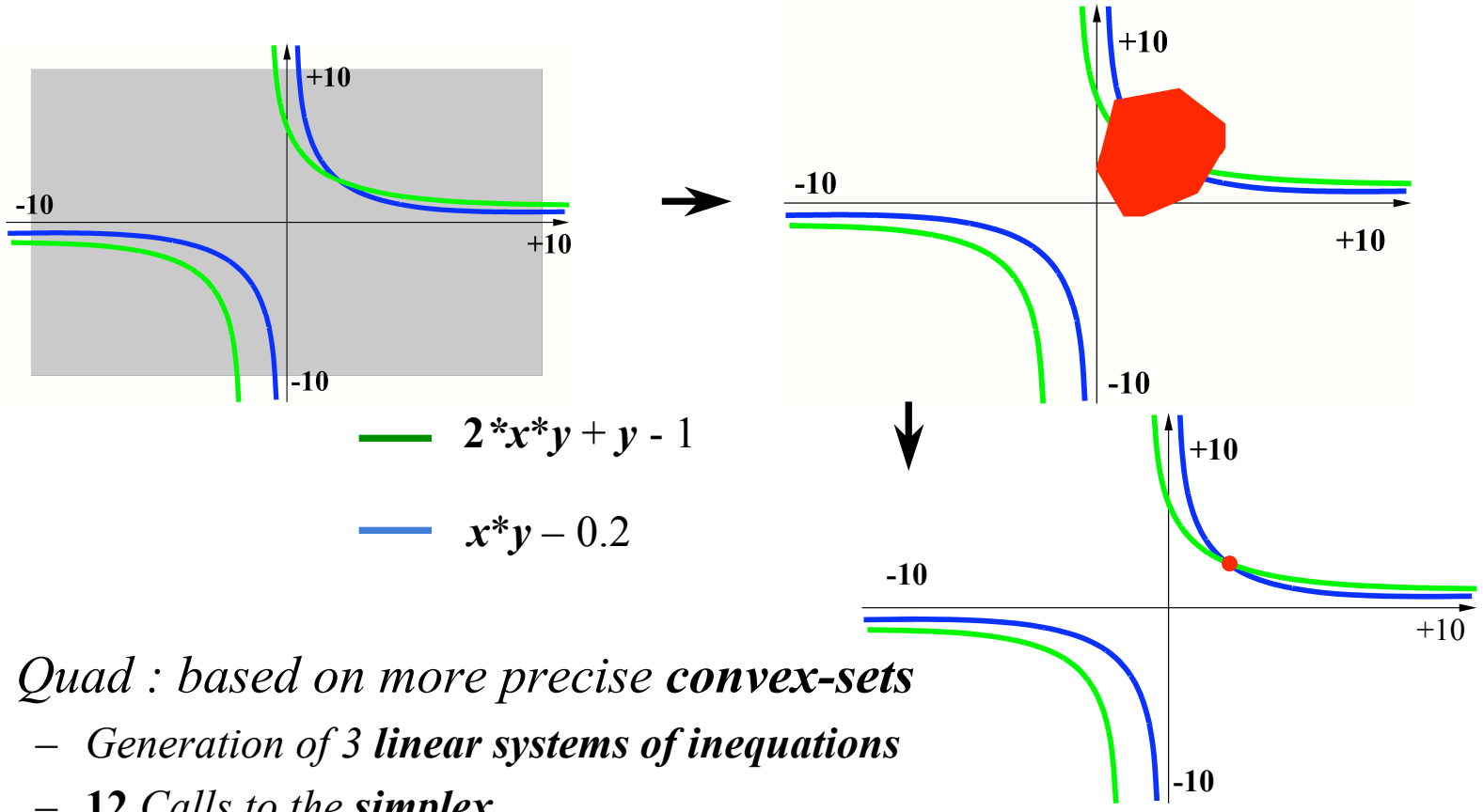
  $0 \in [-211, 209]$ $0 \in [-211, 189]$

  $0 \in [-100.2, 99.8]$ $0 \in [-100.2, 99.8]$

- ***2B-consistency : no reduction***

  (division by zero when computing the projection functions)

QUAD_SOLVER



- *Quad : based on more precise* ***convex-sets***
  - *Generation of 3* ***linear systems of inequations***
  - **12** *Calls to the* ***simplex***
  - *Provides* ***exact solutions***

QUAD_SOLVER: SAFE USE OF LINEAR RELAXATION

- ***A global constraint to handle a tight approximation of the constraint system with the Simplex***
- ***Combines***
  - ***safe and rigorous linear relaxations***
  - ***local consistencies (2B, Box) and interval methods (Newton)***

THE QUAD_SOLVER FRAMEWORK

- **Reformulation**

  ➢ capture the linear part of the problem
  → replace each non linear term by a new variable
  eg x^2 by y_i

- **Linearisation/relaxation**

  ➢ introduce ***redundant linear constraints***
  → tight approximations of the non-linear terms (RLT)

- **Computing min(x)** $= \underline{x_i}$ and **max(x)** $= \overline{x_i}$ in LP

LINEARISATION OF x^2

➢ $f(x) = x^2$ with $\underline{x} \leq x \leq \overline{x}$ is approximated by :

$$\begin{aligned} L_1(y, \alpha) &\equiv [(x - \alpha)^2 \geq 0]_l \text{ where } \alpha \in [\underline{x}, \overline{x}] \\ L_2(y) &\equiv (\underline{x} + \overline{x})x - y - \underline{x} * \overline{x} \geq 0 \end{aligned}$$

- $[(x - \alpha_i)^2 = 0]_l$ generates the tangents to $y = x^2$ at $x = \alpha_i$
- $L_1(y, \overline{x})$ and $L_1(y, \underline{x})$: underestimations of y
  $L_2(y)$: overestimation of y

QUAD_SOLVER only computes $L_1(y, \overline{x})$ and $L_1(y, \underline{x})$

LINEARISATION OF x^2

Example 1: relaxation of $y = x^2$ with $x \in [-4, 5]$

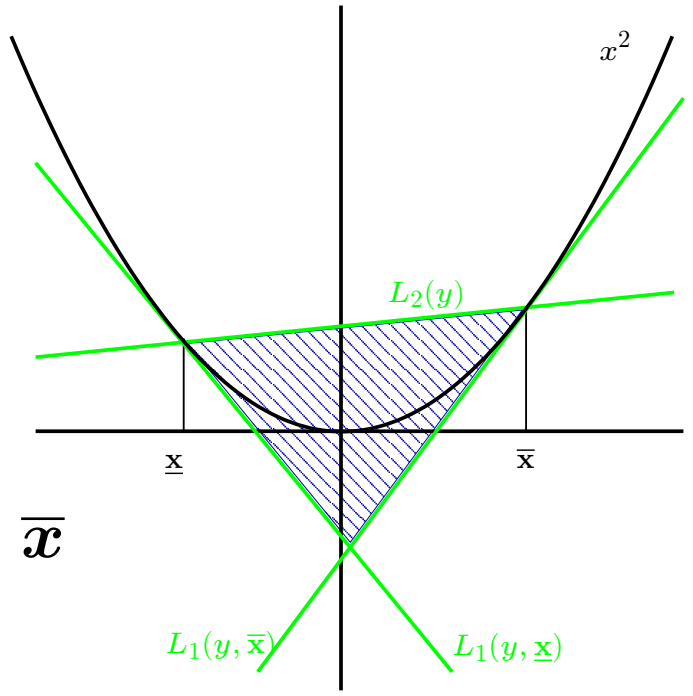
- $L_1(y, \alpha) \equiv y \geq 2\alpha x - \alpha^2$

  $\boldsymbol{L_1(y, -4) : y \geq -8x - 16}$

  $\boldsymbol{L_1(y, 5) : y \geq 10x - 25}$

- $\boldsymbol{L_2(y) \equiv y \leq (\underline{x} + \overline{x})x - \underline{x} * \overline{x}}$

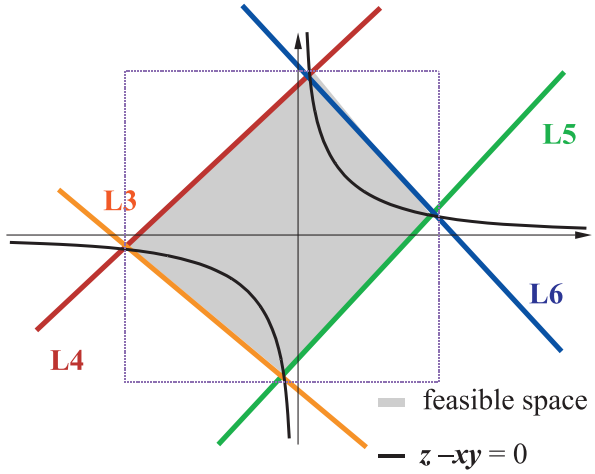
  $\boldsymbol{L_2(y) : y \leq x + 20}$



LINEARISATION OF xy

Example 2: relaxation of $z = xy$
with $x \in [-5, +5], y \in [-5, +5]$

$$L_3(z) \equiv [(x - \underline{x_i})(y - \underline{x_j}) \geq 0]_l$$
$$L_4(z) \equiv [(x - \underline{x_i})(\overline{x_j} - y) \geq 0]_l$$
$$L_5(z) \equiv [(\overline{x_i} - x)(y - \underline{x_j}) \geq 0]_l$$
$$L_6(z) \equiv [(\overline{x_i} - x)(\overline{x_j} - y) \geq 0]_l$$



The *Quad_Solver* filtering algorithm

Function `Quad`_filtering(`IN`: $\mathcal{X}$, $\mathcal{D}$, $\mathcal{C}$, ϵ) **return** $\mathcal{D}'$

1. ***Reformulation***
   $\longrightarrow$ linear inequalities $[\mathcal{C}]_R$ for the nonlinear terms in $\mathcal{C}$

2. ***Linearisation/relaxation of the whole system*** $[\mathcal{C}]_L$
   $\longrightarrow$ a linear system $LR = [\mathcal{C}]_L \cup [\mathcal{C}]_R$

3. $\mathcal{D}'$:= $\mathcal{D}$

4. ***Pruning***:
   While amount of reduction of some bound $> \epsilon$ **and** $\emptyset \notin \mathcal{D}'$ **Do**

   (a) Update the coefficients of $[\mathcal{C}]_R$ according to $\mathcal{D}'$
   (b) Reduce the lower and upper bounds $\underline{x}'_i$ and $\overline{x}'_i$ of each ***initial*** variable $x_i \in \mathcal{X}$ computing $\mathbf{min}$ and $\mathbf{max}$ of x_i subject to LR with a LP solver)

ISSUES IN THE USE OF LINEAR RELAXATION

- ◇ Coefficients of linear relaxations are scalars

  $\Rightarrow$ computed with ***floating point numbers***

- ◇ Efficient implementations of the simplex algorithm

  $\Rightarrow$ use ***floating point numbers***

➢ All the computations with floating point numbers require ***right corrections***

SAFE APPROXIMATIONS OF L_1

$$L_1(y, \alpha) \equiv y \geq 2\alpha x - \alpha^2$$

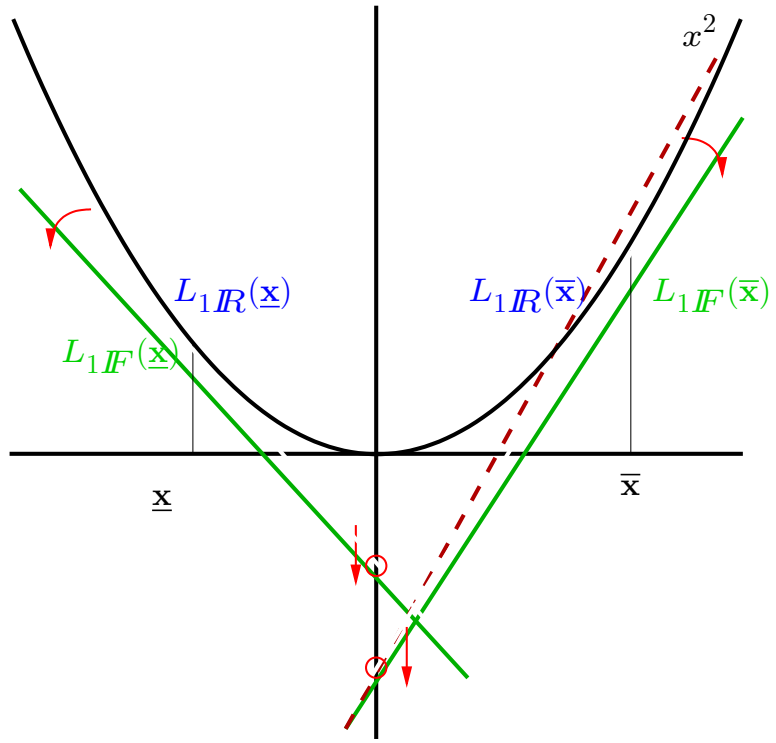
Effects of rounding:

- rounding of 2α

  $\Rightarrow$ rotation on y axis

- rounding of α^2

  $\Rightarrow$ translation on y axis



SAFE APPROXIMATIONS OF L_1

[$L_{1\mathbb{F}}(y, \alpha)$ approximations]

Let $\alpha \in \mathbb{F}$ and

$$\boldsymbol{L_{1\mathbb{F}}(y,\alpha)} \equiv \begin{cases} \boldsymbol{y - \lfloor 2\alpha \rfloor x + \lceil \alpha^2 \rceil \geq 0} & \text{iff } \boldsymbol{\alpha \geq 0} \\ \boldsymbol{y - \lceil 2\alpha \rceil x + \lceil \alpha^2 \rceil \geq 0} & \text{iff } \boldsymbol{\alpha < 0} \end{cases}$$

$\forall x \in \mathbf{x}$, and $y \in [0, max\{\underline{\mathbf{x}}^2, \overline{\mathbf{x}}^2\}]$,
if $L_1(y, \alpha)$ holds, then $L_{1\mathbb{F}}(y, \alpha)$ holds too

GENERALISATION TO N-ARY LINEARISATIONS

Let $\sum_{i=1}^{n} a_i x_i + b \geq 0$

then $\forall x_i \in \mathbf{x}_i$:

$$\sum_{i=1}^{n} \overline{a}_i x_i + \sup(\overline{b} + \sum_{i=1}^{n} \sup(\sup(\mathbf{a}_i \underline{x}_i) - \overline{a}_i \underline{x}_i)) \geq \sum_{i=1}^{n} a_i x_i + b \geq 0$$

CORRECTION OF THE SIMPLEX ALGORITHM

Consider the following LP :

$$\text{minimise } c^T x$$

$$\text{subject to } \underline{\mathbf{b}} \leq Ax \leq \overline{\mathbf{b}}$$

- Solution $=$ vector $x_{\mathbb{R}} \in \mathbb{R}^n$
- CPLEX computes a vector $x_{\mathbb{F}} \in \mathbb{F}^n \neq x_{\mathbb{R}}$.
- $x_{\mathbb{F}}$ is safe for the objective if $c^T x_{\mathbb{R}} \geq c^T x_{\mathbb{F}}$.

➢ Neumaier and Shcherbina

→ ***cheap method to obtain a rigorous bound of the objective***

→ ***rigorous computation of the certificate of infeasibility***

QUAD-OPT: A SAFE GLOBAL OPTIMISATION FRAMEWORK

FROM QUAD_SOLVER TO GLOBAL OPTIMISATION

QUAD_SOLVER ***offers a safe and efficient framework to solve global optimisation problems***

- Switch to global optimisation

  → ***Add constraint*** $\mathbf{f}^* = f(X)$

- Key of success

  → ***Adapt the search tree***

GENERAL SCHEMA OF THE QUAD_OPT SOLVER

Function `Quad_Opt(IN` $x, \epsilon_1, \epsilon_2$ `OUT` $\mathcal{S}, [\underline{\mathrm{f}}^*, \overline{\mathrm{f}}^*]$)

% $\mathcal{S}$: set of proved feasible points

% f_x denotes the set of possible values for f in x

$\mathcal{L} \leftarrow \{x\}$; $\mathcal{S} \leftarrow \emptyset$; $(\underline{\mathrm{f}}^*, \overline{\mathrm{f}}^*) \leftarrow (-\infty, +\infty)$;

while $w([\underline{\mathrm{f}}^*, \overline{\mathrm{f}}^*]) > \epsilon_1$ **and** $\mathcal{L} \neq \emptyset$ **do**

$x' \leftarrow x''$ such that $\underline{f}_{x''} = \min\{\underline{f}_{x''} : x'' \in \mathcal{L}\}$; $\mathcal{L} \leftarrow \mathcal{L} \backslash x'$;

$\overline{f}_{x'} \leftarrow \min(\overline{f}_{x'}, \overline{\mathrm{f}}^*)$; $x' \leftarrow$ **Quad_filtering**(x');

$\underline{f}_{x'} \leftarrow$ **Lower_Bound**(x'); $\underline{\mathrm{f}}^* \leftarrow \max(\underline{f}_{x'}, \underline{\mathrm{f}}^*)$

$(\overline{f}_{x'}, x_p,$**Proved**$) \leftarrow$**Upper_Bound**(x');

if Proved **then** $\mathcal{S} \leftarrow \mathcal{S} \cup \{x_p\}$; $\overline{\mathrm{f}}^* \leftarrow \min(\overline{f}_{x'}, \overline{\mathrm{f}}^*)$

if $w(x') > \epsilon_2$ **then** $(x'_1, x'_2) \leftarrow Split(x')$; $\mathcal{L} \leftarrow \mathcal{L} \cup \{x'_1, x'_2\}$;

endwhile

COMPUTING THE LOWER BOUND

➢ QUAD_SOLVER ***framework***

→ rigorous computation

➢ ***light version of*** QUAD_SOLVER

→ linear relaxations (RLT) + $2B$-consistency

Applied to $f(X) \wedge g_{1..m}(X)$

COMPUTING THE UPPER BOUND

- Use of **local methods** (COIN/IPOPT) to find potential feasible points

- Use of the **Newton algorithm** and **Hansen's heuristics** to prove the feasibility

CONCLUSION AND FUTURE WORKS

- ***Contribution:*** **a new safe and efficient framework**
- ***Details***

  . . . ***Next talk***