Interval Global Optimization: Techniques, Challenges, Related Problems, Future Directions

Martine Ceberio[1]

[1] Department of Computer Science
University of Texas at El Paso

October 14, 2014 / CPS Seminar

Outline

Target problem

Global Optimization with Intervals

Current work, Open Problems, ...

Future Directions

Global Optimization

What are we talking about?

- **Problem definition:**

$$\min_x f(x),\\ \text{where } x \in D \subseteq \mathbb{R}^n\\ \text{and } \forall i \in \{1, ..., p\},\ c_i : g_i(x) \bowtie 0 \text{ holds}\\ \bowtie \in \{\geq, \leq, =\}$$

- We are interested in global results: finding x^* such that:

$$f(x^*) \leq f(x),\ \forall x\ inD$$

- This proves to be a hard problems, to which we can add computational hardship... fighting rounding errors...

Global Optimization

What are we talking about?

- **Problem definition:**

$$\min_x f(x),$$
$$\text{where } x \in D \subseteq \mathbb{R}^n$$
$$\text{and } \forall i \in \{1, ..., p\},\ c_i : g_i(x) \bowtie 0 \text{ holds}$$
$$\bowtie \in \{\geq, \leq, =\}$$

- We are interested in global results: finding x^* such that:

$$f(x^*) \leq f(x),\ \forall x\ inD$$

- This proves to be a hard problems, to which we can add computational hardship... fighting rounding errors...

Global Optimization

What are we talking about?

- **Problem definition:**

$$\min_x f(x),\\ \quad \text{where } x \in D \subseteq \mathbb{R}^n\\ \quad \text{and } \forall i \in \{1, ..., p\},\ c_i : g_i(x) \bowtie 0 \text{ holds}\\ \qquad \bowtie \in \{\geq, \leq, =\}$$

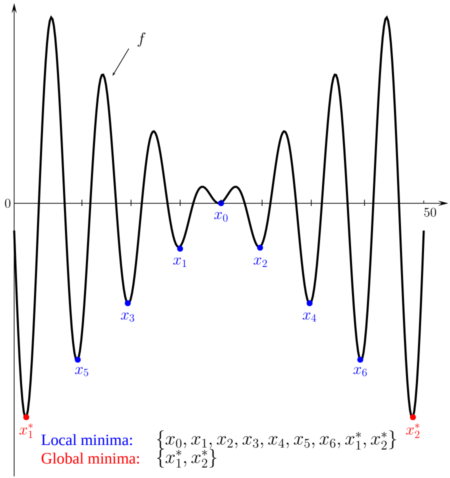
- We are interested in global results: finding x^* such that:

$$f(x^*) \leq f(x),\ \forall x\ inD$$

- This proves to be a hard problems, to which we can add computational hardship... fighting rounding errors...

Global Optimization

What are we talking about?



Rounding errors? Discrete line of reals?

- When do rounding errors occur? In a computer, only a finite amount of numbers are available...
- Discrete line? e.g., floating-point numbers
- What is the risk with rounding errors? with a discrete line of reals?
  - Well... rounding...
  - Missing a result? A solution that is not, e.g., a floating-point number
- How do we deal with that?

Rounding errors? Discrete line of reals?

- When do rounding errors occur? In a computer, only a finite amount of numbers are available...
- Discrete line? e.g., floating-point numbers
- What is the risk with rounding errors? with a discrete line of reals?
  - Well... rounding...
  - Missing a result? A solution that is not, e.g., a floating-point number
- How do we deal with that?

Rounding errors? Discrete line of reals?

- When do rounding errors occur? In a computer, only a finite amount of numbers are available...
- Discrete line? e.g., floating-point numbers
- What is the risk with rounding errors? with a discrete line of reals?
  - Well... rounding...
  - Missing a result? A solution that is not, e.g., a floating-point number
- How do we deal with that?

Rounding errors? Discrete line of reals?

- When do rounding errors occur? In a computer, only a finite amount of numbers are available...
- Discrete line? e.g., floating-point numbers
- What is the risk with rounding errors? with a discrete line of reals?
  - Well... rounding...
  - Missing a result? A solution that is not, e.g., a floating-point number
- How do we deal with that?

Rounding errors? Discrete line of reals?

- When do rounding errors occur? In a computer, only a finite amount of numbers are available...
- Discrete line? e.g., floating-point numbers
- What is the risk with rounding errors? with a discrete line of reals?
  - Well... rounding...
  - Missing a result? A solution that is not, e.g., a floating-point number
- How do we deal with that?

Rounding errors? Discrete line of reals?

- When do rounding errors occur? In a computer, only a finite amount of numbers are available...
- Discrete line? e.g., floating-point numbers
- What is the risk with rounding errors? with a discrete line of reals?
  - Well... rounding...
  - Missing a result? A solution that is not, e.g., a floating-point number
- How do we deal with that?

Interval Computations

As a mean to avoid the pitfalls mentioned earlier (1)

- Closed intervals of reals: $[a, b] = \{x \in \mathbb{R} \mid a \leq x \leq b\}$
- What do we do with intervals?
  - All otherwise real computations are conducted on intervals $\longrightarrow$ computations are guaranteed to be correct
  - Computations? following very well defined arithmetic rules:

$$I_1 \bowtie I_2 = \{z \in \mathbb{R} \mid \exists x \in I_1 \text{ and } \exists y \in I_2,\ z = x \bowtie y\}$$

    - $[a, b] + [c, d] = [a + c, b + d]$
    - $[a, b] - [c, d] = [a - d, b - c]$
    - $[a, b]/[c, d] = [a/d, b/c]$ where $0 \notin [c, d]$
    - Different case: $[2, 4]/[-1, 1] =]-\infty, -2] \cup [2, +\infty[$ not an interval!!!

Interval Computations

As a mean to avoid the pitfalls mentioned earlier (1)

- Closed intervals of reals: $[a,b] = \{x \in \mathbb{R} \mid a \leq x \leq b\}$
- What do we do with intervals?
  - All otherwise real computations are conducted on intervals $\longrightarrow$ computations are guaranteed to be correct
  - Computations? following very well defined arithmetic rules:

$$I_1 \bowtie I_2 = \{z \in \mathbb{R} \mid \exists x \in I_1 \text{ and } \exists y \in I_2,\ z = x \bowtie y\}$$

    - $[a,b] + [c,d] = [a+c, b+d]$
    - $[a,b] - [c,d] = [a-d, b-c]$
    - $[a,b]/[c,d] = [a/d, b/c]$ where $0 \notin [c,d]$
    - Different case: $[2,4]/[-1,1] =]-\infty,-2] \cup [2,+\infty[$: not an interval!!!

Interval Computations

As a mean to avoid the pitfalls mentioned earlier (1)

- Closed intervals of reals: $[a, b] = \{x \in \mathbb{R} \mid a \leq x \leq b\}$
- What do we do with intervals?
  - All otherwise real computations are conducted on intervals $\longrightarrow$ computations are guaranteed to be correct
  - Computations? following very well defined arithmetic rules:

$$I_1 \bowtie I_2 = \{z \in \mathbb{R} \mid \exists x \in I_1 \text{ and } \exists y \in I_2,\ z = x \bowtie y\}$$

    - $[a, b] + [c, d] = [a + c, b + d]$
    - $[a, b] - [c, d] = [a - d, b - c]$
    - $[a, b]/[c, d] = [a/d, b/c]$ where $0 \notin [c, d]$
    - Different case: $[2, 4]/[-1, 1] =]-\infty, -2] \cup [2, +\infty[$: not an interval!!!

Interval Computations

As a mean to avoid the pitfalls mentioned earlier (1)

- Closed intervals of reals: $[a, b] = \{x \in \mathbb{R} \mid a \leq x \leq b\}$
- What do we do with intervals?
  - All otherwise real computations are conducted on intervals $\longrightarrow$ computations are guaranteed to be correct
  - Computations? following very well defined arithmetic rules:

$$I_1 \bowtie I_2 = \{z \in \mathbb{R} \mid \exists x \in I_1 \text{ and } \exists y \in I_2,\ z = x \bowtie y\}$$

- $[a, b] + [c, d] = [a + c, b + d]$
- $[a, b] - [c, d] = [a - d, b - c]$
- $[a, b]/[c, d] = [a/d, b/c]$ where $0 \notin [c, d]$
- Different case: $[2, 4]/[-1, 1] =]-\infty, -2] \cup [2, +\infty[$: not an interval!!!

Interval Computations

As a mean to avoid the pitfalls mentioned earlier (1)

- Closed intervals of reals: $[a, b] = \{x \in \mathbb{R} \mid a \leq x \leq b\}$
- What do we do with intervals?
  - All otherwise real computations are conducted on intervals $\longrightarrow$ computations are guaranteed to be correct
  - Computations? following very well defined arithmetic rules:

$$I_1 \bowtie I_2 = \{z \in \mathbb{R} \mid \exists x \in I_1 \text{ and } \exists y \in I_2,\ z = x \bowtie y\}$$

    - $[a, b] + [c, d] = [a + c, b + d]$
    - $[a, b] - [c, d] = [a - d, b - c]$
    - $[a, b]/[c, d] = [a/d, b/c]$ where $0 \notin [c, d]$
    - Different case: $[2, 4]/[-1, 1] =]-\infty, -2] \cup [2, +\infty[$: not an interval!!!

Interval Computations

As a mean to avoid the pitfalls mentioned earlier (1)

- Closed intervals of reals: $[a, b] = \{x \in \mathbb{R} \mid a \leq x \leq b\}$
- What do we do with intervals?
  - All otherwise real computations are conducted on intervals $\longrightarrow$ computations are guaranteed to be correct
  - Computations? following very well defined arithmetic rules:

$$I_1 \bowtie I_2 = \{z \in \mathbb{R} \mid \exists x \in I_1 \text{ and } \exists y \in I_2,\ z = x \bowtie y\}$$

- $[a, b] + [c, d] = [a + c, b + d]$
- $[a, b] - [c, d] = [a - d, b - c]$
- $[a, b]/[c, d] = [a/d, b/c]$ where $0 \notin [c, d]$
- Different case: $[2, 4]/[-1, 1] =]-\infty, -2] \cup [2, +\infty[$: not an interval!!!

Interval Computations

As a mean to avoid the pitfalls mentioned earlier (1)

- Closed intervals of reals: $[a, b] = \{x \in \mathbb{R} \mid a \leq x \leq b\}$
- What do we do with intervals?
    - All otherwise real computations are conducted on intervals $\longrightarrow$ computations are guaranteed to be correct
    - Computations? following very well defined arithmetic rules:

$$I_1 \bowtie I_2 = \{z \in \mathbb{R} \mid \exists x \in I_1 \text{ and } \exists y \in I_2,\ z = x \bowtie y\}$$

- $[a, b] + [c, d] = [a + c, b + d]$
- $[a, b] - [c, d] = [a - d, b - c]$
- $[a, b]/[c, d] = [a/d, b/c]$ where $0 \notin [c, d]$
- Different case: $[2, 4]/[-1, 1] =]-\infty, -2] \cup [2, +\infty[$: not an interval!!!

Interval Computations

As a mean to avoid the pitfalls mentioned earlier (1)

- Closed intervals of reals: $[a, b] = \{x \in \mathbb{R} \mid a \leq x \leq b\}$
- What do we do with intervals?
    - All otherwise real computations are conducted on intervals $\longrightarrow$ computations are guaranteed to be correct
    - Computations? following very well defined arithmetic rules:

$$I_1 \bowtie I_2 = \{z \in \mathbb{R} \mid \exists x \in I_1 \text{ and } \exists y \in I_2,\ z = x \bowtie y\}$$

        - $[a, b] + [c, d] = [a + c, b + d]$
        - $[a, b] - [c, d] = [a - d, b - c]$
        - $[a, b]/[c, d] = [a/d, b/c]$ where $0 \notin [c, d]$
        - Different case: $[2, 4]/[-1, 1] =]-\infty, -2] \cup [2, +\infty[$: not an interval!!!

Interval Computations

As a mean to avoid the pitfalls mentioned earlier (2)

$$I_1 \bowtie I_2 = \Box\{z \in \mathbb{R} \mid \exists x \in I_1 \text{ and } \exists y \in I_2,\ z = x \bowtie y\}$$

- How do intervals actually solve the “computer" problem (rounding, discretization of reals)?
  - **“Floating-point" intervals:** set of intervals $[a, b]$ where both a and b are floating-point numbers $\longrightarrow$ no value is missed
  - **Outward rounding of intervals:** the $\Box$ is applied to all interval computations (not just division by 0) to enforce outward rounding

Interval Computations

As a mean to avoid the pitfalls mentioned earlier (2)

$$I_1 \bowtie I_2 = \square\{z \in \mathbb{R} \mid \exists x \in I_1 \text{ and } \exists y \in I_2,\ z = x \bowtie y\}$$

- How do intervals actually solve the “computer" problem (rounding, discretization of reals)?
  - **“Floating-point" intervals:** set of intervals $[a, b]$ where both a and b are floating-point numbers $\longrightarrow$ no value is missed
  - **Outward rounding of intervals:** the $\square$ is applied to all interval computations (not just division by 0) to enforce outward rounding

Interval Computations

As a mean to avoid the pitfalls mentioned earlier (2)

$$I_1 \bowtie I_2 = \Box\{z \in \mathbb{R} \mid \exists x \in I_1 \text{ and } \exists y \in I_2,\ z = x \bowtie y\}$$

- How do intervals actually solve the “computer" problem (rounding, discretization of reals)?
  - **“Floating-point" intervals:** set of intervals $[a, b]$ where both a and b are floating-point numbers $\longrightarrow$ no value is missed
  - **Outward rounding of intervals:** the $\Box$ is applied to all interval computations (not just division by 0) to enforce outward rounding

Intervals for Global Optimization?

Let’s backtrack to “simple" optimization for a moment

$$\min_{x \in D \subseteq \mathbb{R}} f(x)$$

- What’s f’s minimum?
  - We don’t know yet, but...
  - We know it is not outside of $f(D)$: not lower, not higher...
- What is $f(D)$?
  - The function f evaluated on D.
    Remember: now intervals are values that we can evaluate functions on.
  - E.g., $(x + y)([1, 2], [3.4]) = [4, 6]$
  - Not the exact range of f: instead, an outer estimation
    - E.g., $(x - x)([0, 1]) = [-1, 1]$, which encloses the actual range: 0, but $[-1, 1] \neq 0$

Intervals for Global Optimization?

Let's backtrack to “simple" optimization for a moment

$$\min_{x \in D \subseteq \mathbb{R}} f(x)$$

- What's f's minimum?
  - We don't know yet, but...
  - We know it is not outside of $f(D)$: not lower, not higher...
- What is $f(D)$?
  - The function f evaluated on D.
    Remember: now intervals are values that we can evaluate functions on.
  - E.g., $(x+y)([1,2],[3.4]) = [4,6]$
  - Not the exact range of f: instead, an outer estimation
    - E.g., $(x-x)([0,1]) = [-1,1]$, which encloses the actual range: 0, but $[-1,1] \neq 0$

Intervals for Global Optimization?

Let's backtrack to "simple" optimization for a moment

$$\min_{x \in D \subseteq \mathbb{R}} f(x)$$

- What's f's minimum?
  - We don't know yet, but...
  - We know it is not outside of $f(D)$: not lower, not higher...
- What is $f(D)$?
  - The function f evaluated on D.
    Remember: now intervals are values that we can evaluate functions on.
  - E.g., $(x+y)([1,2],[3,4]) = [4,6]$
  - Not the exact range of f: instead, an outer estimation
    - E.g., $(x-x)([0,1]) = [-1,1]$, which encloses the actual range: 0, but $[-1,1] \neq 0$

Intervals for Global Optimization?

Let’s backtrack to “simple" optimization for a moment

$$\min_{x \in D \subseteq \mathbb{R}} f(x)$$

- What’s f’s minimum?
  - We don’t know yet, but...
  - We know it is not outside of $f(D)$: not lower, not higher...
- What is $f(D)$?
  - The function f evaluated on D.
    Remember: now intervals are values that we can evaluate functions on.
  - E.g., $(x+y)([1,2],[3.4]) = [4,6]$
  - Not the exact range of f: instead, an outer estimation
    - E.g., $(x-x)([0,1]) = [-1,1]$, which encloses the actual range: 0, but $[-1,1] \neq 0$

Intervals for Global Optimization?

Let's backtrack to "simple" optimization for a moment

$$\min_{x \in D \subseteq \mathbb{R}} f(x)$$

- What's f's minimum?
  - We don't know yet, but...
  - We know it is not outside of $f(D)$: not lower, not higher...
- What is $f(D)$?
  - The function f evaluated on D.
    Remember: now intervals are values that we can evaluate functions on.
  - E.g., $(x + y)([1, 2], [3.4]) = [4, 6]$
  - Not the exact range of f: instead, an outer estimation
    - E.g., $(x - x)([0, 1]) = [-1, 1]$, which encloses the actual range: 0, but $[-1, 1] \neq 0$

Intervals for Global Optimization?

Let's backtrack to "simple" optimization for a moment

$$\min_{x \in D \subseteq \mathbb{R}} f(x)$$

- What's f's minimum?
  - We don't know yet, but...
  - We know it is not outside of $f(D)$: not lower, not higher...
- What is $f(D)$?
  - The function f evaluated on D.
    Remember: now intervals are values that we can evaluate functions on.
  - E.g., $(x + y)([1, 2], [3.4]) = [4, 6]$
  - Not the exact range of f: instead, an outer estimation
    - E.g., $(x - x)([0, 1]) = [-1, 1]$, which encloses the actual range: 0, but $[-1, 1] \neq 0$

Intervals for Global Optimization?

Let's backtrack to “simple" optimization for a moment

$$\min_{x \in D \subseteq \mathbb{R}} f(x)$$

- What's f's minimum?
  - We don't know yet, but...
  - We know it is not outside of $f(D)$: not lower, not higher...
- What is $f(D)$?
  - The function f evaluated on D.
    Remember: now intervals are values that we can evaluate functions on.
  - E.g., $(x + y)([1, 2], [3.4]) = [4, 6]$
  - Not the exact range of f: instead, an outer estimation
    - E.g., $(x - x)([0, 1]) = [-1, 1]$, which encloses the actual range: 0, but $[-1, 1] \neq 0$

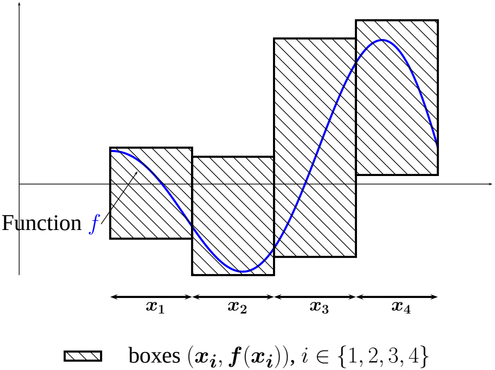
Intervals for Global Optimization?

Let's backtrack to “simple" optimization for a moment

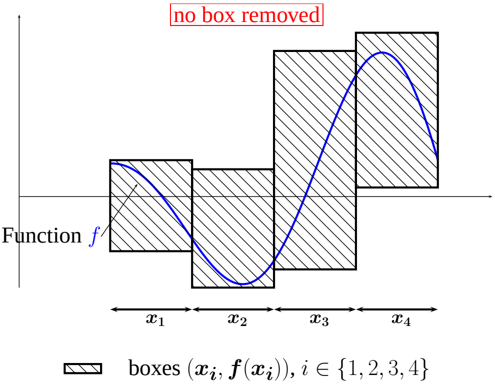
$$\min_{x \in D \subseteq \mathbb{R}} f(x)$$

- What's f's minimum?
  - We don't know yet, but...
  - We know it is not outside of $f(D)$: not lower, not higher...
- What is $f(D)$?
  - The function f evaluated on D.
    Remember: now intervals are values that we can evaluate functions on.
  - E.g., $(x + y)([1, 2], [3.4]) = [4, 6]$
  - Not the exact range of f: instead, an outer estimation
    - E.g., $(x - x)([0, 1]) = [-1, 1]$, which encloses the actual range: 0, but $[-1, 1] \neq 0$

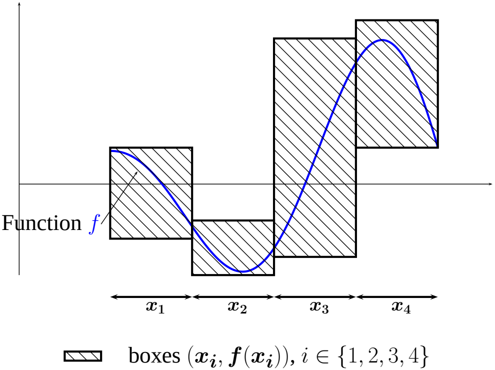
Simple Interval Global Optimization Framework



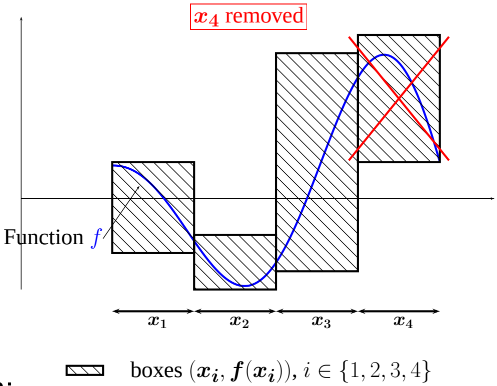
Simple Interval Global Optimization Framework



Simple Interval Global Optimization Framework

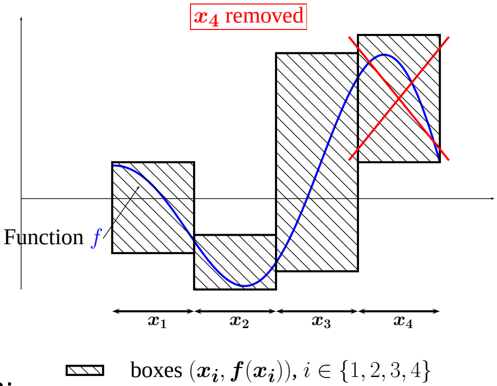


Simple Interval Global Optimization Framework



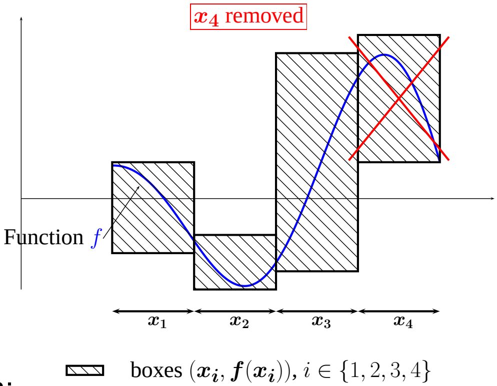
- Conclusion:
  - Intervals are useful
  - But: there is room for improvement: **Interval evaluations, symbolic expressions, etc.**

Simple Interval Global Optimization Framework



- Conclusion:
  - Intervals are useful
  - But: there is room for improvement: **Interval evaluations, symbolic expressions, etc.**

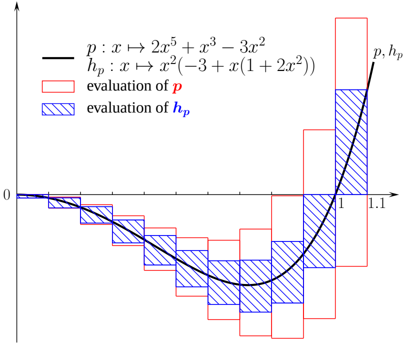
Simple Interval Global Optimization Framework



- Conclusion:
  - Intervals are useful
  - But: there is room for improvement: **Interval evaluations, symbolic expressions, etc.**

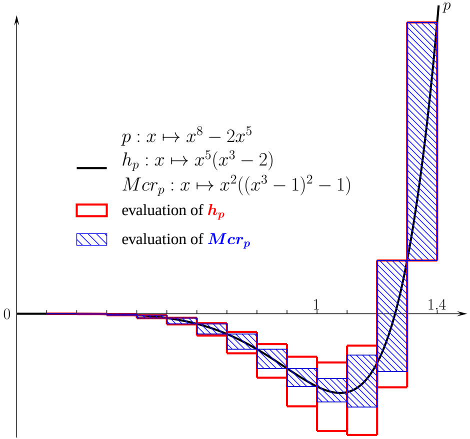
Simple Interval Global Optimization Framework

Example of interval evaluation differences:



Simple Interval Global Optimization Framework

Example of interval evaluation differences:



How to Improve this approach?

- So far: foundation of the use of intervals for both:
  - An exhaustive search of the domain
  - Reliable computations and reliable information about the expected minimum
  - Used in a **Branch-and-Bound framework**
- But we can do much better than that:
  - **Pruning**
  - ... and many other tricks to improve the pruning and discarding of subspaces
    - E.g., Evaluation of f at mid-point to lower the known upper bound of f
    - E.g., Using information (possibly intervals) about the derivatives

How to Improve this approach?

- So far: foundation of the use of intervals for both:
  - An exhaustive search of the domain
  - Reliable computations and reliable information about the expected minimum
  - Used in a **Branch-and-Bound framework**
- But we can do much better than that:
  - **Pruning**
  - ... and many other tricks to improve the pruning and discarding of subspaces
    - E.g., Evaluation of f at mid-point to lower the known upper bound of f
    - E.g., Using information (possibly intervals) about the derivatives

How to Improve this approach?

- So far: foundation of the use of intervals for both:
  - An exhaustive search of the domain
  - Reliable computations and reliable information about the expected minimum
  - Used in a **Branch-and-Bound framework**
- But we can do much better than that:
  - **Pruning**
  - ... and many other tricks to improve the pruning and discarding of subspaces
    - E.g.: Evaluation of f at mid-point to lower the known upper bound of f
    - E.g.: Using information (possibly intervals) about the derivatives

How to Improve this approach?

- So far: foundation of the use of intervals for both:
  - An exhaustive search of the domain
  - Reliable computations and reliable information about the expected minimum
  - Used in a **Branch-and-Bound framework**
- But we can do much better than that:
  - **Pruning**
  - ... and many other tricks to improve the pruning and discarding of subspaces
    - E.g., Evaluation of f at mid-point to lower the known upper bound of f
    - E.g., Using information (possibly intervals) about the derivatives

How to Improve this approach?

- So far: foundation of the use of intervals for both:
  - An exhaustive search of the domain
  - Reliable computations and reliable information about the expected minimum
  - Used in a **Branch-and-Bound framework**
- But we can do much better than that:
  - **Pruning**
  - ... and many other tricks to improve the pruning and discarding of subspaces
    - E.g., Evaluation of f at mid-point to lower the known upper bound of f
    - E.g., Using information (possibly intervals) about the derivatives

How to Improve this approach?

- So far: foundation of the use of intervals for both:
  - An exhaustive search of the domain
  - Reliable computations and reliable information about the expected minimum
  - Used in a **Branch-and-Bound framework**
- But we can do much better than that:
  - **Pruning**
  - ... and many other tricks to improve the pruning and discarding of subspaces
    - E.g., Evaluation of f at mid-point to lower the known upper bound of f
    - E.g., Using information (possibly intervals) about the derivatives

How to Improve this approach?

- So far: foundation of the use of intervals for both:
  - An exhaustive search of the domain
  - Reliable computations and reliable information about the expected minimum
  - Used in a **Branch-and-Bound framework**
- But we can do much better than that:
  - **Pruning**
  - ... and many other tricks to improve the pruning and discarding of subspaces
    - E.g., Evaluation of f at mid-point to lower the known upper bound of f
    - E.g., Using information (possibly intervals) about the derivatives

How to Improve this approach?

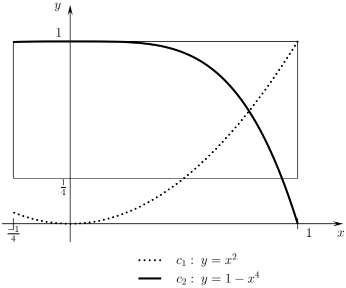
- So far: foundation of the use of intervals for both:
  - An exhaustive search of the domain
  - Reliable computations and reliable information about the expected minimum
  - Used in a **Branch-and-Bound framework**
- But we can do much better than that:
  - **Pruning**
  - ... and many other tricks to improve the pruning and discarding of subspaces
    - E.g., Evaluation of f at mid-point to lower the known upper bound of f
    - E.g., Using information (possibly intervals) about the derivatives

How to Improve this approach?

- So far: foundation of the use of intervals for both:
  - An exhaustive search of the domain
  - Reliable computations and reliable information about the expected minimum
  - Used in a **Branch-and-Bound framework**
- But we can do much better than that:
  - **Pruning**
  - ... and many other tricks to improve the pruning and discarding of subspaces
    - E.g., Evaluation of f at mid-point to lower the known upper bound of f
    - E.g., Using information (possibly intervals) about the derivatives

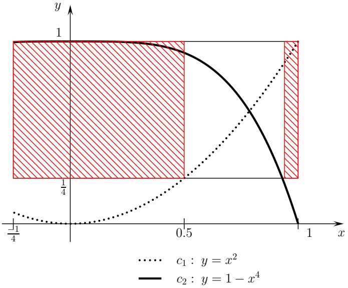
A Branch-and-Prune Approach...

... for Interval Global Optimization



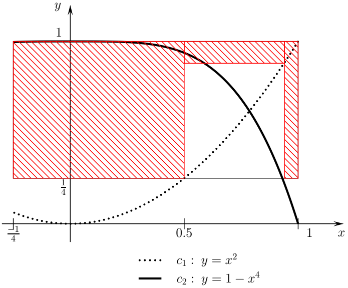
A Branch-and-Prune Approach...

... for Interval Global Optimization



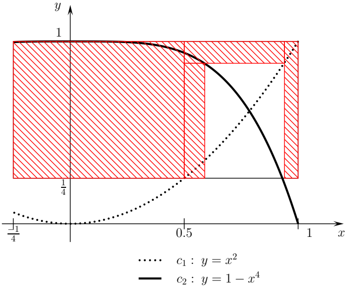
A Branch-and-Prune Approach...

... for Interval Global Optimization



A Branch-and-Prune Approach...

... for Interval Global Optimization



Constrained Global Optimization?

- Now we know the "*ingredients*" of Branch-and-Prune Optimization for **unconstrained** optimization
  - It is all a matter of combining them :)
- What about **constrained** optimization?
  - It is not so simple... Why?
  - Because the evaluation of f is not relevant until we know we are considering a feasible subspace
  - This makes shrinking the search space (= converging on solutions) much harder...
  - The key ingredient is: **Constraint Solving** and **Domain Contraction** / **Pruning**
  - With a hint of "*tricks*" (a.k.a., heuristics)

Constrained Global Optimization?

- Now we know the "*ingredients*" of Branch-and-Prune Optimization for **unconstrained** optimization
  - It is all a matter of combining them :)
- What about **constrained** optimization?
  - It is not so simple... Why?
  - Because the evaluation of f is not relevant until we know we are considering a feasible subspace
  - This makes shrinking the search space (= converging on solutions) much harder...
  - The key ingredient is: **Constraint Solving** and **Domain Contraction** / **Pruning**
  - With a hint of "*tricks*" (a.k.a., heuristics)

Constrained Global Optimization?

- Now we know the “*ingredients*" of Branch-and-Prune Optimization for **unconstrained** optimization
  - It is all a matter of combining them :)
- What about **constrained** optimization?
  - It is not so simple... Why?
  - Because the evaluation of f is not relevant until we know we are considering a feasible subspace
  - This makes shrinking the search space (= converging on solutions) much harder...
  - The key ingredient is: **Constraint Solving** and **Domain Contraction** / **Pruning**
  - With a hint of “*tricks*" (a.k.a., heuristics)

Constrained Global Optimization?

- Now we know the "*ingredients*" of Branch-and-Prune Optimization for **unconstrained** optimization
  - It is all a matter of combining them :)
- What about **constrained** optimization?
  - It is not so simple... Why?
  - Because the evaluation of *f* is not relevant until we know we are considering a feasible subspace
  - This makes shrinking the search space (= converging on solutions) much harder...
  - The key ingredient is: **Constraint Solving** and **Domain Contraction** / **Pruning**
  - With a hint of "*tricks*" (a.k.a., heuristics)

Constrained Global Optimization?

- Now we know the "*ingredients*" of Branch-and-Prune Optimization for **unconstrained** optimization
  - It is all a matter of combining them :)
- What about **constrained** optimization?
  - It is not so simple... Why?
  - Because the evaluation of f is not relevant until we know we are considering a feasible subspace
  - This makes shrinking the search space (= converging on solutions) much harder...
  - The key ingredient is: **Constraint Solving** and **Domain Contraction** / **Pruning**
  - With a hint of "*tricks*" (a.k.a., heuristics)

Constrained Global Optimization?

- Now we know the "*ingredients*" of Branch-and-Prune Optimization for **unconstrained** optimization
  - It is all a matter of combining them :)
- What about **constrained** optimization?
  - It is not so simple... Why?
  - Because the evaluation of f is not relevant until we know we are considering a feasible subspace
  - This makes shrinking the search space (= converging on solutions) much harder...
  - The key ingredient is: **Constraint Solving** and **Domain Contraction** / **Pruning**
  - With a hint of "*tricks*" (a.k.a., heuristics)

Constrained Global Optimization?

- Now we know the "*ingredients*" of Branch-and-Prune Optimization for **unconstrained** optimization
  - It is all a matter of combining them :)
- What about **constrained** optimization?
  - It is not so simple... Why?
  - Because the evaluation of f is not relevant until we know we are considering a feasible subspace
  - This makes shrinking the search space (= converging on solutions) much harder...
  - The key ingredient is: **Constraint Solving** and **Domain Contraction / Pruning**
  - With a hint of "*tricks*" (a.k.a., heuristics)

Constrained Global Optimization?

- Now we know the “*ingredients*" of Branch-and-Prune Optimization for **unconstrained** optimization
  - It is all a matter of combining them :)
- What about **constrained** optimization?
  - It is not so simple... Why?
  - Because the evaluation of f is not relevant until we know we are considering a feasible subspace
  - This makes shrinking the search space (= converging on solutions) much harder...
  - The key ingredient is: **Constraint Solving** and **Domain Contraction** / **Pruning**
  - With a hint of “*tricks*" (a.k.a., heuristics)

Current Work and Considerations

- Constraint and optimization **solver**
  - Combining the above-mentioned *ingredients*
- **Larger-scale** optimization
  - Very challenging for interval approaches
- **Dynamic** systems
  - To bridge the gap between what constraints and optimization solver can do and more real applications
  - Somewhere in between current static approaches and large systems' own challenges

Current Work and Considerations

- Constraint and optimization **solver**
  - Combining the above-mentioned *ingredients*
- **Larger-scale** optimization
  - Very challenging for interval approaches
- **Dynamic** systems
  - To bridge the gap between what constraints and optimization solver can do and more real applications
  - Somewhere in between current static approaches and large systems' own challenges

Current Work and Considerations

- Constraint and optimization **solver**
  - Combining the above-mentioned *ingredients*
- **Larger-scale** optimization
  - Very challenging for interval approaches
- **Dynamic** systems
  - To bridge the gap between what constraints and optimization solver can do and more real applications
  - Somewhere in between current static approaches and large systems' own challenges

Current Work and Considerations

- Constraint and optimization **solver**
  - Combining the above-mentioned *ingredients*
- **Larger-scale** optimization
  - Very challenging for interval approaches
- **Dynamic** systems
  - To bridge the gap between what constraints and optimization solver can do and more real applications
  - Somewhere in between current static approaches and large systems' own challenges

Current Work and Considerations

- Constraint and optimization **solver**
  - Combining the above-mentioned *ingredients*
- **Larger-scale** optimization
  - Very challenging for interval approaches
- **Dynamic** systems
  - To bridge the gap between what constraints and optimization solver can do and more real applications
  - Somewhere in between current static approaches and large systems' own challenges

Current Work and Considerations

- Constraint and optimization **solver**
  - Combining the above-mentioned *ingredients*
- **Larger-scale** optimization
  - Very challenging for interval approaches
- **Dynamic** systems
  - To bridge the gap between what constraints and optimization solver can do and more real applications
  - Somewhere in between current static approaches and large systems' own challenges

Current Work and Considerations

- Constraint and optimization **solver**
  - Combining the above-mentioned *ingredients*
- **Larger-scale** optimization
  - Very challenging for interval approaches
- **Dynamic** systems
  - To bridge the gap between what constraints and optimization solver can do and more real applications
  - Somewhere in between current static approaches and large systems' own challenges

Some Applications

- Optimization as a model to **solve decision-making prediction problems**
  - How to best predict decisions?
  - Get prior decisions (very close to Machine Learning)
  - Pick a decision model and **fit** the prior data to it
  - = Optimization: there is no perfect fit because prior decisions are never perfect, so we look for the best fit instead (the one that deviates the least from prior decisions)
- Pet problem: generating **t-wise covering test suites**
  - An optimization problem
  - Challenge = modeling
  - Once it is modeled, it should be pretty reasonable to solve

Some Applications

- Optimization as a model to **solve decision-making prediction problems**
  - How to best predict decisions?
  - Get prior decisions (very close to Machine Learning)
  - Pick a decision model and **fit** the prior data to it
  - = Optimization: there is no perfect fit because prior decisions are never perfect, so we look for the best fit instead (the one that deviates the least from prior decisions)
- Pet problem: generating **t-wise covering test suites**
  - An optimization problem
  - Challenge = modeling
  - Once it is modeled, it should be pretty reasonable to solve

Some Applications

- Optimization as a model to **solve decision-making prediction problems**
  - How to best predict decisions?
  - Get prior decisions (very close to Machine Learning)
  - Pick a decision model and **fit** the prior data to it
  - = Optimization: there is no perfect fit because prior decisions are never perfect, so we look for the best fit instead (the one that deviates the least from prior decisions)
- Pet problem: generating **t-wise covering test suites**
  - An optimization problem
  - Challenge = modeling
  - Once it is modeled, it should be pretty reasonable to solve

Some Applications

- Optimization as a model to **solve decision-making prediction problems**
  - How to best predict decisions?
  - Get prior decisions (very close to Machine Learning)
  - Pick a decision model and **fit** the prior data to it
  - = Optimization: there is no perfect fit because prior decisions are never perfect, so we look for the best fit instead (the one that deviates the least from prior decisions)
- Pet problem: generating **t-wise covering test suites**
  - An optimization problem
  - Challenge = modeling
  - Once it is modeled, it should be pretty reasonable to solve

Some Applications

- Optimization as a model to **solve decision-making prediction problems**
  - How to best predict decisions?
  - Get prior decisions (very close to Machine Learning)
  - Pick a decision model and **fit** the prior data to it
  - = Optimization: there is no perfect fit because prior decisions are never perfect, so we look for the best fit instead (the one that deviates the least from prior decisions)
- Pet problem: generating **t-wise covering test suites**
  - An optimization problem
  - Challenge = modeling
  - Once it is modeled, it should be pretty reasonable to solve

Some Applications

- Optimization as a model to **solve decision-making prediction problems**
  - How to best predict decisions?
  - Get prior decisions (very close to Machine Learning)
  - Pick a decision model and **fit** the prior data to it
  - = Optimization: there is no perfect fit because prior decisions are never perfect, so we look for the best fit instead (the one that deviates the least from prior decisions)
- Pet problem: generating **t-wise covering test suites**
  - An optimization problem
  - Challenge = modeling
  - Once it is modeled, it should be pretty reasonable to solve

Some Applications

- Optimization as a model to **solve decision-making prediction problems**
  - How to best predict decisions?
  - Get prior decisions (very close to Machine Learning)
  - Pick a decision model and **fit** the prior data to it
  - = Optimization: there is no perfect fit because prior decisions are never perfect, so we look for the best fit instead (the one that deviates the least from prior decisions)
- Pet problem: generating **t-wise covering test suites**
  - An optimization problem
  - Challenge = modeling
  - Once it is modeled, it should be pretty reasonable to solve

Some Applications

- Optimization as a model to **solve decision-making prediction problems**
  - How to best predict decisions?
  - Get prior decisions (very close to Machine Learning)
  - Pick a decision model and **fit** the prior data to it
  - = Optimization: there is no perfect fit because prior decisions are never perfect, so we look for the best fit instead (the one that deviates the least from prior decisions)
- Pet problem: generating **t-wise covering test suites**
  - An optimization problem
  - Challenge = modeling
  - Once it is modeled, it should be pretty reasonable to solve

Some Applications

- Optimization as a model to **solve decision-making prediction problems**
  - How to best predict decisions?
  - Get prior decisions (very close to Machine Learning)
  - Pick a decision model and **fit** the prior data to it
  - = Optimization: there is no perfect fit because prior decisions are never perfect, so we look for the best fit instead (the one that deviates the least from prior decisions)
- Pet problem: generating **t-wise covering test suites**
  - An optimization problem
  - Challenge = modeling
  - Once it is modeled, it should be pretty reasonable to solve

Future Directions

Summary and Future Directions

- What we have seen today:
  - Ways (Intervals) to cope with computer limitations
  - Limitations of intervals...
- **Conclusion:** there is still a lot to be done
- **Future directions** will have to be a combination of:
  - Methods ignoring computer limitations (for speed)... and
  - Intervals (for reliability)

Summary and Future Directions

- What we have seen today:
  - Ways (Intervals) to cope with computer limitations
  - Limitations of intervals...
- **Conclusion:** there is still a lot to be done

- **Future directions** will have to be a combination of:
  - Methods ignoring computer limitations (for speed)... and
  - Intervals (for reliability)

Summary and Future Directions

- What we have seen today:
  - Ways (Intervals) to cope with computer limitations
  - Limitations of intervals...
- **Conclusion:** there is still a lot to be done
- **Future directions** will have to be a combination of:
  - Methods ignoring computer limitations (for speed)... and
  - Intervals (for reliability)

Summary and Future Directions

- What we have seen today:
  - Ways (Intervals) to cope with computer limitations
  - Limitations of intervals...
- **Conclusion:** there is still a lot to be done

- **Future directions** will have to be a combination of:
  - Methods ignoring computer limitations (for speed)... and
  - Intervals (for reliability)

Summary and Future Directions

- What we have seen today:
  - Ways (Intervals) to cope with computer limitations
  - Limitations of intervals...
- **Conclusion:** there is still a lot to be done
- **Future directions** will have to be a combination of:
  - Methods ignoring computer limitations (for speed)... and
  - Intervals (for reliability)

Summary and Future Directions

- What we have seen today:
  - Ways (Intervals) to cope with computer limitations
  - Limitations of intervals...
- **Conclusion:** there is still a lot to be done

- **Future directions** will have to be a combination of:
  - Methods ignoring computer limitations (for speed)... and
  - Intervals (for reliability)

Summary and Future Directions

- What we have seen today:
  - Ways (Intervals) to cope with computer limitations
  - Limitations of intervals...
- **Conclusion:** there is still a lot to be done

- **Future directions** will have to be a combination of:
  - Methods ignoring computer limitations (for speed)... and
  - Intervals (for reliability)

Thank you for your attention!

Feel free to contact me:
mceberio@utep.edu