Constraint Optimization: From Efficient Computation of What Can Be Achieved to Efficient Computation of How to Achieve The Corresponding Optimum

Ali Jalal-Kamali, Martine Ceberio, and Vladik Kreinovich
Department of Computer Science, University of Texas at El Paso
El Paso, TX 79968, USA
ajalalkamai@miners.utep.edu, mceberio@utep.edu, vladik@utep.edu
1. Need for Optimization: General Reminder

- In many practical situations, we need to select the best alternative:
  - a location of a plant,
  - values of the control to apply to a system, etc.
- Let $n$ be the total number of parameters $x_1, \ldots, x_n$ needed to uniquely determine an alternative.
- For each parameter $x_i$, we know the range $x_i = [\underline{x}_i, \overline{x}_i]$ of its possible values.
- The “best” alternative is defined as the one for which an appropriate objective function $f(x_1, \ldots, x_n)$ is max.
- It is reasonable to assume that the objective function is feasibly computable.
- Then, the problem is to find the best values $x_1, \ldots, x_n$ for which $f(x_1, \ldots, x_n) \to \text{max}$. 
2. First Step: Computing the Largest Possible Value of the Objective Function

- It often makes sense to first check what we can, in principle, achieve within the given setting.
- Example: if min possible pollution of a coal-burning steam engine is too high, look for different engines.
- So, we need to compute the max $\bar{y}$ (or min $\underline{y}$) of a given function $f(x_1, \ldots, x_n)$ over given intervals $x_i$.
- The problem of computing the range $[\underline{y}, \bar{y}]$ of the function under $x_i \in x_i$ is known as interval computations.
- The values $\underline{y}$ and $\bar{y}$ are, in general, irrational and thus, cannot be exactly computer represented.
- So, what we need is, given any rational number $\varepsilon > 0$, compute $\underline{r}$ and $\bar{r}$ s.t. $|\underline{r} - \underline{y}| \leq \varepsilon$ and $|\bar{r} - \bar{y}| \leq \varepsilon$. 
3. Interval Computation Is, in General, NP-hard

- It is known that in general, the problem of computing the corresponding range is NP-hard.

- This means, crudely speaking, that it is not possible to have:
  - a feasible algorithm
  - that would always compute the desired range.

- Because of this, it is important to find:
  - practically useful classes of problems
  - for which it is feasibly possible to compute this range.

- Many such classes are known.
4. Formulation of the Problem

- In practice, we often have additional constraints of equality or inequality type.
- In such situations, it is necessary to restrict ourselves only to values \((x_1, \ldots, x_n)\) which satisfy these constraints.
- Once we know the largest value, we need to find the values \(x_1, \ldots, x_n\) that lead to this largest value.
- At present:
  - once we have developed an algorithm for computing the max of a given function \(f(x_1, \ldots, x_n)\),
  - we need to develop a second algorithm – for locating this largest value.
- In this talk, we describe a general technique for generating the second algorithm once the first one is known.
5. **Main Result**

- Let $\mathcal{F}$ be a class of functions, and let $\mathcal{C}$ be a class of constraints.

- We consider two problems, in both we are given:
  - a f-n $f(x_1, \ldots, x_n) \in \mathcal{F}$ and constraints $C \in \mathcal{C}$,
  - rational-valued intervals $[x_1, \overline{x_1}], \ldots, [x_n, \overline{x_n}]$, and
  - a rational number $\varepsilon > 0$,

- **Problem 1**: compute rational values $\underline{r}$ and $\overline{r}$ which are $\varepsilon$-close to the endpoints $\underline{y}$ and $\overline{y}$ of the range
  
  $$[\underline{y}, \overline{y}] = \{f(x_1, \ldots, x_n) : x_i \in [\underline{x_i}, \overline{x_i}], (x_1, \ldots, x_n) \in C\}.$$

- **Problem 2**: compute rational $r_1, \ldots, r_n$ s.t. $f(x_1, \ldots, x_n) \geq \overline{y} - \varepsilon$ for some $x_i$ which are $\varepsilon$-close to $r_i$ and satisfy $C$.

- **Main Result**: once we have a feasible algorithm for solving Problem 1, we can feasible solve Problem 2.
6. Additional Result

• Reminder: we compute rat. $r_1, \ldots, r_n$ s.t. $f(x_1, \ldots, x_n) \geq \bar{y} - \varepsilon$ for some $x_i$ which are $\varepsilon$-close to $r_i$ and satisfy $C$.

• Important case:
  
  – there are no additional constraints, only interval bounds $\underline{x}_i \leq x_i \leq \bar{x}_i$, and
  
  – we can also feasibly compute the bound $M$ on all partial derivatives of a function $f$.

• In this case, we can also feasibly produce:
  
  – given a rational number $\varepsilon > 0$,
  
  – rational values $r_1, \ldots, r_n$ for which already for these values $r_i$, we have
  
  $$f(r_1, \ldots, r_n) \geq \bar{y} - \varepsilon.$$
7. Comparison to Interval Computations

- Locating maxima is one of the main applications of interval computations in optimization; main idea:
  - use interval computations to find the enclosure of a function on subboxes;
  - compute values in the subboxes’ midpoints;
  - compute maximum-so-far as the maximum of all midpoint values;
  - and then dismiss the subboxes for which the upper bound is smaller than the maximum-so-far;
  - bisect remaining boxes.

- What is new:
  - the above idea can take exponential time – by requiring us to consider $2^n$ sub-boxes, while
  - the computation time for our algorithm is always feasible (polynomial).
8. Constraints-Based Intuitive Explanation of Our Result

- There are two different constraint problems:
  - constraint satisfaction – finding values that satisfy given constraints, and
  - constraint optimization – among all values that satisfy constraints, find the ones for which \( f \to \text{max} \).

- It is clear that constraint optimization is harder than constraint satisfaction.

- Once we know \( \bar{y} = \max f \), locating max becomes a constraint satisfaction problem: just add a constraint
  \[
  f(x_1, \ldots, x_n) \geq \bar{y} - \varepsilon.
  \]

- Thus, to locate the maximum, it is sufficient to solve an easier-to-solve constraint satisfaction problem.
9. Algorithm: General Overview

- At each stage of this algorithm, we will have a box $B_k$.
- We start with the original box $B_0 = B$.
- Then, we repeatedly decrease the $x_1$-size of this box in half until its size is smaller than or equal to $2\varepsilon$.
- After this, we decrease the $x_2$-size of this box in half, etc., until all $n$ sizes are bounded by $2\varepsilon$.
- For each side, we start with the interval $[x_i, \bar{x}_i]$ of width $w_i = \bar{x}_i - x_i$.
- After $s_i$ bisection steps, the width decreases to $w_i \cdot 2^{-s_i}$.
- One can see that we need $\lceil \ln \left( \frac{w_i}{2\varepsilon} \right) \rceil$ steps to reach the desired size ($\leq 2\varepsilon$) of the $i$-th side.
- Overall, we need $s \overset{\text{def}}{=} \sum_{i=1}^{n} \ln \left( \left\lceil \frac{w_i}{2\varepsilon} \right\rceil \right)$ bisection steps.
10. A Bisection Step

- Start with the box
  \[ B_k = \ldots \times [b_{i-1}, \bar{b}_{i-1}] \times [b_i, \bar{b}_i] \times [b_{i+1}, \bar{b}_{i+1}] \times \ldots \]

- Divide the \( i \)-th side into equal intervals \([b_i, m_i]\) and \([m_i, \bar{b}_i]\), with \( m_i = \frac{b_i + \bar{b}_i}{2} \). This divides \( B_k \) into:
  \[ B'_k = \ldots \times [b_{i-1}, \bar{b}_{i-1}] \times [b_i, m_i] \times [b_{i+1}, \bar{b}_{i+1}] \times \ldots \quad \text{and} \]
  \[ B''_k = \ldots \times [b_{i-1}, \bar{b}_{i-1}] \times [m_i, \bar{b}_i] \times [b_{i+1}, \bar{b}_{i+1}] \times \ldots \]

- We apply the original range estimation algorithm to \( B'_k \) and \( B''_k \) and get \( \bar{r}'_k \) and \( \bar{r}''_k \) s.t.
  \[ |\bar{r}'_k - \max\{f(x) : x \in B'_k\}| \leq \frac{\varepsilon}{2s}, \quad |\bar{r}''_k - \max\{f(x) : x \in B''_k\}| \leq \frac{\varepsilon}{2s}. \]

- If \( \bar{r}'_k \geq \bar{r}''_k \), choose \( B_{k+1} = B'_k \), else choose \( B_{k+1} = B''_k \).

- At the end, we return the coordinates of the midpoint of the final box \( B_s \) as the desired values \( r_1, \ldots, r_n \).
11. Proof that Our Algorithm Is Feasible

- The number of steps $s$ feasibly (polynomially) depends on the size of the input.
- The range estimation algorithm that we use on each step is also polynomial-time.
- Thus, all we do is repeat a polynomial-time algorithm polynomially many times.
- The computation time of the resulting algorithm is:
  - bounded by the product of the two corresponding polynomials and
  - is, thus, itself polynomial.
- Hence, our algorithm is indeed feasible.
12. Proof that Our Algorithm is Correct

- Let $\bar{y}_k$ denote the (constraint) maximum of the function $f(x_1, \ldots, x_n)$ over the box $B_k$.

- We will prove, by induction, that for each box $B_k$, we have $\bar{y}_k \geq \bar{y} - \frac{k}{s} \cdot \varepsilon$.

- Then, after all $s$ steps, we will be able to conclude that $\bar{y}_s \geq \bar{y} - \varepsilon$.

- By definition of $\bar{y}_s$, there exist a point $(x_1, \ldots, x_n) \in B_s$ which satisfies the constraints and at which $f(x_1, \ldots, x_n) = \bar{y}_s \geq \bar{y} - \varepsilon$.

- Since the box is of width $\leq 2\varepsilon$ in all directions, each value $x_i$ is $\varepsilon$-close to the midpoint $r_i$.

- So, to prove correctness, it is sufficient to prove that $\bar{y}_k \geq \bar{y} - \frac{k}{s} \cdot \varepsilon$. 
13. Proof: Details

- **Induction base:** for \( k = 0 \), \( B_0 \) is the original box and thus, the max \( \bar{y}_0 \) over \( B_0 \) is equal to \( \bar{y} \).

- **Induction step:** assume that \( \bar{y}_k \geq \bar{y} - \frac{k}{s} \cdot \varepsilon \).

- Let us show that this inequality holds for \( k + 1 \).

- Since \( B_k = B'_k \cup B''_k \), the max \( \bar{y}_k \) of \( f \) over \( B_k \) is equal to the largest of the maxima \( \bar{y}'_k \) and \( \bar{y}''_k \) over \( B'_k \), \( B''_k \):
  \[
  \bar{y}_k = \max(\bar{y}'_k, \bar{y}''_k).
  \]

- For computed approximate maxima \( \bar{r}'_k \) and \( \bar{r}''_k \), we have
  \[
  \bar{r}'_k \geq \bar{y}'_k - \frac{\varepsilon}{2s} \quad \text{and} \quad \bar{r}''_k \geq \bar{y}''_k - \frac{\varepsilon}{2s}.
  \]

- Thus, \( \max(\bar{r}'_k, \bar{r}''_k) \geq \max(\bar{y}'_k, \bar{y}''_k) - \frac{\varepsilon}{2s} = \bar{y}_k - \frac{\varepsilon}{2s} \).

- In our algorithm, we select \( B_{k+1} \) for which the maximum is the largest, i.e., for which \( \bar{r}_{k+1} = \max(\bar{r}'_k, \bar{r}''_k) \).
14. Proof (cont-d)

- **Reminder:** we proved that
  \[
  \bar{r}_{k+1} = \max(\bar{r}'_k, \bar{r}''_k) \text{ and } \max(\bar{r}'_k, \bar{r}''_k) \geq \bar{y}_k - \frac{\varepsilon}{2s}.
  \]

- Thus, we conclude that \(\bar{r}_{k+1} \geq \bar{y}_k - \frac{\varepsilon}{2s}\).

- Since \(\bar{y}_{k+1}\) is \(\frac{\varepsilon}{2s}\)-close to \(\bar{r}_{k+1}\), we get
  \[
  \bar{y}_{k+1} \geq \bar{r}_{k+1} - \frac{\varepsilon}{2s} \geq \left(\bar{y}_k - \frac{\varepsilon}{2s}\right) - \frac{\varepsilon}{2s} = \bar{y}_k - \frac{\varepsilon}{s}.
  \]

- So, from \(\bar{y}_k \geq y - \frac{k}{s} \cdot \varepsilon\), we can now conclude that
  \[
  \bar{y}_{k+1} \geq \bar{y}_k - \frac{\varepsilon}{s} \geq \left(y - \frac{k}{s} \cdot \varepsilon\right) - \frac{\varepsilon}{s} = y - \frac{k + 1}{s} \cdot \varepsilon.
  \]

- The inequality is proven, and so is the algorithm’s correctness.
15. What If We Know the Bound $\left| \frac{\partial f}{\partial x_i} \right| \leq M$ on all the Partial Derivatives

- In this case, we have

$$|f(r_1, \ldots, r_n) - f(x_1, \ldots, x_n)| \leq \sum_{i=1}^{n} \left| \frac{\partial f}{\partial x_i} \right| \cdot |x_i - r_i| \leq n \cdot M \cdot \varepsilon.$$

- We know that $f(x_1, \ldots, x_n) \geq \bar{y} - \varepsilon$.

- Therefore, we conclude that

$$f(r_1, \ldots, r_n) \geq f(x_1, \ldots, x_n) - n \cdot M \cdot \varepsilon \geq \bar{y} - (\varepsilon + n \cdot M \cdot \varepsilon).$$

- So:

  - if we want to find the values $r_1, \ldots, r_n$ for which $f(r_1, \ldots, r_n) \geq \bar{y} - \eta$,
  - it is sufficient to apply the above algorithm with

$$\varepsilon = \frac{\eta}{1 + n \cdot M};$$

then, $\varepsilon + n \cdot M \cdot \varepsilon = \eta$ and

$$f(r_1, \ldots, r_n) \geq \bar{y} - \eta.$$
16. Acknowledgments

This work was supported in part:

- by the National Science Foundation grants HRD-0734825 and DUE-0926721, and

- by Grant 1 T36 GM078000-01 from the National Institutes of Health.