If We Take Into Account that Constraints Are Soft, Then Processing Constraints Becomes Algorithmically Solvable

Quentin Brefort and Luc Jaulin
ENSTA-Bretagne, LabSTICC, IHSEV, OSM
2 rue François Verny, 29806 Brest, France
Quentin.Brefort@ensta-bretagne.org, luc.jaulin@ensta-bretagne.fr

Martine Ceberio and Vladik Kreinovich
Department of Computer Science, University of Texas at El Paso
El Paso, Texas 79968, USA, mceberio@utep.edu, vladik@utep.edu
1. Main Objectives of Science and Engineering

- The main objective of science is to describe the world – and to predict its future state.
- The main objective of engineering is to select actions and designs \( a \) which lead to the desired future state.
- Information about the physical world usually comes in terms of the numerical values \( x_i \) of physical quantities.
- To describe the world means to find the values
  \[
  x = (x_1, \ldots, x_n).
  \]
- Each measurement constraints values \( x \) to a set \( X \).
- The relation between current state \( x \) and future state \( y \) is often also only approximately known, as the set
  \[
  R \overset{\text{def}}{=} \{(x, y) : (x, y) \text{ is possible}\} \subseteq X \times Y.
  \]
2. First Computational Problem

• We need to describe the results of each measurement.
• It is often difficult to directly measure the values $x_i$.
• Instead, we measure an easier-to-measure quantity $y$ related to $x_i$ by a known formula $y = f(x_1, \ldots, x_n)$.
• We usually know the upper bound $\Delta$ on the measurement accuracy.
• So, once we know the measurement result $\tilde{y}$, we can conclude that the actual value $y$ is in

$$[\underline{y}, \overline{y}] \overset{\text{def}}{=} [\tilde{y} - \Delta, \tilde{y} + \Delta].$$

• Given: a computable function $f(x_1, \ldots, x_n)$ and computable values $\underline{y}$ and $\overline{y}$.
• We need to describe: the set $\{x : \underline{y} \leq f(x) \leq \overline{y}\}$ of all the states $x$ consistent with this measurement.
3. **Second Computational Problem**

- We need to be able to *combine* the results of several measurements.

- *We know:* the set $S_1$ of all the tuples which are consistent with the first measurement.

- *We know:* the set $S_2$ of all the tuples which are consistent with the second measurement.

- ... 

- *We know:* the set $S_m$ of all the tuples which are consistent with the $m$-th measurement.

- *We need to describe:* the set of all the tuples $x$ consistent with the results of all available measurements:

$$S = S_1 \cap \ldots \cap S_m.$$
4. Third Computational Problem

• Then, we need to be able to predict the future state.

• *We know:* the set $S \subseteq X$ of possible states of the world.

• *We know:* the relation $R \subseteq X \times Y$ that describes the system’s dynamics.

• *We need to describe:* the set of possible $Y$ of possible future states:

$$Y = \{y : (x, y) \in R \text{ for some } x \in S\}.$$

• In mathematical terms, $Y$ is known as a *composition*:

$$Y = R \circ S.$$
5. Fourth Computational Problem

- Finally, we need to describe the set of possible actions.
- *We know:* the set $S \subseteq X$ of possible states.
- *We know:* the set $A$ of possible actions.
- *We know:* the set $D \subseteq Y$ of desired future states.
- *We know:* a computable function $f(x, a)$ that describes how the future state depends:
  - on the initial state $x$, and
  - on the action $a$.
- *We need to describe:* the set of all actions $a$ that lead to the desired goal
  
  $$\{ a \in A : f(x, a) \in D \text{ for all } x \in S \}.$$
6. What We Do in This Paper

- All four problems are *algorithmically unsolvable* if we assume that all the constraints are known *exactly*:
  - that we know the exact bounds on the measurement error,
  - that we know the exact relation between the present and future states, etc.
- However, in reality, these constraints are only know *approximately*.
- In other words, the constraints are actually *soft*.
- We show that:
  - if we take this softness into account,
  - then all four fundamental problems become algorithmically solvable.
7. What Is Computable: A Reminder

- A real number \( x \) is computable if there exists an algorithm that, given \( k \in \mathbb{N} \), returns a rational \( r_k \) for which
  \[ |x - r_k| \leq 2^{-k}. \]

- A function \( f : X \to Y \) is called computable if there exist algorithms \( A_1 \) and \( A_2 \):
  - \( A_1 \), given a rational tuple \( r \in X \) and \( \ell \in \mathbb{N} \), computes a \( 2^{-\ell} \)-approximation to \( f(r) \);
  - \( A_2 \), given \( \ell \in \mathbb{N} \), generates \( k \in \mathbb{N} \) such that
    \[ d(x, x') \leq 2^{-k} \text{ implies } d(f(x), f(x')) \leq 2^{-\ell}. \]

- A bounded closed set \( S \) is called computable if there exists an algorithm that:
  - given \( k \in \mathbb{N} \),
  - produces a finite list \( S_k \) of computable points which is \( 2^{-k} \)-close to \( S \): 
    \[ d_H(S_k, S) \leq 2^{-k}. \]
8. Known Positive Results

- For every two computable numbers \( \ell < u \), we can, given a computable \( x \), check whether \( x > \ell \) or \( x < u \).
- To check this, it is sufficient to compute \( x, \ell, \) and \( u \) with a sufficient accuracy.
- There is an algorithm that, given two computable tuples \( x \) and \( y \), computes the distances
  \[
  d_\infty(x, y) \overset{\text{def}}{=} \max(|x_1 - y_1|, \ldots, |x_n - y_n|)
  \]
  \[
  d_2(x, y) \overset{\text{def}}{=} \sqrt{(x_1 - y_1)^2 + \ldots + (x_n - y_n)^2}.
  \]
- Minimum and maximum are computable.
- Composition of computable functions is computable.
- In particular, maximum and minimum of finitely many computable functions is computable.
9. **Known Positive Results (cont-d)**

- It is algorithmically possible, given a computable set $S$ and a computable function $F(x)$, to compute

$$\max_{x \in S} F(x) \text{ and } \min_{x \in S} F(x).$$

- For every computable f-n $F(x, y)$ and computable set $S \subseteq X$, the following f-ns are computable:

$$M(y) \overset{\text{def}}{=} \max_{x \in S} f(x, y) \text{ and } m(y) \overset{\text{def}}{=} \min_{x \in S} f(x, y).$$

- There is an algorithm that, given a computable tuple $x$ and a computable set $S$, returns the distance $d(x, S)$.

- For every two computable sets $A$ and $B$, their union $A \cup B$ is also computable.

- One can easily check that if $A_k$ approximate $A$ and $B_k$ approximate $B$, then $A_k \cup B_k$ approximates $A \cup B$. 
10. Known Positive Results (cont-d)

- For every two computable sets $A$ and $B$, the set of pairs $A \times B$ is also computable.
- There exists an algorithm that:
  - given:
    * a computable set $S$,
    * a computable function $f$, and
    * computable real numbers $a < b$,
  - returns a computable number $\eta \in (a, b)$ for which the set $\{x \in S : f(x) \leq \eta\}$ is also computable.
11. Known Negative Results about Computable Objects

- No algorithm is possible that, given a computable real number \( a \), would check whether \( a = 0 \) or \( a \neq 0 \).
- No algorithm is possible that, given a computable real number \( a \), would check whether \( a \geq 0 \) or \( a < 0 \).
- No algorithm is possible that, given a computable real number \( a \), would check whether \( a \leq 0 \) or \( a > 0 \).
12. First Problem Is Not Algorithmically Solvable

- **No algorithm is possible that:**
  
  - given a computable function \( f(x_1, \ldots, x_n) \) and computable numbers \( y \) and \( \bar{y} \),
  
  - returns the set \( \{x : y \leq f(x) \leq \bar{y}\} \).

- **Proof:**
  
  - The function \( f(x_1) = \max(\min(x_1, 0), x_1 - 1) \) is computable.
  
  - For \( y = -1 \) and \( \bar{y} = a \), the set \( \{x : y \leq f(x) \leq \bar{y}\} \)
    is equal to \([-1, 1 + a]\) if \( a \geq 0 \) and to \([-1, a]\) else.
  
  - Thus, the maximum \( M \) of \( F(x_1) = x_1 \) on this set is equal to \( 1 + a \) for \( a \geq 0 \) and to \( a \) else.
  
  - In particular, for \( |a| < 0.1 \), we get \( M \geq 0.9 \) when \( a \geq 0 \) and \( M < 0.1 \) when \( a < 0 \).
  
  - So, we could check whether \( a \geq 0 \) or \( a < 0 \), which is known to be impossible.
13. Under Crisp Constraints, All Four Problems Are Not Algorithmically Solvable

- **No algorithm is possible that:**
  - given a computable function $f(x)$ and computable numbers $y$ and $\overline{y}$,
  - returns the set $\{x : y \leq f(x) \leq \overline{y}\}$.

- **No algorithm is possible that**, given two computable sets $S_1$ and $S_2$, computes their intersection.

- **No algorithm is possible that**, given computable sets $S \subseteq X$ and $R \subseteq X \times Y$, returns the composition
  $$Y = R \circ S.$$  

- **No algorithm is possible that**, given computable sets $S$, $A$, and $D$, returns the set
  $$\{a \in A : f(x, a) \in D \text{ for all } x \in S\}.$$
14. Under Soft Constraints, All Four Problems Are Algorithmically Solvable

- **There is an algorithm that**, given a computable function $f(x)$ and computable numbers $y < \bar{y}$ and $\varepsilon > 0$, returns:
  
  - a computable value $\underline{Y}$ which is $\varepsilon$-close to $y$;
  - a computable value $\overline{Y}$ which is $\varepsilon$-close to $\bar{y}$, and
  - a computable set $\{x : \underline{Y} \leq f(x) \leq \overline{Y}\}$.

- **For each set $S$ and for each real number $\eta > 0$**, by an $\eta$-neighborhood $N_\eta(S)$, we mean $\{x : d(x, S) \leq \eta\}$.

- **There exists an algorithm that**:
  
  - given $m$ computable sets $S_1, \ldots, S_m$, and a computable real number $\varepsilon > 0$,
  - returns a computable number $\eta \in (0, \varepsilon)$ for which the intersection $N_\eta(S_1) \cap \ldots \cap N_\eta(S_m)$ is computable.
15. Under Soft Constraints, All Four Problems Are Algorithmically Solvable (cont-d)

- There exists an algorithm that:
  - given computable sets $S \subseteq X$ and $R \subseteq X \times Y$ and a computable real number $\varepsilon > 0$,
  - returns a computable number $\eta \in (0, \varepsilon)$ for which the composition $N_\eta(R) \circ N_\eta(S)$ is computable.

- There exists an algorithm that:
  - given computable $S \subseteq X$, $A$, $D$, a computable function $f(x, a)$, and a computable real number $\varepsilon > 0$,
  - returns a computable $\eta \in (0, \varepsilon)$ for which the following set is computable:

$$\{a \in A : f(x, a) \in N_\eta(D) \text{ for all } x \in S\}.$$

16. What If Some Measurements Are Faulty

- In the above analysis, we assumed that all the measurements are reliable.

- A measuring instrument sometimes mis-performs, resulting in a outlier numerical value.

- Usually, we know what fraction of measurement results is unreliable.

- So, we know that out of $m$ measurements, at least $q$ are correct.

- The set $S$ of possible states is thus equal to
  \[ S = \bigcup_{I: |I|=q} \left( \bigcap_{i \in I} S_i \right). \]

- Under soft constraints, this set $S$ is computable.

- However, one can prove that, in general, computing this set is an NP-hard problem.
17. Example: Locating an Underwater Robot

- To locate a robot, stationary sonars placed at known locations periodically send pings in all directions.
- A sonar receives a signal reflected from the robot.
- We can measure the “travel time” \( t_i \).
- Once we know the speed of sound \( v \), we can get the distance \( d_i = (v \cdot t_i)/2 \) to the robot.
- In the ideal case, we can find the coordinates \( x, y, \) and \( z \) if we know three distances

\[
\|r - r_i\| = \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2} = d_i.
\]

- In practice, measurements are imprecise: \( d_i \leq \tilde{d}_i \leq \bar{d}_i \).
- For each triple of measurement results \( \tilde{d}_i, \tilde{d}_j, \) and \( \tilde{d}_k \), we know that \( r \in S_{ijk} \overset{\text{def}}{=} S_i \cap S_j \cap S_k \).
18. Locating an Underwater Robot (cont-d)

- For each of the three coordinates $x$, $y$, and $z$, we compute the corresponding intervals

$$[\bar{x}_{ijk}, \underline{x}_{ijk}] = \{x : (x, y, z) \in S_{ijk}\},$$

$$[\bar{y}_{ijk}, \underline{y}_{ijk}] = \{y : (x, y, z) \in S_{ijk}\},$$

$$[\bar{z}_{ijk}, \underline{z}_{ijk}] = \{z : (x, y, z) \in S_{ijk}\}.$$

- Then, we select values that belong to $\geq q$ of such intervals.

- The resulting algorithm locates the robot in about 90% of the cases – better than known methods.

- This algorithm is also much faster than all previous algorithms.
19. Acknowledgment

- This work was supported in part by the US National Science Foundation grants:
  - HRD-0734825 and HRD-1242122 (Cyber-ShARE Center of Excellence) and
  - DUE-0926721.
- The work was performed when Quentin Brefort was visiting the University of Texas at El Paso.
- This visit was supported by ENSTA-Bretagne.
- The authors are thankful to the anonymous referees for valuable suggestions.