Algorithmics of Checking Whether a Mapping Is Injective, Surjective, and/or Bijective

E. Cabral Balreira[1], Olga Kosheleva[2], and Vladik Kreinovich[2]

[1]Department of Mathematics, Trinity University
San Antonio, TX 78212, USA
ebalreir@trinity.edu

[2]University of Texas at El Paso
El Paso, TX 79968, USA
olgak@utep.edu, vladik@utep.edu

1. Introduction

- States of real-life systems change with time.
- In some cases, this change comes "by itself", from laws of physics.
- Examples: radioactive materials decays, planets go around each other, etc.
- In other cases, the change comes from our interference.
- E.g., a spaceship changes trajectory after we send a signal to an engine to perform a trajectory correction.
- In many situations, we have equations that describe this change, i.e., we know a function $f : A \to B$ that:
  - transform the original state $a \in A$
  - into a state $f(a) \in b$ at a future moment of time.
- In such situations, the following two natural problems arise.

2. First Problem: Checking Injectivity

- The first natural question is: Are the changes reversible?
- When we erase the value of the variable in a computer, by replacing it with 0s, the changes are not reversible.
- In such situations, two different original states $a \neq a'$ lead to the exact same new state $f(a) = f(a')$.
- If different states $a \neq a'$ always lead to different $f(a) \neq f(a')$, then, in principle, reconstruction is possible.
- In mathematics, mappings $f : A \to B$ that map different elements into different ones are called *injective*.
- So the question is: checking whether a given mapping is injective.

3. Second Problem: Checking Surjectivity

- The second natural question is: Are all the states $b \in B$ possible as a result of this dynamics.
- In other words, is it true that every state $b \in B$ can be obtained as $f(a)$ for some $a \in A$.
- In mathematical terms, mappings that have this property are called *surjective.*
- We may also want to check whether f is both injective and surjective, i.e., whether it is a *bijection.*
- In this paper, we analyze these problems from an algorithmic viewpoint.

4. Case of Semi-Algebraic Mappings

- Let's first the case when A, B are described by finitely many polynomial (in)equalities w/rational coefficients.
- Such sets are called *semi-algebraic*.
- Example: the upper half of the unit circle centered at the point $(0, 0)$ is

$$\{(x_1, x_2) : x_1^2 + x_2^2 = 1 \text{ and } x_2 \geq 0\}.$$

- We also assume that the graph $\{(a, f(a)) : a \in A\}$ is semi-algebraic; such maps are called *semi-algebraic*.
- Example: every polynomial mapping is semi-algebraic.
- Polynomial mappings are very important:
  - every continuous f-n on a bounded set can be, w/any given accuracy, approximated by a polynomial;
  - this means that, in practice, every action can be represented by a polynomial mapping.

5. First Result: Algorithms Are Possible

- **Proposition:** *There exists an algorithm, that:*
  - *given two semi-algebraic sets A and B and a semi-algebraic mapping $f : A \to B$,*
  - *checks whether f is injective, surjective, and/or bijective.*
- Each of the relations $a \in A$, $b \in B$, and $f(a) = b$ can be described by a finite set of polynomial (in)equalities.
- A polynomial is, by definition, a composition of additions and multiplications.
- Thus, both the injectivity and surjectivity can be described in terms of the first order language with:
  - variables running over real numbers, and
  - elementary formulas coming from addition, multiplication, and equality:

6. Proof (cont-d)

- *Reminder:* we consider a first order language with:
  - variables running over real numbers, and
  - elementary formulas coming from addition, multiplication, and equality:
- In this language, injectivity can be described as:

$$\forall a\, \forall a'\, \forall b\, ((a \in A \,\&\, a' \in A \,\&\, f(a) = b \,\&\, f(a') = b \,\&\, b \in B) \Rightarrow a = a').$$

- Surjectivity can be described as:

$$\forall b\, (b \in B \Rightarrow \exists a\, (a \in A \,\&\, f(a) = b)).$$

- For such first order formulas, there is a deciding algorithm designed by a famous logician A. Tarski.
- Thus, the problems of checking injectivity and surjectivity are indeed algorithmically decidable.

7. Remark

- One of the main open problems in this area is Jacobian Conjecture, according to which:
  - every polynomial map $f : C^n \to C^n$ from n-dimensional complex space into itself
  - for which the Jacobi determinant $\det\left(\dfrac{\partial f_i}{\partial x_j}\right)$ equals 1,
  - is injective.
- This is an open problem; however:
  - for any given dimension n and for any given degree d of the polynomial,
  - the corresponding case of this conjecture can be resolved by applying the Tarski algorithm.

8. How Efficient Are the Corresponding Algorithms? Related Results

- **Proposition:** *Checking whether a given polynomial mapping $f : \mathbb{R}^n \to \mathbb{R}^n$ is injective is NP-hard.*
- **Proposition:** *Checking whether a given polynomial mapping $f : \mathbb{R}^n \to \mathbb{R}^n$ is surjective is NP-hard.*
- **Proposition:** *Checking whether a given polynomial mapping $f : \mathbb{R}^n \to \mathbb{R}^n$ is bijective is NP-hard.*
- *Open questions:* check whether the following problems are NP-hard:
  - checking whether a given injective polynomial mapping is also surjective, and
  - checking whether a given surjective polynomial mapping is also injective.

9. Proof that Checking Injectivity Is NP-Hard

- By definition, a problem is NP-hard if every problem from the class NP can be reduced to it.
- Thus, to prove that this problem is NP-hard, let us reduce a known NP-hard problem to it.
- As such a known problem, we take a *subset problem*:
  - given $n+1$ positive integers $s_1, \ldots, s_n, S$,
  - check whether there exist values $x_1, \ldots, x_n \in \{0, 1\}$ for which $\sum_{i=1}^{n} s_i \cdot x_i = S$.
- For each instance of the subset sum problem, take

$$f(x_1, \ldots, x_n, x_{n+1}) = (x_1, \ldots, x_n, P(x_1, \ldots, x_n) \cdot x_{n+1}),$$

$$P(x_1, \ldots, x_n) \stackrel{\text{def}}{=} \sum_{i=1}^{n} x_i^2 \cdot (1 - x_i)^2 + \left(\sum_{i=1}^{n} s_i \cdot x_i - S \right)^2.$$

10. Proof for injectivity (cont-d)

- *Reminder:*

$$f(x_1, \ldots, x_n, x_{n+1}) = (x_1, \ldots, x_n, P(x_1, \ldots, x_n) \cdot x_{n+1}),$$

$$P(x_1, \ldots, x_n) \stackrel{\text{def}}{=} \sum_{i=1}^{n} x_i^2 \cdot (1 - x_i)^2 + \left(\sum_{i=1}^{n} s_i \cdot x_i - S \right)^2.$$

- If the original instance has a solution $(x_1, \ldots, x_n)$, then $(x_1, \ldots, x_n, 0) \neq (x_1, \ldots, x_n, 1) \to (x_1, \ldots, x_n, 0)$.
- If the original instance does not have a solution, then $P(x_1, \ldots, x_n) > 0$, so we can recover x_{n+1}.
- So, the above mapping f is injective if and only if the original instance of the subset problem has a solution.
- The reduction is proven, so the problem of checking injectivity is indeed NP-hard.

11. Proofs for surjectivity and bijectivity

- *Same mapping:*

$$f(x_1, \ldots, x_n, x_{n+1}) = (x_1, \ldots, x_n, P(x_1, \ldots, x_n) \cdot x_{n+1}),$$

$$P(x_1, \ldots, x_n) \stackrel{\text{def}}{=} \sum_{i=1}^{n} x_i^2 \cdot (1 - x_i)^2 + \left(\sum_{i=1}^{n} s_i \cdot x_i - S \right)^2.$$

- If the original instance has a solution $(x_1, \ldots, x_n)$, then $P(x_1, \ldots, x_n) = 0$, so $(x_1, \ldots, x_n, 1)$ is not in the range.
- If the original instance does not have a solution, then $P(x_1, \ldots, x_n) > 0$, so all values are in the range. x_{n+1}.
- So, the above mapping f is surjective if and only if the original instance of the subset problem has a solution.
- Similarly, f is bijective if and only if the original instance has a solution.
- The reduction is proven, so the problem of checking injectivity is indeed NP-hard.

12. Polynomial Mapping with Computable Coefficients: Checking Injectivity

- A number x is called *computable* if $\exists$ algorithm s.t.:
  - given a natural number n,
  - returns a rational r_n s.t. $|r_n - x| \leq 2^{-m}$.
- **Proposition:** *No algorithm is possible that:*
  - *given a polynomial mapping $f : \mathbb{R}^n \to \mathbb{R}^n$ with computable coefficients,*
  - *decides whether this mapping is injective.*
- *Known result:* no algorithm is possible that, given a computable real number a, decides whether $a = 0$.
- *Proof:* take $n = 1$; $f(x) = a \cdot x$ is injective if and only if $a \neq 0$.
- Since we cannot algorithmically decide whether $a \neq 0$, we cannot algorithmically check whether f is injective.

13. Polynomial Mapping with Computable Coefficients: Checking Surjectivity and Bijectivity

- **Proposition:** *No algorithm is possible that:*
  - *given a polynomial mapping $f : \mathbb{R}^n \to \mathbb{R}^n$ with computable coefficients,*
  - *decides whether this mapping is surjective.*
- **Proposition:** *No algorithm is possible that:*
  - *given a polynomial mapping $f : \mathbb{R}^n \to \mathbb{R}^n$ with computable coefficients,*
  - *decides whether this mapping is bijective.*
- *Proof:* consider the same example $f(x) = a \cdot x$.

14. Auxiliary Results

- **Proposition:** *No algorithm is possible that:*
  - *given an injective polynomial mapping $f : [0,1] \to [0,1]$ with computable coefficients,*
  - *decides whether this mapping is also surjective.*
- Proof: for all $a \in [0, 0.5]$, $f(x) = (1-a^2) \cdot x$ is injective, but it is surjective only for $a = 0$.
- **Proposition:** *No algorithm is possible that:*
  - *given a surjective polynomial mapping $f : \mathbb{R}^n \to \mathbb{R}^n$ with computable coefficients,*
  - *decides whether this mapping is also injective.*
- *Proof:* $f(x) = -a^2 \cdot x^2 + x^3$ is always surjective, but it is injective only when $a^2 = 0$, i.e., when $a = 0$.

15. Case of Analytical Expressions

- Expression in terms of elem. constants (π, etc.) and elem. functions (sin, etc.) are called *analytical*.
- *Proposition: For analytical expressions, all three problems are algorithmically undecidable.*
- *Proof:* take a polynomial F and form a map:

$$f(x_1, \ldots, x_n, x_{n+1}) = (x_1, \ldots, x_n, P(x_1, \ldots, x_n) \cdot x_{n+1}),$$

$$P(x_1, \ldots, x_n) \stackrel{\text{def}}{=} \sum_{i=1}^{n} \sin^2(\pi \cdot x_i) + F^2(x_1, \ldots, x_n).$$

- This map is in-, sur-, and bi-jective $\Leftrightarrow$ P is never 0.
- $P = 0 \Leftrightarrow$ the equation $F = 0$ has an integer solution.
- *Matiyasevich's result:* no algorithm can check whether a polynomial equation $F = 0$ has an integer solution.

16. Checking Approximate Injectivity

- *Case:* computable mapping f between computable compact sets A and B.
- *Injectivity (reminder:)* $f(a) = f(a') \Rightarrow a = a'$.
- (δ, ε)-*injectivity:* $d(f(a), f(a')) \le \delta \Rightarrow d(a, a') \le \varepsilon$, i.e.,

$$m \stackrel{\text{def}}{=} \max\{d(a, a') : d(f(a), f(a')) \le \delta\} \le \varepsilon.$$

- *Known:* $\forall \underline{\delta} < \overline{\delta}\, \exists \delta \in (\underline{\delta}, \overline{\delta})$ s.t. S_δ is a computable compact, where:

$$S_\delta \stackrel{\text{def}}{=} \{(a, a') : d(f(a), f(a')) \le \delta\}.$$

- Thus, $m = \max\{d(a, a') : (a, a') \in S_\delta\}$ is computable.
- Thus, if we have two computable numbers $0 \le \underline{\varepsilon} < \overline{\varepsilon}$, we can check whether $m \ge \overline{\varepsilon}$ or $m \not\ge \underline{\varepsilon}$.
- So, for $(\underline{\delta}, \overline{\delta})$ and $(\underline{\varepsilon}, \overline{\varepsilon})$, we can algor. find $\delta \in (\underline{\delta}, \overline{\delta})$ and $\varepsilon \in (\underline{\varepsilon}, \overline{\varepsilon})$ s.t. (δ, ε)-injectivity is algor. decidable.

17. Checking Approximate Surjectivity

- *Case:* computable mapping f between computable compact sets A and B.
- *Surjectivity (reminder:)* $\forall b \in B\, \exists a \in A\, (b = f(a))$.
- *ε-surjectivity:* every $b \in B$ is ε-close to some $f(a)$, i.e.,

$$s \stackrel{\text{def}}{=} \max_{b \in B} \min_{a \in A} d(b, f(a)) \le \varepsilon.$$

- For computable mappings, s is computable.
- Thus, with each interval $(\underline{\varepsilon}, \overline{\varepsilon})$, we can algorithmically find ε s.t. ε-surjectivity is algorithmically decidable.

18. 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.