Can we verify a program if we can't do the maths?

James Davenport (Bath)

Thanks to Russell Bradford (Bath CS), Acyr Locatelli (Bath Maths), Gregory Sankaran (Bath Maths) and David Wilson (Bath CS) and Scott McCallum (Macquarie)

Computer Science Department 10th Anniversary
11 October 2012

Traditional Classification of Problems

blunder	(of the coding variety) This is the sort of error traditionally addressed in “program verification”. Typically independent of the arithmetic.
parallelism	Issues of deadlocks or races occurring due to the parallelism of an otherwise correct sequential program. Again, arithmetic-independent.
numerical	Do truncation and round-off errors, individually or combined, mean that the program computes approximations to the “true” answers which are out of tolerance.

Traditional Classification of Problems

How often are they considered?
Statistics from [CE05]

blunder (83%)	(of the coding variety) This is the sort of error traditionally addressed in "program verification". Typically independent of the arithmetic.
parallelism (13%)	Issues of deadlocks or races occurring due to the parallelism of an otherwise correct sequential program. Again, arithmetic-independent.
numerical (3%)	Do truncation and round-off errors, individually or combined, mean that the program computes approximations to the "true" answers which are out of tolerance.

To this, I wish to add a fourth kind

“The bug that dares not speak its name”

manipulation (0%!)	A piece of algebra, which is “obviously correct”, turns out not to be correct when interpreted, not as abstract algebra, but as the manipulation of functions $\mathbf{R} \to \mathbf{R}$ or $\mathbf{C} \to \mathbf{C}$.

A note on complex numbers

Most of our examples involve complex numbers, and people say

real programs don't use complex numbers

However

- `COMPLEX` in Fortran II (1958–61) was the first programming language data type not corresponding to a machine one
- Even C99 introduced `_Complex`
- Many examples, notably in fluid mechanics.

Kahan's example [Kah87]

Flow in a slotted strip, transformed by

$$w = g(z) := 2\,\mathrm{arccosh}\left(1 + \frac{2z}{3}\right) - \mathrm{arccosh}\left(\frac{5z+12}{3(z+4)}\right) \qquad (1)$$

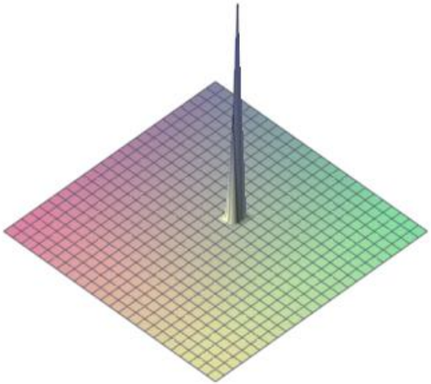
into a more tractable region.
Is this the same transformation as

$$w \overset{?}{=} q(z) := 2\,\mathrm{arccosh}\left(2(z+3)\sqrt{\frac{z+3}{27(z+4)}}\right)? \qquad (2)$$

Or possibly

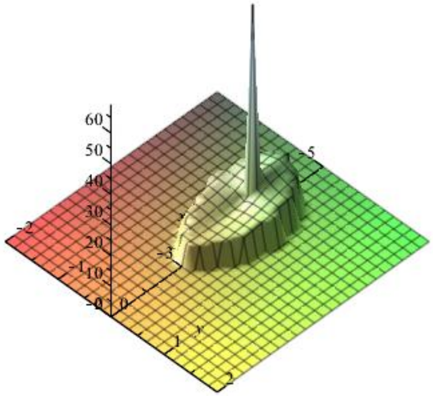
$$w \overset{?}{=} h(z) := 2\ln\left(\frac{1}{3}\,\frac{\sqrt{3\,z+12}\left(\sqrt{z+3}+\sqrt{z}\right)^2}{2\sqrt{z+3}+\sqrt{z}}\right)? \qquad (3)$$

$g - q$ might look OK



“OK apart from a slight glitch.”

But if we look closer



Definitely not OK

But, in fact $g = h$

Most computer algebra systems (these days!) will refuse to "simplify" g to q
But will also refuse to simplify g to h.
Indeed Maple's `coulditbe(g<>h);` returns `true`, which *ought* to indicate that there is a counter-example.
If $g = h$ then $g - h$ is zero:

$$\frac{d(g-h)}{dz} = 2\left(\sqrt{\frac{z}{z+4}}\sqrt{\frac{z+3}{z+4}}z^{3/2} - 2\,z^{3/2} + 2\sqrt{z+3}\sqrt{\frac{z}{z+4}}\cdot\right.$$
$$\sqrt{\frac{z+3}{z+4}}z - z\sqrt{z+3} + 4\sqrt{\frac{z}{z+4}}\sqrt{\frac{z+3}{z+4}}\sqrt{z} + 8\sqrt{z+3}\sqrt{\frac{z}{z+4}}\cdot$$
$$\left.\sqrt{\frac{z+3}{z+4}} - 6\sqrt{z}\right)\frac{1}{\sqrt{z+3}}\frac{1}{\sqrt{z}}\frac{1}{\sqrt{\frac{z}{z+4}}}\frac{1}{\sqrt{\frac{z+3}{z+4}}}(z+4)^{-2}\left(2\sqrt{z+3}+\sqrt{z}\right)^{-1}$$

and it's a bold person who would say "$= 0$"

Challenges

Challenge (1)

Demonstrate automatically that g and q are not equal, by producing a z at which they give different results.

The technology described in [BBDP07] will isolate the curve $y = \pm\sqrt{\frac{(x+3)^2(-2x-9)}{2x+5}}$ as a potential obstacle (it is the branch cut of q), but the geometry questions are too hard for a fully-automated solution at the moment.

Challenge (2)

Demonstrate automatically that g and h are equal.

Again, the technology in [BBDP07], implemented in a mixture of Maple and QEPCAD, could in principle do this

Joukowski (a)

Consider the Joukowski map [Hen74, pp. 294–298]:

$$f : z \mapsto \frac{1}{2}\left(z + \frac{1}{z}\right). \tag{4}$$

Lemma

f is injective as a function from $D := \{z : |z| > 1\}$.

This is also a function $\mathbf{R}^2 \to \mathbf{R}^2$:

$$f_R : (x, y) \mapsto \left(\frac{1}{2}x + \frac{1}{2}\frac{x}{x^2+y^2}, \frac{1}{2}y - \frac{1}{2}\frac{y}{x^2+y^2}\right) \tag{5}$$

Joukowski challenge (a1)

Challenge (3)

Demonstrate automatically that f_R is injective, i.e.

$$\begin{aligned} \forall x_1 \forall x_2 \forall y_1 \forall y_2 \quad & \Bigl(x_1^2 + y_1^2 > 1 \land x_2^2 + y_2^2 > 1 \land \\ x_1 + \tfrac{x_1}{x_1^2+y_1^2} = x_2 + \tfrac{x_2}{x_2^2+y_2^2} \quad \land \quad & y_1 - \tfrac{y_1}{x_1^2+y_1^2} = y_2 - \tfrac{y_2}{x_2^2+y_2^2} \Bigr) \\ \Rightarrow \quad & \bigl(x_1 = x_2 \land y_1 = y_2 \bigr). \end{aligned} \tag{6}$$

We have failed to do this automatically, but Brown can reformulate manually then solve in QEPCAD ($<$ 12 seconds)

Challenge (4)

Automate these techniques and transforms.

Joukowski challenge (a2)

So it's a bijection: what's the inverse?

Figure: Maple's `solve` on inverting Joukowski

```
> [solve(zeta = 1/2*(z+1/z), z)];
```

$$\left[\zeta + \sqrt{\zeta^2 - 1}, \zeta - \sqrt{\zeta^2 - 1}\right]$$

The only challenge might be the choice implicit in the $\pm\sqrt{}$ idea: which do we choose? Unfortunately, the answer is "neither", or at least "neither, uniformly".

Joukowski challenge (a2 continued)

$$f_1(\zeta) = \zeta \begin{cases} +\sqrt{\zeta^2 - 1} & \Im(\zeta) > 0 \\ -\sqrt{\zeta^2 - 1} & \Im\zeta) < 0 \\ +\sqrt{\zeta^2 - 1} & \Im(\zeta) = 0 \wedge \Re(\zeta) > 1 \\ -\sqrt{\zeta^2 - 1} & \Im(\zeta) = 0 \wedge \Re(\zeta) < -1 \end{cases} \tag{7}$$

In fact, a better (at least, free of case distinctions) definition is

$$f_2(\zeta) = \zeta + \sqrt{\zeta - 1}\sqrt{\zeta + 1}. \tag{8}$$

The techniques of [BBDP07] are able to **verify** (8), in the sense of showing that $f_2(f(z)) - z$ is the zero function on $\{z : |z| > 1\}$.

Challenge (5)

Derive automatically, and demonstrate the validity of, either (7) or (8). In terms of Maple, this would be . . .

Joukowski challenge (a2 continued)

Figure: Bad: Maple's actual `solve` on inverting injective Joukowski

```
> [solve(zeta = 1/2*(z+1/z), z)] assuming abs(z) > 1
```

$$\left[\zeta+\sqrt{\zeta^2-1}, \zeta-\sqrt{\zeta^2-1}\right]$$

Figure: Good: Ideal software inverting injective Joukowski

```
> solve(zeta = 1/2*(z+1/z), z) assuming abs(z) > 1
```

$$\zeta+\sqrt{\zeta-1}\sqrt{\zeta+1}$$

As far as I can tell (supported by the documentation), Maple ignores the "`assuming`" as it's on the codomain, not the domain. Currently, we can't solve quadratics!

Joukowski (b) challenge

Lemma

f is injective as a function from $H := \{z : \Im z > 0\}$.

Challenge (6)

Demonstrate automatically the truth of

$$\forall x_1 \forall x_2 \forall y_1 \forall y_2 \quad \Bigl(y_1 > 0 \wedge y_2 > 0 \wedge x_1 + \tfrac{x_1}{x_1^2+y_1^2} = x_2 + \tfrac{x_2}{x_2^2+y_2^2} \wedge y_1 - \tfrac{y_1}{x_1^2+y_1^2} = y_2 - \tfrac{y_2}{x_2^2+y_2^2} \Bigr) \Rightarrow \bigl(x_1 = x_2 \wedge y_1 = y_2 \bigr). \tag{9}$$

Brown's ideas probably apply here as well, but this is unproven

Joukowski (b) challenge continued

So it's a bijection: what's the inverse?
[Hen74, (5.1-13), p. 298] argues for

$$f_3(\zeta) = \zeta + \underbrace{\sqrt{\zeta - 1}}_{\arg\in(-\pi/2,\pi/2]} \underbrace{\sqrt{\zeta + 1}}_{\arg\in(0,\pi]}. \tag{10}$$

Challenge (7)

Find a way to represent functions such as $\underbrace{\sqrt{\zeta + 1}}_{\arg\in(0,\pi]}$

Alternative formulations

Fortunately this one is soluble in this case, we can write

$$\underbrace{\sqrt{\zeta+1}}_{\arg\in(0,\pi]} = i \underbrace{\sqrt{-\zeta-1}}_{\arg\in(-\pi/2,\pi/2]},$$

so we have an inverse function

$$f_4(\zeta) = \zeta + \sqrt{\zeta-1}\,i\sqrt{-\zeta-1}. \tag{11}$$

Challenge (8)

Demonstrate automatically that this is an inverse to f on $\{z : \Im z > 0\}$.

Why is this so hard? (1) — CAD

The first truly algorithmic approach is ten years old ([BCD$^+$02], refined in [BBDP07]), and has various difficulties.
At its core is the use of Cylindrical Algebraic Decomposition of $\mathbf{R}^N$ to find the connected components of $\mathbf{C}^{N/2} \setminus \{\mathrm{branch\ cuts}\}$. The complexity of this is doubly exponential in N: upper bound of $d^{O(2^N)}$ and lower bounds of $2^{2^{(N-1)/3}}$.
While better algorithms are in principle known ($d^{O(N\sqrt{N})}$), we do not know of any accessible implementations.
Furthermore, we are clearly limited to small values of N, at which point looking at $O(\ldots)$ complexity is of limited use. We note that the cross-over point between $2^{(N-1)/3}$ and $N\sqrt{N}$ is at $N = 21$.
A more detailed comparison is given in [Hon91]. Hence there is a need for practical research on low-N Cylindrical Algebraic Decomposition.

Why is this so hard? (1) — CAD continued

While the fundamental branch cut of log is simple enough, being $\{z = x + iy | y = 0 \wedge x < 0\}$, actual branch cuts are messier. Part of the branch cut of (2) is

$$2x^3 + 21x^2 + 72x + 2xy^2 + 5y^2 + 81 = 0 \wedge \text{other conditions}, \quad (12)$$

whose solution accounts for the curious boundary of the bad region. While there has been some progress in manipulating such images of half-lines (described in Phisanbut's Bath PhD), there is almost certainly more to be done.

Why is this so hard? (2) — Injectivity

Lemmas 1, 2 are statements about complex functions of one variable, so why do we need statements about four real variables to prove them? There are three reasons.

1. The statements require the $|\cdot|$ or $\Im$ functions, neither of which are **C** analytic functions. Hence some recourse to **R** (twice as many variables) seems inevitable (proof?)
2. Equations (6) and (9) are the direct translations of the basic definition of injectivity. In practice, certainly if we were looking at functions $\mathbf{R} \to \mathbf{R}$, we would want to use the fact that the function concerned was continuous.

Challenge (9)

Find a better formulation of injectivity questions $\mathbf{R}^N \to \mathbf{R}^N$*, making use of the properties of the functions concerned (certainly continuity, possibly rationality).*

Why is this so hard? (2) — Injectivity continued

1. While these injectivity equations are statements from the existential theory of the reals, and so the theoretically more efficient algorithms quoted in [Hon91] are in principle applicable, the more modern developments described in [PJ09] do not seem to be directly applicable. However, we can transform then into a disjunction of statements to each of which the Weak Positivstellensatz [PJ09, Theorem 1] is applicable.

Challenge (10)

Solve these problems using the techniques of [PJ09],

Bibliography

J.C. Beaumont, RJB, JHD, and N. Phisanbut.
Testing Elementary Function Identities Using CAD.
AAECC, 18:513–543, 2007.

RJB, R.M. Corless, JHD, D.J. Jeffrey, and S.M. Watt.
Reasoning about the Elementary Functions of Complex Analysis.
Ann. Math. Artificial Intelligence, 36:303–318, 2002.

R. Cousot (Ed.).
Verification, Model Checking, and Abstract Interpretation.
Springer Lecture Notes in Computer Science 3385, 2005.

P. Henrici.
Applied and Computational Complex Analysis I. Wiley, 1974.

H. Hong. Comparison of several decision algorithms for the existential theory of the reals. RISC Technical Report 91-41, 1991.

W. Kahan. Branch Cuts for Complex Elementary Functions.
Proc. The State of Art in Numerical Analysis, pages 165–211, 1987.

G.O. Passmore and P.B. Jackson.
Combined Decision Techniques for the Existential Theory of the Reals. *Proc. Intelligent Computer Mathematics, pages 122–137,*