Niche decision procedures

John Harrison
Intel Corporation

Calculemus, RISC Linz

Wed 27th June 2007

Guiding principles

- Traditional decidable fragments (Presburger, Tarski etc.) have been thoroughly mined, and are not always practical/useful.
- Real applications throw up requirements for customized niche decision procedures
- May use more refined features of the problem that are often ignored, or may try to solve a slightly “more general” problem.

Guiding principles

- Traditional decidable fragments (Presburger, Tarski etc.) have been thoroughly mined, and are not always practical/useful.
- Real applications throw up requirements for customized niche decision procedures
- May use more refined features of the problem that are often ignored, or may try to solve a slightly “more general” problem.
- Examples
  - Verification of inductive invariants exploiting types (Pnueli et al, Fontaine)
  - Automatic proof of divisibility properties via ideal membership
  - Linear reasoning in normed spaces

Example I
Verification of inductive invariants exploiting types

Parametrized systems

Important target for verification is *parametrized systems*.

N equivalent replicated components, so the state space involves some Cartesian product

$$\Sigma = \Sigma_0 \times \overbrace{\Sigma_1 \times \cdots \times \Sigma_1}^{N \text{ times}}$$

and the transition relation is symmetric between the replicated components.

Sometimes we have subtler symmetry, but we'll just consider full symmetry.

Multiprocessors with private cache

Example: multiprocessor where each processor has its own cache.

We have N cacheing agents with state space Σ_1 each, and maybe some special ‘home node’ with state space Σ_0.

We can consider Σ_1 as finite with two radical but not unreasonable simplifications:

- Assume all cache lines are independent (no resource allocation conflicts)
- Ignore actual data and consider only state of cache line (dirty, clean, whatever)

Coherence

The permitted transitions are constrained by a protocol designed to ensure that all caches have a coherent view of memory.

On some simplifying assumptions, we can express this adequately just using the cache states.

In classic MESI protocols, each cache can be in four states: `Modified`, `Exclusive`, `Shared` and `Invalid`.

Coherence means:

$$
\begin{array}{l}
\forall \mathtt{i}.\ \mathtt{Cache(i)\ IN}\ \{\mathtt{Modified,\ Exclusive}\} \\
\qquad \Rightarrow \forall \mathtt{j}.\ \neg(\mathtt{j} = \mathtt{i}) \Rightarrow \mathtt{Cache(j)} = \mathtt{Invalid}
\end{array}
$$

Parametrized verification

Even if Σ_0 and Σ_1 are finite, we can only use straightforward model checking when N is a specific number.

In practice, only small N may be feasible.

Yet the system is often expected/supposed to work for *arbitrary* N.

So we would like a proof that is general, with N treated as an arbitrary parameter.

Might use induction to prove an invariant (even coherence itself in very simple abstract case).

Inductive proof

Inductiveness statement is

$$I(\sigma) \wedge R(\sigma, \sigma') \Rightarrow I(\sigma')$$

The inductive invariant I is universally quantified, and occurs in both antecedent and consequent.

The transition relation has outer existential quantifiers $\exists i.\ \cdots$ because we have a symmetric choice between all components.

Inside, we may also have universal quantifiers if we choose to express array updates `a(i) := Something` as relations between functions:

$$a'(i) = \texttt{Something} \wedge \forall j.\, \neg(j = i) \Rightarrow a'(j) = a(j)$$

Our quantifier prefix

So our inductiveness claim may look like

$$(\forall i, j, \ldots .\ \cdots) \wedge (\exists i.\ \forall j.\ \cdots) \Rightarrow (\forall i, j, \ldots .\ \cdots)$$

If we put this into prenex normal form in the right way, the quantifier prefix is of the form:

$$\forall \cdots \forall \exists \cdots \exists$$

Suppose we don't need any arithmetic.

We can add assumptions for exhaustiveness and exclusiveness of the 4-element type of cache states without disturbing the logical form.

Is this problem decidable?

The AE fragment

A classic decidability result for first order logic due to Bernays, Schönfinkel and Ramsey.

A first-order formula is in AE form if it contains no function symbols and has, or can obviously be transformed into, the following prenex form:

$$\forall x_1, \ldots, x_n. \exists y_1, \ldots, y_m. P[x_1, \ldots, x_n, y_1, \ldots, y_m]$$

with $P[x_1, \ldots, x_n, y_1, \ldots, y_m]$ quantifier-free. Dually, EA form is

$$\exists x_1, \ldots, x_n. \forall y_1, \ldots, y_m. P[x_1, \ldots, x_n, y_1, \ldots, y_m]$$

Logical validity for AE formulas / satisfiability for EA formulas is decidable.

Skolem-Gödel-Herbrand proof

By Skolemization, the formula is satisfiable iff this is:

$$\forall y_1, \ldots, y_m.\ P[c_1, \ldots, c_n, y_1, \ldots, y_m]$$

By the Skolem-Gödel-Herbrand theorem this is unsatisfiable iff the set of all ground instances

$$\bigwedge_{t_1, \ldots, t_m} P[c_1, \ldots, c_n, t_1, \ldots, t_m]$$

with t_i ranging over all ground terms.

But the only ground terms are the constants c_i, so this is a finite conjunction, and we can decide it propositionally.

Again, this fails if we have a function symbol, because then we need to consider the infinite set of instantiations to c, $f(c)$, $f(f(c))$, ...

Not quite what we need

Our inductive invariance claim does have an AE quantifier prefix.

And it doesn't need any background theory like arithmetic.

Not quite what we need

Our inductive invariance claim does have an AE quantifier prefix.

And it doesn't need any background theory like arithmetic.

Unfortunately it *does* include functions! We have the function `Cache` representing the array of caches . . .

Many-sorted Skolem-Gödel-Herbrand

In many sorted-logic, the obvious analog of the Skolem-Gödel-Herbrand theorem holds.

However, the construction of ground terms is constrained by type: we only consider well-typed combinations.

In particular, since `Cache` has type `Node` $\rightarrow$ `State`, terms like `Cache`$($`Cache`$(i))$ are ill-typed.

So there is *still* only a finite set of ground terms!

Practical implications

Our inductiveness problem *is* decidable. The decision method: a relatively modest finite expansion then hit it with a free-variable SMT solver.

Works for relatively complex transition relations and invariants, provided their logical form is right.

We can even add theories such as arithmetic, even though in general this leads to undecidability.

Still some limitations, since many non-trivial protocols have arrays of nodes (FLASH etc.)

Example II
Automatic proof of divisibility properties

Solving a more general problem

Classic example is proving a universally quantified linear formula over the integers.

Just solve the 'LP relaxation'.

Combine with some simple discretization, e.g. $x < y \Leftrightarrow x \leq y - 1$; usually very effective.

However, misses simple formulas like $\neg(2x = 2y + 1)$.

Divisibility properties over the integers

Often want to prove tedious lemmas like

$$\forall a\ n\ x\ y.\ ax \equiv ay \pmod{n} \wedge \mathsf{coprime}(a, n) \Rightarrow x \equiv y \pmod{n}$$

Expanding divisibility properties

Eliminate divisibility notions in terms of existentials:

- $s \mid t$ to $\exists d.\ t = sd$
- $s \equiv t \pmod{u}$ to $\exists d.\ t - s = ud$
- $\mathsf{coprime}(s,t)$ to $\exists x\ y.\ sx + ty = 1$.

Applied to the example

$$\begin{aligned} \forall a\ n\ x\ y.\quad & (\exists d.\ ay - ax = nd) \wedge \\ & (\exists u\ v.\ au + nv = 1) \\ & \Rightarrow (\exists e.\ y - x = ne) \end{aligned}$$

Pull out the quantifiers in the antecedent:

$$\forall a\ n\ x\ y\ d\ u\ v.\ ay - ax = nd \wedge au + nv = 1 \Rightarrow \exists e.\ y - x = ne$$

Solving a more general problem

We are already well into the realm of ‘undecidable in general’ thanks to the unsolvability of Hilbert’s 10^{th} problem.

Solving a more general problem

We are already well into the realm of 'undecidable in general' thanks to the unsolvability of Hilbert's 10^{th} problem.

Instead, attempt to prove the property holds *in all rings*.

It turns out that this problem *is* decidable using well-known methods.

Word problem for rings

$$\forall \overline{x}.\ p_1(\overline{x}) = 0 \wedge \cdots \wedge p_n(\overline{x}) = 0 \Rightarrow q(\overline{x}) = 0$$

holds in all rings iff

$$q \in \mathsf{Id}_{\mathbb{Z}}\ \langle p_1, \ldots, p_n \rangle$$

i.e. there exist ‘cofactor’ polynomials with integer coefficients such that

$$p_1 \cdot q_1 + \cdots + p_n \cdot q_n = q$$

Generalizes to linear existential theorems

$$\forall \overline{x}.\ \bigwedge_{i=1}^{m} e_i(\overline{x}) = 0 \Rightarrow \exists y_1 \ \cdots \ y_n.\ \ p_1(\overline{x})y_1 + \cdots + p_n(\overline{x})y_n = a(\overline{x})$$

holds in all rings iff (Horn-Herbrand) there are terms in the language of rings s.t.

$$\textsf{Ring} \vdash \forall \overline{x}.\ \bigwedge_{i=1}^{m} e_i(\overline{x}) = 0 \Rightarrow \ \ p_1(\overline{x})t_1(\overline{x}) + \cdots + p_n(\overline{x})t_n(\overline{x}) = a(\overline{x})$$

iff (previous theorem)

$$a \in \textsf{Id}_{\mathbb{Z}}\, \langle e_1, \ldots, e_m, p_1, \ldots, p_n \rangle$$

. . . and simultaneous linear existentials

$$\forall \overline{x}.\ \bigwedge_{i=1}^{m} e_i(\overline{x}) = 0 \Rightarrow \exists y_1 \ \cdots \ y_n.\ \begin{array}{l} p_{11}(\overline{x})y_1 + \cdots + p_{1n}(\overline{x})y_n = a_1(\overline{x}) \wedge \\ \cdots \wedge \\ p_{k1}(\overline{x})y_1 + \cdots + p_{kn}(\overline{x})y_n = a_k(\overline{x}) \end{array}$$

holds in all rings iff

$$\begin{array}{l} (a_1 u_1 + \cdots + a_k u_k) \\ \in \mathsf{Id}_{\mathbb{Z}} \langle e_1, \ldots, e_m, (p_{11}u_1 + \cdots + p_{k1}u_k), (p_{1n}u_1 + \cdots + p_{kn}u_k) \rangle \end{array}$$

where the u_i are fresh variables.

Solving ideal membership problems

The most natural approach to solving ideal membership problem is Gröbner bases.

Strictly, should use an integer version. However, can use the rational version speculatively and see if we get integer cofactors.

With an instrumented version of Buchberger's algorithm, can generate cofactors and hence easily generate a rigorous formal proof.

Successful examples

$$d|a \wedge d|b \Rightarrow d|(a-b)$$

$$\text{coprime}(d,a) \wedge \text{coprime}(d,b) \Rightarrow \text{coprime}(d,ab)$$

$$\text{coprime}(d,ab) \Rightarrow \text{coprime}(d,a)$$

$$\text{coprime}(a,b) \wedge x \equiv y \pmod{a} \wedge x \equiv y \pmod{b} \Rightarrow x \equiv y \pmod{(ab)}$$

$$m|r \wedge n|r \wedge \text{coprime}(m,n) \Rightarrow (mn)|r$$

$$\text{coprime}(xy, x^2+y^2) \Leftrightarrow \text{coprime}(x,y)$$

$$\text{coprime}(a,b) \Rightarrow \exists x.\ x \equiv u \pmod{a} \wedge x \equiv v \pmod{b}$$

$$ax \equiv ay \pmod{n} \wedge \text{coprime}(a,n) \Rightarrow x \equiv y \pmod{n}$$

$$\gcd(a,n) \mid b \Rightarrow \exists x.\ ax \equiv b \pmod{n}$$

Failures

Can't solve problems where special properties of the integers are used

$$2|x^2 + x$$

This fails over some rings, e.g. $\mathbb{R}[x]$.

However, such examples very seldom appear in typical routine lemmas.

Example III
Linear reasoning in normed spaces

Norm properties in analysis

In the formalization of complex analysis or analysis in $\mathbb{R}^n$, often need tedious lemmas about distances like

$$|\|w - z\| - r| = d \wedge \|u - w\| < d/2 \wedge \|x - z\| = r \Rightarrow d/2 \leq \|x - u\|$$

Solvable in principle

- Could replace each variable x with a pair (x_1, x_2) and replace $\|x\|$ by $\sqrt{x_1^2 + x_2^2}$.
- Even in general $\mathbb{R}^n$ with a norm defined via inner products, can use Solovay's procedure

Solvable in principle

- Could replace each variable x by a pair (x_1, x_2) and replace $\|x\|$ by $\sqrt{x_1^2 + x_2^2}$.
- Even in general $\mathbb{R}^n$ with a norm defined via inner products, can use Solovay's procedure

However *linear* vector problems give rise to *nonlinear* real problems.

Solving a more general problem

Instead try to show that the property holds in *all normed spaces*.

In this setting we can preserve linearity in the vector problem.

Normed spaces

Usual vector space axioms plus properties of norms:

$$\|x\| = 0 \Leftrightarrow x = 0$$

$$\|cx\| = |c|\|x\|$$

$$\|x + y\| \leq \|x\| + \|y\|$$

Euclidean norm $\sqrt{\sum_i x_i^2}$ satisfies these, but so do many others, e.g. 1-norm $\sum_i |x_i|$ and the infinity-norm $\max_i |x_i|$.

Suppose that we use *just* these three norm properties.

In principle, all different

Some problems hold in 1-D Euclidean space, but not in general

$$\|x-y\|+\|y-z\| = \|x-z\| \vee \|y-z\|+\|z-x\| = \|y-x\| \vee \|z-x\|+\|x-y\| = \|y-x\|$$

and others depend on the norm, e.g. this fails in 1-norm

$$\begin{aligned}
&\|a-c\| = \|b-c\| \wedge \|b-a\| = 2\|a-c\| \wedge \\
&\|a-c'\| = \|b-c'\| \wedge \|b-a\| = 2\|a-c'\| \\
&\Rightarrow c' = c
\end{aligned}$$

But in practice, most routine lemmas still work in any normed space.

The linear universal theory of normed spaces

This example shows the key ideas:

$$\begin{aligned}
&\|x + y\| \leq 1 \wedge \\
&\|2x + 3y\| \leq 2 \wedge \\
&\|x - 5y\| \leq 3 \wedge \\
&\|3x - 4y\| \leq 4 \\
&\Rightarrow \|y\| \leq \text{??}
\end{aligned}$$

What can we deduce?

Using the norm properties, we can generate any upper bounds of the form:

$$\|a(x+y)+b(2x+3y)+c(x-5y)+d(3x-4y)\| \leq |a|+2|b|+3|c|+4|d|$$

for $a, b, c, d \in \mathbb{R}$.

What can we deduce?

Using the norm properties, we can generate any upper bounds of the form:

$$\|a(x+y)+b(2x+3y)+c(x-5y)+d(3x-4y)\| \leq |a|+2|b|+3|c|+4|d|$$

for $a, b, c, d \in \mathbb{R}$.

If we want this to be a bound on y we need:

$$a+2b+c+3d=0$$

$$a+3b-5c-4d=1$$

A familiar problem

We can do some case splits to eliminate the absolute value function, so we get 16 cases of the form:

Minimize $a + 2b + 3c + 4d$ subject to:

$$a - 2b + c - 3d = 0$$

$$a - 3b - 5c + 4d = 1$$

$$a \geq 0$$

$$b \geq 0$$

$$c \geq 0$$

$$d \geq 0$$

Just a linear programming problem! Solution is $8/13$

The general case

In general we have linear forms in real variables and other norms as bounds:

$$\begin{aligned}
&\|x+y\| \leq s \wedge \\
&\|2x+3y\| \leq t \wedge \\
&\|x-5y\| \leq u \wedge \\
&\|3x-4y\| \leq v \\
&\Rightarrow \|y\| \leq \text{??}
\end{aligned}$$

Parametrized linear programming

Now we need to minimize $as + bt + cu + dv$ subject to:

$$a - 2b + c - 3d = 0$$

$$a - 3b - 5c + 4d = 1$$

$$a \geq 0$$

$$b \geq 0$$

$$c \geq 0$$

$$d \geq 0$$

A parametrized form of linear programming.

Naive solution

The constraining polytope is still unparametrized.

Enumerate all its vertices (well-studied problem, or use stupid algorithm of solving all n-tuples of constraints with unique solutions).

Each vertex gives rise to a linear constraint in terms of s, t, u, v.

In our example

We can do limited naive subsumption, but in general we get many bounds:

$$\|y\| \leq 3/11u + 1/11v$$
$$\|y\| \leq 3/17t + 2/17v$$
$$\|y\| \leq 1/13t + 2/13u$$
$$\|y\| \leq 2s + t$$
$$\|y\| \leq 3/7s + 1/7v$$
$$\|y\| \leq 1/6s + 1/6u$$

Can integrate this into standard linear prover to get a complete proof procedure.

Successful examples

$$||x\| - \|y\|| \leq \|x - y\|$$

$$||w - z\| - r| = d \wedge \|u - w\| < d/2 \wedge \|x - z\| = r \Rightarrow d/2 \leq \|x - u\|$$

$$\neg(x = u) \wedge \neg(x = w) \wedge \|x - z\| = r \wedge \|u - w\| < d/2 \wedge 0 < \|u - w\| \wedge$$
$$0 < d \wedge ||w - z\| - r| = d \wedge 0 < e \wedge 0 \leq r \wedge \neg(\|w - z\| = r) \wedge 0 < r$$
$$\Rightarrow d \leq \|x - w\|$$

Conclusions

- Some examples of non-traditional logical decision procedures
  - More refined view from sorts
  - Assuming assertion is true in a more general setting

Conclusions

- Some examples of non-traditional logical decision procedures
  - More refined view from sorts
  - Assuming assertion is true in a more general setting
- Often respond to a real practical need: necessity is the mother of invention!

Conclusions

- Some examples of non-traditional logical decision procedures
  - More refined view from sorts
  - Assuming assertion is true in a more general setting
- Often respond to a real practical need: necessity is the mother of invention!
- There are probably many more useful examples to be found . . .