# MetiTarski: An Automatic Prover for Real-Valued Special Functions

Behzad Akbarpour and Lawrence C. Paulson
Computer Laboratory, Cambridge

# special functions

- Many application domains concern statements involving the functions sin, cos, ln, exp, etc.
- We prove them by combining a *resolution theorem prover* (Metis) with a decision procedure for *real closed fields* (QEPCAD).
- MetiTarski works automatically and delivers machine-readable proofs.

# the basic idea

- Our approach involves replacing functions by *rational function upper or lower bounds*.
- The eventual polynomial inequalities belong to a decidable theory: *real closed fields (RCF).*
- Logical formulae over the reals involving + – × ≤ and quantifiers are decidable (Tarski).

We call such formulae algebraic.

# bounds for exp

- Special functions can be approximated, e.g. by Taylor series or continued fractions.
- Typical bounds are only valid (or close) over a restricted range of arguments.
- We need several formulas to cover a range of intervals. Here are a few of the options.

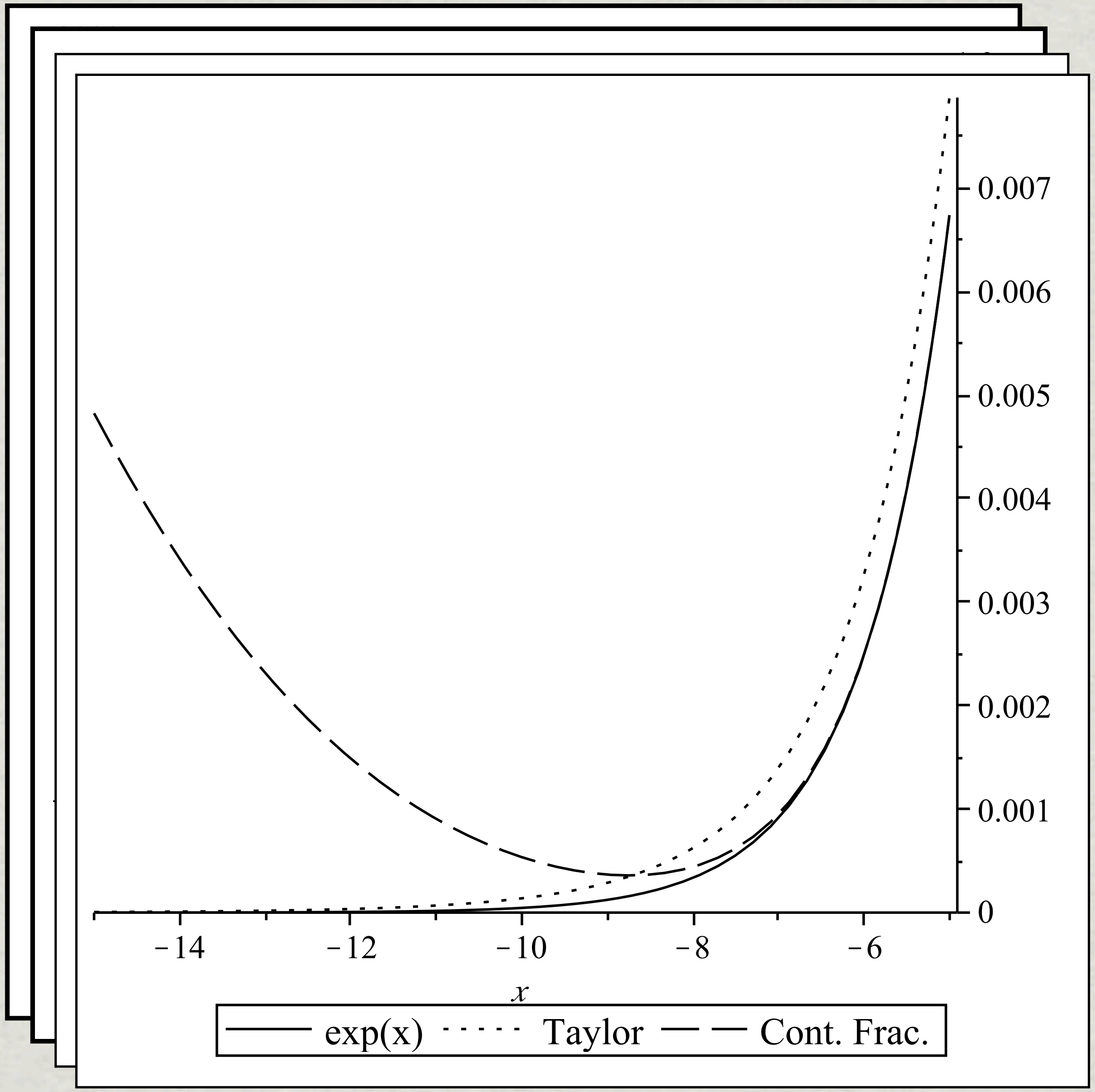
$$\exp(x) \geq 1 + x + \cdots + x^n/n! \qquad (n \text{ odd})$$

$$\exp(x) \leq 1 + x + \cdots + x^n/n! \qquad (n \text{ even}, x \leq 0)$$

$$\exp(x) \leq 1/(1 - x + x^2/2! - x^3/3!) \qquad (x < 1.596)$$

# Bounds and their quirks

- Some are extremely accurate at first, but veer away drastically.
- There is no general upper bound for the exponential function.



# bounds for ln

- based on the continued fraction for ln(x+1)
- much more accurate than the Taylor expansion

$$\frac{(1+19x+10x^2)(x-1)}{3x(3+6x+x^2)} \leq \ln x \leq \frac{(x^2+19x+10)(x-1)}{3(3x^2+6x+1)}$$

# RCF decision procedure

- Quantifier elimination reduces a formula to TRUE or FALSE, provided it has no free variables.
- HOL-Light implements Hörmander's decision procedure. It is fairly simple, but it hangs if the polynomial's degree exceeds 6.
- Cylindrical Algebraic Decomposition (due to Collins) is still doubly exponential in the number of variables, but it is polynomial in other parameters. We use QEPCAD B (Hoon Hong, C. W. Brown).

# Metis resolution prover

- a full implementation of the superposition calculus
- integrated with interactive theorem provers (HOL4, Isabelle)
- coded in Standard ML
- acceptable performance
- easy to modify
- due to Joe Hurd

# resolution primer

- Resolution provers work with *clauses*: disjunctions of *literals* (atoms or their negations).
- They seek to *contradict* the *negation* of the goal.
- Each step combines two clauses and yields new clauses, which are *simplified* and perhaps kept.
- If the *empty clause* is produced, we have the desired contradiction.

# a resolution step

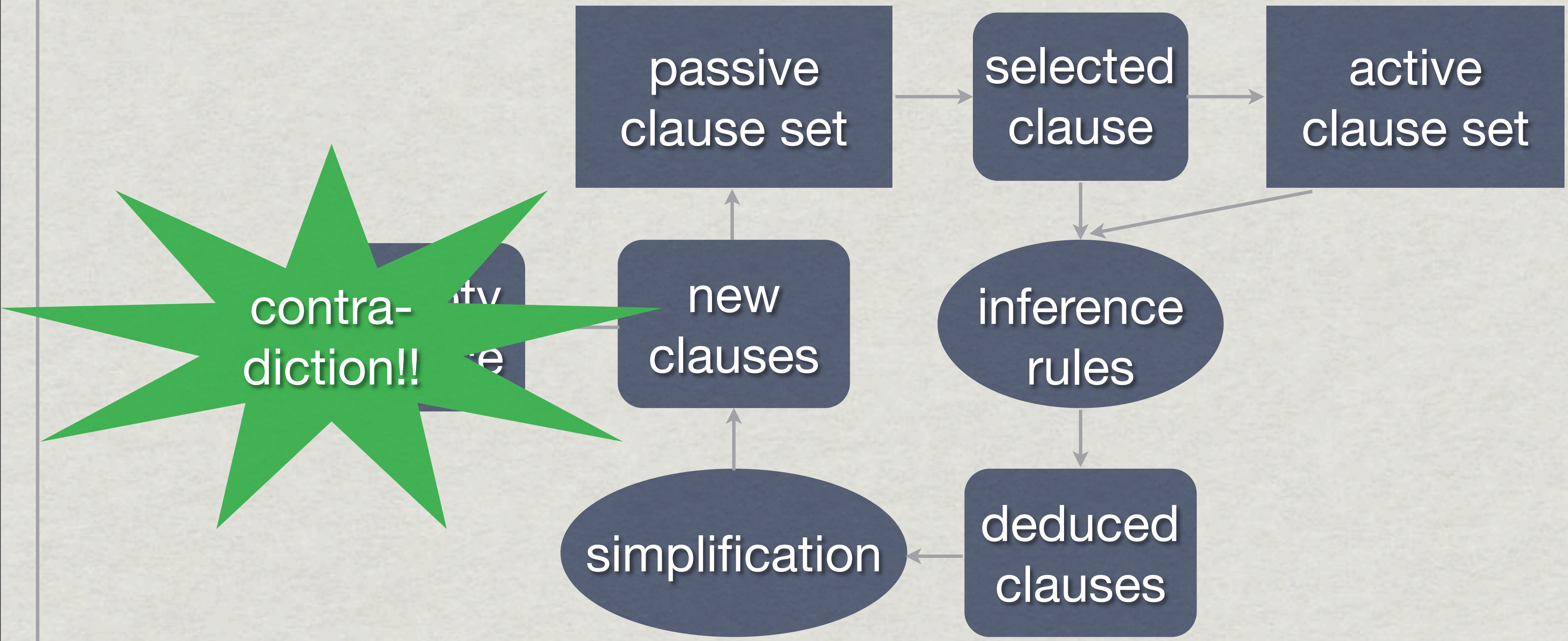
$$R(x,1) \vee P(x)$$
$$\neg R(0,y) \vee Q(y)$$

$$R(0,1) \vee P(0)$$
$$\neg R(0,1) \vee Q(1)$$

$$x \mapsto 0$$
$$y \mapsto 1$$

$$P(0) \vee Q(1)$$

# resolution data flow



# modifications to Metis

- algebraic literal deletion, via decision procedure
- algebraic redundancy test (subsumption)
- formula normalization and simplification
- modified Knuth-Bendix ordering
- “dividing out” products

# algebraic literal deletion

- Our version of Metis keeps a list of all *ground, algebraic clauses* (+ − × ≤, no variables).
- Any literal that is inconsistent with those clauses can be *deleted*.
- Metis simplifies new clauses by calling QEPCAD to detect inconsistent literals.
- Deleting literals brings us closer to the empty clause!

# literal deletion examples

- We delete $x^2+1 < 0$, as it has no real solutions.
- Knowing $xy > 1$, we delete the literal $x=0$.
- We take adjacent literals into account: in the clause $x^2 > 2 \vee x > 3$, we delete $x > 3$.

Specifically, QEPCAD finds $\exists x\,[x^2 \leq 2 \wedge x > 3]$ to be equivalent to FALSE.

# algebraic subsumption

- If a new clause is an instance of another, it is *redundant* and should be DELETED.
- We apply this idea to *ground algebraic formulas,* deleting any that follow from existing facts.
- Example: knowing $x^2 > 4$ we can delete the clause $x < -1 \vee x > 2$.

QEPCAD: $\exists x\, [x^2 > 4 \wedge \neg(x < -1 \vee x > 2)]$

is equivalent to FALSE.

# formula normalization

- How do we suppress redundant equivalent forms such as $2x+1$, $x+1+x$, $2(x+1)-1$? *Horner canonical form* is a recursive representation of polynomials.

$$
\begin{aligned}
&a_n x^n + \cdots + a_1 x + a_0 \\
&\quad = a_0 + x(a_1 + x(a_2 + \cdots x(a_{n-1} + x a_n))
\end{aligned}
$$

*The normalised formula is unique and reasonably compact.*

# normalization example

$$3xy^2 + 2x^2yz + zx + 3yz$$
$$= [y(z3)] + x([z1 + y(y3)] + x[y(z2)])$$

first variable

second variable

- The "variables" can be arbitrarily non-algebraic sub-expressions.
- Thus, formulas containing special functions can also be simplified, and the function *isolated*.

# formula simplification

- Finally we  simplify the output of the Horner transformation using laws like $0+z=z$ and $1\times z=z$.
- The maximal function term, say ln *E*, is isolated (if possible) on one side of an inequality.
- Formulas are converted to *rational functions*:

$$\left(\frac{x}{y}\right)\frac{1}{(x+\frac{1}{x})}=\frac{x^2}{y(x^2+1)}$$

# choosing the best literal

$$x \leq 2 \vee \exp x \leq 2 \vee \frac{1}{x} \leq u$$

*This is the critical one:*
*it is the most difficult!*

*And then this one*
*should be tackled next.*

# Knuth-Bendix ordering

- *Superposition* is a refinement of resolution, selecting the *largest* literals using an *ordering*.
- Since ln, exp, ... are complex, we give them high *weights*. This focuses the search on them.
- The Knuth-Bendix ordering (KBO) also counts *occurrences* of variables, so *t* is more complex than *u* if it contains more variables.

# modified KBO

- Our bounds for $f(x)$ contain multiple occurrences of $x$, so standard KBO regards the bounds as worse than the functions themselves!
- Ludwig and Waldmann (2007) propose a modification of KBO that lets us say e.g. "ln($x$) is more complex than 100 occurrences of $x$."
- This change greatly improves the is performance for our examples.

# dividing out products

- The heuristics presented so far only isolate function occurrences that are *additive*.
- If a function is MULTIPLIED by an expression $u$, then we must divide both sides of the inequality by $u$.
- The outcome depends upon the sign of $u$.
- In general, $u$ could be positive, negative or zero; its sign does not need to be fixed.

# dividing out example

- Given a clause of the form $f(t) \cdot u \leq v \vee C$

- deduce the three clauses $f(t) \leq v/u \vee u \leq 0 \vee C$

$$0 \leq v \vee u \neq 0 \vee C$$

$$f(t) \geq v/u \vee u \geq 0 \vee C$$

- Numerous problems can only be solved using this form of inference.

# notes on the axioms

- We omit general laws: transitivity is too prolific!
  - The decision procedure, QEPCAD, catches many instances of general laws.
  - We build *transitivity* into our bounding axioms.
- We use lgen(R,X,Y) to express both $X \leq Y$ (when $R=0$) and $X<Y$ (when $R=1$).
- We identify $x<y$ with $\neg(y \leq x)$.

# some exp lower bounds

Covers both

< and ≤

```
cnf(exp_lower_taylor_1,axiom,
    ( ~ lgen(R,Y,1+X)
    | lgen(R,Y,exp(X)) )).

cnf(exp_lower_bound_cf2,axiom,
    ( ~ lgen(R, Y, (X^2 + 6*X + 12) /
                   (X^2 - 6*X + 12))
    | lgen(R,Y,exp(X)) )).
```

Transitivity is built in: to show $Y<exp(X)$, show $Y<1+X$.

# absolute value axioms

- Simply |X| = X if X≥0 and |X| = –X otherwise.
- It helps to give abs a high *weight*, discouraging the introduction of occurrences of abs.

```
cnf(abs_nonnegative,axiom,
   ( ~ 0 <= X
   | abs(X) = X )).

cnf(abs_negative,axiom,
   ( 0 <= X
   | abs(X) = -X )).
```

# a few solved problems

| **problem** | **seconds** |
|---|---|
| $\lvert x\rvert < 1 \Longrightarrow \lvert\ln(1+x)\rvert \leq -\ln(1-\lvert x\rvert)$ | 0.153 |
| $\lvert\exp(x)-1\rvert \leq \exp(\lvert x\rvert)-1$ | 0.318 |
| $-1 < x \Longrightarrow 2\lvert x\rvert/(2+x) \leq \lvert\ln(1+x)\rvert$ | 4.266 |
| $\lvert x\rvert < 1 \Longrightarrow \lvert\ln(1+x)\rvert \leq \lvert x\rvert(1+\lvert x\rvert)/\lvert 1+x\rvert$ | 0.604 |
| $0 < x \leq \pi/2 \Longrightarrow 1/\sin^2 x < 1/x^2 + 1 - 4/\pi^2$ | 410 |

# hybrid systems

- Many hybrid systems can be specified by systems of linear differential equations. (The HSOLVER Benchmark Database presents 18 examples.)
- We can solve these equations using Maple, typically yielding a problem involving the exponential function.
- MetiTarski can often solve these problems.

# collision avoidance system

- differential equations for the velocity, acceleration and gap between two vehicles:

$$\dot{v} = a, \quad \dot{a} = -3a - 3(v - v_f) + gap - (v + 10), \quad \dot{gap} = v_f - v$$

- solution for the gap (as a function of *t*):

$$gap = 12 - 14.2e^{-0.318t} + 3.24e^{-1.34t}\cos(1.16t) - 0.154e^{-1.34t}\sin(1.16t)$$

- MetiTarski can prove that the gap is positive!

# some limitations

- No range reduction: proofs about exp(20) or sin(3000) are likely to fail.
- Not everything can be proved using upper and lower bounds. Adding laws like exp(X+Y) = exp(X)exp(Y) greatly increases the search space.
- Problems can have only a few variables or QEPCAD will never terminate.

# example of a limitation

- We can prove this theorem if we replace 1/2 by 100/201. Approximating π by a fraction loses information.

$$0 < \mathrm{x} < 1/2 \Longrightarrow \cos(\pi x) > 1 - 2x$$

# related work?

- SPASS+T and SPASS(T) combine the SPASS prover with various decision procedures.
- Ratschan's RSOLVER solves quantified inequality constraints over the real numbers using constraint programming methods.
- There are many attempts to add quantification to *SMT solvers*, which solve propositional assertions involving linear arithmetic, etc.

# final remarks

- By combining a resolution prover with a decision procedure, we can solve many hard problems.
- The system works by *deduction* and outputs *proofs* that could be checked independently.
- A similar architecture would probably perform well using other decision procedures.

# acknowledgements

* Assistance from C. W. Brown, A. Cuyt, I. Grant, J. Harrison, J. Hurd, D. Lester, C. Muñoz, U. Waldmann, etc.

* The research was supported by the Engineering and Physical Sciences Research Council [grant number EP/C013409/1].