Automatic Proof Procedures for Polynomials and Special Functions

Prof. Lawrence C. Paulson

Computer Laboratory, University of Cambridge

Dr. Paul B. Jackson

School of Informatics, University of Edinburgh

1 Previous Research Track Record

1.1 Highlights

The key prior items of work are the MetiTarski theorem prover (Cambridge) and the RAHD decision procedure (Edinburgh). RAHD[1] is a proof procedure for polynomials over the reals, and MetiTarski additionally is tailored for handling special functions such as log, exp, sin and cos. Both systems are well beyond the first prototype stage, and both have already shown world-leading characteristics [1, 2, 10, 20, 25, 24, 26].

However our experiments have also made us very aware of gap between their current capabilities and what is needed to support formal verification of hybrid systems – the primary application domain we are considering. Our previous work has thrown up a wealth of ideas for enhancing MetiTarski and RAHD, and we put forward this proposal as an opportunity to explore and realise these ideas. Further, we are very excited by the potential of a unified RAHD-MetiTarski system that combines the strengths of each. Hence we are not submitting two separate proposals, but rather this joint proposal to bring work on these two systems together.

1.2 In more depth

Lawrence C. Paulson is Professor of Computational Logic at the University of Cambridge, where he has held established posts since 1983. One of his main activities is developing proof tools. His early work made fundamental contributions to Prof. M. J. C. Gordon's proof assistant, HOL. In 1986, Paulson introduced Isabelle [22], a generic proof assistant. Isabelle supports higher-order logic (HOL), Zermelo-Fraenkel set theory (ZF) and other formalisms. Starting in 1996, he made significant advances in the verification of security protocols. More recently, Paulson developed MetiTarski, an automatic theorem prover for functions such as logarithms and exponentials. In 2008, the ACM designated him an ACM Fellow "for contributions to theorem provers and verification techniques."

Paulson's work has had a major impact. His H-index is approximately 42, which means that he has published over 42 papers each of which has been cited at least 42 times. His top five publications have been cited approximately 3660 times. (Data from Google Scholar.) His work on Isabelle has had a profound impact on research in the UK; for example, Prof Alan Bundy's current EPSRC grant portfolio includes nearly £2 million worth of Isabelle-related projects. Paulson's work on computer security will eventually improve the quality of life of everyone living in the UK, because improved security will reduce the incidence of identity theft and similar crimes.

The project will be done within the Cambridge Automated Reasoning Group. This group has built two of the world's leading proof environments: HOL and Isabelle. Hardware verification was pioneered here by Prof. Gordon and his students. Our HOL technology has become part of the design workflow for major companies such as NVIDIA and Intel. Isabelle is used as a research tool in most of the world's advanced countries.

The EPSRC has funded several projects at Cambridge with Paulson as the principal investigator. Most relevant is *Beyond Linear Arithmetic: Automatic Proof Procedures for the Reals* (EPSRC ref. EP/C013409/1), 2005–08. This project investigated advanced methods of proving theorems about the transcendental functions: log, exp, sin, cos, etc. It delivered the MetiTarski theorem prover [1] and demonstrated its potential for verifying hybrid systems

[1] Real Algebra in High Dimensions

[2] and analogue circuits [10]. MetiTarski is the focus of the Cambridge half of this proposal.

Paul B. Jackson is Lecturer in the School of Informatics at the University of Edinburgh, where he has held established posts since 1995. His research interests since 1988 have been in the development and use of formal verification technologies, principally theorem provers and model checkers. Jackson gained his PhD in 1995 from Cornell University for work which included formalising computer algebra algorithms in the Nuprl interactive theorem prover, and extending the prover with linear arithmetic decision procedures. In 2010, he was elected as a trustee of the Calculemus Interest Group, a leading international consortium of researchers working towards the integration of algorithms for algebraic symbolic computation and mechanical theorem proving.

Most recently, Jackson's work has centred around (i) the development of novel decision procedures for non-linear real arithmetic, and (ii) the use of SMT (Satisfiability Modulo Theories) solvers to verify high-integrity programs.

He has overseen the design and development of the RAHD decision procedure for non-linear real arithmetic by his Ph.D. student Grant Passmore, and has authored the Victor tool for the verification of SPARK Ada programs by high-performance SMT solvers. RAHD is the focus of the Edinburgh arm of this project. Both of these efforts have resulted in robust tools which are being taken up both by other researchers and by industry.

The development of these tools has led to strong research relationships between Jackson, Passmore, Microsoft Research, Altran Praxis, AdaCore, SRI International, and INRIA/IRISA. We plan to continue these collaborations and take full advantage of them to improve the practical impact of this work.

Jackson also has expertise in integrated circuit design and verification that will enable him to stimulate and guide our efforts to apply our work in the field of formal analogue hardware verification. He has an undergraduate studies in electronics, 2 years work experience as an integrated circuit designer, and has regularly taught hardware verification courses and supervised projects at the Institute for System-Level Integration, a partnership of four Scottish universities and the electronics industry.

The EPSRC has funded multiple projects at Edinburgh with Jackson as investigator. He was PI on *Hardware Verification by Combining Model Checking and Theorem Proving Technologies* (GR/N64243/01), 2000–2004, and has been coinvestigator on three platform grants, the two most recent entitled *The Integration and Interaction of Multiple Mathematical Reasoning Processes* (GR/S01771/01, 2002–07 and EP/E005713/1 2007–11).

This project will be done within both the Laboratory of Foundations of Computer Science and the Mathematical Reasoning Group of the University of Edinburgh. Both groups are world-leaders in the theory and practice of decision procedures and mechanical theorem provers.

Jackson is a member of EU COST Action IC0901: *Rich-Model Toolkit - An Infrastructure for Reliable Computer Systems*[2] which started in October 2009 and runs for 4 years. The central topic of this Action is the integration of automated reasoning and synthesis tools to support formal hardware and software verification. The Action funds activities such as visits and meetings to encourage coordination of research between Action members. Members have expertise in SMT solvers and in techniques for non-linear arithmetic reasoning, and have expressed interest in the reasoning services this proposal shall provide. We shall collaborate with other Action members and present our work at Action meetings.

Grant Passmore is currently completing his PhD at the University of Edinburgh. He is Edinburgh's designated Research Assistant and will undertake much of Edinburgh's part of the work plan. He is one of the world's leading experts on RCF (real-closed fields) decision procedures and is the developer of RAHD. With Jackson, he has co-authored a paper on combining Gröbner basis calculations with a restricted variant of cylindrical algebraic decomposition for many-variable non-linear real arithmetic [26]. With de Moura (of Microsoft Research, and the author of the SMT solver Z3) and Jackson, he has co-authored papers on a new breed of Gröbner basis algorithms based on high-performance saturation loops used in resolution theorem provers [24, 25]. These Gröbner basis algorithms significantly out-perform previously available methods for classes of large non-linear polynomial systems arising in industrial program verification, and are being used by Passmore and de Moura as the foundation of new decision methods for non-linear real arithmetic in both RAHD and Z3.

Over the period of his PhD studies, Passmore has held a 5 month Visiting Fellow position at SRI International and a 3 month internship at Microsoft Research. The first supported the development of an initial RAHD prototype and the second this investigation of new Gröbner basis algorithms tailored to formal verification needs. This proposal builds on the close research relationship Passmore has with both these research centres, and we expect fruitful mutually-beneficial collaborations with both over the course of this project.

[2] `http://richmodels.epfl.ch/`

2 Proposed Research

2.1 Background

Highlights

Hybrid systems – control systems and their environments, for example – are ubiquitous in today's technological society and we all depend on their safety and predictability. Formal approaches to verifying that hybrid systems behave as expected promise much stronger guarantees of correctness than the main analysis techniques in use today. However, their practicality is restricted by the current capabilities of underlying automated reasoning engines. The proposed research will radically enhance core reasoning engines and hence enable much improved formal verification of hybrid systems.

The core engines addressed in this proposal involve procedures for proving mathematical assertions involving polynomials and special functions over the reals. The proposal brings together world-class experts in such procedures: Paulson, whose MetiTarski system is virtually unique in handling special functions, and Passmore, whose RAHD system integrates the best of a variety of current approaches for polynomials.

The key idea is to combine these two systems and further enhance each to produce a new system with radically improved capabilities. Further, we plan to collaborate on integrating major components of our new system into SMT solvers. These are the central reasoning components in many current formal verification approaches, but currently they have non-existent or limited capabilities for handling polynomials and special functions.

A critical element of our proposal is that we engage with application domains from the start, to ensure we steer our research in the most fruitful directions: much of our success will depend on targetting classes of problems with particular structure that we can exploit.

In more depth

Automatic decision procedures are central to most practical applications of formal methods. These procedures address problem domains such as propositional logic, uninterpreted functions with equality, arrays and linear arithmetic, which are ubiquitous in applications. Much progress has been made since the pioneering work of Nelson and Oppen [21] and Shostak [34] 30 years ago. Modern methods can solve problems that are orders of magnitude larger. However the problem domains tackled have changed little. Our project concerns automatic tools and techniques for solving classes of problems that have even higher intrinsic complexity: inequalities involving polynomials or real-valued special functions. Such problems are common in hybrid system applications that involve continuously-varying physical quantities.

There will never be a decision procedure for special function inequalities because the problem is undecidable, but there is an automatic theorem prover: MetiTarski [1]. This software system, developed with EPSRC funding, has a unique and novel architecture: it combines a resolution theorem prover (Metis) with a decision procedure for the theory of real closed fields (QEPCAD [3]). The decision procedure simplifies clauses generated by resolution, deleting literals that are inconsistent with other facts. MetiTarski automatically proves theorems such as

$$\begin{aligned}&0 \le x \le 2 \Longrightarrow\\ &\qquad 14.2\exp(-0.318x) -\\ &\qquad\qquad \Big[3.3\cos(1.16x) - 0.16\sin(1.16x)\Big]e^{-1.34x} < 12\end{aligned}$$

This formula, which arises from a collision avoidance system [33], is proved in under 3 seconds. This sort of problem arises in several types of control and engineering applications. We have successfully applied MetiTarski in the verification of two classes of systems:

1. Linear systems described by a linear differential equation. Applying a Laplace transform yields a closed-form solution; we are left to prove an inequality typically involving the exponential, sine and cosine functions. MetiTarski proves many such inequalities easily [2].

2. Analogue circuits, such as tunnel diode oscillators. These models are especially hard to verify since they are described by nonlinear differential equations, and we have approximated them by piecewise linear differential equations in order to obtain a hybrid linear model [10].

MetiTarski delivers explicit proofs that can be checked without performing search. Few tools exist that can solve such difficult problems while delivering evidence of correctness. To the best of our knowledge, nobody else in the world is investigating automated proof procedures for special functions. The most relevant related work involves interval arithmetic [8, 31].

The decision problem for polynomial inequalities (more formally, real-closed fields or RCF [13]) has been known to be solvable since the 1930s. The decision problem is intractable, with doubly exponential complexity [7], but many special cases can be solved with reasonable efficiency. For instance, an existentially quantified variable appearing at most quadratically in a formula can be eliminated in polynomial time.

Such decision method "sweet spots" can routinely solve problems in many more variables than those solvable by general methods. Moreover, decision methods for such fragments can often be combined to solve problems beyond the reach of any individual method. RAHD [26] (Real Algebra in High Dimensions) is an RCF decision procedure which tightly integrates a heterogeneous collection of decision methods operating within their respective "sweet spots," and incorporates a number of heuristics for intelligently combining them. In doing so, RAHD is able to solve problems in many variables, and can be tailored to solve problems of a particular shape; in this way, it can be tuned to particular applications.

Our project will develop MetiTarski and RAHD in conjunction. MetiTarski will be integrated with an interval arithmetic constraint solver. RAHD will replace QEPCAD as the decision procedure component of MetiTarski. Our project will pursue a variety of ideas—such as recent advances in algorithmic algebra and algebraic geometry—for improving the scope and performance of both systems.

Applications we shall explore as sources of problems to direct our research include analogue mixed signal electronics, air traffic control and engineering control systems. All these systems are instances of hybrid systems. We shall also provide specialist automated reasoning services to other formal verification tools closer to the applications. Examples of such tools include the KeYmaera theorem prover for hybrid systems verification, the hybrid-SAL hybrid systems model checker, general purpose interactive theorem provers such as Isabelle, PVS and Coq, verification condition generators for programming languages such as SPARK-Ada, and SMT solvers such as Z3 and Alt-Ergo. Whenever possible, we shall collaborate with experts in the application areas and with developers of these other formal verification tools.

2.2 Research Hypothesis and Objectives

Our research hypothesis is that

> *automatic proof procedures for polynomials and special functions can be made powerful enough to solve engineering problems of realistic size.*

We intend to demonstrate this by improving our existing software tools and by applying them to problems derived from real-world engineering domains. We shall investigate a variety of heuristics and techniques, described below.

The most exciting new developments in the field of decision procedures concern non-linear real arithmetic. Researchers have developed novel decision methods for fragments of RCF. While their approaches vary widely, they generally take the view that full RCF quantifier elimination (as in QEPCAD) is usually overkill: simply deciding the satisfiability of low-degree formulas is often sufficient. Impressive milestones on this path include sums of squares methods based on semidefinite programming [17, 23], methods based on Gröbner bases and ideal saturation [30, 39], virtual term substitution (VTS) with integrated simplification and dimensional reduction [11], combinations of VTS and partial CAD [37], PSPACE methods based on critical point analysis and connected component sampling [4], and powerful techniques based upon combinations of control methods and interval arithmetic [8, 31]. RAHD has been developed with the goal of (i) providing robust implementations and enhancements of these new techniques, and (ii) developing automatic methods for orchestrating their combination.

MetiTarski's chief limitation is that it can only handle problems in a few variables. This limitation stems from QEPCAD, whose complexity in the number of variables is hyper-exponential. While QEPCAD is old and largely unsupported, RAHD is under active development and has been tailored to solve problems generated by MetiTarski. Improvements to RAHD will allow MetiTarski to solve problems in 10 variables, enough to express many deep problems. Integrating MetiTarski with an interval arithmetic solver could relax the restrictions on variables entirely.

The assessment criteria are thus as follows:

- The size and complexity of the formulas processed by MetiTarski and RAHD. Complexity can be measured in several ways, including the number of variables, the maximum polynomial degree and the maximum nesting depth of special functions.
- The scale of the engineering problems that can be modelled and verified using our tools.

The project will produce two main deliverables:

- Substantially improved versions of our software.
- An enormous corpus of test data. We already have thousands of problems; we shall augment this with many application-level problems.

2.3 Programme and Methodology

Cambridge will focus on MetiTarski: refinements to the software, experiments with new techniques and application case studies. Edinburgh will similarly focus on RAHD. Including RAHD as a component of MetiTarski will give the two sites a common vision: to develop automatic procedures for solving mathematical problems that go far beyond the comfortable realm of linear arithmetic.

Programme of work (Cambridge)

The Cambridge work will be done by a Research Assistant and a PhD student, with a significant contribution (20%) from the principal investigator.

Task 1: Developments to MetiTarski. A few specific improvements will dramatically improve MetiTarski's power and performance:

The RCF decision procedure is a crucial component. RAHD will replace QEPCAD; we shall also consider REDLOG [12] and other RCF procedures that emerge in the course of the project. It is easy to allow MetiTarski to invoke multiple decision procedures.

Interval arithmetic constraint solvers [15, 32] are another technology that could solve problems involving special functions. Their strengths and weaknesses are complementary to RCF decision procedures: they are less sensitive to the number of variables, but they often fail for other reasons. We shall integrate an interval arithmetic solver with MetiTarski, where it will play a similar role to the RCF procedure: simplifying clauses and identifying redundant ones. For example, it could simplify the clause $x \leq 1 \vee x \geq 2 \vee \log x > 1$ to $x \leq 1 \vee x \geq 2$, eliminating an occurrence of logarithm.

MetiTarski can support a huge class of mathematically well-behaved functions. It currently uses a fixed set of axioms that describe upper and lower bounds for special functions. These bounds are typically rational functions derived from Taylor or continued fraction expansions. We shall identify bounds for other functions needed for our applications, for example, the Bessel functions $J_v(z)$. We shall also implement a mechanism to introduce bounds of increasing accuracy automatically.

This Task will be done by the PI, supported by the Cambridge Research Assistant.

Task 2: A Front-End for MetiTarski. MetiTarski currently accepts problems in the form of text files. Converting a real-world problem to such a text file requires many tiresome manual steps. This workflow must be automated before MetiTarski will be taken up by outside users. Therefore, we shall write a problem-oriented user interface to MetiTarski. Such an interface could accept a user's problem in a more natural form and output a MetiTarski problem file, or simply call MetiTarski and display the results. The interface will perform the following functions.

- Invocation of an external computer algebra system in order to solve problems that involve differential equations. Currently, the user must transfer the output of a computer algebra system into a MetiTarski input file using copy and paste.
- Support for *range reduction*. A problem involving $\ln t$ where $t \approx 100$ should be transformed by $\ln t = \ln(100) + \ln(t/100)$ because our bounds for $\ln x$ are most accurate when $x \approx 1$. Our front-end could identify the need for range reduction using interval analysis, then apply it.
- The ability to apply *simplifying problem transformations*. One example is to replace a pair of terms $\sin t$, $\cos t$ by a pair of variables x, y constrained by $x^2 + y^2 = 1$, creating a more abstract and often easier problem.
- Support for *inspecting proofs*, based on the Interactive Derivation Viewer [40], and possibly for diagnosing failing proof attempts.

We shall also make MetiTarski more robust, with better error messages and documentation. Our objective is to transform our research prototypes into useful tools for the scientific community. You cannot achieve impact without usability.

Activities in this task will also support the construction of rich interfaces to other formal verification tools, as required by other tasks.

This Task will be done by the Cambridge RA.

Task 3: Case studies using MetiTarski. We shall focus on a suitable engineering application and develop a methodology for verifying that the engineer's specified design goals are met.

Our primary candidate for an application area is analogue and AMS (analogue mixed-signal) circuit verification. Currently steady state, frequency domain or time domain (simulation-based) analysis of analogue circuits must be repeated for a sample of values for fabrication-process parameters and operating conditions (supply voltage, ambient temperature). Much experience is needed to judge when the sample set is sufficient. There are benefits to integrating analogue and digital circuits on SoC (System-on-Chip) devices using the latest process technologies, but analogue design engineers often hold back on this integration and use separate older processes for analogue and AMS circuits: the challenge is that the latest processes have wider variability of process parameters which makes it more difficult to ensure that fabricated analogue and AMS designs will meet their specifications. Formal analogue verification approaches enable process parameters and operating conditions to be handled symbolically rather than numerically; they can address this challenge of wider variability.

This work will be initiated by the Cambridge PhD student, who will investigate existing formal verification methods and the role MetiTarski could feasibly play. Expertise from the Edinburgh PI will also help guide this initial phase. As we then progress to exploring verification of larger designs, we will inevitably reach a point beyond which further progress is only possible after significant improvements to our tools and modelling techniques. Other team members will therefore share in this task, and this task will help shape the direction of other tasks.

An example of an advanced issue we expect to encounter is that differential equations used in designs will not necessarily be linear, and are unlikely to admit closed-form solutions. However, a differential equation that is part of the model of a sound engineering design will naturally not be expected to exhibit chaotic behaviour. Many such equations will pass the Painlevé test [6], and the solution to such an equation can often be expressed in the form of a power series from which upper and lower bounds can be obtained. Equipped with these bounds, MetiTarski could reason about the solution without requiring the equation to be solved explicitly. This is one of many ideas that will allow us to tackle a wide range of problems.

Programme of work (Edinburgh)

The Edinburgh work will be done by a Research Assistant and a PhD student, with significant contribution (20%) from the principal investigator.

Task 4: Developments to RAHD. We shall improve RAHD by using practical problems to guide the development and implementation of new RCF decision techniques. We shall develop methods for combining these techniques to significantly increase the size of real-world problems soluble by RAHD and MetiTarski.

Many new approaches (RCF "sweet spots") have appeared recently in the algorithmic real algebraic geometry community but have not been exploited sufficiently. These include

- virtual term substitution and dimensional reduction methods of Redlog [12],
- PSPACE sampling and critical point methods of SALSA/Raglib [4],
- a new way of computing CADs by complex triangulation, prototyped in Maple [5],
- recent improvements to general CAD in Mathematica [35],
- techniques for paving high-dimensional real solution sets in RealPaver [15],
- interval methods for numerically robust predicates found in RSolver [31], and
- methods for computing Real Nullstellensatz witnesses via SOS and Gröbner bases found in KeYmaera [29].

We shall investigate these published RCF decision methods as well as developing novel techniques tailored to classes of problems encountered in practice. This line of research has the potential to deliver both theoretical and practical advances.

This task will be primarily undertaken by the Edinburgh RA.

Task 5: Integration of RAHD with MetiTarski. As stated in Task 1, RAHD will become the RCF decision procedure for MetiTarski. But there are many other possibilities for obtaining a deep and extensible integration of these two tools.

For example, RAHD allows users a high level of control over the strategies used to combine its decision methods. We shall further develop these control mechanisms in RAHD so that they are easy to exploit by users of MetiTarski.

We shall also investigate new ways in which RAHD can contribute to MetiTarski proof search. Currently, the RCF procedure is used by MetiTarski to simplify clauses by eliminating contextually inconsistent ground arithmetical facts. RAHD computes much global information about the real solution space that could be communicated to MetiTarski.

This task will be undertaken jointly by the Edinburgh RA and the Cambridge PI.

Task 6: Integration of RAHD with SMT solvers. We shall develop a version of RAHD tailored to the needs of SMT solvers. Several research groups producing high-performance SMT solvers have requested that we undertake this work. As many industrial program verification systems use SMT solvers as their theorem proving back-end, this will have immediate impact by allowing more programs with non-linear arithmetical components to be verified.

Some preliminary work has been done in this direction. De Moura and Passmore have given structure-sharing methods for minimising complex Nullstellensatz proofs to 'UNSAT cores' in the context of SMT decision loops [9]. Much further work, both theoretical and practical, must be done to obtain a robust integration of RAHD and high-performance SMT. SMT teams we have had contact with include those for Z3 (Microsoft Research) Yices2 (SRI), CVC3 (New York University and University of Iowa), OpenSMT (Università della Svizzera) and Alt-Ergo (INRIA Saclay - Île-de-France). We expect to actively collaborate with a subset of these teams, perhaps eventually focussing our integration activities on just one or two of these solvers. This task will be undertaken primarily by the Edinburgh RA.

Task 7: Tool Integration and Case Studies with RAHD and MetiTarski. Real polynomial and special function constraints arise in virtually all mathematical sciences. Non-linear decision procedures therefore have numerous applications, and our overriding objective is to ensure that our decision procedures scale up sufficiently.

Many formal verification tools need improved support for non-linear real arithmetic. We shall integrate RAHD and MetiTarski as trusted reasoning components in a number of such tools. We shall then apply these integrations in practice.

Verification domains for which non-linear decision procedures are already finding use include control systems [41], hybrid systems [38], aircraft collision avoidance [28], robot motion planning [14], metric geometry [27], loop invariant generation [19], numerical algorithms [8], physical networks [36], and formalised mathematics (Hales's Flyspeck project [16]).

We shall seek to integrate MetiTarski and RAHD with a number of interactive proof tools, with the assistance of the corresponding development teams. Specifically, we have in mind Isabelle/HOL from Cambridge and Munich, PVS from SRI International and Coq from INRIA. Preliminary integrations of RAHD with PVS and Coq are ongoing and will be extended. We shall also integrate RAHD and MetiTarski with the hybrid systems analysers KeYmaera from Carnegie Mellon and HybridSAL from SRI International. Target applications for which improved non-linear arithmetic reasoning has already been requested include the verification of aircraft collision avoidance algorithms in PVS and KeYmaera, physical layer protocols in HybridSAL and Isabelle/HOL, and Java byte code in Coq. Non-linear reasoning is much needed for verifying SPARK-Ada programs [18], and, as the integration of RAHD with SMT solvers from Task 6 becomes available, we shall also be pursuing this application.

To reach a wider audience, we shall integrate parts of our software with the Sage computer algebra system.[3] Sage is open source and welcomes contributions: QEPCAD, for example, is a component of Sage.

This array of applications will provide us with a rich collection of practical problems of various forms, guaranteeing our software to be effective over broad classes of problems from different domains.

Most the work here will be undertaken by the Edinburgh PhD student and the Edinburgh PI.

Project management

Project management should be straightforward, in view of the clear separation of tasks between Cambridge and Edinburgh. Each site will have weekly project meetings, and there will be joint project meetings every quarter as well as extended visits by project staff to the opposite site. We are already using collaboration technology: project files are managed in a Subversion repository, hosted at Cambridge; we are trialling Google Wave as an informal blackboard to exchange ideas. An equally simple management structure has supported the development of Isabelle by Cambridge and Munich since the early 1990s.

2.4 Relevance to Academic Beneficiaries

The project will deliver robust, documented mathematical software that will support applications in engineering and applied mathematics. The project has the objective of finding applications in control engineering and hybrid systems. Decision procedures for RCF are a crucial component in many reasoning tools, such as KeYmaera [29], a hybrid system verifier. MetiTarski and RAHD will be valuable to research communities in the mathematical sciences. By integrating our software with heavily-used formal verification tools and with the Sage computer algebra system, we shall put our software at the disposal of thousands of users.

Our findings and our vast corpus of test data will be valuable to researchers studying automatic theorem proving and decision procedures. Relevant fields of computer science include formal methods/verification, computer algebra, computational logic and artificial intelligence.

[3] `http://www.sagemath.org`

References

[1] B. Akbarpour and L. Paulson. MetiTarski: An automatic theorem prover for real-valued special functions. *J. Auto. Reas.*, 44(3):175–205, Mar. 2010.

[2] B. Akbarpour and L. C. Paulson. Applications of MetiTarski in the verification of control and hybrid systems. In R. Majumdar and P. Tabuada, editors, *Hybrid Systems: Computation and Control*, LNCS 5469, pages 1–15. Springer, 2009.

[3] C. W. Brown. QEPCAD B: a program for computing with semi-algebraic sets using CADs. *SIGSAM Bulletin*, 37(4):97–108, 2003.

[4] F. Caruso. The SARAG library: Some algorithms in real algebraic geometry. In A. Iglesias and N. Takayama, editors, *Mathematical Software — ICMS 2006*, LNCS 4151, pages 122–131. Springer, 2006.

[5] C. Chen et al. Computing cylindrical algebraic decomposition via triangular decomposition. In *International Symposium on Symbolic and Algebraic Computation*, pages 95–102. ACM, 2009.

[6] R. M. Conte and M. Musette. *The Painlevé Handbook*. Springer, 2008.

[7] J. H. Davenport and J. Heintz. Real quantifier elimination is doubly exponential. *J. Symbolic Comp.*, 5:29–35, 1988.

[8] F. de Dinechin, C. Q. Lauter, and G. Melquiond. Assisted verification of elementary functions using Gappa. In *SAC '06: Proceedings of the 2006 ACM symposium on Applied computing*, pages 1318–1322. ACM, 2006.

[9] L. de Moura and G. O. Passmore. On locally minimal nullstellensatz proofs. In *SMT '09: Proceedings of the 7th International Workshop on Satisfiability Modulo Theories*, pages 35–42, New York, NY, USA, 2009. ACM.

[10] W. Denman, B. Akbarpour, S. Tahar, M. Zaki, and L. C. Paulson. Formal verification of analog designs using MetiTarski. In A. Biere and C. Pixley, editors, *Formal Methods in Computer Aided Design*, pages 93–100. IEEE, 2009.

[11] A. Dolzmann. *Algorithmic Strategies for Applicable Real Quantifier Elimination*. PhD thesis, University of Passau, 2000.

[12] A. Dolzmann and T. Sturm. REDLOG: Computer algebra meets computer logic. *SIGSAM Bulletin*, 31(2):2–9, 1997.

[13] A. Dolzmann, T. Sturm, and V. Weispfenning. Real quantifier elimination in practice. Technical Report MIP-9720, Universität Passau, D-94030, Germany, 1997.

[14] A. Dolzmann and V. Weispfenning. Multiple object semilinear motion planning. *J. Symb. Comput.*, 42(3):324–337, 2007.

[15] L. Granvilliers and F. Benhamou. Algorithm 852: RealPaver: an interval solver using constraint satisfaction techniques. *ACM Trans. Math. Softw.*, 32(1):138–156, 2006.

[16] T. C. Hales. Some methods of problem solving in elementary geometry. In *LICS*, pages 35–40. IEEE Computer Society, 2007.

[17] J. Harrison. Verifying nonlinear real formulas via sums of squares. In K. Schneider and J. Brandt, editors, *Theorem Proving in Higher Order Logics*, LNCS 4732, pages 102–118. Springer, 2007.

[18] P. B. Jackson, B. J. Ellis, and K. Sharp. Using SMT solvers to verify high-integrity programs. In J. Rushby and N. Shankar, editors, *Automated Formal Methods, 2nd Workshop, AFM 07*, pages 60–68. ACM, 2007. Preprint available at `http://fm.csl.sri.com/AFM07/afm07-preprint.pdf`.

[19] D. Kapur. Automatically generating loop invariants using quantifier elimination. In F. Baader, P. Baumgartner, R. Nieuwenhuis, and A. Voronkov, editors, *Deduction and Applications*, volume 05431 of *Dagstuhl Seminar Proceedings*. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany, 2005.

[20] R. Narayanan, B. Akbarpour, M. H. Zaki, S. Tahar, and L. C. Paulson. Formal verification of analog circuit in the presence of noise and process variation. In *DATE, Design, Automation and Test in Europe*, 2010.

[21] G. Nelson and D. C. Oppen. Fast decision procedures based on congruence closure. *J. ACM*, 27(2):356–364, 1980.

[22] T. Nipkow, L. C. Paulson, and M. Wenzel. *Isabelle/HOL: A Proof Assistant for Higher-Order Logic*. Springer, 2002. LNCS Tutorial 2283.

[23] P. A. Parrilo. Semidefinite programming relaxations for semialgebraic problems. *Mathematical Programming*, 96(2):293–320, 2003.

[24] G. O. Passmore and L. de Moura. Superfluous S-polynomials in Strategy-Independent Gröbner Bases. In *SYNASC'09, 11th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*. IEEE Computer Society, 2009.

[25] G. O. Passmore, L. de Moura, and P. B. Jackson. Gröbner basis construction algorithms based on superposition loops. *In preparation*, 2010.

[26] G. O. Passmore and P. B. Jackson. Combined decision techniques for the existential theory of the reals. In J. Carette et al., editors, *Intelligent Computer Mathematics — Calculemus 2009*, LNCS 5625. Springer, 2009.

[27] H. Peyrl and P. A. Parrilo. Computing sum of squares decompositions with rational coefficients. *Theor. Comput. Sci.*, 409(2):269–281, 2008.

[28] A. Platzer and E. M. Clarke. Formal Verification of Curved Flight Collision Avoidance Maneuvers: A Case Study. *Lecture Notes in Computer Science*, 5850:547–+, 2009.

[29] A. Platzer and J.-D. Quesel. KeYmaera: A hybrid theorem prover for hybrid systems. In A. Armando et al., editors, *Automated Reasoning — 4th International Joint Conference*, LNCS 5195, pages 171–178. Springer, 2008.

[30] A. Platzer, J.-D. Quesel, and P. Rümmer. Real world verification. In R. Schmidt, editor, *Conference on Automated Deduction, CADE'09*, LNCS 5663, pages 485–501. Springer, 2009.

[31] S. Ratschan. *RSolver User Manual*. Academy of Sciences of the Czech Republic, 2007. `http://rsolver.sourceforge.net/documentation/manual.pdf`.

[32] S. Ratschan and Z. She. Safety verification of hybrid systems by constraint propagation based abstraction refinement. *ACM Transactions in Embedded Computing Systems*, 6(1), 2007.

[33] S. Ratschan and Z. She. Benchmarks for safety verification of hybrid systems, 2008. `http://hsolver.sourceforge.net/benchmarks/`.

[34] R. Shostak. A practical decision procedure for arithmetic with function symbols. *J. Assoc. Comput. Mach.*, 26:351–360, 1979.

[35] A. Strzebonski. Solving algebraic inequalities in Mathematica. *Mathematica Journal*, Vol. 7, 2000.

[36] T. Sturm. Reasoning over networks by symbolic methods. *Appl. Algebra Eng. Commun. Comput.*, 10(1):79–96, 1999.

[37] T. Sturm. REDLOG as a tool in symbolic algebra and trustworthy computing, 2008. `http://www.is.pku.edu.cn/~xbc/SRATC2008/Thomas-sratc08.pdf`.

[38] A. Taly, S. Gulwani, and A. Tiwari. Synthesizing switching logic using constraint solving. In *Verification, Model Checking and Abstract Interpretation, VMCAI*, LNCS 5403, pages 305–319. Springer, 2009.

[39] A. Tiwari. An algebraic approach for the unsatisfiability of nonlinear constraints. In L. Ong, editor, *Computer Science Logic*, volume 3634 of *LNCS*, pages 248–262. Springer, 2005.

[40] S. Trac, Y. Puzis, and G. Sutcliffe. An interactive derivation viewer. In S. Autexier and C. Benzmüller, editors, *User Interfaces for Theorem Provers*, ENTCS 174, pages 109–123, 2006.

[41] H. Yanami and H. Anai. The Maple package SyNRAC and its application to robust control design. *Future Gener. Comput. Syst.*, 23(5):721–726, 2007.