Prover's Palette: A User-Centric Approach to Verification with Isabelle and QEPCAD-B

Tool Paper

Laura I. Meikle and Jacques D. Fleuriot

School of Informatics, University of Edinburgh, Appleton Tower, EH8 9LE
{lauram,jdf}@dai.ed.ac.uk

Abstract. We present Prover's Palette, a general framework for formal verification using multiple tools, centred around the user. We illustrate the framework by describing a concrete integration of the theorem prover Isabelle with the computer algebra system QEPCAD-B.

1 What Is New

The Prover's Palette is a user-centric approach to integrating theorem provers with external tools whereby the user is provided with a novel way of interacting intelligently with the different systems. Our guiding principle is that integrations should make external tools easy to use in the proof environment, but without limiting the power of the tool (by cutting out functionality) and without limiting the potential audience (by making it too difficult or intrusive). This may entail supporting both automatic and interactive usage — allowing both black-box and glass-box integrations and making it accessible to both novice and expert users. These aims are similar to those of other integration frameworks, in particular the PROSPER project [3], but recent advances in IDE systems mean these aims can now be realized to a greater extent. We build on the Eclipse Proof General Kit [1], a modular communications infrastructure for proof coupled with a rich IDE in the extensive and extensible Eclipse framework. More details on the underlying architecture and comparisons with other systems is presented in a longer paper [5].

The focus of this paper is to illustrate our concept by describing a concrete integration which can be used for non-trivial formal verification tasks involving continuous mathematics in the theorem prover Isabelle [6]. In this setting, QEPCAD-B [2], one of the most sophisticated tools for solving problems in nonlinear algebra, is used to enhance Isabelle by relieving the user from the burden of reasoning deductively about nonlinear arithmetic. Moreover, because the Prover's Palette approach allows external tools to be used in a variety of ways, our QEPCAD integration can provide proof guidance and loop invariant discovery in many situations (see §2.2). It is this versatility which distinguishes this work from other integrations with QEPCAD (*e.g.* the one with PVS [7]); this will be shown by the illustrations in this paper, however for a more comprehensive comparison see [5]. Our framework is available at `www.cognetics.org/proverspalette`.

A. Gupta and S. Malik (Eds.): CAV 2008, LNCS 5123, pp. 309–313, 2008.

2 What Is Possible

In the Prover's Palette, QEPCAD is always available in the IDE (as an Eclipse plugin, contributing a widget as shown in Fig. 1). As the prover subgoal changes, the widget updates automatically, configuring intelligently chosen defaults so that QEPCAD can be used with a single click of a button (and without any previous experience with the tool). When an Isabelle problem is not completely suitable to send to QEPCAD, *e.g.* because it is not in prenex normal form or it contains incompatible types or predicates, the widget warns the user and automatically selects the subset of assumptions and/or conclusion which can meaningfully be sent. The user also has the option to try an automatic conversion of such subgoals to ones which can be sent in their entirety to QEPCAD. Furthermore, the user can adjust the full range of QEPCAD operating parameters and select how the QEPCAD result should be used in the prover. This allows QEPCAD to be used in many different ways: as a trusted oracle, as an untrusted assistant giving insight, or even as a stand-alone client. We illustrate the first two uses through examples taken from our verification of Graham's Scan (GS) algorithm for finding convex hulls [4]. The algorithm relies heavily on the notion of "left turn", where pqr means that the point r lies to the left of the directed line from p to q, and can be defined in terms of Cartesian coordinates:

$$pqr \equiv (q_x - p_x)(r_y - p_y) - (q_y - p_y)(r_x - p_x) > 0$$

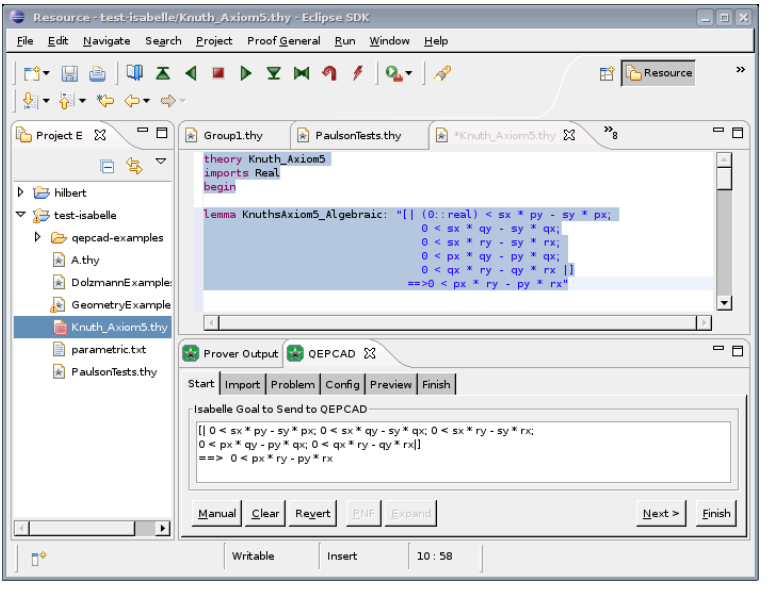

Fig. 1. Isabelle Eclipse Proof General with QEPCAD Proof Palette Widget

2.1 QEPCAD as an Automated Oracle

Using Hoare Logic to verify the GS algorithm in Isabelle entailed proving many difficult subgoals. However, using our new Isabelle/QEPCAD framework, we obtain instant help. For instance, many subgoals can quickly be checked for validity, and if the user is willing to trust QEPCAD, the integration can automatically apply any simplified result to the current proof. As an example, consider:

$$tsp \wedge tsq \wedge tsr \wedge tpq \wedge tqr \longrightarrow tpr$$

This is well known to be true, although it is not easy to prove in Isabelle alone. Taking t to be the origin (this is not necessary, but QEPCAD runs more quickly with it), we get the Isabelle lemma shown in Fig. 1 (top box). Our QEPCAD widget monitors the proof state, and whenever a subgoal looks amenable to proof using QEPCAD, the "Finish" button is enabled, indicating to the user that QEPCAD should be able to finish solving this subgoal. Clicking this button sends the translated problem to QEPCAD[1]. When a result is found — a matter of milliseconds in this case — the "Finish" tab is displayed (see Fig. 2). Selecting "Oracle" will generate the appropriate Isabelle command for the result to be trusted. In this example, the Isabelle lemma is then proved.

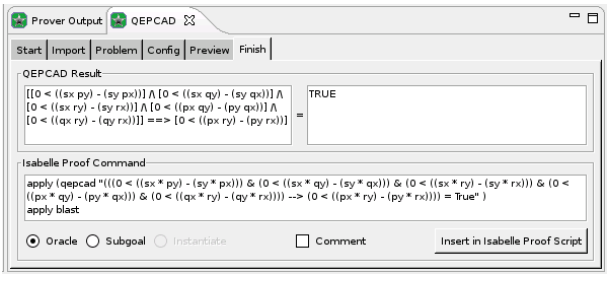

Fig. 2. QEPCAD Finish Tab

2.2 Formal Correctness: QEPCAD as Guide and Discoverer

For some applications, formal correctness requirements might disallow the use of QEPCAD as an oracle. Nevertheless, it can still be a boon to the human prover, giving insight into the problem or providing a subgoal simplification. As an example, consider the case when there are superfluous assumptions in a goal. These can obscure the relevant facts (a common difficulty of interactive proof). Our QEPCAD widget can be used to find a minimal set of assumptions which entail the conclusion of the goal. This enables many problems to be simplified without reference to an untrusted system in the formal proof.

Our framework can also guide verifications by aiding the discovery of loop invariants, a task generally accepted as non-trivial. From our experience in verifying GS, we observed that our initial loop invariant needed several refinements

[1] QEPCAD runs in the background, so a user is not blocked from using the prover.

– a process guided by failed proof attempts. Often, the root cause was a missing assumption, but identifying this was hard. One lemma we encountered was:

$$bea \wedge abd \wedge cab \wedge ade \longrightarrow ace$$

Using our new framework, the QEPCAD widget quickly tells us this is false. In this particular case, we know from the context that a subgoal similar to this is required. By using this integration interactively, we can easily identify what is missing. As QEPCAD eliminates bound variables from a problem, it can be used to yield an equivalent result in terms of the free variables only: this result may reveal useful information. By default all variables are bound when sent to QEPCAD. There is some art in selecting which variables should be free. The "Import" tab allows the user to do this interactively. In this example, a and b are used the most, so we translate a to be the origin and heuristically choose to keep b bound. With the other variables free, QEPCAD then returns:

$$d_y c_x \text{ - } d_x c_y \geq 0 \vee d_x e_y \text{ - } d_y e_x \leq 0 \vee e_y c_x \text{ - } e_x c_y > 0$$

The second and third disjuncts are unenlightening (a negated assumption and the conclusion), but the first disjunct, however, is a hitherto missing condition to our Isabelle lemma. QEPCAD has told us that the lemma can be proven given $\neg adc$ with a at the origin. With this new fact, the framework has therefore led us to the discovery of a missing component in the loop invariant.

3 What Is Planned

We plan to continue to use the Prover's Palette to verify algorithms in computational geometry, and, as we discover the need, to develop new integrations and refine existing ones. With QEPCAD, we have noticed that although it is a powerful tool, it has one major drawback: its methods can have double exponential complexity. For some problems, performance can be improved by exploiting symmetries to reduce the number of variables, or using QEPCAD's specialised quantifiers. Currently, a user can set these quantifiers manually – which we feel is a benefit of the Prover's Palette – but it may be more desirable to automate this. We are identifying when these translations can be applied in a formally correct way. We also intend to use the software to produce witnesses where applicable: a user will then be able to use these values in a fully formal proof, even though they come from a tool which does not provide any formal proofs of its results.

Acknowledgements. We would like to thank the reviewers for their useful comments. This work was funded by the EPSRC grant EP/E005713/1.

References

1. Aspinall, D., Lüth, C., Winterstein, D.: A Framework for Interactive Proof. In: Kauers, M., Kerber, M., Miner, R., Windsteiger, W. (eds.) MKM/CALCULEMUS 2007. LNCS (LNAI), vol. 4573, pp. 161–175. Springer, Heidelberg (2007)

2. Brown, C.W.: QEPCAD B: a program for computing with semi-algebraic sets using CADs. SIGSAM Bulletin 37(4), 97–108 (2003)
3. Dennis, L.A., Collins, G., Norrish, M., Boulton, R., Slind, K., Melham, T.: The PROSPER Toolkit. Int. J Software Tools for Technology Transfer 4(2) (2003)
4. Meikle, L.I., Fleuriot, J.D.: Mechanical Theorem Proving in Computational Geometry. In: Hong, H., Wang, D. (eds.) ADG 2004. LNCS (LNAI), vol. 3763, pp. 1–18. Springer, Heidelberg (2006)
5. Meikle, L.I., Fleuriot, J.D.: Combining Isabelle and QEPCAD-B in the Prover's Palette. In: Proceedings of Calculemus (to appear, 2008)
6. Paulson, L.C. (ed.): Isabelle: A Generic Theorem Prover. LNCS, vol. 828. Springer, Heidelberg (1994)
7. Tiwari, A.: PVS-QEPCAD, `www.csl.sri.com/users/tiwari/qepcad.html`