Toward Nitpick and Sledgehammer for Coq

Jasmin Christian Blanchette[1,2]

[1] Inria Nancy & LORIA, Villers-lès-Nancy, France
[2] Max-Planck-Institut für Informatik, Saarbrücken, Germany

My overall goal is to make proof assistants easier to use, by increasing their automation and expressiveness. As part of my Ph.D., I developed or co-developed two tools for Isabelle/HOL: the counterexample generator Nitpick and the proof tool Sledgehammer.

Nitpick builds on the third-party SAT-based first-order model finder Kodkod, and handles idioms such as (co)inductive predicates, (co)datatypes, and (co)recursive functions specially. The SAT-based approach is quite different from the testing-based approach (e.g., Quickcheck and QuickChick). Neither is clearly superior to the other, and there are problems that clearly benefit from a SAT solver. For example, in joint work with Cambridge, I applied Nitpick to debug a specification of the C++ memory model (which revealed some flaws in the Isabelle formalization).

Sledgehammer integrates third-party automatic theorem provers such as CVC4, E, SPASS, Vampire, veriT, and Z3. It heuristically selects a few hundred facts (lemmas, definitions, etc.) from Isabelle's libraries, translates them to first-order logic along with the conjecture, delegates the proof search to external provers, and produces a (one-line or multi-line) textual Isar proof that rechecks the proof in Isabelle.

My foremost vehicle is Isabelle, but I am also interested in Coq, which is probably the more widely used of the two but (perhaps due to its richer logic) lacks many of the conveniences offered by its rival. In the Coq world, there is plenty of room for tools like Nitpick and Sledgehammer. I have a few sketchy ideas which I would like to discuss at the Coq workshop.

On the counterexample generation side, Inria has accepted my project proposal to develop Nitpick's successor, tentatively called Nunchaku. The idea for Nunchaku emerged in 2013, to address Nitpick's main limitations: (1) It is too tied to its frontend, Isabelle/HOL; (2) it is too tied to its backend, Kodkod. The idea with Nunchaku is to allow many frontends, including Isabelle and (a fragment of) Coq, and many backends, notably SMT solvers. Nunchaku itself would be a standalone application. Arthur Charguéraud is a partner for the Coq frontend.

On the proving side, my ideas are more sketchy. SMTCoq shows a lot of promise, but it has not yet reached the point where most end users can benefit from it. With Sledgehammer, we noticed that three ingredients are necessary to succeed: (1) a heuristic relevance filter to select a few hundreds of lemmas to include in the generated problems; (2) a translation module that can cope, locally, with higher-order constructs; (3) a reconstruction module that translates the essence of the external proof into something that can be inserted in the user's formalization. Coq's logic poses particular challenges, but I am hopeful that something can be done for a practically relevant fragment of Coq, with classical axioms and no or few dependent types.