VGC'05 Panel on Deduction

N. Shankar

shankar@csl.sri.com

URL: http://www.csl.sri.com/~shankar/

Computer Science Laboratory
SRI International
Menlo Park, CA

The Relevance of Automated Deduction

The verification challenge ranges from discharging assertions to proving correctness.

Unlike previous panels, we are focused on proof methods for expressive theories.

Proofs require step-by-step reasoning and problem decomposition rather than brute-force search/propagation.

Current automation can check proofs at an rigorous, informal level of discourse, and solve the occasional open problem.

Prediction: Over the next fifteen years, we will be able to automate the bulk of the verification task through the use of static analysis, decision procedures, model checking, proof strategies, and libraries.

Varieties of Deduction

Many dimensions to classify automated deduction systems:

Logic: Quantifier-free first-order logic, quantified first-order logic, higher-order logic, type theories, non-standard logics.

Kind of Automation: Mechanical uniform search methods versus human-oriented problem reduction (induction, simplification).

Degree of Automation: Interactive, tactic-oriented proof checking versus autonomous. Built-in decision procedures, support for libraries.

Interfaces: For adding low-level automation, and for embedding theorem proving within other applications.

Issues Facing Users

“Why can’t I express . . . ?”: Formalizing mathematical ideas is quite tricky.

“Why isn’t this proof going through?”: Is the theorem incorrect or the theorem prover on the wrong track, or both? Lack of feedback and counterexamples.

“This is obvious to me, why isn’t it obvious to the theorem prover?”: Need lots of special-purpose automation.

“Why do I need to provide all the background definitions/theorems?”: Library development and maintenance is a tough challenge.

“I made some small changes, and my proofs don’t work anymore.”: Need proof strategies and automation that are robust.

The Panelists

- **J Moore (U. Texas):** Boyer–Moore induction provers (Thm, Nqthm, ACL2), CLI Stack.
- **Deepak Kapur (UNM):** RRL and number of techniques for rewriting, induction, and proof search.
- **Jose Meseguer (UIUC):** OBJ family, Maude rewriting logic/engine, reflection.
- **Carsten Schürmann (Yale)**: Metalogical frameworks, formal digital libraries.
- **John Harrison (Intel):** HOL-Lite, Tactical theorem proving, Floating-point algorithms,
- **Konrad Slind (Utah):** HOL4, PROSPER, Program verification.