Research and cooperations related to COST Action IC0901 Rich-model Toolkit

An Infrastructure for Reliable Computer Systems

Maria Paola Bonacina

Dipartimento di Informatica
Università degli Studi di Verona
Verona, Italy

October 30, 2009

People

Motivation

Research Interests

People at the Università degli Studi di Verona

- Maria Paola Bonacina
- Moa Kristin Johansson, postdoc
  (PhD University of Edinburgh June 2009)
- Nicola Zantedeschi, MS student

Some recent or ongoing collaborations

- Alessandro Armando, Università degli Studi di Genova
- Massimo Benerecetti, Università degli Studi di Napoli Federico II
- Nachum Dershowitz, Tel Aviv University
- Mnacho Echenim, Institut National Polytechnique de Grenoble
- Silvio Ghilardi, Università degli Studi di Milano
- Leonardo de Moura, Microsoft Research, Redmond
- Chris Lynch, Potsdam University, New York State
- Roberto Sebastiani, Università degli Studi di Trento

Motivation: reliability of computer systems

- Computer systems: both software and hardware
- Software/hardware border: blurred, evolving
- They are everywhere
- Needed: *Reliability*
- Difficult goal: e.g., software may be
  - Complex
  - Huge
  - Varied (many programming languages, paradigms, styles)
  - Artful (programmers are smart)
  - Old (and undocumented)

Some approaches to software reliability

- Program testing
- Defect checking
- Program checking
- Program analysis
- Software verification

Of course, they overlap

Some technologies for software reliability

- Automated test case generators
- Programmer assistants
- Program analyzers
  - Static analysis (types, extended static checking, abstract interpretations ...)
  - Dynamic analysis (traces ...)
- Software model checkers + *theorem proving*
  - CEGAR-SMC
  - SMT-BMC

Systems with embedded reasoning

Typical architecture:

- *Front-end*: interface, problem modelling, compiling
  - From programs to formulæ
    (via specifications, annotations, intermediate languages)
- *Back-end*: problem solving by **reasoning engine**
  - **Problem**: determine *satisfiability* of formulæ
  - **Objective**: decision procedures

Reasoning about software

Problem statement

- Decide *satisfiability* of first-order formulæ generated by SW verification tools
- Satisfiability w.r.t. *background theories*
- With *quantifiers* to write, e.g.,
  - frame conditions over loops
  - auxiliary invariants over heaps
  - axioms of *type systems* and
  - *application-specific theories* without decision procedure
- Emphasis on *automation* (or interaction with a longer cycle)

Balancing *expressivity* and *efficiency*

Typical shape of problem

- Background theory $\mathcal{T}$
  - $\mathcal{T} = \bigcup_{i=1}^{n} \mathcal{T}_i$, e.g., linear arithmetic
- Set of formulæ: $\mathcal{R} \cup P$
  - Axiomatized theory $\mathcal{R}$: *non-ground* clauses without $\mathcal{T}$-symbols
  - P: large ground formula (ground clauses) with $\mathcal{T}$-symbols
- Determine whether $\mathcal{R} \cup P$ is *satisfiable* modulo $\mathcal{T}$
  (Equivalently: determine whether $\mathcal{T} \cup \mathcal{R} \cup P$ is *satisfiable*)

Tools

- Davis-Putnam-Logemann-Loveland (DPLL) procedure for SAT
- $\mathcal{T}_i$-solvers: *Satisfiability procedures* for the $\mathcal{T}_i$'s
- DPLL($\mathcal{T}$)-based SMT-solver: *Decision procedure* for $\mathcal{T}$ with
  - *Nelson-Oppen combination*
  - *Model-based theory combination*

  of the $\mathcal{T}_i$-sat procedures
- First-order engine Γ to handle $\mathcal{R}$ (additional theory): Resolution+Rewriting+Superposition: *Superposition-based*

Combining strengths of different tools

- DPLL: SAT-problems; large non-Horn clauses
- Theory solvers: e.g., ground equality, linear arithmetic
- DPLL($\mathcal{T}$)-based SMT-solver: efficient, scalable, integrated theory reasoning
- Superposition-based inference system Γ:
  - Horn clauses, equalities with *universal quantifiers* (*automated* instantiation)
  - Sat-procedure for several theories of data structures

Rewrite-based satisfiability procedures

- *Satisfiability problem:* set of ground $\mathcal{R}$-literals
- $\mathcal{R}$: axiomatized theory
- Termination results: $\mathcal{R}$-sat procedures based on generic reasoning
- *Modularity theorem* for combination of theories
- Experiments on $\mathcal{R}$-sat problems with the E prover *taken off the shelf* and optimized for very different search problems

Joint work with Alessandro Armando, Silvio Ranise and Stephan Schulz at PDPAR@CAV 2005, FroCoS 2005, and ACM ToCL Jan. 2009.

Joint work with Mnacho Echenim at PDPAR@FLoC 2006.

Rewrite-based decision procedures

- *Decision problem:* set of ground $\mathcal{R}$-clauses
- $\mathcal{R}$: axiomatized theory
- *Generalization* of termination results:
  $\mathcal{R}$-decision procedures based on generic reasoning
- *Decision by stages*: pipeline of FOL+= prover and SMT-solver

Joint work with Mnacho Echenim at
STRATEGIES@FLoC 2006 and JLC Feb. 2008
CADE 2007 and JSC 2009 (in press).

SMT(FOL): DPLL(Γ+$\mathcal{T}$)

- DPLL(Γ+$\mathcal{T}$) where $\mathcal{T} \neq \emptyset$: refutational completeness + combination of both built-in and axiomatized theories
- DPLL(Γ+$\mathcal{T}$) + *speculative inferences*: termination
- Decision procedures for type systems with multiple/single inheritance used in ESC/Java and Spec#

Joint work with Chris Lynch and Leonardo de Moura at CADE 2009, journal version in preparation.

Building on top of joint work with Silvio Ghilardi, Enrica Nicolini, Silvio Ranise and Daniele Zucchelli at IJCAR 2006.

Thanks

Looking for more

- friends to work with, including post-doc's, students,
- problems, applications, theories to try ...

Thanks!