# BUILDING BRIDGES

C. Scharff

Assistant Professor

Pace University, NY

## Building Bridges in Research in Automated Deduction

- Ideas of Uniform Proof Search Procedures into Problem-Specific Decision Procedures

  – e.g. from completion to congruence closure

## Direct Combination of Completion and Congruence Closure (1)

- Joint work with L. Bachmair
- Word problem in the ground case: Let $E$ and $s \approx t$ be ground. Is $s \approx t$ true in all models of $E$?
- Completion: Uniform Proof Search procedure based on Rewriting
- Congruence Closure: Problem-specific decision procedure
- Ground Completion is in general not as efficient as Congruence Closure

  – Ground completion in O(nlog(n)) [W. Snyder, 1993]

## Combination of Completion and Congruence Closure (2)

- Keys ideas of completion integrated in a graph-based Congruence Closure method where inference rules are expressed as graphs transformations

  – SOUR Graphs + Abstract Congruence Closure = SER graphs

- Obtaining a rewriting system on an extended signature:

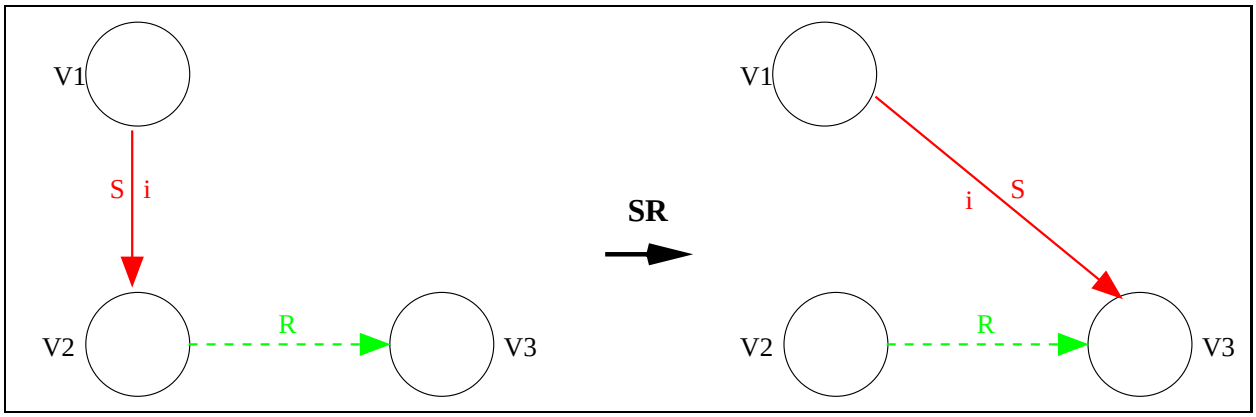
  – Implemented by Graph Transformations

  – Orient (an equality)

  – SR, RR (Critical Pairs / Simplification)

  – Merge (Closure under congruence)

- Example of the SR rule:




- Obtaining a rewriting system on the initial signature
  - Implemented by Graph Transformations
  - Compression and Selection rules

- Implementation available at:

  http://www.unitedthinkers.com/cc (Eugene Kipnis)

## Developing Theorem Provers

- Theorem Prover = Inference Rules + Strategy

- Inference Rules

- Strategy:

  – Where is the inference rule applied in the term?

  – How to apply the inference rules?

  – Expansion versus contraction/simplification strategy

  – How do we choose the strategy?

  – How do we implement the strategy?

  – What about changing the strategy?

  – Completeness?

- Implementing a theorem prover from scratch?

- Prototyping a theorem prover using a rewriting language:

  – Rewriting Program = Rewriting Rules + Strategy

  – e.g. ELAN, MAUDE, ASF+SDF

  – Interest: Change the strategy without changing the rules

- Testing and benchmarking a theorem prover

## Future plans in the Infer! project

- Import/Export ideas to/from other communities

- Techniques for integrating decision procedures within rewriting and uniform proof search methods

- Development of interfaces for combining inference procedures

## Building Bridges in teaching Automated Deduction

- We are facing the same problems in teaching automated deduction as in research in automated deduction.

## Introducing Automated Deduction in the curricula

- How to teach automated deduction?
- How to integrate automated deduction in courses? – Connections
- Undergraduate and graduate curriculum
- Motivate the students
- Formal methods in practice
  - Small examples w.r.t. examples of real/industrial applications
  - Choice of tool(s) to be used
  - Lectures and labs

## Pace University Environment

- NSF CCLI EMD grant of S. Skevoulis: Integrating Formal Methods Tools into the Undergraduate Curriculum
  - Use of the Z notation in a course called *Formal notations*
- Software Engineering Master Program
  - Mathematical Modeling of Software Artifacts
  - Use of Z
- CAFME (Center for the Advancement of Formal Methods in Education)

## Integrating Mathematical Thinking into the CS curricula

- D. Baldwin, P. Henderson

  http://cs.geneseo.edu/∼baldwin/math-thinking

- Mathematical Reasoning: Applying mathematical techniques, concepts and processes, either explicitly or implicitly, in the solution of problems – in other words, mathematical modes of thought that help us to solve problems in any domain. In its most general interpretation, every problem-solving activity requires mathematical thinking. For example, basic logic, be it used explicitly or implicitly is required for all problem-solving activities.

- "Materials Development in Support of Mathematical Thinking", the report of an ITiCSE 2002 working group chaired by Peter Henderson, in SIGCSE Inroads, June 2003.

- Repository of problems showing the connections between CS and mathematics to be used by instructors

  Examples:

  – Truth tables and if-then-else statements

  – Select in databases implemented using linear search

- Repositories: CITIDEL, CSTC, JERIC for Computer Science instructors and Math Forum, GEM, NSDL, Cut the Knot for Mathematics instructors

- Lots of educational theorem provers e.g. Logic Tutor, Oliver for propositional logic

- Communications of the ACM, September 2003 section on "Why Universities Require Computer Science Students to Take Math."

## Future Goals: Curricula at Pace University

- Undergraduate, Software Engineering, Spring 2004
  - Software verification
  - Use of tool(s) for support
- Undergraduate, Software Quality, Spring 2005
  - Best practices in Programming
  - Validation versus Verification
  - Use of tool(s) for support
- Graduate, specification and verification, Fall 2004
  - Use of PVS