Towards comparing transformation systems and formalisms

Torbjörn Ekman
Görel Hedin

Department of Computer Science
Lund University, Sweden
Our background: ReRAGs
Rewritable Reference Attributed Grammars

- **Mechanisms**
  - Equations that define attribute values, reference attributes, conditional rewrites that may use attributes, object-oriented, aspect-oriented

- **Implementation**
  - demand-driven evaluation engine that can interleave attribute evaluation with rewrites

- **Advantages**
  - Allows highly context-dependent stepwise transformation, while being completely declarative
Large benchmark: Java 1.4 compiler

- Completeness: Runs 3932 of the 4006 semantic analysis Jacks testcases
- Size: comparable to JLS and javac
- Performance: around 4 times slower than javac
We need small benchmarks that many tool builders can implement

- How does my system compare to others?
- Users can select system for given problem
- Learn by comparing solutions
What benchmarks?

- Translation
  - Compilation, optimization, refactoring, preprocessing, aspect weaving, …

- Analysis
  - Metrics, name analysis, type checking, …

- Size?
  - Small languages (Tiger, …), micro benchmarks for illustrating specific things, …
What to compare?

- Specification size (lines, tokens, gzipped number of bytes, …)
- Transformation performance
- Generation performance
- Modularity, extensibility
- Character of specification (pass-oriented, imperative, declarative, …)
How define a benchmark?

- Suite of input-output test cases
- Formats: Text, XML, …
Collective effort

• Open process over several years
• What do we have already?
• Network for contributing test suites, ideas, and implementations
• Evolve over time
LDTA 2005
Language Descriptions, Tools and Applications

• Satellite event of ETAPS 2005
  April 3, 2005
  Edinburgh, Scotland, UK

• Submission deadline: December 1, 2004

• www-i2.informatik.rwth-aachen.de/Events/LDTA2005/