Efficient First-Order Theorem Proving

Kryštof Hoder

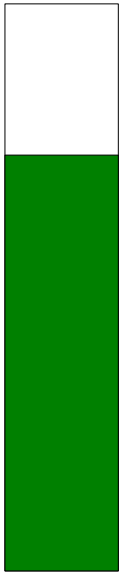
Supervisor: Andrei Voronkov

First-Order Theorem Proving

- Input: first-order formulas (theory & conjecture)
- Conjecture negated, formulas converted to CNF
- Generating inference rules applied to clauses to get new ones
  - until we obtain refutation, or no rule is applicable
- Leads to large amount of clauses
  - simplification and deletion rules reduce it
  - indexing helps us handle it

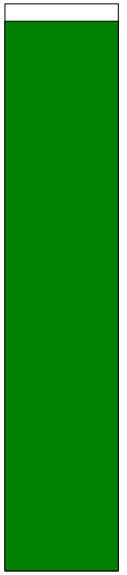
Vampire Theorem Prover

- Consistently winning CASC FOF and CNF 2002-09
- New version (being) developed
  - completely rewritten
  - framework for the research
    - allows immediate applications of intermediate results
  - already helpful on CASC



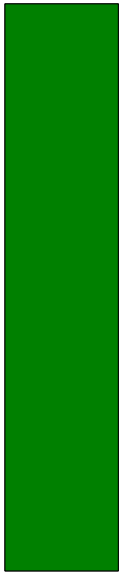
Unification in Term Indexing

- A paper accepted to the KI 2009 conference
  - together with A. Voronkov
  - presented also at the Automated Deduction Seminar in Dagstuhl 2009
- Comparison of unification algorithms in substitution tree indexes
  - different from term-to-term unification
  - building a large substitution in small steps
- Robinson algorithm shown the best
  - our polynomial modification was almost the same (now used in Vampire)



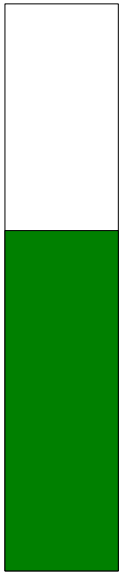
Interpolant Generation

- A paper submitted to the TACAS 2010 conference
  - together with L. Kovacs and A. Voronkov
- Formulas A, B such that $A \vdash B$, Interpolant I is a formula such that $A \vdash I$, $I \vdash B$, and $I \in \mathcal{L}_A \cap \mathcal{L}_B$
  - Useful for software verification
- Interpolant generation and symbol eliminating inferences output implemented in Vampire



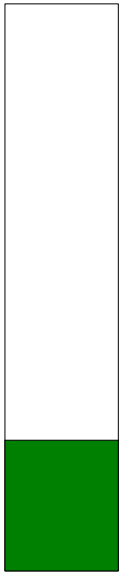
Propositional Reasoning

- First-order not efficient for propositional
- “Clauses” have prop. and non-prop. parts
  - prop. parts might not necessarily be clauses
    - merging common non-propositional parts (→ conj.)
- Representing the prop. part
  - BDD
    - less efficient, allow for more operations
  - SAT solver
    - faster, only checks for unsatisfiability
    - suitable for “empty” clause
- Indexes and prop. parts



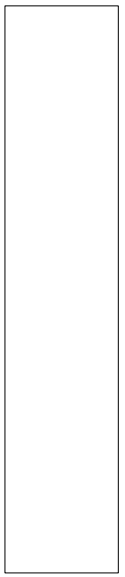
Offline Data Structures

- Axiom selection for large knowledge bases
- Some are too large even for that (memory)
  - DBpedia – 4.7 bln pieces of information
- SInE axiom selection algorithm
  - winner of CASC LTB division
- Transform into offline algorithm
  - building (maintaining) index
  - retrieval of selected axioms



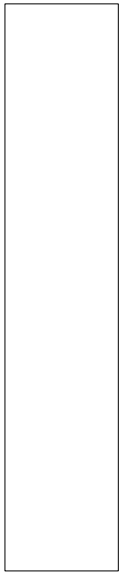
Compiling of Term Indexes

- Code trees
  - Set of clauses compiled to a sequence of byte-code instructions
  - Retrieval from index = execution of the code
- Byte code → native
  - for Java 5–20 times speed-up
  - we need to deal with index modifications
- (dealing with prop. clause parts)



Parallelization

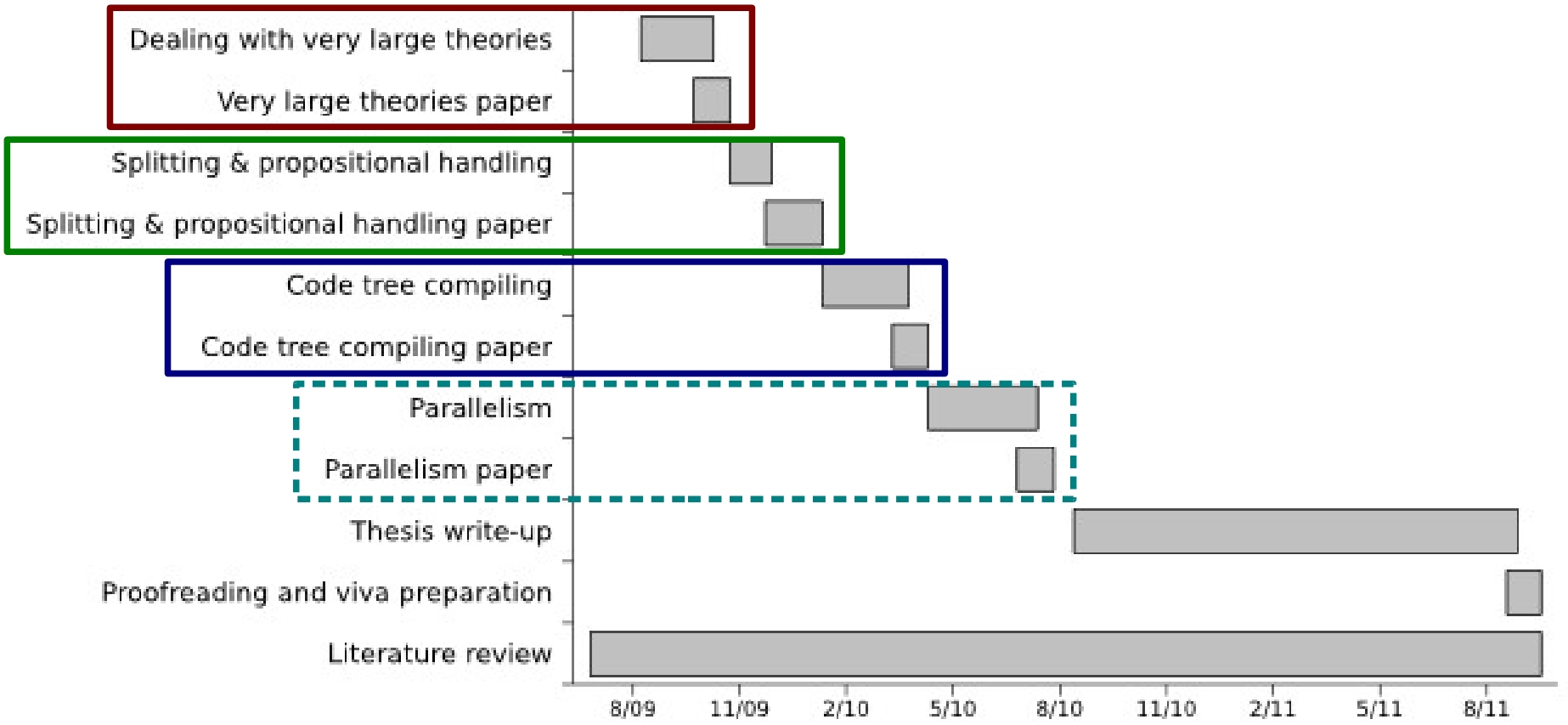
- Growing CPU core number, not speed of a core
- Some tasks should be easily parallelizable
  - retrieval from indexes
    - usually the most expensive
    - read-only access to the index structure
- Multiple Vampire instances can share derived clauses



Evaluation

- TPTP library
  - around 10,000 first-order problems
  - from different areas
  - standard format
    - easy comparison with other first-order provers
- Very large KB problems
  - we expect benchmarks to appear
  - some TPTP problems could be joined with large KB
- Keep winning the CASC competition :)

Plan


Dealing with very large theories
Very large theories paper
Splitting & propositional handling
Splitting & propositional handling paper
Code tree compiling
Code tree compiling paper
Parallelism
Parallelism paper
Thesis write-up
Proofreading and viva preparation
Literature review
8/09
11/09
2/10
5/10
8/10
11/10
2/11
5/11
8/11

Thank you for your attention