iProver v3.5 (SMT-COMP)

Konstantin Korovin, André Duarte and Edvard K. Holden
The University of Manchester, UK
{konstantin.korovin|andre.duarte|edvard.holden}@manchester.ac.uk

iProver is a theorem prover for quantified first-order logic with support for arithmetical reasoning. For quantified reasoning iProver interleaves instantiation calculus Inst-Gen \[5,11\] with ordered resolution and superposition calculi \[18\]. First-order clauses are exchanged between calculi during the proof search. Clauses generated during the proof search are grounded and submitted into SAT and SMT solvers for ground reasoning. In turn, instantiations in the Inst-Gen calculus are guided by propositional models. The method behind iProver is refutationally complete for pure first-order logic with equality.

iProver is implemented in OCaml. It natively accepts typed first-order formulas with arithmetic TFF0 \[19\] in CNF, with extensions it accepts SMT2, TPTP FOF/TFF0, AIG, QBF among other formats. In SMT iProver supports all combinations of quantifiers, uninterpreted functions, data types, linear and non-linear arithmetic: UF, UFDT, LIA, LRA, NIA, NRA, UFDTLIA, UFDTLIRA, UFDTNIA, UFDTNIRA, UFIDL, UFLIA, UFLRA, UFNIA. iProver supports arbitrary precision arithmetic. iProver integrates Z3 \[16\] and MiniSAT \[4\] for ground reasoning during the reasoning phase and uses Vampire \[14\] for clausification and axiomatisations of theories \[17\] which is done as the initial problem transformation into the TFF0 format.

**Key components of iProver include:**

- Preprocessing including \[6\]: predicate elimination \[9\], splitting, semantic filtering, subtyping and definition elimination.
- Inst-Gen calculus \[5,11\]: model guided incremental instantiation with unification, dismatching constraints and global subsumption \[10,11\].
- Resolution with dynamic literal selection, ordering restrictions and simplifications forward/backward: subsumption, global subsumption and subsumption resolution.
- Superposition with simplifications \[2\]: forward/backward: demodulation, light normalisation, subsumption, global subsumption and subsumption resolution. A range of indexes are implemented for inference rules and simplifications including perfect and non-perfect discrimination trees, feature vector indexes. Simplifications and indexes are partitioned within iProver’s given clause algorithm into active, passive and immediate sets.
- AC reasoning: AC joinability and AC normalisation \[3\].
- All calculi above operate on a combination of priority queues for selecting next formulas for inferences. These priorities can be specified in the command line as a lexicographic combination of different formula characteristics.

\[1\] iProver is available at: [http://www.cs.man.ac.uk/~korovink/iprover](http://www.cs.man.ac.uk/~korovink/iprover)
Abstraction-refinement loop interleaving under and over approximations of first-order formulas \cite{15} on top of the calculi above.

Finite model finding for first-order logic via translation into the EPR fragment \cite{1} and non-cyclic sorts \cite{12}.

Proof and model reconstruction, which is non-trivial due to simplifications involving the global search state such as global subsumption \cite{13}.

Heuristics selection and scheduling which specify over 100 prover parameters governing simplifications and interleaving of the calculi. The heuristics are learnt using machine learning HOS-ML framework which is based on Bayesian hyper parameter optimisation and dynamic clustering \cite{7,8}. The heuristic learning process is syntax independent and we reused heuristics trained on TPTP problems (rather than SMT) which should be unbiased towards SMT problems. A parallel 4-core schedule was precomputed from the learnt heuristics using constraint solving, maximising the schedule coverage of solved training problems for the given time limit \cite{7}.

References


