THE RISCTP SOFTWARE

Some Recent Developments...

Wolfgang Schreiner
Research Institute for Symbolic Computation (RISC)
Johannes Kepler University Linz, Austria

JOHANNES KEPLER UNIVERSITY LINZ

The RISCTP Theorem Proving Interface

An abstraction layer for equipping RISCAL with theoring proving capabilities.

- RISCTP: an intermediate language for stating proof problems.
  - Lower level of abstraction than RISCAL, higher level than SMT-LIB.
  - Like SMT-LIB: typed variant of first-order logic, algebraic data types, functional arrays with extensionality, integer arithmetic.
  - Beyond SMT-LIB: overloading, subtypes, tuple types, choose expressions.
  - Implementation by a Java library and as a standalone program.
- Solution of proof problems possible in various ways.
  - Currently: translation to SMT-LIB using full quantification and SMT-LIB logics `ArraysEx` and `Ints`; connection to SMT solvers/provers cvc5, Vampire, Z3.
  - Ongoing: internal provers (with interfaces to SMT solvers) based on resolution (V. Langenreither) and model elimination (W. Schreiner), respectively.

`https://www.risc.jku.at/research/formal/software/RISCTP`

The RISCTP Language

```
// problem file "arrays.txt"
const N:Nat; axiom posN ⇔ N > 0;
type Index = Nat with value < N;
type Value; type Elem = Tuple[Int,Value]; type Array = Map[Index,Elem];
fun key(e:Elem):Int = e.1;
pred sorted(a:Array,from:Index,to:Index) ⇔
  ∀i:Index,j:Index. from ≤ i ∧ i < j ∧ j ≤ to ⇒ key(a[i]) ≤ key(a[j]);
theorem T ⇔
  ∀a:Array,from:Index,to:Index,x:Int.
    from ≤ to ∧ sorted(a,from,to) ⇒
    // let i = (from+to)/2 in
    let i = choose i:Index with from ≤ i ∧ i ≤ to in
    key(a[i]) < x ⇒ ¬∃j:Index. from ≤ j ∧ j < i ∧ key(a[j]) = x;
```

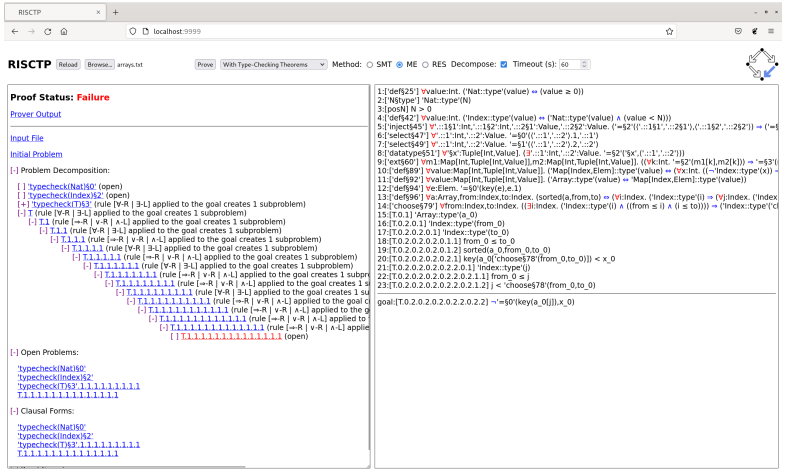
Translation to the SMT-LIB language for SMT solving and to classical first-order logic for theorem proving.

Processing Steps

1. Parse.
2. Type-check.
3. Remove subtypes.
4. Resolve overloading.
5. Remove choose expressions.
6. (FOL) Replace constants denoting variables by actual variables.
7. (FOL) Replace datatype declarations and match expressions.
8. (FOL) Replace let expressions.
9. (FOL) Replace function definitions by axioms.
10. (FOL) Separate terms from formulas.
11. Determine theorems and prune problem accordingly.
12. (SMT) Translation to SMT-LIB.
13. (FOL) Decompose problem into subproblems by sequent calculus.
14. (FOL) Transform problem into clausal form.

Starting point of first-order proofs.

RISCTP GUI



A web-based frontend to monitor prover and inspect proofs.