DÉduction CERTifiée – ANR DEFI (2009 - 2011)

CEA LIST, Nancy, Orsay, Rennes, Sophia-Antipolis, Systerel

Affordable highest Evaluation Assurance Levels (EAL7)
$\Rightarrow$ Satisfiability Modulo Theory (SMT) provers

1) Design new, efficient cooperating decision procedures.
VeriT, Alt-Ergo SMT provers

2) Design a standard output interface (certificates, proof objects) From untrusted computations to trusted result

3) Integrate 1-2 with skeptical proof assistant, Proof Carrying Code, Rodin tool for B, CEA's Frama-C for C

Baseline: Smaller Trusted Computing Base, Better automation

Organisation du projet

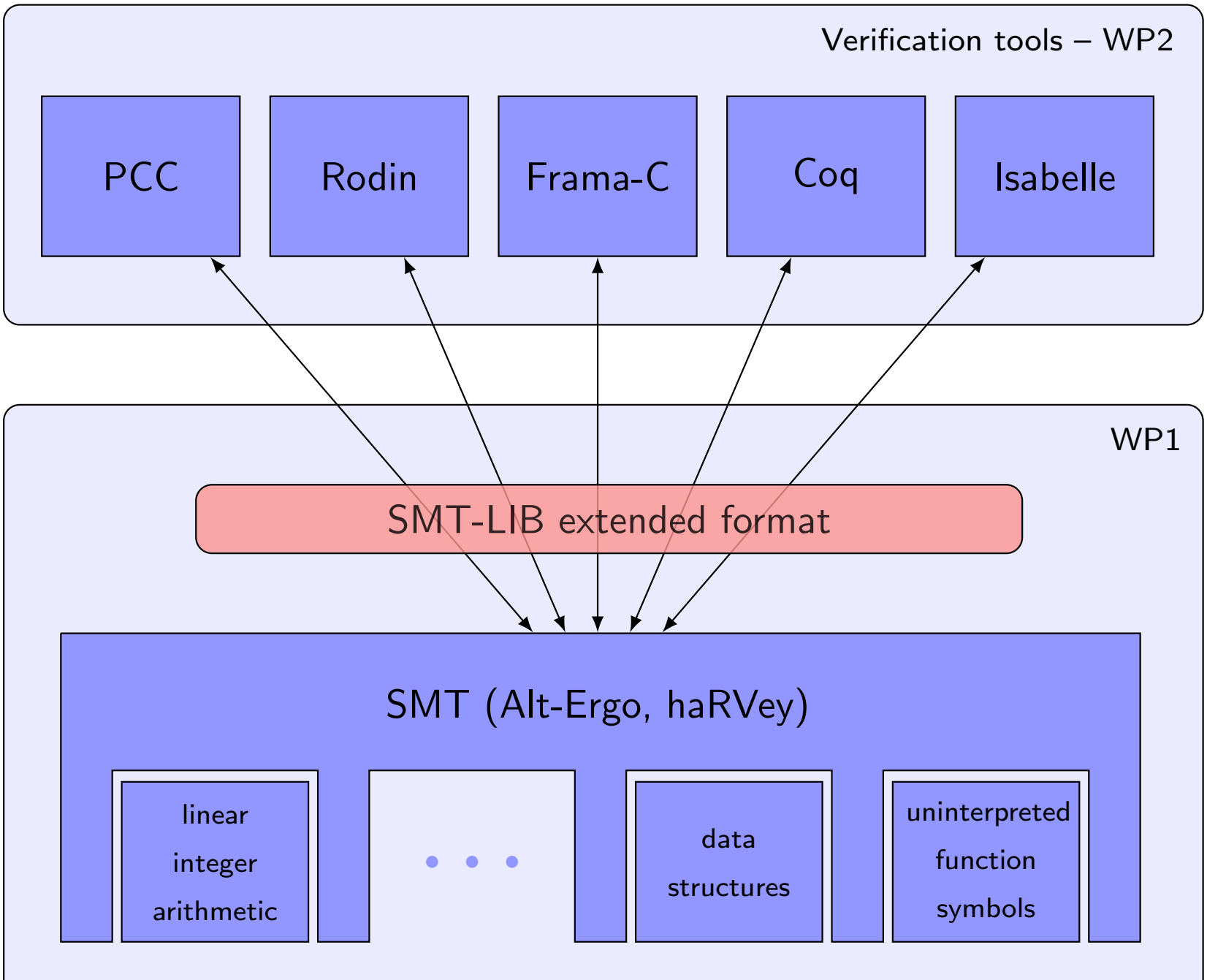
WP1 : Decision procedures and their combination

- Task 1 : Requirement analysis
- Task 2 : Expressiveness (Arithmetics + new combinations)
- Task 3 : Efficiency
- Task 4 : Proof witnesses for decision procedures

WP2 : Integrating decision procedures into verification tools

- Task 5 : Proof assistants (Coq, Isabelle)
- Task 6 : PCC
- Task 7 : Rodin
- Task 8 : Frama-C

Vue d'ensemble



T1 : Requirement analysis (T0 $\longrightarrow$ T0+6)

Repository of problems to be tackled by the project

- Expressiveness
- Efficiency (expected time / memory)
- Proof witnesses (quantitative / qualitative)

$\Rightarrow$ your inputs are most welcome!

T2 : Expressiveness

Arithmetique (linéaire, non-linéaire, modulo)
Simplex, Gröbner bases, Positivstellenstaz, etc

Combinations (beyond Shostak and Nelson-Oppen)
structures with resource functions, Bernays-Schönfinkel scheme

T3 : Efficiency

Collaborative decision procedures for arithmetic
difference, linear, non-linear

Strategies for efficient deduction
choice of axioms, instantiations

T4: Proof witnesses for decision procedures

Efficient decision procedures are highly optimised (C,C++)

How can we use them safely in other verification tools ?
$\Rightarrow$ generation of verifiable proof witnesses

A proof format for SMTs – propose and experiment

- deduction tree
- execution trace
- *ad'hoc* certificate (notably for arithmetic)

Accomodate different levels of abstraction

- verbose $\Rightarrow$ easily checkable but proof too big?
- ...
- terse $\Rightarrow$ by omega

T5 : Integration into proof-assistants

Different proof-assistants, different approaches

Coq: proof witnesses are (part of) the proof-term
proof by reflection

Isabelle: proof witnesses can be consumed on the fly
proof by tactics

T6-7-8 : PCC - Rodin - PCC

Conclusion

Coq Next Gen is a theorem prover ?

SMT are about SAT + uninterpreted function symbols How

combination schemes fit with Coq constructive logic
SAT $\neq$ Prop

How Coq congruence fits into the picture

Micromega : how (not?) to bind with Coq

Reflexive verifier for arithmetic proofs

- Complete for non-linear arithmetics over $\mathbb{R}$
  Positivstellensatz
- Complete for linear arithmetics over $\mathbb{Z}$
  Farkas Lemma + cutting planes (+ enumeration)
- Works for any ordered ring (Evgeny Makarov)
  $(\mathbb{Z}, \mathbb{Z})$ lia, $(\mathbb{Q}, \mathbb{Q})$ psatz Q, $(\mathbb{R}, \{0, 1\})$ psatz R
- handles propositional logic (naive CNF)

Syntaxification

1. Ltac is too slow for big goals;
2. For outsiders, writing Caml code is hard – pollute the code base
3. Corner cases (typing, modulo conversion)
   `forall A: Set, A -> (A -> x >= 1) -> x >= 0`
4. `external` is probably the way forward
5. Proof caches can be handy – in the absence of the prover

How generic is the tactic ?

I have a new ordered ring, can I get a tactic ?

Coq checker is generic

CamI implementation is open source ☹
The parser needs to be modified...

Future work

Port the tactic for *cousin* datatypes : `nat` , `positive` , etc `positive` first

Enrich the set of operators (division (euclidian) , max , etc)

Use a fast CNF and smaller certificates