Paral-ITP: Pervasive Parallelism in Interactive Theorem Provers

Coordinator: Burkhart Wolff, LRI, Université Paris-Sud

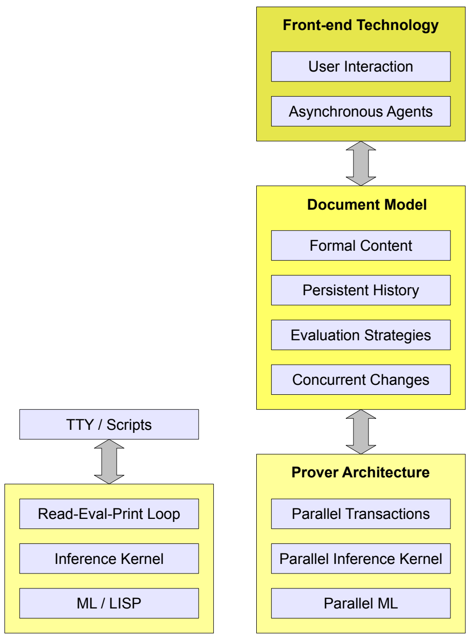
Background

Interactive theorem proving is a technology of fundamental importance for mathematics and computer-science. It is based on expressive logical foundations and implemented in a highly trustable way. Applications include huge mathematical proofs and semi-automated verifications of complex software systems. Interactive development of larger and larger proofs increases the demand for computing power, which means explicit parallelism on current multicore hardware.

The architecture of contemporary interactive provers such as Coq, Isabelle or the HOL family goes back to the influential LCF system (from 1979), which has pioneered key principles like correctness by construction for primitive inferences and definitions, free programmability in userspace via ML, and toplevel command interaction. Both Coq and Isabelle have elaborated the prover architecture over the years, driven by the demands of sophisticated proof procedures, derived specification principles, large theory libraries etc. Despite this success, the operational model of interactive proof checking is largely limited by sequential ML evaluation and the sequential read-eval-print loop, as inherited from LCF.

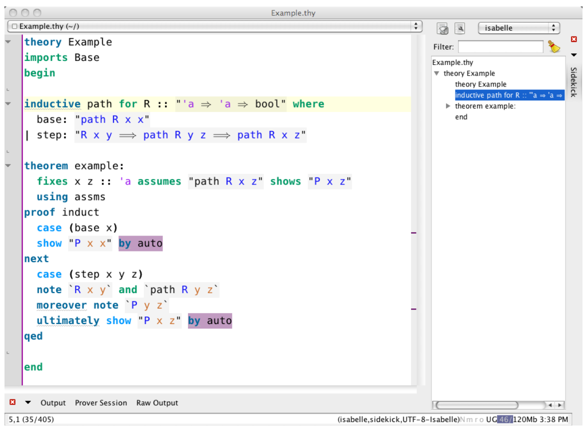
The Project Aims

The project intends to overcome the sequential model both for Coq and Isabelle, to make the resources of multi-core hardware available for even larger proof developments. Beyond traditional processing of proof scripts as sequence of proof commands, and batch-loading of theory modules, there is a large space of possibilities and challenges for pervasive parallelism. Updating the traditional LCF architecture affects many layers of each prover system:



Parallelization of the different layers is required on the level of the execution environments (SML, OCaml), which must include some form of multi-threading or multi-processing supported by multi-core architectures. This carries over to the **inference kernel**, which must be extended by means to decompose proof checking tasks into independent parts and to check them concurrently. The tactic code of **prover components** such as proof procedures or derived specification packages must be parallelized, and structuring mechanisms in the proof command language must support parallel checking.

Our approach is **document centric**: the user edits a document containing text, code, definitions, and proofs to be checked incrementally. This means that checking is to be split into parallel subtasks reporting their results asynchronously; the **document model** and its protocols must support this. Finally, a system front-end must be provided that makes all these features **accessible to the user**. Instead of a conventional proof-editor, the project engages to provide an Prover-IDE following the paradigm of "continuous build – continuous check" usable both for novices and experts.



```
theory Example
imports Base
begin

inductive path for R :: "'a ⇒ 'a ⇒ bool" where
  base: "path R x x"
| step: "R x y ⟹ path R y z ⟹ path R x z"

theorem example:
  fixes x z :: 'a assumes "path R x z" shows "P x z"
  using assms
proof induct
  case (base x)
  show "P x x" by auto
next
  case (step x y z)
  note `R x y` and `path R y z`
  moreover note `P y z`
  ultimately show "P x z" by auto
qed

end
```

Some of these aspects need to be addressed for Coq and Isabelle in slightly different ways, to accommodate different approaches in either system tradition.

These substantial extensions of the operational aspects of interactive theorem proving shall retain the trustability of LCF-style proving at the very core. The latter has to be shown with a collection of formal proofs concerning chosen aspects of the prover architecture.

Results so far: The Isabelle Version 2012 from Tue Sep 04 00:16:03 changeset `61e222517d06` supports all forms of parallelism described here. The interface is the jEdit client of the PIDE framework.

The Partners

The **LRI ForTesSE** team (UPSud), incl. members from the **Cedric** team (CNAM), the **INRIA Pi.r2** team (PPS, UParis-Diderot), incl. members from the **INRIA Gallium** team, and the **INRIA Marelle-TypiCal** team (LIX, Ecole Polytechnique)

The project involves three sites and is labellized by System@tic.