Abstract. Interactive theorem proving is tackling ever larger formalization and verification projects, and there is a critical need for theory engineering techniques to support these efforts. One such technique is cross-prover package management, which has the potential to simplify the development of logical theories and effectively share theories between different theorem prover implementations. The OpenTheory project has developed standards for packaging theories of the higher order logic implemented by the HOL family of theorem provers. What is currently missing is a standard theory library that can serve as a published contract of interoperability and contain proofs of basic properties that would otherwise appear in many theory packages. The core contribution of this paper is the presentation of a standard theory library for higher order logic represented as an OpenTheory package. We identify the core theory set of the HOL family of theorem provers, and describe the process of instrumenting the HOL Light theorem prover to extract a standardized version of its core theory development. We profile the axioms and theorems of our standard theory library and investigate the performance cost of separating the standard theory library into coherent hierarchical theory packages.

1 Introduction

Interactive theorem proving has grown from toy examples to major formalization and verification projects in mathematics and computer science. Recent examples include: the 20 man-year verification of the seL4 operating system kernel [24]; the CompCert project, which verified an optimizing compiler from a large subset of C to PowerPC assembly code [25]; and the Flyspeck project, which aims to mechanize a proof of the Kepler sphere-packing conjecture [14].

Just as the term software engineering was coined in 1968 [26] to give a name to techniques for developing increasingly large programs, there is now a need for theory engineering techniques to develop increasingly large proofs ("proving in the large"). One software engineering technique that can be applied to proof development is effective package management. Modern operating systems [8] and programming languages [6] bundle software into packages that carry their dependencies, supporting easy distribution and automatic checking at installation time to ensure that the system can properly support the package. The goal of
the OpenTheory project is to transfer the benefits of package management to aid the development of logical theories.

The initial case study of the OpenTheory project is to develop the infrastructure necessary to port theories between three related interactive theorem provers: HOL Light [15], HOL4 [28] and ProofPower [23]. These three theorem provers implement the same higher order logic, namely Church’s simple theory of types extended with Hindley-Milner style type variables [10]. They also have a similar design of an interactive interface where the user invokes proof tools to prove subgoals, built on top of a small logical kernel that enforces soundness. The logical kernel design is inherited from Milner’s pioneering work on the LCF theorem prover [11], which Gordon reused to implement higher order logic in the HOL theorem prover [12], and from which the three chosen theorem provers are all descended [13].

Even though HOL Light, HOL4 and ProofPower implement the same logic using the same conceptual design, they each contain significant theory formalizations that are not accessible to each other. For example, HOL Light has a formalization of complex analysis [16], HOL4 has a formalization of probability theory [18], and ProofPower has a formalization of the Z specification language [2]. The reason that these useful theories are not available in all of the theorem provers is that it requires significant human effort to port a theory to a new environment, due to differences in the native theories and proof tools.

To overcome the differences between the name and behavior of proof tools between the theorem provers, the OpenTheory project has developed a standard article file format for serializing proofs of higher order logic [21]. Proofs are reduced to a standard set of primitive inferences that are precisely specified and can be simulated by any theorem prover in the HOL family. This bypasses the differences in the proof tools, at the cost of archiving proofs in a format that is hard to modify.

Once the differences between the proof tools have been removed as an obstacle, the challenge that remains is to reconcile the differences between the native theories available in each theorem prover. To illustrate the need for this, suppose we desire to port the theory of complex numbers from HOL Light to HOL4. One way to do this is to export every theory that the HOL Light complex numbers depend on as proof articles, and then import these into HOL4. However, now we have two copies of the theory of real numbers inside HOL4: the original real number theory of HOL4 and the real number theory imported from HOL Light that the complex numbers depend on. Because of this, we cannot easily combine the new theory of complex numbers with other HOL4 theories that depend on the original real number theory, such as the theory of probability.

To avoid this duplicate theory problem, when we speak of porting theories between theorem provers we usually have in mind the following procedure:

---

1 The OpenTheory project homepage is [http://gilith.com/research/opentheory](http://gilith.com/research/opentheory)

2 The author has first-hand experience of this: his introduction to theorem proving was porting a theory of real numbers from HOL Light to HOL4.