Automated Theorem Proving in Higher-Order Logics

Alexander Steen, University of Luxembourg

Berlin, September 2021

Automated theorem proving (ATP) denotes the automation of deduction procedures that, given a set of assumptions (the axioms or premises) and a conjecture as input, try to decide whether the input conjecture is a logical consequence of the assumptions. A formula is said to be a logical consequence if it is not possible that all assumptions are validated while the conjecture is not. In the context of ATP systems this reasoning process is conducted fully automatically, i.e. no further user-interaction is necessary. In a most simplistic setting, the output of an ATP system is a yes-or-no answer, depending on whether the conjecture was proven or refuted, respectively. Contemporary ATP systems additionally output a so-called proof object that certifies the afore stated answer. There exist many different ATP systems today; one of the main differences between these systems being what formalism (that is logical language) they support. In my work, I am particularly interested in theoretical foundations and practical aspects of ATP systems for classical higher-order logic (HOL).

Higher-order logics are expressive logical formalisms that allow for quantification over predicate and function variables, i.e. quantification is not restricted to individuals as in first-order logic. The most common formu- lations used today in the context of computer-assisted reasoning are based on a simply typed λ-calculus as proposed by Church in the 1940s (more formally, Extensional Type Theory by Hekin). HOL provides λ-notation as an elegant and useful means to denote unnamed functions, predicates and sets (by their characteristic functions), and comes with built-in principles of Boolean and functional extensionality as well as type-restricted comprehension. HOL reasoning systems have successfully been employed in various applications, such as software and hardware verification, mathematical reasoning and automation of non-classical logics,

One long-term goal of Leo-III is to provide means for reasoning within (and about) non-classical logics including, e.g., free logic, quantified modal logic and deontic logics. As a first step towards this goal, Leo-III is capable of reasoning within all Kripke-complete normal (first- and higher- order quantified) modal logics as well as dyadic deontic logic without requiring any further external tool.

In this talk, I will present the automated theorem prover Leo-III for classical higher-order logic with Henkin semantics. Leo-III is based on extensional higher-order paramodulation and accepts every common TPTP dialect (FOF, TFF, THF), including their recent extensions to rank-1 polymorphism (TF1, TH1). In addition, the prover natively supports reasoning in almost every normal higher-order modal logic. I will briefly sketch Leo-III's underlying theoretical principles, but focus primarily on its architecture, pragmatics and applications.