Decision Procedure for Equivalence Relations

Internship Supervisors	Pierre Corbineau & Karine Altisen & Lionel Rieg Pierre.Corbineau@univ-grenoble-alpes.fr Karine.Altisen@univ-grenoble-alpes.fr Lionel.Rieg@univ-grenoble-alpes.fr
Research Team	PACSS & Synchrone Teams
Research Lab	Verimag Lab http://www-verimag.imag.fr
Lab Director	Florence Maraninchi Florence.Maraninchi@univ-grenoble-alpes.fr
Research Institution	Université Grenoble Alpes, Grenoble, France

Keywords Logic & Verification, Automated reasoning, Proof assistants

Scientific Context Automated reasoning is a widely-used technique for system and program verification, where problems are translated into logical formulas. These formulas can be checked by automated theorem provers. However, an automated prover may fail to answer a specific problem because of undecidability.

By contrast, interactive proof assistants allow the user to manually guide the proof process. This allows for more expressive proof techniques to be used at the cost of more user time and expertise. In order to reduce the burden on the expert user, it is desirable to provide suitable automation, especially for the seemingly trivial parts of interactive proofs.

The goal of the proposed research is to design a decision procedure for the Coq proof assistant [1] that would extend equality-based reasoning (e.g., congruence-closure) to heterogeneous problems where equalities are expressed using multiple equivalence relations.

Scientific Problem Equality reasoning is a very common paradigm for proofs both in the field of automated theorem proving and when using interactive proof assistants such as Coq [1]. The Coq proof assistant comes with a natural notion of equality which encompasses the notion of *computation* within the language (mostly typed λ-calculus). When reasoning over functions, the Coq equality captures equality of the *programming code* (as a λ-term) of the function rather than *pointwise equality* (for all inputs). However, most properties about functions are *extensional*, *i.e.*, they are in fact properties of the images of the function.

A common approach to circumvent the problem is to work in a *setoid* (that is, a set equipped with an equivalence relation): instead of using Coq equality, one can define an *ad hoc* user-defined equality. Note that even if this relation is an equivalence for base types, this cannot be ensured for function types; we can only obtain partial (i.e., non-reflexive) equivalence relations. The main consequence is that replacement of equivalent objects can only occur in a suitable context, under adequate *relation morphisms*.

In practice, user-defined as well as Coq equalities are very often involved in the same proofs and reasoning. The Coq proof assistant is equiped with procedures for replacing equivalent objects under morphisms and with an efficient decision procedure for quantifier-free reasoning on Coq equality [2, 3]. However, this mechanism lacks a procedure which could combine Coq and user-defined equalities and provide a generic equality reasoning procedure.

One key ingredient that should be studied is the possibility of matching terms up to equivalence, in order to process consequences of general hypotheses.

Proposed Work We propose during the internship to tackle this problem by:

- Formally defining the problem on a small generic language (independant from Coq).
- Designing a decision procedure for object replacement in case of user-defined as well as Coq equalities.
- Studying the corresponding matching problem (up to term equivalence).

- Implementing and experimenting with a matching algorithm prototype.
- Depending on the student's interests, integrating the algorithm as a Coq tactic.

Required Skills The main goal of the internship is to study automated reasoning rather than Coq itself, so no prior knowledge of Coq is required. We expect the candidate to be familiar with basic first-order logic. Typed $\lambda-$calculus is a plus. Development skills in Objective Caml are mandatory.

Working Context The internship is part of ANR project ESTATE[1].

The subject can be extended into a PhD thesis.

Bibliography

[1] The Coq Development Team. The Coq Proof Assistant Reference Manual. `http://coq.inria.fr`.

[2] P. Corbineau. Deciding equality in the constructor theory. In *TYPES 2006 : Types for Proofs and Programs*, volume 4502 of *Lecture Notes in Computer Science*, pages 78–92. Springer-Verlag, 2007.

[3] Downey, Peter J. and Sethi, Ravi and Tarjan, Robert Endre. Variations on the Common Subexpression Problem. In *Journal of the ACM*, volume 27 of issue 4, October 1980.

[1] https://wp-systeme.lip6.fr/estate/