Applications of Craig Interpolation to Model Checking

Kenneth McMillan

Cadence Berkeley Labs

A Craig interpolant [1] for a mutually inconsistent pair of formulas (A,B) is a formula that is (1) implied by A, (2) inconsistent with B, and (3) expressed over the common variables of A and B. It is known that a Craig interpolant can be efficiently derived from a refutation of $A \wedge B$, for certain theories and proof systems. For example, interpolants can be derived from resolution proofs in propositional logic, and for "cutting planes" proofs for systems of linear inequalities over the reals [5, 3]. These methods have been extended to the theory of linear inequalities with uninterpreted function symbols [4].

The derivation of interpolants from proofs has a number of applications in model checking. For example, interpolation can be used to construct an inductive invariant for a transition system that is strong enough to prove a given property. In effect, we can use interpolation to construct an abstract "image operator" that can be iterated to a fixed point to obtain an invariant. This invariant contains only information actually deduced by a prover in refuting counterexamples to the property of a fixed number of steps. Thus, in a certain sense, we abstract the invariant relative to a given property. This avoids the complexity of computing the strongest inductive invariant (i.e., the reachable states) as is typically done in model checking.

This approach gives us a complete procedure for model checking temporal properties of finite-state systems that allows us to exploit recent advances in SAT solvers for the proof generation phase. Experimentally, this method is found to be quite robust for verifying properties of industrial hardware designs, relative to other model checking approaches.

The same approach can be applied to infinite-state systems, such as programs and parameterized protocols, although there is no completeness guarantee in this case. Alternatively, interpolants derived from proofs can be used to infer predicates that are useful for predicate abstraction [6]. This approach has been used in a software model checking to verify properties of C programs with in excess of 100K lines of code [2].

References

1. W. Craig. Linear reasoning: A new form of the Herbrand-Gentzen theorem. *J. Symbolic Logic*, 22(3):250–268, 1957.

2. T. A. Henzinger, R. Jhala, Rupak Majumdar, and K. L. McMillan. Abstractions from proofs. In *ACM Symp. on Principles of Prog. Lang. (POPL 2004),* 2004. to appear.
3. J. Krajíček. Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic. *J. Symbolic Logic,* 62(2):457–486, June 1997.
4. K. L. McMillan. An interpolating theorem prover. In *TACAS 2004,* pages 16–30, 2004.
5. P. Pudlák. Lower bounds for resolution and cutting plane proofs and monotone computations. *J. Symbolic Logic,* 62(2):981–998, June 1997.
6. Hassen Saïdi and Susanne Graf. Construction of abstract state graphs with PVS. In Orna Grumberg, editor, *Computer-Aided Verification, CAV '97,* volume 1254, pages 72–83, Haifa, Israel, 1997. Springer-Verlag.