Program Synthesis and Transformation Techniques for Simpuation, Optimization and Constraint Satisfaction

Thomas Ellman
Department of Computer Science
Rutgers University
ellman@cs.rutgers.edu

Deductive Synthesis of Numerical Simulation Programs from Networks of Algebraic and Ordinary Differential Equations

Scientists and engineers face recurring problems of constructing, testing and modifying numerical simulation programs. The process of coding and revising such simulators is extremely time-consuming, because they are almost always written in conventional programming languages. Scientists and engineers can therefore benefit from software that facilitates construction of programs for simulating physical systems. Our research adapts the methodology of deductive program synthesis to the problem of constructing numerical simulation codes. We have focused on simulators that can be represented as second order functional programs composed of numerical integration and root extraction routines. We have developed a system that uses first order Horn logic to synthesize numerical simulators built from these components. Our approach is based on two ideas: First, we axiomatize only the relationship between integration and differentiation. We neither attempt nor require a complete axiomatization of mathematical analysis. Second, our system uses a representation in which functions are reified as objects. Function objects are encoded as lambda expressions. Our knowledge base includes an axiomatization of term equality in the lambda calculus. It also includes axioms defining the semantics of numerical integration and root extraction routines. We use depth bounded SLD resolution to construct proofs and synthesize programs. Our system has successfully constructed numerical simulators for computational design of jet engine nozzles and sailing yachts, among others. Our results demonstrate that deductive synthesis techniques can be used to construct numerical simulation programs for realistic applications (Ellman and Murata 1998).

A Transformation System for Interactive Reformulation of Design Optimization Strategies

Automatic design optimization is highly sensitive to problem formulation. The choice of objective function, constraints and design parameters can dramatically impact the computational cost of optimization and the quality of the resulting design. The best formulation varies from one application to another. A design engineer will usually not know the best formulation in advance. In order to address this problem, we have developed a system that supports interactive formulation, testing and reformulation of design optimization strategies. Our system includes an executable, data-flow language for representing optimization strategies. The language allows an engineer to define multiple stages of optimization, each using different approximations of the objective and constraints or different abstractions of the design space. We have also developed a set of transformations that reformulate strategies represented in our language. The transformations can approximate objective and constraint functions, abstract or reparameterize search spaces, or divide an optimization process into multiple stages. The system is applicable in principle to any design problem that can be expressed in terms of constrained optimization; however, we expect the system to be most useful when the design artifact is governed by algebraic and ordinary differential equations. We have tested the system on problems of racing yacht design and jet engine nozzle design. We report experimental results demonstrating that our reformulation techniques can significantly improve the performance of automatic design optimization. Our research demonstrates the viability of a reformulation methodology that combines symbolic program transformation with numerical experimentation. It is an important first step in a research program aimed at automating the entire strategy formulation process (Ellman *et al.* 1998).

Multi-Level Modeling for Engineering Design Optimization

Physical systems can be modeled at many levels of approximation. The right model depends on the problem to be solved. In many cases, a combination of models will be more effective than a single model alone. Our research investigates this idea in the context of engineering design optimization. We present a family of strategies that use multiple models for unconstrained optimization of engineering designs. The strategies are useful when multiple approximations of an objective function can be implemented by compositional modeling techniques. We show how a compositional modeling library can be used to construct a variety of locally calibratable approximation schemes that can be incorporated into the optimization strategies. We analyze the optimization strategies and approximation schemes to formulate and prove sufficient conditions for correctness and convergence. We also report experimental tests of our methods in the domain of sailing yacht design. Our results demonstrate dramatic reductions in the CPU time required for optimization, on the problems we tested, with no significant loss in design quality (Ellman *et al.* 1997).

Abstraction of Constraint Satisfaction Problems via Approximate Symmetry

Abstraction techniques are important for solving constraint satisfaction problems with global constraints and low solution density. In the presence of global constraints, backtracking search is unable to prune partial solutions. It therefore operates like pure generate-and-test. Abstraction improves on generate-and-test by enabling entire subsets of the solution space to be pruned early in a backtracking search process. These papers describe how abstraction spaces can be characterized in terms of approximate symmetries of the original, concrete search space. They define two special types of approximate symmetry, called "range symmetry" and "domain symmetry". They also present algorithms for automatically synthesizing hierarchic problem solvers based on range or domain symmetry. The algorithms operate by analyzing declarative descriptions of classes of constraint satisfaction problems. Both algorithms have been fully implemented. These papers also present data from experiments testing the two synthesis algorithms and the resulting problem solvers on several NP-hard constraint satisfaction problems (Ellman 1993a), (Ellman 1993b).

References

T. Ellman and T. Murata. Deductive synthesis of numerical simulation programs from networks of algebraic and ordinary differential equations. *Automated Software Engineering*, 5(3), 1998.

T. Ellman, J. Keane, M. Schwabacher, and K. Yao. Multi-level modeling for engineering design optimization. *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing*, 11(5), 1997.

T. Ellman, J. Keane, A. Banerjee, and G. Armhold. A transformation system for interactive reformulation of design optimization strategies. Fully accepted to *Research in Engineering Design*, 1998.

T. Ellman. Abstraction via approximate symmetry. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, Chambery, France, August 1993.

T. Ellman. Synthesis of abstraction hierarchies for constraint satisfaction by clustering approximately equivalent objects. In *Proceedings of the Tenth International Conference on Machine Learning*, Amherst, MA, 1993.