A Survey of Reactive Synthesis

Nikolaj S. Bjørner

April 10, 1996

The different synthesis problems summarized in Figure 1 were surveyed in the talk. Functional synthesis

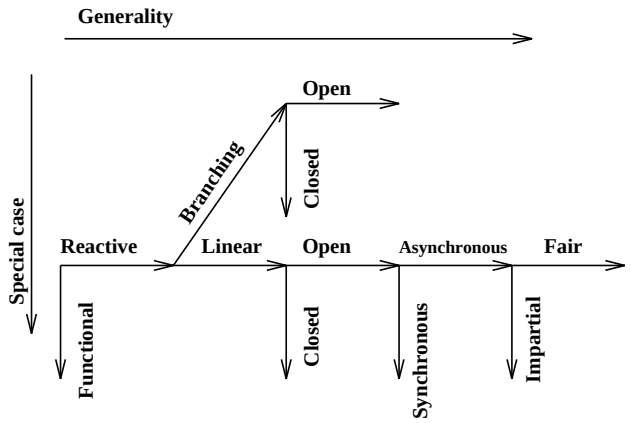

Figure 1: Road map of the reactive synthesis talk

addresses the problem of synthesizing a functional program from a first or higher-order logical specification of the form $\forall x \exists y \cdot \varphi(x, y)$. A functional or sequential program f is extracted from a proof whose inference rules correspond to programming constructs. The program f satisfies $\forall x \cdot \varphi(x, f(x))$.

The challenge in reactive synthesis is to extract programs realizing specifications in temporal logics. We focus on temporal logics that can express progress properties.

Derivation of synchronization skeletons for closed systems is directly related to the satisfiability problem for temporal logics. Thus, the synchronization skeletons correspond to constructive elimination of the existential quantifiers in the specification $\exists y \cdot \varphi(y)$.

Open synchronous reactive synthesis is the temporal counterpart of functional synthesis. A temporal specification of the form $\forall x \exists y \cdot \varphi(x, y)$ is realized by a module f satisfying $\forall x \cdot \varphi(x, f(x))$. Requirements like: *$f$ may not depend on the future* and *f may not depend on the far past* are imposed to ensure the extracted module can be implemented. We summarize three approaches to this problem: *(i)* the tree-automata, *(ii)* the game automata, and *(iii)* the control theoretic based approach. We demonstrate how the approaches are easily inter-reducible. The main difficulty encountered in the realizability problem is rather the computational complexity of extracting f as well as the resulting size of f. In the branching time spectrum we consider the control theoretic approach for branching time specifications.

In asynchronous synthesis one allows the environment and synthesized module to interact by interleaving. A third player, the scheduler, takes the role of controlling the synthesized module's behavior. By imposing fairness constraints on the scheduler one faces the *fair* synthesis problem.

The talk concludes with a summary of open theoretical and practical problems. Though the most impressive work has been carried out in the theoretical domain the field seems to need more ambitious examples that will demonstrate how reactive synthesis can be applied in praxis.

A copy of the slides including a list of continuously updated references can be obtained via the web address below.

Address
Gates Building room 460
Computer Science Department
Stanford University
Stanford CA 94305-9045
e-mail: nikolaj@cs.stanford.edu
www: http://theory.stanford.edu/people/nikolaj/index.html