Counter-Strategy Guided Refinement of GR(1) Temporal Logic Specifications

Rajeev Alur, Salar Moarref, and Ufuk Topcu. University of Pennsylvania

Reactive Synthesis Problem

Find a finite state controller that satisfies a given temporal logic specification regardless of how its environment behaves.
What if the specification is unrealizable?

Motivation

• Developing a correct and complete formal specification is challenging and tedious
• Initial specifications are often unrealizable
• Unrealizability of specification is often due to inadequate environment assumptions
• Unrealizable specification cannot be executed or simulated
• Debugging an unrealizable specification is hard

Goal

Automatically refining the constraints over the environment by adding assumptions in order to achieve realizability

Applications

• Giving the user an insight into the specification
• Correcting the specification
• Constructing interface rules between the components in the context of compositional synthesis

Patterns

• Patterns are LTL formulas of special form which hold over all runs of the abstraction of the counter-strategy
• Candidate assumptions are generated based on patterns

Results

For a given counter-strategy, synthesize assumptions of the form \( \square \phi \) such that adding any of them:
• Rules out the counter-strategy
• Constrains the environment as weakly as possible in the specified structure

Counter-Strategies

How the environment can behave in order to enforce the system to violate the specification

An unrealizable specification

\( \varphi = \varphi_{env} \implies \varphi_{sys} \)

Environment

Plant

Controller

Main Flow of the Method

Specification

Realizable

Choose & Add

Generating Candidates

Yes

Done

Counter-Strategy

Patterns Synthesis

Subset of Variables

Patterns

• State predicate: Each state of the counter-strategy outputs a valuation over environment variables

\[ q_2 \sim (c \land r) \]

• Candidate assumptions: Replace the states with corresponding state predicates and complement the formulas:

\[ q_2 \sim (c \land r) \]

• User input: a subset of variables which

• may contribute to unrealizability

• are underspecified

• Smaller subset of variables leads to simpler formulas

GR(1) Specifications

Realizability and synthesis problems can be solved in polynomial time

\[ \varphi_{init} \land \bigwedge_{k \in K_s} \varphi^*_{k} \land \bigwedge_{k \in K_s} \varphi^*_{k} \rightarrow \varphi_{init} \land \bigwedge_{k \in K_s} \varphi^*_{k} \land \bigwedge_{k \in K_s} \varphi^*_{k} \]

Case Study

Initial Specification

• The requirements:

- \( \Box (\text{pick up}): \text{infinitely often visit "pick up" cell} \)
- \( \Box (\text{drop off}): \text{infinitely often visit "drop off" cell} \)
- \( \Box(\text{not colliding with the object}) \)
- Unrealizable

Running the refining method

• Refinement:

- \( \Box(\text{Obstacle\ is\ Pick\ up}) \)
- \( \Box(\text{Obstacle\ is\ Drop\ off}) \)
- Obstacle should infinitely often move out of "pick up" and "drop"
- Refined specification: realizable

Conclusions

• Refining the unrealizable specification by adding assumptions:

- Simple GR(1) formulas, easy to understand and validate
- As weak as possible in the specified structure

Current and Future Work

• Taking advantage of multiplicity of generated candidates
• Automatically finding good subset of variables to be used in candidates
• Extending the method to more general fragments of LTL