Environment Assumptions for Synthesis

Verification of Reactive Systems

- Reactive sequential system (finite state, signals)
- Formal specification (e.g., LTL formula)

Synthesis of Reactive Systems

- Automatically construct system from specification
- Correct by construction

Problem

The synthesis problem asks to construct a reactive finite-state system from an ω-regular specification. Initial specifications are often unrealizable, which means that there is no system that implements the specification. A common reason for unrealizability is that assumptions on the environment of the system are incomplete.

We study the problem of correcting an unrealizable specification ψ by computing an environment assumption ψ' such that the new specification ψ → ψ' is realizable. Our aim is to construct an assumption ψ that constrains only the environment and is as weak as possible.

Approach

We present a two-step algorithm for computing assumptions. The algorithm operates on the game graph that is used to answer the realizability question. First, we compute a safety assumption that removes a minimal set of environment edges from the graph. Second, we compute a liveness assumption that puts fairness conditions on some of the remaining environment edges.

Safety Assumption

- What is a safety error?
- System cannot win with help of environment

Liveness Assumption

- What is a liveness error?
- Environment can help at arbitrary points in play
- Help is never “too late”
- Idea: put fairness on edges of environment to rule out liveness errors

Strongly-Fair Assumption

- Environment has to take strongly-fair edge infinitely often if it is enabled infinitely often
- Given
  - game G = ((S, S), E, Φ)
  - set of strongly-fair edges E ⊆ E
- System wins a play q₁, q₂, … iff
  - either Φ is satisfied or
  - there is an edge (s, t) ∈ E such that there exists j > 0 such that (q_j = s) and
  - i ≤ 0 for all i > 0: (q_i = s) → (q_i = t)
- Deciding games with strongly-fair assumption
  - probabilistic games (better complexity than Streett games)
  - Reduction to probabilistic games

Computing Environment Assumption

- Construct synthesis game
- Compute safety assumption and adapt game
- Compute strongly-fair assumption on new game
  - Computing minimal strongly-fair assumption is NP-hard
  - Local optimum: start with all edges and remove one by one with reduction to probabilistic games
- Outcome: automaton A representing ψ

Reference

Krishnendu Chatterjee, Thomas Henzinger, Barbara Jobstmann: Environment Assumptions for Synthesis. CONCUR 2008: 147-161