How should agents view their environment in Reactive Synthesis? Trace-view vs Strategy-view

Benjamin Aminof
Giuseppe De Giacomo
Sasha Rubin
Moshe Y. Vardi

VardiFest 2022

THE UNIVERSITY OF SYDNEY

Reactive Systems

In Reactive Systems, the agent and its environment interact: each takes an action based on the history.

Formally:

- Players use functions from relevant histories to actions.

  $$S : \mathrm{Hist} \to \mathrm{Act}$$

- For agents, called strategies (aka controllers, policies).

  $$\mathbf{S}_{ag} : \mathrm{Hist}_{env} \to \mathrm{Act}_{ag}$$

- For environments, whether or not they are rational, also called strategies (aka choice-functions).

  $$\mathbf{S}_{env} : \mathrm{Hist}_{ag} \to \mathrm{Act}_{env}$$

- Every pair $(\mathbf{S}_{ag}, \mathbf{S}_{env})$ induces a sequence of actions.

  $$\mathrm{play}(\mathbf{S}_{ag}, \mathbf{S}_{env})$$

Environment Traces

- An environment trace is a sequence of environment actions.

  $$\mathbf{O}_{env} : \mathbb{N} \to \mathrm{Act}_{env}$$

- NB. This is just an environment strategy whose actions only depend on the length of the history
- oblivious strategies
  - represent environments that do not respond to the agent (only to time).
  - very restrictive.
  - should be carefully justified if used.

Synthesis under assumptions: two views

- goal Φ_{goal}.
- assumption (aka environment specification) Φ_{asm}.
  e.g., planning domain, fairness

Strategy-view:

Find an agent strategy $\mathbf{S}_{ag}$:
s.t. for every env strategy $\mathbf{S}_{env}$ enforcing Φ_{asm}:
the induced $\mathrm{play}(\mathbf{S}_{ag}, \mathbf{S}_{env})$ satisfies Φ_{goal}.

Trace-view:

Find an agent strategy $\mathbf{S}_{ag}$:
s.t. for every env trace $\mathbf{O}_{env}$:
the induced $\mathrm{play}(\mathbf{S}_{ag}, \mathbf{O}_{env})$ satisfies $\Phi_{asm} \rightarrow \Phi_{goal}$.

- Equirealisable, but the set of agent strategies that solve the problems need not be the same!

Synthesis of best-effort strategies

A shady casino is offering a promotion to place a single color-bet at roulette for free.

- Act_{ag} = bet red, bet black, cheat.
- Act_{env} = land red, land black, throw the agent out.
- Φ_{goal} = (bet red $\rightarrow$ X land red) and (bet black $\rightarrow$ X land black)
  i.e., place a winning bet.
- Φ_{asm} = (cheat $\rightarrow$ X throw the agent out)
  i.e., if the agent cheats then the casino throws it out.

Clearly the agent can't enforce its goal. What should it do?

- Place a (red or black) bet.
- Cheating is guaranteed losing.

Synthesis of best-effort strategies

Best-effort strategy is one that is not (weakly-)dominated.

Strategy-view:

Find an agent strategy $\mathbf{S}_{ag}$:

s.t. for a maximal set of env strategies $\mathbf{S}_{env}$ enforcing Φ_{asm}:

the induced $\mathrm{play}(\mathbf{S}_{ag}, \mathbf{S}_{env})$ satisfies Φ_{goal}.

1. Placing a bet is best-effort in the strategy-view.
2. However, different ways of trying to define a trace-version of best-effort (e.g., using the implication) all have the major problem that cheating is a best-effort strategy.

Thus, the trace-view is not adequate for this complex form of synthesis, and the strategy view should be used instead.

Conventional wisdom

- The trace-view is adequate for linear properties
  - Reactive Synthesis for linear-time objectives (80's)
- and the strategy-view is only needed for branching properties:
  - Module Checking (90's, 00's)
  - Alternating-time logic (90's)
  - Strategy logic (10's, 20's)

Takeaway

- For complex forms of synthesis, unless carefully justified in special cases, the trace view is not adequate also for linear properties.
- Instead, take a strategy view of the environment.