Stackelberg-Pareto Synthesis

Clément Tamines (Université de Mons)

Joint work with

Véronique Bruyère (Université de Mons)

Jean-François Raskin (Université libre de Bruxelles)

August 2021

CONCUR 2021

Outline

1. Reactive Synthesis

2. Stackelberg-Pareto Synthesis

3. Our Results

Outline

Reactive Synthesis

Reactive systems: systems which constantly interact with the environment

Problem of **Reactive Synthesis** (RS)

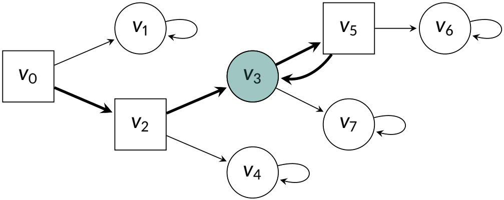
- given a **specification** for the system
- synthesize an adequate **controller** for the system
- enforce the specification **whatever the behavior** of the environment

Classical approach for RS [GTW02]

- interaction is modeled using a two-player game
- Player 0 = system, Player 1 = environment
- specification = objective

Games: Arenas, Plays and Objectives

Game Arena: tuple $G = (V, V_0, V_1, E, v_0)$ with (V, E) a directed graph



Play: infinite path starting with the **initial vertex** v_0, $\rho = v_0 v_2 (v_3 v_5)^\omega$

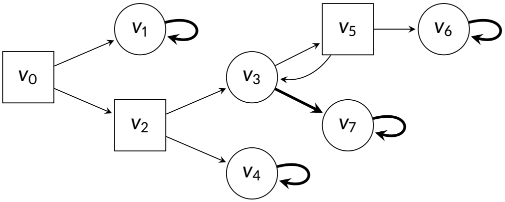
Objective Ω_i for Player $i \in \{0, 1\}$:

- subset of plays, ρ **satisfies** Ω_i if $\rho \in \Omega_i$
- **reachability**: plays which *visit* $\mathbf{T} \subseteq V$

Games: Strategies and Consistency

Strategy $\sigma_i : V^* \times V_i \to V$ **dictates the choices** of Player i

$\to$ given $\underbrace{v_0 v_1 \ldots}_{h} \underbrace{v_k}_{\in V_i}$ yields v_{k+1} from hv_k (memory) or v_k (without)



A play is **consistent** with σ_i if $v_{k+1} = \sigma_i(v_0 \ldots v_k)\ \forall k \in \mathbb{N}, \forall v_k \in V_i$

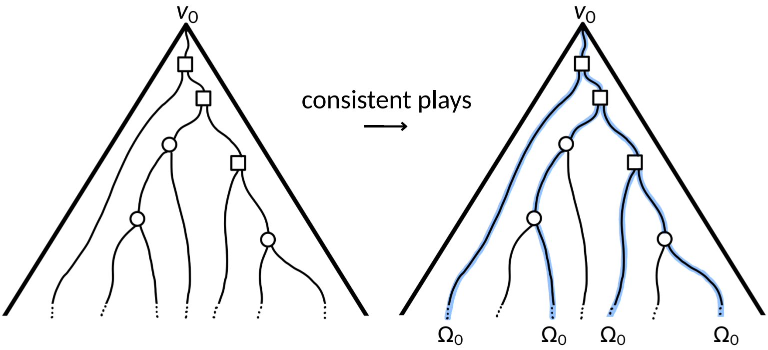
Consider the **set of plays consistent** with a strategy σ_0

$\to \text{Plays}_{\sigma_0} = \{v_0 {v_1}^\omega, v_0 v_2 {v_4}^\omega, v_0 v_2 v_3 {v_7}^\omega\}$

Back to Reactive Synthesis

Classical approach for RS: **zero-sum** games [GTW02]

- objective of environment is **opposite** objective of system: $\Omega_1 = \neg\Omega_0$
- if $\Omega_0 = \text{Reach}(T)$, then $\Omega_1 = \text{Avoid}(T)$
- **adversarial environment**: we want a winning strategy for the system



Setbacks and Alternative

Fully adversarial environment: **bold abstraction of reality**

- assumes the **only goal** of the environment is to **make the system fail**
- environment can be composed of **one or several** components
- each with **own objective**

Alternative: framework of **Stackelberg games** [vS37] (non-zero-sum)

- Player 0 **announces** his strategy σ_0
- Player 1 **rationally answers** with **optimal response** w.r.t. his objective
- goal of Player 0:
  - announce a strategy that **satisfies his objective**
  - **whatever the rational response** of Player 1

Outline

Our New Model

Stackelberg-Pareto game (SP game): $\mathcal{G} = (G, \Omega_0, \Omega_1, \dots, \Omega_t)$

- Player 0 (system): objective Ω_0, announces strategy σ_0
- Player 1 (environment): **several objectives** $\Omega_1, \dots, \Omega_t$ (components)

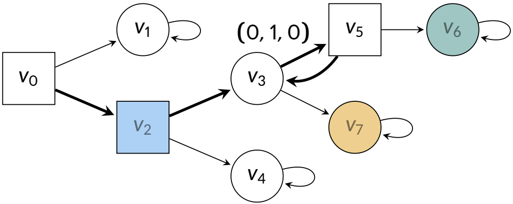
Payoff of ρ for Player 1 is the **vector of Booleans** $\mathrm{pay}(\rho) \in \{0, 1\}^t$

- **order** $\leq$ on payoffs, e.g., $(0, 1, 0) < (0, 1, 1)$

$\Omega_1 = \mathrm{Reach}(\{v_6\})$

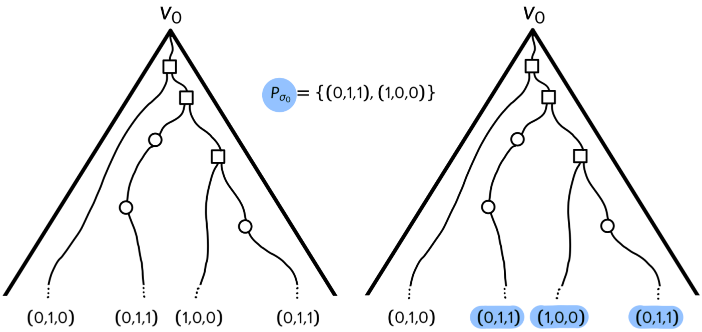
$\Omega_2 = \mathrm{Reach}(\{v_2\})$

$\Omega_3 = \mathrm{Reach}(\{v_7\})$



Pareto-Optimal Payoffs

1. Player 0 announces his strategy σ_0
2. Player 1 considers $\mathsf{Plays}_{\sigma_0}$
   - corresponding **set of payoffs** $\{\mathsf{pay}(\rho) \mid \rho \in \mathsf{Plays}_{\sigma_0}\}$
   - identify **Pareto-optimal** (PO) payoffs (maximal w.r.t. $\leq$) : set P_{σ_0}



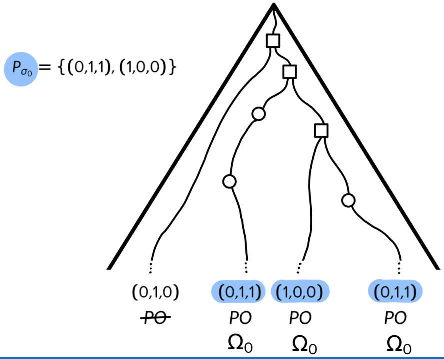
Stackelberg-Pareto Synthesis Problem

Stackelberg-Pareto Synthesis Problem (SPS problem)

The SPS problem is to decide whether there exists a strategy σ_0 for Player 0 such that for every play $\rho \in \mathrm{Plays}_{\sigma_0}$ with $\mathrm{pay}(\rho) \in P_{\sigma_0}$, it holds that $\rho \in \Omega_0$

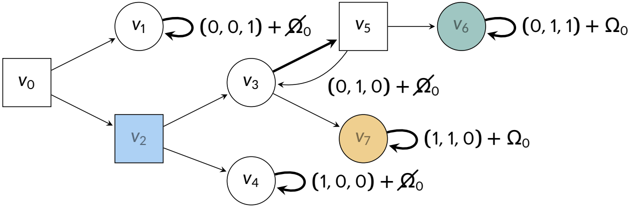
Environment is **rational** and responds to σ_0 **to get a Pareto-optimal payoff**

$\longrightarrow$ Player 0 must **satisfy** Ω_0 in every such **rational response**



SPS Problem Example (1/2)

Consider σ_0 such that $\sigma_0(v_3) = v_5$

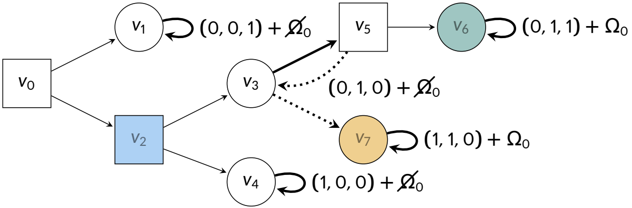


- $\text{Plays}_{\sigma_0} = \{ \; v_0v_1{}^\omega, \; v_0v_2v_4{}^\omega, \; v_0v_2(v_3v_5)^+v_6{}^\omega, \; v_0v_2(v_3v_5)^\omega \; \}$
- $\text{payoffs} = \{ \, (0, 0, 1), \; (1, 0, 0), \; (0, 1, 1), \; (0, 1, 0) \; \}$
- $P_{\sigma_0} = \{ \, (1, 0, 0), \; (0, 1, 1) \, \}$

Strategy σ_0 is **not a solution** to the SPS problem, e.g., $\rho = v_0v_2(v_4)^\omega \notin \Omega_0$
$\longrightarrow$ the only other **memoryless** strategy is **not a solution either**

SPS Problem Example (2/2)

Finite-memory strategy σ'_0 s.t. $\sigma'_0(v_0v_2v_3) = v_5$ and $\sigma'_0(v_0v_2v_3v_5v_3) = v_7$



σ'_0 **is a solution** to the SPS problem: $\rho \in \Omega_0$ when $\text{pay}(\rho) \in P_{\sigma'_0}$
$\longrightarrow$ Player 0 **may need memory** to have a solution to the SPS problem

We consider SP games where **every objective** is

- parity (**parity SP games**): models general class of ω-regular objectives
- reachability (**reachability SP games**): simpler setting

Outline

Our Results on SP games

NEXPTIME-Completeness of the SPS problem

The SPS problem is NEXPTIME-complete for reachability SP games and for parity SP games

Fixed-Parameter Complexity of the SPS problem

Solving the SPS problem is FPT for reachability SP games for parameter t (number of objectives of Player 1) and FPT for parity SP games for parameters t and the maximal priority according to each parity objective of Player 1

Sound: in practice, we can assume those parameters to have **small values**

NEXPTIME algorithm **not** FPT & FPT algorithm **not usable** for membership

Complexity Class

NEXPTIME-Membership

The SPS problem is in NEXPTIME for reachability and for parity SP games

Use **important result on the strategies** which are solution to the problem

- if Player 0 has a solution, he has a **finite-memory one**
- with at most an **exponential number** of memory states

Membership: NEXPTIME **algorithm** where

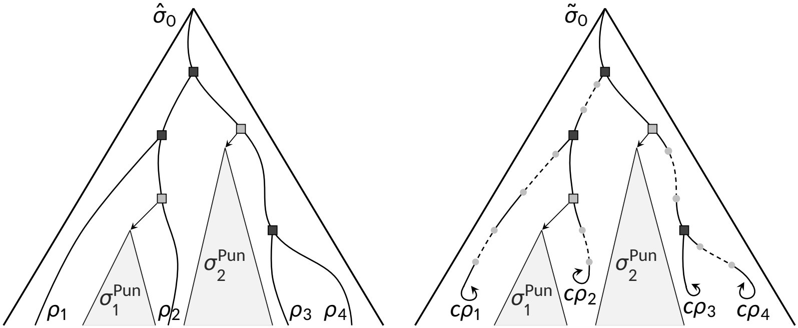
- non-deterministically **guess** a strategy (with exponential size)
- **check** that it is a solution **in exponential time** (using automaton)

Constructing a Finite-Memory Strategy

Start from a solution σ_0 to the SPS problem and **one play ρ_i per PO payoff**

Create $\hat{\sigma}_0$ which **follows** σ_0 in prefix of ρ_i
$\longrightarrow$ on deviation, switch to **punishing strategy** σ^{Pun} that imposes Ω_0 or ~~PO~~

Decompose ρ_i into at most **exponentially many** parts and **compact** them



The SPS problem is NP-hard on Tree Arenas

Simple setting of **tree arenas**: trees with loops on leaves

NP-hardness is shown using the **Set Cover problem** (NP-complete) [Kar72]

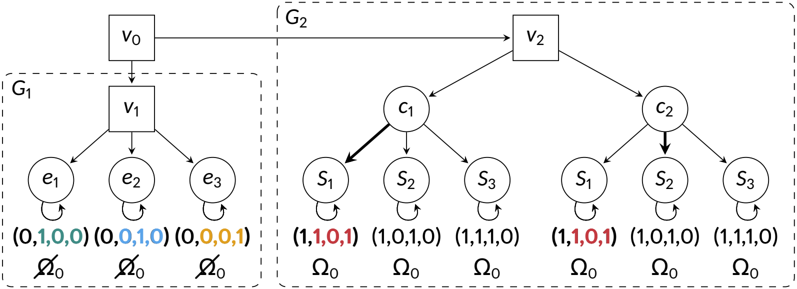
- $C = \{e_1, e_2, \ldots, e_n\}$ of n **elements**
- m **subsets** $S_1, S_2, \ldots, S_m$ s.t. $S_i \subseteq C$
- an **integer k** $\leq m$
- **find** k **indexes** $i_1, i_2, \ldots, i_k$ s.t. $C = \bigcup_{j=1}^{k} S_{i_j}$.

Devise a **SP game** such that:

Player 0 has a **solution** to the **SPS problem** $\Longleftrightarrow$ **solution** to the **SC problem**

Reduction to the Set Cover Problem

$C = \{e_1, e_2, e_3\},\ S_1 = \{e_1, e_3\}, S_2 = \{e_2\},\ S_3 = \{e_1, e_2\},\ k = 2$



Every play in G_1 is **consistent with any strategy** of Player 0 and $\notin \Omega_0$
$\longrightarrow$ in a solution, payoffs from G_1 **cannot be Pareto-Optimal**

Each payoff in G_1 must be **$<$ than some payoff in G_2** (corresponding to a set)

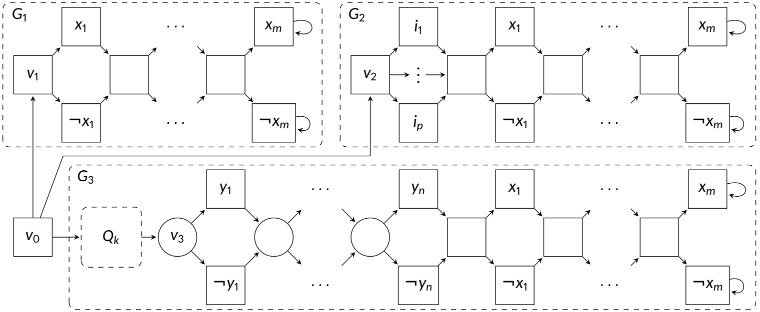
Hardness

NEXPTIME-Hardness

The SPS problem is NEXPTIME-hard for reachability and parity SP games

Intuition: use **succinct variant** of Set Cover problem (NEXPTIME-complete)

$\longrightarrow$ Set Cover problem succinctly defined using **CNF formulas**



Challenger-Prover Game

To show FPT results: reduction to **Challenger-Prover game** (C-P game)

- **two-player zero-sum game** $\mathcal{G}'$, created from $\mathcal{G}$
- played between **Challenger** ($\mathcal{C}$) and **Prover** ($\mathcal{P}$)
- **solution** to the SPS problem in $\mathcal{G} \iff$ **winning strategy** for $\mathcal{P}$ in $\mathcal{G}'$
- described in a **generic way**, later **adapted** to parity/reachability

Intuition: $\mathcal{P}$ tries to **show** the existence of a solution, $\mathcal{C}$ tries to **disprove** it

Bibliography I

[GTW02] Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors.

Automata, Logics, and Infinite Games: A Guide to Current Research [outcome of a Dagstuhl seminar, February 2001], volume 2500 of *Lecture Notes in Computer Science*. Springer, 2002.

[Kar72] Richard M. Karp.

Reducibility among combinatorial problems.

In Raymond E. Miller and James W. Thatcher, editors, *Proceedings of a symposium on the Complexity of Computer Computations, held March 20-22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, USA*, The IBM Research Symposia Series, pages 85–103. Plenum Press, New York, 1972.

Bibliography II

[vS37] Heinrich Freiherr von Stackelberg.
Marktform und Gleichgewicht.
Wien und Berlin, J. Springer, 1937.