Pareto-Rational Verification

Clément Tamines (Université de Mons)

Joint work with

Véronique Bruyère (Université de Mons)

Jean-François Raskin (Université libre de Bruxelles)

June 29, 2022

Highlights ’22

Formal Verification

Motivation: ensure the correctness of systems responsible for **critical tasks**

Formal Verification

Motivation: ensure the correctness of systems responsible for **critical tasks**

Classical approach to **Formal Verification** (FV)

Formal Verification

Motivation: ensure the correctness of systems responsible for **critical tasks**

Classical approach to **Formal Verification** (FV)

- **model of the system** to verify

Formal Verification

Motivation: ensure the correctness of systems responsible for **critical tasks**

Classical approach to **Formal Verification** (FV)

- **model of the system** to verify
- **model of the environment** in which it is executed

Formal Verification

Motivation: ensure the correctness of systems responsible for **critical tasks**

Classical approach to **Formal Verification** (FV)

- **model of the system** to verify
- **model of the environment** in which it is executed
- **specification** φ to be enforced by the system

Formal Verification

Motivation: ensure the correctness of systems responsible for **critical tasks**

Classical approach to **Formal Verification** (FV)

- **model of the system** to verify
- **model of the environment** in which it is executed
- **specification** φ to be enforced by the system

Goal: check if φ **satisfied in all executions** of the system in the environment

Formal Verification

Motivation: ensure the correctness of systems responsible for **critical tasks**

Classical approach to **Formal Verification** (FV)

- **model of the system** to verify
- **model of the environment** in which it is executed
- **specification** φ to be enforced by the system

Goal: check if φ **satisfied in all executions** of the system in the environment

Limitations:

Formal Verification

Motivation: ensure the correctness of systems responsible for **critical tasks**

Classical approach to **Formal Verification** (FV)

- **model of the system** to verify
- **model of the environment** in which it is executed
- **specification** φ to be enforced by the system

Goal: check if φ **satisfied in all executions** of the system in the environment

Limitations:

- check **single behavior** of the system

Formal Verification

Motivation: ensure the correctness of systems responsible for **critical tasks**

Classical approach to **Formal Verification** (FV)

- **model of the system** to verify
- **model of the environment** in which it is executed
- **specification** φ to be enforced by the system

Goal: check if φ **satisfied in all executions** of the system in the environment

Limitations:

- check **single behavior** of the system
- against **potentially irrational behaviors** of environment

The Model

Stackelberg-Pareto game (SP game) [BRT21]: $\mathcal{G} = (G, \Omega_0, \Omega_1, \ldots, \Omega_t)$

The Model

Stackelberg-Pareto game (SP game) [BRT21]: $\mathcal{G} = (G, \Omega_0, \Omega_1, \dots, \Omega_t)$

- Player 0 (system): objective Ω_0

The Model

Stackelberg-Pareto game (SP game) [BRT21]: $\mathcal{G} = (G, \Omega_0, \Omega_1, \dots, \Omega_t)$

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

The Model

Stackelberg-Pareto game (SP game) [BRT21]: $\mathcal{G} = (G, \Omega_0, \Omega_1, \dots, \Omega_t)$

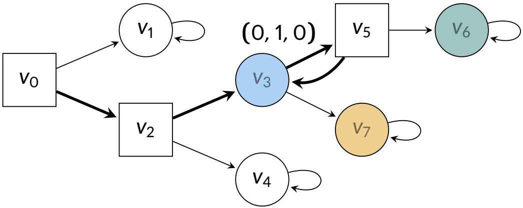
- Player 0 (system): objective Ω_0
- Player 1 (environment): **several objectives** $\Omega_1, \dots, \Omega_t$ (components)

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

$\Omega_1 = \text{Inf}(\{v_6\})$

$\Omega_2 = \text{Inf}(\{v_3\})$

$\Omega_3 = \text{Inf}(\{v_7\})$



The Model

Stackelberg-Pareto game (SP game) [BRT21]: $\mathcal{G} = (G, \Omega_0, \Omega_1, \ldots, \Omega_t)$

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

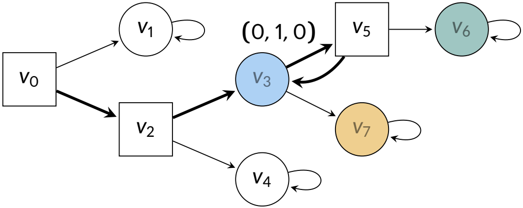
Payoff of ρ for Player 1 is the **vector of Booleans** $\text{pay}(\rho) \in \{0, 1\}^t$

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

$\Omega_1 = \text{Inf}(\{v_6\})$

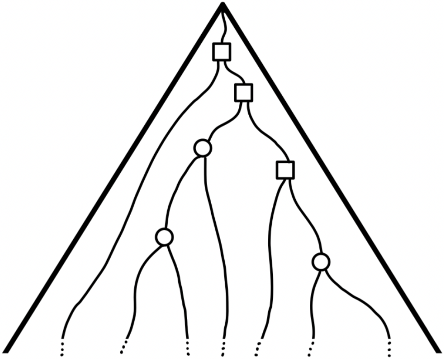
$\Omega_2 = \text{Inf}(\{v_3\})$

$\Omega_3 = \text{Inf}(\{v_7\})$



What Do We Verify?

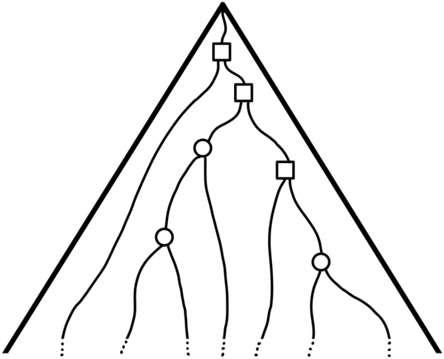
Behavior of the system: **finite-memory strategy** for Player 0



What Do We Verify?

Behavior of the system: **finite-memory strategy** for Player 0

→ verify the **correctness** of this strategy

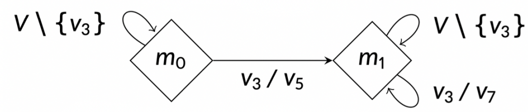


What Do We Verify?

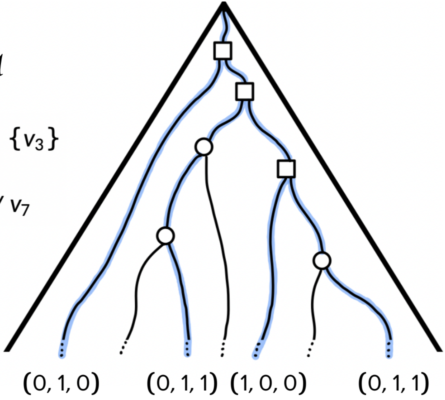
Behavior of the system: **finite-memory strategy** for Player 0

→ verify the **correctness** of this strategy

Deterministic Moore machine $\mathcal{M}$



Set of **consistent plays**: Plays_{σ_0}

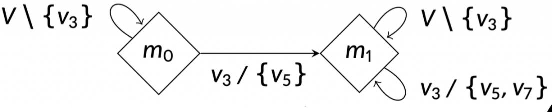


What Do We Verify?

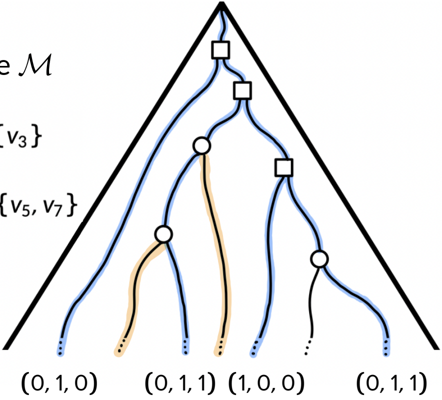
Behavior of the system: **finite-memory strategy** for Player 0

→ verify the **correctness** of this strategy

Nondeterministic Moore machine $\mathcal{M}$



Embeds a **set of strategies** $[\![\mathcal{M}]\!]$



Universal Pareto-Rational Verification Problem (UPRV problem)

Decide whether for all strategies $\sigma_0 \in [\![\mathcal{M}]\!]$ of Player 0, every play $\rho \in \mathsf{Plays}_{\sigma_0}$ with $\mathsf{pay}(\rho) \in P_{\sigma_0}$ are such that $\rho \in \Omega_0$

Universal Pareto-Rational Verification Problem (UPRV problem)

Decide whether for all strategies $\sigma_0 \in [\![\mathcal{M}]\!]$ of Player 0, every play $\rho \in \mathsf{Plays}_{\sigma_0}$ with $\mathsf{pay}(\rho) \in P_{\sigma_0}$ are such that $\rho \in \Omega_0$

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

Universal Pareto-Rational Verification Problem (UPRV problem)

Decide whether for all strategies $\sigma_0 \in [\![\mathcal{M}]\!]$ of Player 0, every play $\rho \in \text{Plays}_{\sigma_0}$ with $\text{pay}(\rho) \in P_{\sigma_0}$ are such that $\rho \in \Omega_0$

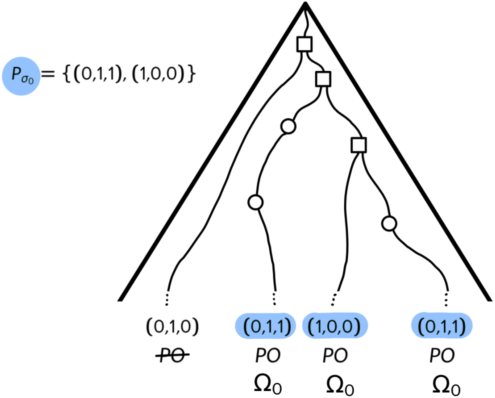
Environment is **rational** and responds to σ_0 **to get a Pareto-optimal payoff**
→ Player 0 must **satisfy** Ω_0 in every such **rational response**

Universal Pareto-Rational Verification Problem (UPRV problem)

Decide whether for all strategies $\sigma_0 \in [\![\mathcal{M}]\!]$ of Player 0, every play $\rho \in \text{Plays}_{\sigma_0}$ with $\text{pay}(\rho) \in P_{\sigma_0}$ are such that $\rho \in \Omega_0$

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

→ Player 0 must **satisfy** Ω_0 in every such **rational response**



Complexity Results

Study both problems for **parity**, **Boolean Büchi**, and **LTL** objectives

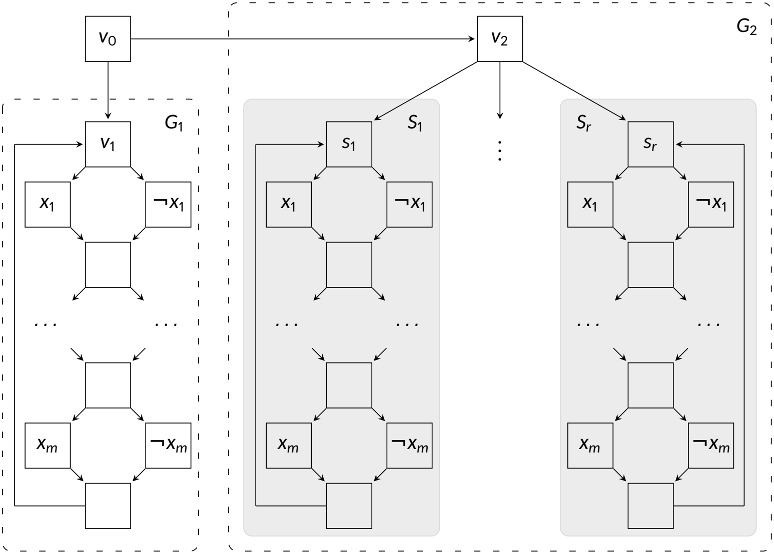
PRV Problem

Objective	Complexity class
Parity	co-NP-complete
Boolean Büchi	Π_2P-complete
LTL	PSPACE-complete

UPRV Problem

Objective	Complexity class
Parity	PSPACE, NP-hard, co-NP-hard
Boolean Büchi	PSPACE-complete
LTL	2EXPTIME-complete

Cool Reductions: co3SAT, Σ_2QBF, solving games, ...



Fixed-Parameter Complexity

PRV and UPRV problem

Both problems are fixed-parameter tractable (FPT) for parity and Boolean Büchi with various parameters

Fixed-Parameter Complexity

PRV and UPRV problem

Both problems are fixed-parameter tractable (FPT) for parity and Boolean Büchi with various parameters

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

Fixed-Parameter Complexity

PRV and UPRV problem

Both problems are fixed-parameter tractable (FPT) for parity and Boolean Büchi with various parameters

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

Additional Algorithm: based on **counterexamples**

Fixed-Parameter Complexity

PRV and UPRV problem

Both problems are fixed-parameter tractable (FPT) for parity and Boolean Büchi with various parameters

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

Additional Algorithm: based on **counterexamples**
→ **implemented and compared** using **toy example** and random instances

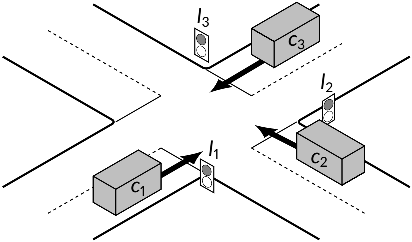
Fixed-Parameter Complexity

PRV and UPRV problem

Both problems are fixed-parameter tractable (FPT) for parity and Boolean Büchi with various parameters

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

Additional Algorithm: based on **counterexamples**
→ **implemented and compared** using **toy example** and random instances



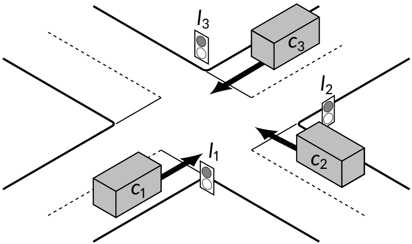
Fixed-Parameter Complexity

PRV and UPRV problem

Both problems are fixed-parameter tractable (FPT) for parity and Boolean Büchi with various parameters

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

Additional Algorithm: based on **counterexamples**
→ **implemented and compared** using **toy example** and random instances



Thank you!

Bibliography

[BRT21] Véronique Bruyère, Jean-François Raskin, and Clément Tamines.
Stackelberg-pareto synthesis.
In Serge Haddad and Daniele Varacca, editors, *32nd International Conference on Concurrency Theory, CONCUR 2021, August 24-27, 2021, Virtual Conference*, volume 203 of *LIPIcs*, pages 27:1–27:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[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.

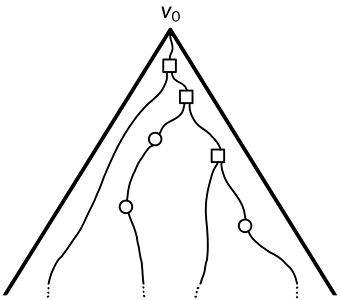
[vS37] Heinrich Freiherr von Stackelberg.
Marktform und Gleichgewicht.
Wien und Berlin, J. Springer, 1937.

Pareto-Optimal Payoffs

1. Player 0 announces his strategy σ_0

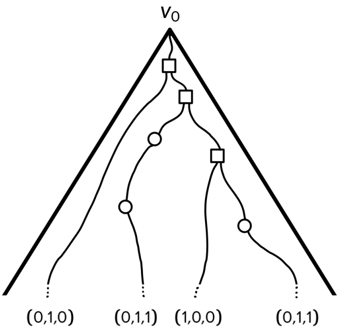
Pareto-Optimal Payoffs

1. Player 0 announces his strategy σ_0
2. Player 1 considers Plays_{σ_0}



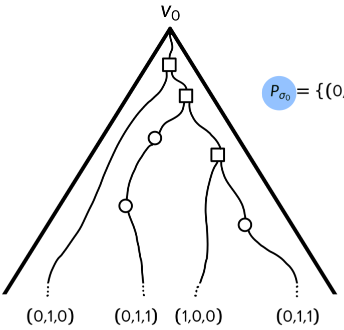
Pareto-Optimal Payoffs

1. Player 0 announces his strategy σ_0
2. Player 1 considers Plays_{σ_0}
   - corresponding **set of payoffs** $\{\text{pay}(\rho) \mid \rho \in \text{Plays}_{\sigma_0}\}$



Pareto-Optimal Payoffs

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



$P_{\sigma_0} = \{(0,1,1), (1,0,0)\}$