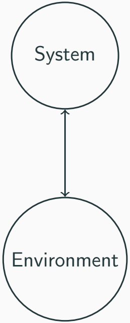
Efficient Trace Encodings of Bounded Synthesis for Asynchronous Distributed Systems

Jesko Hecking-Harbusch, **Niklas O. Metzger**

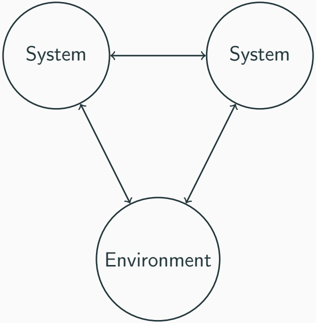
October 30, 2019

Saarland University - Reactive Systems Group

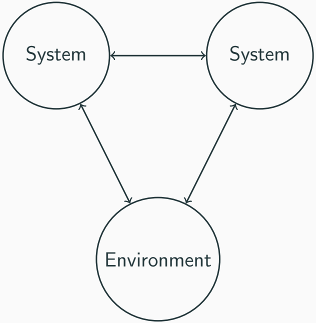
Motivation - Distributed Synthesis



Motivation - Distributed Synthesis



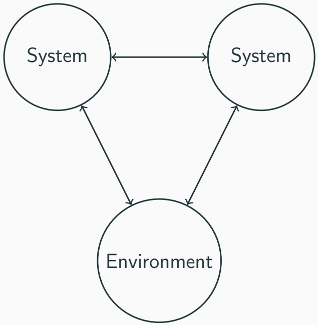
Motivation - Distributed Synthesis



- ✗ Distributed Systems are hard to synthesize [1]
- ✓ Petri games as framework for distributed synthesis [2]
- ✓ Bounded Synthesis for Petri games[3]

[1] Pnueli and Rosner, "Distributed Reactive Systems Are Hard to Synthesize".
[2] Finkbeiner and Olderog, "Petri Games: Synthesis of Distributed Systems with Causal Memory".
[3] Finkbeiner, "Bounded Synthesis for Petri Games".

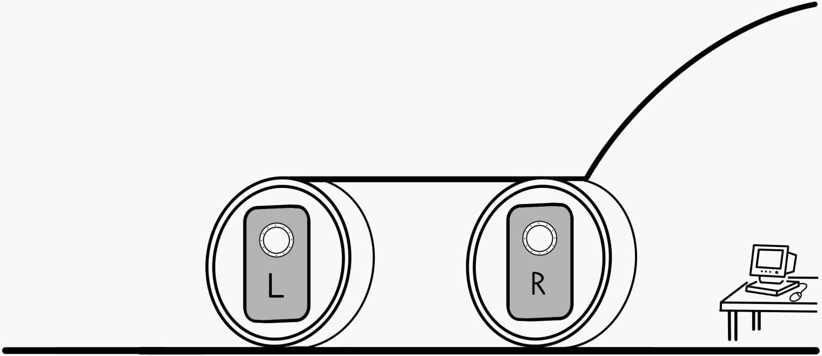
Motivation - Distributed Synthesis



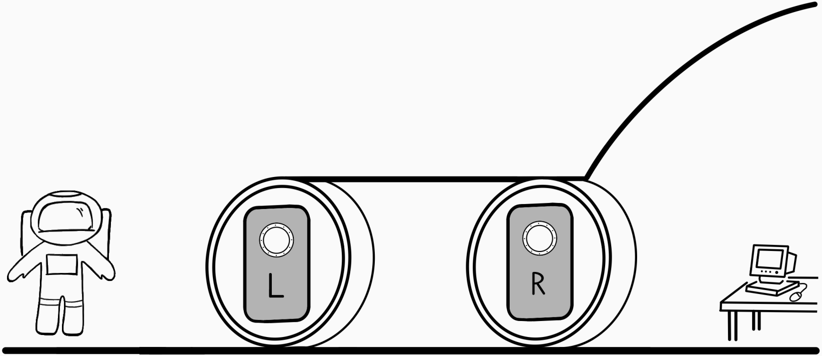
- ✗ Distributed Systems are hard to synthesize [1]
- ✓ Petri games as framework for distributed synthesis [2]
- ✓ Bounded Synthesis for Petri games[3]
- ✗ Asynchronous nature is encoded to interleaving

[1] Pnueli and Rosner, "Distributed Reactive Systems Are Hard to Synthesize".
[2] Finkbeiner and Olderog, "Petri Games: Synthesis of Distributed Systems with Causal Memory".
[3] Finkbeiner, "Bounded Synthesis for Petri Games".

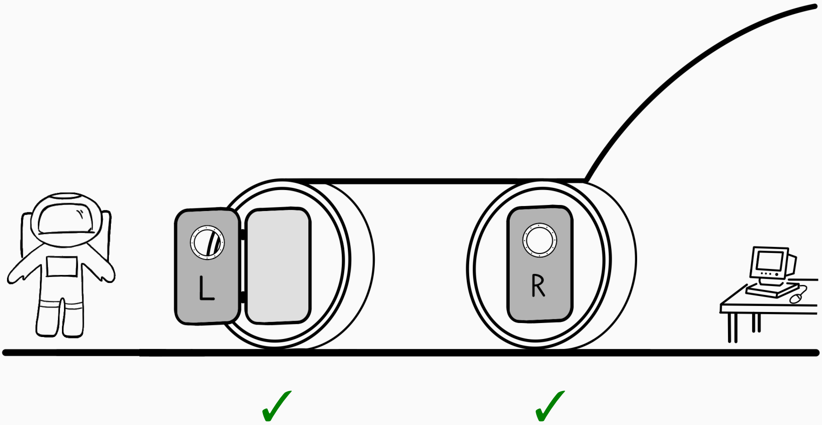
The Martian Problem



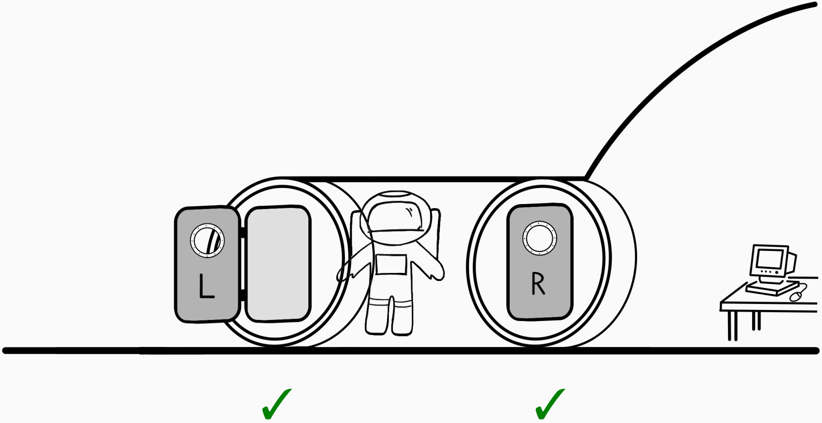
The Martian Problem



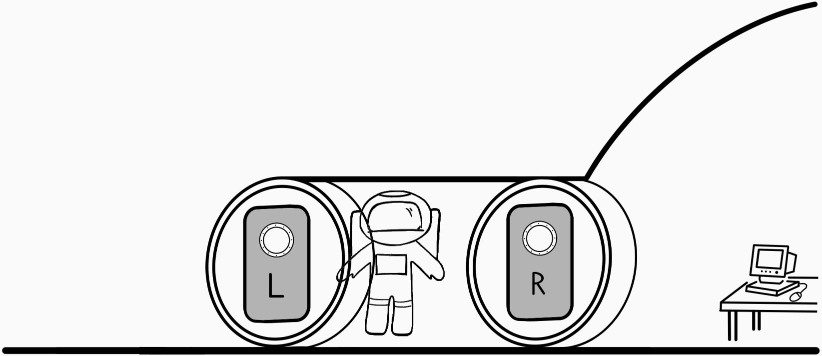
The Martian Problem



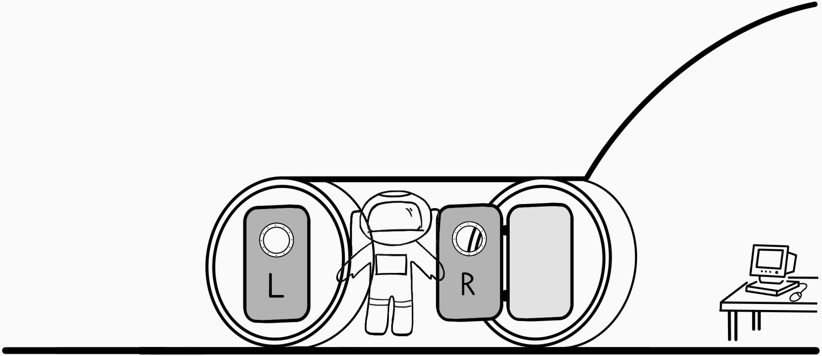
The Martian Problem



The Martian Problem

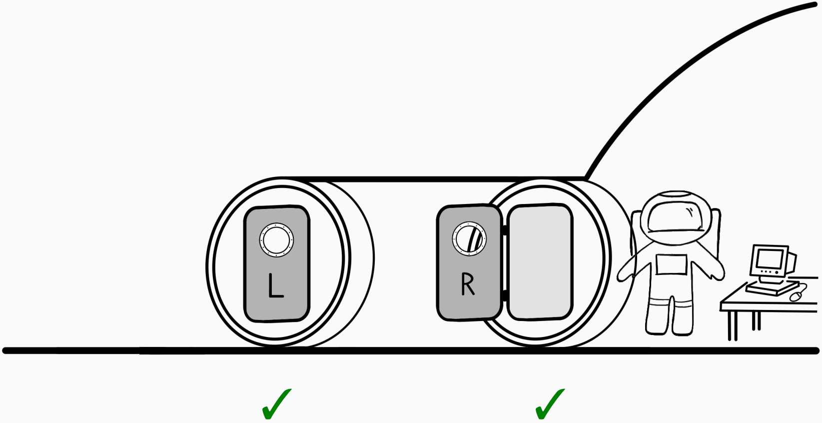


The Martian Problem

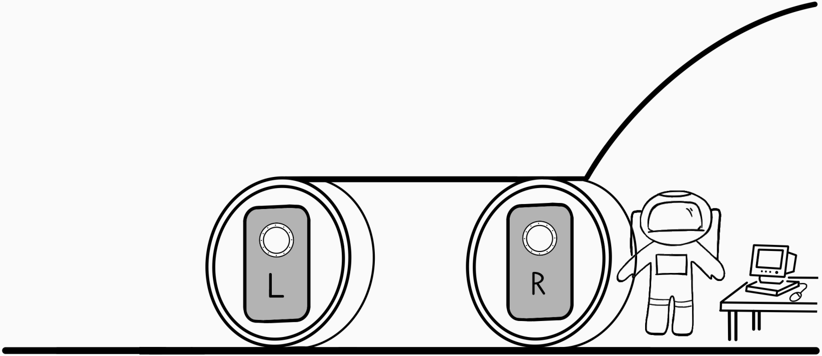


✓ ✓

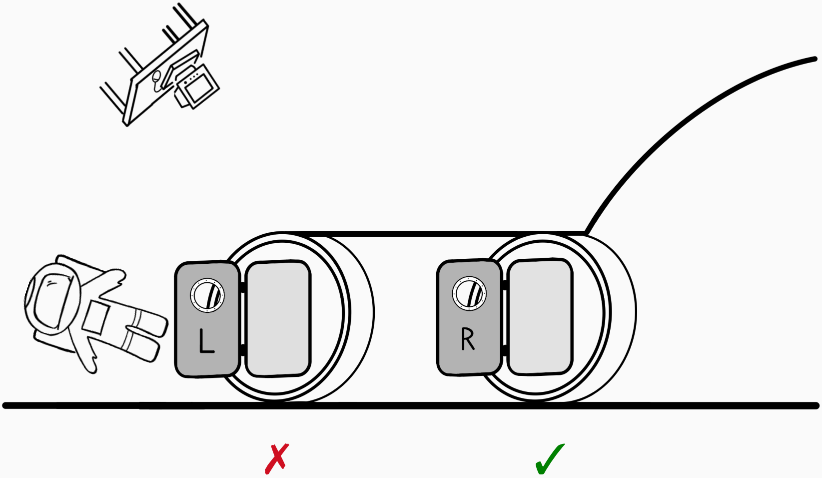
The Martian Problem



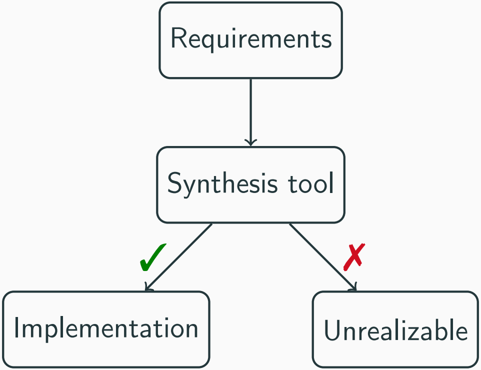
The Martian Problem



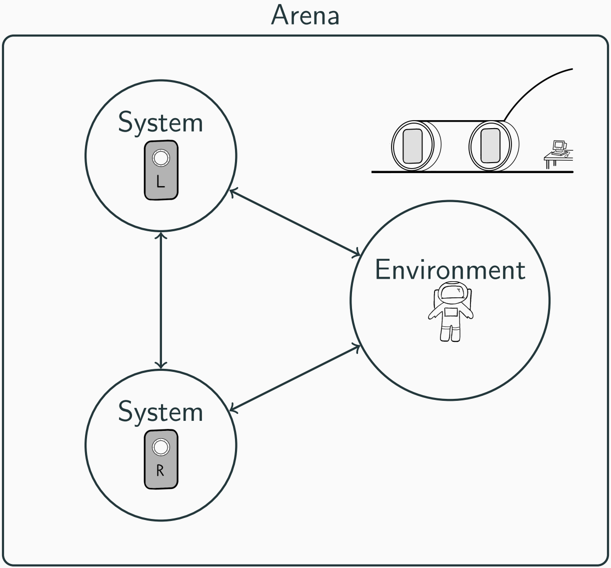
The Martian Problem



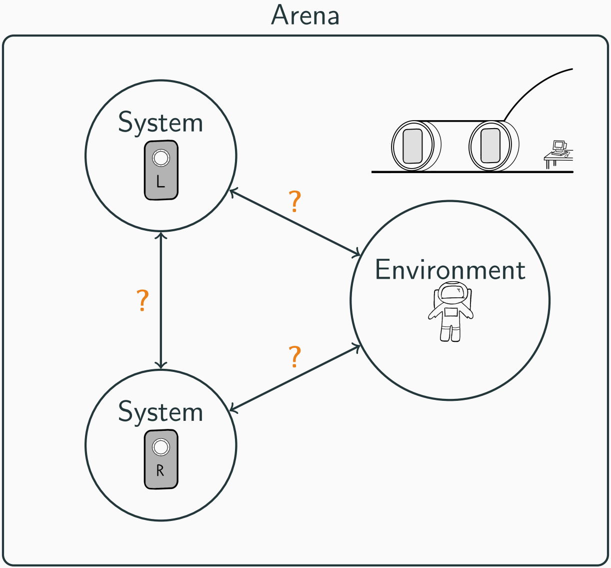
Synthesis


Requirements
Synthesis tool
✓
✗
Implementation
Unrealizable

Synthesis of Distributed Systems as a Game



Synthesis with Local Information

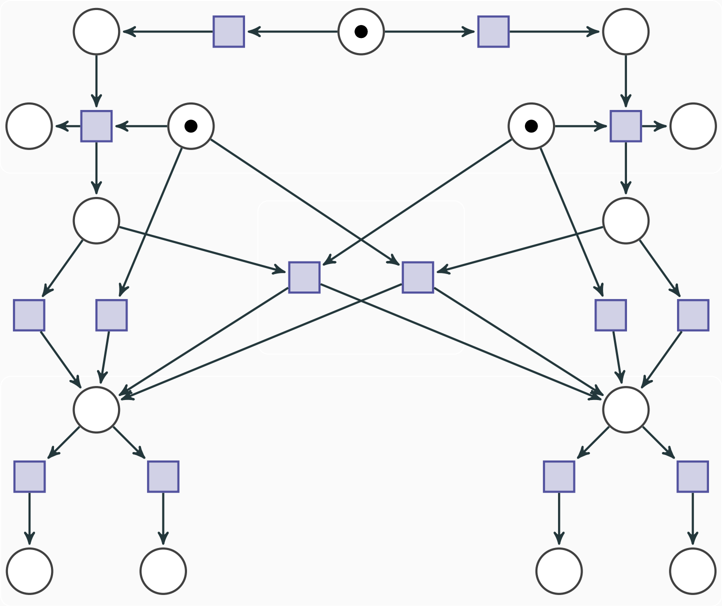


Overview

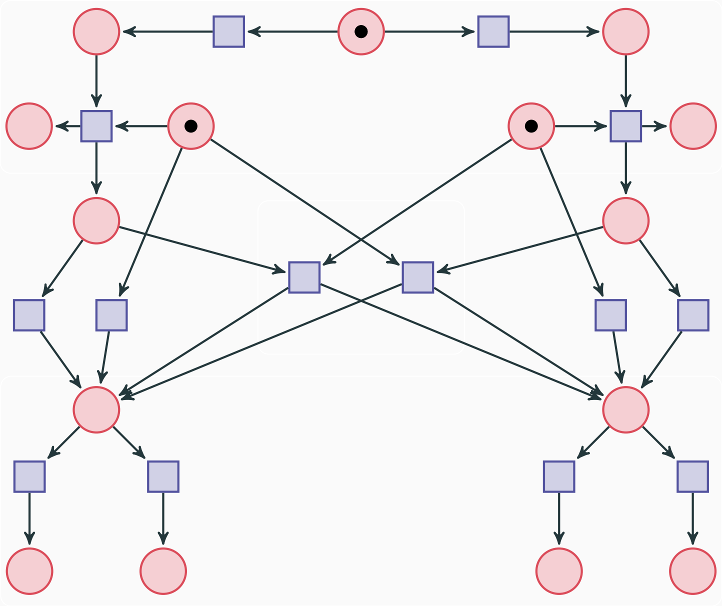
1. Petri Games
2. Bounded Synthesis
3. True Concurrency in Petri Games
4. True Concurrency in Bounded Synthesis of Petri Games

Petri Games

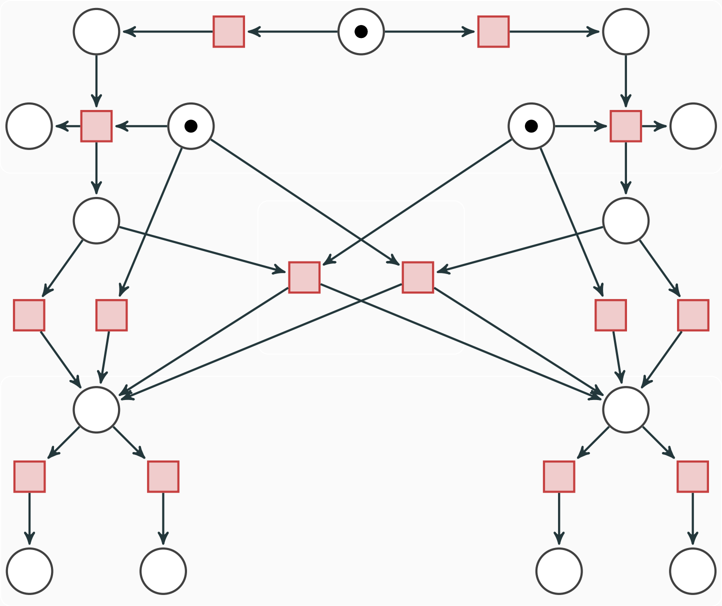
Petri Net as Game Arena of Petri Game



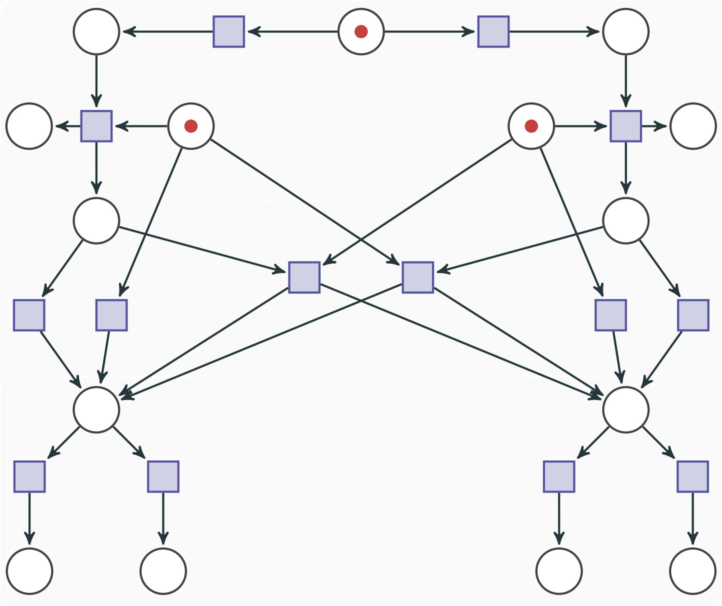
Places $\mathscr{P}$ in a Petri Net



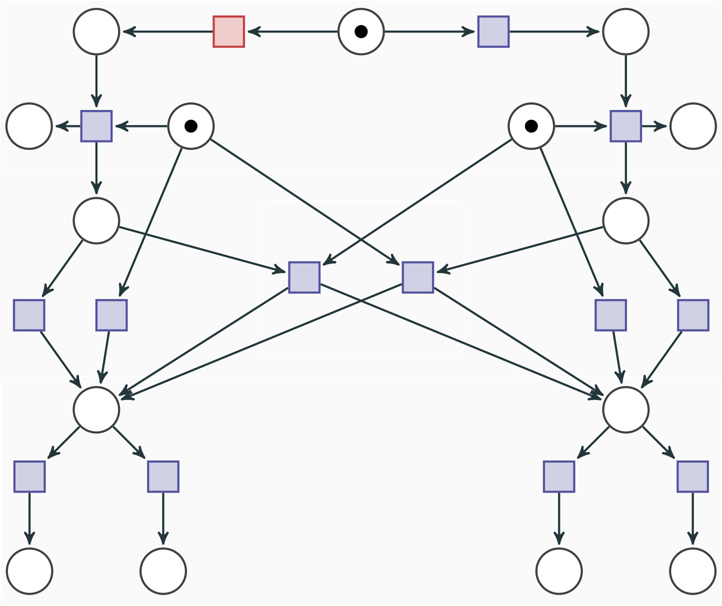
Transitions $\mathscr{T}$ in a Petri Net



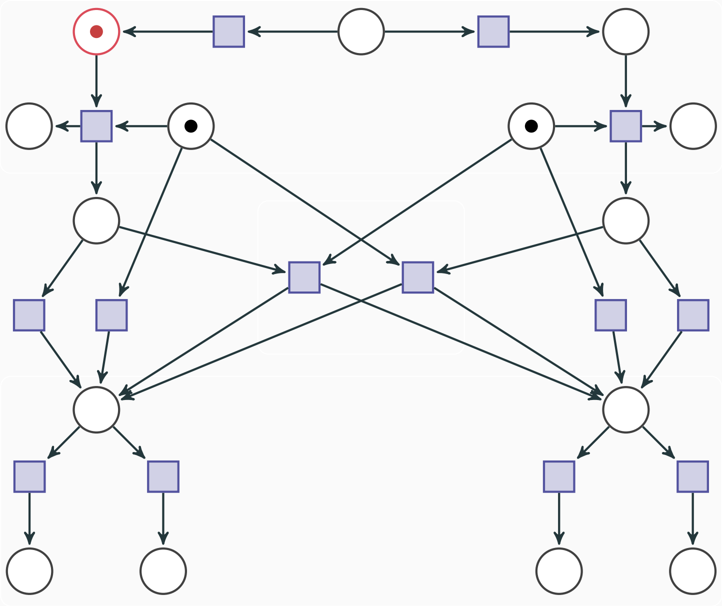
Tokens in a Petri Net



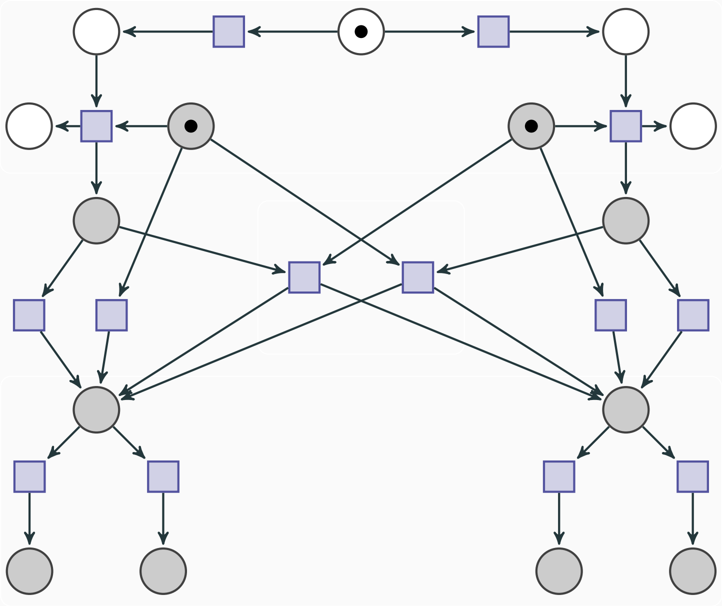
An enabled transition...



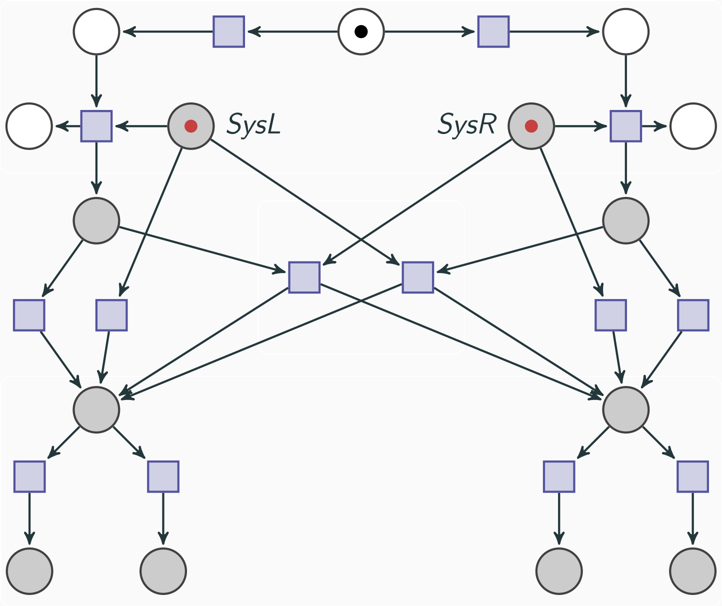
...can be fired in a Petri Net



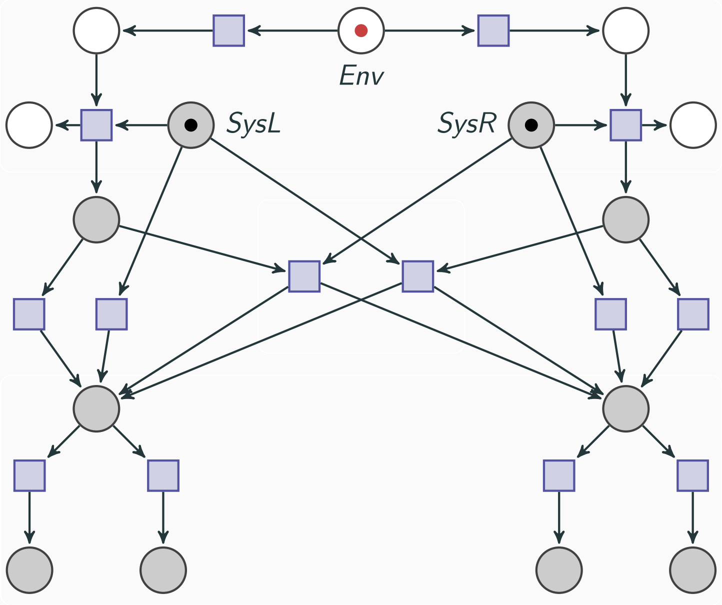
From Net to Game



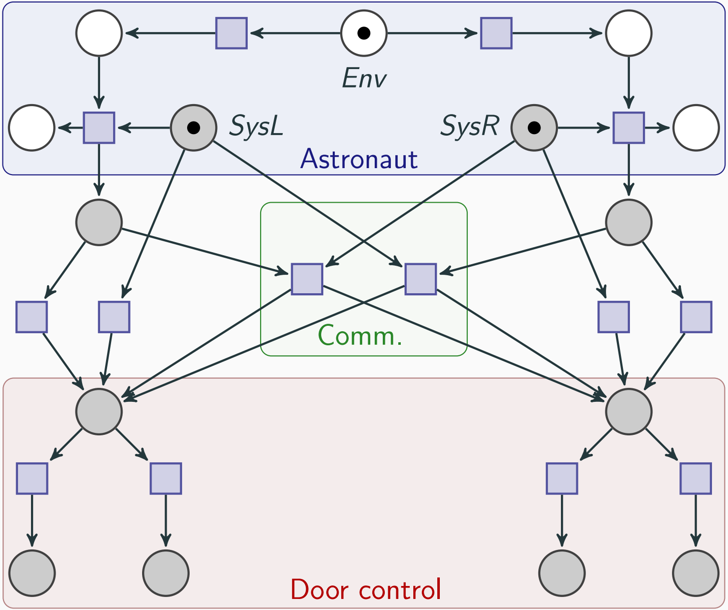
Two System Players



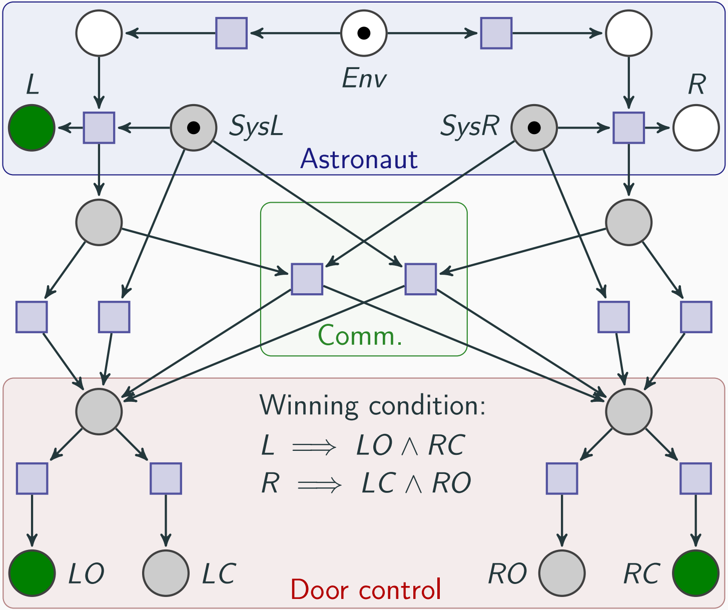
One Environment Player



Airlock as Petri Game



Winning Conditions of the Petri Game

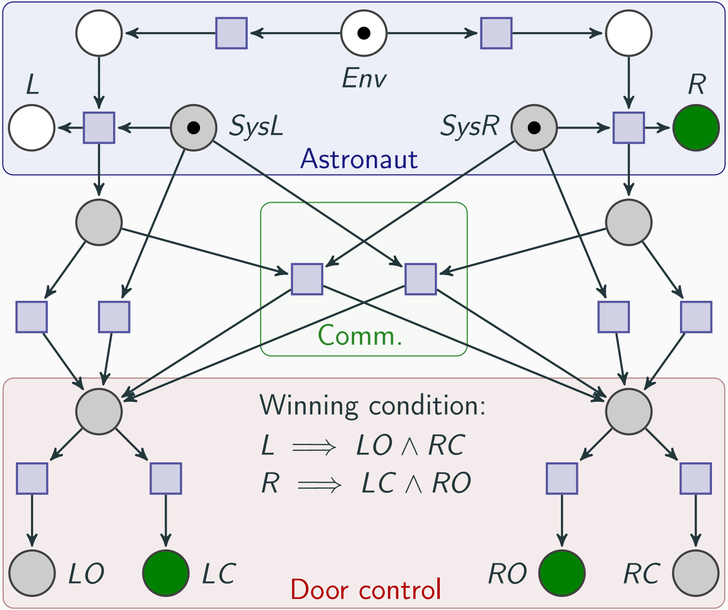


Winning condition:

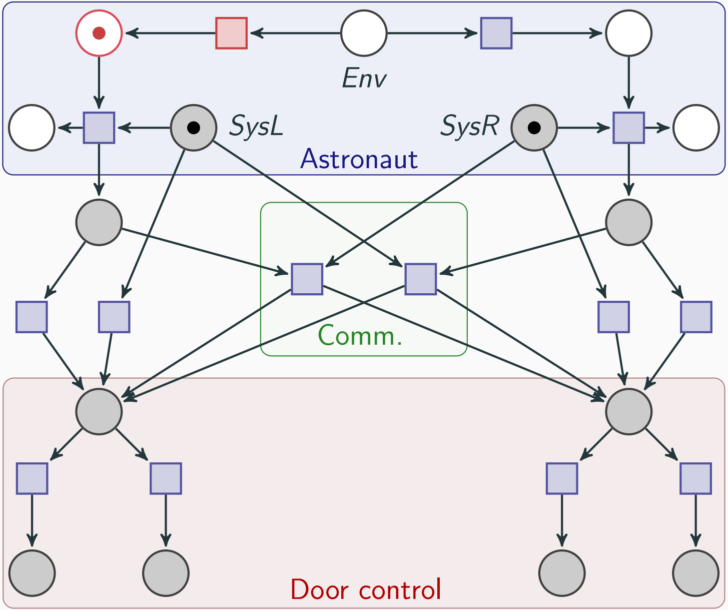
$L \implies LO \wedge RC$

$R \implies LC \wedge RO$

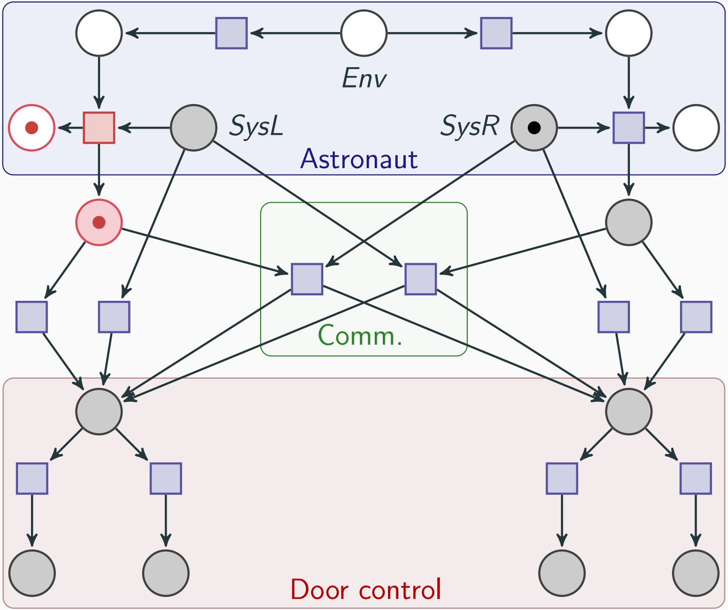
Winning Conditions of the Petri Game



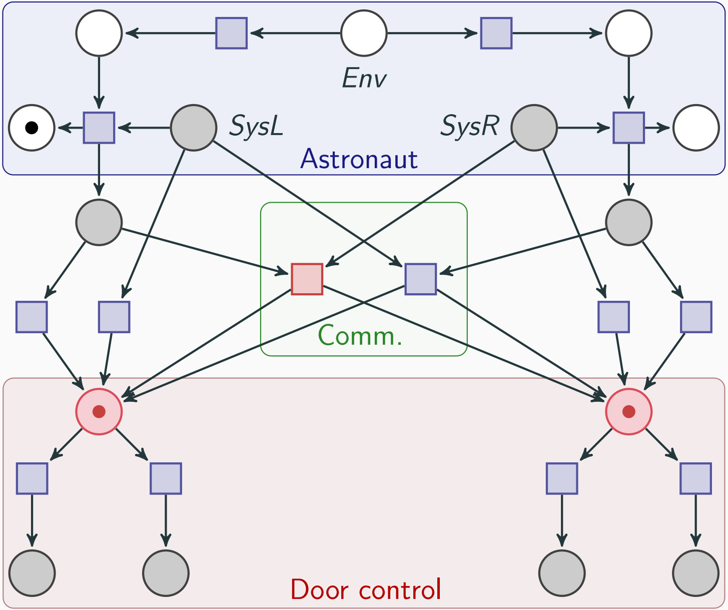
Decision for left Door



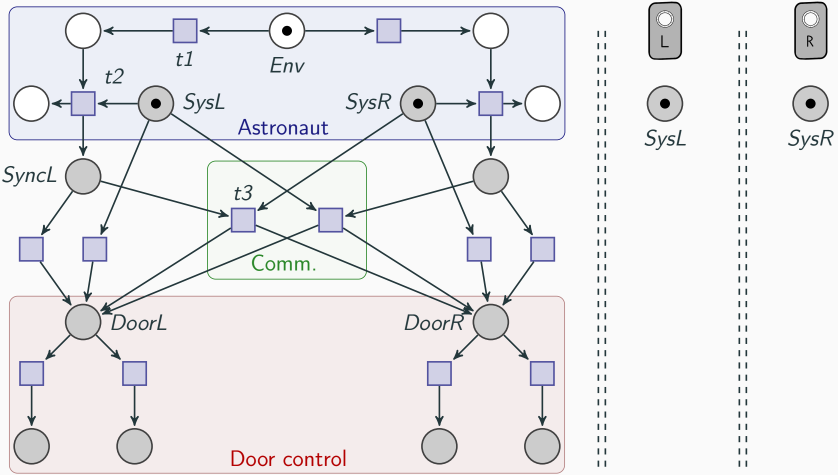
Synchronization with the System



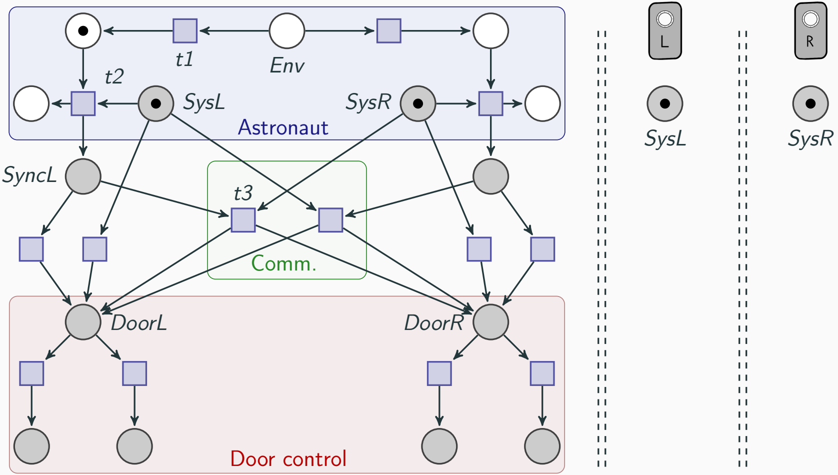
Exchange of Information



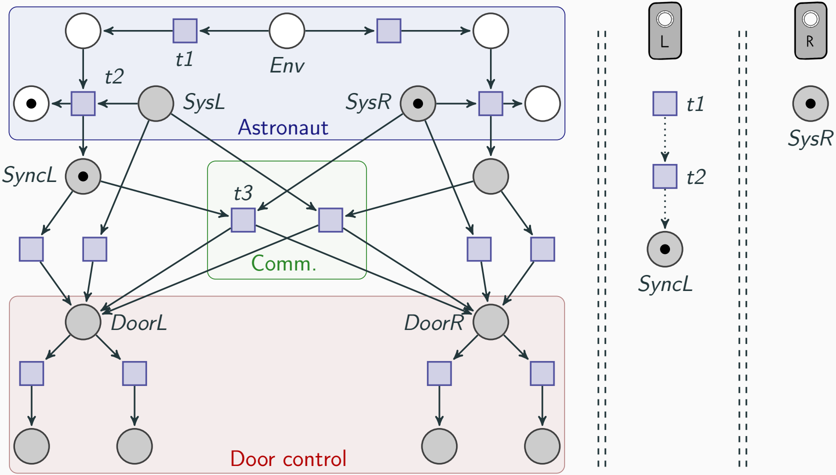
Memory Model of Petri Games: Causal Past



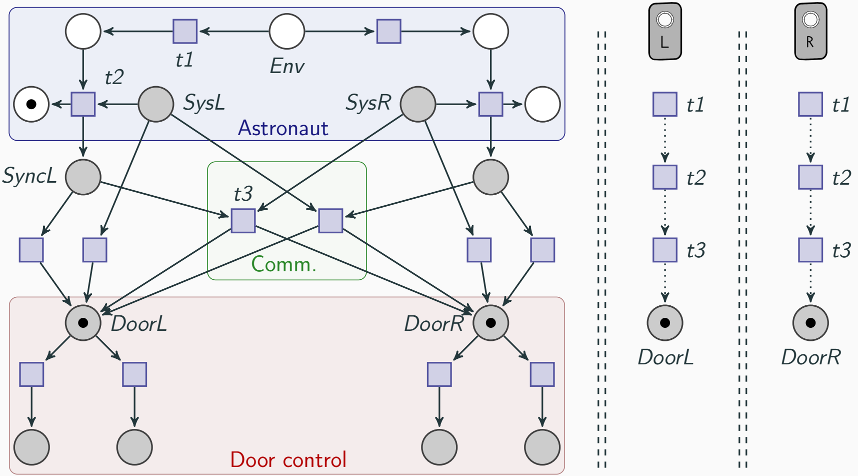
Memory Model of Petri Games: Causal Past



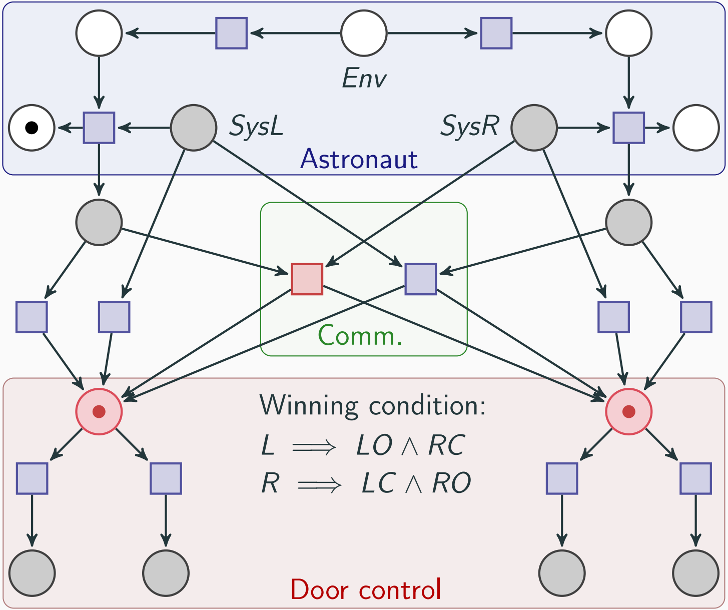
Memory Model of Petri Games: Causal Past



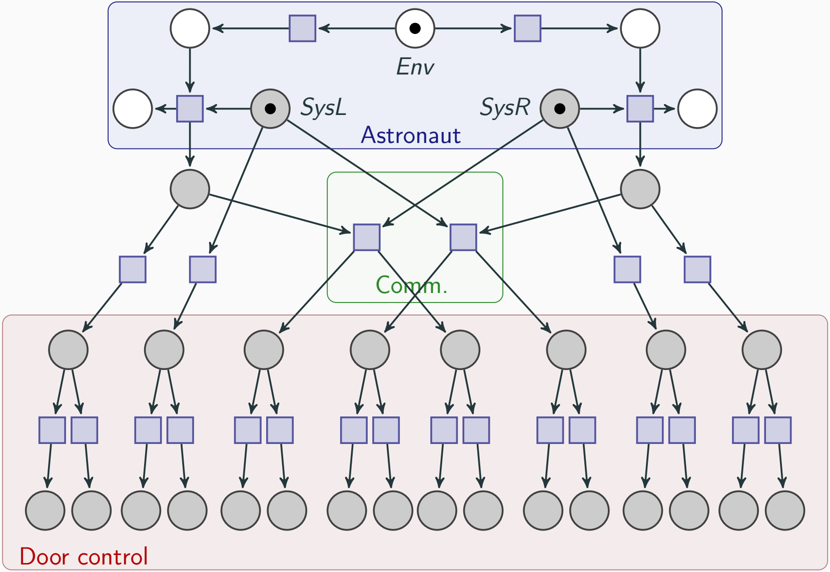
Memory Model of Petri Games: Causal Past



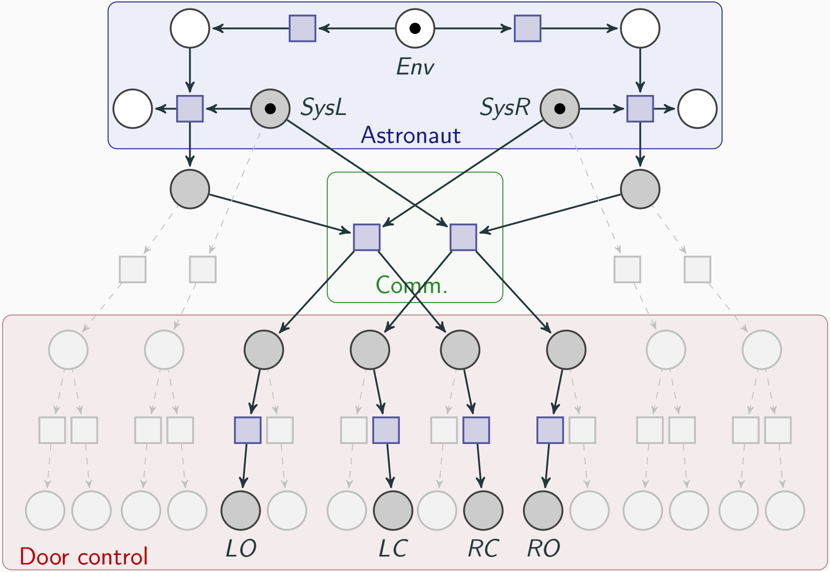
Refuse transitions based on *Causal Past*



Unfolding of Airlock



Winning Strategy of Airlock σ



Outcome of the Petri Game

Reachable Markings

$$\mathscr{R}(\mathscr{N}) = \{M \subseteq \mathscr{P} \mid \exists t_1, ..., t_n \in \mathscr{T} : \exists M_1, ..., M_n \subseteq \mathscr{P} : In[t_1\rangle M_1 ... [t_n\rangle M_n = M\}$$

Winning Safety Condition

A system strategy σ is *winning* for the condition *safety* ($\mathscr{B}$) iff

$$\forall M \in \mathscr{R}(\mathscr{N}^\sigma) : \sigma[M] \cap \mathscr{B} = \emptyset.$$

Outcome of the Petri Game

Reachable Markings

$$\mathcal{R}(\mathcal{N}) = \{M \subseteq \mathcal{P} \mid \exists t_1, ..., t_n \in \mathcal{T} : \exists M_1, ..., M_n \subseteq \mathcal{P} : In[t_1\rangle M_1 ... [t_n\rangle M_n = M\}$$

Winning Safety Condition

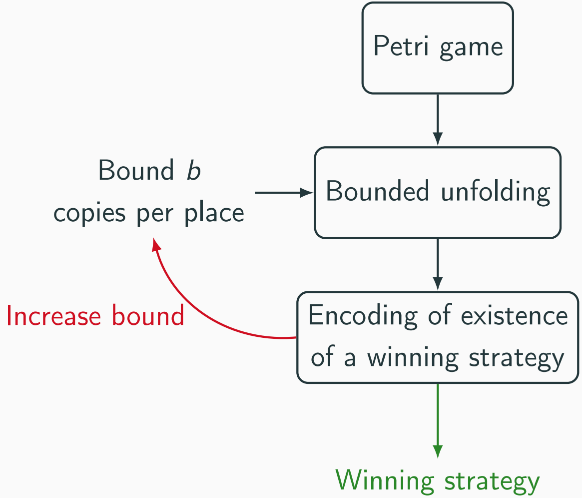
A system strategy σ is *winning* for the condition *safety* ($\mathcal{B}$) iff

$$\forall M \in \mathcal{R}(\mathcal{N}^{\sigma}) : \sigma[M] \cap \mathcal{B} = \emptyset.$$

A Petri game $\mathcal{G}$ is winning iff there exists a winning strategy.

Bounded Synthesis

Bounded Synthesis for Petri Games



Sequential Encoding

Quantified Boolean Formula (QBF):

$$\exists \mathscr{S} : \forall \mathscr{M} : \phi$$

Variables encoding choices of the *strategy*.

Sequential Encoding

Quantified Boolean Formula (QBF):

$$\exists \mathscr{S} : \forall \mathscr{M} : \phi$$

Variables encoding choices of the *strategy*.

Variables encoding all possible sequences of *markings*.

Sequential Encoding

Quantified Boolean Formula (QBF):

$$\exists \mathscr{S} : \forall \mathscr{M} : \phi$$

Variables encoding choices of the *strategy*.

Boolean formula encoding whether the strategy is *winning*.

Variables encoding all possible sequences of *markings*.

Sequential Encoding

Quantified Boolean Formula (QBF):

$$\exists \mathscr{S} : \forall \mathscr{M} : \phi$$

Variables encoding choices of the *strategy*.

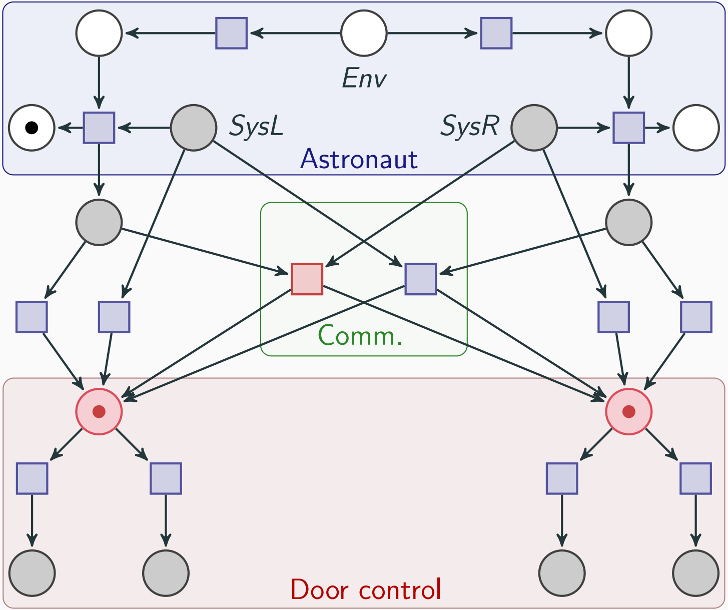
Variables encoding all possible sequences of *markings*.

Boolean formula encoding whether the strategy is *winning*.

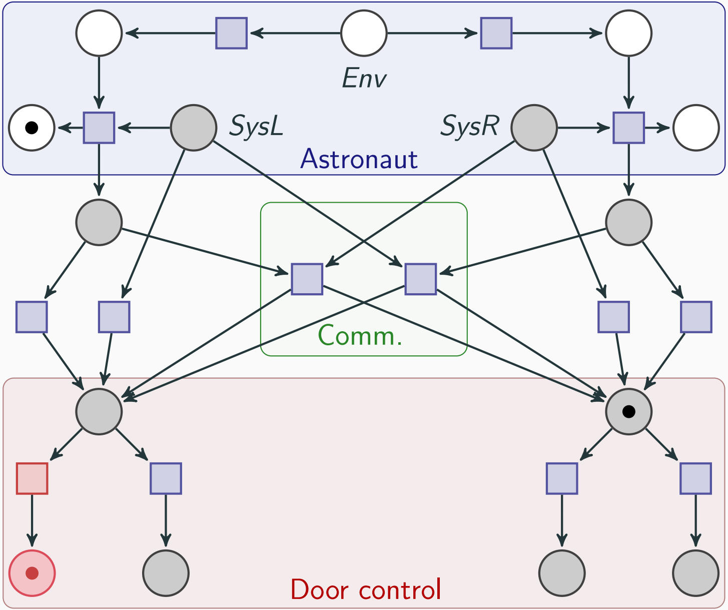
$\phi = \textit{validStrategy} \wedge \textit{validSequence} \wedge \textit{terminating} \wedge \textit{winningStrategy}$

True Concurrency in Petri Games

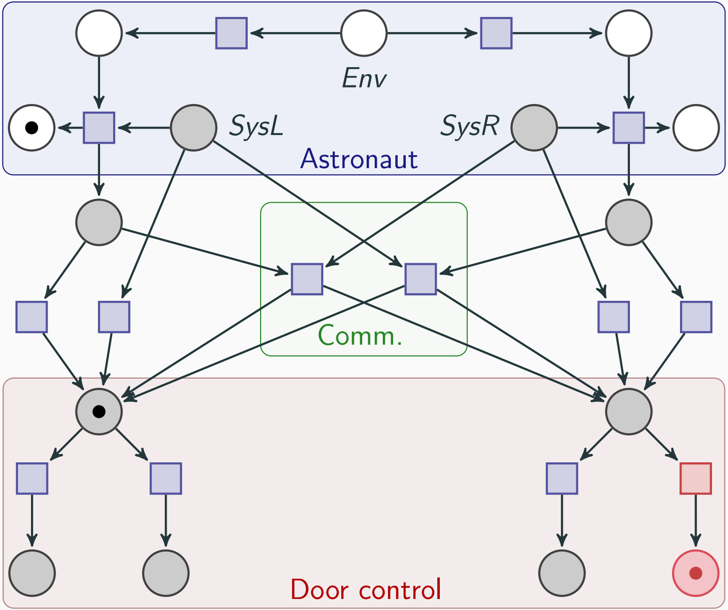
Which Player progresses Next?



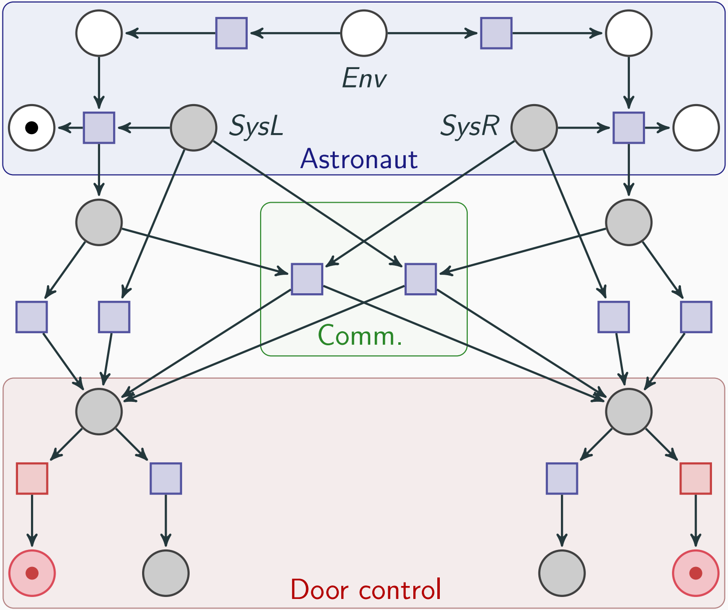
Left Door can be First



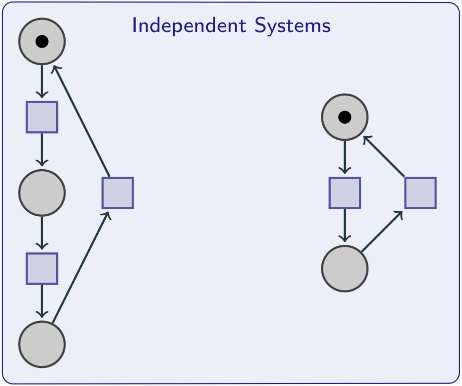
Right Door can be First



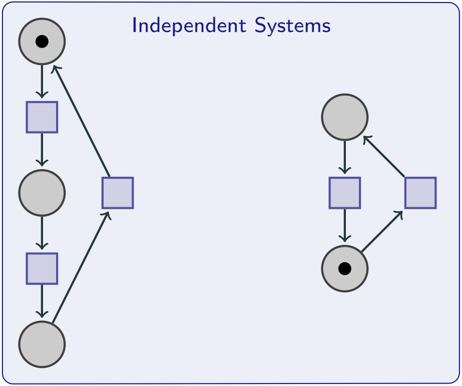
Both System Players



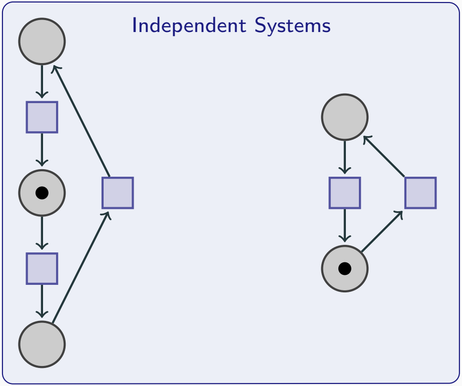
Sequential Firing



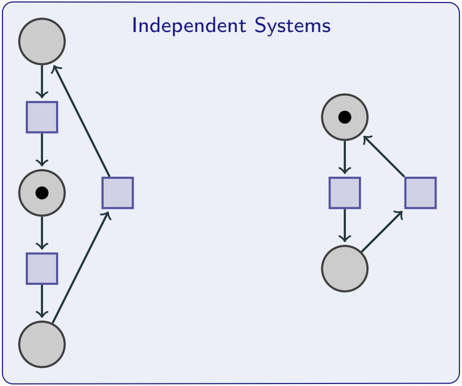
Sequential Firing



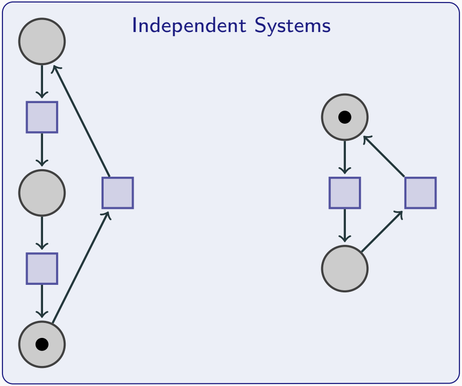
Sequential Firing



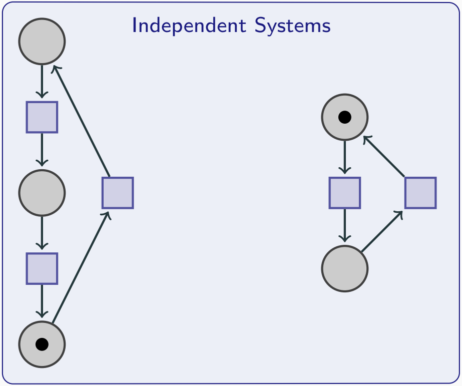
Sequential Firing



Sequential Firing

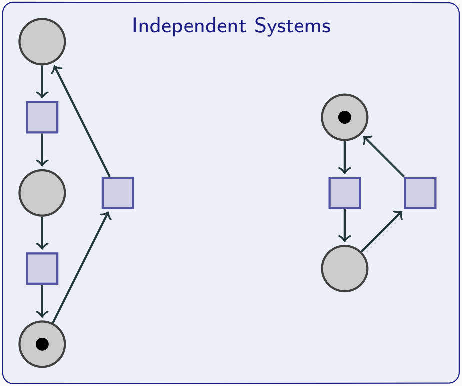


Sequential Firing



Many interleavings with same causal past!

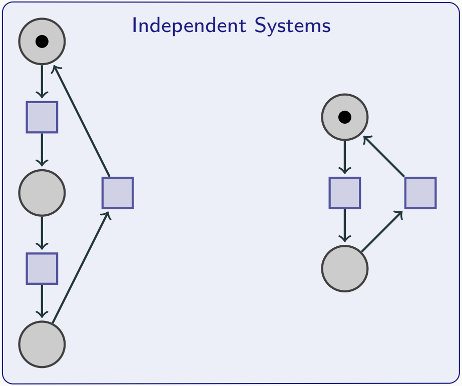
Sequential Firing



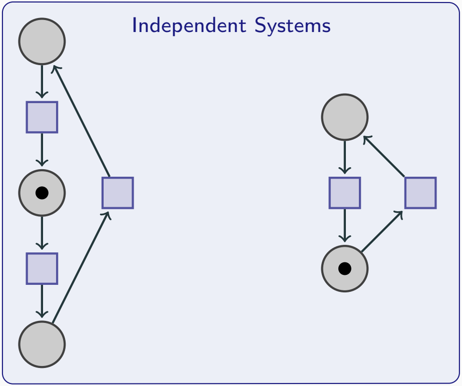
Many interleavings with same causal past!

Fire all enabled transitions

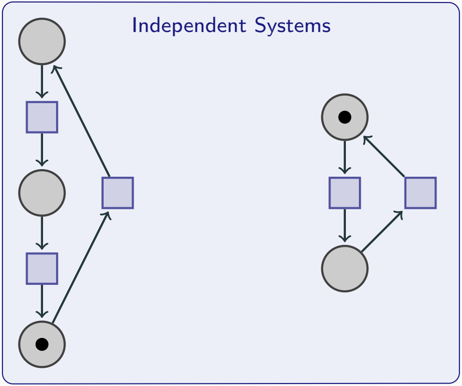
True Concurrent Firing



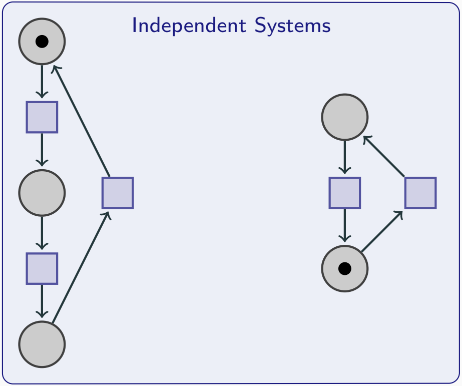
True Concurrent Firing



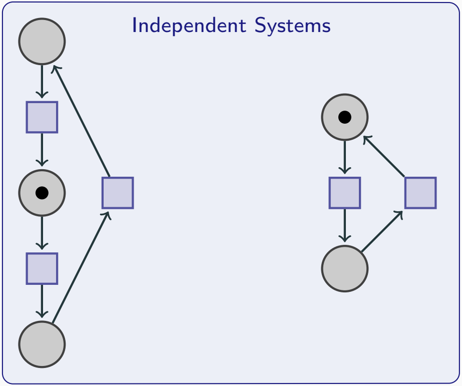
True Concurrent Firing



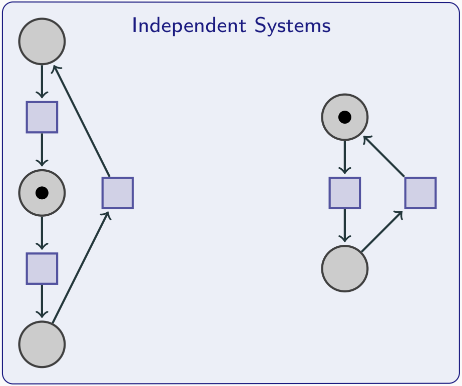
True Concurrent Firing



True Concurrent Firing

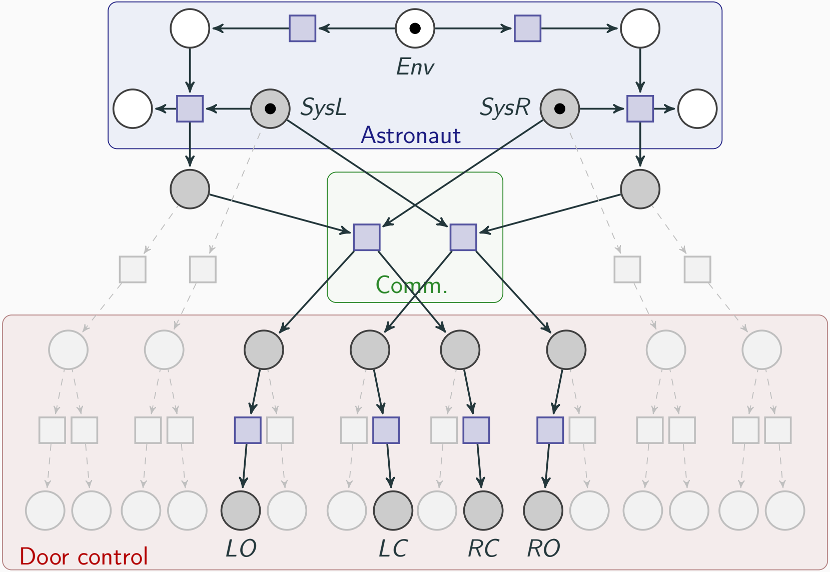


True Concurrent Firing

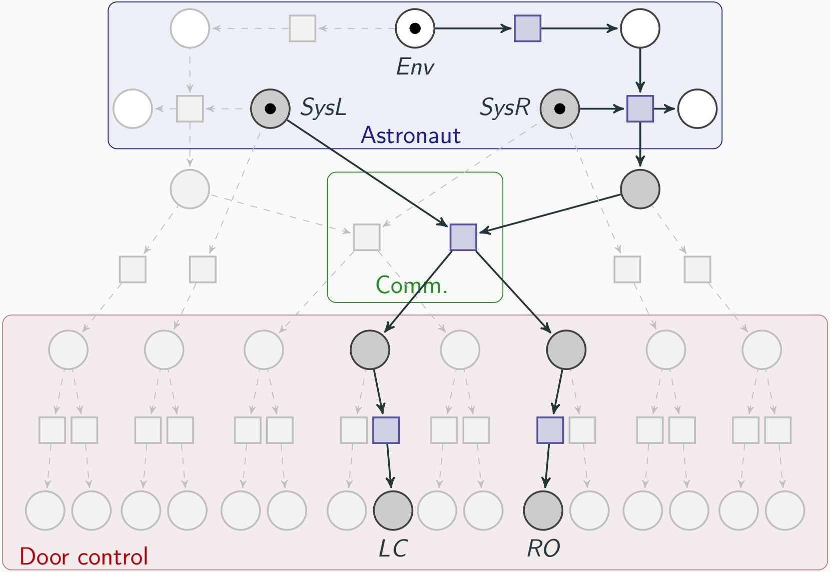


How to remain correct?

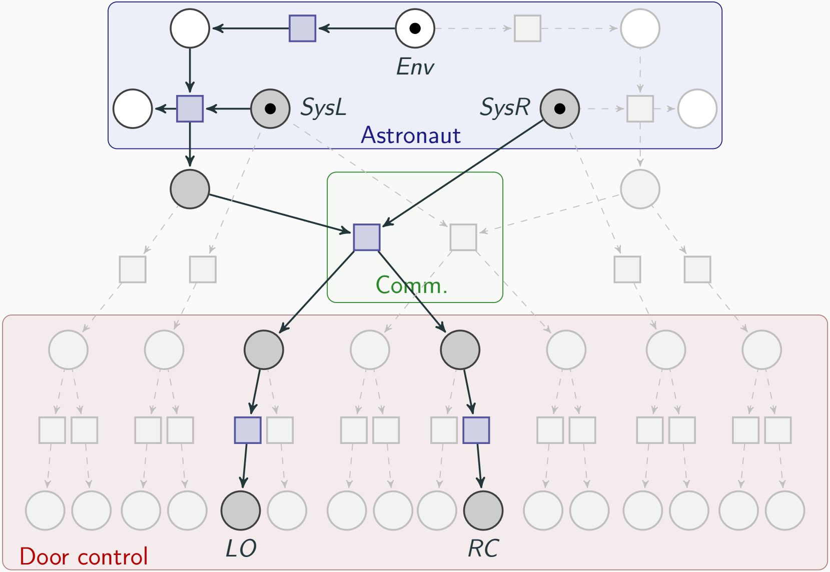
Environment Strategies for Airlock



Environment Strategies for Airlock



Environment Strategies for Airlock



Environment Strategy

Definition

An *environment strategy* γ is a subnet of a system strategy σ that satisfies the conditions *explicit choice*, *environmental refusal*, *and progress*.

Environment Strategy

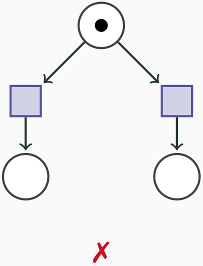
Definition

An *environment strategy* γ is a subnet of a system strategy σ that satisfies the conditions *explicit choice, environmental refusal, and progress*.

Explicit choice

Environmental refusal

Progress



Environment Strategy

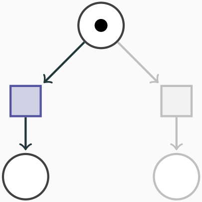
Definition

An *environment strategy* γ is a subnet of a system strategy σ that satisfies the conditions *explicit choice*, *environmental refusal*, *and progress*.

Explicit choice

Environmental refusal

Progress



✓

Environment Strategy

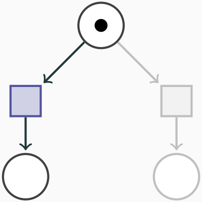
Definition

An *environment strategy* γ is a subnet of a system strategy σ that satisfies the conditions *explicit choice, environmental refusal, and progress*.

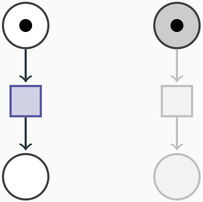
Explicit choice

Environmental refusal

Progress



✓



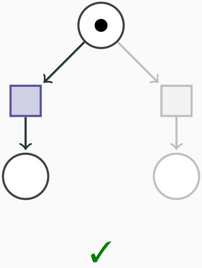
✗

Environment Strategy

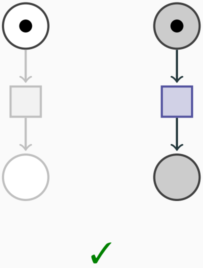
Definition

An *environment strategy* γ is a subnet of a system strategy σ that satisfies the conditions *explicit choice, environmental refusal, and progress*.

Explicit choice



Environmental refusal



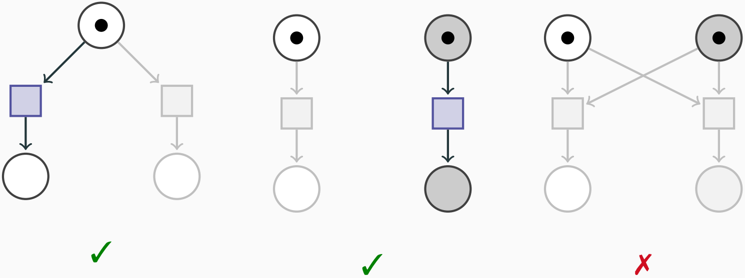
Progress

Environment Strategy

Definition

An *environment strategy* γ is a subnet of a system strategy σ that satisfies the conditions *explicit choice*, *environmental refusal*, *and progress*.

Explicit choice | Environmental refusal | Progress

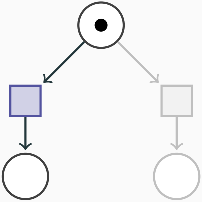


Environment Strategy

Definition

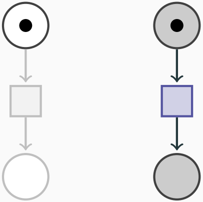
An *environment strategy* γ is a subnet of a system strategy σ that satisfies the conditions *explicit choice, environmental refusal, and progress*.

Explicit choice



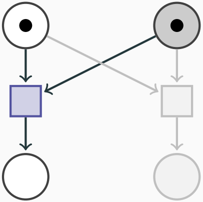
✓

Environmental refusal



✓

Progress



✓

Unique Transition Sequence

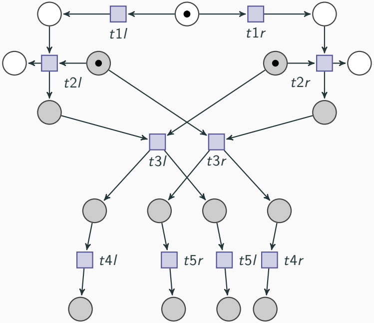
Theorem

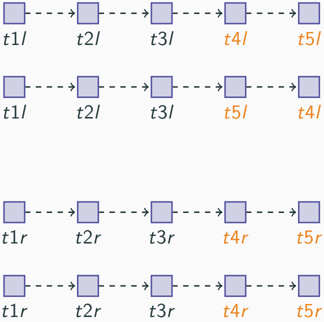
An environment strategy γ leads to a *unique sequence* of fired transitions up to reordering of independent transitions.

Unique Transition Sequence

Theorem

An environment strategy γ leads to a *unique sequence* of fired transitions up to reordering of independent transitions.

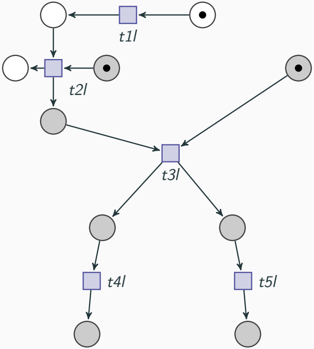


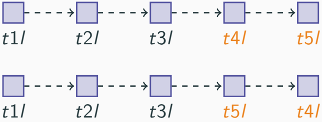


Unique Transition Sequence

Theorem

An environment strategy γ leads to a *unique sequence* of fired transitions up to reordering of independent transitions.

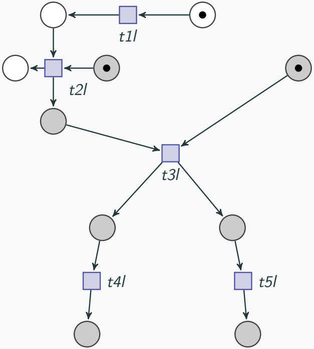


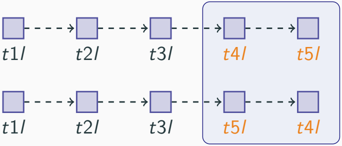


Unique Transition Sequence

Theorem

An environment strategy γ leads to a *unique sequence* of fired transitions up to reordering of independent transitions.

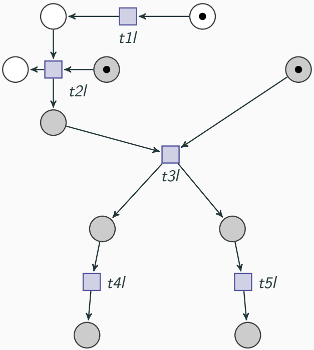


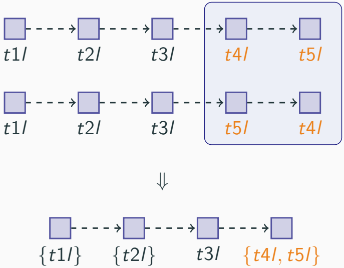


Unique Transition Sequence

Theorem

An environment strategy γ leads to a *unique sequence* of fired transitions up to reordering of independent transitions.





Reachable Markings

$$\mathscr{R}^{seq}(\mathscr{N}) = \{M \subseteq \mathscr{P} \mid \exists t_1, ..., t_n \in \mathscr{T} : \exists M_1, ..., M_n \subseteq \mathscr{P} : In[t_1\rangle M_1 ... [t_n\rangle M_n = M\}$$

$$\mathscr{R}^{tc}(\mathscr{N}) = \{M \subseteq \mathscr{P} \mid \exists T_1, \ldots, T_n \subseteq \mathscr{T} : \exists M_1, \ldots, M_n \subseteq \mathscr{P} : In[T_1\rangle M_1 \ldots [T_n\rangle M_n = M\}$$

Outcome of the Petri Game

A system strategy σ is *winning* for the condition *safety* ($\mathscr{B}$) iff

$$\forall\gamma : \forall M \in \mathscr{R}(\mathscr{N}^{\sigma\gamma}) : \sigma\gamma[M] \cap \mathscr{B} = \emptyset.$$

Correctness

Theorem

The true concurrent semantics is correct iff:

$$\forall \gamma : \forall M \in \mathscr{R}^{tc}(\mathcal{N}^{\sigma\gamma}) : \sigma\gamma[M] \cap \mathscr{B} = \emptyset$$

$$\Leftrightarrow$$

$$\forall M \in \mathscr{R}^{seq}(\mathcal{N}^{\sigma}) : \sigma[M] \cap \mathscr{B} = \emptyset$$

Correctness

Theorem

The true concurrent semantics is correct iff:

$$\forall \gamma : \forall M \in \mathscr{R}^{tc}(\mathscr{N}^{\sigma\gamma}) : \sigma\gamma[M] \cap \mathscr{B} = \emptyset$$

$$\Leftrightarrow$$

$$\forall M \in \mathscr{R}^{seq}(\mathscr{N}^{\sigma}) : \sigma[M] \cap \mathscr{B} = \emptyset$$

$$\mathscr{R}^{seq}(\mathscr{N}^{\sigma}) = \bigcup_{\gamma \in \mathscr{N}^{\sigma}} (\mathscr{R}^{seq}(\mathscr{N}^{\sigma\gamma}))$$

Correctness

Theorem

The true concurrent semantics is correct iff:

$$\forall\gamma : \forall M \in \mathscr{R}^{tc}(\mathscr{N}^{\sigma\gamma}) : \sigma\gamma[M] \cap \mathscr{B} = \emptyset$$

$$\Leftrightarrow$$

$$\forall M \in \mathscr{R}^{seq}(\mathscr{N}^{\sigma}) : \sigma[M] \cap \mathscr{B} = \emptyset$$

$$\mathscr{R}^{seq}(\mathscr{N}^{\sigma}) = \bigcup_{\gamma \in \mathscr{N}^{\sigma}} (\mathscr{R}^{seq}(\mathscr{N}^{\sigma\gamma}))$$

$$\mathscr{R}^{seq}(\mathscr{N}^{\sigma}) \supseteq \bigcup_{\gamma \in \mathscr{N}^{\sigma}} (\mathscr{R}^{tc}(\mathscr{N}^{\sigma\gamma}))$$

Correctness

Theorem

The true concurrent semantics is correct iff:

$$\forall \gamma : \forall M \in \mathscr{R}^{tc}(\mathscr{N}^{\sigma\gamma}) : \sigma\gamma[M] \cap \mathscr{B} = \emptyset$$

$$\Leftrightarrow$$

$$\forall M \in \mathscr{R}^{seq}(\mathscr{N}^{\sigma}) : \sigma[M] \cap \mathscr{B} = \emptyset$$

$$\mathscr{R}^{seq}(\mathscr{N}^{\sigma}) = \bigcup_{\gamma \in \mathscr{N}^{\sigma}} (\mathscr{R}^{seq}(\mathscr{N}^{\sigma\gamma}))$$

$$\mathscr{R}^{seq}(\mathscr{N}^{\sigma}) \supseteq \bigcup_{\gamma \in \mathscr{N}^{\sigma}} (\mathscr{R}^{tc}(\mathscr{N}^{\sigma\gamma}))$$

$$\bigcup_{M \in \mathscr{R}^{seq}(\mathscr{N}^{\sigma})} \bigcup_{p \in M} p = \bigcup_{M \in \bigcup_{\gamma \in \mathscr{N}^{\sigma}} (\mathscr{R}^{tc}(\mathscr{N}^{\sigma\gamma}))} \bigcup_{p \in M} p$$

True Concurrency in Bounded Synthesis of Petri Games

True Concurrent Encoding

$$\exists \mathscr{S} : \forall \mathscr{E} : \forall \mathscr{M} : \phi'$$

True Concurrent Encoding

$$\exists \mathscr{S} : \forall \mathscr{E} : \forall \mathscr{M} : \phi'$$

Variables encoding choices of the *strategy*.

True Concurrent Encoding

$$\exists \mathscr{S} : \forall \mathscr{E} : \forall \mathscr{M} : \phi'$$

Variables encoding choices of the *strategy*.

Variables encoding all possible sequences of *markings*.

True Concurrent Encoding

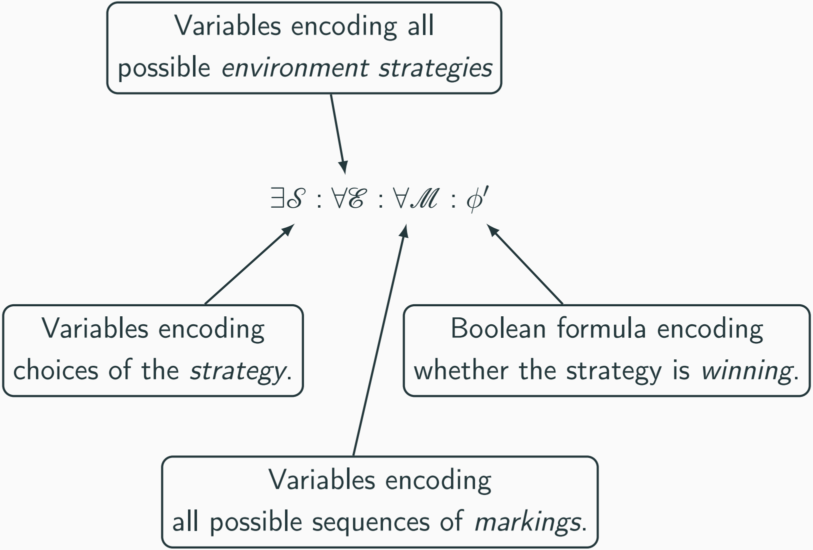
$$\exists \mathscr{S} : \forall \mathscr{E} : \forall \mathscr{M} : \phi'$$

Variables encoding choices of the *strategy*.

Variables encoding all possible sequences of *markings*.

Boolean formula encoding whether the strategy is *winning*.

True Concurrent Encoding



Encoding of the Game ϕ'

$$\phi' = \mathit{validEnvStrategy} \Rightarrow$$
$$(\mathit{validStrategy} \wedge \mathit{validSequence} \wedge \mathit{terminating} \wedge \mathit{winningStrategy})$$

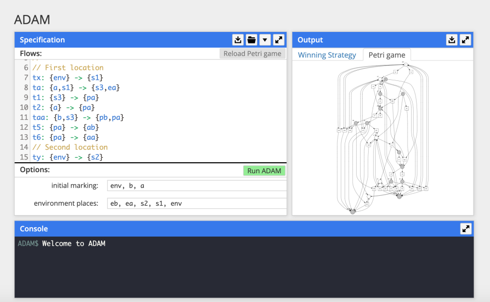
Encoding of the Game ϕ'

$$\phi' = \textit{validEnvStrategy} \Rightarrow (\textit{validStrategy} \wedge \textit{validSequence} \wedge \textit{terminating} \wedge \textit{winningStrategy})$$

validEnvStrategy:	filters invalid environment strategies
validSequence:	encodes true concurrent firing semantics
terminating:	encodes termination of SCCs

Bounded Synthesis Implementation ADAM[4]

- Implementation of Petri game decision procedures
- Online interface for bounded synthesis
- Try it online: https://react.uni-saarland.de/ADAM



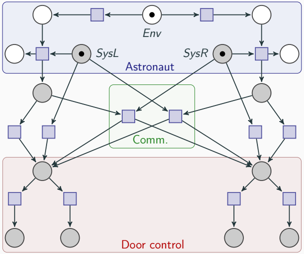
[4] Finkbeiner, Gieseking, and Olderog, "Adam: Causality-Based Synthesis of Distributed Systems".

Experimental Evaluation

		Sequential		*True Concurrent*	
Benchmark	*Parameter*	*Iteration*	*Runtime in seconds*	*Iteration*	*Runtime in seconds*
Alarm System	2	7	13.26	**6**	**11.15**
	3	-	timeout	-	timeout
Collision Avoidance	2	8	7.27	**5**	**6.25**
	3	-	timeout	**6**	**14.21**
	4	-	timeout	**7**	**346.23**
	5	-	timeout	-	timeout
Disjoint Routing	2	8	6.16	**7**	**6.05**
	3	11	11.03	**9**	**10.07**
	4	14	69.50	**11**	**65.31**
	5	-	timeout	-	timeout
Production Line	1	**4**	**5.59**	**4**	**5.59**
	2	5	6.08	**4**	**5.85**
	...	...	...	...	...
	5	8	87.33	**4**	**41.95**
	6	-	timeout	**4**	**742.36**
	7	-	timeout	-	timeout
Document Workflow	1	8	5.90	**7**	**5.79**
	2	10	6.58	**9**	**6.44**
	...	...	...	...	...
	10	26	**716.61**	**25**	823.94
	11	**28**	**1304.14**	-	timeout
	12	-	timeout	-	timeout

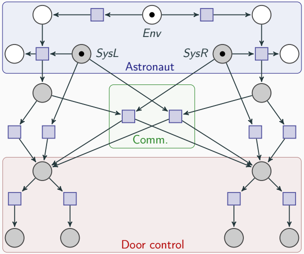
Summary

Airlock as Petri Game

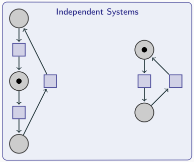


Summary

Airlock as Petri Game



True Concurrent Firing



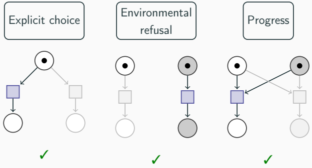
How to remain correctness?

Summary

Environment Strategy

Definition

An *environment strategy* γ is a subnet of a system strategy σ that satisfies the conditions *explicit choice*, *environmental refusal*, *and progess*.

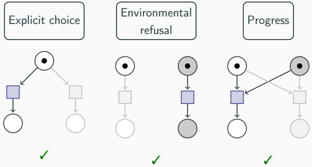


Summary

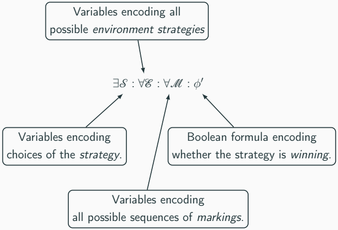
Environment Strategy

Definition

An *environment strategy* γ is a subnet of a system strategy σ that satisfies the conditions *explicit choice, environmental refusal, and progess*.



True Concurrent Encoding

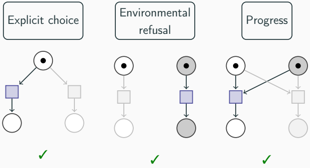


Summary

Environment Strategy

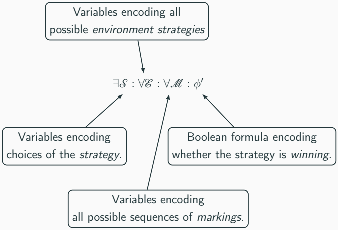
Definition

An *environment strategy* γ is a subnet of a system strategy σ that satisfies the conditions *explicit choice, environmental refusal, and progess.*



15

True Concurrent Encoding



21

It is beneficial to implement asynchronicity as true concurrency in distributed synthesis!

References

Beutner, Raven, Bernd Finkbeiner, and Jesko Hecking-Harbusch. "Translating Asynchronous Games for Distributed Synthesis". In: *Proceedings of CONCUR*. 2019, 22:1–22:16.

Finkbeiner, Bernd. "Bounded Synthesis for Petri Games". In: *Correct System Design*. 2015, pp. 223–237.

Finkbeiner, Bernd, Manuel Gieseking, and Ernst-Rüdiger Olderog. "Adam: Causality-Based Synthesis of Distributed Systems". In: *Proceedings of CAV*. 2015, pp. 433–439.

Finkbeiner, Bernd and Paul Gölz. "Synthesis in Distributed Environments". In: *Proceedings of FSTTCS*. 2017, 28:1–28:14.

Finkbeiner, Bernd and Ernst-Rüdiger Olderog. "Petri Games: Synthesis of Distributed Systems with Causal Memory". In: *Proceedings Fifth International Symposium on Games, Automata, Logics and Formal Verification, GandALF 2014, Verona, Italy, September 10-12, 2014*. 2014, pp. 217–230. DOI: 10.4204/EPTCS.161.19. URL: https://doi.org/10.4204/EPTCS.161.19.

Pnueli, A. and R. Rosner. "Distributed Reactive Systems Are Hard to Synthesize". In: *Proceedings of the 31st Annual Symposium on Foundations of Computer Science*. SFCS '90. Washington, DC, USA: IEEE Computer Society, 1990, 746–757 vol.2. ISBN: 0-8186-2082-X. DOI: 10.1109/FSCS.1990.89597. URL: https://doi.org/10.1109/FSCS.1990.89597.

Known Decidability Classes of Petri Games

- 1 environment player, bounded system players
  $\Rightarrow$ EXPTIME-complete[5]
- bounded environment players, 1 system player
  $\Rightarrow$ EXPTIME-complete[6]
- Acyclic communication
  $\Rightarrow$ Non-elementary[7]

---

[5] Finkbeiner and Olderog, "Petri Games: Synthesis of Distributed Systems with Causal Memory".

[6] Finkbeiner and Gölz, "Synthesis in Distributed Environments".

[7] Beutner, Finkbeiner, and Hecking-Harbusch, "Translating Asynchronous Games for Distributed Synthesis".