Efficient Trace Encodings of Bounded Synthesis for Asynchronous Distributed Systems

Jesko Hecking-Harbusch, Niklas O. Metzger
October 30, 2019

Saarland University - Reactive Systems Group
Distributed Systems are hard to synthesize. Petri games are used as a framework for distributed synthesis. Bounded synthesis is applied to Petri games. The asynchronous nature is encoded to interleaving.
Motivation - Distributed Synthesis

Distributed Systems are hard to synthesize.

Petri games as framework for distributed synthesis.

Bounded Synthesis for Petri games.

Asynchronous nature is encoded to interleaving.
Motivation - Distributed Synthesis

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

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

Diagram:
- System
- Environment
- System

Arrows indicate interaction and influence between systems and the environment.
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

Arena

System L → Environment → System R

System L → ? → System R

System L → ? → Environment

System L → Environment → System R
Overview

1. Petri Games

2. Bounded Synthesis

3. True Concurrency in Petri Games

4. True Concurrency in Bounded Synthesis of Petri Games
Petri Net as Game Arena of Petri Game
Places $\mathcal{P}$ in a Petri Net
Transitions $\mathcal{T}$ in a Petri Net
Tokens in a Petri Net
An enabled transition...
Two System Players

SysL

SysR
One Environment Player

Env

SysL

SysR
Airlock as Petri Game

Env

SysL

Astronaut

SysR

Comm.

Door control
Winning condition:

\[ L \implies LO \land RC \]

\[ R \implies LC \land RO \]
Winning Conditions of the Petri Game

Winning condition:

\[ L \implies LO \land RC \]

\[ R \implies LC \land RO \]

Astronaut

Comm.

Door control
Decision for left Door
Synchronization with the System

Env

SysL

SysR

Astronaut

Comm.

Door control
Exchange of Information

Env
SysL
SysR
Astronaut
Comm.
Door control
Memory Model of Petri Games: Causal Past

- Env
- SysL
- SysR
- Astronaut
- SyncL
- Comm.
- DoorL
- DoorR
- Door control
Memory Model of Petri Games: Causal Past
Memory Model of Petri Games: Causal Past

Diagram showing the memory model of Petri games with causal past. The diagram includes components such as Env, SysL, SysR, Astronaut, DoorL, DoorR, SyncL, and SyncR. Transitions are indicated by t1, t2, and t3.
Memory Model of Petri Games: Causal Past
Refuse transitions based on *Causal Past*

Winning condition:

\[ L \implies LO \land RC \]
\[ R \implies LC \land RO \]

Door control
Unfolding of Airlock

Door control
Winning Strategy of Airlock $\sigma$

Diagram showing the interaction between System Left (SysL), System Right (SysR), Environmental (Env), Astronaut (Astronaut), Communication (Comm), and Door Control (LO, LC, RC, RO).
### Outcome of the Petri Game

#### Reachable Markings

\[
R(N) = \{ M \subseteq P \mid \exists t_1, \ldots, t_n \in T : \exists M_1, \ldots, M_n \subseteq P : \\
In[t_1]M_1 \ldots [t_n]M_n = M \}
\]

#### Winning Safety Condition

A system strategy \( \sigma \) is *winning* for the condition *safety* \( (B) \) iff

\[
\forall M \in R(N^\sigma) : \sigma[M] \cap B = \emptyset.
\]
Outcome of the Petri Game

Reachable Markings

\[ \mathcal{R}(\mathcal{N}) = \{ M \subseteq \mathcal{P} | \exists t_1, \ldots, t_n \in \mathcal{T} : \exists M_1, \ldots, M_n \subseteq \mathcal{P} : \]
\[ \text{ln}[t_1]M_1 \cdots [t_n]M_n = M \} \]

Winning Safety Condition

A system strategy \( \sigma \) is \emph{winning} for the condition \emph{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

- Petri game
- Bounded unfolding
- Encoding of existence of a winning strategy
  - Bound $b$ copies per place
  - Increase bound

Winning strategy
Sequential Encoding

Quantified Boolean Formula (QBF):

$$\exists S : \forall M : \phi$$

Variables encoding choices of the strategy.
Sequential Encoding

Quantified Boolean Formula (QBF):

$$\exists S : \forall M : \phi$$

- Variables encoding choices of the *strategy*.
- Variables encoding all possible sequences of *markings*.

$$\phi = \text{validStrategy} \land \text{validSequence} \land \text{terminating} \land \text{winningStrategy}$$
Quantified Boolean Formula (QBF):

$$\exists S : \forall M : \phi$$

- Variables encoding choices of the strategy.
- Boolean formula encoding whether the strategy is winning.
- Variables encoding all possible sequences of markings.
Quantified Boolean Formula (QBF):

$$\exists S : \forall M : \phi$$

- Variables encoding choices of the strategy.
- Variables encoding all possible sequences of markings.
- Boolean formula encoding whether the strategy is winning.

$$\phi = \text{validStrategy} \land \text{validSequence} \land \text{terminating} \land \text{winningStrategy}$$
True Concurrency in Petri Games
Which Player progresses Next?

Env

SysL

SysR

Astronaut

Comm.

Door control
Left Door can be First

Env

SysL

Astronaut

SysR

Comm.

Door control
Right Door can be First
Both System Players

Env

SysL

SysR

Astronaut

Comm.

Door control
Sequential Firing

Many interleavings with same causal past!

Fire all enabled transitions

Independent Systems
Sequential Firing

Many interleavings with same causal past!

Fire all enabled transitions

Independent Systems
Sequential Firing

Many interleavings with same causal past!

Fire all enabled transitions

Independent Systems
Sequential Firing

Many interleavings with same causal past!

Fire all enabled transitions

Independent Systems
Sequential Firing

Many interleavings with same causal past!

Fire all enabled transitions

Independent Systems
Sequential Firing

Many interleavings with same causal past!

Independent Systems

Many interleavings with same causal past!
Sequential Firing

Many interleavings with same causal past!
Fire all enabled transitions
True Concurrent Firing

Independent Systems
True Concurrent Firing

Independent Systems
True Concurrent Firing

Independent Systems
True Concurrent Firing

Independent Systems
True Concurrent Firing

Independent Systems
How to remain correct?
Environment Strategies for Airlock

Astronaut

Comm.

Door control

Env

SysL

SysR

Astronaut

Comm.
Environment Strategies for Airlock

Door control

Env

SysL

SysR

Astronaut

Comm.

LC

RO
Environment Strategies for Airlock

![Diagram of airlock environment strategies]

- *Env*: Environment
- *SysL*: System Left
- *SysR*: System Right
- *Astronaut*: Astronaut
- *Comm.*: Communication
- *Door control*: Door control
- *LO*: Left Output
- *RC*: Right Control
## Environment Strategy

### Definition

An *environment strategy* $\gamma$ is a subnet of a system strategy $\sigma$ that satisfies the conditions *explicit choice*, *environmental refusal*, and *progress*. 
Definition

An *environment strategy* $\gamma$ is a subnet of a system strategy $\sigma$ that satisfies the conditions *explicit choice*, *environmental refusal*, and *progress*.
Environment Strategy

**Definition**

An *environment strategy* $\gamma$ is a subnet of a system strategy $\sigma$ that satisfies the conditions *explicit choice, environmental refusal, and progress*.
Environment Strategy

**Definition**

An *environment strategy* $\gamma$ is a subnet of a system strategy $\sigma$ that satisfies the conditions *explicit choice, environmental refusal, and progress*.

![Diagram](image-url)
Environment Strategy

**Definition**

An *environment strategy* $\gamma$ is a subnet of a system strategy $\sigma$ that satisfies the conditions *explicit choice*, *environmental refusal*, and *progress*.

Explicit choice

Environmental refusal

Progress
Environment Strategy

**Definition**

An environment strategy $\gamma$ is a subnet of a system strategy $\sigma$ that satisfies the conditions *explicit choice*, *environmental refusal*, and *progress*.

- **Explicit choice**
- **Environmental refusal**
- **Progress**
An *environment strategy* $\gamma$ is a subnet of a system strategy $\sigma$ that satisfies the conditions *explicit choice*, *environmental refusal*, and *progress*.
Theorem

An environment strategy $\gamma$ leads to a *unique sequence* of fired transitions up to reordering of independent transitions.
Theorem

An environment strategy $\gamma$ leads to a *unique sequence* of fired transitions up to reordering of independent transitions.
Theorem

An environment strategy $\gamma$ leads to a *unique sequence* of fired transitions up to reordering of independent transitions.
An environment strategy $\gamma$ leads to a *unique sequence* of fired transitions up to reordering of independent transitions.
Theorem

An environment strategy $\gamma$ leads to a unique sequence of fired transitions up to reordering of independent transitions.
Reachable Markings

\[ R^{seq}(\mathcal{N}) = \{ M \subseteq \mathcal{P} \mid \exists t_1, \ldots, t_n \in \mathcal{T} : \exists M_1, \ldots, M_n \subseteq \mathcal{P} : \text{ln}[t_1] M_1 \cdots [t_n] M_n = M \} \]

\[ R^{tc}(\mathcal{N}) = \{ M \subseteq \mathcal{P} \mid \exists T_1, \ldots, T_n \subseteq \mathcal{T} : \exists M_1, \ldots, M_n \subseteq \mathcal{P} : \text{ln}[T_1] M_1 \cdots [T_n] M_n = M \} \]
A system strategy $\sigma$ is winning for the condition safety ($\mathbb{B}$) iff

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

**Theorem**

The true concurrent semantics is correct iff:

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

\[ \iff \]

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

Theorem

The true concurrent semantics is correct iff:

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

\[ \iff \]

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

\[ R^{seq}(N^{\sigma}) = \bigcup_{\gamma \in N^{\sigma}} (R^{seq}(N^{\sigma\gamma})) \]
The true concurrent semantics is correct iff:

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

\[ \iff \]

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

\[ R^{seq}(N^{\sigma}) = \bigcup_{\gamma \in N^{\sigma}} (R^{seq}(N^{\sigma \gamma})) \]

\[ R^{seq}(N^{\sigma}) \supseteq \bigcup_{\gamma \in N^{\sigma}} (R^{tc}(N^{\sigma \gamma})) \]
Theorem

The true concurrent semantics is correct iff:

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

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

\[
R^{\text{seq}}(N^{\sigma}) = \bigcup_{\gamma \in N^{\sigma}} (R^{\text{seq}}(N^{\sigma \gamma}))
\]

\[
R^{\text{seq}}(N^{\sigma}) \supseteq \bigcup_{\gamma \in N^{\sigma}} (R^{\text{tc}}(N^{\sigma \gamma}))
\]

\[
\bigcup_{M \in R^{\text{seq}}(N^{\sigma})} \bigcup_{p \in M} p = \bigcup_{M \in \bigcup_{\gamma \in N^{\sigma}} (R^{\text{tc}}(N^{\sigma \gamma}))} \bigcup_{p \in M} p
\]
True Concurrency in Bounded Synthesis of Petri Games
True Concurrent Encoding

\[ \exists S : \forall E : \forall M : \phi' \]

Variables encoding choices of the strategy.
Variables encoding all possible sequences of markings.
Boolean formula encoding whether the strategy is winning.
Variables encoding all possible environment strategies.
Variables encoding choices of the strategy.
True Concurrent Encoding

$\exists S : \forall E : \forall M : \phi'$

Variables encoding choices of the strategy.

Variables encoding all possible sequences of markings.
True Concurrent Encoding

\[ \exists S : \forall \exists : \forall M : \phi' \]

- Variables encoding choices of the *strategy*.
- Boolean formula encoding whether the strategy is *winning*.
- Variables encoding all possible sequences of *markings*.
True Concurrent Encoding

Variables encoding all possible environment strategies

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

- Variables encoding choices of the strategy.
- Boolean formula encoding whether the strategy is winning.
- Variables encoding all possible sequences of markings.
Encoding of the Game $\phi'$

$\phi' = validEnvStrategy \Rightarrow (validStrategy \land validSequence \land terminating \land winningStrategy)$
Encoding of the Game $\phi'$

\[ \phi' = validEnvStrategy \Rightarrow (validStrategy \land validSequence \land terminating \land winningStrategy) \]

- **validEnvStrategy**: filters invalid environment strategies
- **validSequence**: encodes true concurrent firing semantics
- **terminating**: encodes termination of SCCs
Bounded Synthesis Implementation ADAM

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

---

4Finkbeiner, Gieseking, and Olderog, “Adam: Causality-Based Synthesis of Distributed Systems”.
## Experimental Evaluation

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Parameter</th>
<th>Iteration</th>
<th>Sequential Runtime in seconds</th>
<th>True Concurrent Runtime in seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alarm System</td>
<td>2</td>
<td>7</td>
<td>13.26</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>-</td>
<td>timeout</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>-</td>
<td>timeout</td>
<td>-</td>
</tr>
<tr>
<td>Collision Avoidance</td>
<td>2</td>
<td>8</td>
<td>7.27</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>-</td>
<td>timeout</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>-</td>
<td>timeout</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>-</td>
<td>timeout</td>
<td>-</td>
</tr>
<tr>
<td>Disjoint Routing</td>
<td>2</td>
<td>8</td>
<td>6.16</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>11</td>
<td>11.03</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>14</td>
<td>69.50</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>-</td>
<td>timeout</td>
<td>-</td>
</tr>
<tr>
<td>Production Line</td>
<td>1</td>
<td>4</td>
<td>5.59</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>5</td>
<td>6.08</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>8</td>
<td>87.33</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>-</td>
<td>timeout</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>-</td>
<td>timeout</td>
<td>-</td>
</tr>
<tr>
<td>Document Workflow</td>
<td>1</td>
<td>8</td>
<td>5.90</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>10</td>
<td>6.58</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>26</td>
<td>716.61</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>28</td>
<td>1304.14</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>-</td>
<td>timeout</td>
<td>-</td>
</tr>
</tbody>
</table>
Summary

Airlock as Petri Game

[Diagram of a Petri net with labeled places and transitions, including labels for Env, SysL, SysR, Astronaut, Comm., and Door control]
Summary

Airlock as Petri Game

True Concurrent Firing

Independent Systems

How to remain correctness?
Environment Strategy

**Definition**
An *environment strategy* $\gamma$ is a subnet of a system strategy $\sigma$ that satisfies the conditions *explicit choice*, *environmental refusal*, and *progress*.

- **Explicit choice**
- **Environmental refusal**
- **Progress**

It is beneficial to implement asynchronicity as true concurrency in distributed synthesis!
An environment strategy \( \gamma \) is a subnet of a system strategy \( \sigma \) that satisfies the conditions explicit choice, environmental refusal, and progress.

Variables encoding all possible environment strategies.

Boolean formula encoding whether the strategy is winning.

Variables encoding all possible sequences of markings.

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


Known Decidability Classes of Petri Games

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

---

5 Finkbeiner and Olderog, “Petri Games: Synthesis of Distributed Systems with Causal Memory”.

6 Finkbeiner and Götz, “Synthesis in Distributed Environments”.

7 Beutner, Finkbeiner, and Hecking-Harbusch, “Translating Asynchronous Games for Distributed Synthesis”.