Switched Probabilistic I/O Automata

Ling Cheung\textsuperscript{1} \quad Nancy Lynch\textsuperscript{2} \quad Roberto Segala\textsuperscript{3} \\
Frits Vaandrager\textsuperscript{1}

\textsuperscript{1}Nijmegen Institute for Computing and Information Sciences \\
University of Nijmegen, the Netherlands

\textsuperscript{2}MIT Computer Science and Artificial Intelligence Laboratory, U.S.A.

\textsuperscript{3}Dipartimento di Informatica, Università di Verona, Italy

ICTAC 2004, Guiyang, China
Outline

1 Introduction
   - Basics
   - Randomization
Outline

1. Introduction
   - Basics
   - Randomization

2. The trouble with composition
   - What is parallel composition?
   - How much does the daemon know?
   - Global choice vs local choice
Outline

1. Introduction
   - Basics
   - Randomization

2. The trouble with composition
   - What is parallel composition?
   - How much does the daemon know?
   - Global choice vs local choice

3. Switched PIOA
   - The Switched PIOA model
   - Implementing parallel compositions
Outline

1. **Introduction**
   - Basics
   - Randomization

2. **The trouble with composition**
   - What is parallel composition?
   - How much does the daemon know?
   - Global choice vs local choice

3. **Switched PIOA**
   - The Switched PIOA model
   - Implementing parallel compositions

4. **Summary and future work**
   - Summary
   - Future work
To NIII Colloquium Attendees:

Thank you all for coming to my talk!
For this talk . . .

- We need very little probability theory: *discrete distributions.* Examples:
  - fair coin: \{⟨Head, \frac{1}{2}⟩, ⟨Tail, \frac{1}{2}⟩\};
  - fair dice: \{⟨i, \frac{1}{6}⟩ | 1 ≤ i ≤ 6⟩.\}
We need very little probability theory: *discrete distributions*. Examples:
- fair coin: \( \{ \langle \text{Head}, \frac{1}{2} \rangle, \langle \text{Tail}, \frac{1}{2} \rangle \} \);
- fair dice: \( \{ \langle i, \frac{1}{6} \rangle \mid 1 \leq i \leq 6 \} \).

Underlying model: nondeterministic automata with asynchronous composition.
(In our paper: input/output distinction, combination of synchronous and asynchronous compositions, etc.)
We need very little probability theory: *discrete distributions*. Examples:

- fair coin: \(\{\langle \text{Head}, \frac{1}{2}\rangle, \langle \text{Tail}, \frac{1}{2}\rangle\}\);
- fair dice: \(\{\langle i, \frac{1}{6}\rangle \mid 1 \leq i \leq 6\}\).

Underlying model: nondeterministic automata with asynchronous composition.
(In our paper: input/output distinction, combination of synchronous and asynchronous compositions, etc.)

Total order semantics: if both actions \(a\) and \(b\) occur, one must precede the other.
Schedulers and trace distributions

- History-dependent, randomized schedulers transform nondeterministic choices into probabilistic choices.

\[ \{ \langle a, p \rangle, \langle ab, p(1-q) \rangle, \langle b, 1-p \rangle \} \]

Cheung, Lynch, Segala, Vaandrager
Schedulers and trace distributions

- *History-dependent, randomized* schedulers transform nondeterministic choices into probabilistic choices.

\[ \begin{align*}
  a &\xrightarrow{p} b \\
  a &\xrightarrow{1-p} b \\
  a &\xrightarrow{q} 1-q \\
  a &\xrightarrow{1-q} 1-q
\end{align*} \]
Schedulers and trace distributions

- *History-dependent, randomized* schedulers transform nondeterministic choices into probabilistic choices.

- Each scheduler induces a *trace distribution*: a discrete distribution on finite traces.

\[
\{(aa, pq), (ab, p(1-q)), (b, 1-p)\}
\]
Nondeterministic parallel composition

\[ P \xrightarrow{a} \quad Q \xrightarrow{b} \]

The interleaving axiom:

\[ P \parallel Q \]

Cheung, Lynch, Segala, Vaandrager

Switched Probabilistic I/O Automata
Nondeterministic parallel composition

The *interleaving* axiom:
Probabilistic parallel composition

\[ P \]
\[ \downarrow \]
\[ \uparrow \rightarrow \]
\[ \rightarrow \rightarrow \]
\[ a \]

\[ Q \]
\[ \downarrow \]
\[ \uparrow \rightarrow \]
\[ \rightarrow \rightarrow \]
\[ b \]
Probabilistic parallel composition

What is a probabilistic behavior of $P\parallel Q$?

Quick answer: bias factor $\theta$. Imagine a coin-flipping daemon.
What is a \textit{probabilistic} behavior of $P \parallel Q$? Quick answer: \textit{bias factor $\theta$}.

\begin{center}
\begin{tikzpicture}
\node (P) at (0,0) {$P$};
\node (Q) at (2,0) {$Q$};
\draw[->] (P.north) -- +(0,0.5) node[fill=white] {$a$};
\draw[->] (Q.south) -- +(0,-0.5) node[fill=white] {$b$};
\end{tikzpicture}
\end{center}
Probabilistic parallel composition

What is a probabilistic behavior of $P \parallel Q$?
Quick answer: bias factor $\theta$.
Imagine a coin-flipping daemon.
What is the value of $\theta$?

\[ P \parallel Q \]

- $a$ with probability $\theta$
- $b$ with probability $1 - \theta$

Fixed $\theta$: parameterized composition operator $\parallel \theta$.

Limitations: static parameter, not commutative, not associative.

Variable $\theta$: a supply of coins with different biases; imaginary daemon chooses a coin based on his knowledge.
What is the value of $\theta$?

Fixed $\theta$: *parameterized* composition operator $\parallel^\theta$.

\[ \begin{array}{l}
  P \parallel Q \\
  \theta \quad 1-\theta \\
  a \quad b \\
  \end{array} \]
What is the value of $\theta$?

Fixed $\theta$: parameterized composition operator $P \parallel \theta$. Limitations: static parameter, not commutative, not associative.
What is the value of $\theta$?

Fixed $\theta$: *parameterized* composition operator $P \parallel Q^\theta$.
Limitations: static parameter, not commutative, not associative.

Variable $\theta$:
- a supply of coins with different biases;
- imaginary daemon chooses a coin based on his knowledge.
How much does the daemon know?

There are two scenarios:
How much does the daemon know?

There are two scenarios:

Scenario 1: *context-independent*
How much does the daemon know?

There are two scenarios:

Scenario 1: *context-independent*

Scenario 2: *context-dependent*
Scenario 1: context-independent composition
How much does the daemon know?

Daemon, $P$ and $Q$ all inside a big black box.
Scenario 1: context-independent composition
How much does the daemon know?

Daemon, $P$ and $Q$ all inside a big black box.

Daemon knows the histories of $P$ and $Q$, but nothing about the outside world.
Scenario 1: context-independent composition
How much does the daemon know?

Daemon, $P$ and $Q$ all inside a big black box.

Daemon knows the histories of $P$ and $Q$, but nothing about the outside world.

Problem: non-associativity.
Non-associativity: $P\parallel(Q\parallel R)$

Context-independent composition

Inner daemon: $\langle R, 1 \rangle$. 

\[
\begin{align*}
P & \quad a \downarrow \\
Q & \quad b \downarrow \\
R_c & \quad \begin{array}{c}
p \rightarrow \\
1-p \rightarrow \\
d \rightarrow \\
p \leftarrow \\
1-p \leftarrow \\
c \rightarrow \\
\end{array}
\end{align*}
\]
Non-associativity: $P \parallel (Q \parallel R)$

Context-independent composition

Inner daemon: $\langle R, 1 \rangle$.

Outer daemon: $\langle Q \parallel R, 1 \rangle$; if $c$, then $\langle P, 1 \rangle$, else $\langle Q \parallel R, 1 \rangle$. 
Non-associativity: $P \parallel (Q \parallel R)$

Context-independent composition

Inner daemon: $\langle R, 1 \rangle$.

Outer daemon: $\langle Q \parallel R, 1 \rangle$; if $c$, then $\langle P, 1 \rangle$, else $\langle Q \parallel R, 1 \rangle$.

Result: $\{\langle cab, p \rangle, \langle dba, 1 - p \rangle\}$. 
Non-associativity: \((P \parallel Q) \parallel R\)

Context-independent composition

Claim: \{\langle cab, p \rangle, \langle dba, 1 - p \rangle \} not possible!
Non-associativity: \((P \parallel Q) \parallel R\)

Context-independent composition

Claim: \(\{\langle cab, p \rangle, \langle dba, 1 - p \rangle\}\) not possible!

- **Outer** daemon: \(\langle R, 1 \rangle\).
Non-associativity: \((P \parallel Q) \parallel R\)

Context-independent composition

Claim: \{\(\langle cab, p \rangle, \langle dba, 1 - p \rangle\}\} not possible!

- **Outer** daemon: \(\langle R, 1 \rangle\).
- **Inner** daemon:
  \{\(\langle P, q \rangle, \langle Q, 1 - q \rangle\}\}.
Non-associativity: \((P \parallel Q) \parallel R\)

Context-independent composition

Claim: \(\{\langle cab, p \rangle, \langle dba, 1 - p \rangle\}\) not possible!

- **Outer daemon:** \(\langle R, 1 \rangle\).
- **Inner daemon:** \(\{\langle P, q \rangle, \langle Q, 1 - q \rangle\}\).

Cheung, Lynch, Segala, Vaandrager

Switched Probabilistic I/O Automata
Non-associativity: $(P \parallel Q) \parallel R$

Context-independent composition

Claim: $\{⟨cab, p⟩, ⟨dba, 1 − p⟩\}$ not possible!

- **Outer daemon:** $⟨R, 1⟩$.
- **Inner daemon:** $\{⟨P, q⟩, ⟨Q, 1 − q⟩\}$.
- **Conclusion:** inner daemon doesn't know enough.
Scenario 2: context-dependent composition
How much does the daemon know?

Daemon sees the outside world.
Scenario 2: context-dependent composition
How much does the daemon know?

Daemon sees the outside world.

Daemon knows the histories of $P$, $Q$ and $Env$. 
Scenario 2: context-dependent composition
How much does the daemon know?

Daemon sees the outside world.

Daemon knows the histories of $P$, $Q$ and $Env$.

**Problem**: violation of the interleaving axiom!
I.e., there exists $Env$ such that

$$(a \parallel b) \parallel Env \not\sim (a.b + b.a) \parallel Env.$$
Non-interleaving semantics
Context-dependent composition

\[(a \parallel b) \parallel \text{Env:}\]

\[
\begin{align*}
P & \quad a \\
Q & \quad b \\
\text{Env} & \quad c \quad d \\
 & \quad p \quad 1-p
\end{align*}
\]
Non-interleaving semantics
Context-dependent composition

\[(a\parallel b)\parallel \text{Env:}\]

\[(a.b + b.a)\parallel \text{Env:}\]
The (same) counterexample

Context-dependent composition

The trace distribution

$$\{\langle cab, p \rangle, \langle dba, 1 - p \rangle\}$$

is possible in \((a \parallel b) \parallel \text{Env}\), but not in \((a.b + b.a) \parallel \text{Env}\).
The (same) counterexample

Context-dependent composition

The trace distribution

\[ \{\langle cab, p\rangle, \langle dba, 1-p\rangle\} \]

is possible in \((a \parallel b) \parallel \text{Env}\), but not in \((a.b + b.a) \parallel \text{Env}\).

Conclusion: we have a non-interleaving, but total order semantics.
What's wrong?

Something is wrong with our understanding of parallel composition.
What’s wrong?

Something is wrong with our understanding of parallel composition.

In context-independent composition, the problem shows up as non-associativity.
What’s wrong?

Something is wrong with our understanding of parallel composition.

In context-independent composition, the problem shows up as non-associativity.

In context-dependent composition, the same problem leads to difference between $a \parallel b$ and $a.b + b.a$. 
Two types of nondeterministic choices: global vs. local

**Global** choice: $a \parallel b$, resolved by a daemon.

![Diagram of global choice](image)

Behavior varies depending on the perspective!
Two types of nondeterministic choices: global vs. local

**Global choice:** $a \parallel b$, resolved by a daemon.

**Local choice:** $a \cdot b + b \cdot a$, resolved by a local scheduler.

Behavior varies depending on the perspective!
Dissecting the problem, Part I: eliminate global choices.

To better understand the problem, we developed the model of *Switched PIOA*.

\[ P \xrightarrow{a} . \]

\[ Q \xrightarrow{b} . \]
Dissecting the problem, Part I: eliminate global choices.

To better understand the problem, we developed the model of Switched PIOA.

- active states (foreground) vs. inactive states (background);
Dissecting the problem, Part I: eliminate global choices.

To better understand the problem, we developed the model of \textit{Switched PIOA}.

- active states (foreground) vs. inactive states (background);
- control exchange via special actions (e.g. \textit{go}_P, \textit{go}_Q);
Dissecting the problem, Part I: eliminate global choices.

To better understand the problem, we developed the model of *Switched PIOA*.

- **active** states (foreground) vs. **inactive** states (background);
- **control exchange** via special actions (e.g. go\(_P\), go\(_Q\));

Every decision is made locally, so no more daemons.
Due to the absence of global choices . . .

Parallel composition in Switched PIOA:
- easy to define;
Due to the absence of global choices . . .

Parallel composition in Switched PIOA:

- easy to define;
- commutative and associative;
Due to the absence of global choices . . .

Parallel composition in Switched PIOA:

- easy to define;
- commutative and associative;
- deep/semantic compositionality of trace distribution semantics;
Due to the absence of global choices . . .

Parallel composition in Switched PIOA:
- easy to define;
- commutative and associative;
- deep/semantic compositionality of trace distribution semantics;

That’s all very nice, but parallel processes don’t really exchange control . . .
Dissecting the problem, Part II: reintroduce global choices.

- Control-exchange should not be taken semantically.
Dissecting the problem, Part II: reintroduce global choices.

- Control-exchange should **not** be taken semantically.
- Switch PIOA is an **implementation tool** for various composition operators.
Dissecting the problem, Part II: reintroduce global choices.

- Control-exchange should not be taken semantically.
- Switch PIOA is an implementation tool for various composition operators.

Examples:
- fixed bias factor $\theta$;
Dissecting the problem, Part II: reintroduce global choices.

- Control-exchange should not be taken semantically.
- Switch PIOA is an implementation tool for various composition operators.

Examples:
  - fixed bias factor $\theta$;
  - context-independent.
Implementing biased composition

Local schedulers: always return control after one local move.

Arbiter: usually schedule $\langle \text{go } P, \theta \rangle$, $\langle \text{go } Q, 1-\theta \rangle$; if final $P$ then $\langle \text{go } Q, 1 \rangle$ and vice versa.

Examples:
$\langle \text{go } P, a, \text{final } P, \text{go } Q, b, \text{final } Q, \theta \rangle$;
$\langle \text{go } Q, b, \text{final } Q, \text{go } P, a, \text{final } P, 1-\theta \rangle$. 

Cheung, Lynch, Segala, Vaandrager
Implementing **biased** composition

- Local schedulers: always return control after one local move.
Implementing *biased* composition

- Local schedulers: always return control after one local move.
- Arbiter:
  - usually schedule $\{\langle \text{go}_P, \theta \rangle, \langle \text{go}_Q, 1 - \theta \rangle \}$;
Implementing biased composition

- Local schedulers: always return control after one local move.
- Arbiter:
  - usually schedule \( \{ \langle \text{go}_P, \theta \rangle, \langle \text{go}_Q, 1 - \theta \rangle \} \);
  - if \( \text{final}_P \) then \( \langle \text{go}_Q, 1 \rangle \) and vice versa.
Implementing biased composition

- Local schedulers: always return control after one local move.
- Arbiter:
  - usually schedule $\{\langle \text{go}_P, \theta \rangle, \langle \text{go}_Q, 1 - \theta \rangle\}$;
  - if $\text{final}_P$ then $\langle \text{go}_Q, 1 \rangle$ and vice versa.

Examples:
- $\langle \text{go}_P . a . \text{final}_P . \text{go}_Q . b . \text{final}_Q, \theta \rangle$;
Implementing biased composition

- Local schedulers: always return control after one local move.
- Arbiter:
  - usually schedule $\{\langle \text{go}_P, \theta \rangle, \langle \text{go}_Q, 1 - \theta \rangle\}$;
  - if $\text{final}_P$ then $\langle \text{go}_Q, 1 \rangle$ and vice versa.

Examples:
- $\langle \text{go}_P \cdot a \cdot \text{final}_P \cdot \text{go}_Q \cdot b \cdot \text{final}_Q, \theta \rangle$;
- $\langle \text{go}_Q \cdot b \cdot \text{final}_Q \cdot \text{go}_P \cdot a \cdot \text{final}_P, 1 - \theta \rangle$. 
Implementing context-independent composition

\[
P \xrightarrow{\text{go}_P} \text{Arb} \xleftarrow{\text{done}_Q} \xrightarrow{\text{final}_Q} Q
\]

\[
P \xrightarrow{\text{done}_P} \text{Arb} \xleftarrow{\text{final}_P} \xrightarrow{\text{go}_Q} Q
\]

Local schedulers: no scheduling restrictions ("run to completion").

Arbiter: if final \( P \) then \( \langle \text{go}_Q, 1 \rangle \) and vice versa.
Implementing context-independent composition

- Local schedulers: no scheduling restrictions ("run to completion").
Implementing context-independent composition

- Local schedulers: no scheduling restrictions ("run to completion").
- Arbiter: if final_P then \( \langle \text{go}_Q, 1 \rangle \) and vice versa.
To summarize . . .

- Parallel composition is trickier than we thought.
To summarize . . .

- Parallel composition is trickier than we thought.
- Switched PIOA is a probabilistic model without global choices.
To summarize . . .

- Parallel composition is trickier than we thought.
- Switched PIOA is a probabilistic model without global choices.
- Parallel composition in Switched PIOA is well-behaved.
To summarize . . .

- Parallel composition is trickier than we thought.
- Switched PIOA is a probabilistic model without global choices.
- Parallel composition in Switched PIOA is well-behaved.
- Switched PIOA can be used to study various “real” parallel composition operators.
Future work

- Philosophical: is there a “most intuitive” parallel composition operator?
Future work

- Philosophical: is there a “most intuitive” parallel composition operator?
- Technical: “decomposing” trace distribution semantics.

```
Trace Distribution → Trace Set

‖ ‖ ‖ ‖
↓ ‖ ‖ ‖
Trace Likelihood → Trace
```

Philosophical: is there a “most intuitive” parallel composition operator?

Technical: “decomposing” trace distribution semantics.
Future work

- Philosophical: is there a “most intuitive” parallel composition operator?
- Technical: “decomposing” trace distribution semantics.

```
Trace Distribution → Trace Set
          ↓   ↓
          ↓   ↓
Trace Likelihood → Trace
```

- Practical: modeling communication and/or security protocols in Switched PIOA.
Future work

- Philosophical: is there a “most intuitive” parallel composition operator?
- Technical: “decomposing” trace distribution semantics.

Trace Distribution $\rightarrow$ Trace Set

\[ \begin{array}{c}
\text{Trace Likelihood} \\
\downarrow \\
\downarrow \\
\downarrow \\
\downarrow \\
\text{Trace}
\end{array} \]

- Practical: modeling communication and/or security protocols in Switched PIOA.

– End –
Appendix

- Trace Set Semantics
- Trace Likelihood Semantics
- Getting Stuck
Trace Set Semantics

Loosely speaking, a trace distribution is a discrete probability distribution over the set of finite traces.
Trace Set Semantics

Loosely speaking, a trace distribution is a discrete probability distribution over the set of finite traces.

To go from trace distribution to trace set, we forget probabilities by:

\[ \text{DiscDistr(Traces)} \xrightarrow{\text{support}} \text{Powerset(Traces)} \]
Trace Set Semantics

Loosely speaking, a trace distribution is a discrete probability distribution over the set of finite traces.

To go from trace distribution to trace set, we forget probabilities by:

\[
\text{DiscDistr(Traces)} \xrightarrow{\text{support}} \text{Powerset(Traces)}
\]

That is, schedulers return sets of possible transitions, rather than discrete distributions over possible transitions.
Trace Set Semantics: Example

Trace Set Semantics: Example

Add input $e,f$-loops.
Trace Set Semantics: Example

Add input $e,f$-loops.

Equivalent in semantics: trace.
Not equivalent in semantics: trace set, trace distribution, bisimulation.
Trace Set Semantics: Example

Trace set \(\{aec, afd\}\) not possible in Early'.

Consider the case in which: Early' chooses the left-hand branch; and environment performs \(f\).

At this point, Early' does not have the option to perform \(d\).

Important: Early' cannot choose between inputs \(e\) and \(f\).
Trace Set Semantics: Example

Trace set \( \{aec, afd\} \) not possible in Early'.
Consider the case in which:

- Early' chooses the left-hand branch; and
Trace Set Semantics: Example

Trace set \{aec, afd\} not possible in Early'.

Consider the case in which:
- Early' chooses the left-hand branch; and
- environment performs \( f \).

Cheung, Lynch, Segala, Vaandrager
Switched Probabilistic I/O Automata
Trace Set Semantics: Example

Trace set \{aec, afd\} not possible in Early'.
Consider the case in which:
- Early' chooses the left-hand branch; and
- environment performs \( f \).
At this point, Early' does not have the option to perform \( d \).
Trace Set Semantics: Example

Trace set \{aec,afd\} not possible in Early'.

Consider the case in which:
- Early' chooses the left-hand branch; and
- environment performs f.

At this point, Early' does not have the option to perform d.

Important: Early' cannot choose between inputs e and f.
What’s the lesson here?
What’s the lesson here?

It’s not about the numbers . . .
What’s the lesson here?

It’s not about the numbers . . .

Examples of ”undesirable” properties of trace distribution semantics can be reproduced in trace set semantics.
What’s the lesson here?

It’s not about the numbers . . .

Examples of ”undesirable” properties of trace distribution semantics can be reproduced in trace set semantics.

Key: each trace distribution contains a collection of traces, rather than a single trace.

In some cases, this allows us to observe branching structure.
An Alternative: Trace Likelihood Semantics

Each behavior is represented by a pair \( \langle \alpha, p \rangle \).
An Alternative: Trace Likelihood Semantics

Each behavior is represented by a pair $\langle \alpha, p \rangle$.

Intended meaning: under some scenario, trace $\alpha$ occurs with probability $p$. 
An Alternative: Trace Likelihood Semantics

Each behavior is represented by a pair \( \langle \alpha, p \rangle \).

Intended meaning: under some scenario, trace \( \alpha \) occurs with probability \( p \).

Difficulty: what is a possible scenario?
An Alternative: Trace Likelihood Semantics

Each behavior is represented by a pair $\langle \alpha, p \rangle$.

Intended meaning: under some scenario, trace $\alpha$ occurs with probability $p$.

Difficulty: what is a possible scenario?

Frequentist probabilities: prediction about a large number of experiments, not about a single experiment.
Getting Stuck . . .

my current strategy:

- stop thinking, start reading;
Getting Stuck . . .

my current strategy:

- stop thinking, start reading;
- do something concrete: modeling oblivious transfer.