On Decidability of Concurrent Kleene Algebra

CONCUR in Berlin – September 5-8, 2017

Paul Brunet, Damien Pous, and Georg Struth

University College London
ENS de Lyon – CNRS
University of Sheffield

Concurrent Kleene Algebra

$$e, f ::= 0 \mid 1 \mid a \mid e + f \mid e \cdot f \mid e^\star$$

- **Kleene algebra**: rational expressions.
  - Can be used to reason about **sequential** programs.
  - Canonical model: regular languages, *i.e.* sets of words.

Concurrent Kleene Algebra

$$e, f ::= 0 \mid 1 \mid a \mid e + f \mid e \cdot f \mid e^\star \mid e \parallel f$$

- **Kleene algebra**: rational expressions.
  - Can be used to reason about **sequential** programs.
  - Canonical model: regular languages, *i.e.* sets of words.

- **bi-Kleene algebra**: series-rational expressions.
  - Can be used to reason about **concurrent** programs.
  - Canonical model: pomset languages *i.e.* sets of **partially ordered** words.

Laurence & Struth, **Completeness theorems for bi-Kleene algebras and series-parallel rational pomset languages**, *2014*

Concurrent Kleene Algebra

$$e, f ::= 0 \mid 1 \mid a \mid e + f \mid e \cdot f \mid e^\star \mid e \parallel f$$

- **Kleene algebra**: rational expressions.
  - Can be used to reason about **sequential** programs.
  - Canonical model: regular languages, *i.e.* sets of words.
- **bi-Kleene algebra**: series-rational expressions.
  - Can be used to reason about **concurrent** programs.
  - Canonical model: pomset languages *i.e.* sets of **partially ordered** words.

  Laurence & Struth, **Completeness theorems for bi-Kleene algebras and series-parallel rational pomset languages**, *2014*
- **Concurrent Kleene algebra**: series-rational expressions.
  - Can be used to reason about **concurrent** programs with a **refinement** order.
  - "Canonical" model: **downwards-closed** pomset languages.

  Hoare, Möller, Struth & Wehrman, **Concurrent Kleene algebra and its foundations**, *2011*

Concurrent Kleene Algebra

$$e, f ::= 0 \mid 1 \mid a \mid e + f \mid e \cdot f \mid e^\star \mid e \parallel f$$

- **Kleene algebra**: rational expressions.
  - Can be used to reason about **sequential** programs.
  - Canonical model: regular languages, *i.e.* sets of words.
  - Decision procedure: equivalence of **finite state automata**.
- **bi-Kleene algebra**: series-rational expressions.
  - Can be used to reason about **concurrent** programs.
  - Canonical model: pomset languages *i.e.* sets of **partially ordered** words.

  Laurence & Struth, **Completeness theorems for bi-Kleene algebras and series-parallel rational pomset languages**, *2014*
- **Concurrent Kleene algebra**: series-rational expressions.
  - Can be used to reason about **concurrent** programs with a **refinement** order.
  - "Canonical" model: **downwards-closed** pomset languages.

  Hoare, Möller, Struth & Wehrman, **Concurrent Kleene algebra and its foundations**, *2011*

Concurrent Kleene Algebra

$$e, f ::= 0 \mid 1 \mid a \mid e + f \mid e \cdot f \mid e^\star \mid e \parallel f$$

- **Kleene algebra**: rational expressions.
  - Can be used to reason about **sequential** programs.
  - Canonical model: regular languages, *i.e.* sets of words.
  - Decision procedure: equivalence of **finite state automata**.
- **bi-Kleene algebra**: series-rational expressions.
  - Can be used to reason about **concurrent** programs.
  - Canonical model: pomset languages *i.e.* sets of **partially ordered** words.
  - Decision procedure: equivalence of **Petri nets**.

  Laurence & Struth, **Completeness theorems for bi-Kleene algebras and series-parallel rational pomset languages**, *2014*
- **Concurrent Kleene algebra**: series-rational expressions.
  - Can be used to reason about **concurrent** programs with a **refinement** order.
  - "Canonical" model: **downwards-closed** pomset languages.

Hoare, Möller, Struth & Wehrman, **Concurrent Kleene algebra and its foundations**, *2011*

Concurrent Kleene Algebra

$$e, f ::= 0 \mid 1 \mid a \mid e + f \mid e \cdot f \mid e^{\star} \mid e \parallel f$$

- **Kleene algebra**: rational expressions.
  - Can be used to reason about **sequential** programs.
  - Canonical model: regular languages, *i.e.* sets of words.
  - Decision procedure: equivalence of **finite state automata**.
- **bi-Kleene algebra**: series-rational expressions.
  - Can be used to reason about **concurrent** programs.
  - Canonical model: pomset languages *i.e.* sets of **partially ordered** words.
  - Decision procedure: equivalence of **Petri nets**.

  Laurence & Struth, **Completeness theorems for bi-Kleene algebras and series-parallel rational pomset languages**, *2014*
- **Concurrent Kleene algebra**: series-rational expressions.
  - Can be used to reason about **concurrent** programs with a **refinement** order.
  - "Canonical" model: **downwards-closed** pomset languages.
  - Decision procedure: containment of **Petri nets**.

  Hoare, Möller, Struth & Wehrman, **Concurrent Kleene algebra and its foundations**, *2011*

Concurrent Kleene Algebra

$$e, f ::= 0 \mid 1 \mid a \mid e + f \mid e \cdot f \mid e^\star \mid \color{red}{e \parallel f}$$

- **Kleene algebra**: rational expressions.
  - Can be used to reason about **sequential** programs.
  - Canonical model: regular languages, *i.e.* sets of words.
  - Decision procedure: equivalence of **finite state automata**.
- **bi-Kleene algebra**: series-rational expressions.
  - Can be used to reason about **concurrent** programs.
  - Canonical model: pomset languages *i.e.* sets of **partially ordered** words.
  - Decision procedure: equivalence of **Petri nets**.

  Laurence & Struth, **Completeness theorems for bi-Kleene algebras and series-parallel rational pomset languages**, *2014*
- **Concurrent Kleene algebra**: series-rational expressions.
  - Can be used to reason about **concurrent** programs with a **refinement** order.
  - "Canonical" model: **downwards-closed** pomset languages.
  - Decision procedure: containment of **Petri nets**.

  Hoare, Möller, Struth & Wehrman, **Concurrent Kleene algebra and its foundations**, *2011*

Outline

I. Pomsets

II. Petri Nets

III. Summary and Outlook

Outline

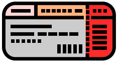
I. Pomsets

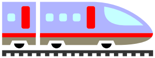
II. Petri Nets

III. Summary and Outlook

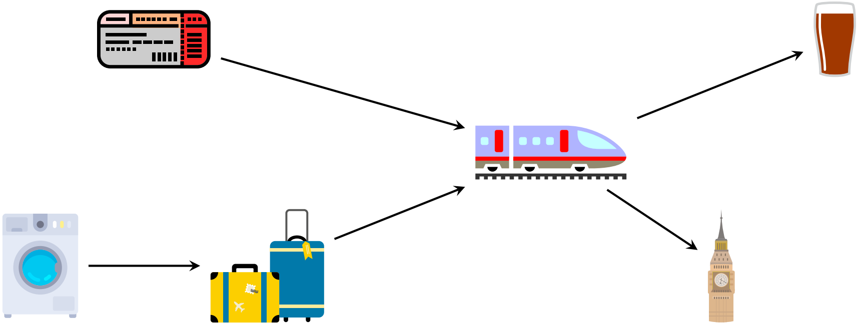
Let's visit London!

Let's visit London!

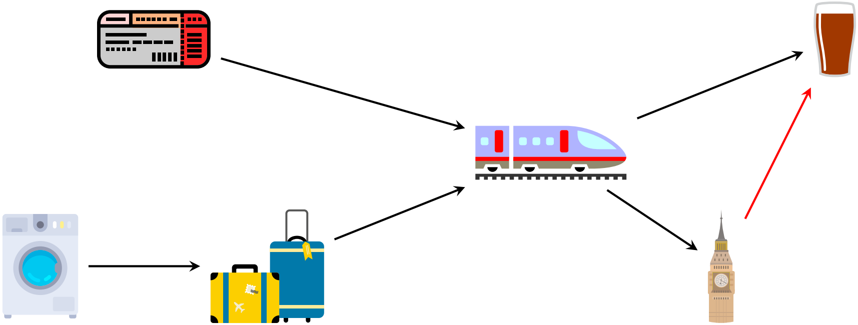




Let's visit London!



Let's visit London!



Pomsets products

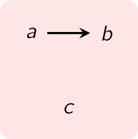
$$P_1 = \begin{array}{ccc} a & \longrightarrow & b \\ & c & \end{array} \qquad\qquad P_2 = \begin{array}{c} d \\ e \end{array}$$

Pomsets products

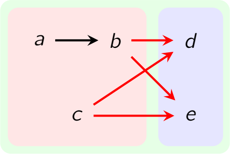
$$P_1 = \begin{tikzcd} a \arrow[r] & b \\ & c \end{tikzcd} \qquad P_2 = \begin{tikzcd} d \\ e \end{tikzcd}$$

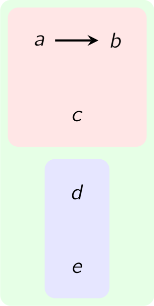
$$P_1 \cdot P_2 = \begin{tikzcd} a \arrow[r] & b \arrow[r] \arrow[dr] & d \\ & c \arrow[ur] \arrow[r] & e \end{tikzcd}$$

Pomsets products

$P_1 =$ 

$P_2 =$ 

$P_1 \cdot P_2 =$ 

$P_1 \parallel P_2 =$ 

Pomset order

$$
\begin{tikzcd}
a \arrow[r] \arrow[dr] & b \\
c \arrow[r] \arrow[ur] & d
\end{tikzcd}
\quad \sqsubseteq \quad
\begin{tikzcd}
a \arrow[r] & b \\
c \arrow[r] & d
\end{tikzcd}
$$

Pomset order

$$\begin{tikzcd} a \arrow[r] \arrow[dr] & b \\ c \arrow[r] \arrow[ur] & d \end{tikzcd} \quad \sqsubseteq \quad \begin{tikzcd} a \arrow[r] & b \\ c \arrow[r] & d \end{tikzcd}$$

Definition

$P_1 \sqsubseteq P_2$ if there is a function $\varphi : P_2 \to P_1$ such that:

1. φ is a bijection
2. φ preserves labels
3. φ preserves ordered pairs

Gischer, **The equational theory of pomsets**, *1988*

Grabowski, **On partial languages**, *1981*

Pomset order

$$\begin{tikzcd} a \arrow[r] \arrow[dr] & b \\ c \arrow[r] \arrow[ur] & d \end{tikzcd} \quad \sqsubseteq \quad \begin{tikzcd} a \arrow[r] & b \\ c \arrow[r] & d \end{tikzcd}$$

Definition

$P_1 \sqsubseteq P_2$ if there is a function $\varphi : P_2 \to P_1$ such that:

1. φ is a bijection
2. φ preserves labels
3. φ preserves ordered pairs

Gischer, **The equational theory of pomsets**, *1988*

Grabowski, **On partial languages**, *1981*

Notation

$^{\sqsubseteq}S := \{P \mid \exists P' \in S : P \sqsubseteq P'\}$.

Rational pomset languages

$$e, f \in \mathbb{E}_\Sigma ::= a \;\mid\; 0 \;\mid\; 1 \;\mid\; e \cdot f \;\mid\; e \parallel f \;\mid\; e + f \;\mid\; e^\star.$$

Rational pomset languages

$$e, f \in \mathbb{E}_\Sigma ::= a \mid 0 \mid 1 \mid e \cdot f \mid e \parallel f \mid e + f \mid e^\star.$$

$$\llbracket a \rrbracket := \left\{ \boxed{a} \right\}$$

$$\llbracket 0 \rrbracket := \emptyset$$

$$\llbracket e \cdot f \rrbracket := \llbracket e \rrbracket \cdot \llbracket f \rrbracket$$

$$\llbracket e^\star \rrbracket := \bigcup_{n \in \mathbb{N}} \llbracket e \rrbracket^n$$

$$\llbracket 1 \rrbracket := \left\{ \boxed{} \right\}$$

$$\llbracket e + f \rrbracket := \llbracket e \rrbracket \cup \llbracket f \rrbracket$$

$$\llbracket e \parallel f \rrbracket := \llbracket e \rrbracket \parallel \llbracket f \rrbracket$$

Rational pomset languages

$$e, f \in \mathbb{E}_\Sigma ::= a \mid 0 \mid 1 \mid e \cdot f \mid e \parallel f \mid e + f \mid e^\star.$$

$$\llbracket a \rrbracket := \left\{ \boxed{a} \right\} \qquad \llbracket 1 \rrbracket := \left\{ \boxed{} \right\}$$

$$\llbracket 0 \rrbracket := \emptyset \qquad \llbracket e + f \rrbracket := \llbracket e \rrbracket \cup \llbracket f \rrbracket$$

$$\llbracket e \cdot f \rrbracket := \llbracket e \rrbracket \cdot \llbracket f \rrbracket \qquad \llbracket e \parallel f \rrbracket := \llbracket e \rrbracket \parallel \llbracket f \rrbracket$$

$$\llbracket e^\star \rrbracket := \bigcup_{n \in \mathbb{N}} \llbracket e \rrbracket^n$$

Definition

A set of pomsets S is called a **rational pomset language** if there is an expression $e \in \mathbb{E}_\Sigma$ such that $S = \llbracket e \rrbracket$.

Two decision problems

biKA

Given two expressions e, f, are $[\![e]\!]$ and $[\![f]\!]$ equal?

CKA

Given two expressions e, f, are ${}^{\sqsubseteq}[\![e]\!]$ and ${}^{\sqsubseteq}[\![f]\!]$ equal?

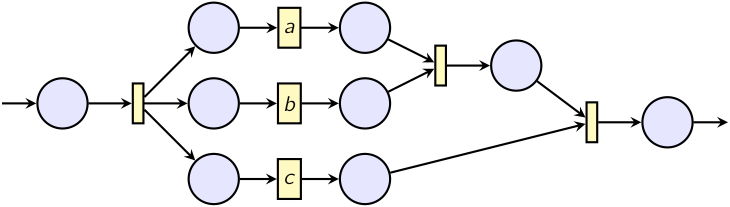
Outline

I. Pomsets

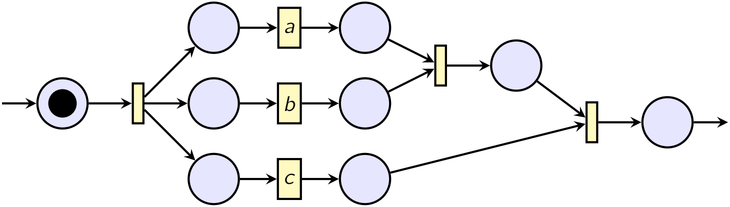
II. Petri Nets

III. Summary and Outlook

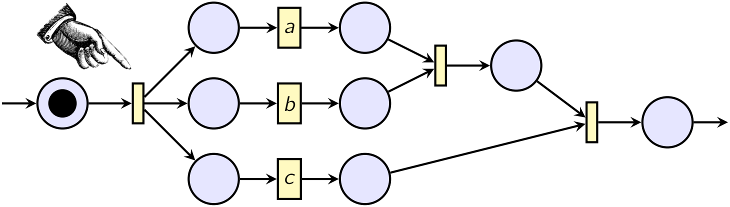
Labelled Petri nets



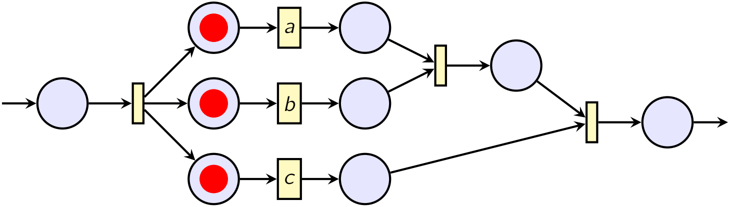
Labelled Petri nets



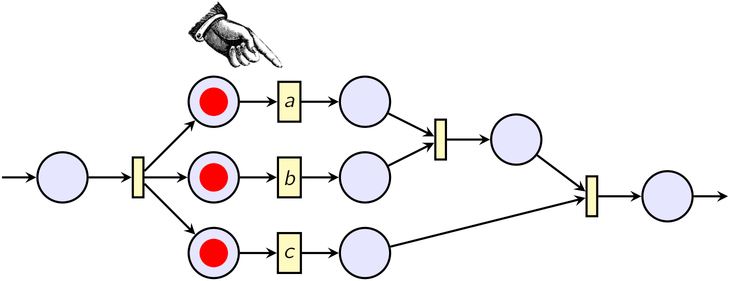
Labelled Petri nets



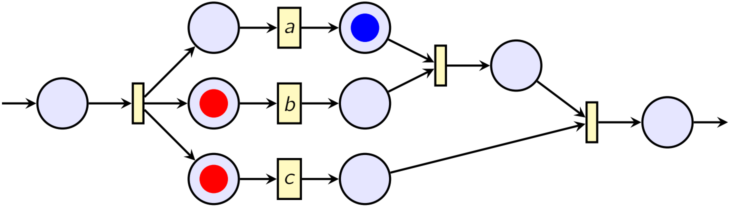
Labelled Petri nets

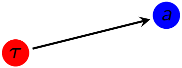


Labelled Petri nets

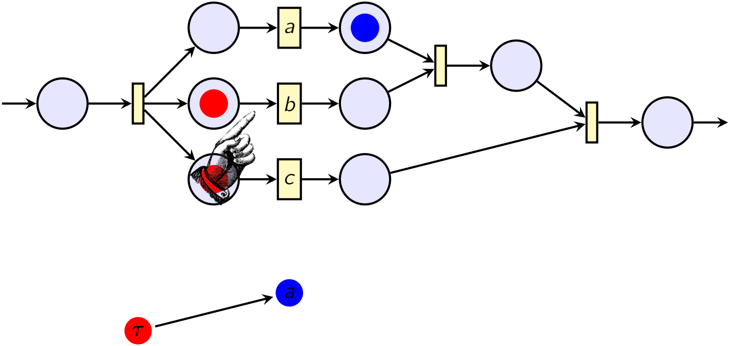


Labelled Petri nets

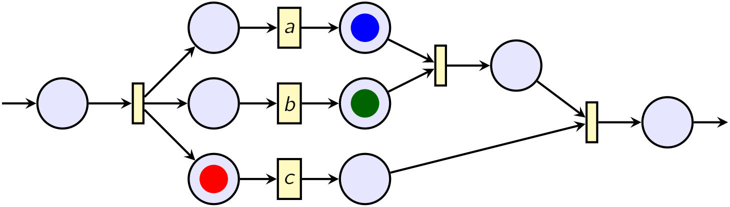


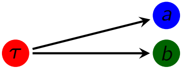


Labelled Petri nets

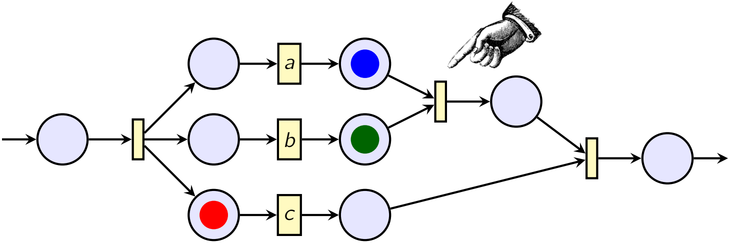


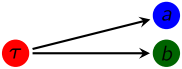
Labelled Petri nets



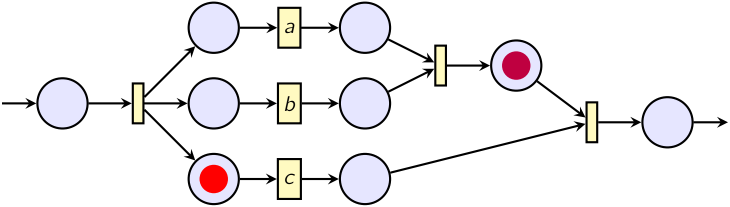


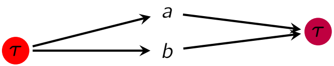
Labelled Petri nets



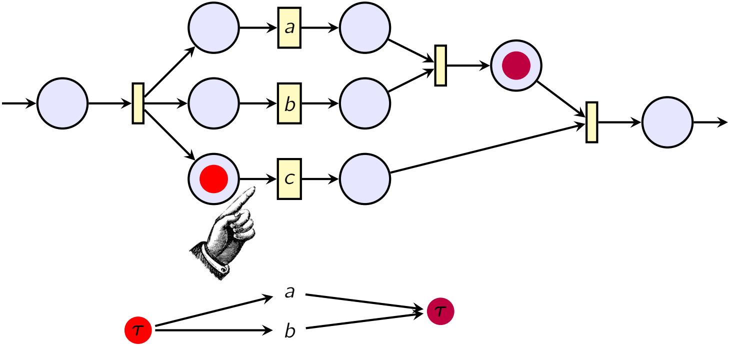


Labelled Petri nets

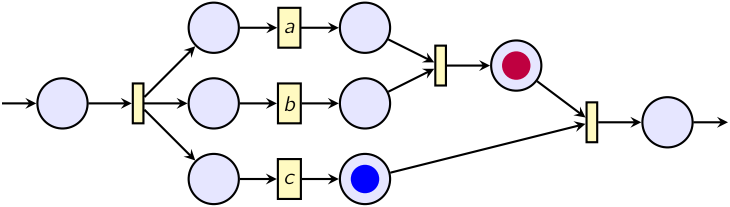


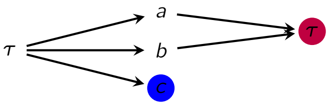


Labelled Petri nets

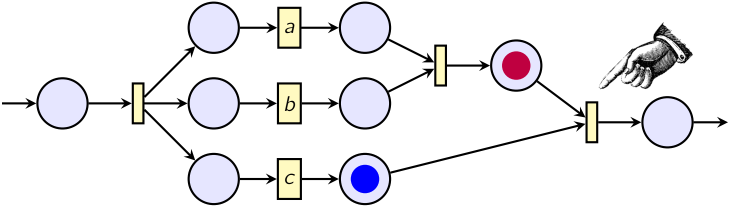


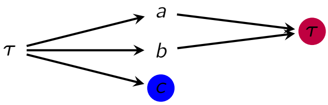
Labelled Petri nets



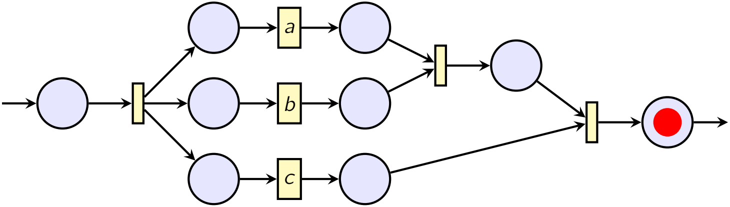


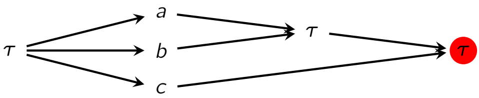
Labelled Petri nets



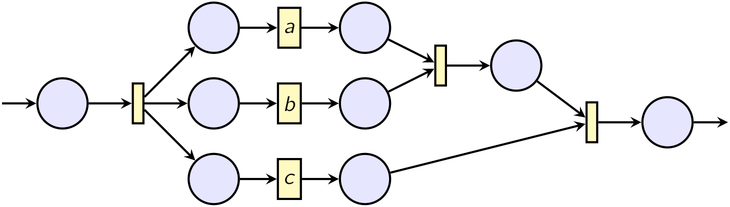


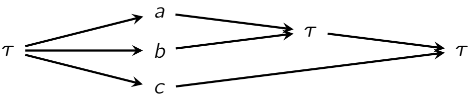
Labelled Petri nets



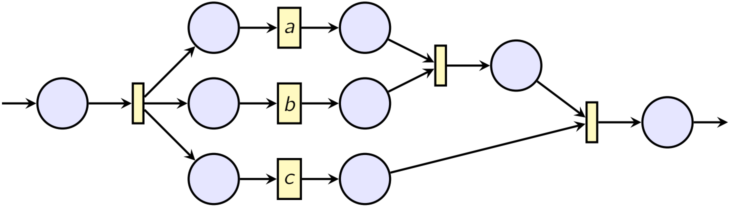


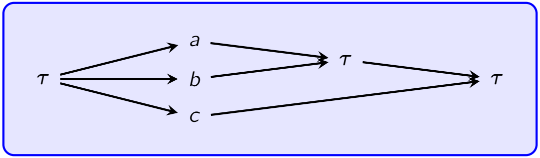
Labelled Petri nets





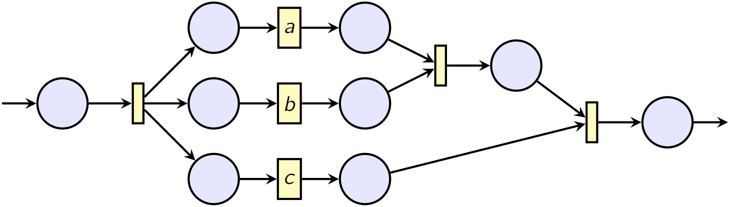
Labelled Petri nets

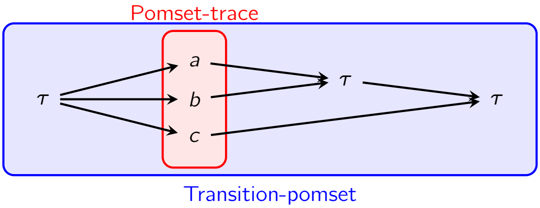




Transition-pomset

Labelled Petri nets





Recognisable pomset languages

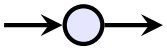
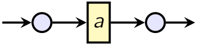
Language generated by a net

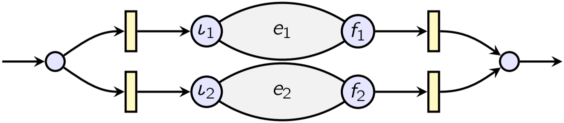
$[\![\mathcal{N}]\!]$ is the set of pomset-traces of accepting runs of $\mathcal{N}$.

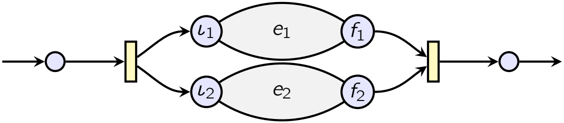
Definition

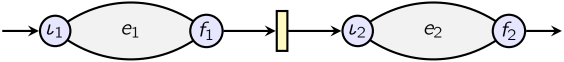
A set of pomsets S is a **recognisable pomset language** if there is a net $\mathcal{N}$ such that $S = [\![\mathcal{N}]\!]$.

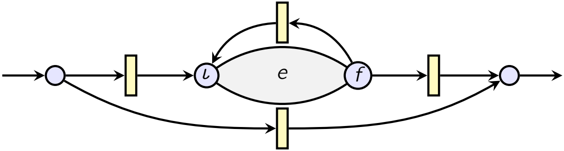
From expressions to automata

$\mathcal{N}(0) :=$  $\mathcal{N}(1) :=$  $\mathcal{N}(a) :=$ 

$\mathcal{N}(e_1 + e_2) :=$ 

$\mathcal{N}(e_1 \parallel e_2) :=$ 

$\mathcal{N}(e_1 \cdot e_2) :=$ 

$\mathcal{N}(e^\star) :=$ 

Solving biKA

Lemma

$$[\![e]\!] = [\![\mathcal{N}(e)]\!].$$

Corollary

Rational pomset languages are recognisable.

Solving biKA

Lemma

$$[\![e]\!] = [\![\mathcal{N}(e)]\!].$$

Corollary

Rational pomset languages are recognisable.

Theorem

Testing containment of pomset-trace languages of two Petri nets is an **ExpSpace**-complete problem.

Jategaonkar & Meyer, **Deciding true concurrency equivalences on safe, finite nets**, *1996*

Corollary

The problem biKA lies in the class **ExpSpace**.

What about CKA?

$${}^{\sqsubseteq}[\![e]\!] = {}^{\sqsubseteq}[\![f]\!]$$

What about CKA?

$${}^{\sqsubseteq}[\![e]\!] = {}^{\sqsubseteq}[\![f]\!] \Leftrightarrow \quad {}^{\sqsubseteq}[\![e]\!] \subseteq {}^{\sqsubseteq}[\![f]\!] \quad \wedge \quad {}^{\sqsubseteq}[\![e]\!] \supseteq {}^{\sqsubseteq}[\![f]\!]$$

What about CKA?

$$
\begin{aligned}
{}^{\sqsubseteq}[\![e]\!] = {}^{\sqsubseteq}[\![f]\!] &\Leftrightarrow \quad {}^{\sqsubseteq}[\![e]\!] \subseteq {}^{\sqsubseteq}[\![f]\!] \quad \wedge \quad {}^{\sqsubseteq}[\![e]\!] \supseteq {}^{\sqsubseteq}[\![f]\!] \\
&\Leftrightarrow \quad [\![e]\!] \subseteq {}^{\sqsubseteq}[\![f]\!] \quad \wedge \quad {}^{\sqsubseteq}[\![e]\!] \supseteq [\![f]\!]
\end{aligned}
$$

What about CKA?

$$\begin{aligned}
{}^{\sqsubseteq}[\![e]\!] = {}^{\sqsubseteq}[\![f]\!] &\Leftrightarrow \quad {}^{\sqsubseteq}[\![e]\!] \subseteq {}^{\sqsubseteq}[\![f]\!] \quad \wedge \quad {}^{\sqsubseteq}[\![e]\!] \supseteq {}^{\sqsubseteq}[\![f]\!] \\
&\Leftrightarrow \quad [\![e]\!] \subseteq {}^{\sqsubseteq}[\![f]\!] \quad \wedge \quad {}^{\sqsubseteq}[\![e]\!] \supseteq [\![f]\!] \\
&\Leftrightarrow [\![\mathcal{N}(e)]\!] \subseteq {}^{\sqsubseteq}[\![\mathcal{N}(f)]\!] \wedge {}^{\sqsubseteq}[\![\mathcal{N}(e)]\!] \supseteq [\![\mathcal{N}(f)]\!]
\end{aligned}$$

What about CKA?

$$\begin{aligned}
{}^{\sqsubseteq}[\![e]\!] = {}^{\sqsubseteq}[\![f]\!] &\Leftrightarrow \quad {}^{\sqsubseteq}[\![e]\!] \subseteq {}^{\sqsubseteq}[\![f]\!] \quad \wedge \quad {}^{\sqsubseteq}[\![e]\!] \supseteq {}^{\sqsubseteq}[\![f]\!] \\
&\Leftrightarrow \quad [\![e]\!] \subseteq {}^{\sqsubseteq}[\![f]\!] \quad \wedge \quad {}^{\sqsubseteq}[\![e]\!] \supseteq [\![f]\!] \\
&\Leftrightarrow [\![\mathcal{N}(e)]\!] \subseteq {}^{\sqsubseteq}[\![\mathcal{N}(f)]\!] \wedge {}^{\sqsubseteq}[\![\mathcal{N}(e)]\!] \supseteq [\![\mathcal{N}(f)]\!]
\end{aligned}$$

Problem

Let $\mathcal{N}_1, \mathcal{N}_2$ be well behaved nets. Is it true that for every run R_1 of $\mathcal{N}_1$ there is a run R_2 in $\mathcal{N}_2$ such that

$$\mathcal{P}om(R_1) \sqsubseteq \mathcal{P}om(R_2)?$$

Idea of the algorithm

Problem

Let $\mathcal{N}_1, \mathcal{N}_2$ be well behaved nets. Is it true that for every run R_1 of $\mathcal{N}_1$ there is a run R_2 in $\mathcal{N}_2$ such that

$$\mathcal{P}om(R_1) \sqsubseteq \mathcal{P}om(R_2)?$$

Idea of the algorithm

Problem

Let $\mathcal{N}_1, \mathcal{N}_2$ be well behaved nets. Is it true that for every run R_1 of $\mathcal{N}_1$ there is a run R_2 in $\mathcal{N}_2$ such that

$$\mathcal{P}om(R_1) \sqsubseteq \mathcal{P}om(R_2)?$$

- build an automaton $\mathscr{A}_1$ for $[\![\mathcal{N}_1]\!]$

Idea of the algorithm

Problem

Let $\mathcal{N}_1, \mathcal{N}_2$ be well behaved nets. Is it true that for every run R_1 of $\mathcal{N}_1$ there is a run R_2 in $\mathcal{N}_2$ such that

$$\mathcal{P}om(R_1) \sqsubseteq \mathcal{P}om(R_2)?$$

- build an automaton $\mathscr{A}_1$ for $[\![\mathcal{N}_1]\!]$
- build an automaton $\mathscr{A}_2$ for $[\![\mathcal{N}_1]\!] \cap^{\sqsubseteq} [\![\mathcal{N}_2]\!]$

Idea of the algorithm

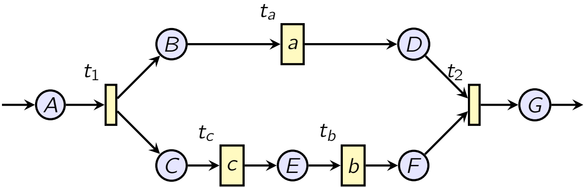
Problem

Let $\mathcal{N}_1, \mathcal{N}_2$ be well behaved nets. Is it true that for every run R_1 of $\mathcal{N}_1$ there is a run R_2 in $\mathcal{N}_2$ such that

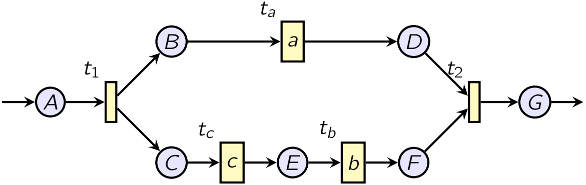
$$\mathcal{P}om\,(R_1) \sqsubseteq \mathcal{P}om\,(R_2)?$$

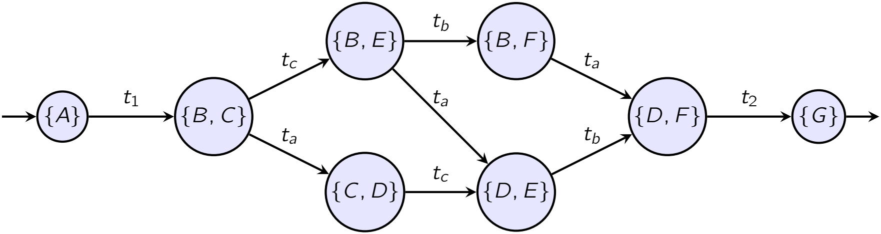
- build an automaton $\mathscr{A}_1$ for $[\![\mathcal{N}_1]\!]$
- build an automaton $\mathscr{A}_2$ for $[\![\mathcal{N}_1]\!] \cap^{\sqsubseteq} [\![\mathcal{N}_2]\!]$
- $[\![\mathcal{N}_1]\!] \subseteq^{\sqsubseteq} [\![\mathcal{N}_2]\!]$ if and only if $\mathcal{L}\,(\mathscr{A}_1) = \mathcal{L}\,(\mathscr{A}_2)$.

Transition automaton

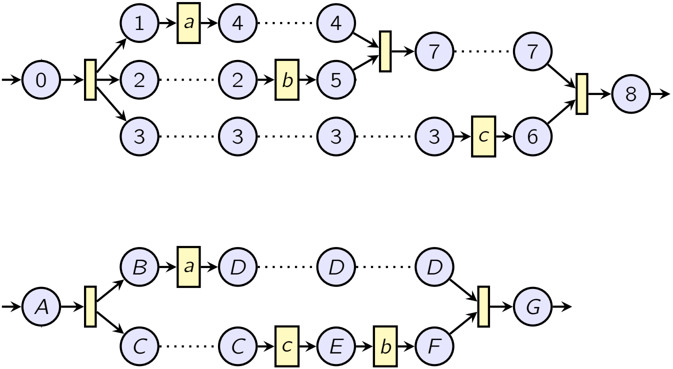


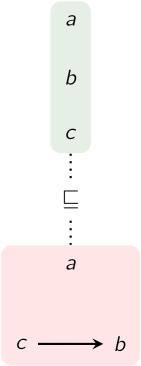
Transition automaton



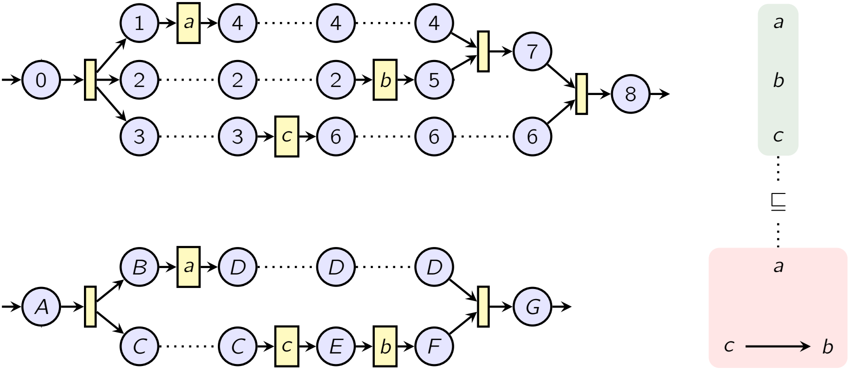


Massaging runs

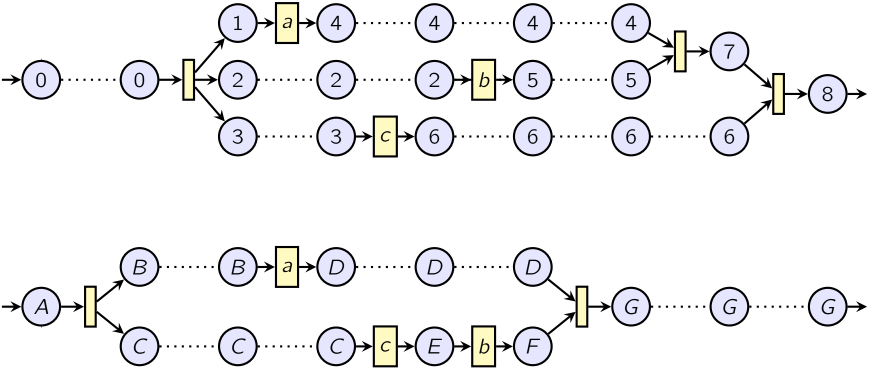




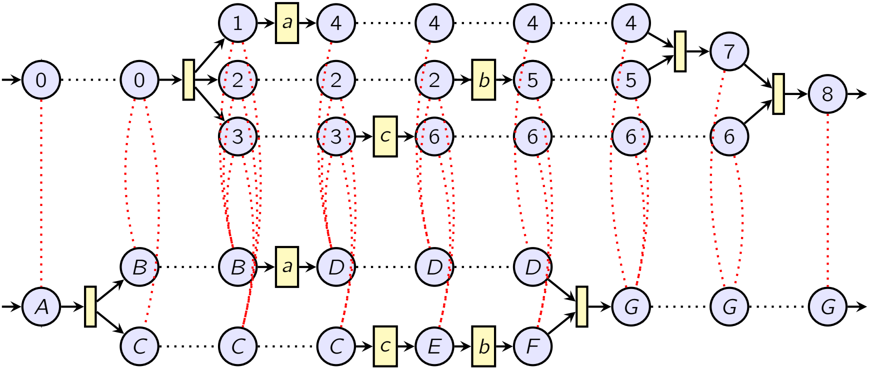
Massaging runs



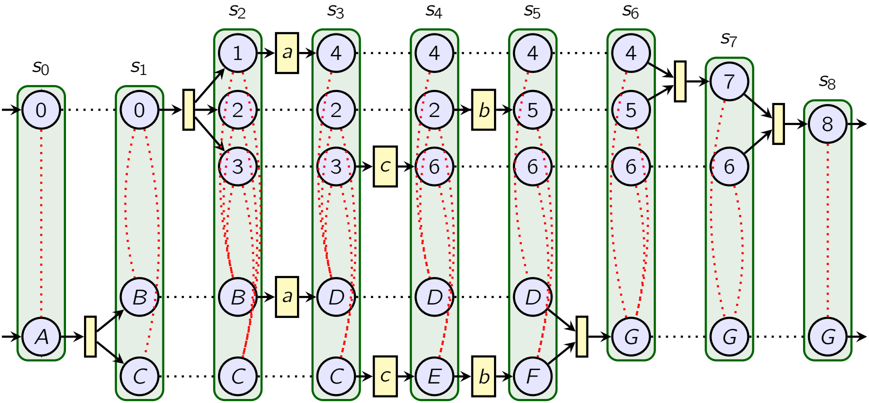
Massaging runs



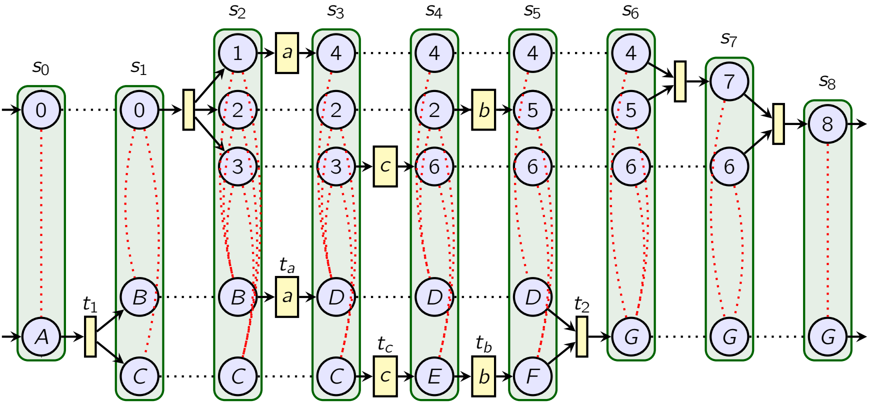
Massaging runs

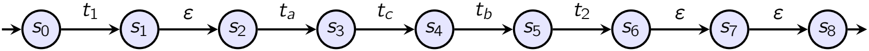


Massaging runs



Massaging runs





Reduction to automata

Let $\mathcal{N}_1$ and $\mathcal{N}_2$ be some polite nets, of size n, m.

Lemma

There is an automaton $\mathscr{A}(\mathcal{N}_1)$ with $\mathcal{O}(2^n)$ states that recognises the set of accepting runs in $\mathcal{N}_1$.

Reduction to automata

Let $\mathcal{N}_1$ and $\mathcal{N}_2$ be some polite nets, of size n, m.

Lemma

There is an automaton $\mathscr{A}(\mathcal{N}_1)$ with $\mathcal{O}(2^n)$ states that recognises the set of accepting runs in $\mathcal{N}_1$.

Lemma

There is an automaton $\mathcal{N}_1 \prec \mathcal{N}_2$ with $\mathcal{O}(2^{n+m+nm})$ states that recognises the set of accepting runs in $\mathcal{N}_1$ whose pomset belongs to ${}^{\sqsubseteq}[\![\mathcal{N}_2]\!]$.

Outline

I. Pomsets

II. Petri Nets

III. Summary and Outlook

Main result

Theorem

Given two expressions $e, f \in \mathbb{E}_{\Sigma}$, we can test if $[\![e]\!] \subseteq^{\sqsubseteq} [\![f]\!]$ in **ExpSpace**.

Proof.

1. build $\mathcal{N}(e)$ and $\mathcal{N}(f)$;
2. build $\mathscr{A}(\mathcal{N}(e))$ and $\mathcal{N}(e) \prec \mathcal{N}(f)$;
3. compare them.

$\square$

Main result

Theorem

Given two expressions $e, f \in \mathbb{E}_\Sigma$, we can test if $[\![e]\!] \subseteq^{\sqsubseteq} [\![f]\!]$ in **ExpSpace**.

Proof.

1. build $\mathcal{N}(e)$ and $\mathcal{N}(f)$;
2. build $\mathscr{A}(\mathcal{N}(e))$ and $\mathcal{N}(e) \prec \mathcal{N}(f)$;
3. compare them.

□

Theorem

The problem CKA is **ExpSpace**-complete.

Proof.

1. In the class **ExpSpace**: see above.
2. **ExpSpace**-hard: Reduction from the universality problem for regular expressions with interleaving.

Mayer & Stockmeyer, **The complexity of word problems – this time with interleaving**, *1994* □

To sum up

Done:

To do:

To sum up

Done:

- Reduction of biKA and CKA to **Petri nets**.

To do:

To sum up

Done:

- Reduction of biKA and CKA to **Petri nets**.
- New automaton-like **semantics** for Petri nets.

To do:

To sum up

Done:

- Reduction of biKA and CKA to **Petri nets**.
- New automaton-like **semantics** for Petri nets.
- biKA is **ExpSpace-solvable**.

To do:

To sum up

Done:

- Reduction of biKA and CKA to **Petri nets**.
- New automaton-like **semantics** for Petri nets.
- biKA is **ExpSpace-solvable**.
- CKA is **ExpSpace-complete**.

To do:

To sum up

Done:

- Reduction of biKA and CKA to **Petri nets**.
- New automaton-like **semantics** for Petri nets.
- biKA is **ExpSpace-solvable**.
- CKA is **ExpSpace-complete**.

To do:

- Extend the algorithm to a larger class of Petri nets.

To sum up

Done:

- Reduction of biKA and CKA to **Petri nets**.
- New automaton-like **semantics** for Petri nets.
- biKA is **ExpSpace-solvable**.
- CKA is **ExpSpace-complete**.

To do:

- Extend the algorithm to a larger class of Petri nets.
- Add **tests** because they're useful!

To sum up

Done:

- Reduction of biKA and CKA to **Petri nets**.
- New automaton-like **semantics** for Petri nets.
- biKA is **ExpSpace-solvable**.
- CKA is **ExpSpace-complete**.

To do:

- Extend the algorithm to a larger class of Petri nets.
- Add **tests** because they're useful!
- Add **names** because they're fun!

To sum up

Done:

- Reduction of biKA and CKA to **Petri nets**.
- New automaton-like **semantics** for Petri nets.
- biKA is **ExpSpace-solvable**.
- CKA is **ExpSpace-complete**.

To do:

- Extend the algorithm to a larger class of Petri nets.
- Add **tests** because they're useful!
- Add **names** because they're fun!
- *Insert you favourite operator here...*

That's all folks!

Thank you!

See more at:
`http://paul.brunet-zamansky.fr`

Outline

I. Pomsets

II. Petri Nets

III. Summary and Outlook