Simple Grammar Bisimilarity, with an Application to Session Type Equivalence

Diogo Poças
Instituto Superior Técnico

joint work with:
Gil Silva, Vasco T. Vasconcelos

12 January 2026

Introduction [Gla90, Mil71, Mil89]

Equivalence problem: when are two machines equivalent?

Equivalence problem: when are two machines equivalent?

- **Computation models** (automata, Turing machines): equivalence = language equivalence (do the machines recognize the same language?)

Equivalence problem: when are two machines equivalent?

- **Computation models** (automata, Turing machines): equivalence = language equivalence (do the machines recognize the same language?)
- **Process theory**: finer notions of equivalence, among which bisimilarity, or bisimulation equivalence.

Equivalence problem: when are two machines equivalent?

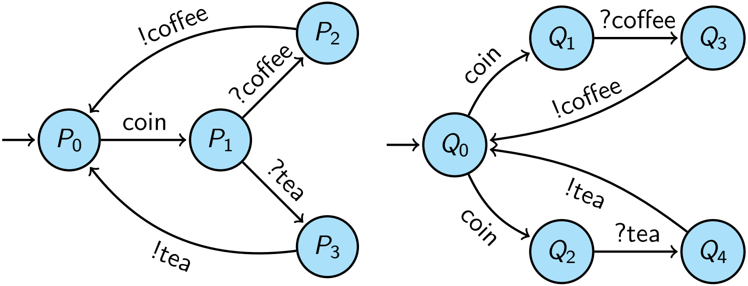
- **Computation models** (automata, Turing machines): equivalence = language equivalence (do the machines recognize the same language?)
- **Process theory**: finer notions of equivalence, among which bisimilarity, or bisimulation equivalence.
- **Some applications**: Calculus of Communicating Systems, π-calculus.

Bisimulation equivalence [San14]

- Subtleties of bisimilarity: **nontermination** and **nondeterminism**.

Bisimulation equivalence [San14]

- Subtleties of bisimilarity: **nontermination** and **nondeterminism**.



Context-free grammars

- **Context-free grammar** (CFG) in Greibach normal form: $(\mathcal{V}, \mathcal{T}, \mathcal{P})$
  - $\mathcal{V}$: finite set of **nonterminal** symbols $X, Y, Z, \ldots$
  - $\mathcal{T}$: finite set of **terminal** symbols $a, b, c, \ldots$
  - $\mathcal{P}$: **productions**, subset of $\mathcal{V} \times \mathcal{T} \times \mathcal{V}^*$;

$$(X, a, \gamma) \in \mathcal{P} \quad \Longleftrightarrow \quad X \xrightarrow{a} \gamma$$

- **Example** (palindromes):

$$X \xrightarrow{a} \varepsilon \qquad X \xrightarrow{a} A \qquad X \xrightarrow{a} XA \qquad A \xrightarrow{a} \varepsilon$$

$$X \xrightarrow{b} \varepsilon \qquad X \xrightarrow{b} B \qquad X \xrightarrow{b} XB \qquad B \xrightarrow{b} \varepsilon$$

Context-free grammars

- **Context-free grammar** (CFG) in Greibach normal form: $(\mathcal{V}, \mathcal{T}, \mathcal{P})$
  - $\mathcal{V}$: finite set of **nonterminal** symbols $X, Y, Z, \ldots$
  - $\mathcal{T}$: finite set of **terminal** symbols $a, b, c, \ldots$
  - $\mathcal{P}$: **productions**, subset of $\mathcal{V} \times \mathcal{T} \times \mathcal{V}^*$;

$$(X, a, \gamma) \in \mathcal{P} \quad \Longleftrightarrow \quad X \xrightarrow{a} \gamma$$

- **Example** (palindromes):

$$\begin{array}{cccc} X \xrightarrow{a} \varepsilon & X \xrightarrow{a} A & X \xrightarrow{a} XA & A \xrightarrow{a} \varepsilon \\ X \xrightarrow{b} \varepsilon & X \xrightarrow{b} B & X \xrightarrow{b} XB & B \xrightarrow{b} \varepsilon \end{array}$$

- **Labeled transition system** induced by $(\mathcal{V}, \mathcal{T}, \mathcal{P})$:
  - **states**: words of nonterminals, $\mathcal{V}^*$;
  - **labelled transitions**:

$$X\delta \xrightarrow{a} \gamma\delta \quad \text{for each production } X \xrightarrow{a} \gamma \text{ and word } \delta$$

- **Example**: $X \xrightarrow{a} XA \xrightarrow{b} XBA \xrightarrow{a} BA \xrightarrow{b} A \xrightarrow{a} \varepsilon$

Bisimulation equivalence

- **Bisimulation** for context-free grammars: a relation $\mathcal{R} \subseteq \mathcal{V}^* \times \mathcal{V}^*$ such that for every $(\gamma, \delta) \in \mathcal{R}$ and $a \in L$,

  (zig) if $\gamma \xrightarrow{a} \gamma'$, then there exists δ' such that $\delta \xrightarrow{a} \delta'$ and $(\gamma', \delta') \in \mathcal{R}$;

  (zag) if $\delta \xrightarrow{a} \delta'$, then there exists γ' such that $\gamma \xrightarrow{a} \gamma'$ and $(\gamma', \delta') \in \mathcal{R}$.

Bisimulation equivalence

- **Bisimulation** for context-free grammars: a relation $\mathcal{R} \subseteq \mathcal{V}^* \times \mathcal{V}^*$ such that for every $(\gamma, \delta) \in \mathcal{R}$ and $a \in L$,

  (zig) if $\gamma \xrightarrow{a} \gamma'$, then there exists δ' such that $\delta \xrightarrow{a} \delta'$ and $(\gamma', \delta') \in \mathcal{R}$;
  (zag) if $\delta \xrightarrow{a} \delta'$, then there exists γ' such that $\gamma \xrightarrow{a} \gamma'$ and $(\gamma', \delta') \in \mathcal{R}$.
- **Bisimilarity**: γ, δ are bisimilar, $\gamma \sim \delta$, if there exists a bisimulation $\mathcal{R}$ such that $(\gamma, \delta) \in R$.

Bisimulation equivalence

- **Bisimulation** for context-free grammars: a relation $\mathcal{R} \subseteq \mathcal{V}^* \times \mathcal{V}^*$ such that for every $(\gamma, \delta) \in \mathcal{R}$ and $a \in L$,

  (zig) if $\gamma \xrightarrow{a} \gamma'$, then there exists δ' such that $\delta \xrightarrow{a} \delta'$ and $(\gamma', \delta') \in \mathcal{R}$;
  (zag) if $\delta \xrightarrow{a} \delta'$, then there exists γ' such that $\gamma \xrightarrow{a} \gamma'$ and $(\gamma', \delta') \in \mathcal{R}$.
- **Bisimilarity**: γ, δ are bisimilar, $\gamma \sim \delta$, if there exists a bisimulation $\mathcal{R}$ such that $(\gamma, \delta) \in R$.

Bisimilarity problem: given two words γ, δ over a context-free grammar, determine whether $\gamma \sim \delta$.

State of the art [HJM96, Kie13, BCS95, Jan12]

- **Deterministic**: for each X and a, there is at most one production $X \xrightarrow{a} \gamma$. (also called **simple grammars**)
- **Normed**: for each X, there is some sequence $u = a_1 \ldots a_n$ such that $X \xrightarrow{u} \varepsilon$.

State of the art [HJM96, Kie13, BCS95, Jan12]

- **Deterministic**: for each X and a, there is at most one production $X \xrightarrow{a} \gamma$. (also called **simple grammars**)
- **Normed**: for each X, there is some sequence $u = a_1 \ldots a_n$ such that $X \xrightarrow{u} \varepsilon$.

CFG bisimilarity	deterministic	non-deterministic
normed	P [HJM96]	P [HJM96]
unnormed	**EXPTIME**	EXPTIME-hard [Kie13] 2-EXPTIME [BCS95, Jan12]

State of the art [HJM96, Kie13, BCS95, Jan12]

- **Deterministic**: for each X and a, there is at most one production $X \xrightarrow{a} \gamma$. (also called **simple grammars**)
- **Normed**: for each X, there is some sequence $u = a_1 \dots a_n$ such that $X \xrightarrow{u} \varepsilon$.

CFG bisimilarity	deterministic	non-deterministic
normed	P [HJM96]	P [HJM96]
unnormed	**EXPTIME**	EXPTIME-hard [Kie13] 2-EXPTIME [BCS95, Jan12]

Theorem. There is an exponential-time algorithm for determining whether two words γ and δ in a simple grammar are bisimilar.

Application: session types [Hon93, THK94, HVK98, TV16]

- **Type theory**: design type systems supporting rich **programming languages**.

Application: session types [Hon93, THK94, HVK98, TV16]

- **Type theory**: design type systems supporting rich **programming languages**.
- **Context-free session types**: an approach to specifying communication protocols.

$$
\begin{aligned}
M &::= \mathsf{Int} \mid \mathsf{Bool} \mid \cdots \\
T &::= ?M \mid !M \mid \oplus\{\ell\colon T_\ell\}_{\ell \in L} \mid \&\{\ell\colon T_\ell\}_{\ell \in L} \\
&\quad\ \mid \mathsf{Skip} \mid T; U \mid x \mid \mu x.T
\end{aligned}
$$

Application: session types [Hon93, THK94, HVK98, TV16]

- **Type theory**: design type systems supporting rich **programming languages**.
- **Context-free session types**: an approach to specifying communication protocols.

$$\begin{aligned} M &::= \mathsf{Int} \mid \mathsf{Bool} \mid \cdots \\ T &::= ?M \mid !M \mid \oplus\{\ell\colon T_\ell\}_{\ell\in L} \mid \&\{\ell\colon T_\ell\}_{\ell\in L} \\ &\quad \mid \mathsf{Skip} \mid T;U \mid x \mid \mu x.T \end{aligned}$$

- **Type equivalence**: given two types T, U, decide whether $T \sim U$.

$$\mathsf{Skip}; T \sim T \qquad (T;U);V \sim T;(U;V)$$

$$\oplus\{\ell : T_\ell\}; U \sim \oplus\{\ell : T_\ell; U\} \qquad \mu x.T \sim T[\mu x.T/\alpha]$$

Application: session types [Hon93, THK94, HVK98, TV16]

- **Type theory**: design type systems supporting rich **programming languages**.
- **Context-free session types**: an approach to specifying communication protocols.

$$
\begin{aligned}
M &::= \mathsf{Int} \mid \mathsf{Bool} \mid \cdots \\
T &::= ?M \mid !M \mid \oplus\{\ell\colon T_\ell\}_{\ell\in L} \mid \&\{\ell\colon T_\ell\}_{\ell\in L} \\
&\quad\ \mid \mathsf{Skip} \mid T;U \mid x \mid \mu x.T
\end{aligned}
$$

- **Type equivalence**: given two types T, U, decide whether $T \sim U$.

$$
\begin{array}{cc}
\mathsf{Skip}; T \sim T & (T;U);V \sim T;(U;V) \\
\oplus\{\ell : T_\ell\}; U \sim \oplus\{\ell : T_\ell; U\} & \mu x.T \sim T[\mu x.T/\alpha]
\end{array}
$$

- Type equivalence is a subroutine in **type checking** with an associated cost at compile time.

Decidability of type equivalence [AMV20, FST19]

- There is a **translation** $T \mapsto \mathrm{word}(T)$ from context-free session types into words in a simple grammar.
- $T \sim U$ iff $\mathrm{word}(T) \sim \mathrm{word}(U)$.

Decidability of type equivalence [AMV20, FST19]

- There is a **translation** $T \mapsto \mathrm{word}(T)$ from context-free session types into words in a simple grammar.
- $T \sim U$ iff $\mathrm{word}(T) \sim \mathrm{word}(U)$.

Theorem. There is a polynomial-time algorithm for determining whether two context-free session types T and U are equivalent.

Decidability of type equivalence [AMV20, FST19]

- There is a **translation** $T \mapsto \text{word}(T)$ from context-free session types into words in a simple grammar.
- $T \sim U$ iff $\text{word}(T) \sim \text{word}(U)$.

Theorem. There is a polynomial-time algorithm for determining whether two context-free session types T and U are equivalent.

- Application: **FreeST** programming language (v5.0. . .).

Bisimulation basis [Cau90, JM99]

- $\gamma \sim \delta$ if there exists a bisimulation $\mathcal{R} \subseteq \mathcal{V}^* \times \mathcal{V}^*$ such that ...

Bisimulation basis [Cau90, JM99]

- $\gamma \sim \delta$ if there exists a bisimulation $\mathcal{R} \subseteq \mathcal{V}^* \times \mathcal{V}^*$ such that ...
- Define a **basis** as a finite relation $\mathcal{B} \subseteq \mathcal{V}^+ \times \mathcal{V}^+$.

Bisimulation basis [Cau90, JM99]

- $\gamma \sim \delta$ if there exists a bisimulation $\mathcal{R} \subseteq \mathcal{V}^* \times \mathcal{V}^*$ such that ...
- Define a **basis** as a finite relation $\mathcal{B} \subseteq \mathcal{V}^+ \times \mathcal{V}^+$.
- **Coinductive congruence** with respect to $\mathcal{B}$: relation $\gamma \equiv_{\mathcal{B}} \delta$ coinductively defined.

$$\text{ε-Ax}\ \frac{}{\varepsilon \equiv_{\mathcal{B}} \varepsilon} \qquad \text{BPA1}\ \frac{\beta\alpha' \equiv_{\mathcal{B}} \beta' \qquad (X, Y\beta) \in \mathcal{B}}{X\alpha' \equiv_{\mathcal{B}} Y\beta'}$$

$$\text{BPA2}\ \frac{\alpha \equiv_{\mathcal{B}} \alpha' \qquad \beta \equiv_{\mathcal{B}} \beta' \qquad (X\alpha, Y\beta) \in \mathcal{B}}{X\alpha' \equiv_{\mathcal{B}} Y\beta'}$$

Bisimulation basis [Cau90, JM99]

- $\gamma \sim \delta$ if there exists a bisimulation $\mathcal{R} \subseteq \mathcal{V}^* \times \mathcal{V}^*$ such that ...
- Define a **basis** as a finite relation $\mathcal{B} \subseteq \mathcal{V}^+ \times \mathcal{V}^+$.
- **Coinductive congruence** with respect to $\mathcal{B}$: relation $\gamma \equiv_{\mathcal{B}} \delta$ coinductively defined.

$$\varepsilon\text{-Ax}\ \frac{}{\varepsilon \equiv_{\mathcal{B}} \varepsilon} \qquad \text{BPA1}\ \frac{\beta\alpha' \equiv_{\mathcal{B}} \beta' \qquad (X, Y\beta) \in \mathcal{B}}{X\alpha' \equiv_{\mathcal{B}} Y\beta'}$$

$$\text{BPA2}\ \frac{\alpha \equiv_{\mathcal{B}} \alpha' \qquad \beta \equiv_{\mathcal{B}} \beta' \qquad (X\alpha, Y\beta) \in \mathcal{B}}{X\alpha' \equiv_{\mathcal{B}} Y\beta'}$$

Intuition:

- if $\beta\alpha' \sim \beta'$ and $X \sim Y\beta$ then $X\alpha' \sim Y\beta'$;
- if $\alpha \sim \alpha'$, $\beta \sim \beta'$ and $X\alpha \sim Y\beta$ then $X\alpha' \sim Y\beta'$.

Self-bisimulation

- A basis $\mathcal{B}$ is a **self-bisimulation** if for every $(\gamma, \delta) \in \mathcal{B}$ and every $a \in \mathcal{T}$:

  (zig) if $\gamma \xrightarrow{a} \gamma'$, then there exists δ' such that $\delta \xrightarrow{a} \delta'$ and $\gamma' \equiv_{\mathcal{B}} \delta'$;

  (zag) if $\delta \xrightarrow{a} \delta'$, then there exists γ' such that $\gamma \xrightarrow{a} \gamma'$ and $\gamma' \equiv_{\mathcal{B}} \delta'$.

Self-bisimulation

- A basis $\mathcal{B}$ is a **self-bisimulation** if for every $(\gamma, \delta) \in \mathcal{B}$ and every $a \in \mathcal{T}$:

  (zig) if $\gamma \xrightarrow{a} \gamma'$, then there exists δ' such that $\delta \xrightarrow{a} \delta'$ and $\gamma' \equiv_{\mathcal{B}} \delta'$;

  (zag) if $\delta \xrightarrow{a} \delta'$, then there exists γ' such that $\gamma \xrightarrow{a} \gamma'$ and $\gamma' \equiv_{\mathcal{B}} \delta'$.
- **Lemma.** If $\mathcal{B}$ is a **self-bisimulation** then $\equiv_{\mathcal{B}} \subseteq \sim$.
- **Lemma.** $\gamma \sim \delta$ iff there exists a self-bisimulation basis $\mathcal{B}$ such that $\gamma \equiv_{\mathcal{B}} \delta$.

Self-bisimulation

- A basis $\mathcal{B}$ is a **self-bisimulation** if for every $(\gamma, \delta) \in \mathcal{B}$ and every $a \in \mathcal{T}$:
  (zig) if $\gamma \xrightarrow{a} \gamma'$, then there exists δ' such that $\delta \xrightarrow{a} \delta'$ and $\gamma' \equiv_{\mathcal{B}} \delta'$;
  (zag) if $\delta \xrightarrow{a} \delta'$, then there exists γ' such that $\gamma \xrightarrow{a} \gamma'$ and $\gamma' \equiv_{\mathcal{B}} \delta'$.
- **Lemma.** If $\mathcal{B}$ is a **self-bisimulation** then $\equiv_{\mathcal{B}} \subseteq \sim$.
- **Lemma.** $\gamma \sim \delta$ iff there exists a self-bisimulation basis $\mathcal{B}$ such that $\gamma \equiv_{\mathcal{B}} \delta$.
- Key idea: under 'suitable' conditions on $\mathcal{B}$, $\gamma \equiv_{\mathcal{B}} \delta$ is **decidable**.

Basis-updating algorithm

Key ideas:

- Build a **derivation tree** of goals $\gamma \sim \delta$.

Basis-updating algorithm

Key ideas:

- Build a **derivation tree** of goals $\gamma \sim \delta$.
- Keep track of a basis $\mathcal{B} \subseteq \mathcal{V}^+ \times \mathcal{V}^+$, initially containing identical pairs (X, X).

Basis-updating algorithm

Key ideas:

- Build a **derivation tree** of goals $\gamma \sim \delta$.
- Keep track of a basis $\mathcal{B} \subseteq \mathcal{V}^+ \times \mathcal{V}^+$, initially containing identical pairs (X, X).
- When examining a goal (leaf in the derivation tree):

Basis-updating algorithm

Key ideas:

- Build a **derivation tree** of goals $\gamma \sim \delta$.
- Keep track of a basis $\mathcal{B} \subseteq \mathcal{V}^+ \times \mathcal{V}^+$, initially containing identical pairs (X, X).
- When examining a goal (leaf in the derivation tree):
  (a) **Finish** the leaf if $\gamma = \delta$ or if (γ, δ) coincides with an already visited node.

Basis-updating algorithm

Key ideas:

- Build a **derivation tree** of goals $\gamma \sim \delta$.
- Keep track of a basis $\mathcal{B} \subseteq \mathcal{V}^+ \times \mathcal{V}^+$, initially containing identical pairs (X, X).
- When examining a goal (leaf in the derivation tree):
  (a) **Finish** the leaf if $\gamma = \delta$ or if (γ, δ) coincides with an already visited node.
  (b) Apply a **congruence rule** if possible, creating new goals (children).

Basis-updating algorithm

Key ideas:

- Build a **derivation tree** of goals $\gamma \sim \delta$.
- Keep track of a basis $\mathcal{B} \subseteq \mathcal{V}^+ \times \mathcal{V}^+$, initially containing identical pairs (X, X).
- When examining a goal (leaf in the derivation tree):
  (a) **Finish** the leaf if $\gamma = \delta$ or if (γ, δ) coincides with an already visited node.
  (b) Apply a **congruence rule** if possible, creating new goals (children).
  (c) **Update** $\mathcal{B}$ if no congruence rule is available, creating new goals.

Basis-updating algorithm

Key ideas:

- Build a **derivation tree** of goals $\gamma \sim \delta$.
- Keep track of a basis $\mathcal{B} \subseteq \mathcal{V}^+ \times \mathcal{V}^+$, initially containing identical pairs (X, X).
- When examining a goal (leaf in the derivation tree):
  (a) **Finish** the leaf if $\gamma = \delta$ or if (γ, δ) coincides with an already visited node.
  (b) Apply a **congruence rule** if possible, creating new goals (children).
  (c) **Update** $\mathcal{B}$ if no congruence rule is available, creating new goals.
  (d) First try a BPA1 guess, later **backtrack** and try a BPA2 guess if the derivation fails.

Example of application

$$X \xrightarrow{a} \varepsilon \quad X \xrightarrow{b} ZC \quad X \xrightarrow{c} \varepsilon \quad Y \xrightarrow{a} \varepsilon \quad Y \xrightarrow{b} WC \quad Y \xrightarrow{c} V$$

$$Z \xrightarrow{a} \varepsilon \quad Z \xrightarrow{b} XD \quad W \xrightarrow{a} \varepsilon \quad W \xrightarrow{b} YD \quad V \xrightarrow{c} \varepsilon$$

$$C \xrightarrow{c} C \quad D \xrightarrow{d} D$$

$$XC \overset{?}{\sim} YC$$

Example of application

$$X \xrightarrow{a} \varepsilon \quad X \xrightarrow{b} ZC \quad X \xrightarrow{c} \varepsilon \quad Y \xrightarrow{a} \varepsilon \quad Y \xrightarrow{b} WC \quad Y \xrightarrow{c} V$$

$$Z \xrightarrow{a} \varepsilon \quad Z \xrightarrow{b} XD \quad W \xrightarrow{a} \varepsilon \quad W \xrightarrow{b} YD \quad V \xrightarrow{c} \varepsilon$$

$$C \xrightarrow{c} C \quad D \xrightarrow{d} D$$

$$\mathcal{B} = \{(X,X),(Y,Y),(Z,Z),(W,W),(V,V),(C,C),(D,D)\}$$

$$\boxed{XC, YC}$$

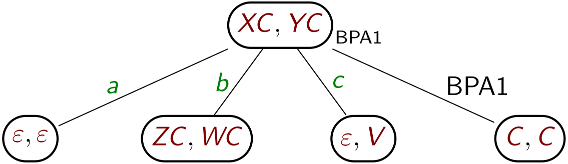
Example of application

$$X \xrightarrow{a} \varepsilon \quad X \xrightarrow{b} ZC \quad X \xrightarrow{c} \varepsilon \quad Y \xrightarrow{a} \varepsilon \quad Y \xrightarrow{b} WC \quad Y \xrightarrow{c} V$$

$$Z \xrightarrow{a} \varepsilon \quad Z \xrightarrow{b} XD \quad W \xrightarrow{a} \varepsilon \quad W \xrightarrow{b} YD \quad V \xrightarrow{c} \varepsilon$$

$$C \xrightarrow{c} C \quad D \xrightarrow{d} D$$

$$\mathcal{B} = \{(X,X),(Y,Y),(Z,Z),(W,W),(V,V),(C,C),(D,D),\boxed{(X,Y)}\}$$



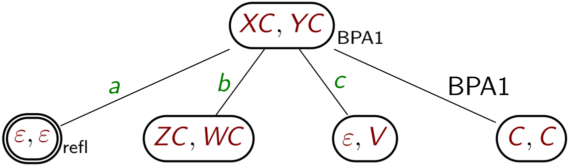
Example of application

$$X \xrightarrow{a} \varepsilon \quad X \xrightarrow{b} ZC \quad X \xrightarrow{c} \varepsilon \quad Y \xrightarrow{a} \varepsilon \quad Y \xrightarrow{b} WC \quad Y \xrightarrow{c} V$$

$$Z \xrightarrow{a} \varepsilon \quad Z \xrightarrow{b} XD \quad W \xrightarrow{a} \varepsilon \quad W \xrightarrow{b} YD \quad V \xrightarrow{c} \varepsilon$$

$$C \xrightarrow{c} C \quad D \xrightarrow{d} D$$

$$\mathcal{B} = \{(X, X), (Y, Y), (Z, Z), (W, W), (V, V), (C, C), (D, D), (X, Y)\}$$



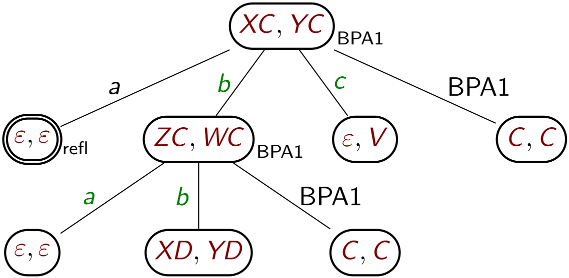
Example of application

$$X \xrightarrow{a} \varepsilon \quad X \xrightarrow{b} ZC \quad X \xrightarrow{c} \varepsilon \quad Y \xrightarrow{a} \varepsilon \quad Y \xrightarrow{b} WC \quad Y \xrightarrow{c} V$$

$$Z \xrightarrow{a} \varepsilon \quad Z \xrightarrow{b} XD \quad W \xrightarrow{a} \varepsilon \quad W \xrightarrow{b} YD \quad V \xrightarrow{c} \varepsilon$$

$$C \xrightarrow{c} C \quad D \xrightarrow{d} D$$

$$\mathcal{B} = \{(X,X),(Y,Y),(Z,Z),(W,W),(V,V),(C,C),(D,D),(X,Y),\boxed{(Z,W)}\}$$



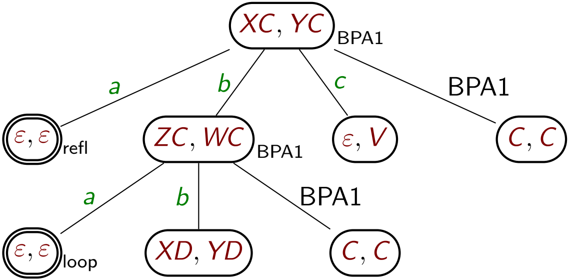
Example of application

$$X \xrightarrow{a} \varepsilon \quad X \xrightarrow{b} ZC \quad X \xrightarrow{c} \varepsilon \quad Y \xrightarrow{a} \varepsilon \quad Y \xrightarrow{b} WC \quad Y \xrightarrow{c} V$$

$$Z \xrightarrow{a} \varepsilon \quad Z \xrightarrow{b} XD \quad W \xrightarrow{a} \varepsilon \quad W \xrightarrow{b} YD \quad V \xrightarrow{c} \varepsilon$$

$$C \xrightarrow{c} C \quad D \xrightarrow{d} D$$

$$\mathcal{B} = \{(X, X), (Y, Y), (Z, Z), (W, W), (V, V), (C, C), (D, D), (X, Y), (Z, W)\}$$



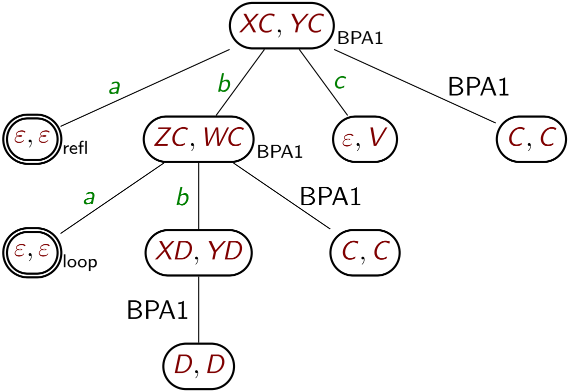
Example of application

$$X \xrightarrow{a} \varepsilon \quad X \xrightarrow{b} ZC \quad X \xrightarrow{c} \varepsilon \quad Y \xrightarrow{a} \varepsilon \quad Y \xrightarrow{b} WC \quad Y \xrightarrow{c} V$$

$$Z \xrightarrow{a} \varepsilon \quad Z \xrightarrow{b} XD \quad W \xrightarrow{a} \varepsilon \quad W \xrightarrow{b} YD \quad V \xrightarrow{c} \varepsilon$$

$$C \xrightarrow{c} C \quad D \xrightarrow{d} D$$

$$\mathcal{B} = \{(X,X),(Y,Y),(Z,Z),(W,W),(V,V),(C,C),(D,D),(X,Y),(Z,W)\}$$



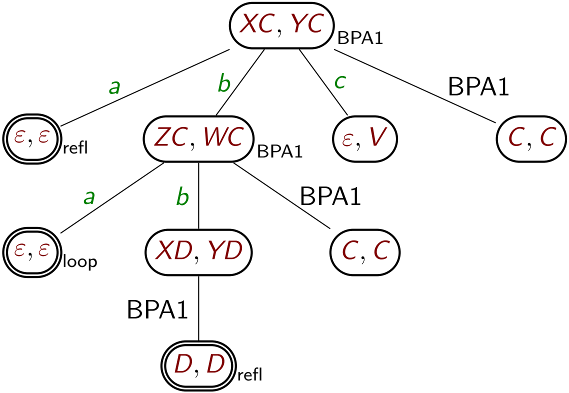
Example of application

$$X \xrightarrow{a} \varepsilon \quad X \xrightarrow{b} ZC \quad X \xrightarrow{c} \varepsilon \quad Y \xrightarrow{a} \varepsilon \quad Y \xrightarrow{b} WC \quad Y \xrightarrow{c} V$$

$$Z \xrightarrow{a} \varepsilon \quad Z \xrightarrow{b} XD \quad W \xrightarrow{a} \varepsilon \quad W \xrightarrow{b} YD \quad V \xrightarrow{c} \varepsilon$$

$$C \xrightarrow{c} C \quad D \xrightarrow{d} D$$

$$\mathcal{B} = \{(X,X),(Y,Y),(Z,Z),(W,W),(V,V),(C,C),(D,D),(X,Y),(Z,W)\}$$



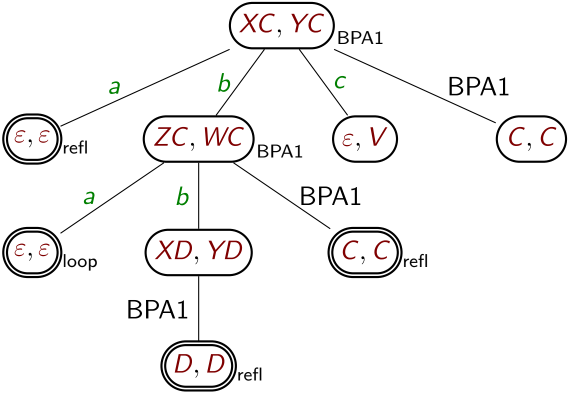
Example of application

$$X \xrightarrow{a} \varepsilon \quad X \xrightarrow{b} ZC \quad X \xrightarrow{c} \varepsilon \quad Y \xrightarrow{a} \varepsilon \quad Y \xrightarrow{b} WC \quad Y \xrightarrow{c} V$$

$$Z \xrightarrow{a} \varepsilon \quad Z \xrightarrow{b} XD \quad W \xrightarrow{a} \varepsilon \quad W \xrightarrow{b} YD \quad V \xrightarrow{c} \varepsilon$$

$$C \xrightarrow{c} C \quad D \xrightarrow{d} D$$

$$\mathcal{B} = \{(X,X),(Y,Y),(Z,Z),(W,W),(V,V),(C,C),(D,D),(X,Y),(Z,W)\}$$



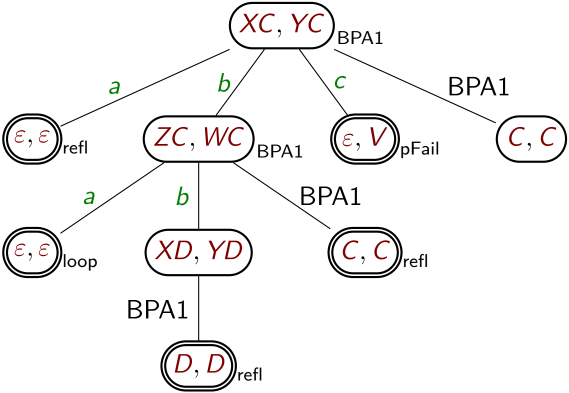
Example of application

$$X \xrightarrow{a} \varepsilon \quad X \xrightarrow{b} ZC \quad X \xrightarrow{c} \varepsilon \quad Y \xrightarrow{a} \varepsilon \quad Y \xrightarrow{b} WC \quad Y \xrightarrow{c} V$$

$$Z \xrightarrow{a} \varepsilon \quad Z \xrightarrow{b} XD \quad W \xrightarrow{a} \varepsilon \quad W \xrightarrow{b} YD \quad V \xrightarrow{c} \varepsilon$$

$$C \xrightarrow{c} C \quad D \xrightarrow{d} D$$

$$\mathcal{B} = \{(X,X),(Y,Y),(Z,Z),(W,W),(V,V),(C,C),(D,D),(X,Y),(Z,W)\}$$



Example of application

$$X \xrightarrow{a} \varepsilon \quad X \xrightarrow{b} ZC \quad X \xrightarrow{c} \varepsilon \quad Y \xrightarrow{a} \varepsilon \qquad Y \xrightarrow{b} WC \quad Y \xrightarrow{c} V$$

$$Z \xrightarrow{a} \varepsilon \quad Z \xrightarrow{b} XD \quad W \xrightarrow{a} \varepsilon \quad W \xrightarrow{b} YD \quad V \xrightarrow{c} \varepsilon$$

$$C \xrightarrow{c} C \quad D \xrightarrow{d} D$$

$$\mathcal{B} = \{(X, X), (Y, Y), (Z, Z), (W, W), (V, V), (C, C), (D, D)\}$$

$$\boxed{XC, YC}$$

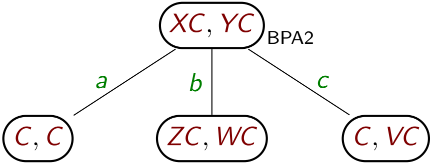
Example of application

$$X \xrightarrow{a} \varepsilon \quad X \xrightarrow{b} ZC \quad X \xrightarrow{c} \varepsilon \quad Y \xrightarrow{a} \varepsilon \quad Y \xrightarrow{b} WC \quad Y \xrightarrow{c} V$$

$$Z \xrightarrow{a} \varepsilon \quad Z \xrightarrow{b} XD \quad W \xrightarrow{a} \varepsilon \quad W \xrightarrow{b} YD \quad V \xrightarrow{c} \varepsilon$$

$$C \xrightarrow{c} C \quad D \xrightarrow{d} D$$

$$\mathcal{B} = \{(X, X), (Y, Y), (Z, Z), (W, W), (V, V), (C, C), (D, D), \boxed{(XC, YC)}\}$$



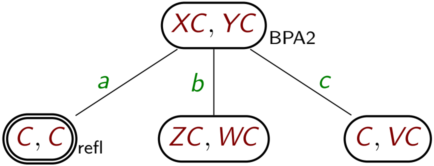
Example of application

$$X \xrightarrow{a} \varepsilon \quad X \xrightarrow{b} ZC \quad X \xrightarrow{c} \varepsilon \quad Y \xrightarrow{a} \varepsilon \quad Y \xrightarrow{b} WC \quad Y \xrightarrow{c} V$$

$$Z \xrightarrow{a} \varepsilon \quad Z \xrightarrow{b} XD \quad W \xrightarrow{a} \varepsilon \quad W \xrightarrow{b} YD \quad V \xrightarrow{c} \varepsilon$$

$$C \xrightarrow{c} C \quad D \xrightarrow{d} D$$

$$\mathcal{B} = \{(X, X), (Y, Y), (Z, Z), (W, W), (V, V), (C, C), (D, D), (XC, YC)\}$$



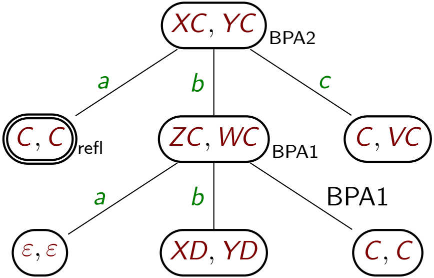
Example of application

$$X \xrightarrow{a} \varepsilon \quad X \xrightarrow{b} ZC \quad X \xrightarrow{c} \varepsilon \quad Y \xrightarrow{a} \varepsilon \quad Y \xrightarrow{b} WC \quad Y \xrightarrow{c} V$$

$$Z \xrightarrow{a} \varepsilon \quad Z \xrightarrow{b} XD \quad W \xrightarrow{a} \varepsilon \quad W \xrightarrow{b} YD \quad V \xrightarrow{c} \varepsilon$$

$$C \xrightarrow{c} C \quad D \xrightarrow{d} D$$

$$\mathcal{B} = \{(X,X),(Y,Y),(Z,Z),(W,W),(V,V),(C,C),(D,D),(XC,YC),\boxed{(Z,W)}\}$$



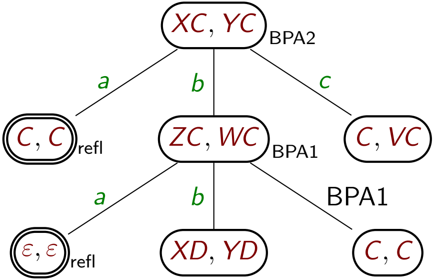
Example of application

$$X \xrightarrow{a} \varepsilon \quad X \xrightarrow{b} ZC \quad X \xrightarrow{c} \varepsilon \quad Y \xrightarrow{a} \varepsilon \quad Y \xrightarrow{b} WC \quad Y \xrightarrow{c} V$$

$$Z \xrightarrow{a} \varepsilon \quad Z \xrightarrow{b} XD \quad W \xrightarrow{a} \varepsilon \quad W \xrightarrow{b} YD \quad V \xrightarrow{c} \varepsilon$$

$$C \xrightarrow{c} C \quad D \xrightarrow{d} D$$

$$\mathcal{B} = \{(X,X),(Y,Y),(Z,Z),(W,W),(V,V),(C,C),(D,D),(XC,YC),(Z,W)\}$$



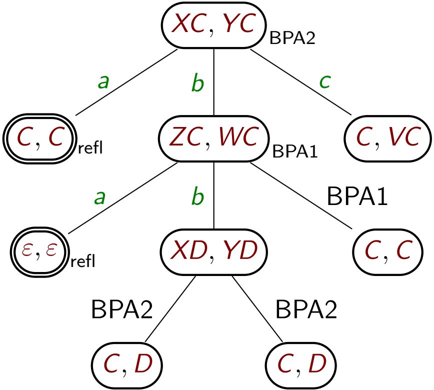
Example of application

$$X \xrightarrow{a} \varepsilon \quad X \xrightarrow{b} ZC \quad X \xrightarrow{c} \varepsilon \quad Y \xrightarrow{a} \varepsilon \quad Y \xrightarrow{b} WC \quad Y \xrightarrow{c} V$$

$$Z \xrightarrow{a} \varepsilon \quad Z \xrightarrow{b} XD \quad W \xrightarrow{a} \varepsilon \quad W \xrightarrow{b} YD \quad V \xrightarrow{c} \varepsilon$$

$$C \xrightarrow{c} C \quad D \xrightarrow{d} D$$

$$\mathcal{B} = \{(X, X), (Y, Y), (Z, Z), (W, W), (V, V), (C, C), (D, D), (XC, YC), (Z, W)\}$$



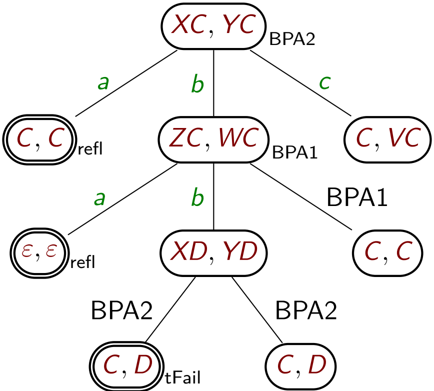
Example of application

$$X \xrightarrow{a} \varepsilon \quad X \xrightarrow{b} ZC \quad X \xrightarrow{c} \varepsilon \quad Y \xrightarrow{a} \varepsilon \quad Y \xrightarrow{b} WC \quad Y \xrightarrow{c} V$$

$$Z \xrightarrow{a} \varepsilon \quad Z \xrightarrow{b} XD \quad W \xrightarrow{a} \varepsilon \quad W \xrightarrow{b} YD \quad V \xrightarrow{c} \varepsilon$$

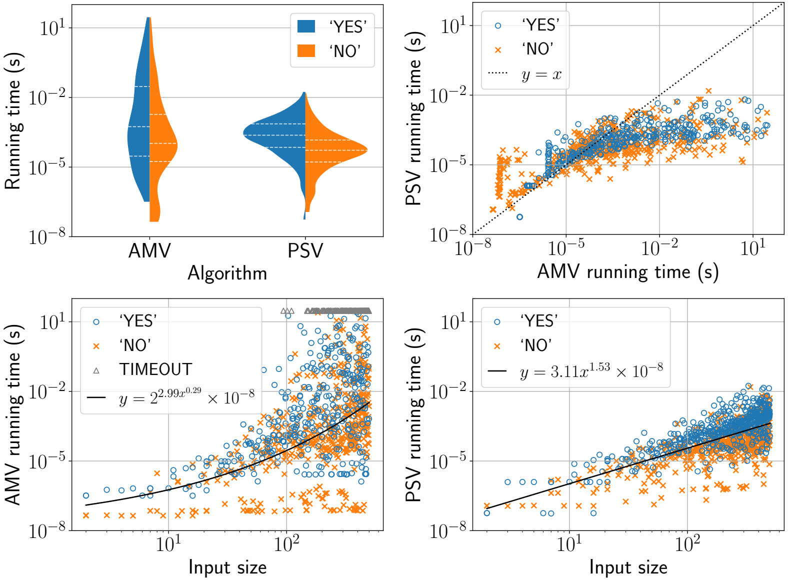
$$C \xrightarrow{c} C \quad D \xrightarrow{d} D$$

$$\mathcal{B} = \{(X,X),(Y,Y),(Z,Z),(W,W),(V,V),(C,C),(D,D),(XC,YC),(Z,W)\}$$



$$XC \not\sim YC$$

Numerical experiments [AMV20]



Conclusion and future work

- **Bisimilarity**: a notion of equivalence, with relevant applications in process theory.
- **Context-free session types**: specifying communication protocols.
- **Type equivalence**: an interesting problem, relevant for type checking, reducible to **simple grammar bisimilarity**.

Conclusion and future work

- **Bisimilarity**: a notion of equivalence, with relevant applications in process theory.
- **Context-free session types**: specifying communication protocols.
- **Type equivalence**: an interesting problem, relevant for type checking, reducible to **simple grammar bisimilarity**.
- **Open problem**: is there a polynomial algorithm for bisimilarity of simple grammars? Is the problem EXPTIME-complete?
- **Open problem**: is there a single-exponential algorithm for bisimilarity of context-free grammars? Is the problem 2-EXPTIME-complete?

Conclusion and future work

- **Bisimilarity**: a notion of equivalence, with relevant applications in process theory.
- **Context-free session types**: specifying communication protocols.
- **Type equivalence**: an interesting problem, relevant for type checking, reducible to **simple grammar bisimilarity**.
- **Open problem**: is there a polynomial algorithm for bisimilarity of simple grammars? Is the problem EXPTIME-complete?
- **Open problem**: is there a single-exponential algorithm for bisimilarity of context-free grammars? Is the problem 2-EXPTIME-complete?

Thank you for your attention!

Bibliography I

Bernardo Almeida, Andreia Mordido, and Vasco T. Vasconcelos.
Deciding the bisimilarity of context-free session types.
In *TACAS*, volume 12079 of *LNCS*, pages 39–56. Springer, 2020.

Olaf Burkart, Didier Caucal, and Bernhard Steffen.
An elementary bisimulation decision procedure for arbitrary context-free processes.
In *MFCS*, volume 969 of *LNCS*, pages 423–433. Springer, 1995.

Didier Caucal.
Graphes canoniques de graphes algébriques.
RAIRO - Theoretical Informatics and Applications, 24(4):339–352, 1990.

The FreeST programming language.
`https://freest-lang.github.io/`, 2019.

Rob J. van Glabbeek.
The linear time-branching time spectrum (extended abstract).
In *CONCUR*, volume 458 of *LNCS*, pages 278–297. Springer, 1990.

Yoram Hirshfeld, Mark Jerrum, and Faron Moller.
A polynomial algorithm for deciding bisimilarity of normed context-free processes.
Theor. Comput. Sci., 158(1):143–159, 1996.

Bibliography II

Kohei Honda.
Types for dyadic interaction.
In *CONCUR*, volume 715 of *LNCS*, pages 509–523. Springer, 1993.

Kohei Honda, Vasco Thudichum Vasconcelos, and Makoto Kubo.
Language primitives and type discipline for structured communication-based programming.
In *ESOP*, volume 1381 of *LNCS*, pages 122–138. Springer, 1998.

Petr Jančar.
Bisimilarity on basic process algebra is in 2-EXPTIME (an explicit proof).
Log. Methods Comput. Sci., 9(1), 2012.

Petr Jančar and Faron Moller.
Techniques for decidability and undecidability of bisimilarity.
In *CONCUR*, volume 1664 of *LNCS*, pages 30–45. Springer, 1999.

Stefan Kiefer.
BPA bisimilarity is EXPTIME-hard.
Inf. Process. Lett., 113(4):101–106, 2013.

Robin Milner.
An algebraic definition of simulation between programs.
In *IJCAI*, pages 481–489. William Kaufmann, 1971.

Bibliography III

R. Milner.
Communication and Concurrency.
Prentice Hall, 1989.

Davide Sangiorgi.
An Introduction to Bisimulation and Coinduction.
Cambridge University Press, 2014.

Kaku Takeuchi, Kohei Honda, and Makoto Kubo.
An interaction-based language and its typing system.
In *PARLE*, volume 817 of *LNCS*, pages 398–413. Springer, 1994.

Peter Thiemann and Vasco T. Vasconcelos.
Context-free session types.
In *ICFP*, pages 462–475. ACM, 2016.