Checking the equivalence of context-free session types

Andreia Mordido and Vasco T. Vasconcelos

LASIGE, Faculdade de Ciências, Universidade de Lisboa

January 2019

Motivation

```
1  data Tree = Leaf | Node Int Tree Tree

1  sendTree Leaf c =
2    select Leaf c
3  sendTree (Node x l r) c =
4    let c1 = select Node c
5        c2 = send x c1
6        c3 = sendTree l c2
7        c4 = sendTree r c3
8      in c4
```
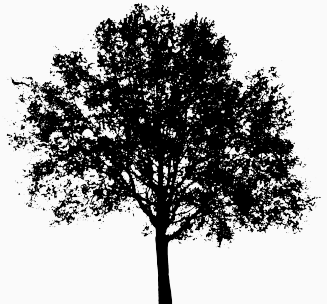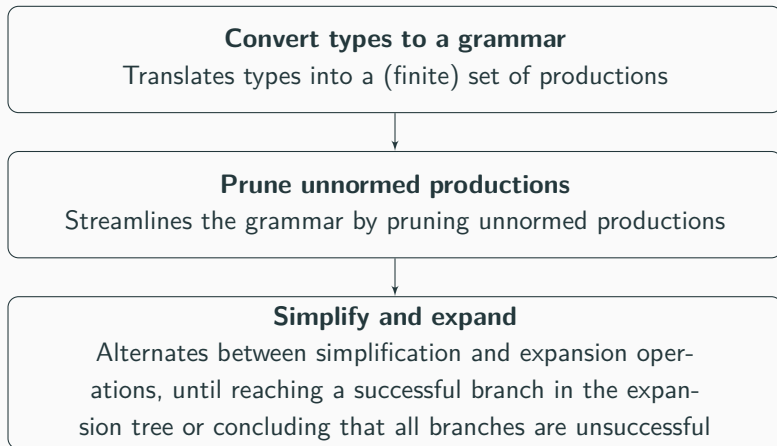
Plan

Definition (Type equivalence problem)

Given any context-free session types S and T, the type equivalence problem consists in deciding if types S and T are equivalent, i.e., $S \sim T$.

- Implement an algorithm that decides the *type equivalence problem*.
- Prove its soundness and completeness w.r.t. the metatheory of context-free session types proposed by Thiemann and Vasconcelos.
- Provide results on complexity.

Convert types to a grammar

Translates types into a (finite) set of productions

Prune unnormed productions

Streamlines the grammar by pruning unnormed productions

Simplify and expand

Alternates between simplification and expansion operations, until reaching a successful branch in the expansion tree or concluding that all branches are unsuccessful

Convert types to a grammar

We consider a finite set of productions $X \to a\vec{Y}$ where:

- X, Y represent *non-terminal variables*
- a is a *label* in the labelled transition system
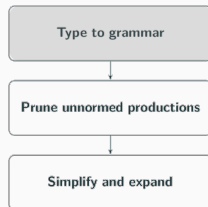
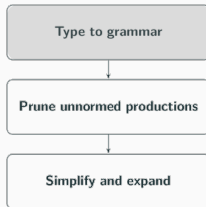Context-free session types are seen as *simple grammars*, i.e.:

- context-free grammars in *Greibach normal form*
- s.t. for each X and a there is at most one production $X \to a\vec{Y}$

Convert types to a grammar

Example

$$S \triangleq (\mu x.\&\{n\colon x; x, ?\,\mathsf{int}, \ell\colon ?\,\mathsf{int}\}); (\mu z.!\,\mathsf{int}; z; z)$$

$$T \triangleq (\mu y.\&\{n\colon y; y, \ell\colon \mathsf{skip}\}; ?\,\mathsf{int}); (\mu z.!\,\mathsf{int}; z)$$

Type to grammar

↓

Prune unnormed productions

↓

Simplify and expand

Convert types to a grammar

Example

$S \triangleq (\mu x.\&\{n\colon x; x, ?\,\text{int}, \ell\colon ?\,\text{int}\}); (\mu z.!\,\text{int}; z; z)$

$T \triangleq (\mu y.\&\{n\colon y; y, \ell\colon \text{skip}\}; ?\,\text{int}); (\mu z.!\,\text{int}; z)$
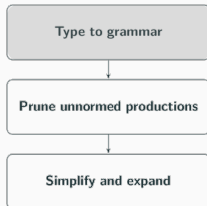
Productions for S	Productions for T
$X_1 \to \&n\, X_1 X_1 X_2$	$Y_1 \to \&n\, Y_1 Y_1 Y_2$
$X_1 \to \&\ell\, X_3$	$Y_1 \to \&\ell\, Y_2$
$X_2 \to ?\text{int}$	$Y_2 \to ?\text{int}$
$X_3 \to ?\text{int}$	$Y_3 \to !\text{int}\, Y_3$
$X_4 \to !\text{int}\, X_4 X_4$	

Type to grammar

Prune unnormed productions

Simplify and expand

6

Convert types to a grammar

Example
$$S \triangleq (\mu x.\&\{n\colon x; x, ?\,\text{int}, \ell\colon ?\,\text{int}\}); (\mu z.!\,\text{int}; z; z)$$
$$T \triangleq (\mu y.\&\{n\colon y; y, \ell\colon \text{skip}\}; ?\,\text{int}); (\mu z.!\,\text{int}; z)$$

Productions for S	Productions for T
$X \to !(\,)\, X_1 X_4$	$Y \to !(\,)\, Y_1 Y_3$
$X_1 \to \&n\, X_1 X_1 X_2$	$Y_1 \to \&n\, Y_1 Y_1 Y_2$
$X_1 \to \&\ell\, X_3$	$Y_1 \to \&\ell\, Y_2$
$X_2 \to ?\text{int}$	$Y_2 \to ?\text{int}$
$X_3 \to ?\text{int}$	$Y_3 \to !\text{int}\, Y_3$
$X_4 \to !\text{int}\, X_4 X_4$	

Type to grammar

Prune unnormed productions

Simplify and expand

Convert types to a grammar

LOHaskellCode ~ 100

Soundness ✓

1. Conversion of types into BPAs is sound[a].

2. Conversion of BPAs to Greibach Normal Form without altering the solution is sound[b].

---

[a]Thiemann and Vasconcelos. Context-free session types. 2016.
[b]Baeten et al. Decidability of bisimulation equivalence for process generating context-free languages. 1993.
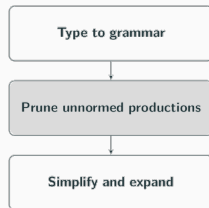
Prune unnormed productions

Simplify and expand

Prune unnormed productions

Definition ((Un)normed terms[1])

A term α is *normed* if there is a path from α to ε.
Otherwise, it is said to be *unnormed*.

Christensen, Huttel, and Stirling[1] noted that:

whenever α is unnormed, $\alpha \sim \alpha\beta$.

Type to grammar
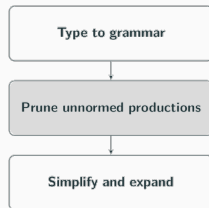
Prune unnormed productions

Simplify and expand

---

[1] Christensen et al. Bisimulation equivalence is decidable for all CF processes. 1995

Prune unnormed productions

Definition ((Un)normed terms[1])

A term α is *normed* if there is a path from α to ε.
Otherwise, it is said to be *unnormed*.

Christensen, Huttel, and Stirling[1] noted that:

   whenever α is unnormed, $\alpha \sim \alpha\beta$.

Type to grammar

Prune unnormed productions

Simplify and expand

Example

$$S \triangleq (\mu x.\&\{n\colon x; x; ?\,\text{int}, \ell\colon ?\,\text{int}\}); (\mu z.!\,\text{int}; z; z)$$

Productions for S

$X_1 \to \&n\, X_1 X_1 X_2$
$X_1 \to \&\ell\, X_3$
$X_2 \to ?\text{int}$
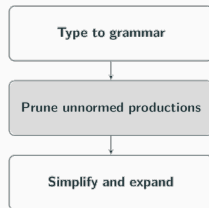$X_3 \to ?\text{int}$
$X_4 \to !\text{int}\, X_4 X_4$

[1] Christensen et al. Bisimulation equivalence is decidable for all CF processes. 1995

Prune unnormed productions

Definition ((Un)normed terms[1])

A term α is *normed* if there is a path from α to ε.
Otherwise, it is said to be *unnormed*.

Christensen, Huttel, and Stirling[1] noted that:

whenever α is unnormed, $\alpha \sim \alpha\beta$.

> Type to grammar
>
> ↓
>
> Prune unnormed productions
>
> ↓
>
> Simplify and expand

Example

$$S \triangleq (\mu x.\&\{n\colon x; x, ?\,\text{int}, \ell\colon ?\,\text{int}\}); (\mu z.!\,\text{int}; z; z)$$

Productions for S	Productions for *pruned* S
$X_1 \to \&n\, X_1 X_1 X_2$	$X_1 \to \&n\, X_1 X_1 X_2$
$X_1 \to \&\ell\, X_3$	$X_1 \to \&\ell\, X_3$
$X_2 \to ?\text{int}$	$X_2 \to ?\text{int}$
$X_3 \to ?\text{int}$	$X_3 \to ?\text{int}$
$X_4 \to !\text{int}\, X_4 X_4$	$X_4 \to !\text{int}\, X_4$

[1] Christensen et al. Bisimulation equivalence is decidable for all CF processes. 1995

Convert types to a grammar

LOHaskellCode $\sim$ 100

Soundness ✓

Prune unnormed productions

LOHaskellCode $\sim$ 30

Soundness ✓

1. Pruning unnormed productions is sound[a]

---

[a]Christensen, Huttel, and Stirling. Bisimulation equivalence is decidable for all context-free processes. 1995
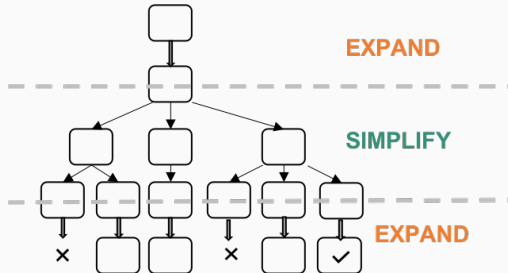

Simplify and expand

Simplify and expand
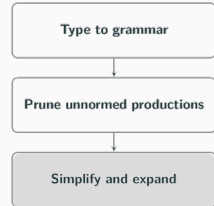
Following the ideas from Hirshfeld, Jančar, Moller, a bisimulation is seen as an **expansion tree**[23] that alternates between:

- expansion operations
- simplification operations

EXPAND

SIMPLIFY

EXPAND

---

[2] Hirshfeld. Bisimulation trees and the decidability of weak bisimulations. 1997
[3] Jančar and Moller. Techniques for decidability and undecidability of bisimilarity. 1999

Simplify and expand

An expansion tree alternates between:

- **expansion operations** - a single derived node results from the expansion of the parent node.

---

[4] Jančar, Moller. Techniques for decidability and undecidability of bisimilarity. 1999
[5] Christensen et al. Bisimulation equivalence is decidable for all CF processes. 1995

Simplify and expand

An expansion tree alternates between:

- **expansion operations** - a single derived node results from the expansion of the parent node.
- **simplification operations**
  - **reflexive rule**: omit from a node N any reflexive pair.
  - **congruence rule**: omit from a node N any pair that belongs to the least congruence containing the ancestors of N.

---

[4] Jančar, Moller. Techniques for decidability and undecidability of bisimilarity. 1999
[5] Christensen et al. Bisimulation equivalence is decidable for all CF processes. 1995

Simplify and expand

An expansion tree alternates between:

- **expansion operations** - a single derived node results from the expansion of the parent node.
- **simplification operations**
  - **reflexive rule**: omit from a node N any reflexive pair.
  - **congruence rule**: omit from a node N any pair that belongs to the least congruence containing the ancestors of N.
  - **basic process algebra rules**[4][5]

BPA1

$\cdots$
$\boxed{\cdots \ (X\alpha', Y\beta') \ \cdots}$ $\cdots$

$\boxed{\cdots \ (X\alpha, Y\beta) \ \cdots}$

$\boxed{\cdots \ (X\alpha, Y\beta) \ \cdots}$ $\boxed{\cdots \ (\alpha, \alpha'), (\beta, \beta') \cdots}$

BPA2

$\vdots$

$\boxed{\cdots \ (X\alpha, Y\beta) \ \cdots}$

$\boxed{\cdots \ (X\alpha, Y\beta) \ \cdots}$ $\boxed{\cdots (X, Y\gamma), (\gamma\alpha, \beta) \cdots}$

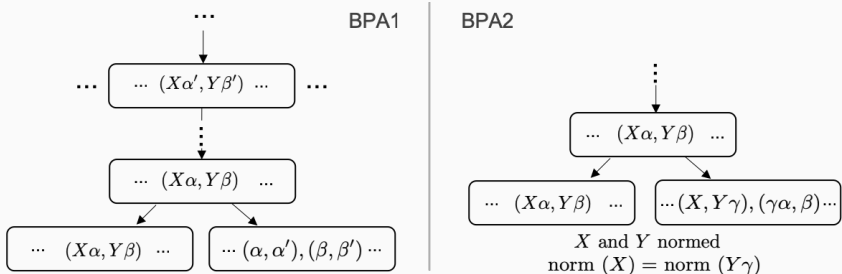X and Y normed
norm $(X) =$ norm $(Y\gamma)$

---

[4] Jančar, Moller. Techniques for decidability and undecidability of bisimilarity. 1999
[5] Christensen et al. Bisimulation equivalence is decidable for all CF processes. 1995

11

Simplify and Expand

All these transformation rules preserve the *safeness property*:

Safeness property[5]

$S \sim T$ iff the expansion tree rooted at $\{(S, T)\}$ has a successful branch.

The finite witness property holds[5]:

Finite witness property[5][6]

If $S \sim T$, then there exists a **finite successful branch** in the expansion tree.

---

[6]Christensen et al. Bisimulation equivalence is decidable for all CF processes. 1995

$$S \triangleq (\mu x.\&\{n: x; x; ?\,\mathsf{int}, \ell: ?\,\mathsf{int}\}); (\mu z.!\,\mathsf{int}; z; z)$$
$$T \triangleq (\mu y.\&\{n: y; y, \ell: \mathsf{skip}\}; ?\,\mathsf{int}); (\mu z.!\,\mathsf{int}; z)$$

$(X_1 X_4, Y_1 Y_3)$

Productions for *pruned S*

$X_1 \to \&n\, X_1 X_1 X_2$
$X_1 \to \&\ell\, X_3$
$X_2 \to ?\mathsf{int}$
$X_3 \to ?\mathsf{int}$
$X_4 \to !\mathsf{int}\, X_4$

Productions for T

$Y_1 \to \&n\, Y_1 Y_1 Y_2$
$Y_1 \to \&\ell\, Y_2$
$Y_2 \to ?\mathsf{int}$
$Y_3 \to !\mathsf{int}\, Y_3$

$$S \triangleq (\mu x.\&\{n\colon x; x; ?\,\mathsf{int}, \ell\colon ?\,\mathsf{int}\}); (\mu z.!\,\mathsf{int}; z; z)$$

$$T \triangleq (\mu y.\&\{n\colon y; y, \ell\colon \mathsf{skip}\}; ?\,\mathsf{int}); (\mu z.!\,\mathsf{int}; z)$$

Productions for *pruned S*

$X_1 \to \&n\, X_1 X_1 X_2$

$X_1 \to \&\ell\, X_3$

$X_2 \to ?\mathsf{int}$

$X_3 \to ?\mathsf{int}$

$X_4 \to !\mathsf{int}\, X_4$

Productions for *T*

$Y_1 \to \&n\, Y_1 Y_1 Y_2$

$Y_1 \to \&\ell\, Y_2$

$Y_2 \to ?\mathsf{int}$

$Y_3 \to !\mathsf{int}\, Y_3$

14

$$S \triangleq (\mu x.\&\{n\colon x; x; ?\,\mathsf{int}, \ell\colon ?\,\mathsf{int}\}); (\mu z.!\,\mathsf{int}; z; z)$$
$$T \triangleq (\mu y.\&\{n\colon y; y, \ell\colon \mathsf{skip}\}; ?\,\mathsf{int}); (\mu z.!\,\mathsf{int}; z)$$

Productions for *pruned S*

$X_1 \to \&n\, X_1 X_1 X_2$
$X_1 \to \&\ell\, X_3$
$X_2 \to ?\mathsf{int}$
$X_3 \to ?\mathsf{int}$
$X_4 \to !\mathsf{int}\, X_4$

Productions for T

$Y_1 \to \&n\, Y_1 Y_1 Y_2$
$Y_1 \to \&\ell\, Y_2$
$Y_2 \to ?\mathsf{int}$
$Y_3 \to !\mathsf{int}\, Y_3$

15

Example

$$S \triangleq (\mu x.\&\{n\colon x;\, x;\, ?\,\mathsf{int}, \ell\colon ?\,\mathsf{int}\});\, (\mu z.!\,\mathsf{int};\, z;\, z)$$
$$T \triangleq (\mu y.\&\{n\colon y;\, y,\, \ell\colon \mathsf{skip}\};\, ?\,\mathsf{int});\, (\mu z.!\,\mathsf{int};\, z)$$

$(X_1 X_4, Y_1 Y_3)$

bpa2   self

$(X_1, Y_1), (X_4, Y_3)$ $(X_1 X_4, Y_1 Y_3)$

expand   ⋮

$(X_1 X_1 X_2, Y_1 Y_1 Y_2), (X_3, Y_2), (X_4, Y_3)$

cong

$(X_1 X_1 X_2, Y_1 Y_1 Y_2), (X_3, Y_2)$

Productions for *pruned S*

$X_1 \rightarrow \&n\, X_1 X_1 X_2$
$X_1 \rightarrow \&\ell\, X_3$
$X_2 \rightarrow ?\mathsf{int}$
$X_3 \rightarrow ?\mathsf{int}$
$X_4 \rightarrow !\mathsf{int}\, X_4$

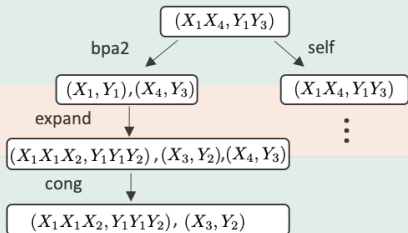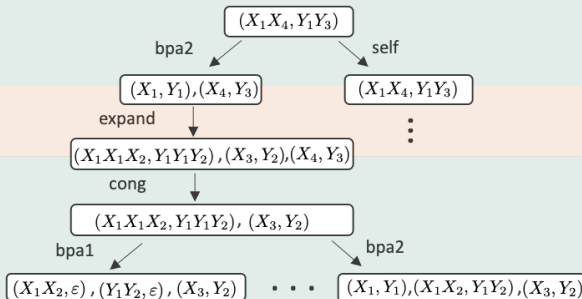Productions for *T*

$Y_1 \rightarrow \&n\, Y_1 Y_1 Y_2$
$Y_1 \rightarrow \&\ell\, Y_2$
$Y_2 \rightarrow ?\mathsf{int}$
$Y_3 \rightarrow !\mathsf{int}\, Y_3$

16

$$S \triangleq (\mu x.\&\{n\colon x; x; ?\,\mathrm{int}, \ell\colon ?\,\mathrm{int}\}); (\mu z.!\,\mathrm{int}; z; z)$$

$$T \triangleq (\mu y.\&\{n\colon y; y, \ell\colon \mathrm{skip}\}; ?\,\mathrm{int}); (\mu z.!\,\mathrm{int}; z)$$

Example

Productions for *pruned S*

$X_1 \to \&n\,X_1 X_1 X_2$

$X_1 \to \&\ell\,X_3$

$X_2 \to ?\mathrm{int}$

$X_3 \to ?\mathrm{int}$

$X_4 \to !\mathrm{int}\,X_4$

Productions for *T*

$Y_1 \to \&n\,Y_1 Y_1 Y_2$

$Y_1 \to \&\ell\,Y_2$

$Y_2 \to ?\mathrm{int}$

$Y_3 \to !\mathrm{int}\,Y_3$

Diagram nodes:

$(X_1 X_4, Y_1 Y_3)$

bpa2 → $(X_1, Y_1), (X_4, Y_3)$

self → $(X_1 X_4, Y_1 Y_3)$ ⋮

expand → $(X_1 X_1 X_2, Y_1 Y_1 Y_2), (X_3, Y_2), (X_4, Y_3)$

cong → $(X_1 X_1 X_2, Y_1 Y_1 Y_2), (X_3, Y_2)$

bpa1 → $(X_1 X_2, \varepsilon), (Y_1 Y_2, \varepsilon), (X_3, Y_2)$ · · ·

bpa2 → $(X_1, Y_1), (X_1 X_2, Y_1 Y_2), (X_3, Y_2)$

17

Example

$$S \triangleq (\mu x. \&\{n \colon x; x, \ell \colon ? \text{int}\}); (\mu z. ! \text{int}; z; z)$$
$$T \triangleq (\mu y. \&\{n \colon y; y, \ell \colon \text{skip}\}; ? \text{int}); (\mu z. ! \text{int}; z)$$

Productions for *pruned S*

$X_1 \rightarrow \&n\, X_1 X_1 X_2$
$X_1 \rightarrow \&\ell\, X_3$
$X_2 \rightarrow ?\text{int}$
$X_3 \rightarrow ?\text{int}$
$X_4 \rightarrow !\text{int}\, X_4$
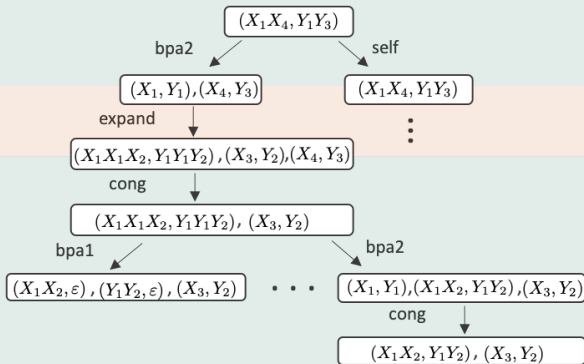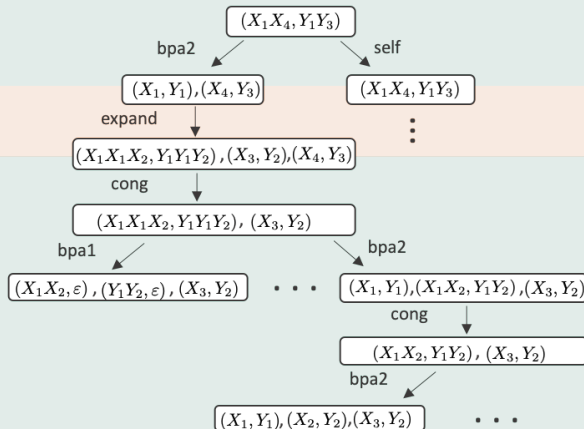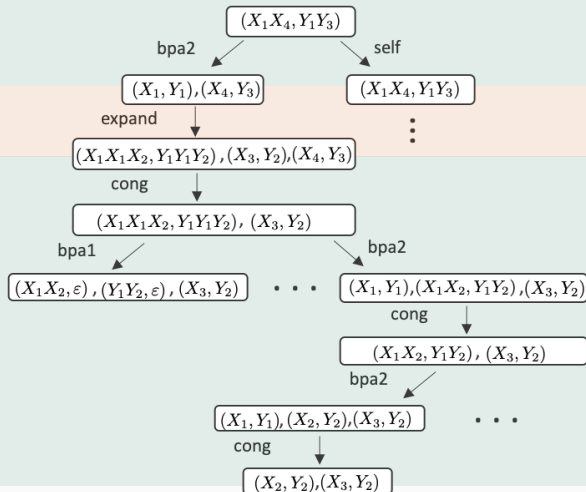
Productions for T

$Y_1 \rightarrow \&n\, Y_1 Y_1 Y_2$
$Y_1 \rightarrow \&\ell\, Y_2$
$Y_2 \rightarrow ?\text{int}$
$Y_3 \rightarrow !\text{int}\, Y_3$

$$S \triangleq (\mu x.\&\{n\colon x; x; ?\,\mathsf{int}, \ell\colon ?\,\mathsf{int}\}); (\mu z.!\,\mathsf{int}; z; z)$$

Example

$$T \triangleq (\mu y.\&\{n\colon y; y, \ell\colon \mathsf{skip}\}; ?\,\mathsf{int}); (\mu z.!\,\mathsf{int}; z)$$

$(X_1 X_4, Y_1 Y_3)$

bpa2   self

$(X_1, Y_1), (X_4, Y_3)$ $(X_1 X_4, Y_1 Y_3)$

expand $\vdots$

$(X_1 X_1 X_2, Y_1 Y_1 Y_2), (X_3, Y_2), (X_4, Y_3)$

cong

$(X_1 X_1 X_2, Y_1 Y_1 Y_2), (X_3, Y_2)$

bpa1   bpa2

$(X_1 X_2, \varepsilon), (Y_1 Y_2, \varepsilon), (X_3, Y_2)$ $\cdots$ $(X_1, Y_1), (X_1 X_2, Y_1 Y_2), (X_3, Y_2)$

cong

$(X_1 X_2, Y_1 Y_2), (X_3, Y_2)$

bpa2

$(X_1, Y_1), (X_2, Y_2), (X_3, Y_2)$ $\cdots$

Productions for *pruned S*

$X_1 \to \&n\, X_1 X_1 X_2$
$X_1 \to \&\ell\, X_3$
$X_2 \to ?\mathsf{int}$
$X_3 \to ?\mathsf{int}$
$X_4 \to !\mathsf{int}\, X_4$

Productions for T

$Y_1 \to \&n\, Y_1 Y_1 Y_2$
$Y_1 \to \&\ell\, Y_2$
$Y_2 \to ?\mathsf{int}$
$Y_3 \to !\mathsf{int}\, Y_3$

$$S \triangleq (\mu x.\&\{n: x; x; ?\,\text{int}, \ell: ?\,\text{int}\}); (\mu z.!\,\text{int}; z; z)$$
$$T \triangleq (\mu y.\&\{n: y; y, \ell: \text{skip}\}; ?\,\text{int}); (\mu z.!\,\text{int}; z)$$

Example

$(X_1 X_4, Y_1 Y_3)$

bpa2 — self

$(X_1, Y_1), (X_4, Y_3)$ $(X_1 X_4, Y_1 Y_3)$

expand $\vdots$

$(X_1 X_1 X_2, Y_1 Y_1 Y_2), (X_3, Y_2), (X_4, Y_3)$

cong

$(X_1 X_1 X_2, Y_1 Y_1 Y_2), (X_3, Y_2)$

bpa1   bpa2

$(X_1 X_2, \varepsilon), (Y_1 Y_2, \varepsilon), (X_3, Y_2)$ $\cdots$ $(X_1, Y_1), (X_1 X_2, Y_1 Y_2), (X_3, Y_2)$

cong

$(X_1 X_2, Y_1 Y_2), (X_3, Y_2)$

bpa2

$(X_1, Y_1), (X_2, Y_2), (X_3, Y_2)$ $\cdots$

cong

$(X_2, Y_2), (X_3, Y_2)$

Productions for _pruned S_

$X_1 \rightarrow \&n\, X_1 X_1 X_2$
$X_1 \rightarrow \&\ell\, X_3$
$X_2 \rightarrow ?\text{int}$
$X_3 \rightarrow ?\text{int}$
$X_4 \rightarrow !\text{int}\, X_4$

Productions for _T_

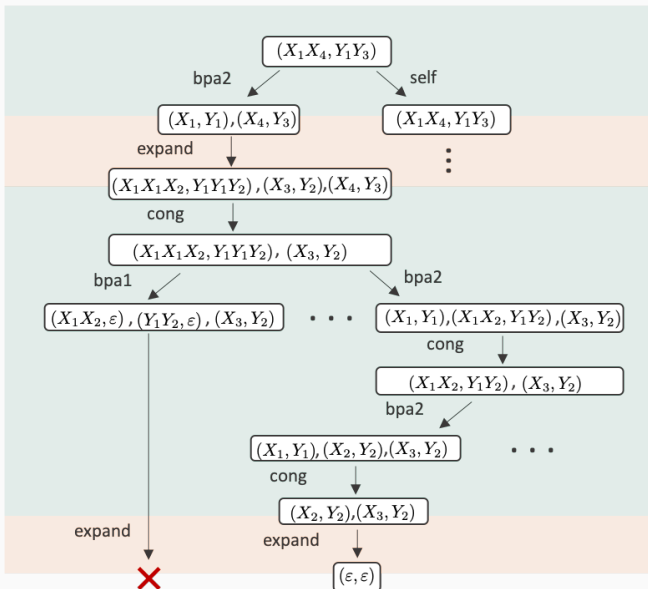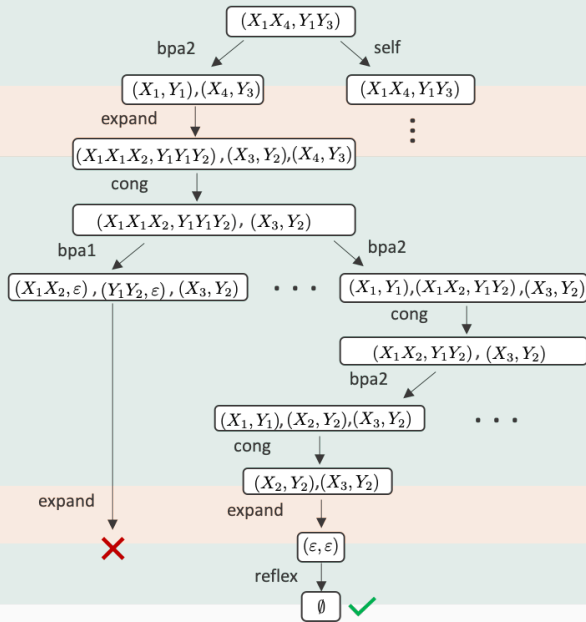$Y_1 \rightarrow \&n\, Y_1 Y_1 Y_2$
$Y_1 \rightarrow \&\ell\, Y_2$
$Y_2 \rightarrow ?\text{int}$
$Y_3 \rightarrow !\text{int}\, Y_3$

Example

$$S \triangleq (\mu x.\&\{n: x; x; ?\,\text{int}, \ell: ?\,\text{int}\}); (\mu z.!\,\text{int}; z; z)$$
$$T \triangleq (\mu y.\&\{n: y; y, \ell: \text{skip}\}; ?\,\text{int}); (\mu z.!\,\text{int}; z)$$

Productions for *pruned S*

$X_1 \to \&n\, X_1 X_1 X_2$
$X_1 \to \&\ell\, X_3$
$X_2 \to ?\text{int}$
$X_3 \to ?\text{int}$
$X_4 \to !\text{int}\, X_4$

Productions for T

$Y_1 \to \&n\, Y_1 Y_1 Y_2$
$Y_1 \to \&\ell\, Y_2$
$Y_2 \to ?\text{int}$
$Y_3 \to !\text{int}\, Y_3$

21

Example

$$S \triangleq (\mu x.\&\{n\colon x; x; ?\,\text{int}, \ell\colon ?\,\text{int}\}); (\mu z.!\,\text{int}; z; z)$$
$$T \triangleq (\mu y.\&\{n\colon y; y, \ell\colon \text{skip}\}; ?\,\text{int}); (\mu z.!\,\text{int}; z)$$

Productions for *pruned S*

$X_1 \to \&n\,X_1 X_1 X_2$
$X_1 \to \&\ell\,X_3$
$X_2 \to ?\text{int}$
$X_3 \to ?\text{int}$
$X_4 \to !\text{int}\,X_4$

Productions for *T*

$Y_1 \to \&n\,Y_1 Y_1 Y_2$
$Y_1 \to \&\ell\,Y_2$
$Y_2 \to ?\text{int}$
$Y_3 \to !\text{int}\,Y_3$

22

Convert types to a grammar

`LOHaskellCode` $\sim$ 100

Soundness ✓

Prune unnormed productions

`LOHaskellCode` $\sim$ 30

Soundness ✓

Simplify and expand

`LOHaskellCode` $\sim$ 170

Soundness ✓

1. The *safeness property ensures soundness*[a]

---

[a]Jančar, Moller. Techniques for decidability and undecidability of bisimilarity. 1999

Implementation strategies

Implementation choice:

- Breadth-first search on the tree

Implementation strategies

Implementation choice:

- Breadth-first search on the tree

Strategic options that can enhance performance:

- Instead of looking for a fixed point, iterate the simplification phase
- Apply BPA rules wrapped with blocks of reflexive and congruence rules

Implementation strategies

Implementation choice:
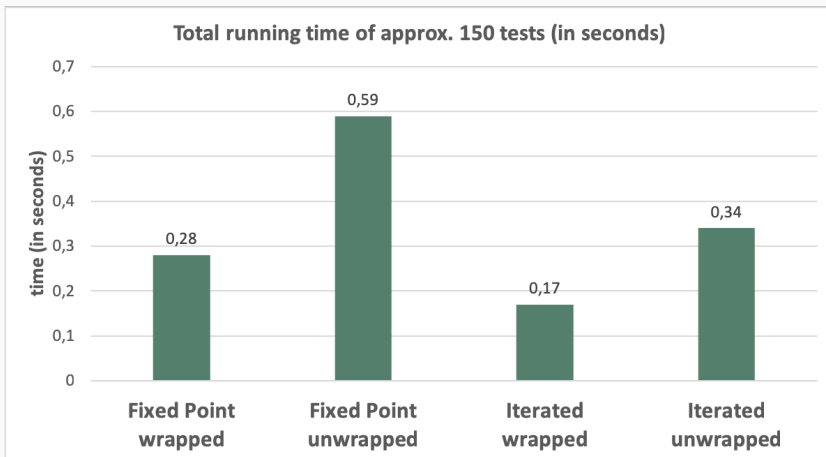
- Breadth-first search on the tree

Strategic options that can enhance performance:

- Instead of looking for a fixed point, iterate the simplification phase
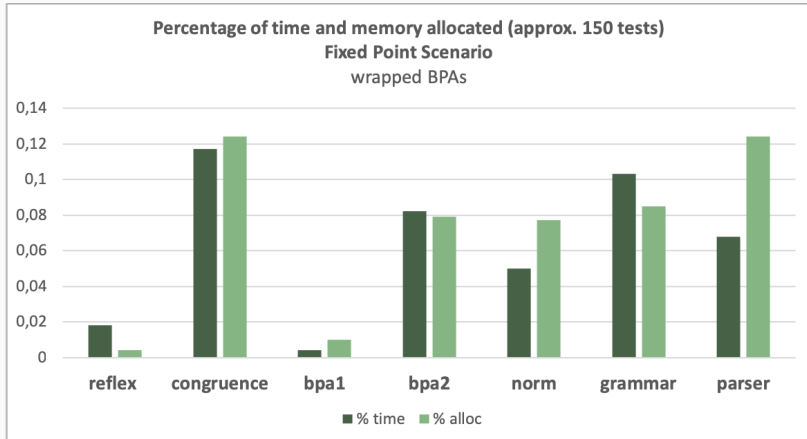- Apply BPA rules wrapped with blocks of reflexive and congruence rules

We showcase runtimes for four scenarios:

	Fixed Point Scenario		Iterated Scenario	
	wrapped	unwrapped	wrapped	unwrapped
Simplification	find fixed point		iterate	
BPAs wrapped	✓	✗	✓	✗

Running times ($\sim$ 150 tests)

Total running time of approx. 150 tests (in seconds)

Bar chart, time (in seconds):
- Fixed Point wrapped: 0,28
- Fixed Point unwrapped: 0,59
- Iterated wrapped: 0,17
- Iterated unwrapped: 0,34

Specs: MacBook Air
        1.8 GHz Intel Core i5
        8GB of memory

Towards completeness...

Finite witness property[7][8]

If $S \sim T$, then there exists a **finite successful branch** in the expansion tree.

Implementation choices aiming to achieve completeness:

- Instead of looking for a fixed point, iterate the simplification phase (there may not be a fixed point)

- Double ended enqueue, prepending *promising* nodes, as opposed to queuing all new nodes

---

[7]Christensen et al. Bisimulation equivalence is decidable for all CF processes. 1995
[8]Jančar, Moller. Techniques for decidability and undecidability of bisimilarity. 1999

Ongoing Work

- Soundness ✓
- Completeness ? ... on our way to achieve it.
- Complexity ? ... in practice it does not seem to take much longer than parsing.
- Lines of Haskell code: approx.\ 300

Coming soon:

FreeST, a compiler for context-free session types!

(demos on demand)

Thank you!