

Checking the equivalence of context-free session types

Andreia Mordido and Vasco T. Vasconcelos

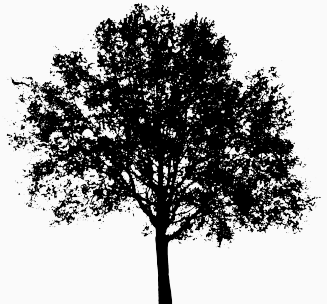
LASIGE, Faculdade de Ciências, Universidade de Lisboa

January 2019

Motivation

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

```
sendTree Leaf c =  
  select Leaf c  
sendTree (Node x l r) c =  
  let c1 = select Node c  
      c2 = send x c1  
      c3 = sendTree l c2  
      c4 = sendTree r c3  
  in c4
```

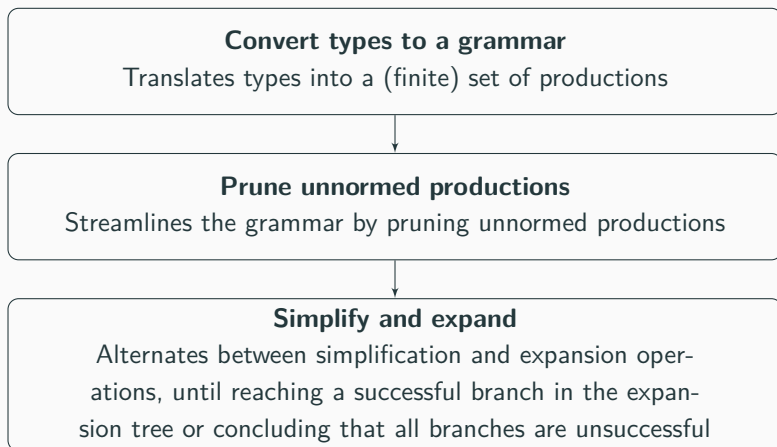


Definition (Type equivalence problem)

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.

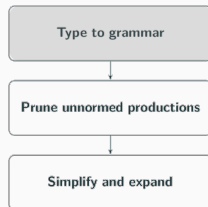
Algorithm for checking the equivalence of CFST **Main stages**



Convert types to a grammar

We consider a finite set of productions $X \xrightarrow{a} \vec{Y}$ 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.:

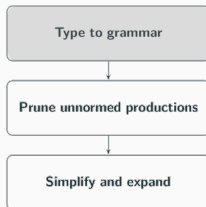
- context-free grammars in *Greibach normal form*
- s.t. for each X and a there is at most one production $X \xrightarrow{a} \vec{Y}$

Convert types to a grammar

Example

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

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



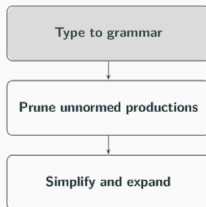
Convert types to a grammar

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 S	Productions for T
$X_1 \xrightarrow{\&n} X_1 X_1 X_2$	$Y_1 \xrightarrow{\&n} Y_1 Y_1 Y_2$
$X_1 \xrightarrow{\&\ell} X_3$	$Y_1 \xrightarrow{\&\ell} Y_2$
$X_2 \xrightarrow{? \text{int}} \varepsilon$	$Y_2 \xrightarrow{? \text{int}} \varepsilon$
$X_3 \xrightarrow{? \text{int}} \varepsilon$	$Y_3 \xrightarrow{! \text{int}} Y_3$
$X_4 \xrightarrow{! \text{int}} X_4 X_4$	



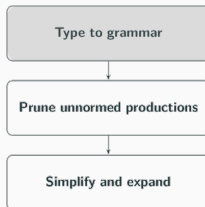
Convert types to a grammar

Example

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

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

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



Algorithm for checking the equivalence of CFST **Main stages**

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.

^aThiemann and Vasconcelos. Context-free session types. 2016.

^bBaeten et al. Decidability of bisimulation equivalence for process generating context-free languages. 1993.



Prune unnormed productions



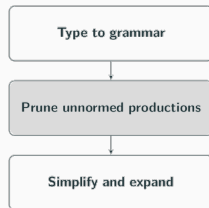
Simplify and expand

Prune unnormed productions

Definition ((Un)normed terms¹)

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

Christensen, Huttel, and Stirling¹ noted that:
whenever α is unnormed, $\alpha \sim \alpha\beta$.



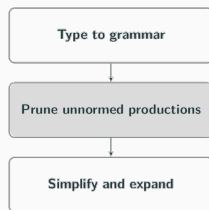
¹ Christensen et al. Bisimulation equivalence is decidable for all CF processes. 1995

Prune unnormed productions

Definition ((Un)normed terms¹)

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

Christensen, Huttel, and Stirling¹ noted that:
whenever α is unnormed, $\alpha \sim \alpha\beta$.



Example

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

Productions for S

$$X_1 \xrightarrow{\&n} X_1 X_1 X_2$$

$$X_1 \xrightarrow{\&\ell} X_3$$

$$X_2 \xrightarrow{?\text{int}} \varepsilon$$

$$X_3 \xrightarrow{?\text{int}} \varepsilon$$

$$X_4 \xrightarrow{!\text{int}} X_4 X_4$$

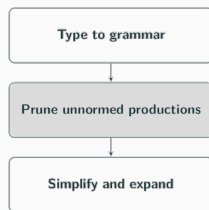
¹ Christensen et al. Bisimulation equivalence is decidable for all CF processes. 1995

Prune unnormed productions

Definition ((Un)normed terms¹)

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

Christensen, Huttel, and Stirling¹ noted that:
whenever α is unnormed, $\alpha \sim \alpha\beta$.



Example

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

Productions for S

$$X_1 \xrightarrow{\&n} X_1 X_1 X_2$$

$$X_1 \xrightarrow{\&\ell} X_3$$

$$X_2 \xrightarrow{?\text{int}} \varepsilon$$

$$X_3 \xrightarrow{?\text{int}} \varepsilon$$

$$X_4 \xrightarrow{!\text{int}} X_4 X_4$$

Productions for *pruned* S

$$X_1 \xrightarrow{\&n} X_1 X_1 X_2$$

$$X_1 \xrightarrow{\&\ell} X_3$$

$$X_2 \xrightarrow{?\text{int}} \varepsilon$$

$$X_3 \xrightarrow{?\text{int}} \varepsilon$$

$$X_4 \xrightarrow{!\text{int}} X_4$$

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

Algorithm for checking the equivalence of CFST Main stages

Convert types to a grammar

LOHaskellCode ~ 100

Soundness ✓



Prune unnormed productions

LOHaskellCode ~ 30

Soundness ✓

1. Pruning unnormed productions is sound^a

^aChristensen, Huttel, and Stirling. Bisimulation equivalence is decidable for all context-free processes. 1995

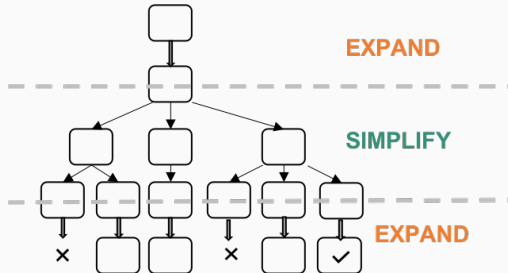
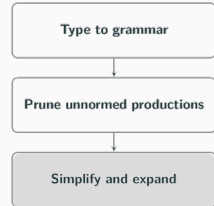


Simplify and expand

Simplify and expand

Following the ideas from Hirshfeld, Jančar, Moller, a bisimulation between processes is seen as an **expansion tree**²³, that alternates between:

- expansion operations
- simplification operations



²Hirshfeld. Bisimulation trees and the decidability of weak bisimulations. 1997

³Jančar 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.

⁴Jančar, Moller. Techniques for decidability and undecidability of bisimilarity. 1999

⁵Christensen 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 .

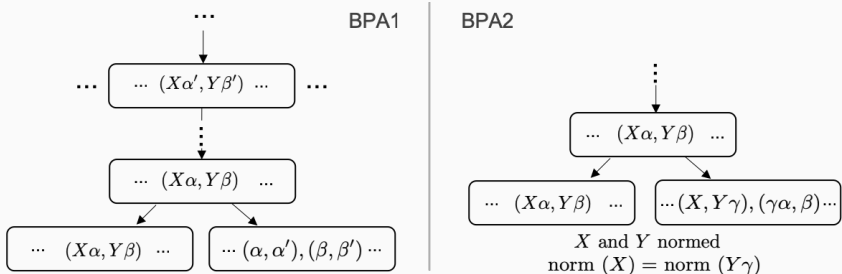
⁴Jančar, Moller. Techniques for decidability and undecidability of bisimilarity. 1999

⁵Christensen 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**⁴⁵



⁴Jančar, Moller. Techniques for decidability and undecidability of bisimilarity. 1999

⁵Christensen et al. Bisimulation equivalence is decidable for all CF processes. 1995

Simplify and Expand

All these transformation rules preserve the *safeness property*:

Safeness property⁵

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

The finite witness property holds⁵:

Finite witness property⁵⁶

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

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

Example

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

$(X_1 X_4, Y_1 Y_3)$

Productions for *pruned S*

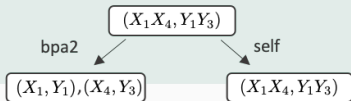
$$X_1 \xrightarrow{\&n} X_1 X_1 X_2$$
$$X_1 \xrightarrow{\&\ell} X_3$$
$$X_2 \xrightarrow{? \text{int}} \varepsilon$$
$$X_3 \xrightarrow{? \text{int}} \varepsilon$$
$$X_4 \xrightarrow{! \text{int}} X_4$$

Productions for *T*

$$Y_1 \xrightarrow{\&n} Y_1 Y_1 Y_2$$
$$Y_1 \xrightarrow{\&\ell} Y_2$$
$$Y_2 \xrightarrow{? \text{int}} \varepsilon$$
$$Y_3 \xrightarrow{! \text{int}} Y_3$$

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*

$$X_1 \xrightarrow{\&n} X_1X_1X_2$$

$$X_1 \xrightarrow{\&\ell} X_3$$

$$X_2 \xrightarrow{? \text{int}} \varepsilon$$

$$X_3 \xrightarrow{? \text{int}} \varepsilon$$

$$X_4 \xrightarrow{! \text{int}} X_4$$

Productions for *T*

$$Y_1 \xrightarrow{\&n} Y_1Y_1Y_2$$

$$Y_1 \xrightarrow{\&\ell} Y_2$$

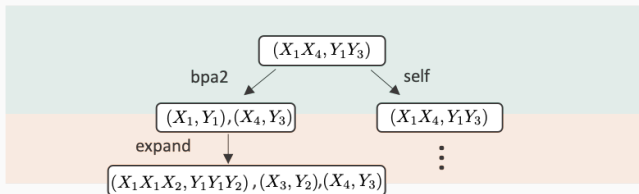
$$Y_2 \xrightarrow{? \text{int}} \varepsilon$$

$$Y_3 \xrightarrow{! \text{int}} Y_3$$

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*

$$X_1 \xrightarrow{\&n} X_1X_1X_2$$

$$X_1 \xrightarrow{\&\ell} X_3$$

$$X_2 \xrightarrow{? \text{int}} \varepsilon$$

$$X_3 \xrightarrow{? \text{int}} \varepsilon$$

$$X_4 \xrightarrow{! \text{int}} X_4$$

Productions for *T*

$$Y_1 \xrightarrow{\&n} Y_1Y_1Y_2$$

$$Y_1 \xrightarrow{\&\ell} Y_2$$

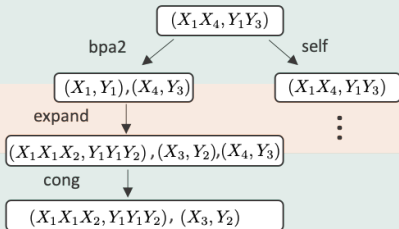
$$Y_2 \xrightarrow{? \text{int}} \varepsilon$$

$$Y_3 \xrightarrow{! \text{int}} Y_3$$

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*

$$X_1 \xrightarrow{\&n} X_1X_1X_2$$

$$X_1 \xrightarrow{\&\ell} X_3$$

$$X_2 \xrightarrow{? \text{int}} \varepsilon$$

$$X_3 \xrightarrow{? \text{int}} \varepsilon$$

$$X_4 \xrightarrow{! \text{int}} X_4$$

Productions for *T*

$$Y_1 \xrightarrow{\&n} Y_1Y_1Y_2$$

$$Y_1 \xrightarrow{\&\ell} Y_2$$

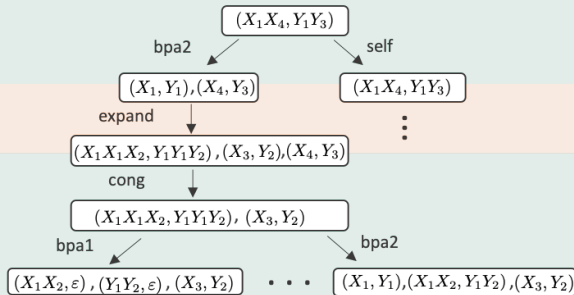
$$Y_2 \xrightarrow{? \text{int}} \varepsilon$$

$$Y_3 \xrightarrow{! \text{int}} Y_3$$

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*

$$X_1 \xrightarrow{\&n} X_1X_1X_2$$

$$X_1 \xrightarrow{\&\ell} X_3$$

$$X_2 \xrightarrow{? \text{int}} \varepsilon$$

$$X_3 \xrightarrow{? \text{int}} \varepsilon$$

$$X_4 \xrightarrow{! \text{int}} X_4$$

Productions for *T*

$$Y_1 \xrightarrow{\&n} Y_1Y_1Y_2$$

$$Y_1 \xrightarrow{\&\ell} Y_2$$

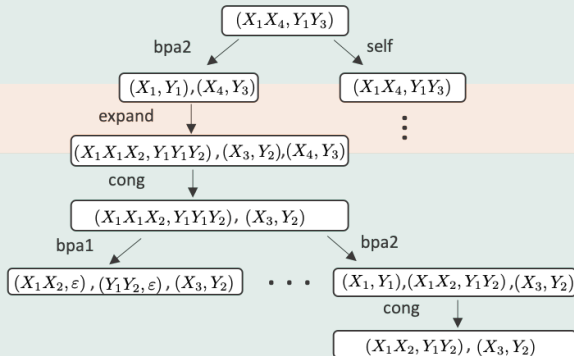
$$Y_2 \xrightarrow{? \text{int}} \varepsilon$$

$$Y_3 \xrightarrow{! \text{int}} Y_3$$

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*

$$X_1 \xrightarrow{\&n} X_1X_1X_2$$

$$X_1 \xrightarrow{\&\ell} X_3$$

$$X_2 \xrightarrow{? \text{int}} \varepsilon$$

$$X_3 \xrightarrow{? \text{int}} \varepsilon$$

$$X_4 \xrightarrow{! \text{int}} X_4$$

Productions for *T*

$$Y_1 \xrightarrow{\&n} Y_1Y_1Y_2$$

$$Y_1 \xrightarrow{\&\ell} Y_2$$

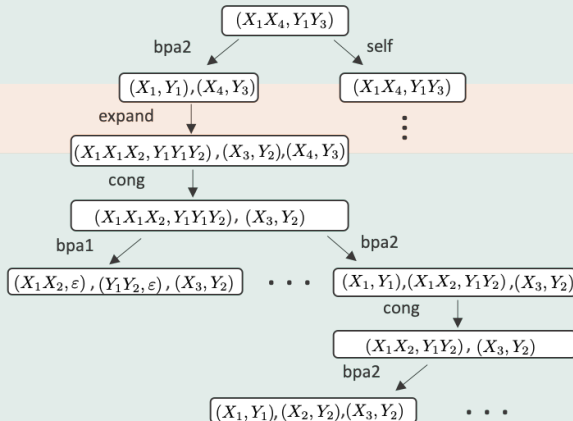
$$Y_2 \xrightarrow{? \text{int}} \varepsilon$$

$$Y_3 \xrightarrow{! \text{int}} Y_3$$

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*

$$X_1 \xrightarrow{\&n} X_1X_1X_2$$

$$X_1 \xrightarrow{\&\ell} X_3$$

$$X_2 \xrightarrow{? \text{int}} \varepsilon$$

$$X_3 \xrightarrow{? \text{int}} \varepsilon$$

$$X_4 \xrightarrow{! \text{int}} X_4$$

Productions for T

$$Y_1 \xrightarrow{\&n} Y_1Y_1Y_2$$

$$Y_1 \xrightarrow{\&\ell} Y_2$$

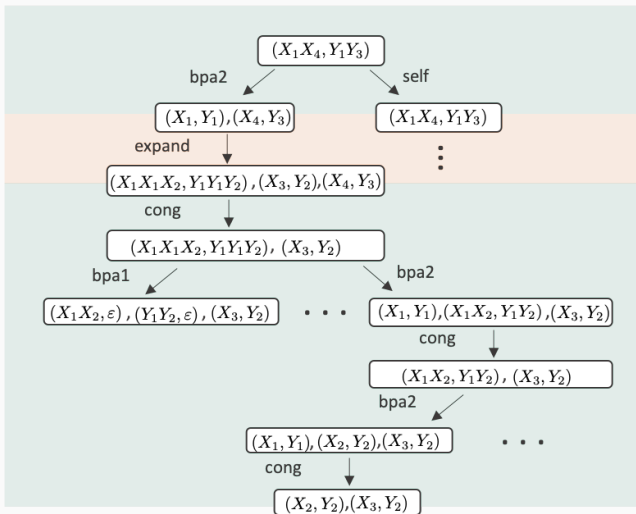
$$Y_2 \xrightarrow{? \text{int}} \varepsilon$$

$$Y_3 \xrightarrow{! \text{int}} Y_3$$

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*

$$X_1 \xrightarrow{\&n} X_1X_1X_2$$

$$X_1 \xrightarrow{\&\ell} X_3$$

$$X_2 \xrightarrow{? \text{int}} \varepsilon$$

$$X_3 \xrightarrow{? \text{int}} \varepsilon$$

$$X_4 \xrightarrow{! \text{int}} X_4$$

Productions for T

$$Y_1 \xrightarrow{\&n} Y_1Y_1Y_2$$

$$Y_1 \xrightarrow{\&\ell} Y_2$$

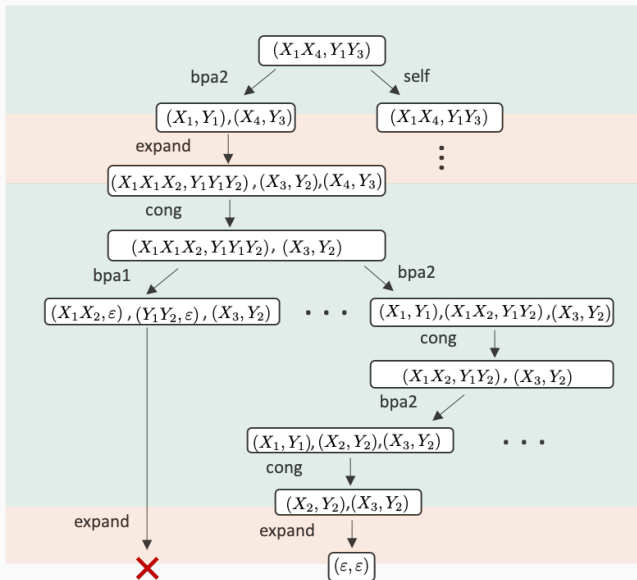
$$Y_2 \xrightarrow{? \text{int}} \varepsilon$$

$$Y_3 \xrightarrow{! \text{int}} Y_3$$

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*

$$X_1 \xrightarrow{\&n} X_1X_1X_2$$

$$X_1 \xrightarrow{\&\ell} X_3$$

$$X_2 \xrightarrow{? \text{int}} \varepsilon$$

$$X_3 \xrightarrow{? \text{int}} \varepsilon$$

$$X_4 \xrightarrow{! \text{int}} X_4$$

Productions for T

$$Y_1 \xrightarrow{\&n} Y_1Y_1Y_2$$

$$Y_1 \xrightarrow{\&\ell} Y_2$$

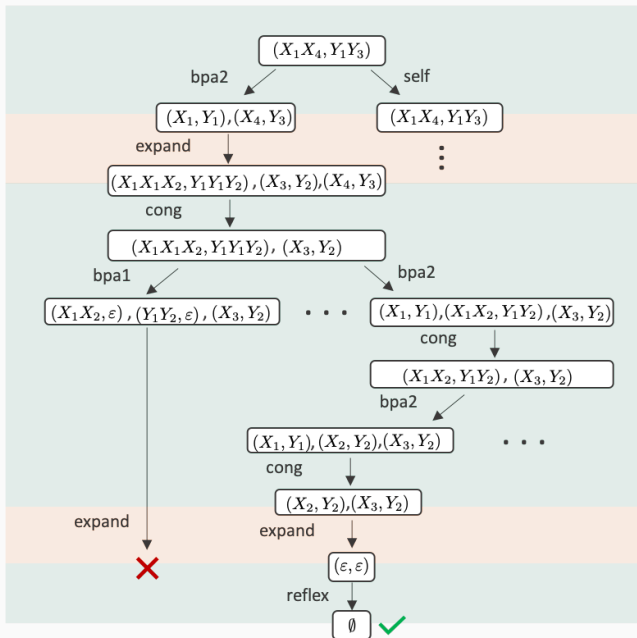
$$Y_2 \xrightarrow{? \text{int}} \varepsilon$$

$$Y_3 \xrightarrow{! \text{int}} Y_3$$

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

$$X_1 \xrightarrow{\&n} X_1X_1X_2$$

$$X_1 \xrightarrow{\&\ell} X_3$$

$$X_2 \xrightarrow{? \text{int}} \varepsilon$$

$$X_3 \xrightarrow{? \text{int}} \varepsilon$$

$$X_4 \xrightarrow{! \text{int}} X_4$$

Productions for T

$$Y_1 \xrightarrow{\&n} Y_1Y_1Y_2$$

$$Y_1 \xrightarrow{\&\ell} Y_2$$

$$Y_2 \xrightarrow{? \text{int}} \varepsilon$$

$$Y_3 \xrightarrow{! \text{int}} Y_3$$

Algorithm for checking the equivalence of CFST Main stages

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

^aJančar, 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

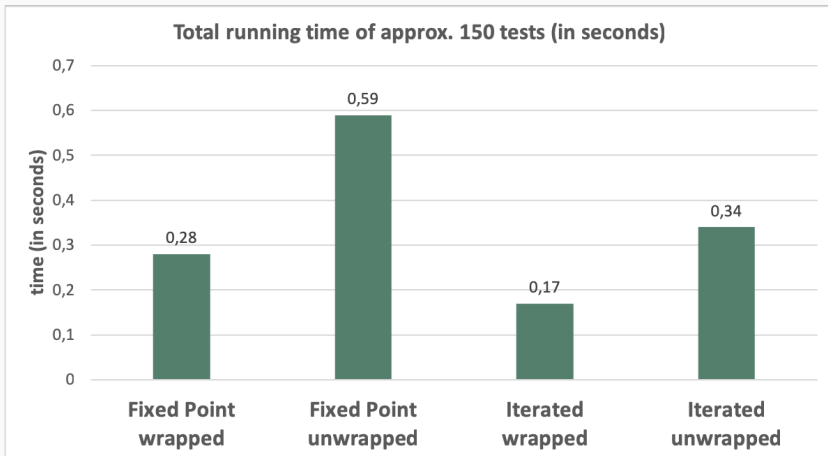
Strategic options that can enhance performance:

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

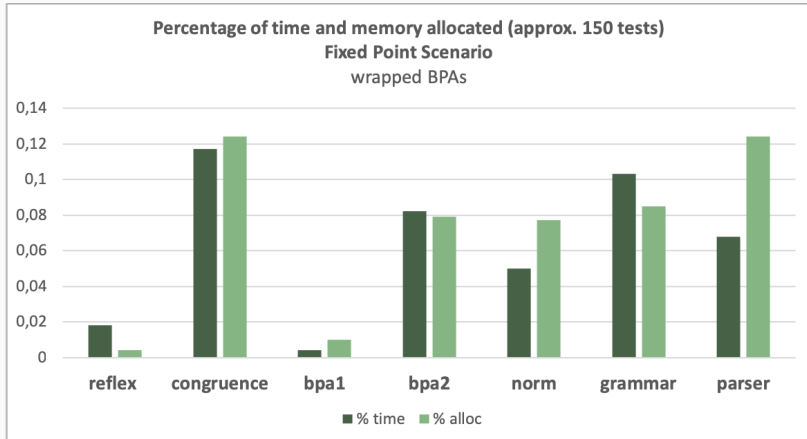
We showcase runtimes for four scenarios:

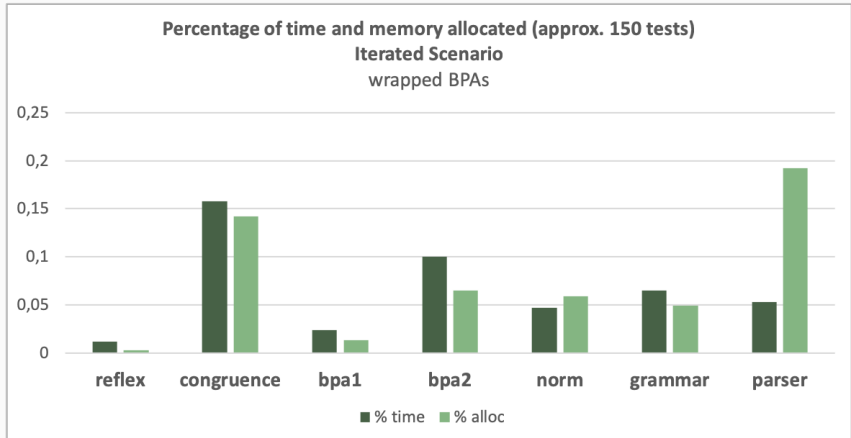
	Fixed Point Scenario		Iterated Scenario	
	wrapped	unwrapped	wrapped	unwrapped
Simplification	find fixed point		iterate	
BPAs wrapped	✓	×	✓	×

Running times (~ 150 tests)



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





Towards completeness...

Finite witness property⁷⁸

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

⁷Christensen et al. Bisimulation equivalence is decidable for all CF processes. 1995

⁸Jančar, Moller. Techniques for decidability and undecidability of bisimilarity. 1999

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

Coming soon:

FreeST, a compiler for context-free session types!
(demos on demand)

Thank you!