Böhm Reduction for Terms and Term Graphs

Confluence in Infinitary Rewriting

Patrick Bahr

IT University of Copenhagen

Infinitary Term Rewriting

Assign outcome to (well-formed) infinite reductions.

Infinitary Term Rewriting

Assign outcome to (well-formed) infinite reductions.

Example

$$\mathit{from}(x) \to x :: \mathit{from}(s(x))$$

Infinitary Term Rewriting

Assign outcome to (well-formed) infinite reductions.

Example

$$\mathit{from}(x) \to x :: \mathit{from}(s(x))$$

$\mathit{from}(0)$

Infinitary Term Rewriting

Assign outcome to (well-formed) infinite reductions.

Example

$$\mathit{from}(x) \to x :: \mathit{from}(s(x))$$

$$\mathit{from}(0) \to 0 :: \mathit{from}(1)$$

Infinitary Term Rewriting

Assign outcome to (well-formed) infinite reductions.

Example

$$\mathit{from}(x) \to x :: \mathit{from}(s(x))$$

$$\begin{aligned} \mathit{from}(0) &\to 0 :: \mathit{from}(1) \\ &\to 0 :: 1 :: \mathit{from}(2) \end{aligned}$$

Infinitary Term Rewriting

Assign outcome to (well-formed) infinite reductions.

Example

$$\mathit{from}(x) \to x :: \mathit{from}(s(x))$$

$$\begin{aligned} \mathit{from}(0) &\to 0 :: \mathit{from}(1) \\ &\to 0 :: 1 :: \mathit{from}(2) \\ &\to 0 :: 1 :: 2 :: \mathit{from}(3) \end{aligned}$$

Infinitary Term Rewriting

Assign outcome to (well-formed) infinite reductions.

Example

$$\mathit{from}(x) \to x :: \mathit{from}(s(x))$$

$$\begin{aligned}
\mathit{from}(0) &\to 0 :: \mathit{from}(1) \\
&\to 0 :: 1 :: \mathit{from}(2) \\
&\to 0 :: 1 :: 2 :: \mathit{from}(3) \\
&\to 0 :: 1 :: 2 :: 3 :: \mathit{from}(4)
\end{aligned}$$

Infinitary Term Rewriting

Assign outcome to (well-formed) infinite reductions.

Example

$$\mathit{from}(x) \to x :: \mathit{from}(s(x))$$

$$\begin{aligned} \mathit{from}(0) &\to 0 :: \mathit{from}(1) \\ &\to 0 :: 1 :: \mathit{from}(2) \\ &\to 0 :: 1 :: 2 :: \mathit{from}(3) \\ &\to 0 :: 1 :: 2 :: 3 :: \mathit{from}(4) \\ &\to \dots \end{aligned}$$

Infinitary Term Rewriting

Assign outcome to (well-formed) infinite reductions.

Example

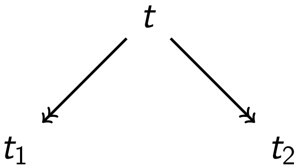
$$\mathit{from}(x) \to x :: \mathit{from}(s(x))$$

$$\begin{aligned} \mathit{from}(0) &\to 0 :: \mathit{from}(1) \\ &\to 0 :: 1 :: \mathit{from}(2) \\ &\to 0 :: 1 :: 2 :: \mathit{from}(3) \\ &\to 0 :: 1 :: 2 :: 3 :: \mathit{from}(4) \\ &\to \dots \end{aligned}$$

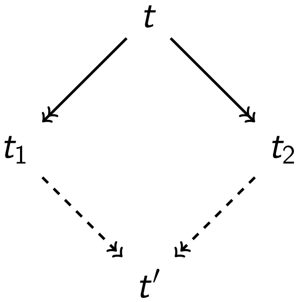
intuitively this converges to the infinite list

$0 :: 1 :: 2 :: 3 :: 4 :: 5 :: \dots$.

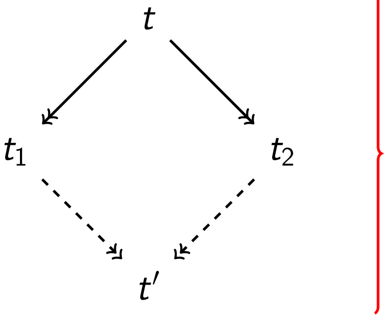
Confluence in Infinitary Rewriting



Confluence in Infinitary Rewriting

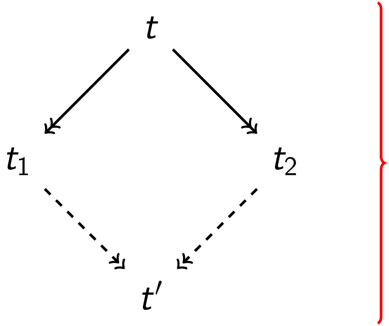


Confluence in Infinitary Rewriting



infinite reductions

Confluence in Infinitary Rewriting



infinite reductions

Infinitary Confluence Breaks for

- orthogonal term rewriting systems
- lambda calculus

Counter Example

for Orthogonal Term Rewriting Systems

$$f(x) \to x \qquad\qquad g(x) \to x$$

$$f \downarrow g \downarrow f \downarrow g \downarrow f \downarrow g \;\vdots \quad \Big\} \; f(g(f(g(f(g(\ldots))))))$$

Counter Example

for Orthogonal Term Rewriting Systems

$f(x) \to x$ $\qquad$ $g(x) \to x$

Counter Example

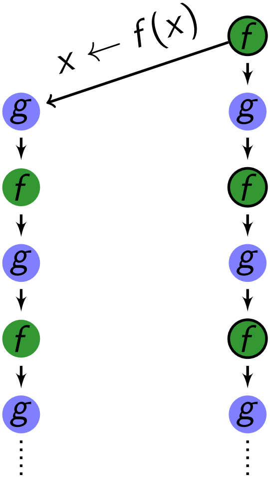
for Orthogonal Term Rewriting Systems

$f(x) \to x \qquad g(x) \to x$

Counter Example

for Orthogonal Term Rewriting Systems

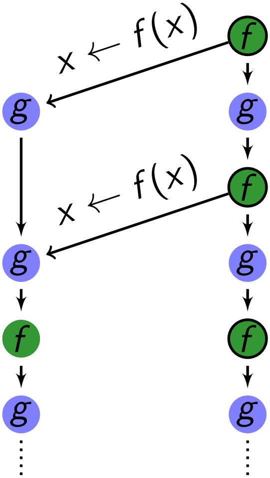
$$f(x) \to x \qquad g(x) \to x$$



Counter Example

for Orthogonal Term Rewriting Systems

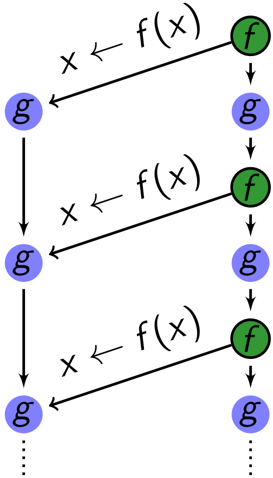
$f(x) \to x$ $\qquad$ $g(x) \to x$



Counter Example

for Orthogonal Term Rewriting Systems

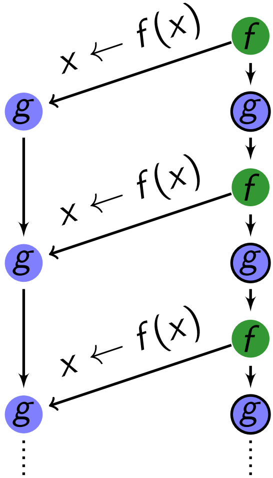
$f(x) \to x \qquad g(x) \to x$



Counter Example

for Orthogonal Term Rewriting Systems

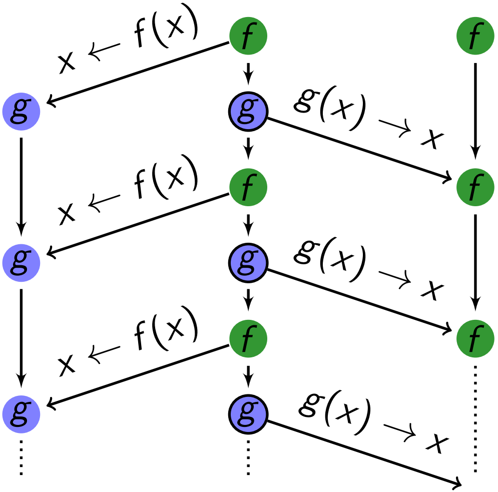
$f(x) \to x$ $\quad\quad$ $g(x) \to x$



Counter Example

for Orthogonal Term Rewriting Systems

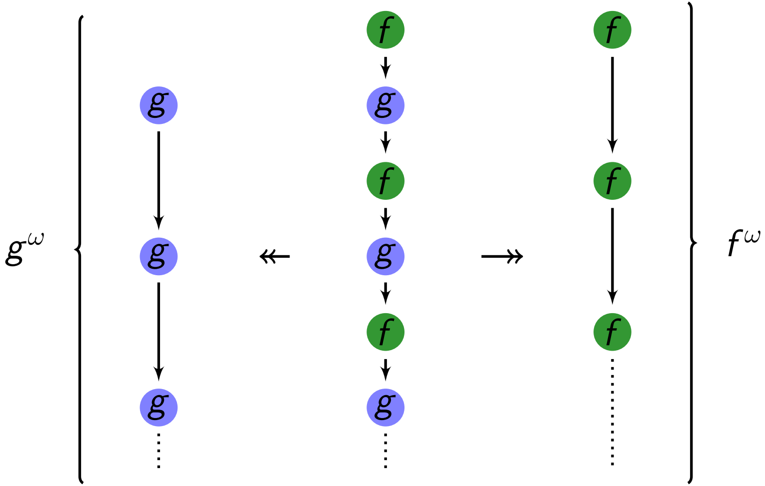
$f(x) \to x$ $\qquad$ $g(x) \to x$



Counter Example

for Orthogonal Term Rewriting Systems

$$f(x) \to x \qquad g(x) \to x$$



Outline

1. Infinitary Term Rewriting
2. Böhm Reduction
3. Partial Order Infinitary Rewriting
4. Term Graph Rewriting

Infinitary Term Rewriting

The Metric Model of Infinitary Rewriting

Convergence

based on the ‘usual’ complete metric space on terms

$$\mathbf{d}(s, t) = 2^{-n}$$

$n =$ depth of the shallowest discrepancy of s and t

The Metric Model of Infinitary Rewriting

Convergence

based on the ‘usual’ complete metric space on terms

$$\mathbf{d}(s, t) = 2^{-n}$$

n = depth of the shallowest discrepancy of s and t

Convergence of reductions

(a.k.a. strong convergence)

- convergence in the metric space, and
- rewrite rules are applied (eventually) at increasingly large depth

The Metric Model of Infinitary Rewriting

Convergence

based on the ‘usual’ complete metric space on terms

$$\mathbf{d}(s, t) = 2^{-n}$$

$n =$ depth of the shallowest discrepancy of s and t

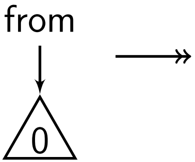
Convergence of reductions

(a.k.a. strong convergence)

- convergence in the metric space, and
- rewrite rules are applied (eventually) at increasingly large depth

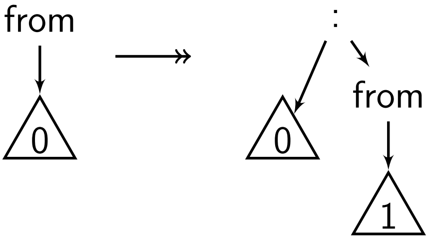
$\rightsquigarrow$ convergence of a reduction: depth at which the rewrite rules are applied tends to infinity

Example: Convergence of a Reduction



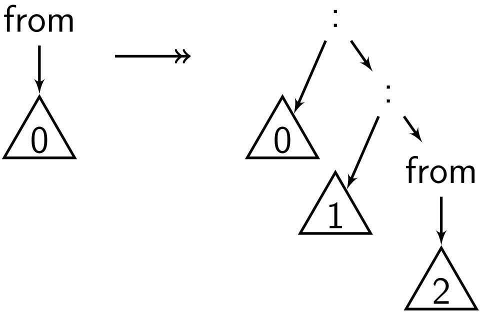
$$from(x) \to x : from(s(x))$$

Example: Convergence of a Reduction



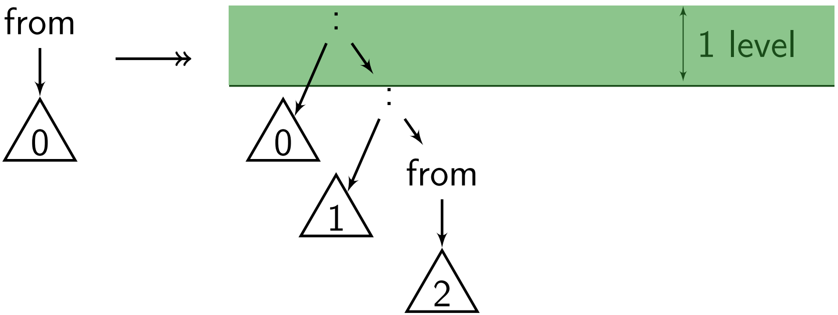
$$\mathit{from}(x) \to x : \mathit{from}(s(x))$$

Example: Convergence of a Reduction



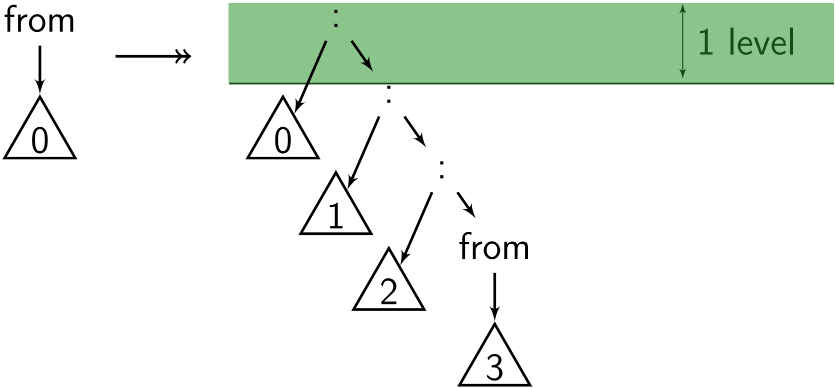
$$from(x) \to x : from(s(x))$$

Example: Convergence of a Reduction



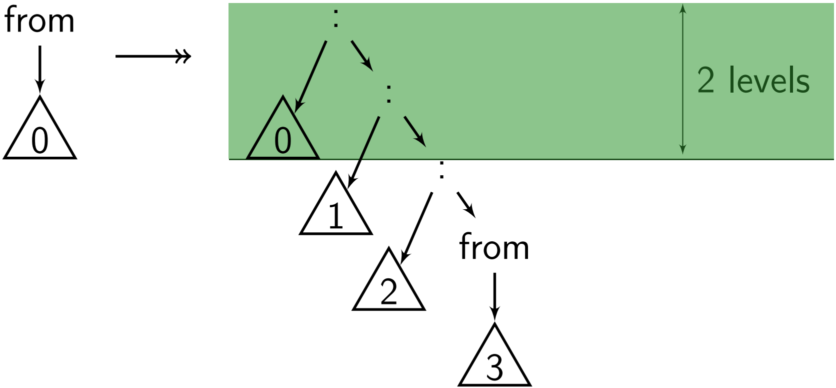
$$from(x) \to x : from(s(x))$$

Example: Convergence of a Reduction



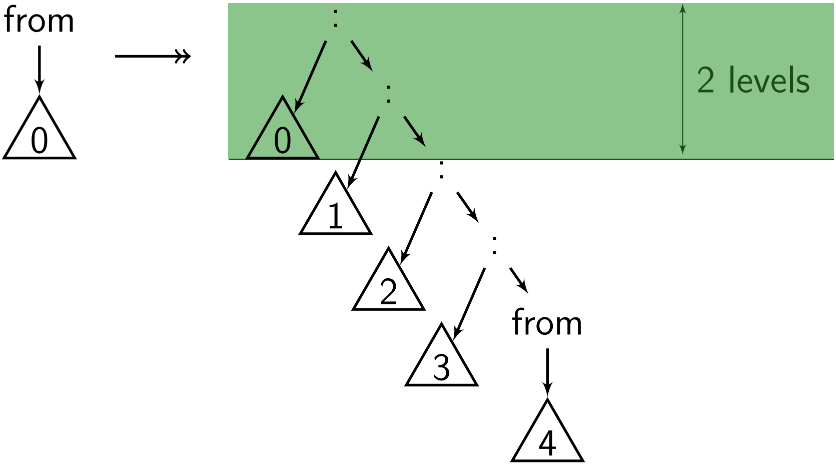
$$from(x) \to x : from(s(x))$$

Example: Convergence of a Reduction



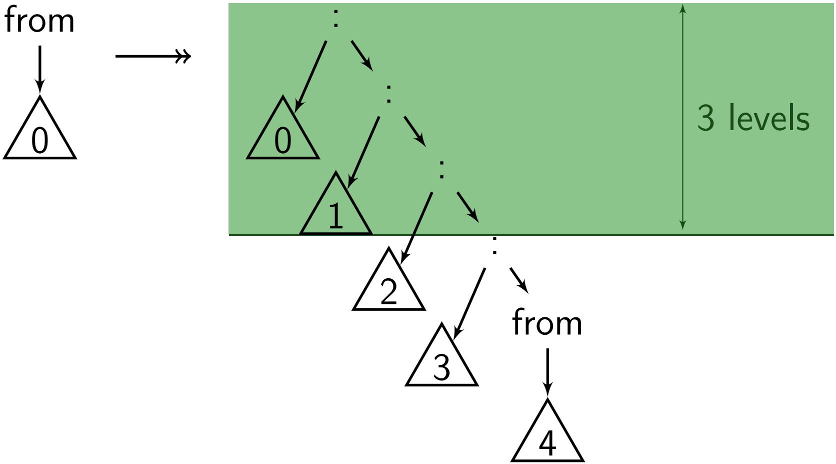
$$from(x) \to x : from(s(x))$$

Example: Convergence of a Reduction



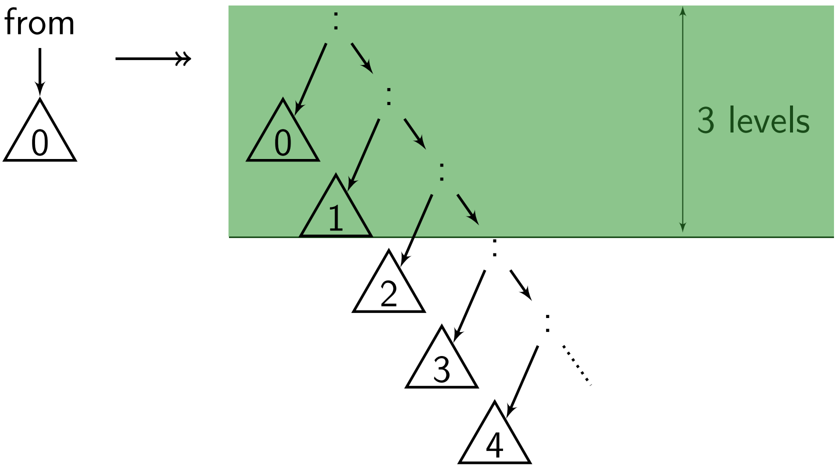
$$from(x) \to x : from(s(x))$$

Example: Convergence of a Reduction



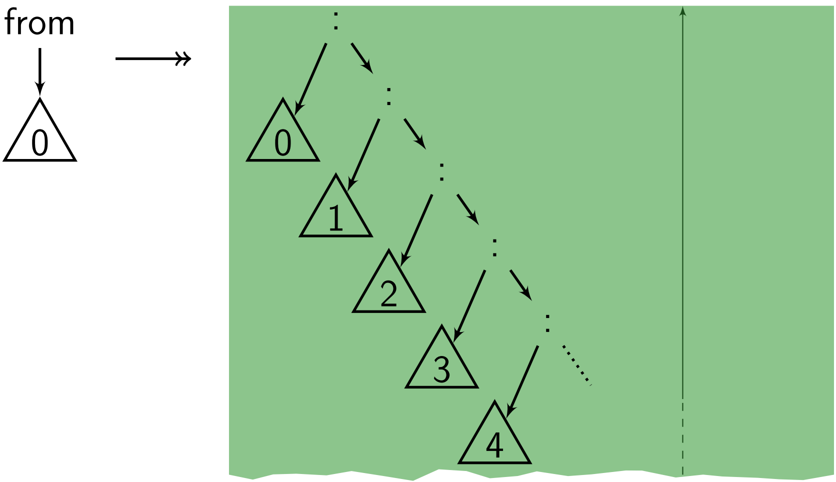
$$from(x) \to x : from(s(x))$$

Example: Convergence of a Reduction



$$from(x) \to x : from(s(x))$$

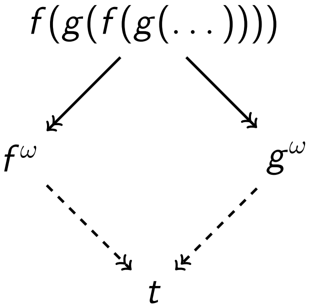
Example: Convergence of a Reduction



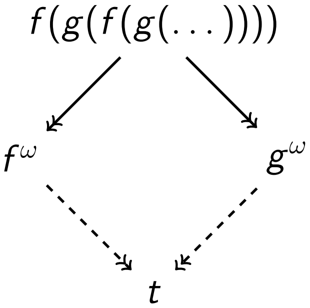
$from(x) \to x : from(s(x))$

Böhm Reduction

Obtaining Infinitary Confluence



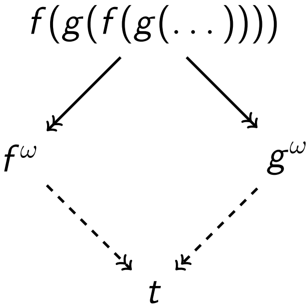
Obtaining Infinitary Confluence



Option 1

Disallow systems with more than one collapsing rule (i.e. rules of the form $t \to x$)

Obtaining Infinitary Confluence



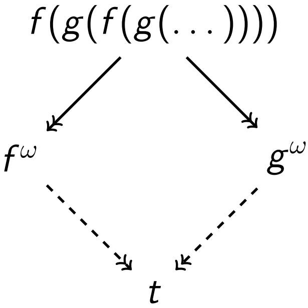
Option 1

Disallow systems with more than one collapsing rule (i.e. rules of the form $t \to x$)

Option 2

Extend the reduction system so that $f^\omega \twoheadrightarrow t \twoheadleftarrow g^\omega$

Obtaining Infinitary Confluence



Option 1

Disallow systems with more than one collapsing rule (i.e. rules of the form $t \to x$)

Option 2

Extend the reduction system so that $f^\omega \twoheadrightarrow t \twoheadleftarrow g^\omega$

Böhm Reduction

Idea

- terms like f^ω and g^ω are considered meaningless
- for each meaningless term t, add rule $t \to \bot$

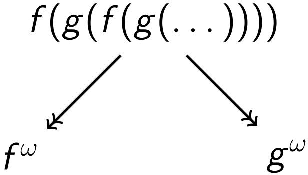
[1] R. Kennaway et al. "Infinitary lambda calculus". In: *Theoretical Computer Science* (1997).

[2] R. Kennaway, V. van Oostrom, and F.-J. de Vries. "Meaningless Terms in Rewriting". In: *J. Funct. Logic Programming* (1999).

Böhm Reduction

Idea

- terms like f^ω and g^ω are considered meaningless
- for each meaningless term t, add rule $t \to \bot$



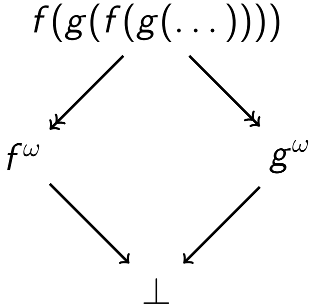
[1] R. Kennaway et al. "Infinitary lambda calculus". In: *Theoretical Computer Science* (1997).

[2] R. Kennaway, V. van Oostrom, and F.-J. de Vries. "Meaningless Terms in Rewriting". In: *J. Funct. Logic Programming* (1999).

Böhm Reduction

Idea

- terms like f^ω and g^ω are considered meaningless
- for each meaningless term t, add rule $t \to \bot$



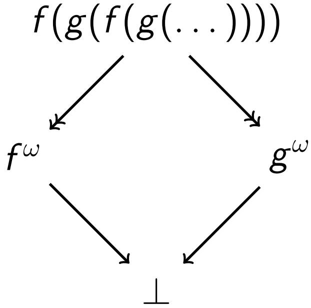
[1] R. Kennaway et al. "Infinitary lambda calculus". In: *Theoretical Computer Science* (1997).

[2] R. Kennaway, V. van Oostrom, and F.-J. de Vries. "Meaningless Terms in Rewriting". In: *J. Funct. Logic Programming* (1999).

Böhm Reduction

Idea

- terms like f^ω and g^ω are considered meaningless
- for each meaningless term t, add rule $t \to \bot$



Böhm reduction = infinitary rewriting with $\bot$-rules

[1] R. Kennaway et al. "Infinitary lambda calculus". In: *Theoretical Computer Science* (1997).

[2] R. Kennaway, V. van Oostrom, and F.-J. de Vries. "Meaningless Terms in Rewriting". In: *J. Funct. Logic Programming* (1999).

Meaningless Terms

Origins in lambda calculus

- Böhm trees[3]
- undefined elements[4]

[3] H. P. Barendregt. *The Lambda Calculus: Its Syntax and Semantics*. 1984.

[4] H. P. Barendregt. "Representing 'undefined' in the lambda calculus". In: *Journal of Functional Programming* 2 (1992), pp. 367–374.

Meaningless Terms

Origins in lambda calculus

- Böhm trees[3]
- undefined elements[4]

Intuition

- terms that have no information content
- because they cannot be distinguished from one another

---

[3] H. P. Barendregt. *The Lambda Calculus: Its Syntax and Semantics*. 1984.

[4] H. P. Barendregt. "Representing 'undefined' in the lambda calculus". In: *Journal of Functional Programming* 2 (1992), pp. 367–374.

Axiomatic Characterisation

A set of terms $\mathcal{U}$ is called meaningless if it satisfies a number of axioms.[5,6]

1. $\mathcal{U}$ is closed under rewriting.
2. If a redex t overlaps a subterm in $\mathcal{U}$, then $t \in \mathcal{U}$.
3. $\mathcal{U}$ is closed under substitution. (for λ-calculus)
4. If t root-active/hypercollapsing, then $t \in \mathcal{U}$.
5. If $s \overset{\mathcal{U}}{\leftrightarrow} t$, then $s \in \mathcal{U}$ if and only if $t \in \mathcal{U}$.

---

[5] Z. M. Ariola et al. "Syntactic definitions of undefined: On defining the undefined". In: *Theoretical Aspects of Computer Software*. 1994.

[6] R. Kennaway, V. van Oostrom, and F.-J. de Vries. "Meaningless Terms in Rewriting". In: *J. Funct. Logic Programming* (1999).

Properties of Böhm Reduction

- Let $\mathcal{R}$ be an orthogonal TRS, and $\mathcal{U}$ a set of meaningless terms.
- Define $\mathcal{B} = \mathcal{R} \cup \{ t \to \bot \mid t \in \mathcal{U}_\bot \setminus \{\bot\} \}$

Properties of Böhm Reduction

- Let $\mathcal{R}$ be an orthogonal TRS, and $\mathcal{U}$ a set of meaningless terms.
- Define $\mathcal{B} = \mathcal{R} \cup \{t \to \bot \mid t \in \mathcal{U}_\bot \setminus \{\bot\}\}$

Theorem

- *$\mathcal{B}$ is infinitarily confluent,*
  i.e. $t_1 \twoheadleftarrow_\mathcal{B} t \twoheadrightarrow_\mathcal{B} t_2$ implies $t_1 \twoheadrightarrow_\mathcal{B} t' \twoheadleftarrow_\mathcal{B} t_2$.
- *$\mathcal{B}$ is infinitarily normalising, i.e. for each term t there is a reduction $t \twoheadrightarrow_\mathcal{B} t'$ to a normal form.*

Properties of Böhm Reduction

- Let $\mathcal{R}$ be an orthogonal TRS, and $\mathcal{U}$ a set of meaningless terms.
- Define $\mathcal{B} = \mathcal{R} \cup \{t \to \bot \mid t \in \mathcal{U}_\bot \setminus \{\bot\}\}$

Theorem

- *$\mathcal{B}$ is infinitarily confluent,*
  i.e. $t_1 \twoheadleftarrow_{\mathcal{B}} t \twoheadrightarrow_{\mathcal{B}} t_2$ implies $t_1 \twoheadrightarrow_{\mathcal{B}} t' \twoheadleftarrow_{\mathcal{B}} t_2$.
- *$\mathcal{B}$ is infinitarily normalising, i.e. for each term t there is a reduction $t \twoheadrightarrow_{\mathcal{B}} t'$ to a normal form.*

Corollary

Each term has a unique infinitary normal form in $\mathcal{B}$ (called Böhm tree).

Partial Order Infinitary Rewriting

Partial Order Infinitary Rewriting

- Alternative characterisation of Böhm reduction
- Changes the notion of convergence instead of adding rules

[7] B. "Partial Order Infinitary Term Rewriting". In: *Logical Methods in Computer Science* (2014).

Partial Order Infinitary Rewriting

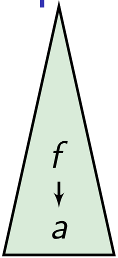
- Alternative characterisation of Böhm reduction
- Changes the notion of convergence instead of adding rules

The Good & The Bad

+ less ad hoc
+ no need for infinitely many reduction rules
- captures only a particular set of meaningless terms

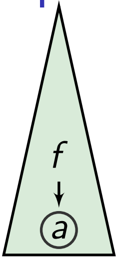
[7] B. "Partial Order Infinitary Term Rewriting". In: *Logical Methods in Computer Science* (2014).

Recap: Strong Convergence



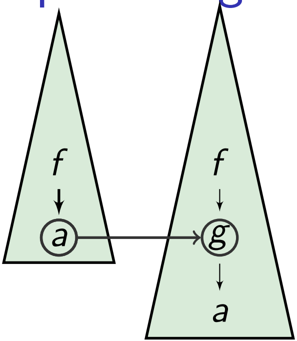
$$\mathcal{R} = \{a \to g(a)\}$$

Recap: Strong Convergence



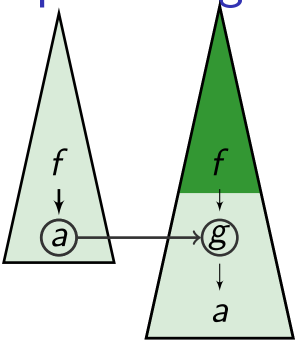
$$\mathcal{R} = \{a \to g(a)\}$$

Recap: Strong Convergence



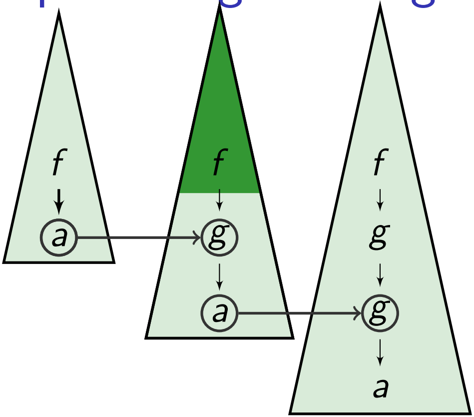
$$\mathcal{R} = \{a \to g(a)\}$$

Recap: Strong Convergence



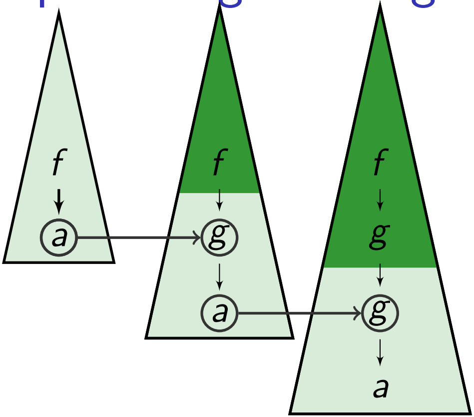
$$\mathcal{R} = \{a \to g(a)\}$$

Recap: Strong Convergence



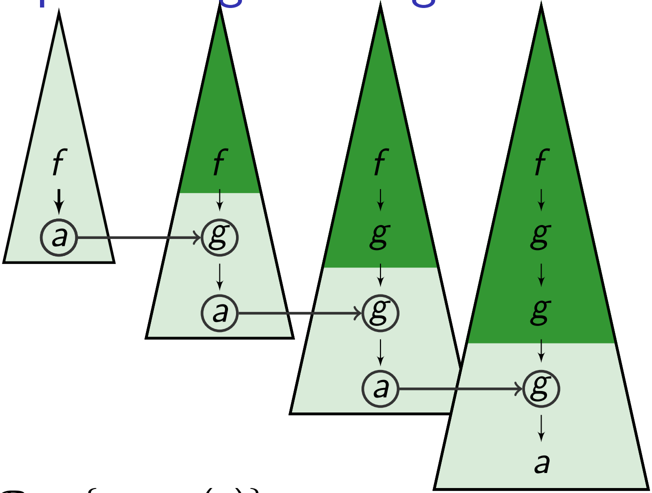
$$\mathcal{R} = \{a \to g(a)\}$$

Recap: Strong Convergence



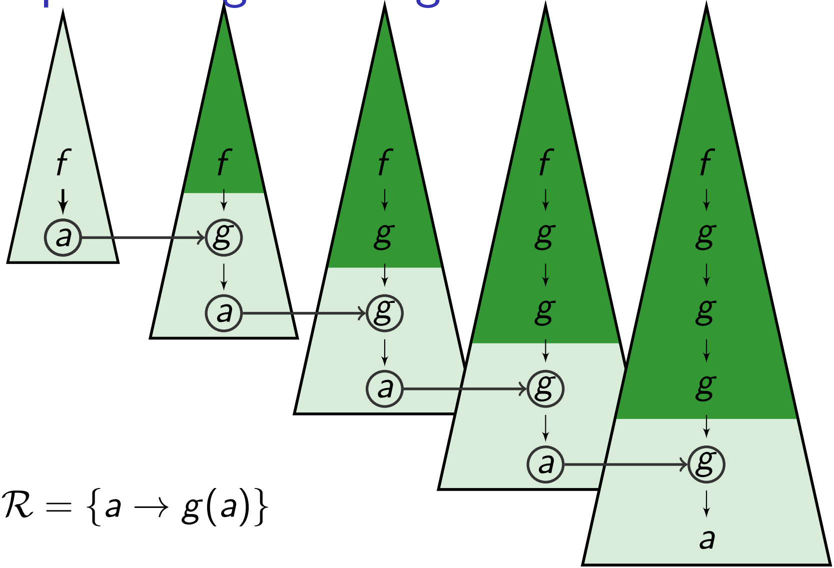
$$\mathcal{R} = \{a \to g(a)\}$$

Recap: Strong Convergence



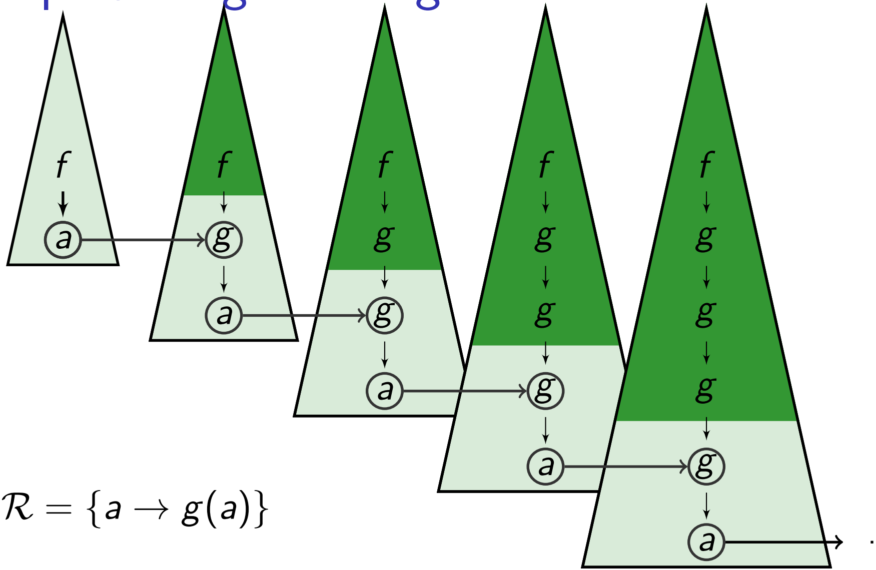
$$\mathcal{R} = \{a \to g(a)\}$$

Recap: Strong Convergence



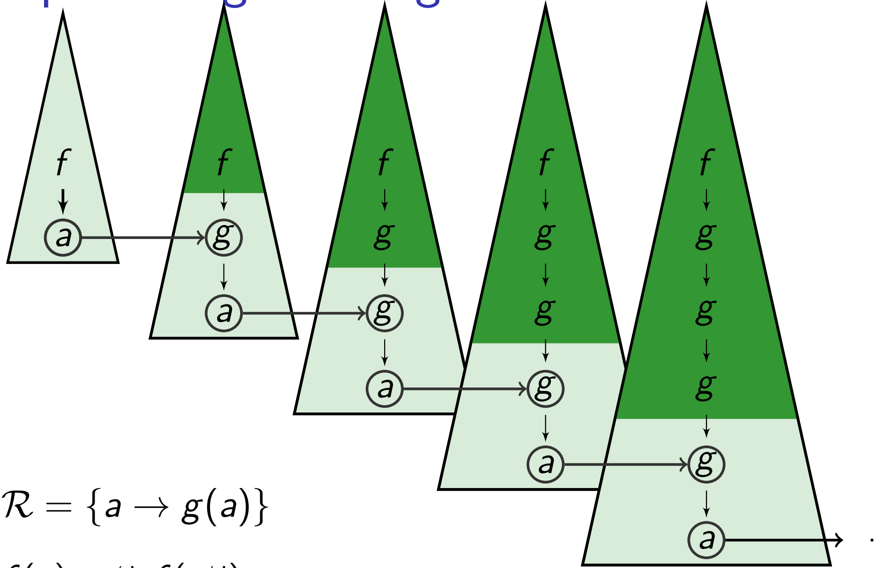
$$\mathcal{R} = \{a \to g(a)\}$$

Recap: Strong Convergence



$$\mathcal{R} = \{a \to g(a)\}$$

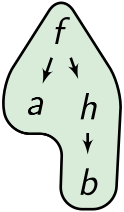
Recap: Strong Convergence



$\mathcal{R} = \{a \to g(a)\}$

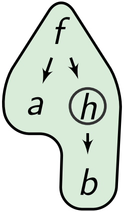
$f(a) \twoheadrightarrow^{\omega}_{\mathcal{R}} f(g^{\omega})$

Example: Non-Convergence



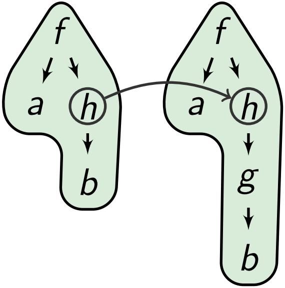
$$\mathcal{R} = \begin{cases} a \to g(a) \\ h(x) \to h(g(x)) \end{cases}$$

Example: Non-Convergence



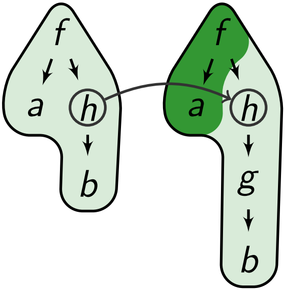
$$\mathcal{R} = \begin{cases} a \to g(a) \\ h(x) \to h(g(x)) \end{cases}$$

Example: Non-Convergence



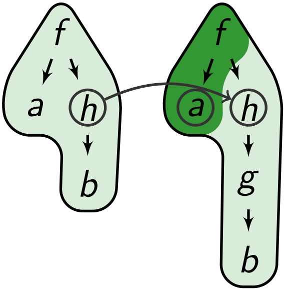
$$\mathcal{R} = \begin{cases} a \to g(a) \\ h(x) \to h(g(x)) \end{cases}$$

Example: Non-Convergence



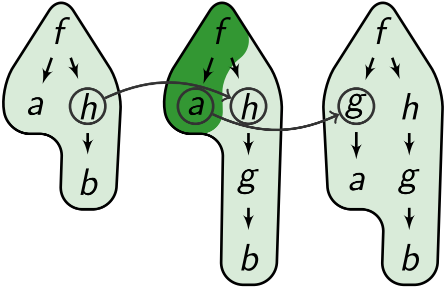
$$\mathcal{R} = \begin{cases} a \to g(a) \\ h(x) \to h(g(x)) \end{cases}$$

Example: Non-Convergence



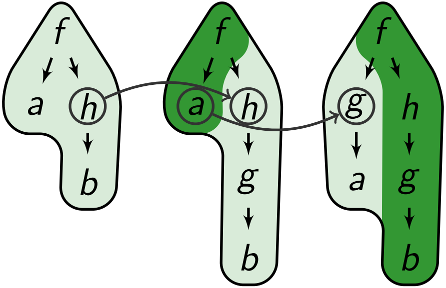
$$\mathcal{R} = \begin{cases} a \to g(a) \\ h(x) \to h(g(x)) \end{cases}$$

Example: Non-Convergence



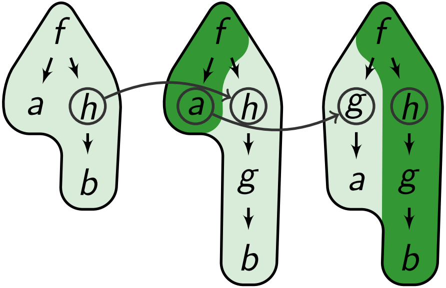
$$\mathcal{R} = \begin{cases} a \to g(a) \\ h(x) \to h(g(x)) \end{cases}$$

Example: Non-Convergence



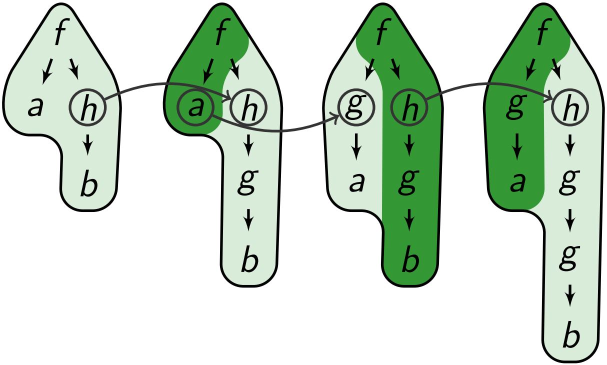
$$\mathcal{R} = \begin{cases} a \to g(a) \\ h(x) \to h(g(x)) \end{cases}$$

Example: Non-Convergence



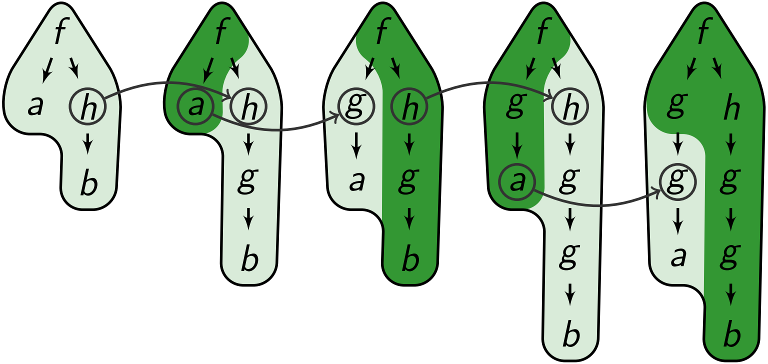
$$\mathcal{R} = \begin{cases} a \to g(a) \\ h(x) \to h(g(x)) \end{cases}$$

Example: Non-Convergence



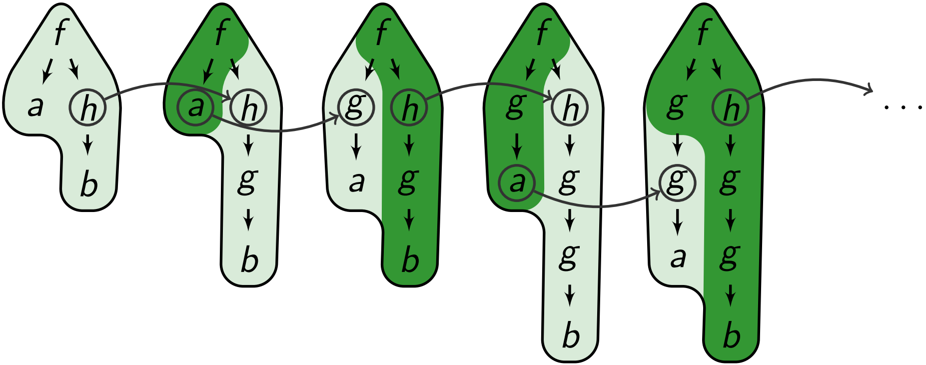
$$\mathcal{R} = \begin{cases} a \to g(a) \\ h(x) \to h(g(x)) \end{cases}$$

Example: Non-Convergence



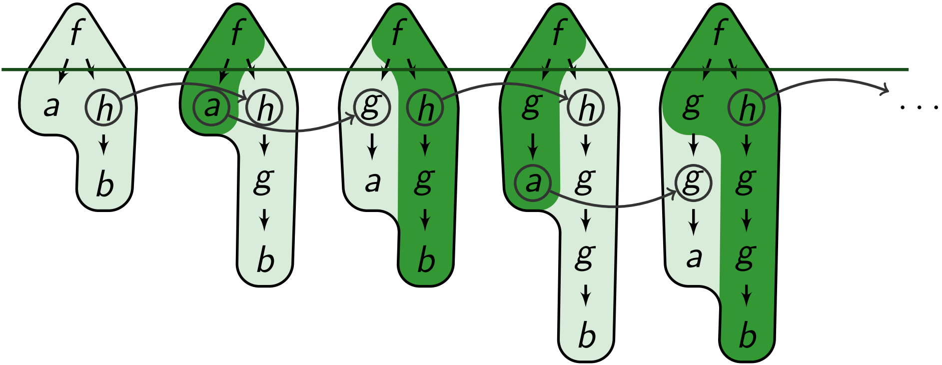
$$\mathcal{R} = \begin{cases} a \to g(a) \\ h(x) \to h(g(x)) \end{cases}$$

Example: Non-Convergence



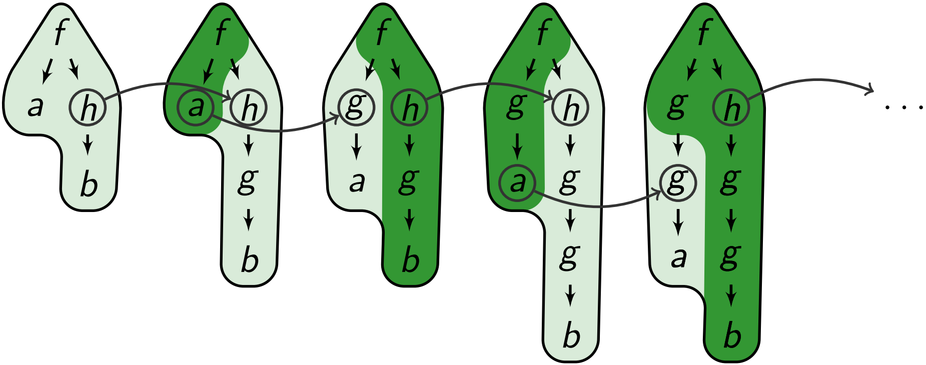
$$\mathcal{R} = \begin{cases} a \to g(a) \\ h(x) \to h(g(x)) \end{cases}$$

Example: Non-Convergence

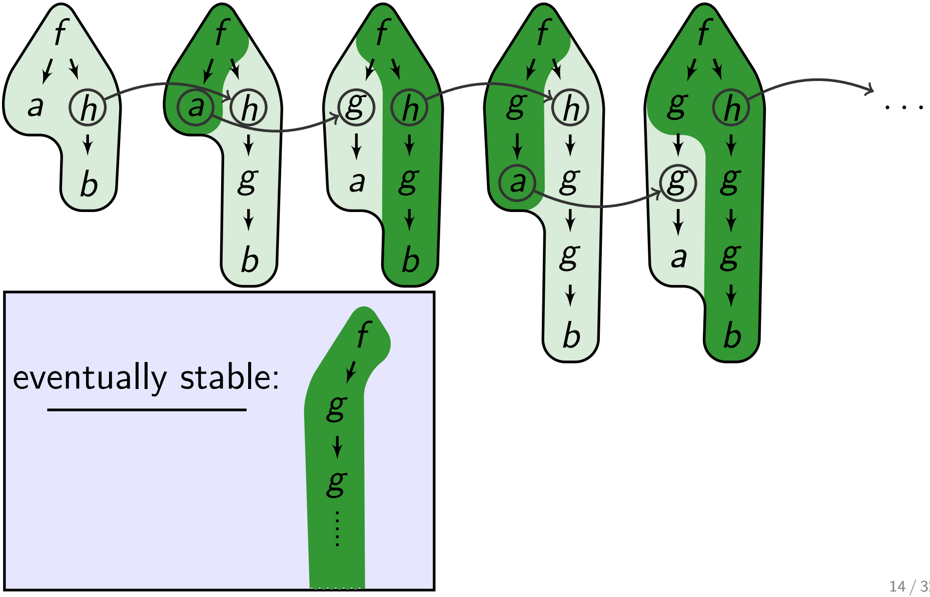


$$\mathcal{R} = \begin{cases} a \to g(a) \\ h(x) \to h(g(x)) \end{cases}$$

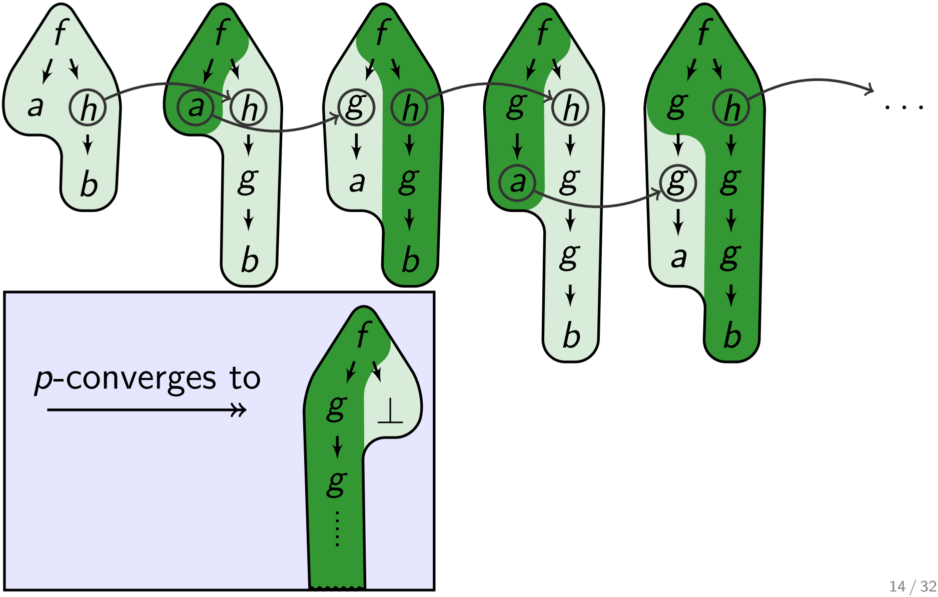
Partial Order Convergence



Partial Order Convergence



Partial Order Convergence



How does it work? (I)

Partial order on terms

- partial terms: terms with additional constant $\bot$
- partial order $\leq_\bot$ reads as: "is less defined than"
  - $\bot \leq_\bot t$,
  - $\bar{s} \leq_\bot \bar{t} \implies f(\bar{s}) \leq_\bot f(\bar{t})$
- e.g. $f(\bot, g(x)) \leq_\bot f(y, g(x))$
- $\leq_\bot$ is a complete semilattice
  ($=$ cpo $+$ glbs of non-empty sets)

How does it work? (II)

Convergence: limit inferior

$$\liminf_{\iota \to \alpha} t_\iota = \bigsqcup_{\beta < \alpha} \bigsqcap_{\beta \leq \iota < \alpha} t_\iota$$

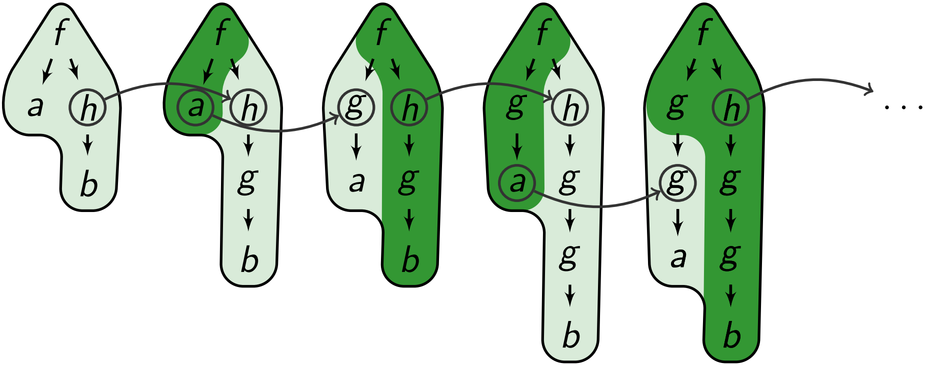
How does it work? (II)

Convergence: limit inferior

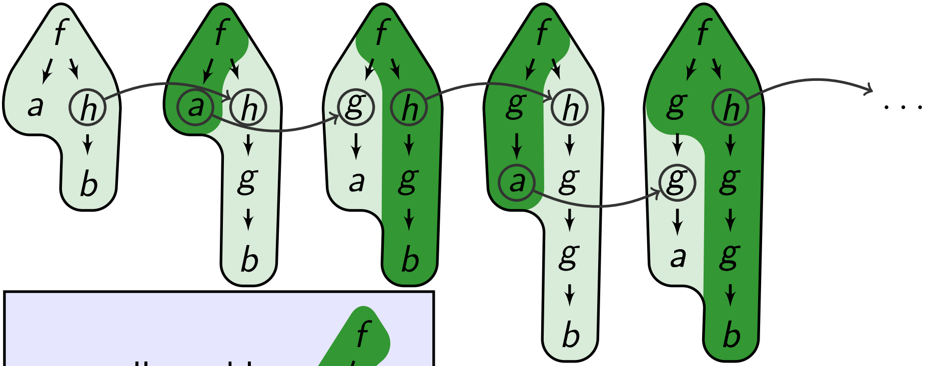
$$\liminf_{\iota \to \alpha} t_\iota = \bigsqcup_{\beta < \alpha} \bigsqcap_{\beta \leq \iota < \alpha} t_\iota$$

- intuition: eventual persistence of nodes in the tree
- strong convergence: limit inferior of the contexts of the reduction

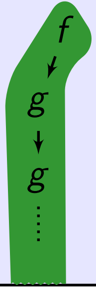
Partial Order Convergence



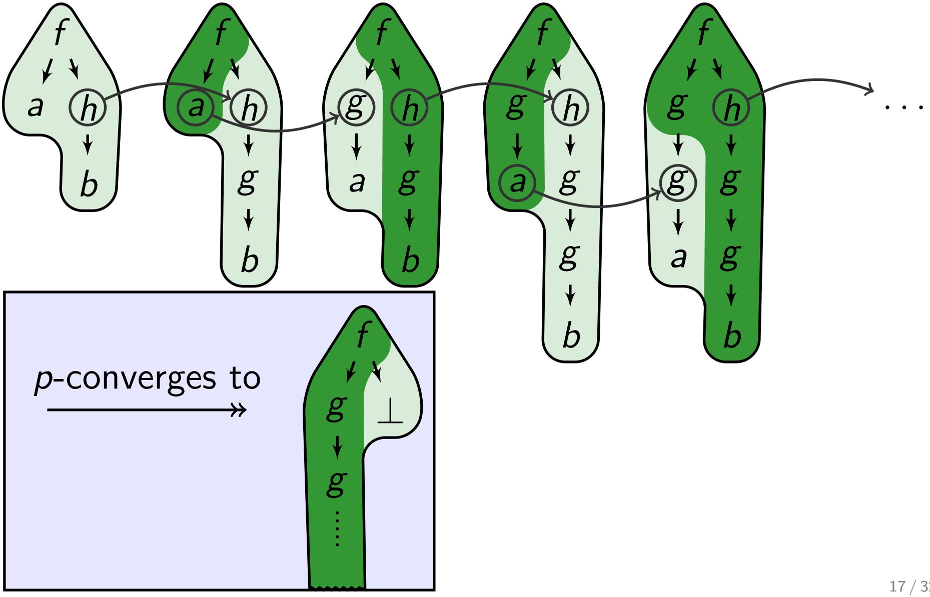
Partial Order Convergence



eventually stable:



Partial Order Convergence



Properties of Orthogonal TRS

property	**metric**	**Böhm red.**
compression	✔	✔
finite approx.	✔	✔
developments	✘	✔
inf. confluence	✘	✔
inf. normalisation	✘	✔

Properties of Orthogonal TRS

property	**metric**	**Böhm red.**	**part. order**
compression	✔	✔	✔
finite approx.	✔	✔	✔
developments	✘	✔	✔
inf. confluence	✘	✔	✔
inf. normalisation	✘	✔	✔

Partial Order vs. Böhm Reduction

Theorem

If $\mathcal{R}$ is an orthogonal TRS and s, t total terms, then

$$s \xrightarrow{p}\!\!\!\!\!\twoheadrightarrow_{\mathcal{R}} t \quad \text{iff} \quad s \xrightarrow{m}\!\!\!\!\!\twoheadrightarrow_{\mathcal{R}} t.$$

Partial Order vs. Böhm Reduction

Theorem

If $\mathcal{R}$ is an orthogonal TRS and s, t total terms, then

$$s \xrightarrow{p}\!\!\!\!\!\rightarrow_{\mathcal{R}} t \quad \text{iff} \quad s \xrightarrow{m}\!\!\!\!\!\rightarrow_{\mathcal{R}} t.$$

Theorem

If $\mathcal{R}$ is an orthogonal TRS and $\mathcal{B}$ the Böhm extension of $\mathcal{R}$ (w.r.t. root-active terms), then

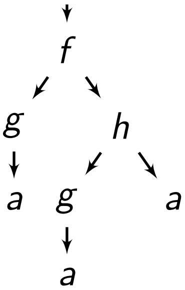
$$s \xrightarrow{p}\!\!\!\!\!\rightarrow_{\mathcal{R}} t \quad \text{iff} \quad s \xrightarrow{m}\!\!\!\!\!\rightarrow_{\mathcal{B}} t.$$

Term Graph Rewriting

From Terms to Term Graphs

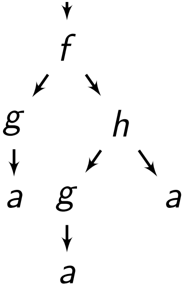
$$f(g(a), h(g(a), a))$$

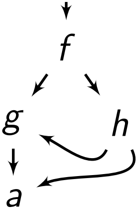
From Terms to Term Graphs



$$f(g(a), h(g(a), a))$$

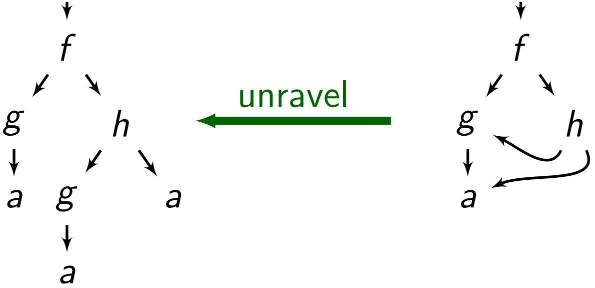
From Terms to Term Graphs





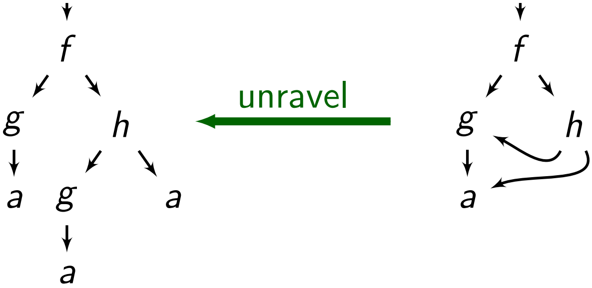
$f(g(a), h(g(a), a))$

From Terms to Term Graphs



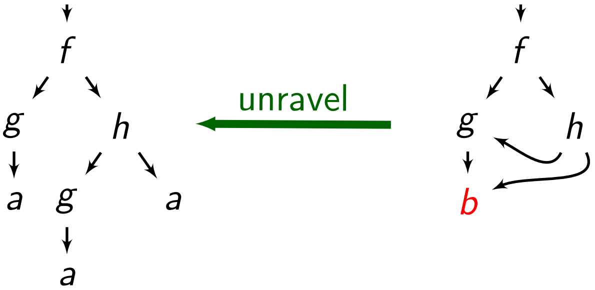
$$f(g(a), h(g(a), a))$$

From Terms to Term Graphs



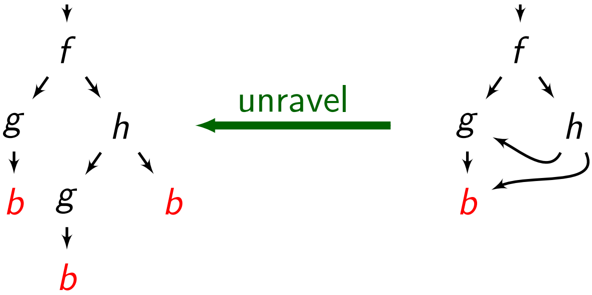
$$a \to b$$

From Terms to Term Graphs



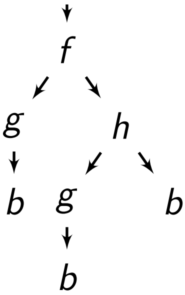
$a \rightarrow b$

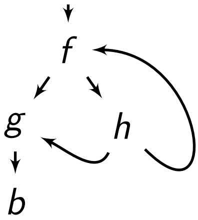
From Terms to Term Graphs



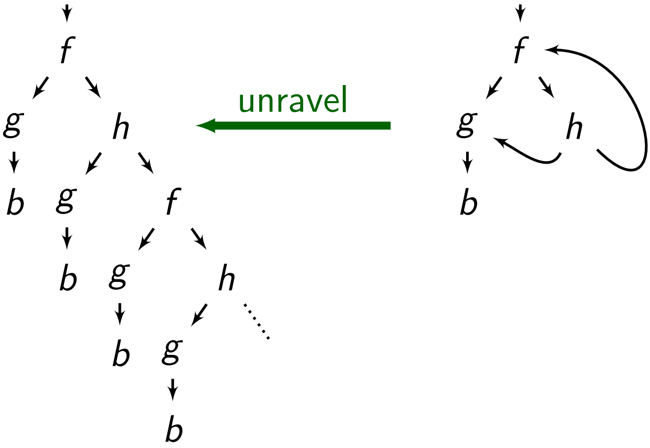
$a \to b$

From Terms to Term Graphs

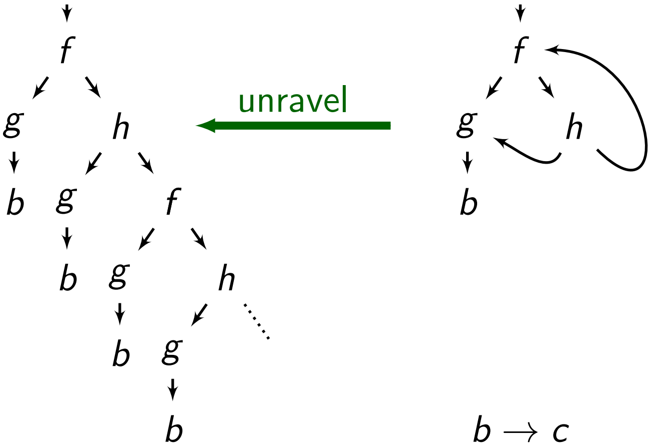




From Terms to Term Graphs

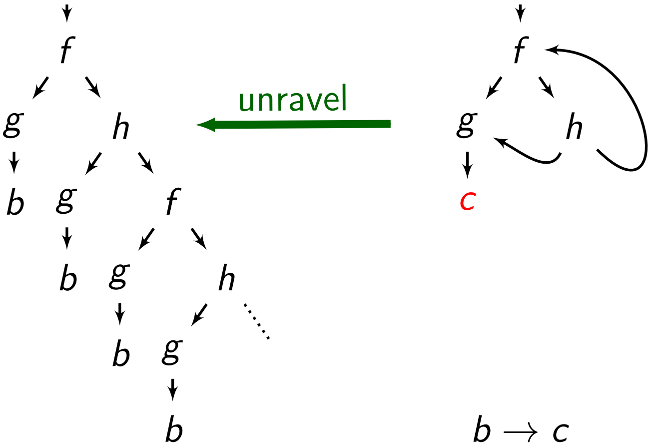


From Terms to Term Graphs



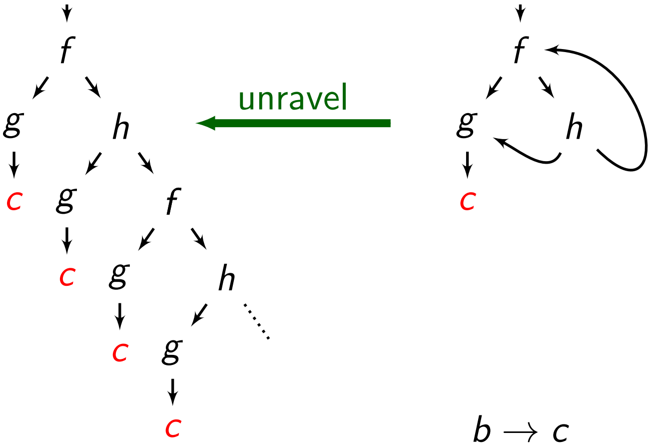
$b \to c$

From Terms to Term Graphs



$b \to c$

From Terms to Term Graphs

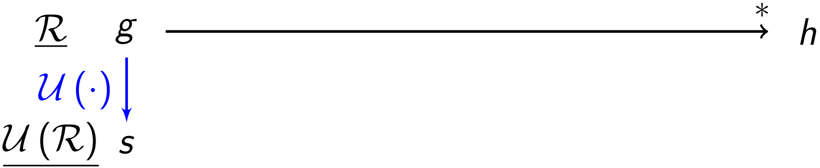


$b \to c$

Soundness & Completeness

Soundness of finite reductions

For every left-linear, left-finite GRS $\mathcal{R}$ we have

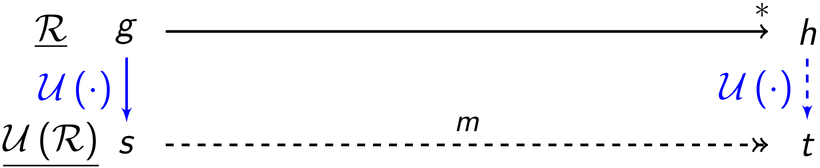


[8] R. Kennaway et al. "On the adequacy of graph rewriting for simulating term rewriting". In: *ACM Transactions on Programming Languages and Systems* (1994).

Soundness & Completeness

Soundness of finite reductions

For every left-linear, left-finite GRS $\mathcal{R}$ we have

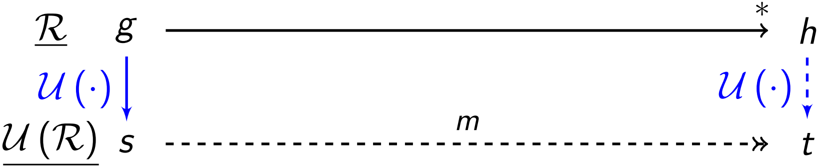


[8] R. Kennaway et al. "On the adequacy of graph rewriting for simulating term rewriting". In: *ACM Transactions on Programming Languages and Systems* (1994).

Soundness & Completeness

Soundness of finite reductions

For every left-linear, left-finite GRS $\mathcal{R}$ we have



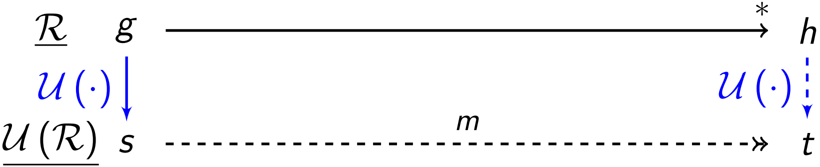
Completeness property

[8] R. Kennaway et al. "On the adequacy of graph rewriting for simulating term rewriting". In: *ACM Transactions on Programming Languages and Systems* (1994).

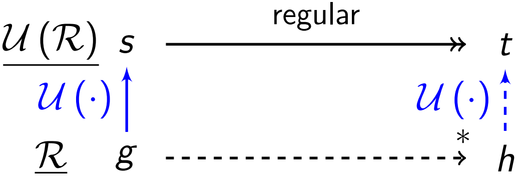
Soundness & Completeness

Soundness of finite reductions

For every left-linear, left-finite GRS $\mathcal{R}$ we have



Completeness property

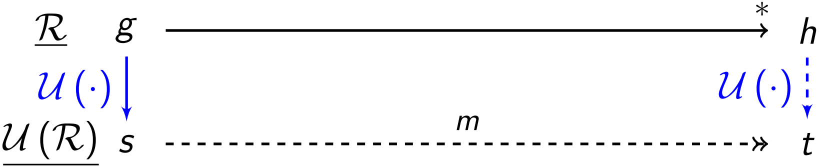


[8] R. Kennaway et al. "On the adequacy of graph rewriting for simulating term rewriting". In: *ACM Transactions on Programming Languages and Systems* (1994).

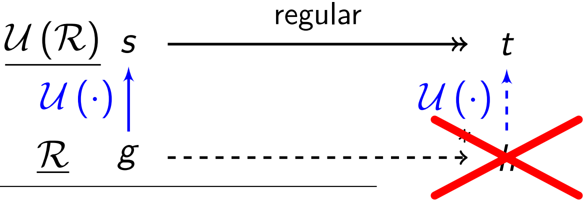
Soundness & Completeness

Soundness of finite reductions

For every left-linear, left-finite GRS $\mathcal{R}$ we have



Completeness property

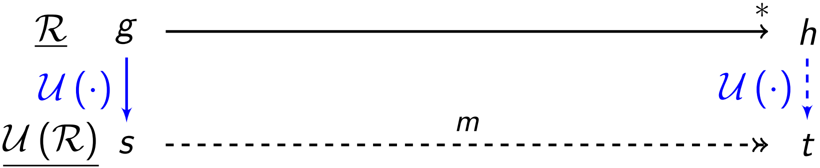


[8] R. Kennaway et al. "On the adequacy of graph rewriting for simulating term rewriting". In: *ACM Transactions on Programming Languages and Systems* (1994).

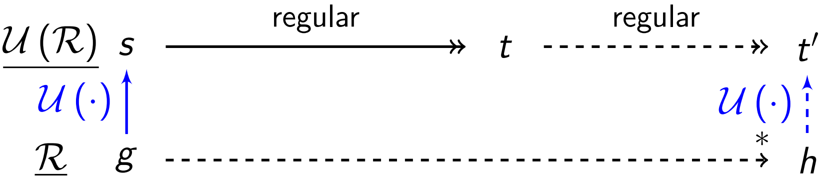
Soundness & Completeness

Soundness of finite reductions

For every left-linear, left-finite GRS $\mathcal{R}$ we have



Completeness property



[8] R. Kennaway et al. "On the adequacy of graph rewriting for simulating term rewriting". In: *ACM Transactions on Programming Languages and Systems* (1994).

Infinitary Term Graph Rewriting

- A common formalism
  - study correspondences between infinitary TRSs and finitary GRSs

[9] Z. M. Ariola and S. Blom. "Skew confluence and the lambda calculus with letrec". In: *Annals of Pure and Applied Logic* (2002).

[10] C. Grabmayer and J. Rochel. "Maximal Sharing in the Lambda Calculus with Letrec". In: *ICFP*. 2014.

Infinitary Term Graph Rewriting

- A common formalism
  - study correspondences between infinitary TRSs and finitary GRSs
- Lazy evaluation
  - infinitary term rewriting only covers non-strictness
  - however: lazy evaluation = non-strictness + sharing

[9] Z. M. Ariola and S. Blom. "Skew confluence and the lambda calculus with letrec". In: *Annals of Pure and Applied Logic* (2002).

[10] C. Grabmayer and J. Rochel. "Maximal Sharing in the Lambda Calculus with Letrec". In: *ICFP*. 2014.

Infinitary Term Graph Rewriting

- A common formalism
  - study correspondences between infinitary TRSs and finitary GRSs
- Lazy evaluation
  - infinitary term rewriting only covers non-strictness
  - however: lazy evaluation = non-strictness + sharing
- infinitary lambda calculi with letrec[9,10]
  - these calculi are non-confluent
  - but there is a notion of infinite normal forms

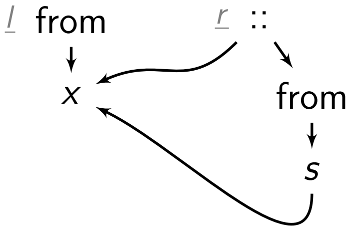
---

[9] Z. M. Ariola and S. Blom. "Skew confluence and the lambda calculus with letrec". In: *Annals of Pure and Applied Logic* (2002).

[10] C. Grabmayer and J. Rochel. "Maximal Sharing in the Lambda Calculus with Letrec". In: *ICFP*. 2014.

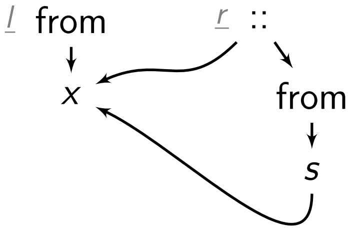
Example: Acyclic Sharing

Term graph rule for $from(x) \to x :: from(s(x))$

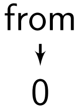


Example: Acyclic Sharing

Term graph rule for $from(x) \to x :: from(s(x))$

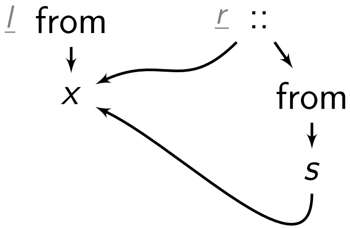


Reductions:

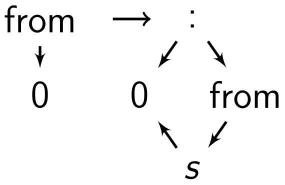


Example: Acyclic Sharing

Term graph rule for $from(x) \to x :: from(s(x))$

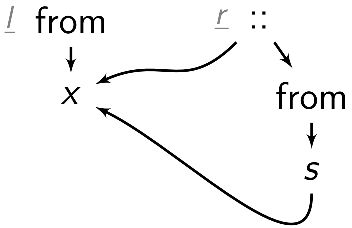


Reductions:

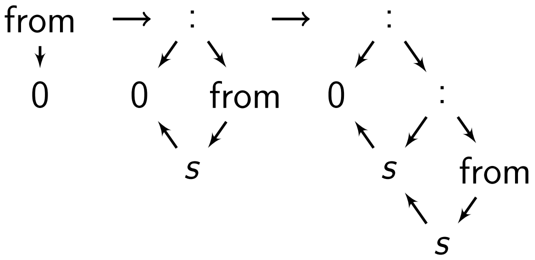


Example: Acyclic Sharing

Term graph rule for $from(x) \to x :: from(s(x))$

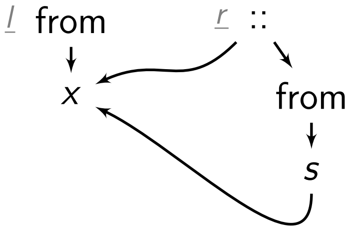


Reductions:

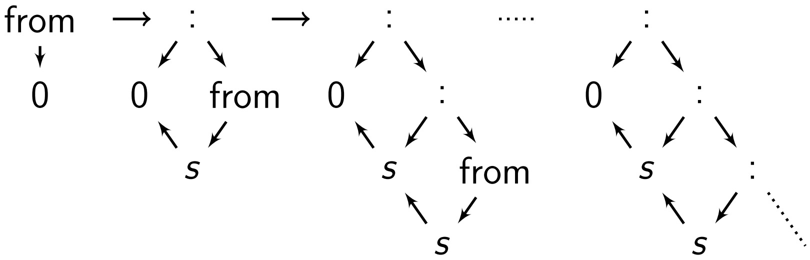


Example: Acyclic Sharing

Term graph rule for $from(x) \to x :: from(s(x))$

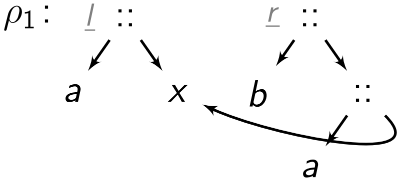


Reductions:



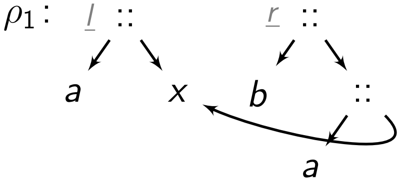
Example: Cyclic Sharing

Term graph rules for $a :: x \to b :: a :: x$

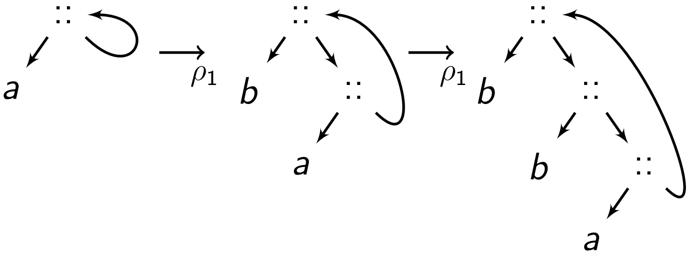


Example: Cyclic Sharing

Term graph rules for $a :: x \to b :: a :: x$

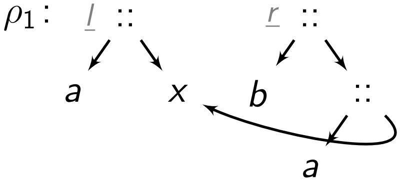


Reductions:

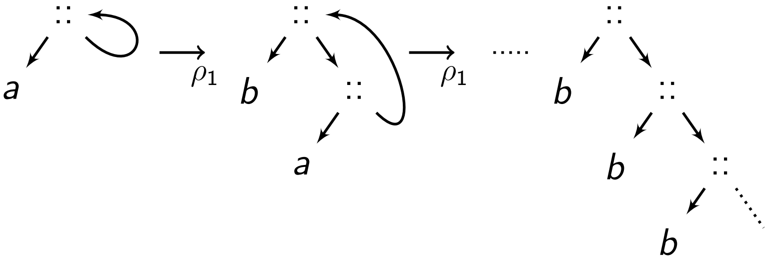


Example: Cyclic Sharing

Term graph rules for $a :: x \to b :: a :: x$

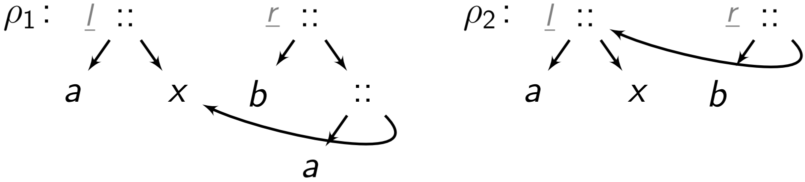


Reductions:

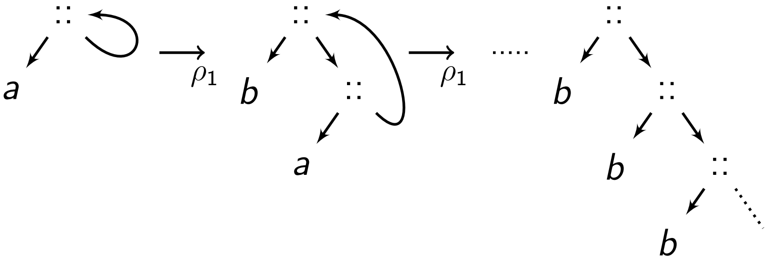


Example: Cyclic Sharing

Term graph rules for $a :: x \to b :: a :: x$

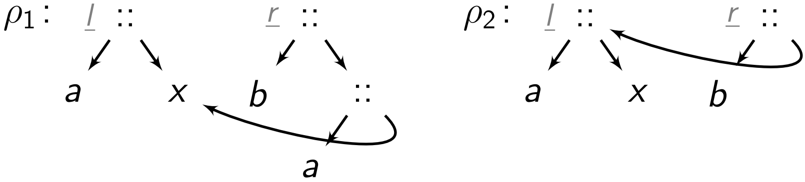


Reductions:

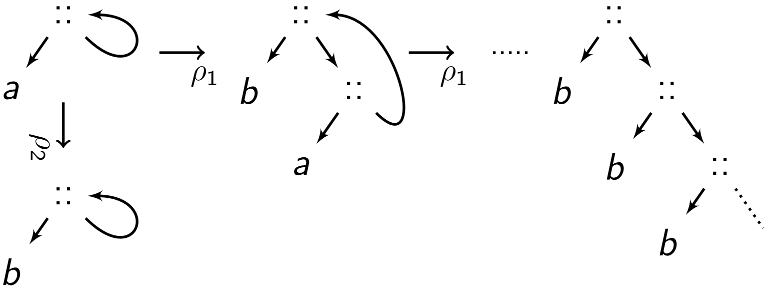


Example: Cyclic Sharing

Term graph rules for $a :: x \to b :: a :: x$

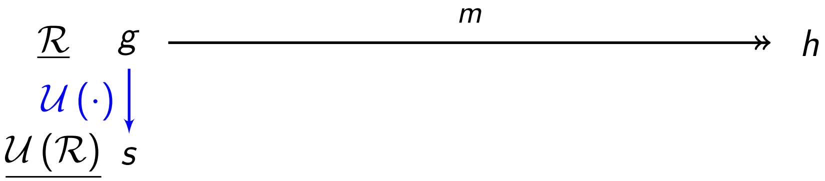


Reductions:



Soundness & Completeness

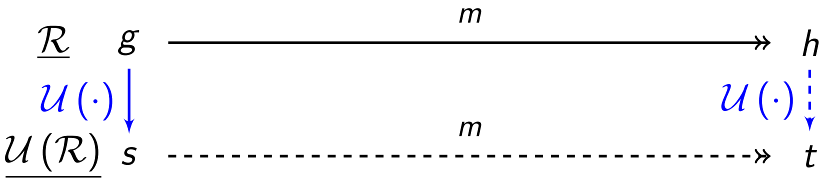
Soundness



[11] B. "Infinitary Term Graph Rewriting is Simple, Sound and Complete". In: *RTA*. 2012.

Soundness & Completeness

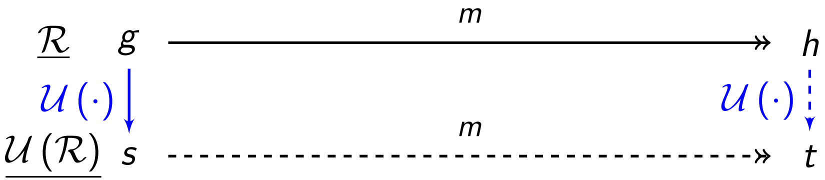
Soundness



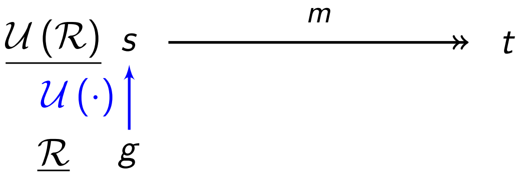
[11] B. “Infinitary Term Graph Rewriting is Simple, Sound and Complete”. In: *RTA*. 2012.

Soundness & Completeness

Soundness



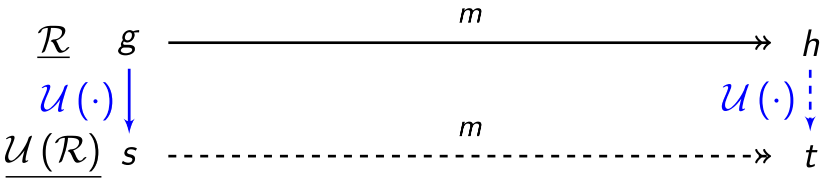
Completeness property



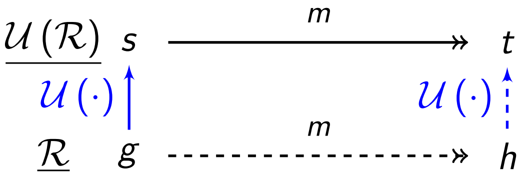
[11] B. "Infinitary Term Graph Rewriting is Simple, Sound and Complete". In: *RTA*. 2012.

Soundness & Completeness

Soundness



Completeness property

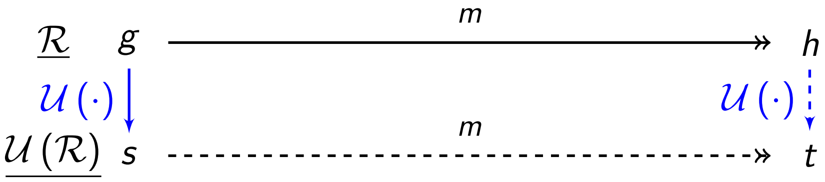


---

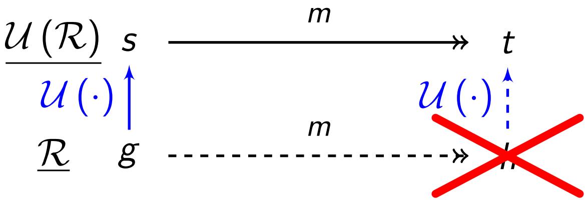
[11] B. “Infinitary Term Graph Rewriting is Simple, Sound and Complete”. In: *RTA*. 2012.

Soundness & Completeness

Soundness



Completeness property

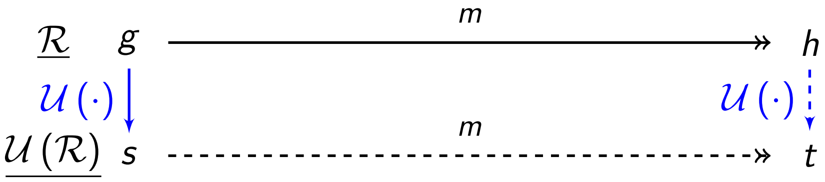


---

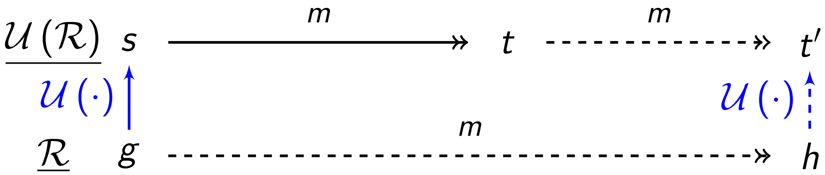
[11] B. "Infinitary Term Graph Rewriting is Simple, Sound and Complete". In: *RTA*. 2012.

Soundness & Completeness

Soundness



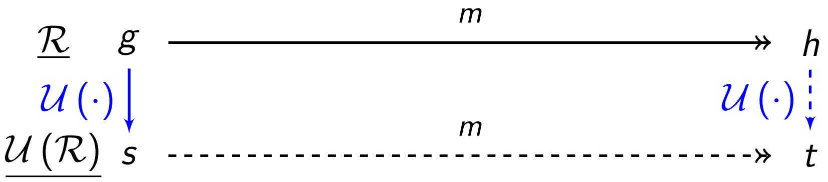
Completeness property



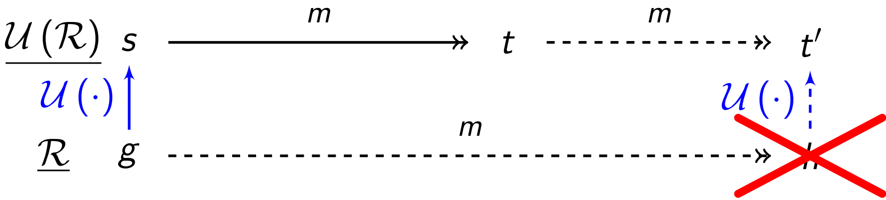
[11] B. "Infinitary Term Graph Rewriting is Simple, Sound and Complete". In: *RTA*. 2012.

Soundness & Completeness

Soundness



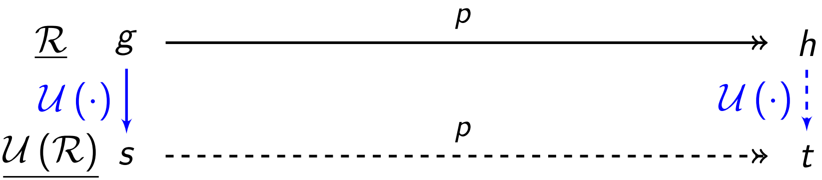
Completeness property



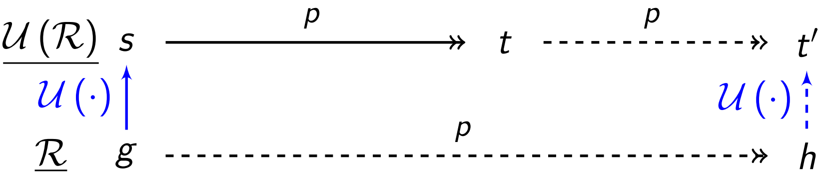
[11] B. "Infinitary Term Graph Rewriting is Simple, Sound and Complete". In: *RTA*. 2012.

Soundness & Completeness

Soundness



Completeness property

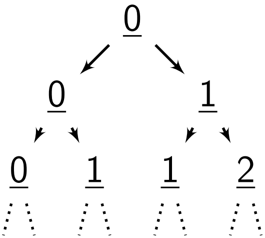


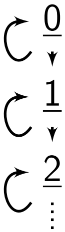
[11] B. “Infinitary Term Graph Rewriting is Simple, Sound and Complete”. In: *RTA*. 2012.

$$\mathcal{R} = \{ \quad \underline{n}(x, y) \to \underline{n+1}(x, y) \quad | \quad n \in \mathbb{N} \quad \}.$$

[12] R. Kennaway et al. "On the adequacy of graph rewriting for simulating term rewriting". In: *ACM Transactions on Programming Languages and Systems* (1994).

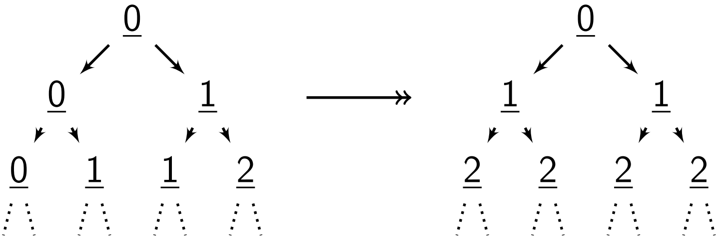
$$\mathcal{R} = \{ \quad \underline{n}(x,y) \to \underline{n+1}(x,y) \quad | \quad n \in \mathbb{N} \quad \}.$$

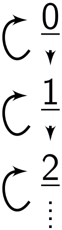




[12] R. Kennaway et al. "On the adequacy of graph rewriting for simulating term rewriting". In: *ACM Transactions on Programming Languages and Systems* (1994).

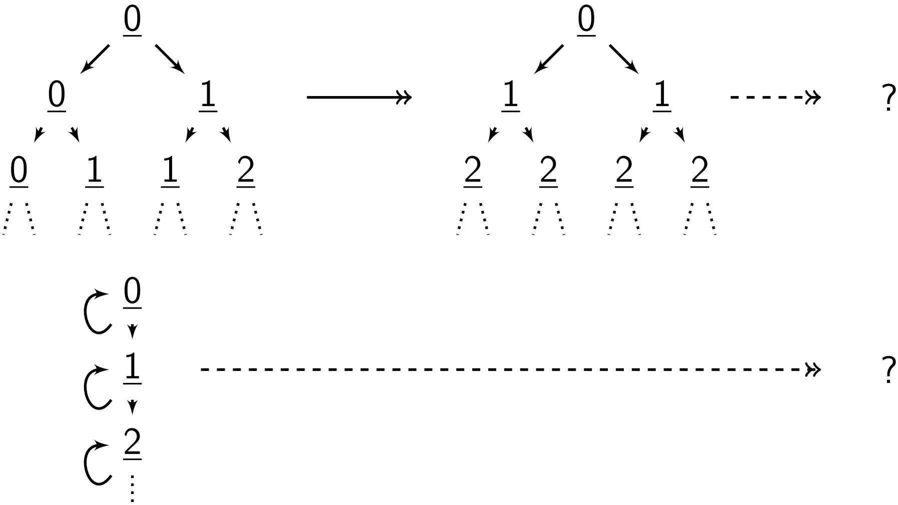
$$\mathcal{R} = \{ \quad \underline{n}(x, y) \to \underline{n+1}(x, y) \quad \mid \quad n \in \mathbb{N} \quad \}.$$





[12] R. Kennaway et al. "On the adequacy of graph rewriting for simulating term rewriting". In: *ACM Transactions on Programming Languages and Systems* (1994).

$$\mathcal{R} = \{ \quad \underline{n}(x, y) \to \underline{n+1}(x, y) \quad | \quad n \in \mathbb{N} \quad \}.$$

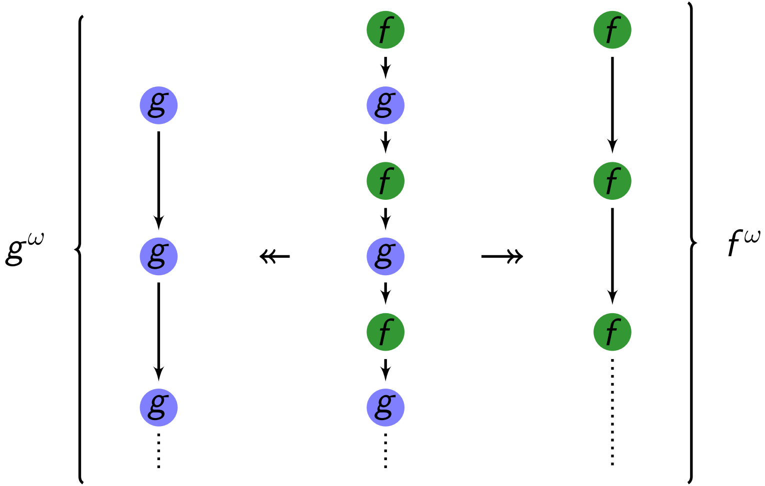


[12] R. Kennaway et al. "On the adequacy of graph rewriting for simulating term rewriting". In: *ACM Transactions on Programming Languages and Systems* (1994).

Confluence Fails

for Orthogonal Term (Graph) Rewriting Systems

$$f(x) \to x \qquad g(x) \to x$$



Properties of Orthogonal GRS

property	**metric**	**Böhm red.**	**part. order**
compression	✔	?	✔
soundness	✔	✔	✔
completeness	✘	✔	✔
inf. strip lemma	✔	✔	✔
developments	✘	✔	✔
inf. normalisation	✘	✔	✔
inf. confluence	✘	?	?

[13] B. "Böhm Reduction in Infinitary Term Graph Rewriting Systems". In: *FSCD*. 2017.

Properties of Orthogonal GRS

property	metric	Böhm red.	part. order
compression	✔	?	✔
soundness	✔	✔	✔
completeness	✘	✔	✔
inf. strip lemma	✔	✔	✔
developments	✘	✔	✔
inf. normalisation	✘	✔	✔
inf. confluence	✘	?	?
inf. confluence modulo bisim.	✘	✔	✔

[13] B. "Böhm Reduction in Infinitary Term Graph Rewriting Systems". In: *FSCD*. 2017.

Partial Order vs. Böhm Reduction

Theorem

If $\mathcal{R}$ is an orthogonal GRS and g, h total term graphs, then

$$g \xrightarrow{p}\!\!\!\!\rightarrow_{\mathcal{R}} h \quad \textit{iff} \quad g \xrightarrow{m}\!\!\!\!\rightarrow_{\mathcal{R}} h.$$

[14] B. "Böhm Reduction in Infinitary Term Graph Rewriting Systems". In: *FSCD*. 2017.

Partial Order vs. Böhm Reduction

Theorem

If $\mathcal{R}$ is an orthogonal GRS and g, h total term graphs, then

$$g \overset{p}{\twoheadrightarrow}_{\mathcal{R}} h \quad \text{iff} \quad g \overset{m}{\twoheadrightarrow}_{\mathcal{R}} h.$$

Theorem

If $\mathcal{R}$ is an orthogonal GRS and $\mathcal{B}$ the Böhm extension of $\mathcal{R}$ (w.r.t. root-active term graphs), then

$$g \overset{p}{\twoheadrightarrow}_{\mathcal{R}} h \quad \text{iff} \quad g \overset{m}{\twoheadrightarrow}_{\mathcal{B}} h.$$

[14] B. "Böhm Reduction in Infinitary Term Graph Rewriting Systems". In: *FSCD*. 2017.

Metric on Term Graphs

Depth of a node = length of a shortest path from the root to the node.

Metric on Term Graphs

Depth of a node = length of a shortest path from the root to the node.

Truncation of term graphs

The truncation $g \dagger d$ is obtained from g by

- relabelling all nodes at depth d with $\bot$, and
- removing all nodes that thus become unreachable from the root.

Metric on Term Graphs

Depth of a node = length of a shortest path from the root to the node.

Truncation of term graphs

The truncation $g \dagger d$ is obtained from g by

- **relabelling** all nodes at depth d with $\perp$, and
- **removing** all nodes that thus become **unreachable** from the root.

Metric on term graphs

$$\mathbf{d}(g, h) = 2^{-n}$$

Where n = maximum depth d s.t. $g \dagger d \cong h \dagger d$.

A Partial Order on Term Graphs – How?

$\bot$-homomorphisms ϕ: $g \to_\bot h$

- homomorphism condition suspended on $\bot$-nodes
- allow mapping of $\bot$-nodes to arbitrary nodes

A Partial Order on Term Graphs – How?

$\bot$-homomorphisms $\phi\colon g \to_\bot h$

- homomorphism condition suspended on $\bot$-nodes
- allow mapping of $\bot$-nodes to arbitrary nodes

Proposition

For all terms s, t: $\quad s \le_\bot t \quad$ *iff* $\quad \exists \phi\colon s \to_\bot t$

A Partial Order on Term Graphs – How?

$\bot$-homomorphisms $\phi\colon\ g \to_\bot h$

- homomorphism condition suspended on $\bot$-nodes
- allow mapping of $\bot$-nodes to arbitrary nodes

Proposition

For all terms s, t: $\quad s \leq_\bot t \quad$ *iff* $\quad \exists\phi\colon\ s \to_\bot t$

Definition

For all term graphs g, h, let $g \leq_\bot h$ iff there is some $\phi\colon\ g \to_\bot h$.

Working with Term Graphs

Some Observations

- Term graphs can be messy
  - Very operational style of term graph rewriting
  - Böhm reduction is not left-linear

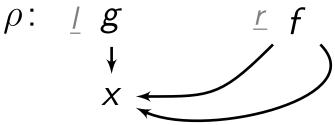
- But: sharing simplifies some things
  - Reduction produces no duplication
  - Residuals & developments are easier

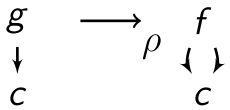
Working with Term Graphs

Some Observations

- Term graphs can be messy
  - Very operational style of term graph rewriting
  - Böhm reduction is not left-linear
- But: sharing simplifies some things
  - Reduction produces no duplication
  - Residuals & developments are easier

Example ($g(x) \to f(x, x)$)





Working with Term Graphs

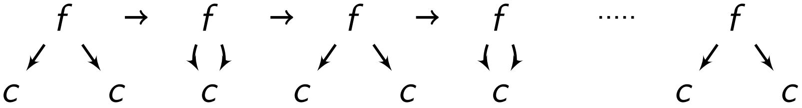
Some Observations

- Term graphs can be messy
  - Very operational style of term graph rewriting
  - Böhm reduction is not left-linear
- But: sharing simplifies some things
  - Reduction produces no duplication
  - Residuals & developments are easier
- Weak convergence is even weirder than on terms:

Working with Term Graphs

Some Observations

- Term graphs can be messy
  - Very operational style of term graph rewriting
  - Böhm reduction is not left-linear
- But: sharing simplifies some things
  - Reduction produces no duplication
  - Residuals & developments are easier
- Weak convergence is even weirder than on terms:



Böhm Reduction for Terms and Term Graphs

Confluence in Infinitary Rewriting

Patrick Bahr

IT University of Copenhagen