Confluence of bang modulo

Giulio Guerrieri & Vincent van Oostrom

$\beta!$

Unveil hidden $\beta!$ redex-patterns by σ

Saturate $\beta!$ redex-patterns into $B!$

Beta bang ($\beta!$) as reified replication

Idea: reification

reify replicability in the β-rule of the λ-calculus

Beta bang

Idea: reification

reify replicability in the β-rule of the λ-calculus
(introduce symbol into the signature to signify replicability / replication)

Beta bang

Idea: reification

reify replicability in the β-rule of the λ-calculus

Beta bang

Idea: reification

reify replicability in the β-rule of the λ-calculus

Example (replicability for β)

$(\lambda x.z)\, y \to_\beta z$ erases y

$(\lambda x.x\, x)\, y \to_\beta y\, y$ duplicates y

Beta bang

Idea: reification

reify replicability in the β-rule of the λ-calculus

Example (replicability reified in $\beta!$)

$(\lambda x.z)\,!y \to_{\beta!} z$ erases replicable y

$(\lambda x.x\,x)\,!y \to_{\beta!} y\,y$ duplicates replicable y

Beta bang

Idea: reification

reify replicability in the β-rule of the λ-calculus

Example (replicability reified in $\beta!$)

$(\lambda x.z)\,!y \to_{\beta!} z$ erases replicable y
$(\lambda x.x\,x)\,!y \to_{\beta!} y\,y$ duplicates replicable y

signature extensions due to reification omnipresent:

Example (collapsingness)

$g(f(x)) \to x$ collapsing (obstacle, Middeldorp 90)

Beta bang

Idea: reification

reify replicability in the β-rule of the λ-calculus

Example (replicability reified in $\beta!$)

$(\lambda x.z)\,!y \to_{\beta!} z$ erases replicable y
$(\lambda x.x\,x)\,!y \to_{\beta!} y\,y$ duplicates replicable y

signature extensions due to reification omnipresent:

Example (collapsingness reified in e)

$g(f(x)) \to e(x)$ hiaton Park 83 / edge Terese 03

Beta bang

Idea: reification

reify replicability in the β-rule of the λ-calculus

Example (replicability reified in $\beta!$)

$(\lambda x.z)\,!y \to_{\beta!} z$ erases replicable y
$(\lambda x.x\,x)\,!y \to_{\beta!} y\,y$ duplicates replicable y

signature extensions due to reification omnipresent:

Example (erasure)

$(\lambda x.z)\,y \to_{\beta} z$ erasing

Beta bang

Idea: reification

reify replicability in the β-rule of the λ-calculus

Example (replicability reified in $\beta!$)

$(\lambda x.z)\,!y \to_{\beta!} z$ erases replicable y
$(\lambda x.x\,x)\,!y \to_{\beta!} y\,y$ duplicates replicable y

signature extensions due to reification omnipresent:

Example (erasure by β reified in [,])

$(\lambda x.z)\,y \to_{[\beta]} [z, y]$ scar Nederpelt 73 / memory Klop 80

Beta bang

Idea: reification

reify replicability in the β-rule of the λ-calculus

Example (replicability reified in $\beta!$)

$(\lambda x.z)\,!y \to_{\beta!} z$ erases replicable y
$(\lambda x.x\,x)\,!y \to_{\beta!} y\,y$ duplicates replicable y

signature extensions due to reification omnipresent:

Example (β-redex-pattern match)

$(\lambda x.x\,x)\,z \to_{\beta} z\,z$ β-redex-pattern match

Beta bang

Idea: reification

reify replicability in the β-rule of the λ-calculus

Example (replicability reified in $\beta!$)

$(\lambda x.z)\,!y \to_{\beta!} z$ erases replicable y
$(\lambda x.x\,x)\,!y \to_{\beta!} y\,y$ duplicates replicable y

signature extensions due to reification omnipresent:

Example (β-redex-pattern match reified in β)

$\mathrm{beta}(x.x\,x, z) \to_{\mathrm{beta}} z\,z$ let Scott 69 / definition de Bruijn 68– / explicit substitution Bloo & Rose 96 / rule-symbol beta Terese 03

Beta bang

Idea: reification

reify replicability in the β-rule of the λ-calculus

Example (replicability reified in $\beta!$)

$(\lambda x.z)\,!y \to_{\beta!} z$ erases replicable y
$(\lambda x.x\,x)\,!y \to_{\beta!} y\,y$ duplicates replicable y

signature extensions due to reification omnipresent:

Example (Curry typing)

$\lambda x.5 + x$ has type $\mathbb{N} \to \mathbb{N}$ Curry typing (external)

Beta bang

Idea: reification

reify replicability in the β-rule of the λ-calculus

Example (replicability reified in $\beta!$)

$(\lambda x.z)\,!y \to_{\beta!} z$ erases replicable y
$(\lambda x.x\,x)\,!y \to_{\beta!} y\,y$ duplicates replicable y

signature extensions due to reification omnipresent:

Example (Curry typing reified in Church typing)

$\lambda x : \mathbb{N}.5 + x$ Church typing (internal)

Beta bang

Idea: reification

reify replicability in the β-rule of the λ-calculus

Example (replicability reified in $\beta!$)

$(\lambda x.z)\,!y \to_{\beta!} z$ erases replicable y
$(\lambda x.x\,x)\,!y \to_{\beta!} y\,y$ duplicates replicable y

signature extensions due to reification omnipresent:

Example (Curry typing reified in Church typing)

$\lambda x : \mathbb{N}.5 + x$ Church typing (internal)

examples abound (λ, at-a-distance, . . .); syntax as reified ideas; signature?

Beta bang as a term rewrite system

Definition (rewrite system for β in usual presentation (Barendregt 84))

terms defined by grammar

$$\Lambda \quad \ni \quad s, t ::= x \mid \lambda x.t \mid s\,t$$

rewrite steps defined by applying rule schema in all contexts

$$(\lambda x.t)\,s \to t\{s/x\}$$

for all terms t and s and variables x, with substitution defined at meta-level

Beta bang as a term rewrite system

Definition (rewrite system for $\beta!$ in usual presentation (Ehrhard 16))

terms defined by grammar

$$!\Lambda \quad \ni \quad s, t ::= x \mid \lambda x.t \mid s\,t \mid !t$$

rewrite steps defined by applying rule schema in all contexts

$$(\lambda x.t)\,!s \to t\{s/x\}$$

for all terms t and s and variables x, with substitution defined at meta-level

Example (Smullyan's mockingbird $\omega := \lambda x.x\,!x$)

$\Omega \to_{\beta!} \Omega$ for $\Omega := \omega\,!\omega$ (in general: embedding $\Lambda \hookrightarrow\ !\Lambda$ using $M\,N \mapsto M\,!N$)

Beta bang as a term rewrite system

Definition (rewrite system for $\beta!$ in usual presentation (Ehrhard 16))

terms defined by grammar

$$!\Lambda \quad \ni \quad s, t ::= x \mid \lambda x.t \mid s\,t$$

rewrite steps defined by applying rule schema in all contexts

$$(\lambda x.t)\,!s \to t\{s/x\}$$

for all terms t and s and variables x, with substitution defined at meta-level

usual presentation drawbacks: grammar, rule schema (infinitary), adapt theory

Beta bang as a term rewrite system

Definition (rewrite system for $\beta!$ in usual presentation (Ehrhard 16))

terms defined by grammar

$$!\Lambda \quad \ni \quad s, t ::= x \mid \lambda x.t \mid s\,t$$

rewrite steps defined by applying rule schema in all contexts

$$(\lambda x.t)\,!s \to t\{s/x\}$$

for all terms t and s and variables x, with substitution defined at meta-level

usual presentation drawbacks: grammar, rule schema, adapt theory
PRS (pattern rewrite system, Nipkow 91): signature, rule, instantiate theory

Beta bang as a term rewrite system

Definition (rewrite system for $\beta!$ in usual presentation (Ehrhard 16))

terms defined by grammar

$$!\Lambda \quad \ni \quad s, t ::= x \mid \lambda x.t \mid s\,t$$

rewrite steps defined by applying rule schema in all contexts

$$(\lambda x.t)\,!s \to t\{s/x\}$$

for all terms t and s and variables x, with substitution defined at meta-level

usual presentation drawbacks: grammar, rule schema, adapt theory
PRS presentation advantages: signature, rule, instantiate theory

Beta bang as a term rewrite system

Definition (PRS for $\beta!$)

rule $\beta! : (\lambda x.S(x))\,!T \to S(T)$ on terms over signature for base type term
$\{$ abs : $(\text{term} \to \text{term}) \to \text{term}$, app : $\text{term} \to \text{term} \to \text{term}$, bang : $\text{term} \to \text{term}$ $\}$

using usual notation as syntactic sugar, *e.g.*, $\lambda \mapsto$ abs and $! \mapsto$ bang
PRS presentation advantages: signature, rule, instantiate theory?

Beta bang as a term rewrite system

Definition (PRS for $\beta!$)

rule $\beta! : (\lambda x.S(x))\,!T \to S(T)$ on terms over signature for base type term
$\{$ abs : $(\text{term} \to \text{term}) \to \text{term}$, app : $\text{term} \to \text{term} \to \text{term}$, bang : $\text{term} \to \text{term}$ $\}$

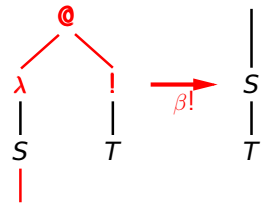
PRS presentation advantages: signature, rule, instantiate theory
orthogonal PRS: Finite Developments (FD) $\implies$ confluent (CR) $\implies$ consistent

Beta bang as a term rewrite system

Definition (PRS for $\beta!$)

rule $\beta! : (\lambda x.S(x))\,!T \to S(T)$ on terms over signature for base type term
$\{$ abs : $(\text{term} \to \text{term}) \to \text{term}$, app : $\text{term} \to \text{term} \to \text{term}$, bang : $\text{term} \to \text{term}$ $\}$

transformation on abstract binding trees (ABT), substituting T for x occurring in S

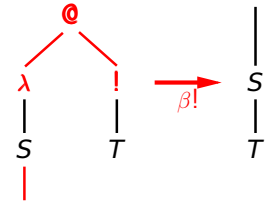


Beta bang as a term rewrite system

Definition (PRS for $\beta!$)

rule $\beta! : (\lambda x.S(x))\,!T \to S(T)$ on terms over signature for base type term
$\{$ abs : $(\text{term} \to \text{term}) \to \text{term}$, app : $\text{term} \to \text{term} \to \text{term}$, bang : $\text{term} \to \text{term}$ $\}$

transformation on abstract binding trees (ABT), substituting T for x occurring in S



notation $\to := \to_{\beta!}$; second-order; link in paper to confluence tool format

How to avoid getting prematurely stuck?

Example (collapsing $f(g(x)) \to x \implies$ non-collapsing $f(g(x)) \to e(x)$)

lifting $f(f(g(g(a)))) \to f(g(a)) \to$ a gets stuck: $f(f(g(g(a)))) \to f(e(g(a))) \to ?$

two standard solutions: modulo and saturation

How to avoid getting prematurely stuck?

Example (collapsing $f(g(x)) \to x \implies$ non-collapsing $f(g(x)) \to e(x)$)

lifting $f(f(g(g(a)))) \to f(g(a)) \to$ a gets stuck: $f(f(g(g(a)))) \to f(e(g(a))) \to$?

two standard solutions: modulo and saturation

rewriting $\to$ relative to / modulo $\rightsquigarrow$

extra directed modulo rule shift-in $e(g(x)) \rightsquigarrow g(e(x))$ / shift-out $f(e(x)) \rightsquigarrow e(f(x))$
lifting progresses modulo: $f(f(g(g(a)))) \to f(e(g(a))) \rightsquigarrow f(g(e(a))) \to e(e(a))$

advantage: finite, local; disadvantage: $\to$ relative to / modulo $\rightsquigarrow$

How to avoid getting prematurely stuck?

Example (collapsing $f(g(x)) \to x \implies$ non-collapsing $f(g(x)) \to e(x)$)

lifting $f(f(g(g(a)))) \to f(g(a)) \to a$ gets stuck: $f(f(g(g(a)))) \to f(e(g(a))) \to ?$

two standard solutions: modulo and saturation

rewriting $\to$ relative to / modulo $\rightsquigarrow$

- shifting memory $[\,,\,]$ in / out (Klop 80, De Groote 93, Kfoury & Wells 95)
- scope-extrusion for λ (Hendriks & Van Oostrom 03)
- ...

advantage: finite, local; disadvantage: $\to$ relative to / modulo $\rightsquigarrow$

How to avoid getting prematurely stuck?

Example (collapsing $f(g(x)) \to x \implies$ non-collapsing $f(g(x)) \to e(x)$)

lifting $f(f(g(g(a)))) \to f(g(a)) \to$ a gets stuck: $f(f(g(g(a)))) \to f(e(g(a))) \to ?$

two standard solutions: modulo and saturation

saturating lhss / rules of $\to$

$f(e(g(x))) \to e(e(x))$, $f(e(e(g(x)))) \to e(e(e(x)))$, $f(e(e(e(g(x))))) \to e(e(e(e(x))))$,...
lifting progresses: $f(f(g(g(a)))) \to f(e(g(a))) \to e(e(a))$

advantage: $\to$; disadvantage: infinite, non-local

How to avoid getting prematurely stuck?

Example (collapsing $f(g(x)) \to x \implies$ non-collapsing $f(g(x)) \to e(x)$)

lifting $f(f(g(g(a)))) \to f(g(a)) \to$ a gets stuck: $f(f(g(g(a)))) \to f(e(g(a))) \to$?

two standard solutions: modulo and saturation

saturating lhss / rules of $\to$

- mini-reduction (De Bruijn 87); AT-couple $S ::= \epsilon \mid @S\lambda \mid SS$ in $@S\lambda$ context-free
- λ-calculus (Hendriks & Van Oostrom 03); end-of-scope $@\lambda^*\lambda$ regular
- structural λ-calculus (Accattoli & Kesner 10); at-a-distance $@let^*\lambda$ regular

advantage: $\to$; disadvantage: infinite, non-local

How to unveil hidden redex-patterns

Example (hidden redex-pattern)

$\omega\,((\lambda x.!\omega)\,(y\,z))$ is in $\beta!$-normal form;

How to unveil hidden redex-patterns

Example (hidden redex-pattern)

$\omega\,((\lambda x.!\omega)\,(y\,z))$ is in β!-normal form; hides redex-pattern $\omega\,!\omega$

How to unveil hidden redex-patterns

Example (hidden redex-pattern)

$\omega\,((\lambda x.!\omega)\,(y\,z))$ is in $\beta!$-normal form; σ_3 unveils redex-pattern $\omega\,!\omega$
$\omega\,((\lambda x.!\omega)\,(y\,z)) \rightsquigarrow_{\sigma_3} (\lambda x.\omega\,!\omega)\,(y\,z) \to_{\beta!} (\lambda x.\omega\,!\omega)\,(y\,z) \to_{\beta!} \ldots$

unveiling by σ-rules is directed modulo $\rightsquigarrow$ / modulo $\sim$

How to unveil hidden redex-patterns

Example (hidden redex-pattern)

$\omega\,((\lambda x.!\omega)\,(y\,z))$ is in $\beta!$-normal form;

unveiling by σ-rules is directed modulo $\rightsquigarrow$ / modulo $\sim$

Definition (clash)

term of shape $x\,N$ or $!M\,N$ or $M\,\lambda x.N$

application-term that (visibly) cannot reduce to redex

How to unveil hidden redex-patterns

Example (hidden redex-pattern)

$\omega\,((\lambda x.!\omega)\,(y\,z))$ is in β!-normal form;

unveiling by σ-rules is directed modulo $\rightsquigarrow$ / modulo $\sim$

Definition (clash)

term of shape $x\,N$ or $!M\,N$ or $M\,\lambda x.N$

Example (hidden clash)

$(\lambda y.z)\,((\lambda x.\lambda y.z)\,((\lambda x.x)\,x))$ is β!-normal and clash-free; hides clash $(\lambda y.z)\,\lambda y.z$

How to unveil hidden redex-patterns

Example (hidden redex-pattern)

$\omega\,((\lambda x.!\omega)\,(y\,z))$ is in $\beta!$-normal form;

unveiling by σ-rules is directed modulo $\rightsquigarrow$ / modulo $\sim$

Definition (clash)

term of shape $x\,N$ or $!M\,N$ or $M\,\lambda x.N$

Example (hidden clash)

$(\lambda y.z)\,((\lambda x.\lambda y.z)\,((\lambda x.x)\,x))$ is $\beta!$-normal and clash-free; σ_2 unveils clash
$(\lambda y.z)\,((\lambda x.\lambda y.z)\,((\lambda x.x)\,x)) \rightsquigarrow_{\sigma_2} (\lambda y.z)\,\lambda y.(\lambda x.z)\,((\lambda x.x)\,x)$

How to unveil hidden redex-patterns

two standard solutions: modulo and saturation

Example (hidden redex-pattern)

$\omega\,((\lambda x.!\omega)\,(y\,z))$ is in $\beta!$-normal form;

unveiling by σ-rules is directed modulo $\rightsquigarrow$ / modulo $\sim$

Definition (clash)

term of shape $x\,N$ or $!M\,N$ or $M\,\lambda x.N$

Example (hidden clash)

$(\lambda y.z)\,((\lambda x.\lambda y.z)\,((\lambda x.x)\,x))$ is $\beta!$-normal and clash-free;

modulo σ to unveil hidden redex-patterns; saturating $\beta!$ to $B!$ to prove that

Beta bang modulo sigma ($\beta!$ modulo σ)

Definition (PRS for σ (Ehrhard & Guerrieri 16))

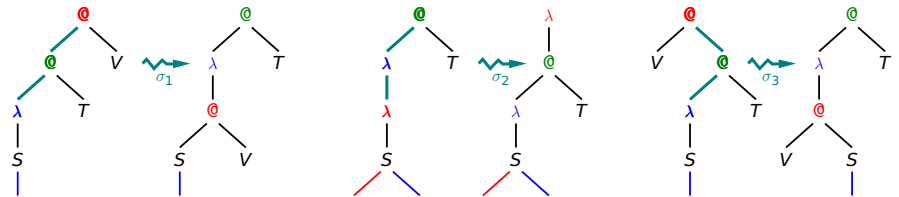
$$\sigma_1 : (\lambda x.S(x))\,T\,V \quad \rightsquigarrow \quad (\lambda x.S(x)\,V)\,T$$
$$\sigma_2 : (\lambda x.\lambda y.S(x,y))\,T \quad \rightsquigarrow \quad \lambda y.(\lambda x.S(x,y))\,T$$
$$\sigma_3 : V\,((\lambda x.S(x))\,T) \quad \rightsquigarrow \quad (\lambda x.V\,S(x))\,T$$

Beta bang modulo sigma ($\beta!$ modulo σ)

Definition (PRS for σ)

$$\sigma_1 : (\lambda x.S(x))\,T\,V \quad \rightsquigarrow \quad (\lambda x.S(x)\,V)\,T$$
$$\sigma_2 : (\lambda x.\lambda y.S(x,y))\,T \quad \rightsquigarrow \quad \lambda y.(\lambda x.S(x,y))\,T$$
$$\sigma_3 : V\,((\lambda x.S(x))\,T) \quad \rightsquigarrow \quad (\lambda x.V\,S(x))\,T$$

notation $\sigma := \{\sigma_1, \sigma_2, \sigma_3\}$ and $\sim := \leftrightsquigarrow_\sigma^*$ (modulo)

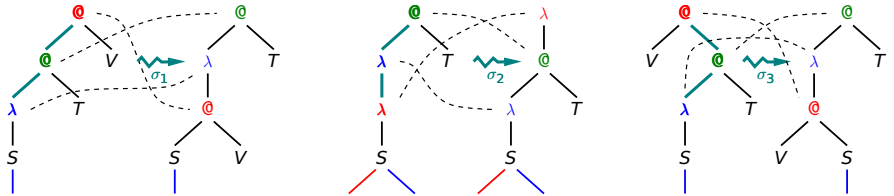


Beta bang modulo sigma ($\beta!$ modulo σ)

Definition (PRS for σ)

$$\sigma_1 : (\lambda x.S(x))\,T\,V \quad \rightsquigarrow \quad (\lambda x.S(x)\,V)\,T$$
$$\sigma_2 : (\lambda x.\lambda y.S(x,y))\,T \quad \rightsquigarrow \quad \lambda y.(\lambda x.S(x,y))\,T$$
$$\sigma_3 : V\,((\lambda x.S(x))\,T) \quad \rightsquigarrow \quad (\lambda x.V\,S(x))\,T$$

notation $\sigma := \{\sigma_1, \sigma_2, \sigma_3\}$ and $\sim := \leftrightsquigarrow^*_\sigma$ (modulo); chemical rules (Terese 03)



Beta bang modulo sigma ($\beta!$ modulo σ)

Definition (PRS for σ)

$$\begin{aligned} \sigma_1 &: (\lambda x.S(x))\,T\,V && \rightsquigarrow && (\lambda x.S(x)\,V)\,T \\ \sigma_2 &: (\lambda x.\lambda y.S(x,y))\,T && \rightsquigarrow && \lambda y.(\lambda x.S(x,y))\,T \\ \sigma_3 &: V\,((\lambda x.S(x))\,T) && \rightsquigarrow && (\lambda x.V\,S(x))\,T \end{aligned}$$

notation $\sigma := \{\sigma_1, \sigma_2, \sigma_3\}$ and $\sim := \leftrightsquigarrow_\sigma^*$ (modulo); chemical rules; symbols trace

Open problem

can $\sim$ be presented by a complete (confluent and terminating) PRS?

for any orientation non-joinable critical peaks; (first-order) completion tools fail
$\sim$ decidable because equivalence classes are finite

Confluence of $\rightarrow$ modulo $\sim$ (of $\beta!$ modulo σ)

From $\rightarrow$ to $\multimap$

confluence of β shown via diamond property of complete developments $\multimap$ develops set of redex-patterns until no residuals remain (Church & Rosser 36)

$\multimap$ has diamond property, developments finite, $\rightarrow \subseteq \multimap \subseteq \twoheadrightarrow$

Confluence of $\multimap$ modulo $\sim$

From $\to$ to $\multimap$

confluence of β! shown via diamond property of complete developments $\multimap$
develops set of redex-patterns until no residuals remain (OPRS Klop 80, 94)

$\multimap$ has diamond property, developments finite, $\to \subseteq \multimap \subseteq \twoheadrightarrow$

Confluence of $\multimap$ modulo $\sim$

From $\to$ to $\multimap$

confluence of β! shown via diamond property of complete developments $\multimap$

Theorem

$[\multimap]$ *is confluent on* $\sim$*-classes, where* $[a]\ [\multimap]\ [b] := a \sim \cdot \multimap \cdot \sim b$

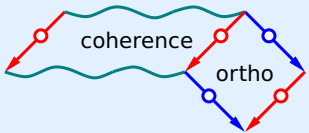
Confluence of $\multimap$ modulo $\sim$

Theorem

$[\multimap]$ is confluent on $\sim$-classes, where $[a]\ [\multimap]\ [b] := a \sim \cdot \multimap \cdot \sim b$

Proof idea .

diamond property of $\sim \cdot \multimap$ through that of $\multimap$ via coherence $\sim \cdot \multimap \subseteq \multimap \cdot \sim$



□

coherence aka postponement, preponement, factorisation, commutation (20)

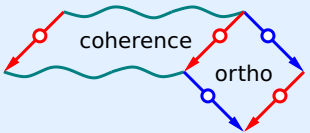
Confluence of $\multimap$ modulo $\sim$

Theorem

$[\multimap]$ *is confluent on* $\sim$*-classes, where* $[a]\ [\multimap]\ [b] := a \sim \cdot \multimap \cdot \sim b$

Proof idea ✗.

diamond property of $\sim \cdot \multimap$ through that of $\multimap$ via coherence $\sim \cdot \multimap \subseteq \multimap \cdot \sim$

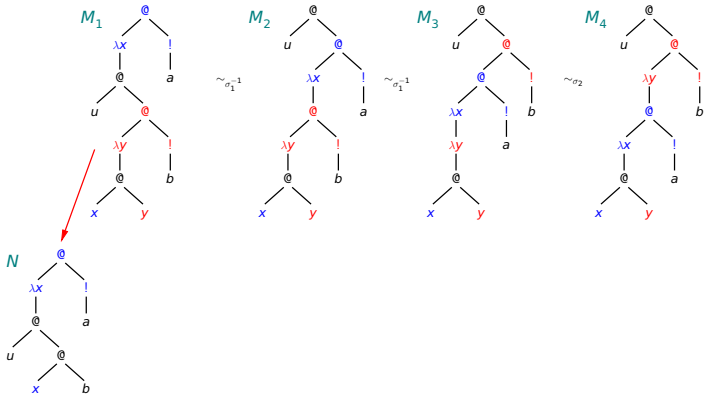


□

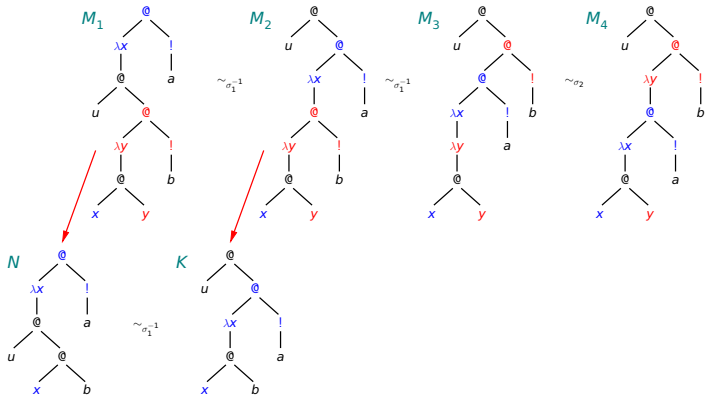
Example (already $\to$ does not cohere with $\sim$)

$M_3 \sim M_1 \to_{\beta!} N$, but for no P do we have $M_3 \to_{\beta!} P \sim N$ (picture next slide)

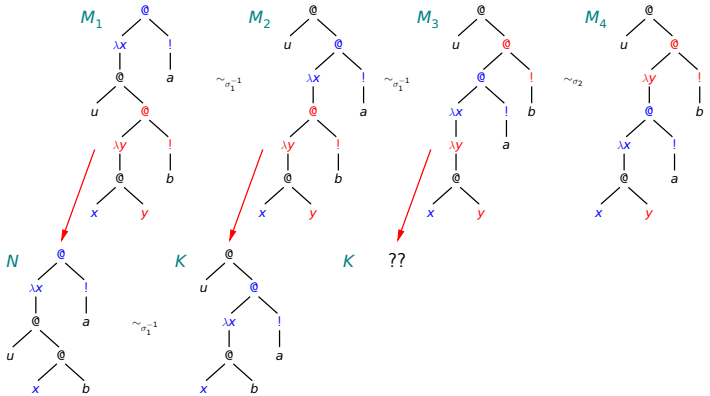
Non-coherence due to divorcing @ from λ by σ_1^{-1} in M_3



Non-coherence due to divorcing @ from λ by σ_1^{-1} in M_3



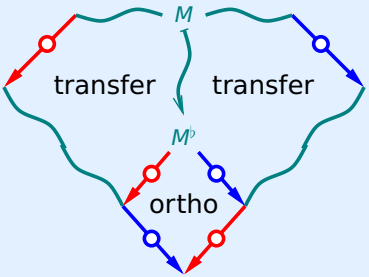
Non-coherence due to divorcing @ from λ by σ_1^{-1} in M_3



Diamond property of $\sim \cdot \multimap\!\!\to$ through transfer

Proof.

by flattening source M to $M^\flat$, transfer twice, and orthogonality of B!



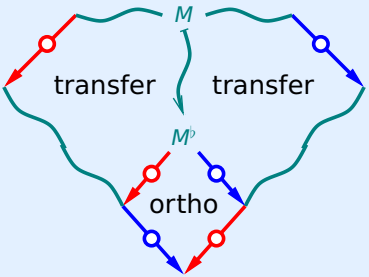
□

transfer $\sim \cdot \multimap\!\!\to \subseteq \multimap\!\!\to \cdot \sim$ from flat terms (M_3 is not flat; $M^\flat$ is)

Diamond property of $\sim \cdot \multimap$ through transfer

Proof.

by flattening source M to $M^\flat$, transfer twice, and orthogonality of B!



□

remains to show transfer $\sim \cdot \multimap \subseteq \multimap \cdot \sim$ from flat terms (M_3 is not flat; $M^\flat$ is)

Saturating β! into *B*!

Saturation

insert @–λ pairs between @ and matching λ or !

Saturating $\beta!$ into $B!$

Definition (colored $B!$ for colors i)

$$B!^{i} \quad : \quad (d[\lambda^{i} x.S(x, \vec{d})])^{i} e[!^{i} T(\vec{e})] \to e[d[S(T(\vec{e}), \vec{d})]]$$

for Dyck spine grammar for single-hole contexts $d, e ::= \Box \mid (d[\lambda^{i} x.\Box])^{i} M \mid d[e]$

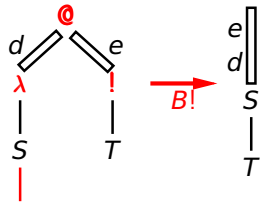
CFG for language of (colored) matching parentheses, where $\lambda =)$ and $@ = ($

Saturating $\beta!$ into $B!$

Definition (colored $B!$ for colors i)

$$B!^i \quad : \quad (d[\lambda^i x.S(x,\vec{d})])^i e[!^i T(\vec{e})] \to e[d[S(T(\vec{e}),\vec{d})]]$$

for Dyck spine grammar for single-hole contexts $d, e ::= \Box \mid (d[\lambda^i x.\Box])^i M \mid d[e]$



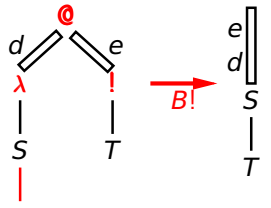
if $d = \Box = e$ then $B! = \beta!$ (flat); term is flat if all $B!$ redexes are

Saturating $\beta!$ into $B!$

Definition (colored $B!$ for colors i)

$$B!^i \quad : \quad (d[\lambda^i x.S(x,\vec{d})])^i e[!^i T(\vec{e})] \to e[d[S(T(\vec{e}),\vec{d})]]$$

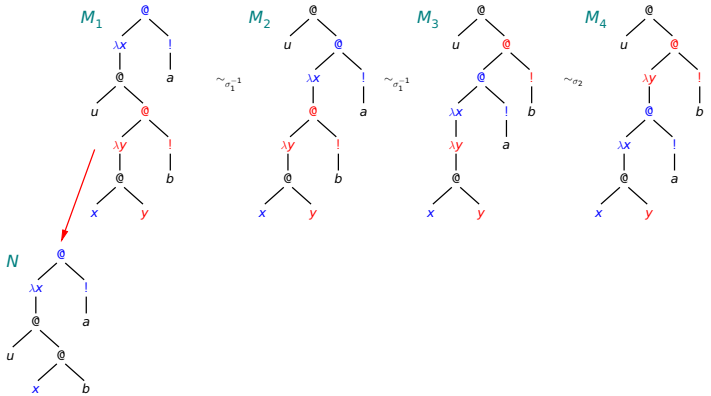
for Dyck spine grammar for single-hole contexts $d, e ::= \Box \mid (d[\lambda^i x.\Box])^i M \mid d[e]$



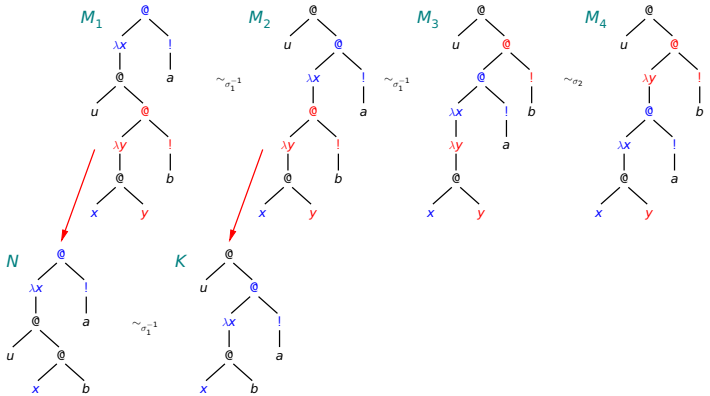
Proposition (main)

$\to_{B!}$ *coheres with* $\sim$

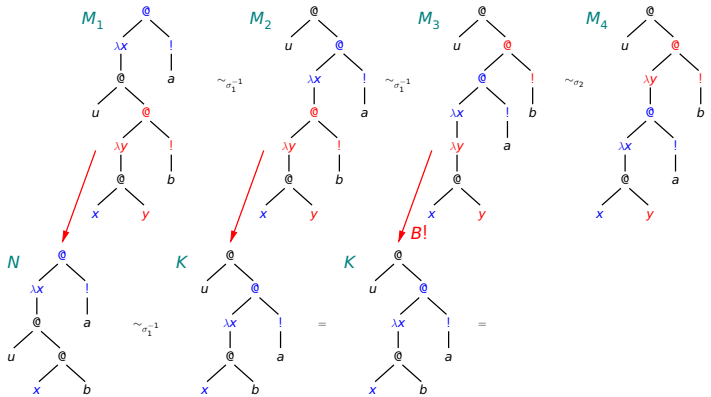
Coherence of $B!$ with σ by example



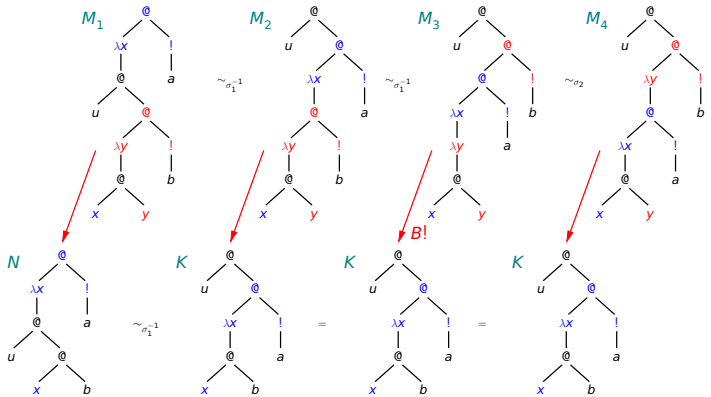
Coherence of $B!$ with σ by example



Coherence of $B!$ with σ by example



Coherence of $B!$ with σ by example



Flattening

Another characterisation of Dyck language D (matching parentheses

$w \in D$ iff its normal form w.r.t. $(() \rightsquigarrow ()($ is in $D^\flat := ()^*$ (list of monochrome pairs)

Flattening

Another characterisation of Dyck language D

$w \in D$ iff its normal form w.r.t. $(() \rightsquigarrow ()($ is in $D^\flat := ()^*$

Example

$((())) \rightsquigarrow (()()) \rightsquigarrow ()(()) \rightsquigarrow ()()()$ ✓ (sequence of monochrome () pairs)

$(()((\rightsquigarrow ()(((\ $ ✗ (non-matched parentheses in $(((()$

$(()) \rightsquigarrow ()()\ $ ✗ (non-matching colors in $())$)

Flattening

Another characterisation of Dyck language D

$w \in D$ iff its normal form w.r.t. $(() \rightsquigarrow ()($ is in $D^\flat := ()^*$

Example

$((())) \rightsquigarrow (()()) \rightsquigarrow ()(()) \rightsquigarrow ()()()$ ✓ (sequence of monochrome () pairs)

$(()((\rightsquigarrow ()(((\ $ ✗ (non-matched parentheses in $(((()$

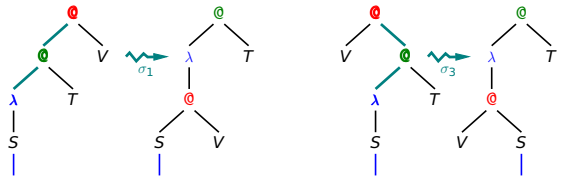
$(()) \rightsquigarrow ()()\ $ ✗ (non-matching colors in $())$)

characterise context-free language D via regular language $D^\flat$ (preserves length)

Flattening

Definition

flattening is σ_1, σ_3 $\rightsquigarrow$-normalisation



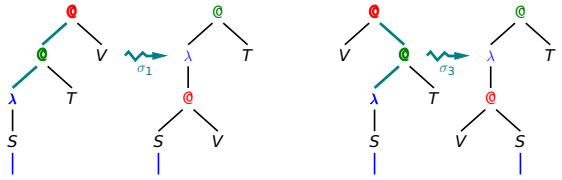
$\rightsquigarrow$ normalising for same reason as for D; () pairs are ‘pushed up’

normal forms need not be unique (@ offers choice which pair to ‘push up’ first)

Flattening

Definition

flattening is σ_1, σ_3 $\rightsquigarrow$-normalisation



Lemma

flattening yields flat terms

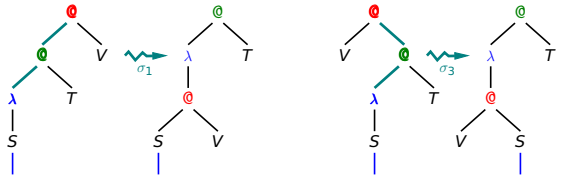
Proof.

$\rightsquigarrow$-normal forms do not have nested @–λ-pairs as for D □

Flattening

Definition

flattening is σ_1, σ_3 $\rightsquigarrow$-normalisation



Lemma

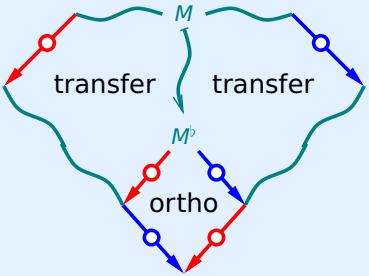
flattening yields flat terms

idea: $\rightsquigarrow$-flattening M to $M^\flat$ transfers $\rightarrow_{B!}$-steps from M to $\rightarrow_{\beta!}$-steps from $M^\flat$

Diamond property of $\sim \cdot \multimap$ through transfer

Proof.

by flattening source M to $M^\flat$, transfer twice, and orthogonality of B!



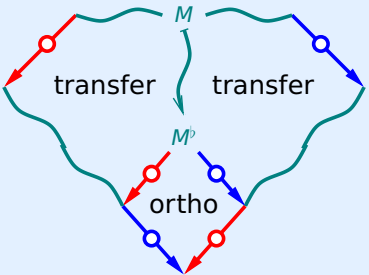
□

transfer $\sim \cdot \multimap \subseteq \multimap \cdot \sim$ from flat terms ($M^\flat$ is flat)

Diamond property of $\sim \cdot \multimap$ through transfer

Proof.

by flattening source M to $M^\flat$, transfer twice, and orthogonality of B!



□

remains to show transfer $\sim \cdot \multimap \subseteq \multimap \cdot \sim$ from flat terms ($M^\flat$ is flat)

Transfer

Lemma

if $M^\flat$ is flat in $M^\flat \sim M \multimap N$, then $M^\flat \multimap N' \sim N$ for some N'

Transfer

Lemma

if $M^\flat$ is flat in $M^\flat \sim M \multimap N$, then $M^\flat \multimap N' \sim N$ for some N'

Proof.

- $M^\flat \sim M \multimap N$

Transfer

Lemma

if $M^\flat$ is flat in $M^\flat \sim M \multimap N$, then $M^\flat \multimap N' \sim N$ for some N'

Proof.

(i) $M^\flat \sim M \multimap N$

(ii) $M^\flat \sim M \multimap N$ by $\mathcal{C}$-coloring redex-patterns completely developed in M to M

Transfer

Lemma

if $M^\flat$ is flat in $M^\flat \sim M \multimap N$, then $M^\flat \multimap N' \sim N$ for some N'

Proof.

(i) $M^\flat \sim M \multimap N$

(ii) $M^\flat \sim M \multimap N$ by $\mathcal{C}$-coloring redex-patterns completely developed in M to M

(iii) $M^\flat \sim M \twoheadrightarrow N$ by (arbitrarily) developing $\multimap$ into a $\to$-reduction

Transfer

Lemma

if $M^\flat$ is flat in $M^\flat \sim M \multimap N$, then $M^\flat \multimap N' \sim N$ for some N'

Proof.

(i) $M^\flat \sim M \multimap N$

(ii) $M^\flat \sim M \multimap N$ by $\mathcal{C}$-coloring redex-patterns completely developed in M to M

(iii) $M^\flat \sim M \twoheadrightarrow N$ by (arbitrarily) developing $\multimap$ into a $\to$-reduction

(iv) $M^\flat \sim M \twoheadrightarrow_{B!} N$ since $\beta!$-steps are (flat) $B!$-steps

Transfer

Lemma

if $M^\flat$ is flat in $M^\flat \sim M \multimap N$, then $M^\flat \multimap N' \sim N$ for some N'

Proof.

(i) $M^\flat \sim M \multimap N$

(ii) $M^\flat \sim M \multimap N$ by $\mathcal{C}$-coloring redex-patterns completely developed in M to M

(iii) $M^\flat \sim M \twoheadrightarrow N$ by (arbitrarily) developing $\multimap$ into a $\to$-reduction

(iv) $M^\flat \sim M \twoheadrightarrow_{B!} N$ since $\beta!$-steps are (flat) $B!$-steps

(v) $M^\flat \twoheadrightarrow_{B!} N' \sim N$ for some N' by repeated coherence of $B!$ with σ

Transfer

Lemma

if $M^\flat$ is flat in $M^\flat \sim M \multimap N$, then $M^\flat \multimap N' \sim N$ for some N'

Proof.

(i) $M^\flat \sim M \multimap N$

(ii) $M^\flat \sim M \multimap N$ by $\mathcal{C}$-coloring redex-patterns completely developed in M to M

(iii) $M^\flat \sim M \twoheadrightarrow N$ by (arbitrarily) developing $\multimap$ into a $\to$-reduction

(iv) $M^\flat \sim M \twoheadrightarrow_{B!} N$ since $\beta!$-steps are (flat) $B!$-steps

(v) $M^\flat \twoheadrightarrow_{B!} N' \sim N$ for some N' by repeated coherence of $B!$ with σ

(vi) $M^\flat \twoheadrightarrow_{B!} N' \sim N$ since $\sim$ retains color and N is not colored neither is N'

Transfer

Lemma

if $M^\flat$ is flat in $M^\flat \sim M \multimap N$, then $M^\flat \multimap N' \sim N$ for some N'

Proof.

(i) $M^\flat \sim M \multimap N$

(ii) $M^\flat \sim M \multimap N$ by $\mathcal{C}$-coloring redex-patterns completely developed in M to M

(iii) $M^\flat \sim M \twoheadrightarrow N$ by (arbitrarily) developing $\multimap$ into a $\to$-reduction

(iv) $M^\flat \sim M \twoheadrightarrow_{B!} N$ since $\beta!$-steps are (flat) $B!$-steps

(v) $M^\flat \twoheadrightarrow_{B!} N' \sim N$ for some N' by repeated coherence of $B!$ with σ

(vi) $M^\flat \twoheadrightarrow_{B!} N' \sim N$ since $\sim$ retains color and N is not colored neither is N'

(vii) $M^\flat \twoheadrightarrow_{\beta!} N' \sim N$ since (residuals of) redex-patterns in $M^\flat$ are flat

Transfer

Lemma

if $M^\flat$ is flat in $M^\flat \sim M \multimap N$, then $M^\flat \multimap N' \sim N$ for some N'

Proof.

(i) $M^\flat \sim M \multimap N$

(ii) $M^\flat \sim M \multimap N$ by $\mathcal{C}$-coloring redex-patterns completely developed in M to M

(iii) $M^\flat \sim M \twoheadrightarrow N$ by (arbitrarily) developing $\multimap$ into a $\to$-reduction

(iv) $M^\flat \sim M \twoheadrightarrow_{B!} N$ since $\beta!$-steps are (flat) $B!$-steps

(v) $M^\flat \twoheadrightarrow_{B!} N' \sim N$ for some N' by repeated coherence of $B!$ with σ

(vi) $M^\flat \twoheadrightarrow_{B!} N' \sim N$ since $\sim$ retains color and N is not colored neither is N'

(vii) $M^\flat \twoheadrightarrow_{\beta!} N' \sim N$ since (residuals of) redex-patterns in $M^\flat$ are flat

(viii) $M^\flat \multimap N' \sim N$ by definition of complete development

Transfer

Lemma

if $M^\flat$ is flat in $M^\flat \sim M \multimap N$, then $M^\flat \multimap N' \sim N$ for some N'

Proof.

(i) $M^\flat \sim M \multimap N$

(ii) $M^\flat \sim M \multimap N$ by $\mathcal{C}$-coloring redex-patterns completely developed in M to M

(iii) $M^\flat \sim M \twoheadrightarrow N$ by (arbitrarily) developing $\multimap$ into a $\to$-reduction

(iv) $M^\flat \sim M \twoheadrightarrow_{B!} N$ since $\beta!$-steps are (flat) $B!$-steps

(v) $M^\flat \twoheadrightarrow_{B!} N' \sim N$ for some N' by repeated coherence of $B!$ with σ

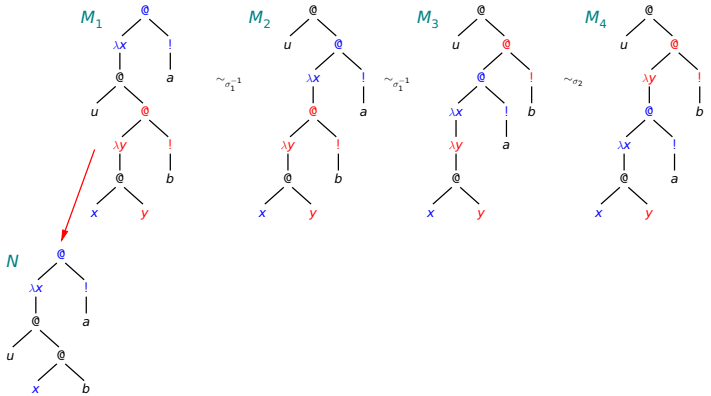
(vi) $M^\flat \twoheadrightarrow_{B!} N' \sim N$ since $\sim$ retains color and N is not colored neither is N'

(vii) $M^\flat \twoheadrightarrow_{\beta!} N' \sim N$ since (residuals of) redex-patterns in $M^\flat$ are flat

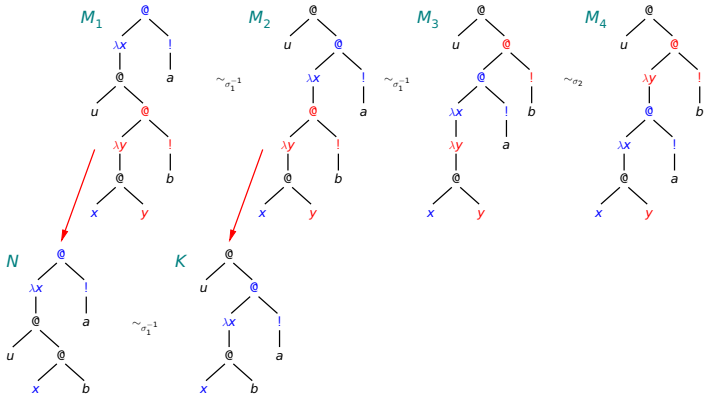
(viii) $M^\flat \multimap N' \sim N$ by definition of complete development

(ix) $M^\flat \multimap N' \sim N$ by decoloring □

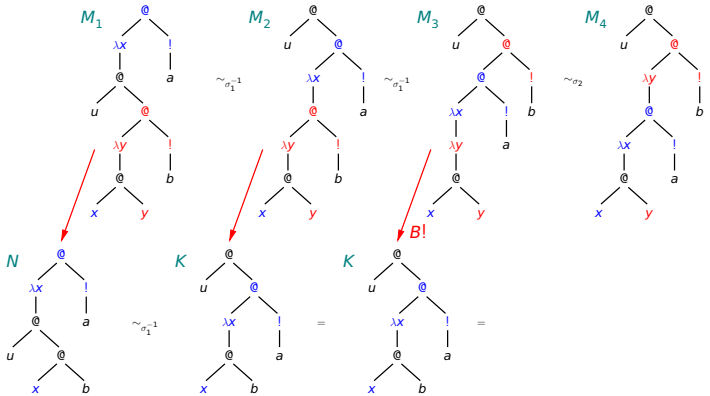
Diamond property of $\rightarrow \cdot \sim$ in a simple example



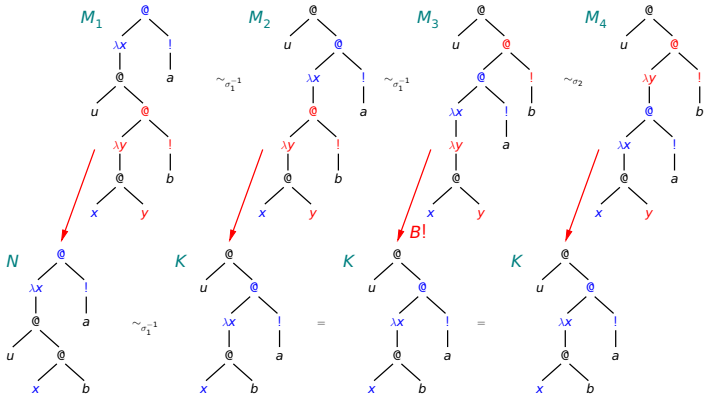
Diamond property of $\rightarrow \cdot \sim$ in a simple example



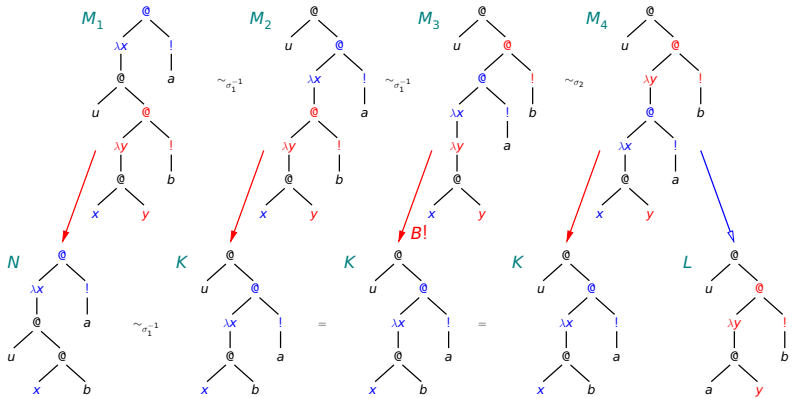
Diamond property of $\rightarrow \cdot \sim$ in a simple example



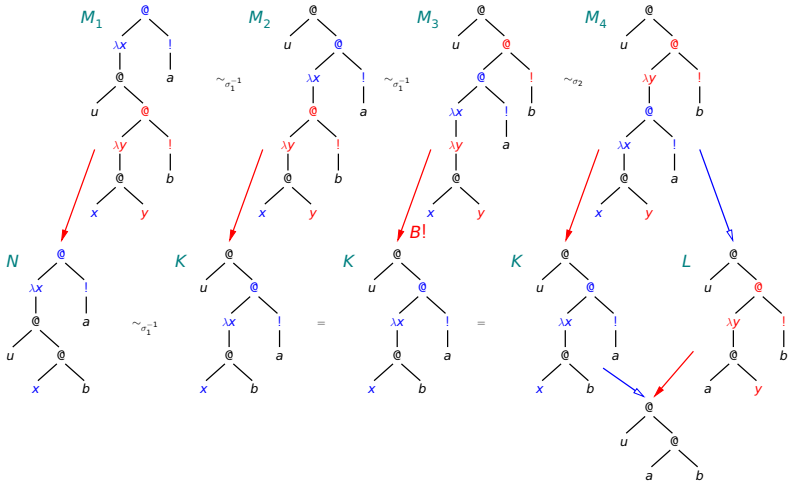
Diamond property of $\rightarrow \cdot \sim$ in a simple example



Diamond property of $\rightarrow \cdot \sim$ in a simple example



Diamond property of $\rightarrow \cdot \sim$ in a simple example



Contributions and future work

Contributions

- flattening $\rightsquigarrow$ giving not-necessarily-unique flat representatives for $\sim$

Contributions and future work

Contributions

- flattening $\rightsquigarrow$ giving not-necessarily-unique flat representatives for $\sim$
- transfer property (coherence on flat terms)

Contributions and future work

Contributions

- flattening $\rightsquigarrow$ giving not-necessarily-unique flat representatives for $\sim$
- transfer property (coherence on flat terms)
- proof of transfer by saturating β! into B! (context-freely insert @–λ)

Contributions and future work

Contributions

- flattening $\rightsquigarrow$ giving not-necessarily-unique flat representatives for $\sim$
- transfer property (coherence on flat terms)
- proof of transfer by saturating β! into B! (context-freely insert @–λ)
- refactoring bang-calculus theory through PRS theory (*e.g.*, orthogonality)

Contributions and future work

Contributions

- flattening $\rightsquigarrow$ giving not-necessarily-unique flat representatives for $\sim$
- transfer property (coherence on flat terms)
- proof of transfer by saturating $\beta!$ into $B!$ (context-freely insert @–λ)
- refactoring bang-calculus theory through PRS theory (*e.g.*, orthogonality)

Future work

- relate to The Bang Calculus Revisited (Bucciarellia & al. 22)
  (explicit substitution is partial reification of $\beta!$-redex-pattern (λ ✓, ! ✗)
- $\beta!$ modulo σ (by analogy with AC, α, . . . ; all chemical)

Obtaining $B!$ an instance of coherification?

Methodology

problem: coherence $\sim \cdot \to \subseteq \to \cdot \sim$ does *a priori* not hold
solution presented: extend $\beta!$ into $B!$ such that $\sim \cdot \to_{B!} \subseteq \to_{B!} \cdot \sim$

without changing the equational theory

Obtaining $B!$ an instance of coherification?

Methodology

problem: coherence $\sim \cdot \to \subseteq \to \cdot \sim$ does *a priori* not hold
solution presented: extend $\beta!$ into $B!$ such that $\sim \cdot \to_{B!} \subseteq \to_{B!} \cdot \sim$

similar to completion but

Obtaining $B!$ an instance of coherification?

Methodology

problem: coherence $\sim \cdot \to \subseteq \to \cdot \sim$ does *a priori* not hold
solution presented: extend $\beta!$ into $B!$ such that $\sim \cdot \to_{B!} \subseteq \to_{B!} \cdot \sim$

similar to completion but

1. two rewrite systems $\to$ and $\sim$ (instead of one)

Obtaining $B!$ an instance of coherification?

Methodology

problem: coherence $\sim \cdot \to \subseteq \to \cdot \sim$ does *a priori* not hold
solution presented: extend $\beta!$ into $B!$ such that $\sim \cdot \to_{B!} \subseteq \to_{B!} \cdot \sim$

similar to completion but

1. two rewrite systems $\to$ and $\sim$
2. termination is not assumed (nor a goal / invariant)

Obtaining $B!$ an instance of coherification?

Methodology

problem: coherence $\sim \cdot \to \subseteq \to \cdot \sim$ does *a priori* not hold
solution presented: extend $\beta!$ into $B!$ such that $\sim \cdot \to_{B!} \subseteq \to_{B!} \cdot \sim$

similar to completion but

1. two rewrite systems $\to$ and $\sim$
2. termination is not assumed

Question: what is a good name for this process

completion?

Obtaining $B!$ an instance of coherification?

Methodology

problem: coherence $\sim \cdot \to \subseteq \to \cdot \sim$ does *a priori* not hold
solution presented: extend $\beta!$ into $B!$ such that $\sim \cdot \to_{B!} \subseteq \to_{B!} \cdot \sim$

similar to completion but

1. two rewrite systems $\to$ and $\sim$
2. termination is not assumed

Question: what is a good name for this process

completion?
bad since a complete (commuting and terminating) need not hold

Obtaining $B!$ an instance of coherification?

Methodology

problem: coherence $\sim \cdot \to \subseteq \to \cdot \sim$ does *a priori* not hold
solution presented: extend $\beta!$ into $B!$ such that $\sim \cdot \to_{B!} \subseteq \to_{B!} \cdot \sim$

similar to completion but

1. two rewrite systems $\to$ and $\sim$
2. termination is not assumed

Question: what is a good name for this process

saturation?

Obtaining $B!$ an instance of coherification?

Methodology

problem: coherence $\sim \cdot \to \subseteq \to \cdot \sim$ does *a priori* not hold
solution presented: extend $\beta!$ into $B!$ such that $\sim \cdot \to_{B!} \subseteq \to_{B!} \cdot \sim$

similar to completion but

1. two rewrite systems $\to$ and $\sim$
2. termination is not assumed

Question: what is a good name for this process

saturation?
bad since method more general than just extending signature

Obtaining $B!$ an instance of coherification?

Methodology

problem: coherence $\sim \cdot \to \subseteq \to \cdot \sim$ does *a priori* not hold
solution presented: extend $\beta!$ into $B!$ such that $\sim \cdot \to_{B!} \subseteq \to_{B!} \cdot \sim$

similar to completion but

1. two rewrite systems $\to$ and $\sim$
2. termination is not assumed

Question: what is a good name for this process

coherification?

Obtaining $B!$ an instance of coherification?

Methodology

problem: coherence $\sim \cdot \to \subseteq \to \cdot \sim$ does *a priori* not hold
solution presented: extend $\beta!$ into $B!$ such that $\sim \cdot \to_{B!} \subseteq \to_{B!} \cdot \sim$

similar to completion but

1. two rewrite systems $\to$ and $\sim$
2. termination is not assumed

Question: what is a good name for this process

coherification?
bad since method more general than for $\sim$ being symmetric / equivalence

Obtaining $B!$ an instance of coherification?

Methodology

problem: coherence $\sim \cdot \to \subseteq \to \cdot \sim$ does *a priori* not hold
solution presented: extend $\beta!$ into $B!$ such that $\sim \cdot \to_{B!} \subseteq \to_{B!} \cdot \sim$

similar to completion but

1. two rewrite systems $\to$ and $\sim$
2. termination is not assumed

Question: what is a good name for this process

commufication / confluificaton (98), triangulation (special case; & Zantema 12), cutting faces / faceting (20), ...

Obtaining $B!$ an instance of coherification?

Methodology

problem: coherence $\sim \cdot \to \subseteq \to \cdot \sim$ does *a priori* not hold
solution presented: extend $\beta!$ into $B!$ such that $\sim \cdot \to_{B!} \subseteq \to_{B!} \cdot \sim$

similar to completion but

1. two rewrite systems $\to$ and $\sim$
2. termination is not assumed

Question: what is a good name for this process

commufication / confluificaton (98), triangulation (special case; & Zantema 12), cutting faces / faceting (20), . . .