universität innsbruck

Z

Syntax-Free Developments

Vincent van Oostrom
http://cl-informatik.uibk.ac.at

dedicated to Patrick Dehornoy

1. **Z**
2. **Confluence, hyper-cofinality, . . .**
3. **Examples (not) having Z**
4. **Z vs. $\langle$**
5. **Syntax-free developments**

Z

Definition

rewrite system $\to$ comprises:

- a set of objects
- a set of (rewrite) steps
- functions $\mathsf{src}, \mathsf{tgt}$ mapping a step to its source, target object

Z

$$
\begin{tikzcd}
a \arrow[r] & b \arrow[dl, dashed, two heads] \\
a^{\bullet} \arrow[r, dashed, two heads] & b^{\bullet}
\end{tikzcd}
$$

Definition (Z)

$\to$ has the Z-property if there is a (bullet) map $\bullet$ from objects to objects such that for any step $a \to b$ from a to b there exist many-step reductions $b \twoheadrightarrow a^{\bullet}$ from b to $a^{\bullet}$ (upper bound) and $a^{\bullet} \twoheadrightarrow b^{\bullet}$ from $a^{\bullet}$ to $b^{\bullet}$ (monotonic)

$$\begin{tikzcd} a \arrow[r] & b \arrow[dl, dashed, two heads] \\ a^\bullet \arrow[r, dashed, two heads] & b^\bullet \end{tikzcd}$$

Definition (Z)

$\exists \bullet : A \to A, \forall a, b \in A : a \to b \implies b \twoheadrightarrow a^\bullet, a^\bullet \twoheadrightarrow b^\bullet$

Z $\Longrightarrow$ confluence

Theorem

If $\to$ has the Z-property, then $\to$ is confluent

Proof.

$$\begin{tikzcd}
& & a_0 \arrow[r] \arrow[dl] & a_1 \arrow[r] & a_2 \arrow[r] & a_3 \arrow[r, two heads] & a_{n+1} \\
& b_0 \arrow[dl] \\
{}
\end{tikzcd}$$

Z $\Longrightarrow$ confluence

Theorem

If $\to$ has the Z-property, then $\to$ is confluent

Proof.

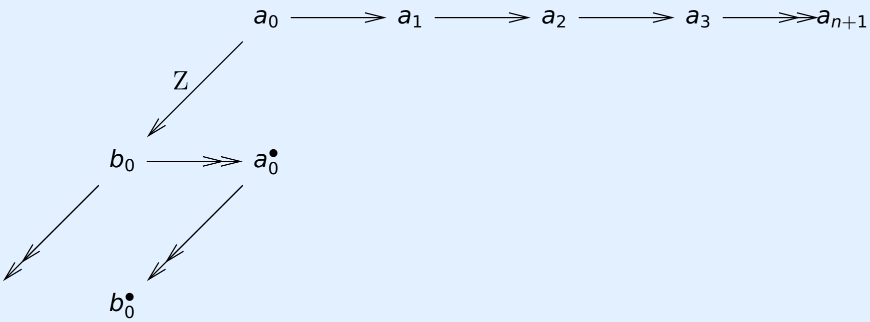
$$
\begin{tikzcd}
& & a_0 \arrow[r] \arrow[dl] & a_1 \arrow[r] & a_2 \arrow[r] & a_3 \arrow[r, ->>] & a_{n+1} \\
& b_0 \arrow[dl, ->>] & & & & & \\
{} & & & & & &
\end{tikzcd}
$$

Z $\Longrightarrow$ confluence

Theorem

If $\to$ has the Z-property, then $\to$ is confluent

Proof.



Z $\Longrightarrow$ confluence

Theorem

If $\to$ has the Z-property, then $\to$ is confluent

Proof.

$$\begin{tikzcd}
 & & a_0 \arrow[r] \arrow[dl, "\mathrm{Z}"'] & a_1 \arrow[r] & a_2 \arrow[r] & a_3 \arrow[r, two heads] & a_{n+1} \\
 & b_0 \arrow[r, two heads] \arrow[dl, two heads] & a_0^\bullet & & & & \\
 {} & & & & & &
\end{tikzcd}$$

$Z \implies$ confluence

Theorem

If $\to$ has the Z-property, then $\to$ is confluent

Proof.

$$
\begin{tikzcd}
 & & a_0 \arrow[r, "Z"'] \arrow[dl] & a_1 \arrow[r] \arrow[dl, twoheadrightarrow] & a_2 \arrow[r] & a_3 \arrow[r, twoheadrightarrow] & a_{n+1} \\
 & b_0 \arrow[r, twoheadrightarrow] \arrow[dl, twoheadrightarrow] & a_0^\bullet \arrow[r, twoheadrightarrow] & a_1^\bullet & & & \\
 {} & & & & & &
\end{tikzcd}
$$

Z $\implies$ confluence

Theorem

If $\to$ has the Z-property, then $\to$ is confluent

Proof.

$$\begin{tikzcd}
 & & a_0 \arrow[r] \arrow[dl] & a_1 \arrow[r, "Z"'] \arrow[dl, two heads] & a_2 \arrow[r] \arrow[dl, two heads] & a_3 \arrow[r, two heads] & a_{n+1} \\
 & b_0 \arrow[r, two heads] \arrow[dl, two heads] & a_0^\bullet \arrow[r, two heads] & a_1^\bullet \arrow[r, two heads] & a_2^\bullet & & \\
 {} & & & & & &
\end{tikzcd}$$

$Z \implies$ confluence

Theorem

If $\to$ has the Z-property, then $\to$ is confluent

Proof.

$$
\begin{tikzcd}
 & & a_0 \arrow[r] \arrow[dl] & a_1 \arrow[r] \arrow[dl] & a_2 \arrow[r, "\mathrm{Z}"'] \arrow[dl] & a_3 \arrow[r, two heads, "\mathrm{Z}"'] \arrow[dl] \arrow[d, phantom, "\text{induction}"] & a_{n+1} \arrow[dl] \\
 & b_0 \arrow[r, two heads] \arrow[dl] & a_0^\bullet \arrow[r, two heads] & a_1^\bullet \arrow[r, two heads] & a_2^\bullet \arrow[r, two heads] & a_n^\bullet & \\
 {} & & & & & &
\end{tikzcd}
$$

Z $\Longrightarrow$ confluence

Theorem

If $\to$ has the Z-property, then $\to$ is confluent

Proof.

$$
\begin{tikzcd}
 & & a_0 \arrow[r] \arrow[dl] & a_1 \arrow[r] \arrow[dl, two heads] & a_2 \arrow[r] \arrow[dl, two heads] & a_3 \arrow[r, two heads] \arrow[dl, two heads] & a_{n+1} \arrow[dl, two heads] \\
 & b_0 \arrow[r, two heads] \arrow[dl, two heads] & a_0^\bullet \arrow[r, two heads] \arrow[dr, phantom, "\text{induction}"] & a_1^\bullet \arrow[r, two heads] & a_2^\bullet \arrow[r, two heads] & a_n^\bullet \arrow[dl, two heads] & \\
 {} \arrow[rrrr, two heads] & & & & {} & &
\end{tikzcd}
$$

Z $\Longrightarrow$ $\multimap\!\!\rightarrow$ strategy is hyper-cofinal

Definition (•-strategy)

- strategy is sub-system of $\rightarrow$ having same normal forms
  (CBV is strat for λ_V, not for λ; strats allowed to be non-deterministic)

Z $\Longrightarrow$ $\multimap\!\!\bullet\!\!\to$ strategy is hyper-cofinal

Definition ($\bullet$-strategy)

- strategy is sub-system of $\to$ having same normal forms
- many-step strategy is $\to^{+}$-strategy
  (strat for many step system)

Z $\Longrightarrow$ $\multimap\!\!\!\bullet\!\!\!\rightarrow$ strategy is hyper-cofinal

Definition (•-strategy)

- strategy is sub-system of $\to$ having same normal forms
- many-step strategy is $\to^+$-strategy
- $a \mathrel{-\!\bullet\!\!\to} a^\bullet$ if a is not a normal form

Z $\Longrightarrow$ $\multimap\!\!\!\!\bullet\!\!\to$ strategy is hyper-cofinal

Definition ($\bullet$-strategy)

- strategy is sub-system of $\to$ having same normal forms
- many-step strategy is $\to^+$-strategy
- $a \mathrel{-\!\!\bullet\!\!\to} a^\bullet$ if a is not a normal form

is many-step strat: if a not normal, $a \to b$ for some b, so $b \twoheadrightarrow a^\bullet$ by Z, so $a \to^+ a^\bullet$

Z $\Longrightarrow$ $\multimap\!\!\bullet\!\!\to$ strategy is hyper-cofinal

Definition ($\bullet$-strategy)

- strategy is sub-system of $\to$ having same normal forms
- many-step strategy is $\to^{+}$-strategy
- $a \multimap\!\!\bullet\!\!\to a^{\bullet}$ if a is not a normal form

is many-step strat: if a not normal, $a \to b$ for some b, so $b \twoheadrightarrow a^{\bullet}$ by Z, so $a \to^{+} a^{\bullet}$

Definition (hyper-cofinality)

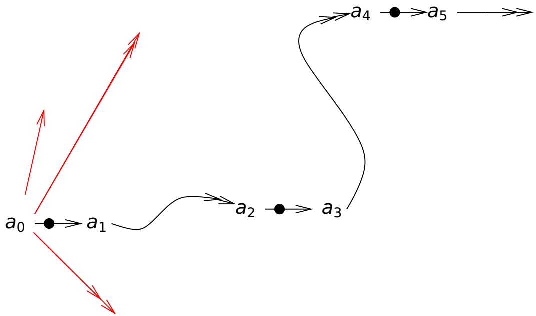
- hyper-strategy: always eventually do a strategy step
- for property P, strategy is hyper-P if hyper-strategy is P
- cofinal: for each strategy reduction any co-initial reduction extendible to it

Z $\Longrightarrow$ $-\!\!\bullet\!\!\to$ strategy is hyper-cofinal

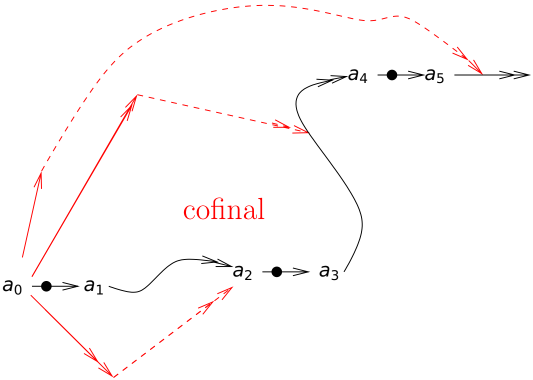
hyper: always eventually a $-\!\!\bullet\!\!\to$-step

$$a_0 \mathrel{-\!\!\bullet\!\!\to} a_1 \twoheadrightarrow a_2 \mathrel{-\!\!\bullet\!\!\to} a_3 \twoheadrightarrow a_4 \mathrel{-\!\!\bullet\!\!\to} a_5 \twoheadrightarrow$$

Z $\Longrightarrow$ $\multimap\!\!\!\!\bullet\!\!\rightarrow$ strategy is hyper-cofinal



Z $\Longrightarrow$ $\multimap\!\!\bullet\!\!\rightarrow$ strategy is hyper-cofinal



Z $\Longrightarrow$ $\rightarrow\!\!\bullet\!\!\rightarrow$ strategy is hyper-cofinal

Theorem

$\rightarrow\!\!\bullet\!\!\rightarrow$ *is hyper-cofinal*

Proof.

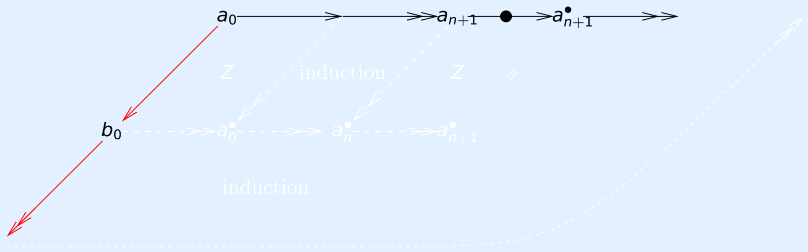
□

Z $\Longrightarrow$ $\multimap\!\!\!\bullet\!\!\rightarrow$ strategy is hyper-cofinal

Theorem

$\rightarrow\!\!\bullet\!\!\rightarrow$ *is hyper-cofinal*

Proof.



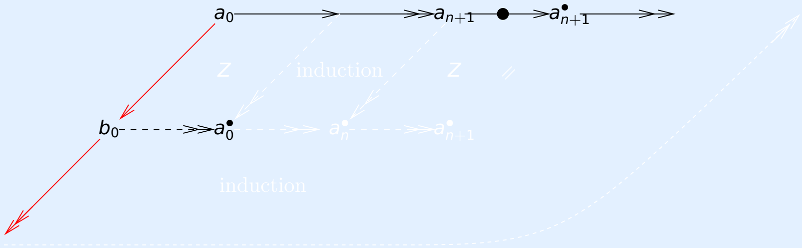
$\square$

$Z \Longrightarrow \rightarrow\!\!\bullet\!\!\rightarrow$ strategy is hyper-cofinal

Theorem

$\rightarrow\!\!\bullet\!\!\rightarrow$ *is hyper-cofinal*

Proof.



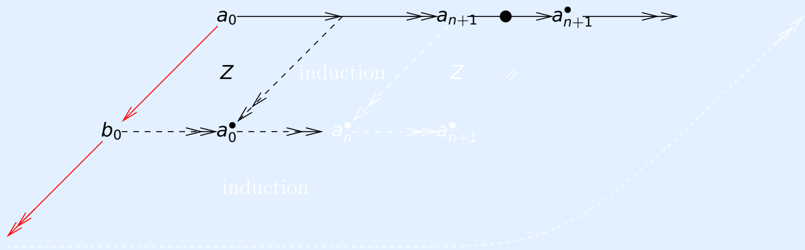
□

$Z \implies \to\!\bullet\!\!\to$ strategy is hyper-cofinal

Theorem

$\to\!\bullet\!\!\to$ *is hyper-cofinal*

Proof.



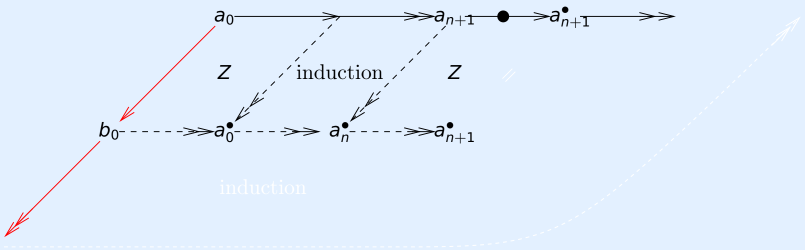
□

$Z \Longrightarrow \;\to\!\bullet\!\to$ strategy is hyper-cofinal

Theorem

$\to\!\bullet\!\to$ *is hyper-cofinal*

Proof.



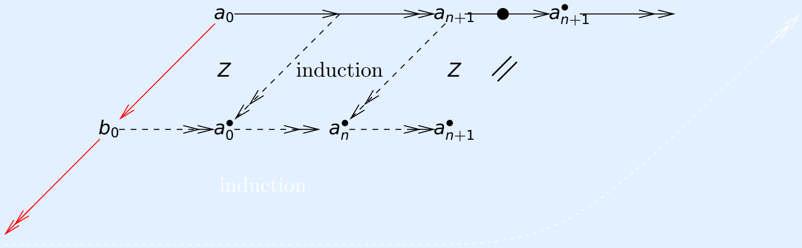
$\square$

Z $\Longrightarrow$ $\multimap\!\!\!\!\bullet\!\!\to$ strategy is hyper-cofinal

Theorem

$\multimap\!\!\!\!\bullet\!\!\to$ *is hyper-cofinal*

Proof.



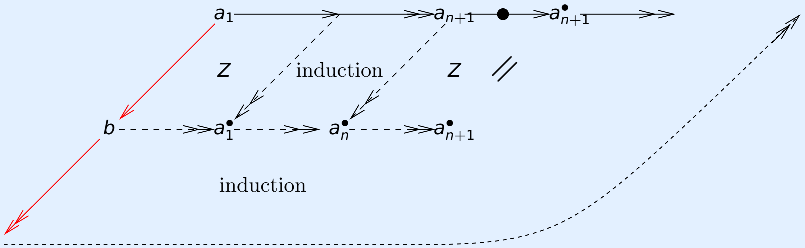
$\square$

$Z \implies \;\to\!\!\!\!\bullet\!\!\to$ strategy is hyper-cofinal

Theorem

$\to\!\!\!\!\bullet\!\!\to$ *is hyper-cofinal*

Proof.



□

Some structured rewrite systems having Z-property

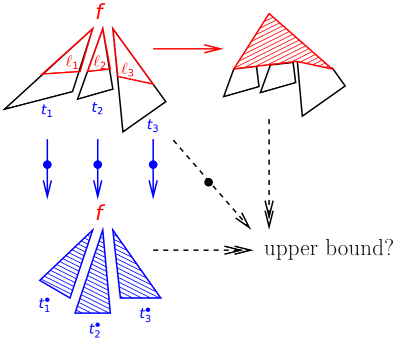
idea for constructing •-function for inductive structures

- suppose to have upper bounds $\vec{t}^{\bullet}$ of sub-structures $\vec{t}$ of $f(\vec{t})$ by induction

Some structured rewrite systems having Z-property

idea for constructing •-function for inductive structures

- suppose to have upper bounds $\vec{t}^{\bullet}$ of sub-structures $\vec{t}$ of $f(\vec{t})$ by induction
- ponder critical peak between those and head step for any rule $f(\vec{\ell}) \to r$



Example of Z: λ-calculus

idea for constructing $\bullet$-function for λ-calculus

- ponder critical peak $(\lambda x.M^\bullet)\,N^\bullet \leftarrow (\lambda x.M)\,N \rightarrow M[x{:=}N]$

$$\begin{tikzcd}
(\lambda x.M)\,N \arrow[rr] \arrow[d, shift right, "\bullet"'] \arrow[d, shift left, "\bullet"] \arrow[dr, dashed, "\bullet"] & & M[x{:=}N] \arrow[dl, dashed, twoheadrightarrow] \\
(\lambda x.M^\bullet)\,N^\bullet \arrow[r, dashed, twoheadrightarrow] & \text{ub?} \arrow[r, dashed, twoheadrightarrow] & M[x{:=}N]^\bullet
\end{tikzcd}$$

Example of Z: λ-calculus

idea for constructing $\bullet$-function for λ-calculus

- ponder critical peak $(\lambda x.M^\bullet)\,N^\bullet \leftarrow (\lambda x.M)\,N \rightarrow M[x{:=}N]$
- contracting $(\lambda x.M^\bullet)\,N^\bullet$ reduces to $M^\bullet[x{:=}N^\bullet]$ reduces to $M[x{:=}N]^\bullet$ for $\bullet$ GK

$$
\begin{tikzcd}
(\lambda x.M)\,N \arrow[rr] \arrow[d, "\bullet"] \arrow[dr, dashed, "\bullet"] & & M[x{:=}N] \arrow[dl, dashed, twoheadrightarrow] \\
(\lambda x.M^\bullet)\,N^\bullet \arrow[r, dashed, twoheadrightarrow] & M^\bullet[x{:=}N^\bullet] \arrow[r, dashed, twoheadrightarrow, "\text{Rhs}"'] & M[x{:=}N]^\bullet
\end{tikzcd}
$$

Gross–Knuth

Example of Z: λ-calculus

Theorem (Loader)

$\rightarrow_\beta$ *has the Z-property for* $\bullet$ *full development (Gross–Knuth):*

$$(\lambda x.M)^\bullet = \lambda x.M^\bullet \qquad x^\bullet = x$$

$$(MN)^\bullet = M'[x{:=}N^\bullet] \quad \textit{if } MN \textit{ is a redex and } M^\bullet = \lambda x.M'\textit{, otherwise } M^\bullet N^\bullet$$

Example of Z: λ-calculus

Theorem (Loader)

$\to_\beta$ has the Z-property for $\bullet$ full development (Gross–Knuth):

$$(\lambda x.M)^\bullet = \lambda x.M^\bullet \qquad x^\bullet = x$$

$$(MN)^\bullet = M'[x{:=}N^\bullet] \quad \text{if } MN \text{ is a redex and } M^\bullet = \lambda x.M'\text{, otherwise } M^\bullet N^\bullet$$

Example

- $I^\bullet = I$; $(I = \lambda x.x)$
- $(I(II))^\bullet = I$, $(III)^\bullet = II$;
- $((\lambda xy.x)zw)^\bullet = (\lambda y.z)w$;
- $((\lambda xy.Iyx)zI)^\bullet = (\lambda y.yz)I$;

Example of Z: λ-calculus

Theorem (Loader)

$\to_\beta$ has the Z-property for $\bullet$ full development (Gross–Knuth):

$$(\lambda x.M)^\bullet = \lambda x.M^\bullet \qquad x^\bullet = x$$
$$(MN)^\bullet = M'[x{:=}N^\bullet] \quad \text{if } MN \text{ is a redex and } M^\bullet = \lambda x.M'\text{, otherwise } M^\bullet N^\bullet$$

Proof by induction on term M:

- (Substitution) $\quad M[y{:=}P][x{:=}N] = M[x{:=}N][y{:=}P[x{:=}N]]$

Example of Z: λ-calculus

Theorem (Loader)

$\to_\beta$ has the Z-property for $\bullet$ full development (Gross–Knuth):

$$(\lambda x.M)^\bullet = \lambda x.M^\bullet \qquad x^\bullet = x$$

$$(MN)^\bullet = M'[x{:=}N^\bullet] \quad \text{if } MN \text{ is a redex and } M^\bullet = \lambda x.M', \text{ otherwise } M^\bullet N^\bullet$$

Proof by induction on term M:

- (Substitution) $M[y{:=}P][x{:=}N] = M[x{:=}N][y{:=}P[x{:=}N]]$
- (Extensive) $M \twoheadrightarrow M^\bullet$

Example of Z: λ-calculus

Theorem (Loader)

$\to_\beta$ *has the Z-property for* $\bullet$ *full development (Gross–Knuth):*

$$
\begin{array}{rclcrcl}
(\lambda x.M)^\bullet & = & \lambda x.M^\bullet & & x^\bullet & = & x \\
(MN)^\bullet & = & M'[x{:=}N^\bullet] & \multicolumn{4}{l}{\textit{if } MN \textit{ is a redex and } M^\bullet = \lambda x.M'\textit{, otherwise } M^\bullet N^\bullet}
\end{array}
$$

Proof by induction on term M:

- (Substitution) $M[y{:=}P][x{:=}N] = M[x{:=}N][y{:=}P[x{:=}N]]$
- (Extensive) $M \twoheadrightarrow M^\bullet$
- (Rhs) $M^\bullet[x{:=}N^\bullet] \twoheadrightarrow M[x{:=}N]^\bullet$

Example of Z: λ-calculus

Theorem (Loader)

$\to_\beta$ has the Z-property for $\bullet$ full development (Gross–Knuth):

$$(\lambda x.M)^\bullet = \lambda x.M^\bullet \qquad x^\bullet = x$$

$$(MN)^\bullet = M'[x{:=}N^\bullet] \quad \text{if } MN \text{ is a redex and } M^\bullet = \lambda x.M', \text{ otherwise } M^\bullet N^\bullet$$

Proof by induction on term M:

- (Substitution) $M[y{:=}P][x{:=}N] = M[x{:=}N][y{:=}P[x{:=}N]]$
- (Extensive) $M \twoheadrightarrow M^\bullet$
- (Rhs) $M^\bullet[x{:=}N^\bullet] \twoheadrightarrow M[x{:=}N]^\bullet$
- (Z) $M \to N \implies N \twoheadrightarrow M^\bullet \twoheadrightarrow N^\bullet$ $\square$

Example of Z: λ-calculus

Theorem (Loader)

$\to_\beta$ has the Z-property for $\bullet$ full development (Gross–Knuth):

$$(\lambda x.M)^\bullet = \lambda x.M^\bullet \qquad x^\bullet = x$$

$$(MN)^\bullet = M'[x{:=}N^\bullet] \quad \text{if } MN \text{ is a redex and } M^\bullet = \lambda x.M', \text{ otherwise } M^\bullet N^\bullet$$

Proof by induction on term M:

- (Substitution) $M[y{:=}P][x{:=}N] = M[x{:=}N][y{:=}P[x{:=}N]]$
- (Extensive) $M \twoheadrightarrow M^\bullet$
- (Rhs) $M^\bullet[x{:=}N^\bullet] \twoheadrightarrow M[x{:=}N]^\bullet$
- (Z) $M \to N \implies N \twoheadrightarrow M^\bullet \twoheadrightarrow N^\bullet$ $\square$

works for all orthogonal structured rewrite systems

Example of Z: λ-calculus

Theorem (cf. Aczel)

$\to_\beta$ *has the Z-property for* $\bullet$ *full superdevelopment:*

$$(\lambda x.M)^\bullet = \lambda x.M^\bullet \qquad x^\bullet = x$$
$$(MN)^\bullet = M'[x{:=}N^\bullet] \quad \text{if } MN \text{ is a term and } M^\bullet = \lambda x.M', \text{ otherwise } M^\bullet N^\bullet$$

Proof by induction on term M:

- (Substitution) $M[y{:=}P][x{:=}N] = M[x{:=}N][y{:=}P[x{:=}N]]$
- (Extensive) $M \twoheadrightarrow M^\bullet$
- (Rhs) $M^\bullet[x{:=}N^\bullet] \twoheadrightarrow M[x{:=}N]^\bullet$
- (Z) $M \to N \implies N \twoheadrightarrow M^\bullet \twoheadrightarrow N^\bullet$ $\square$

full superdevelopment; shortest mechanized proof

Example of Z: self-distributivity

Definition

self-distributivity generated by rule $xyz \to xz(yz)$

Example of Z: self-distributivity

Definition

self-distributivity generated by rule $xyz \to xz(yz)$

idea: distribute of 2nd argument to leaves of 1st argument

Example of Z: self-distributivity

Theorem (Dehornoy)

self-distributivity has Z-property for $\bullet$ full distribution, $t[s]$ uniform distribution:

$$x^\bullet = x \qquad (ts)^\bullet = t^\bullet[s^\bullet] \qquad t[s] = t[x_1{:=}x_1 s, x_2{:=}x_2 s, \ldots]$$

Example of Z: self-distributivity

Theorem (Dehornoy)

self-distributivity has Z-property for $\bullet$ *full distribution,* $t[s]$ *uniform distribution:*

$$x^\bullet = x \qquad (ts)^\bullet = t^\bullet[s^\bullet] \qquad t[s] = t[x_1{:=}x_1 s, x_2{:=}x_2 s, \ldots]$$

Example

- $(xy)^\bullet = x[y] = x[x{:=}xy] = xy$
- $(xyz)^\bullet = (xy)[x{:=}xz, y{:=}yz] = xz(yz)$

Example of Z: self-distributivity

Theorem (Dehornoy)

self-distributivity has Z-property for $\bullet$ *full distribution,* $t[s]$ *uniform distribution:*

$$x^\bullet = x \qquad (ts)^\bullet = t^\bullet[s^\bullet] \qquad t[s] = t[x_1{:=}x_1 s, x_2{:=}x_2 s, \ldots]$$

Proof by induction on term t:

- (Sequentialisation) $\quad ts \twoheadrightarrow t[s]$

Example of Z: self-distributivity

Theorem (Dehornoy)

self-distributivity has Z-property for $\bullet$ *full distribution,* $t[s]$ *uniform distribution:*

$$x^\bullet = x \qquad (ts)^\bullet = t^\bullet[s^\bullet] \qquad t[s] = t[x_1{:=}x_1 s, x_2{:=}x_2 s, \ldots]$$

Proof by induction on term t:

- (Sequentialisation) $ts \twoheadrightarrow t[s]$
- (Substitution) $t[s][r] \twoheadrightarrow t[r][s[r]]$

Example of Z: self-distributivity

Theorem (Dehornoy)

self-distributivity has Z-property for $\bullet$ *full distribution,* $t[s]$ *uniform distribution:*

$$x^\bullet = x \qquad (ts)^\bullet = t^\bullet[s^\bullet] \qquad t[s] = t[x_1{:=}x_1 s, x_2{:=}x_2 s, \ldots]$$

Proof by induction on term t:

- (Sequentialisation) $ts \twoheadrightarrow t[s]$
- (Substitution) $t[s][r] \twoheadrightarrow t[r][s[r]]$
- (Extensive) $t \twoheadrightarrow t^\bullet$

Example of Z: self-distributivity

Theorem (Dehornoy)

self-distributivity has Z-property for $\bullet$ full distribution, $t[s]$ uniform distribution:

$$x^\bullet = x \qquad (ts)^\bullet = t^\bullet[s^\bullet] \qquad t[s] = t[x_1{:=}x_1 s, x_2{:=}x_2 s, \ldots]$$

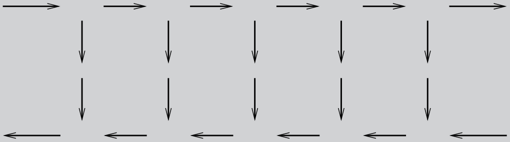
Proof by induction on term t:

- (Sequentialisation) $ts \twoheadrightarrow t[s]$
- (Substitution) $t[s][r] \twoheadrightarrow t[r][s[r]]$
- (Extensive) $t \twoheadrightarrow t^\bullet$
- (Z) $s \twoheadrightarrow t^\bullet \twoheadrightarrow s^\bullet$, if $t \to s$ $\square$

Confluent rewrite systems not having the Z-property

Transitivity considered harmful

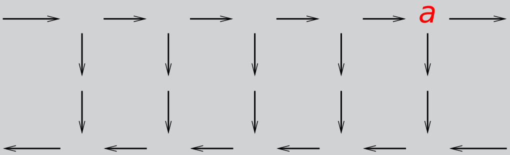
Example (Confluent but not admitting Z)



confluent (by decreasing diagrams); no transitive steps (own transitive reduct)

Transitivity considered harmful

Example (Confluent but not admitting Z)

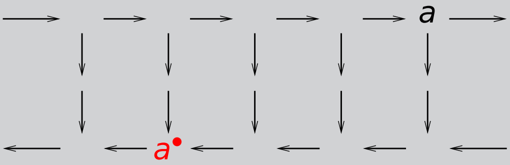


suppose $\bullet$ were to witness Z:

- consider arbitrary a at the top

Transitivity considered harmful

Example (Confluent but not admitting Z)

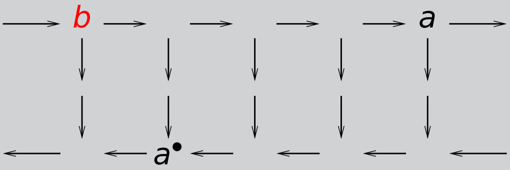


suppose $\bullet$ were to witness Z:

- consider arbitrary a at the top
- $a^\bullet$ must be at bottom, left of a as upper bound of steps from a

Transitivity considered harmful

Example (Confluent but not admitting Z)



suppose $\bullet$ were to witness Z:

- consider arbitrary a at the top
- $a^\bullet$ must be at bottom, left of a as upper bound of steps from a
- consider arbitrary b at top, strictly left of $a^\bullet$

Transitivity considered harmful

Example (Confluent but not admitting Z)

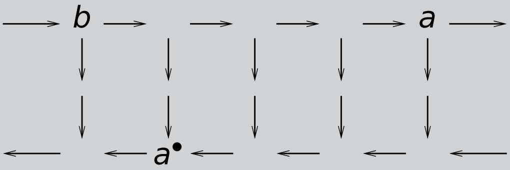
$$\begin{tikzcd}
{} \arrow[r] & b \arrow[r] \arrow[d] & {} \arrow[r] \arrow[d] & {} \arrow[r] \arrow[d] & {} \arrow[r] \arrow[d] & a \arrow[r] \arrow[d] & {} \\
& {} \arrow[d] & {} \arrow[d] & {} \arrow[d] & {} \arrow[d] & {} \arrow[d] & \\
{} & {} \arrow[l] & a^\bullet \arrow[l] & {} \arrow[l] & {} \arrow[l] & {} \arrow[l] & {} \arrow[l]
\end{tikzcd}$$

suppose $\bullet$ were to witness Z:

- consider arbitrary a at the top
- $a^\bullet$ must be at bottom, left of a as upper bound of steps from a
- consider arbitrary b at top, strictly left of $a^\bullet$
- $a^\bullet \to^+ b^\bullet$ by $b^\bullet$ being an upper bound of steps from b

Transitivity considered harmful

Example (Confluent but not admitting Z)



suppose $\bullet$ were to witness Z:

- consider arbitrary a at the top
- $a^\bullet$ must be at bottom, left of a as upper bound of steps from a
- consider arbitrary b at top, strictly left of $a^\bullet$
- $a^\bullet \to^+ b^\bullet$ by $b^\bullet$ being an upper bound of steps from b
- $b^\bullet \twoheadrightarrow a^\bullet$ by $b \twoheadrightarrow a$ and monotonicity; contradiction

Transitivity considered harmful

Example (Confluent but not admitting Z)

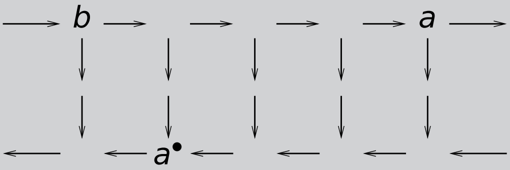
$$
\begin{tikzcd}
{} \arrow[r] & b \arrow[r] \arrow[d] & {} \arrow[r] \arrow[d] & {} \arrow[r] \arrow[d] & {} \arrow[r] \arrow[d] & a \arrow[r] \arrow[d] & {} \\
& {} \arrow[d] & {} \arrow[d] & {} \arrow[d] & {} \arrow[d] & {} \arrow[d] & \\
{} & {} \arrow[l] & a^\bullet \arrow[l] & {} \arrow[l] & {} \arrow[l] & {} \arrow[l] & {} \arrow[l]
\end{tikzcd}
$$

suppose $\bullet$ were to witness Z:

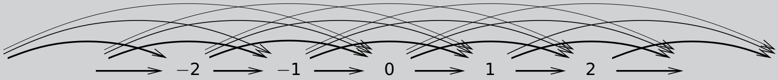
- consider arbitrary a at the top
- $a^\bullet$ must be at bottom, left of a as upper bound of steps from a
- consider arbitrary b at top, strictly left of $a^\bullet$
- $a^\bullet \to^+ b^\bullet$ by $b^\bullet$ being an upper bound of steps from b
- $b^\bullet \twoheadrightarrow a^\bullet$ by $b \twoheadrightarrow a$ and monotonicity; contradiction

Transitivity considered harmful

Example (Confluent but not admitting Z)

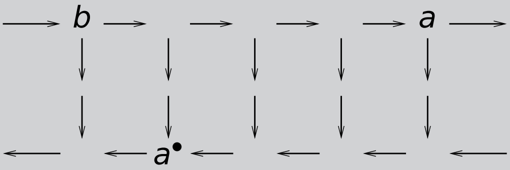


Example (less-than on $\mathbb{Z}$ does not have Z, but transitive reduct does)

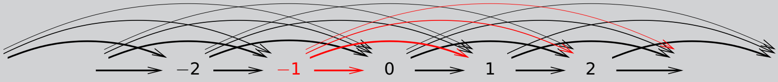


Transitivity considered harmful

Example (Confluent but not admitting Z)



Example (less-than on $\mathbb{Z}$ does not have Z, but transitive reduct does)



for given integer, no upper bound on steps from it

Further examples of rewrite systems having Z

Lemma (Some sufficient conditions for Z)

Z holds if

- *confluent and (weakly) normalising: map to the normal form*

Further examples of rewrite systems having Z

Lemma (Some sufficient conditions for Z)

Z holds if

- *confluent and (weakly) normalising: map to the normal form;*
- *locally confluent and terminating: $\bullet$ maps a to arbitrary $a^\bullet$ in nf s.t. $a \twoheadrightarrow a^\bullet$*
  Z: $a \to b \implies a \twoheadrightarrow a^\bullet = b^\bullet$ by wf-induction on a; Newman/Winkler/Hirokawa

Further examples of rewrite systems having Z

Lemma (Some sufficient conditions for Z)

Z holds if

- *confluent and (weakly) normalising: map to the normal form;*
- *locally confluent and terminating: $\bullet$ maps a to arbitrary $a^\bullet$ in nf s.t. $a \twoheadrightarrow a^\bullet$;*
- *orthogonal: contract all redexes in structure*

Further examples of rewrite systems having Z

Lemma (Some sufficient conditions for Z)

Z holds if

- *confluent and (weakly) normalising: map to the normal form;*
- *locally confluent and terminating: $\bullet$ maps a to arbitrary $a^\bullet$ in nf s.t. $a \twoheadrightarrow a^\bullet$;*
- *orthogonal: contract all redexes in structure;*
- *confluent and finite: map to any object in normal form quotienting out SCCs*

Further examples of rewrite systems having Z

Lemma (Some sufficient conditions for Z)

Z holds if

- *confluent and (weakly) normalising: map to the normal form;*
- *locally confluent and terminating: $\bullet$ maps a to arbitrary $a^\bullet$ in nf s.t. $a \twoheadrightarrow a^\bullet$;*
- *orthogonal: contract all redexes in structure;*
- *confluent and finite: map to any object in normal form quotienting out SCCs*

Example (Some further concrete systems)

- weakly orthogonal: contract maximal redex-set ($\underline{ps}\overline{ps}$, not $p\underline{sp}s$)

Further examples of rewrite systems having Z

Lemma (Some sufficient conditions for Z)

Z holds if

- *confluent and (weakly) normalising: map to the normal form;*
- *locally confluent and terminating: $\bullet$ maps a to arbitrary $a^\bullet$ in nf s.t. $a \twoheadrightarrow a^\bullet$;*
- *orthogonal: contract all redexes in structure;*
- *confluent and finite: map to any object in normal form quotienting out SCCs*

Example (Some further concrete systems)

- weakly orthogonal: contract maximal redex-set
- explicit substitutions: compose maps for Beta and subs (Nakazawa & Fujita)

Further examples of rewrite systems having Z

Lemma (Some sufficient conditions for Z)

Z holds if

- *confluent and (weakly) normalising: map to the normal form;*
- *locally confluent and terminating: $\bullet$ maps a to arbitrary $a^\bullet$ in nf s.t. $a \twoheadrightarrow a^\bullet$;*
- *orthogonal: contract all redexes in structure;*
- *confluent and finite: map to any object in normal form quotienting out SCCs*

Example (Some further concrete systems)

- weakly orthogonal: contract maximal redex-set
- explicit substitutions: compose maps for Beta and subs (Nakazawa & Fujita)
- generalized braids: Garside element tiling genererators (w/ Hans Zantema)

Further examples of rewrite systems having Z

Lemma (Some sufficient conditions for Z)

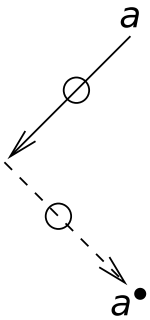
Z holds if

- *confluent and (weakly) normalising: map to the normal form;*
- *locally confluent and terminating: $\bullet$ maps a to arbitrary $a^{\bullet}$ in nf s.t. $a \twoheadrightarrow a^{\bullet}$;*
- *orthogonal: contract all redexes in structure;*
- *confluent and finite: map to any object in normal form quotienting out SCCs*

Example (Some further concrete systems . . .)

- weakly orthogonal: contract maximal redex-set
- explicit substitutions: compose maps for Beta and subs (Nakazawa & Fujita)
- generalized braids: Garside element tiling genererators (w/ Hans Zantema)

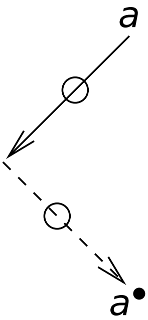
Angle



Definition (Terese 2003)

$\multimap\!\!\!\rightarrow$ has triangle property if there is a (bullet) map $\bullet$ from objects to objects such that for any step $a \multimap\!\!\!\rightarrow b$ from a to b there exists a step $b \multimap\!\!\!\rightarrow a^\bullet$ from b to $a^\bullet$

$\langle$



Definition ($\langle$)

$\exists \bullet : A \to A, \forall a, b \in A : a \mathrel{-\!\circ\!\!\rightarrow} b \implies b \mathrel{-\!\circ\!\!\rightarrow} a^\bullet$

Z vs. $\langle$

Z vs. $\langle$

Theorem

for any map $\bullet$, $Z \iff$ *exists* $\rightarrow\ \subseteq\ \multimap\!\!\rightarrow\ \subseteq\ \twoheadrightarrow$ *such that* $\langle$

Proof.

$\square$

Z vs. $\langle$

Theorem

for any map $\bullet$, $Z \iff$ *exists* $\rightarrow \subseteq \multimap \subseteq \twoheadrightarrow$ *such that* $\langle$

Proof.

$(\Longleftarrow)$ see paper

$\square$

Z vs. $\langle$

Theorem

for any map $\bullet$, $Z \iff$ *exists* $\to\ \subseteq\ \multimap\!\!\!\to\ \subseteq\ \twoheadrightarrow$ *such that* $\langle$

Proof.

$(\implies)$ define $a \multimap\!\!\!\to b$ if b between a and $a^\bullet$, i.e. $a \twoheadrightarrow b \twoheadrightarrow a^\bullet$: $\square$

Syntax-free developments

Recover results on developments in syntax-free way?

$a \multimap\!\!\!\rightarrow b$ defined as $a \twoheadrightarrow b \twoheadrightarrow a^{\bullet}$ can be seen as a $\bullet$-development, as a syntax-free definition of development (Church & Rosser) relative to $\bullet$. which results on developments can be recovered for $\bullet$-developments, i.e. in a syntax-free way?

Syntax-free developments

Recover results on developments in syntax-free way?

which results on developments can be recovered for $\bullet$-developments?

Example (Developments do not coincide with $\bullet$-developments)

let $\bullet$ be full-development map (contract all redexes in term) for orthogonal TRS

Syntax-free developments

Recover results on developments in syntax-free way?

which results on developments can be recovered for $\bullet$-developments?

Example (Developments do not coincide with $\bullet$-developments)

let $\bullet$ be full-development map (contract all redexes in term) for orthogonal TRS

- rules $a \to b \to c \to a$; non-terminating/cyclic
  $a^\bullet = b$ but a $\bullet$-develops to c

Syntax-free developments

Recover results on developments in syntax-free way?

which results on developments can be recovered for $\bullet$-developments?

Example (Developments do not coincide with $\bullet$-developments)

let $\bullet$ be full-development map (contract all redexes in term) for orthogonal TRS

- rules $a \to b \to c \to a$; non-terminating/cyclic
  $a^\bullet = b$ but a $\bullet$-develops to c
- rules $a \to b \to c$, $f(x) \to d$; erasing
  $f(a)^\bullet = d$ but $f(a)$ $\bullet$-develops to $f(c)$

Syntax-free developments

Recover results on developments in syntax-free way?

which results on developments can be recovered for $\bullet$-developments?

Example (Developments do not coincide with $\bullet$-developments)

let $\bullet$ be full-development map (contract all redexes in term) for orthogonal TRS

- rules $a \to b \to c \to a$; non-terminating/cyclic
  $a^\bullet = b$ but a $\bullet$-develops to c
- rules $a \to b \to c$, $f(x) \to d$; erasing
  $f(a)^\bullet = d$ but $f(a)$ $\bullet$-develops to $f(c)$
- rules $g(x) \to h(x) \to i(x) \to x$; collapsing
  $i(h(g(a)))^\bullet = i(h(a))$ but $i(h(g(a)))$ $\bullet$-develops to $i(h(i(a)))$

Syntax-free developments

Recover results on developments in syntax-free way?

which results on developments can be recovered for $\bullet$-developments?

Theorem

for terminating, non-collapsing, and non-erasing orthogonal TRSs, developments and $\bullet$-developments coincide.

Syntax-free developments

Recover results on developments in syntax-free way?

which results on developments can be recovered for $\bullet$-developments?

Theorem

for terminating, non-collapsing, and non-erasing orthogonal TRSs, developments and $\bullet$-developments coincide.

Proof.

conditions guarantee absence of syntactic accidents (Lévy): $t \twoheadrightarrow s \twoheadrightarrow t^\bullet$,
at most one reduction up to permutation equivalence between two terms $\Longrightarrow$
development $t \twoheadrightarrow t^\bullet$, so each step in $t \twoheadrightarrow s$ contracts residual of redex in t $\Longrightarrow$
$t \twoheadrightarrow s$ is a development. $\square$

Syntax-free developments

Recover results on developments in syntax-free way?

which results on developments can be recovered for $\bullet$-developments?

Theorem

for terminating, non-collapsing, and non-erasing orthogonal TRSs, developments and $\bullet$-developments coincide.

Remark

can be regained for arbitrary orthogonal TRSs by lifting: add creation depths (to overcome collapsingness and non-termination; Hyland–Wadsworth/Lévy labels) to yield reconstructibility, and memory (to overcome erasingness; cf. Nederpelt's scars) to yield invertibility. Question: other systems (λ, SD)?

Conclusion

- introduced Z-property

Conclusion

- introduced Z-property
- showed interest of Z: entails confluence, gives hyper-cofinal strategy (computable if $\bullet$ is), allows to characterise recurrence (Statman),...

Conclusion

- introduced Z-property
- showed interest of Z: entails confluence, gives hyper-cofinal strategy (computable if $\bullet$ is), allows to characterise recurrence (Statman),...
- convenient because of choice of monotonic upper bound function $\bullet$ does not always exist though even if confluent

Conclusion

- introduced Z-property
- showed interest of Z: entails confluence, gives hyper-cofinal strategy (computable if $\bullet$ is), allows to characterise recurrence (Statman),...
- convenient because of choice of monotonic upper bound function $\bullet$ does not always exist though even if confluent
- equivalent to triangle property (e.g. Takahashi) but conceptually minimal: no need for separate inductive definition of parallel reduction

Conclusion

- introduced Z-property
- showed interest of Z: entails confluence, gives hyper-cofinal strategy (computable if $\bullet$ is), allows to characterise recurrence (Statman),...
- convenient because of choice of monotonic upper bound function $\bullet$ does not always exist though even if confluent
- equivalent to triangle property (e.g. Takahashi) but conceptually minimal: no need for separate inductive definition of parallel reduction
- spin-off: syntax-free notion of $\bullet$-development; left-divisors (complete developments) of parallel reduction not closed under left-division

Further future work?

- find methods for showing the Z-property does not hold ($\lambda\beta\overline{\eta}$?)

Further future work?

- find methods for showing the Z-property does not hold ($\lambda\beta\overline{\eta}$?)
- try to turn Z into automatable method (for confluence tools)?

Further future work?

- find methods for showing the Z-property does not hold ($\lambda\beta\overline{\eta}$?)
- try to turn Z into automatable method (for confluence tools)?
- what exactly do programming languages want to import?
  PLFA (Wadler, Kokke) uses some version of angle/Z to get confluence

Further future work?

- find methods for showing the Z-property does not hold ($\lambda\beta\overline{\eta}$?)
- try to turn Z into automatable method (for confluence tools)?
- what exactly do programming languages want to import?
  PLFA (Wadler, Kokke) uses some version of angle/Z to get confluence
- what exactly do proof assistants want to import (for partial functions)?
  Agda (Cockx) allows some version of angle/Z to get confluence
  Dedukti allows confluent HRS rules (external check, e.g. via CSI-ho or ACPH)

Further future work?

- find methods for showing the Z-property does not hold ($\lambda\beta\overline{\eta}$?)
- try to turn Z into automatable method (for confluence tools)?
- what exactly do programming languages want to import?
  PLFA (Wadler, Kokke) uses some version of angle/Z to get confluence
- what exactly do proof assistants want to import (for partial functions)?
  Agda (Cockx) allows some version of angle/Z to get confluence
  Dedukti allows confluent HRS rules (external check, e.g. via CSI-ho or ACPH)
- does your favourite rewrite system have the Z-property?

Further future hobby?

- find methods for showing the Z-property does not hold ($\lambda\beta\overline{\eta}$?)
- try to turn Z into automatable method (for confluence tools)?
- what exactly do programming languages want to import?
  PLFA (Wadler, Kokke) uses some version of angle/Z to get confluence
- what exactly do proof assistants want to import (for partial functions)?
  Agda (Cockx) allows some version of angle/Z to get confluence
  Dedukti allows confluent HRS rules (external check, e.g. via CSI-ho or ACPH)
- does your favourite rewrite system have the Z-property?
- ...

Further future hobby?

- find methods for showing the Z-property does not hold ($\lambda\beta\overline{\eta}$?)
- try to turn Z into automatable method (for confluence tools)?
- what exactly do programming languages want to import?
  PLFA (Wadler, Kokke) uses some version of angle/Z to get confluence
- what exactly do proof assistants want to import (for partial functions)?
  Agda (Cockx) allows some version of angle/Z to get confluence
  Dedukti allows confluent HRS rules (external check, e.g. via CSI-ho or ACPH)
- does your favourite rewrite system have the Z-property?
- ...