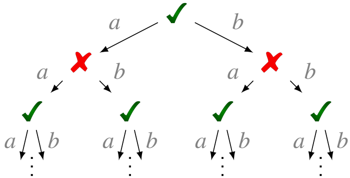
Formal Languages

baab *aa* *bb* *ε* *ba* *ab* …

Formally and Coinductively



Dmitriy Traytel

ETH zürich

Contribution

library of **formal languages** in

Contribution

define regular operations $\emptyset$, ε, Atom, $+$, $\cdot$, *
prove axioms of Kleene Algebra

library of **formal languages** in Isabelle HOL

Contribution

define regular operations $\varnothing$, ε, Atom, $+$, $\cdot$, *
prove axioms of Kleene Algebra

Coinductive library of **formal languages** in

Contribution

define regular operations $\emptyset$, ε, Atom, $+$, $\cdot$, *
prove axioms of Kleene Algebra

Coinductive library of **formal languages** in

Tutorial for **corecursion** and **coinduction**

Contribution

define regular operations $\varnothing$, ε, Atom, $+$, $\cdot$, *
prove axioms of Kleene Algebra

Coinductive library of **formal languages** in

Formal Structure

Tutorial for **corecursion** and **coinduction**

Computation   Deduction

Contribution

define regular operations $\varnothing$, ε, Atom, $+$, $\cdot$, *
prove axioms of Kleene Algebra

Coinductive library of **formal languages** in

Formal Structure

This talk: Tutorial in 20 min

Tutorial for **corecursion** and **coinduction**

Computation Deduction

Related Work: A Selection of CoTutorials

	Codatatype	Corecursion	Coinduction	Proof Assistant
Jacobs, Rutten EATCS'97	stream	✓	✓	✗
Rutten CONCUR'98	language	✓	✓	✗
Giménez, Castéran '98	stream, lazy list	✓	✓	
Rutten *MSCS*'05	stream	✓	✓	✗
Hinze *JFP*'11	stream	✓	✗	✗
Chlipala '13	stream, while	✓	✓	
Rot, Rutten, Bonsangue LATA'13	language	✗	✓	✗
Kozen, Silva *MSCS*'14	stream	✗	✓	✗
Setzer Festschrift Jäger'16	stream	✓	✓	✗
Traytel FSCD'16	language	✓	✓	Isabelle HOL

Codatatype

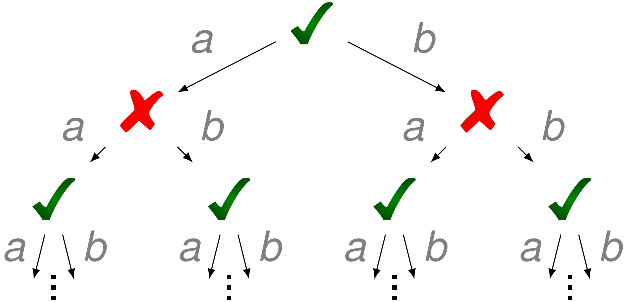
Corecursion

Coinduction

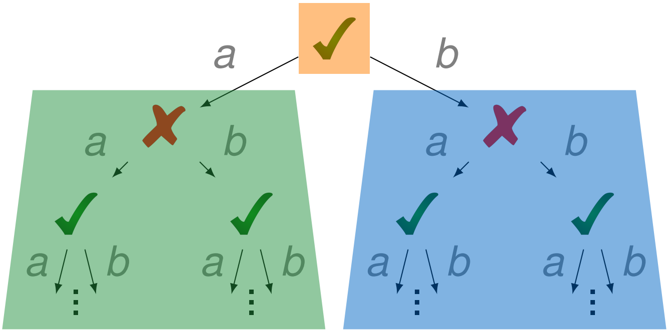
Codatatype

Corecursion

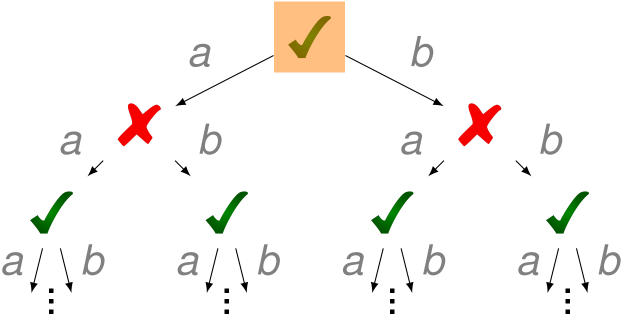
Coinduction


a
b
a
b
a
b
a
b
a
b
a
b
a
b

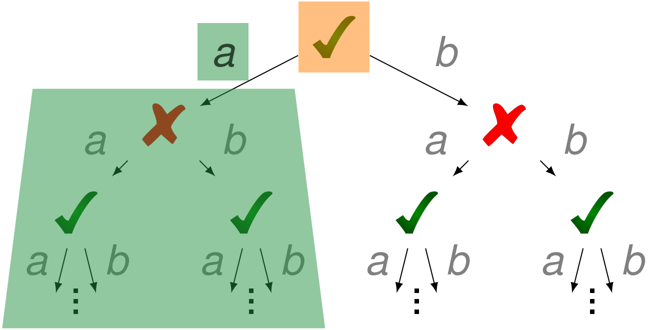
codatatype $\mathit{lang} = \mathscr{L}\ \mathit{bool}\quad \mathit{lang}\quad \mathit{lang}$



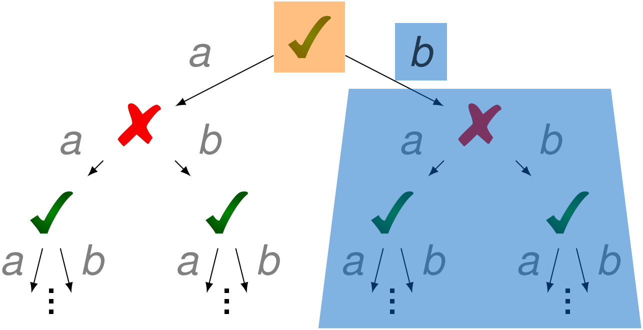
codatatype $\alpha\, lang = \mathscr{L}\ bool$



codatatype α *lang* $= \mathscr{L}$ *bool*



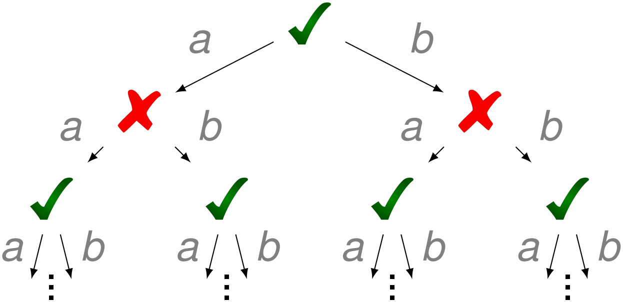
$\texttt{codatatype}\ \alpha\ \mathit{lang} = \mathscr{L}\ \mathit{bool}$



codatatype α *lang* $= \mathscr{L}$ *bool*

$o :: \alpha\ \mathit{lang} \Rightarrow \mathit{bool}$

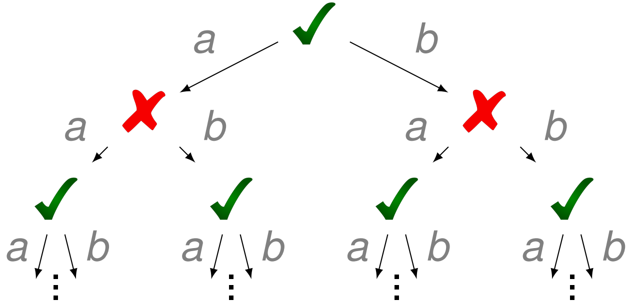
$:: \alpha\ \mathit{lang} \Rightarrow$



$$\texttt{primrec} \in :: \alpha\,\mathit{list} \Rightarrow \alpha\,\mathit{lang} \Rightarrow \mathit{bool}$$

$$[] \in L = o\ L$$

$$aw \in L = w \in \quad L\ a$$

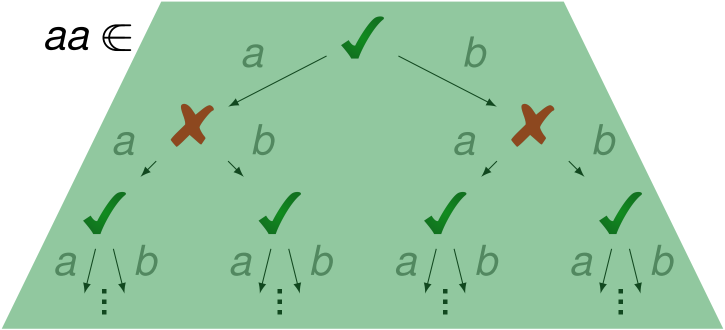


$\texttt{primrec} \in :: \alpha\ \mathit{list} \Rightarrow \alpha\ \mathit{lang} \Rightarrow \mathit{bool}$

$\quad [] \in L = o\ L$

$\quad aw \in L = w \in \quad L\ a$

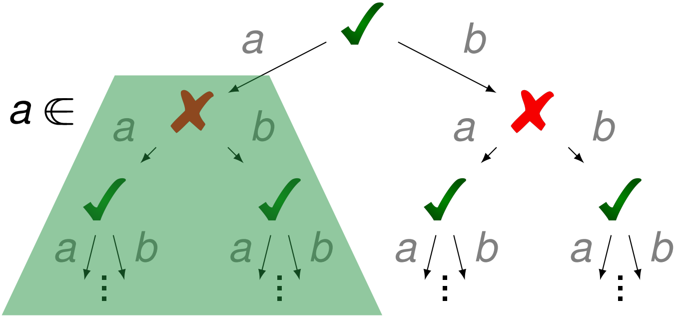
$aa \in$



$$\texttt{primrec} \in :: \alpha\, list \Rightarrow \alpha\, lang \Rightarrow bool$$

$$[] \in L = o\; L$$

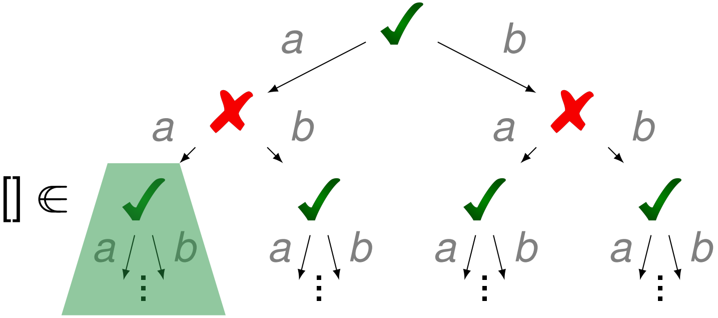
$$aw \in L = w \in \quad L\; a$$



$$\texttt{primrec} \in :: \alpha\,list \Rightarrow \alpha\,lang \Rightarrow bool$$

$$[] \in L = o\ L$$

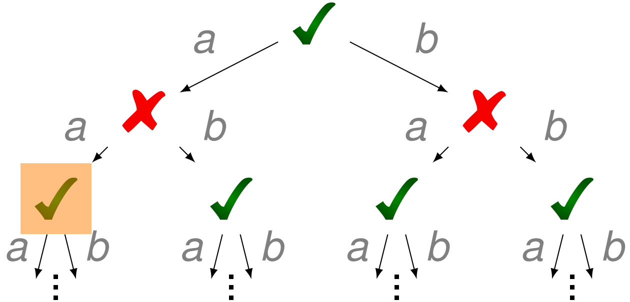
$$aw \in L = w \in \quad L\ a$$



$$\texttt{primrec}\ \in\ ::\ \alpha\ list \Rightarrow \alpha\ lang \Rightarrow bool$$

$$[]\in L = o\ L$$

$$aw \in L = w \in \quad L\ a$$



Codatatype

Corecursion

Coinduction

$\texttt{primcorec}\ \varnothing :: \alpha\ \mathit{lang}$

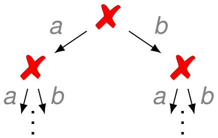
$\quad o\ \varnothing = ✗$

$\quad \varnothing = \lambda\_.\ \varnothing$

`primcorec` $\varnothing :: \alpha$ *lang*

$\quad o\ \varnothing = ✗$

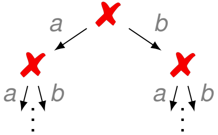
$\quad \varnothing = \lambda\_.\ \varnothing$



$\texttt{primcorec}\ \varnothing :: \alpha\ \mathit{lang}$

$o\ \varnothing = $ ✗

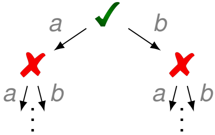
$\varnothing = \lambda\_.\ \varnothing$



$\texttt{primcorec}\ \varepsilon :: \alpha\ \mathit{lang}$

$o\ \varepsilon = $ ✓

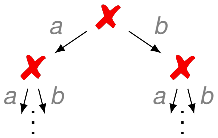
$\varepsilon = \lambda\_.\ \varnothing$



$\texttt{primcorec}\ \varnothing :: \alpha\ \mathit{lang}$

$\quad \mathit{o}\ \varnothing = ✗$

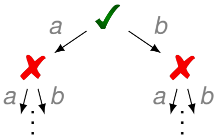
$\quad \varnothing = \lambda\_.\ \varnothing$



$\texttt{primcorec}\ \varepsilon :: \alpha\ \mathit{lang}$

$\quad \mathit{o}\ \varepsilon = ✓$

$\quad \varepsilon = \lambda\_.\ \varnothing$



$\texttt{primcorec}\ \mathsf{Atom} :: \alpha \Rightarrow \alpha\ \mathit{lang}$

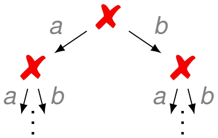
$\quad \mathit{o}\ (\mathsf{Atom}\ a) = ✗$

$\quad (\mathsf{Atom}\ a) = \lambda b.\ \texttt{if}\ a = b\ \texttt{then}\ \varepsilon\ \texttt{else}\ \varnothing$

`primcorec` $\varnothing :: \alpha\ \mathit{lang}$

$\quad o\ \varnothing = ✗$

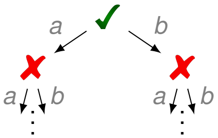
$\quad \varnothing = \lambda\_.\ \varnothing$



`primcorec` $\varepsilon :: \alpha\ \mathit{lang}$

$\quad o\ \varepsilon = ✓$

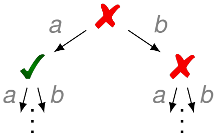
$\quad \varepsilon = \lambda\_.\ \varnothing$

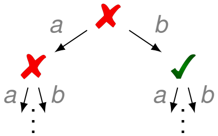


`primcorec` $\mathsf{Atom} :: \alpha \Rightarrow \alpha\ \mathit{lang}$

$\quad o\ (\mathsf{Atom}\ a) = ✗$

$\quad (\mathsf{Atom}\ a) = \lambda b.\ \texttt{if}\ a = b\ \texttt{then}\ \varepsilon\ \texttt{else}\ \varnothing$





	`primrec`	`primcorec`
Syntactic criterion for	termination	productivity
of functions of type	α *list* $\Rightarrow \cdots$	$\cdots \Rightarrow \alpha$ *lang*

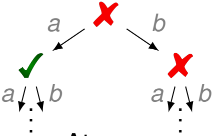
	`primrec`	`primcorec`
Syntactic criterion for of functions of type	termination α *list* $\Rightarrow \cdots$	productivity $\cdots \Rightarrow \alpha$ *lang*
Philosophy	consume 1 pattern match argument	

	`primrec`	`primcorec`
Syntactic criterion for of functions of type	termination α *list* $\Rightarrow \cdots$	productivity $\cdots \Rightarrow \alpha$ *lang*
Philosophy	consume 1 pattern match argument	produce 1 copattern match output

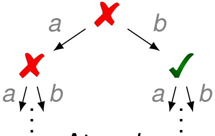
	`primrec`	`primcorec`
Syntactic criterion for of functions of type	termination α *list* $\Rightarrow \cdots$	productivity $\cdots \Rightarrow \alpha$ *lang*
Philosophy	consume 1 pattern match argument	produce 1 copattern match output
(Co)recursive call		
arguments	very restricted	
context	arbitrary	

	`primrec`	`primcorec`
Syntactic criterion for of functions of type	termination α *list* $\Rightarrow \cdots$	productivity $\cdots \Rightarrow \alpha$ *lang*
Philosophy	consume 1 pattern match argument	produce 1 copattern match output
(Co)recursive call		
arguments	very restricted	arbitrary
context	arbitrary	very restricted

	`primrec`	`primcorec`
Syntactic criterion for of functions of type	termination α *list* $\Rightarrow \cdots$	productivity $\cdots \Rightarrow \alpha$ *lang*
Philosophy	consume 1 — pattern match argument	produce 1 — copattern match output
(Co)recursive call		
arguments	very restricted	arbitrary
context	arbitrary	very restricted

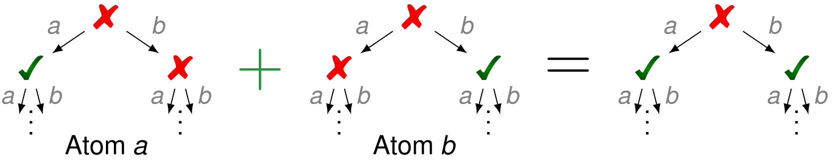


Atom *a*



Atom *b*

	`primrec`	`primcorec`
Syntactic criterion for of functions of type	termination α *list* $\Rightarrow \cdots$	productivity $\cdots \Rightarrow \alpha$ *lang*
Philosophy	consume 1 pattern match argument	produce 1 copattern match output
(Co)recursive call		
arguments	very restricted	arbitrary
context	arbitrary	very restricted

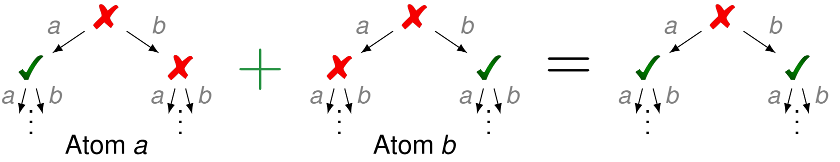


	`primrec`	`primcorec`
Syntactic criterion for of functions of type	termination $\alpha\ \mathit{list} \Rightarrow \cdots$	productivity $\cdots \Rightarrow \alpha\ \mathit{lang}$
Philosophy	consume 1 pattern match argument	produce 1 copattern match output
(Co)recursive call		
arguments	very restricted	arbitrary
context	arbitrary	very restricted

$$\texttt{primcorec}\ + :: \alpha\ \mathit{lang} \Rightarrow \alpha\ \mathit{lang} \Rightarrow \alpha\ \mathit{lang}$$

$$o\ (L + K) = o\ L \vee o\ K$$

$$\delta\ (L + K) = \lambda a.\quad L\ a + \quad K\ a$$



$$o\ \varnothing = \text{✗}$$
$$\varnothing = \lambda\_.\ \varnothing$$

$$o\ \varepsilon = \checkmark$$
$$\varepsilon = \lambda\_.\ \varnothing$$

$$o\ (\text{Atom } a) = \text{✗}$$
$$(\text{Atom } a) = \lambda b.\ \texttt{if}\ a = b\ \texttt{then}\ \varepsilon\ \texttt{else}\ \varnothing$$

$$o\ (L + K) = o\ L \vee o\ K$$
$$(L + K) = \lambda a.\quad L\ a + \quad K\ a$$

$$
\begin{aligned}
o\,\emptyset &= ✗ \\
o\,\varepsilon &= ✓ \\
o\,(\mathrm{Atom}\ a) &= ✗ \\
o\,(L + K) &= o\,L \vee o\,K
\end{aligned}
$$

$L, K :: \alpha\ lang$

$L + \emptyset = L$

$$
\begin{aligned}
\emptyset &= \lambda\_.\ \emptyset \\
\varepsilon &= \lambda\_.\ \emptyset \\
(\mathrm{Atom}\ a) &= \lambda b.\ \texttt{if}\ a = b\ \texttt{then}\ \varepsilon\ \texttt{else}\ \emptyset \\
(L + K) &= \lambda a.\quad L\,a + \quad K\,a
\end{aligned}
$$

$$L, K :: \alpha\ \mathit{regex}$$

$$L + \varnothing \neq L$$

$$
\begin{aligned}
\mathsf{o}\ \varnothing &= \text{✗} \\
\mathsf{o}\ \varepsilon &= \checkmark \\
\mathsf{o}\ (\mathsf{Atom}\ a) &= \text{✗} \\
\mathsf{o}\ (L + K) &= \mathsf{o}\ L \vee \mathsf{o}\ K
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{d}\ \varnothing &= \lambda\_.\ \varnothing \\
\mathsf{d}\ \varepsilon &= \lambda\_.\ \varnothing \\
\mathsf{d}\ (\mathsf{Atom}\ a) &= \lambda b.\ \mathtt{if}\ a = b\ \mathtt{then}\ \varepsilon\ \mathtt{else}\ \varnothing \\
\mathsf{d}\ (L + K) &= \lambda a.\ \mathsf{d}\ L\ a + \mathsf{d}\ K\ a
\end{aligned}
$$

$$L, K :: \alpha\ \mathit{regex}$$

$$L + \varnothing \neq L$$

$$
\begin{aligned}
\mathsf{o}\,\varnothing &= \text{✗} \\
\mathsf{o}\,\varepsilon &= \checkmark \\
\mathsf{o}\,(\mathsf{Atom}\ a) &= \text{✗} \\
\mathsf{o}\,(L + K) &= \mathsf{o}\,L \vee \mathsf{o}\,K \\
\mathsf{o}\,(L \cdot K) &= \mathsf{o}\,L \wedge \mathsf{o}\,K \\
\mathsf{o}\,(L^*) &= \checkmark
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{d}\,\varnothing &= \lambda\_.\ \varnothing \\
\mathsf{d}\,\varepsilon &= \lambda\_.\ \varnothing \\
\mathsf{d}\,(\mathsf{Atom}\ a) &= \lambda b.\ \mathtt{if}\ a = b\ \mathtt{then}\ \varepsilon\ \mathtt{else}\ \varnothing \\
\mathsf{d}\,(L + K) &= \lambda a.\ \mathsf{d}\,L\,a + \mathsf{d}\,K\,a \\
\mathsf{d}\,(L \cdot K) &= \ldots \\
\mathsf{d}\,(L^*) &= \ldots
\end{aligned}
$$

$$L, K :: \alpha\ regex \qquad L + \varnothing \neq L$$

$$\text{nullability}\begin{cases}
\mathsf{o}\ \varnothing &= \text{✗} \\
\mathsf{o}\ \varepsilon &= \checkmark \\
\mathsf{o}\ (\mathsf{Atom}\ a) &= \text{✗} \\
\mathsf{o}\ (L + K) &= \mathsf{o}\ L \lor \mathsf{o}\ K \\
\mathsf{o}\ (L \cdot K) &= \mathsf{o}\ L \land \mathsf{o}\ K \\
\mathsf{o}\ (L^*) &= \checkmark
\end{cases}$$

$$\begin{array}{r}\text{Brzozowski}\\ \text{derivative}\end{array}\begin{cases}
\mathsf{d}\ \varnothing &= \lambda_.\ \varnothing \\
\mathsf{d}\ \varepsilon &= \lambda_.\ \varnothing \\
\mathsf{d}\ (\mathsf{Atom}\ a) &= \lambda b.\ \mathtt{if}\ a = b\ \mathtt{then}\ \varepsilon\ \mathtt{else}\ \varnothing \\
\mathsf{d}\ (L + K) &= \lambda a.\ \mathsf{d}\ L\ a + \mathsf{d}\ K\ a \\
\mathsf{d}\ (L \cdot K) &= \dots \\
\mathsf{d}\ (L^*) &= \dots
\end{cases}$$

$L, K :: \alpha\ \mathit{regex}$

$L + \varnothing \neq L$

$$\mathsf{o}\,(L \cdot K) \quad = \quad \mathsf{o}\,L \wedge \mathsf{o}\,K$$

$$\mathsf{d}\,(L \cdot K) \quad = \quad \ldots$$

$$
\begin{aligned}
&\texttt{primcorec} \cdot :: \alpha\, \mathit{lang} \Rightarrow \alpha\, \mathit{lang} \Rightarrow \alpha\, \mathit{lang} \\
&\quad o\,(L \cdot K) = o\, L \wedge o\, K \\
&\quad\ \ (L \cdot K) = \lambda a.\ \texttt{if}\ o\, L \\
&\qquad\qquad\qquad\qquad \texttt{then}\ \ L\, a \cdot K + \ \ K\, a \\
&\qquad\qquad\qquad\qquad \texttt{else}\ \ L\, a \cdot K
\end{aligned}
$$

Nonprimitively corecursive specification

$$
\begin{aligned}
&\texttt{primcorec}\ \cdot :: \alpha\, lang \Rightarrow \alpha\, lang \Rightarrow \alpha\, lang\\
&\quad o\,(L \cdot K) = o\, L \wedge o\, K\\
&\quad \ (L \cdot K) = \lambda a.\ \texttt{if}\ o\, L\\
&\qquad\qquad\qquad\qquad \texttt{then}\ \ L\, a \cdot K + \ \ K\, a\\
&\qquad\qquad\qquad\qquad \texttt{else}\ \ L\, a \cdot K
\end{aligned}
$$

Nonprimitively corecursive specification

$$\texttt{primcorec} \cdot :: \alpha\, lang \Rightarrow \alpha\, lang \Rightarrow \alpha\, lang$$
$$o\,(L \cdot K) = o\, L \wedge o\, K$$
$$\partial\,(L \cdot K) = \lambda a.\ \texttt{if}\ o\, L$$
$$\texttt{then}\quad \partial\, L\, a \cdot K + \partial\, K\, a$$
$$\texttt{else}\quad \partial\, L\, a \cdot K$$

This paper

$$L \cdot K = \hat{\bigoplus}\,(L \mathbin{\hat{\cdot}} K)$$
with $\hat{\cdot}$ and $\hat{\bigoplus}$ primitively corecursive

$$\texttt{corec}\ \cdot :: \alpha\ lang \Rightarrow \alpha\ lang \Rightarrow \alpha\ lang$$

$$o\ (L \cdot K) = o\ L \wedge o\ K$$

$$(L \cdot K) = \lambda a.\ \texttt{if}\ o\ L$$

$$\texttt{then}\quad L\ a \cdot K + \quad K\ a$$

$$\texttt{else}\quad L\ a \cdot K$$

This paper

$$L \cdot K = \hat{\bigoplus}\ (L \mathbin{\hat{\cdot}} K)$$

with $\hat{\cdot}$ and $\hat{\bigoplus}$ primitively corecursive

Blanchette, Popescu, Traytel @ ICFP'15 +
Blanchette, Bouzy, Lochbihler, Popescu, Traytel

`corec`

Codatatype

Corecursion

Coinduction

`theorem` $\varnothing + L = L$

How to prove?

```
theorem  ∅ + L = L
    by (coinduction arbitrary: L) auto
```

theorem $\varnothing + L = L$

proof (*rule coinduct*$_{lang}$)

    define $R\ K_1\ K_2 = (\exists L.\ K_1 = \varnothing + L \wedge K_2 = L)$

    show $R\ (\varnothing + L)\ L$ by *simp*

    fix L_1 and L_2
    assume $R\ L_1\ L_2$
    then obtain L where $L_1 = \varnothing + L$ and $L_2 = L$ by *simp*
    then **show** $o\ L_1 = o\ L_2 \wedge \forall x.\ R\ (\quad L_1\ x)\ (\quad L_2\ x)$ by *simp*

qed

$$\frac{R\ K_1\ K_2 \qquad \forall L_1\ L_2.\ R\ L_1\ L_2 \longrightarrow (o\ L_1 = o\ L_2 \wedge \forall x.\ R\ (\quad L_1\ x)\ (\quad L_2\ x))}{K_1 = K_2}$$

theorem $\varnothing + L = L$

proof (*rule coinduct*$_{lang}$)

define $R\ K_1\ K_2 = (\exists L.\ K_1 = \varnothing + L \wedge K_2 = L)$

show $R\ (\varnothing + L)\ L$ by *simp*

fix L_1 and L_2
assume $R\ L_1\ L_2$
then obtain L where $L_1 = \varnothing + L$ and $L_2 = L$ by *simp*
then **show** $o\ L_1 = o\ L_2 \wedge \forall x.\ R\ (\quad L_1\ x)\ (\quad L_2\ x)$ by *simp*

qed

$$\frac{R\ K_1\ K_2 \qquad \forall L_1\ L_2.\ R\ L_1\ L_2 \longrightarrow (o\ L_1 = o\ L_2 \wedge \forall x.\ R\ (\ \ L_1\ x)\ (\ \ L_2\ x))}{K_1 = K_2}$$

```
theorem ∅ + L = L
proof (rule coinduct_lang)
    define R K₁ K₂ = (∃L. K₁ = ∅ + L ∧ K₂ = L)

    show R (∅ + L) L by simp

    fix L₁ and L₂
    assume R L₁ L₂
    then obtain L where L₁ = ∅ + L and L₂ = L by simp
    then show o L₁ = o L₂ ∧ ∀x. R (  L₁ x) (  L₂ x) by simp
qed
```

$$\frac{R\ \ K_1\ \ K_2 \qquad \forall L_1\ L_2.\ R\ L_1\ L_2 \longrightarrow (o\ L_1 = o\ L_2 \land \forall x.\ R\ (\quad L_1\ x)\ (\quad L_2\ x))}{K_1 = K_2}$$

theorem $\emptyset + L = L$

proof (*rule coinduct*$_{lang}$)

  define $R\ K_1\ K_2 = (\exists L.\ K_1 = \emptyset + L \land K_2 = L)$

  show $R\ (\emptyset + L)\ L$ by *simp*

  fix L_1 and L_2

  assume $R\ L_1\ L_2$

  then obtain L where $L_1 = \emptyset + L$ and $L_2 = L$ by *simp*

  then **show** $o\ L_1 = o\ L_2 \land \forall x.\ R\ (\quad L_1\ x)\ (\quad L_2\ x)$ by *simp*

qed

$$\frac{R\ K_1\ K_2 \qquad \forall L_1\ L_2.\ R\ L_1\ L_2 \longrightarrow (o\ L_1 = o\ L_2 \wedge \forall x.\ R\ (\quad L_1\ x)\ (\quad L_2\ x))}{K_1 = K_2}$$

theorem $\varnothing + L = L$

proof ($\mathit{rule}\ \mathit{coinduct}_{\mathit{lang}}$)

  define $R\ K_1\ K_2 = (\exists L.\ K_1 = \varnothing + L \wedge K_2 = L)$

  show $R\ (\varnothing + L)\ L$ by *simp*

  fix L_1 and L_2

  assume $R\ L_1\ L_2$

  then obtain L where $L_1 = \varnothing + L$ and $L_2 = L$ by *simp*

  then **show** $o\ L_1 = o\ L_2 \wedge \forall x.\ R\ (\quad L_1\ x)\ (\quad L_2\ x)$ by *simp*

qed

$$\frac{R\ K_1\ K_2 \qquad \forall L_1\ L_2.\ R\ L_1\ L_2 \longrightarrow (o\ L_1 = o\ L_2 \wedge \forall x.\ R\ (\quad L_1\ x)\ (\quad L_2\ x))}{K_1 = K_2}$$

theorem $\varnothing + L = L$

proof (r…

d…

sh…

fix L_1 and L_2

assume $R\ L_1\ L_2$

then obtain L where $L_1 = \varnothing + L$ and $L_2 = L$ by *simp*

then **show** $o\ L_1 = o\ L_2 \wedge \forall x.\ R\ (\quad L_1\ x)\ (\quad L_2\ x)$ by *simp*

qed

$o\ L_1 = o\ (\varnothing + L) = (o\ \varnothing \vee o\ L) = (✗ \vee o\ L) = o\ L = o\ L_2$

$$\begin{aligned} R\ (\quad L_1\ x)\ (\quad L_2\ x) &= R\ (\quad (\varnothing + L)\ x)\ (\quad L\ x) \\ &= R\ (\quad \varnothing\ x + \quad L\ x)\ (\quad L\ x) = R\ (\varnothing + \quad L\ x)\ (\quad L\ x) \\ &= (\exists L'.\ \varnothing + \quad L\ x = \varnothing + L' \wedge \quad L\ x = L') = ✓ \end{aligned}$$

theorem $\varnothing + L = L$

proof ($\mathit{rule\ coinduct}_{lang}$)

  define $R\ K_1\ K_2 = (\exists L.\ K_1 = \varnothing + L \land K_2 = L)$

  show $R\ (\varnothing + L)\ L$ by *simp*

  fix L_1 and L_2

  assume $R\ L_1\ L_2$

  then obtain L where $L_1 = \varnothing + L$ and $L_2 = L$ by *simp*

  then **show** $o\ L_1 = o\ L_2 \land \forall x.\ R\ (\quad L_1\ x)\ (\quad L_2\ x)$ by *simp*

qed

```
theorem  ∅ + L = L
    by (coinduction arbitrary: L) auto
```

theorem $\varnothing + L = L$

proof $(\mathit{rule\ coinduct}_{lang})$

define $R\ K_1\ K_2 = (\exists L.\ K_1 = \varnothing + L \wedge K_2 = L)$

show $R\ (\varnothing + L)\ L$ by *simp*

fix L_1 and L_2

assume $R\ L_1\ L_2$

then obtain L where $L_1 = \varnothing + L$ and $L_2 = L$ by *simp*

then **show** $o\ L_1 = o\ L_2 \wedge \forall x.\ R\ (\ \ L_1\ x)\ (\ \ L_2\ x)$ by *simp*

qed

`theorem` $L + L = L$
  `by` $(coinduction\ arbitrary: L)\ auto$

`theorem` $\varnothing + L = L$
  `by` $(coinduction\ arbitrary: L)\ auto$

`theorem` $L_1 + L_2 = L_2 + L_1$
  `by` $(coinduction\ arbitrary: L_1\ L_2)\ auto$

`theorem` $(L_1 + L_2) + L_3 = L_1 + (L_2 + L_3)$
  `by` $(coinduction\ arbitrary: L_1\ L_2\ L_3)\ auto$

```
theorem   (L + K) · M = (L · M) + (K · M)
    by (coinduction arbitrary: K L) auto
```

`theorem` $(L + K) \cdot M = (L \cdot M) + (K \cdot M)$

`by` $(\mathit{coinduction\ arbitrary} : K\ L)\ \mathit{auto}$

$$\cdots \longrightarrow R = \lambda L_1\ L_2.\ \exists L\ K.\ L_1 = (L + K) \cdot M \wedge L_2 = (L \cdot M) + (K \cdot M) \longrightarrow$$

$$R\ ((L'\ x + K'\ x) \cdot M + M\ x)$$
$$((L'\ x \cdot M + K'\ x \cdot M) + M\ x)$$

$$\texttt{theorem} \quad (L + K) \cdot M = (L \cdot M) + (K \cdot M)$$

$$\texttt{by}\ (\mathit{coinduction\ arbitrary} : L\ K\ \mathit{rule} : \mathit{coinduct}^{+}_{\mathit{lang}})\ \mathit{auto}$$

$$\texttt{theorem} \quad (L + K) \cdot M = (L \cdot M) + (K \cdot M)$$

$$\texttt{by}\ (\mathit{coinduction\ arbitrary} : L\ K\ \mathit{rule} : \mathit{coinduct}^{+}_{\mathit{lang}})\ \mathit{auto}$$

$$\frac{R\ K_1\ K_2 \qquad \forall L_1\ L_2.\ R\ L_1\ L_2 \longrightarrow (o\ L_1 = o\ L_2 \land \forall x.\ R^{+}(\ \ L_1\ x)\ (\ \ L_2\ x))}{K_1 = K_2}$$

Languages, Languages, Languages

`codatatype` $\alpha\,\mathit{lang} = \mathscr{L}\ \mathit{bool}\ (\alpha \Rightarrow \alpha\,\mathit{lang})$

Languages, Languages, Languages

`codatatype` $\alpha\, \mathit{lang} = \mathscr{L}\ \mathit{bool}\ (\alpha \Rightarrow \alpha\, \mathit{lang})$

`codatatype` $(\alpha, \beta)\, \mathit{lang}_{\mathit{Moore}} = \mathscr{L}_{\mathit{Moore}}\, \beta\ (\alpha \Rightarrow (\alpha, \beta)\, \mathit{lang}_{\mathrm{Moore}})$

`codatatype` $(\alpha, \beta)\, \mathit{lang}_{\mathit{Mealy}} = \mathscr{L}_{\mathit{Mealy}}\ (\alpha \Rightarrow \beta)\ (\alpha \Rightarrow (\alpha, \beta)\, \mathit{lang}_{\mathit{Mealy}})$

Languages, Languages, Languages

`codatatype` $\alpha\,\mathit{lang} = \mathscr{L}\ \mathit{bool}\ (\alpha \Rightarrow \alpha\,\mathit{lang})$

`codatatype` $(\alpha,\beta)\,\mathit{lang}_{\mathit{Moore}} = \mathscr{L}_{\mathit{Moore}}\,\beta\ (\alpha \Rightarrow (\alpha,\beta)\,\mathit{lang}_{\mathrm{Moore}})$

`codatatype` $(\alpha,\beta)\,\mathit{lang}_{\mathit{Mealy}} = \mathscr{L}_{\mathit{Mealy}}\,(\alpha \Rightarrow \beta)\ (\alpha \Rightarrow (\alpha,\beta)\,\mathit{lang}_{\mathit{Mealy}})$

`codatatype` $\alpha\,\mathit{lang}_{\mathit{Nondet}} = \mathscr{L}_{\mathit{Nondet}}\ \mathit{bool}\ (\alpha \Rightarrow \alpha\,\mathit{lang}_{\mathit{Nondet}}\ \mathit{set}^{\kappa})$

Languages, Languages, Languages

$\texttt{codatatype}\ \alpha\, \mathit{lang} = \mathscr{L}\ \mathit{bool}\ (\alpha \Rightarrow \alpha\, \mathit{lang})$

$\texttt{codatatype}\ (\alpha, \beta)\, \mathit{lang}_{\mathit{Moore}} = \mathscr{L}_{\mathit{Moore}}\, \beta\ (\alpha \Rightarrow (\alpha, \beta)\, \mathit{lang}_{\mathrm{Moore}})$

$\texttt{codatatype}\ (\alpha, \beta)\, \mathit{lang}_{\mathit{Mealy}} = \mathscr{L}_{\mathit{Mealy}}\ (\alpha \Rightarrow \beta)\ (\alpha \Rightarrow (\alpha, \beta)\, \mathit{lang}_{\mathit{Mealy}})$

$\texttt{codatatype}\ \alpha\, \mathit{lang}_{\mathit{Nondet}} = \mathscr{L}_{\mathit{Nondet}}\ \mathit{bool}\ (\alpha \Rightarrow \alpha\, \mathit{lang}_{\mathit{Nondet}}\ \mathit{set}^{\kappa})$

$\texttt{codatatype}\ \alpha\, \mathit{lang}_{\omega} = \mathscr{L}_{\omega}\ (\alpha\, \mathit{lang})\ (\alpha \Rightarrow \alpha\, \mathit{lang}_{\omega})$

Languages, Languages, Languages

$\mathtt{codatatype}\ \alpha\, \mathit{lang} = \mathscr{L}\ \mathit{bool}\ (\alpha \Rightarrow \alpha\, \mathit{lang})$

$\mathtt{codatatype}\ (\alpha, \beta)\, \mathit{lang}_{\mathit{Moore}} = \mathscr{L}_{\mathit{Moore}}\, \beta\ (\alpha \Rightarrow (\alpha, \beta)\, \mathit{lang}_{\mathrm{Moore}})$

$\mathtt{codatatype}\ (\alpha, \beta)\, \mathit{lang}_{\mathit{Mealy}} = \mathscr{L}_{\mathit{Mealy}}\ (\alpha \Rightarrow \beta)\ (\alpha \Rightarrow (\alpha, \beta)\, \mathit{lang}_{\mathit{Mealy}})$

$\mathtt{codatatype}\ \alpha\, \mathit{lang}_{\mathit{Nondet}} = \mathscr{L}_{\mathit{Nondet}}\ \mathit{bool}\ (\alpha \Rightarrow \alpha\, \mathit{lang}_{\mathit{Nondet}}\ \mathit{set}^{\kappa})$

$\mathtt{codatatype}\ \alpha\, \mathit{lang}_{\omega} = \mathscr{L}_{\omega}\ (\alpha\, \mathit{lang})\ (\alpha \Rightarrow \alpha\, \mathit{lang}_{\omega})$

$\mathtt{codatatype}\ \alpha\, \mathit{lang}_{\mathit{Markov}} = \mathscr{L}_{\mathit{Markov}}\ (\alpha \times \alpha\, \mathit{lang}_{\mathit{Markov}})\ \mathit{pmf}$

$\mathtt{codatatype}\ \alpha\, \mathit{lang}_{\mathit{React}} = \mathscr{L}_{\mathit{React}}\ (\alpha \Rightarrow \alpha\, \mathit{lang}_{\mathit{React}}\ \mathit{pmf}\ \mathit{option})$

$\mathtt{codatatype}\ \alpha\, \mathit{lang}_{\mathit{Segala}} = \mathscr{L}_{\mathit{Segala}}\ (\alpha \times \alpha\, \mathit{lang}_{\mathit{Segala}})\ \mathit{pmf}\ \mathit{set}^{\kappa}$

Hölzl, Lochbihler, Traytel, ITP 2015

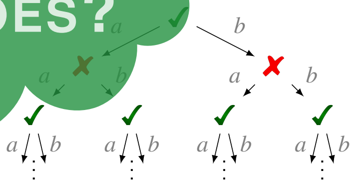
Formal Languages

$baab$ aa bb ε ba ab $\ldots$

OBRIGADO!

Formally and Coinductively

QUESTÕES?



Dmitriy Traytel

ETH zürich

Puzzle for the RTA Community

Brzozowski Derivatives

$$
\begin{array}{lcl}
\mathsf{d}\; a\; \varnothing & = & \varnothing \\
\mathsf{d}\; a\; \varepsilon & = & \varnothing \\
\mathsf{d}\; a\; (\mathrm{Atom}\; b) & = & \texttt{if}\; a = b\; \texttt{then}\; \varepsilon\; \texttt{else}\; \varnothing \\
\mathsf{d}\; a\; (r + s) & = & \mathsf{d}\; a\; r + \mathsf{d}\; a\; s \\
\mathsf{d}\; a\; (r \cdot s) & = & \texttt{if}\; \texttt{o}\; r\; \texttt{then}\; \mathsf{d}\; a\; r \cdot s + \mathsf{d}\; a\; s\; \texttt{else}\; \mathsf{d}\; a\; r \cdot s \\
\mathsf{d}\; a\; (r^*) & = & \mathsf{d}\; a\; r \cdot r^*
\end{array}
$$

Brzozowski Derivatives

Finiteness

$$
\begin{array}{lcl}
\mathsf{d}\,a\,\varnothing & = & \varnothing \\
\mathsf{d}\,a\,\varepsilon & = & \varnothing \\
\mathsf{d}\,a\,(\mathsf{Atom}\ b) & = & \mathtt{if}\ a = b\ \mathtt{then}\ \varepsilon\ \mathtt{else}\ \varnothing \\
\mathsf{d}\,a\,(r + s) & = & \mathsf{d}\,a\,r + \mathsf{d}\,a\,s \\
\mathsf{d}\,a\,(r \cdot s) & = & \mathtt{if}\ \mathsf{o}\ r\ \mathtt{then}\ \mathsf{d}\,a\,r \cdot s + \mathsf{d}\,a\,s\ \mathtt{else}\ \mathsf{d}\,a\,r \cdot s \\
\mathsf{d}\,a\,(r^*) & = & \mathsf{d}\,a\,r \cdot r^*
\end{array}
$$

Brzozowski Derivatives

Finiteness

$$
\begin{array}{lcl}
\mathrm{d}\, a\, \varnothing & = & \varnothing \\
\mathrm{d}\, a\, \varepsilon & = & \varnothing \\
\mathrm{d}\, a\, (\mathrm{Atom}\; b) & = & \mathtt{if}\; a = b\; \mathtt{then}\; \varepsilon\; \mathtt{else}\; \varnothing \\
\mathrm{d}\, a\, (r + s) & = & \mathrm{d}\, a\, r + \mathrm{d}\, a\, s \\
\mathrm{d}\, a\, (r \cdot s) & = & \mathtt{if}\; \mathrm{o}\; r\; \mathtt{then}\; \mathrm{d}\, a\, r \cdot s + \mathrm{d}\, a\, s\; \mathtt{else}\; \mathrm{d}\, a\, r \cdot s \\
\mathrm{d}\, a\, (r^*) & = & \mathrm{d}\, a\, r \cdot r^*
\end{array}
$$

$$
\begin{array}{lcl}
\hat{\mathrm{d}}\; []\; r & = & r \\
\hat{\mathrm{d}}\; aw\; r & = & \hat{\mathrm{d}}\; w\; (\mathrm{d}\; a\; r)
\end{array}
$$

Brzozowski Derivatives

Finiteness

$$
\begin{array}{lcl}
\mathsf{d}\,a\,\varnothing & = & \varnothing \\
\mathsf{d}\,a\,\varepsilon & = & \varnothing \\
\mathsf{d}\,a\,(\mathrm{Atom}\ b) & = & \mathsf{if}\ a = b\ \mathsf{then}\ \varepsilon\ \mathsf{else}\ \varnothing \\
\mathsf{d}\,a\,(r + s) & = & \mathsf{d}\,a\,r + \mathsf{d}\,a\,s \\
\mathsf{d}\,a\,(r \cdot s) & = & \mathsf{if}\ \mathsf{o}\ r\ \mathsf{then}\ \mathsf{d}\,a\,r \cdot s + \mathsf{d}\,a\,s\ \mathsf{else}\ \mathsf{d}\,a\,r \cdot s \\
\mathsf{d}\,a\,(r^*) & = & \mathsf{d}\,a\,r \cdot r^*
\end{array}
$$

$$
\begin{array}{lcl}
\hat{\mathsf{d}}\,[]\,r & = & r \\
\hat{\mathsf{d}}\,aw\,r & = & \hat{\mathsf{d}}\,w\,(\mathsf{d}\,a\,r)
\end{array}
$$

Theorem [Brzozowski'64] $\{[\hat{\mathsf{d}}\ w\ r]_{\mathrm{ACI}} \mid w \in \Sigma^*\}$ is finite for all r

Brzozowski Derivatives

Finiteness

$$
\begin{array}{lcl}
\mathrm{d}\,a\,\varnothing & = & \varnothing \\
\mathrm{d}\,a\,\varepsilon & = & \varnothing \\
\mathrm{d}\,a\,(\mathrm{Atom}\ b) & = & \mathtt{if}\ a = b\ \mathtt{then}\ \varepsilon\ \mathtt{else}\ \varnothing \\
\mathrm{d}\,a\,(r + s) & = & \mathrm{d}\,a\,r + \mathrm{d}\,a\,s \\
\mathrm{d}\,a\,(r \cdot s) & = & \mathtt{if}\ \mathrm{o}\ r\ \mathtt{then}\ \mathrm{d}\,a\,r \cdot s + \mathrm{d}\,a\,s\ \mathtt{else}\ \mathrm{d}\,a\,r \cdot s \\
\mathrm{d}\,a\,(r^*) & = & \mathrm{d}\,a\,r \cdot r^*
\end{array}
$$

$$
\begin{array}{lcl}
\hat{\mathrm{d}}\,[]\,r & = & r \\
\hat{\mathrm{d}}\,aw\,r & = & \hat{\mathrm{d}}\,w\,(\mathrm{d}\,a\,r)
\end{array}
$$

Theorem [Brzozowski'64] $\{[\hat{\mathrm{d}}\ w\ r]_{\mathrm{ACI}} \mid w \in \Sigma^*\}$ is finite for all r

Corollary $\{[\hat{\mathrm{d}}\ w\ r]_{\mathrm{ACIUD}} \mid w \in \Sigma^*\}$ is finite for all r

Brzozowski Derivatives

Finiteness

$$\begin{array}{lcl}
\mathsf{d}\ a\ \varnothing & = & \varnothing \\
\mathsf{d}\ a\ \varepsilon & = & \varnothing \\
\mathsf{d}\ a\ (\mathsf{Atom}\ b) & = & \mathsf{if}\ a = b\ \mathsf{then}\ \varepsilon\ \mathsf{else}\ \varnothing \\
\mathsf{d}\ a\ (r + s) & = & \mathsf{d}\ a\ r + \mathsf{d}\ a\ s \\
\mathsf{d}\ a\ (r \cdot s) & = & \mathsf{if}\ \mathsf{o}\ r\ \mathsf{then}\ \mathsf{d}\ a\ r \cdot s + \mathsf{d}\ a\ s\ \mathsf{else}\ \mathsf{d}\ a\ r \cdot s \\
\mathsf{d}\ a\ (r^*) & = & \mathsf{d}\ a\ r \cdot r^*
\end{array}$$

$$\begin{array}{lcl}
\hat{\mathsf{d}}\ []\ r & = & r \\
\hat{\mathsf{d}}\ aw\ r & = & \hat{\mathsf{d}}\ w\ (\mathsf{d}\ a\ r)
\end{array}$$

$$\begin{array}{lcl}
\hat{\mathsf{d}}_{\mathsf{ACI}}\ []\ r & = & |r|_{\mathsf{ACI}} \\
\hat{\mathsf{d}}_{\mathsf{ACI}}\ aw\ r & = & \hat{\mathsf{d}}_{\mathsf{ACI}}\ w\ |\mathsf{d}\ a\ r|_{\mathsf{ACI}}
\end{array}$$

Theorem [Brzozowski'64] $\{[\hat{\mathsf{d}}\ w\ r]_{\mathsf{ACI}} \mid w \in \Sigma^*\}$ is finite for all r

Corollary $\{[\hat{\mathsf{d}}\ w\ r]_{\mathsf{ACIUD}} \mid w \in \Sigma^*\}$ is finite for all r

Brzozowski Derivatives

Finiteness

$$
\begin{array}{lcl}
\mathrm{d}\,a\,\varnothing & = & \varnothing \\
\mathrm{d}\,a\,\varepsilon & = & \varnothing \\
\mathrm{d}\,a\,(\mathrm{Atom}\ b) & = & \mathtt{if}\ a = b\ \mathtt{then}\ \varepsilon\ \mathtt{else}\ \varnothing \\
\mathrm{d}\,a\,(r + s) & = & \mathrm{d}\,a\,r + \mathrm{d}\,a\,s \\
\mathrm{d}\,a\,(r \cdot s) & = & \mathtt{if}\ \mathrm{o}\ r\ \mathtt{then}\ \mathrm{d}\,a\,r \cdot s + \mathrm{d}\,a\,s\ \mathtt{else}\ \mathrm{d}\,a\,r \cdot s \\
\mathrm{d}\,a\,(r^*) & = & \mathrm{d}\,a\,r \cdot r^*
\end{array}
$$

$$
\begin{array}{lcl}
\hat{\mathrm{d}}\ []\ r & = & r \\
\hat{\mathrm{d}}\ aw\ r & = & \hat{\mathrm{d}}\ w\ (\mathrm{d}\ a\ r)
\end{array}
$$

$$
\begin{array}{lcl}
\hat{\mathrm{d}}_{\mathrm{ACI}}\ []\ r & = & |r|_{\mathrm{ACI}} \\
\hat{\mathrm{d}}_{\mathrm{ACI}}\ aw\ r & = & \hat{\mathrm{d}}_{\mathrm{ACI}}\ w\ |\mathrm{d}\ a\ r|_{\mathrm{ACI}}
\end{array}
$$

Theorem [Brzozowski'64] $\{[\hat{\mathrm{d}}\ w\ r]_{\mathrm{ACI}} \mid w \in \Sigma^*\}$ is finite for all r

Corollary $\{[\hat{\mathrm{d}}\ w\ r]_{\mathrm{ACIUD}} \mid w \in \Sigma^*\}$ is finite for all r

Lemma $|\mathrm{d}\ a\ |r|_{\mathrm{ACI}}|_{\mathrm{ACI}} = |\mathrm{d}\ a\ r|_{\mathrm{ACI}}$

Corollary $\{\hat{\mathrm{d}}_{\mathrm{ACI}}\ w\ r \mid w \in \Sigma^*\}$ is finite for all r

Brzozowski Derivatives

Finiteness

$$
\begin{array}{lcl}
\mathsf{d}\ a\ \varnothing & = & \varnothing \\
\mathsf{d}\ a\ \varepsilon & = & \varnothing \\
\mathsf{d}\ a\ (\mathsf{Atom}\ b) & = & \mathsf{if}\ a = b\ \mathsf{then}\ \varepsilon\ \mathsf{else}\ \varnothing \\
\mathsf{d}\ a\ (r + s) & = & \mathsf{d}\ a\ r + \mathsf{d}\ a\ s \\
\mathsf{d}\ a\ (r \cdot s) & = & \mathsf{if}\ \mathsf{o}\ r\ \mathsf{then}\ \mathsf{d}\ a\ r \cdot s + \mathsf{d}\ a\ s\ \mathsf{else}\ \mathsf{d}\ a\ r \cdot s \\
\mathsf{d}\ a\ (r^*) & = & \mathsf{d}\ a\ r \cdot r^*
\end{array}
$$

$$
\begin{array}{lcl}
\hat{\mathsf{d}}\ []\ r & = & r \\
\hat{\mathsf{d}}\ aw\ r & = & \hat{\mathsf{d}}\ w\ (\mathsf{d}\ a\ r)
\end{array}
$$

$$
\begin{array}{lcl}
\hat{\mathsf{d}}_{\mathsf{ACI}}\ []\ r & = & |r|_{\mathsf{ACI}} \\
\hat{\mathsf{d}}_{\mathsf{ACI}}\ aw\ r & = & \hat{\mathsf{d}}_{\mathsf{ACI}}\ w\ |\mathsf{d}\ a\ r|_{\mathsf{ACI}}
\end{array}
$$

Theorem [Brzozowski'64] $\{[\hat{\mathsf{d}}\ w\ r]_{\mathsf{ACI}} \mid w \in \Sigma^*\}$ is finite for all r

Corollary $\{[\hat{\mathsf{d}}\ w\ r]_{\mathsf{ACIUD}} \mid w \in \Sigma^*\}$ is finite for all r

Lemma $|\mathsf{d}\ a\ |r|_{\mathsf{ACI}}|_{\mathsf{ACI}} = |\mathsf{d}\ a\ r|_{\mathsf{ACI}}$

Corollary $\{\hat{\mathsf{d}}_{\mathsf{ACI}}\ w\ r \mid w \in \Sigma^*\}$ is finite for all r

Conjecture $\{\hat{\mathsf{d}}_{\mathsf{ACIUD}}\ w\ r \mid w \in \Sigma^*\}$ is finite for all r

Brzozowski Derivatives

Finiteness as a Rewriting Problem

Input Convergent ordered regex rewriting system R with $\mathsf{ACI} \subseteq \mathsf{R}$

Question Under which conditions is $\{\hat{\mathsf{d}}_{\mathsf{R}}\ w\ r \mid w \in \Sigma^*\}$ finite for all r?

Brzozowski Derivatives

Finiteness as a Rewriting Problem

Input Convergent ordered regex rewriting system R with $\mathsf{ACI} \subseteq \mathsf{R}$

Question Under which conditions is $\{\hat{\mathsf{d}}_{\mathsf{R}}\ w\ r \mid w \in \Sigma^*\}$ finite for all r?

Negative Example $\mathsf{R} = \mathsf{ACI} \cup \{\varepsilon \cdot r^* \to r^* \cdot r^*\}$

Brzozowski Derivatives

Finiteness as a Rewriting Problem

Input Convergent ordered regex rewriting system R with $\mathsf{ACI} \subseteq \mathsf{R}$

Question Under which conditions is $\{\hat{\mathsf{d}}_{\mathsf{R}}\ w\ r \mid w \in \Sigma^*\}$ finite for all r?

Negative Example $\mathsf{R} = \mathsf{ACI} \cup \{\varepsilon \cdot r^* \to r^* \cdot r^*\}$

$$a^* \quad \xrightarrow{\mathsf{d}} \quad \varepsilon \cdot a^*$$

Brzozowski Derivatives

Finiteness as a Rewriting Problem

Input Convergent ordered regex rewriting system R with $\mathsf{ACI} \subseteq \mathsf{R}$

Question Under which conditions is $\{\hat{\mathsf{d}}_{\mathsf{R}}\ w\ r \mid w \in \Sigma^*\}$ finite for all r?

Negative Example $\mathsf{R} = \mathsf{ACI} \cup \{\varepsilon \cdot r^* \to r^* \cdot r^*\}$

$$\begin{aligned} a^* &\overset{\mathsf{d}}{\to} \varepsilon \cdot a^* \\ &\overset{\mathsf{R!}}{\to} a^* \cdot a^* \end{aligned}$$

Brzozowski Derivatives

Finiteness as a Rewriting Problem

Input Convergent ordered regex rewriting system R with $\mathsf{ACI} \subseteq \mathsf{R}$

Question Under which conditions is $\{\hat{\mathsf{d}}_{\mathsf{R}}\ w\ r \mid w \in \Sigma^*\}$ finite for all r?

Negative Example $\mathsf{R} = \mathsf{ACI} \cup \{\varepsilon \cdot r^* \to r^* \cdot r^*\}$

$$
\begin{aligned}
a^* &\xrightarrow{\mathsf{d}} \varepsilon \cdot a^* \\
&\xrightarrow{\mathsf{R}!} a^* \cdot a^* \\
&\xrightarrow{\mathsf{d}} (\varepsilon \cdot a^*) \cdot a^* + \varepsilon \cdot a^*
\end{aligned}
$$

Brzozowski Derivatives

Finiteness as a Rewriting Problem

Input Convergent ordered regex rewriting system R with ACI $\subseteq$ R

Question Under which conditions is $\{\hat{\mathsf{d}}_{\mathsf{R}}\ w\ r \mid w \in \Sigma^*\}$ finite for all r?

Negative Example $\mathsf{R} = \mathsf{ACI} \cup \{\varepsilon \cdot r^* \to r^* \cdot r^*\}$

$$\begin{aligned}
a^* &\xrightarrow{\mathsf{d}} \varepsilon \cdot a^* \\
&\xrightarrow{\mathsf{R}!} a^* \cdot a^* \\
&\xrightarrow{\mathsf{d}} (\varepsilon \cdot a^*) \cdot a^* + \varepsilon \cdot a^* \\
&\xrightarrow{\mathsf{R}!} (a^* \cdot a^*) \cdot a^* + a^* \cdot a^*
\end{aligned}$$

Brzozowski Derivatives

Finiteness as a Rewriting Problem

Input Convergent ordered regex rewriting system R with $\mathsf{ACI} \subseteq \mathsf{R}$

Question Under which conditions is $\{\hat{\mathsf{d}}_{\mathsf{R}}\ w\ r \mid w \in \Sigma^*\}$ finite for all r?

Negative Example $\mathsf{R} = \mathsf{ACI} \cup \{\varepsilon \cdot r^* \to r^* \cdot r^*\}$

$$
\begin{aligned}
a^* &\xrightarrow{\mathsf{d}} \varepsilon \cdot a^* \\
&\xrightarrow{\mathsf{R!}} a^* \cdot a^* \\
&\xrightarrow{\mathsf{d}} (\varepsilon \cdot a^*) \cdot a^* + \varepsilon \cdot a^* \\
&\xrightarrow{\mathsf{R!}} (a^* \cdot a^*) \cdot a^* + a^* \cdot a^* \\
&\xrightarrow{\mathsf{d}} \dots
\end{aligned}
$$