MCFG and Higher Order Grammars

Sylvain Salvati

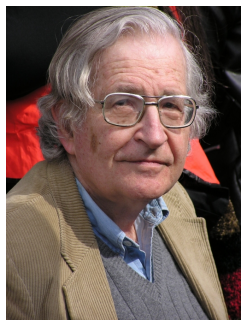
INRIA Bordeaux Sud-Ouest

MCFG+2

Outline

Outline

Linguistics and Mathematics



- Chomsky (2004)

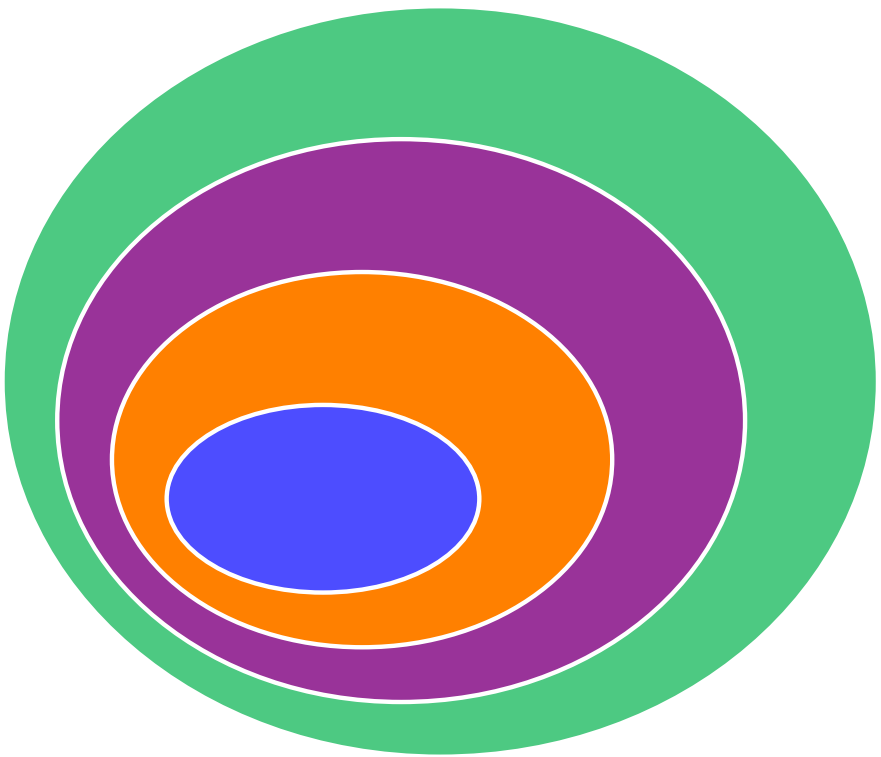
pened. The systems that capture other properties of language, for example transformational grammar, hold no interest for mathematics. But I do not think that that is a necessary truth. It could turn out that there would be richer or more appropriate mathematical ideas that would capture other, maybe deeper properties of language than context-free grammars do. In that case you have another branch of applied mathematics which might have linguistic consequences. That could be exciting.

The origin of mildly context sensitive languages

- Joshi (1985)

Since the late 1970s there has been vigorous activity in constructing highly constrained grammatical systems by eliminating the transformational component either totally or partially. There is increasing recognition of the fact that the entire range of dependencies that transformational grammars in their various incarnations have tried to account for can be captured satisfactorily by classes of rules that are *nontransformational* and at the same time highly constrained in terms of the classes of grammars and languages they define.

The convergences



$MCTAG = MCFG = HR$
$= OUT(DTWT)$
$= yDT_{fc}(REGT) = LUSCG = MG$
$= \lambda\text{-CFL}_{lin}(4) = \lambda\text{-CFL}_{lin} =$
$IO_{sp} = OI_{sp}$

$MCFG_{wn} = IO_{nd} = OI_{nd}$
$= CCFG = \lambda\text{-CFL}_{lin}(3)$
$TAG = LIG = CCG = HG$
$2\text{-}MCFG_{wn}$

$CFL = 1\text{-}MCFL = \lambda\text{-CFL}_{lin}(2)$

Outline

Macro languages: generalizing context free languages

- Fischer (1968)

Definition 2.2.7 (Macro Grammar Structure)

A macro grammar structure is a 6-tuple $(\Sigma, \mathcal{F}, \mathcal{V}, \rho, S, P)$

where:

Σ is a finite set of terminal symbols;

$\mathcal{F}$ is a finite set of non-terminal or function symbols;

$\mathcal{V}$ is a finite set of argument or variable symbols;

ρ is a function from $\mathcal{F}$ into the non-negative integers ($\rho(F)$ is the number of arguments which F takes);

$S \in \mathcal{F}$ is the start or sentence symbol, $\rho(S) = 0$;

P is a finite set of productions or rules of the form $F(x_1, \ldots, x_{\rho(F)}) \rightarrow \nu$ where $F \in \mathcal{F}$, $x_1, \ldots, x_{\rho(F)}$ are distinct members of $\mathcal{V}$, and ν is a quoted term over $\Sigma \cup \{x_1, \ldots, x_{\rho(F)}\}, \mathcal{F}, \rho$.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

IO S

OI S

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ *and* $\mathcal{L}_{OI}$ *are incomparable.*
- $\mathcal{L}_{OI} =$ *Indexed languages.*
- *IO and OI coincide for non-duplicating grammars.*

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

$IO \quad S$

$OI \quad S$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

$IO \quad F(G)$

$OI \quad S$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

IO $F(G)$

OI S

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

IO $F(aGbG)$

OI S

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- *IO and OI coincide for non-duplicating grammars.*

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

IO $F(aGbG)$

OI S

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- *IO and OI coincide for non-duplicating grammars.*

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

$IO \quad F(abG)$

$OI \quad S$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

IO $F(abG)$

OI S

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

$IO \quad F(ab)$

$OI \quad S$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

$IO \quad F(ab)$

$OI \quad S$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

$IO \quad ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

$OI \quad S$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

$IO \quad ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

$OI \quad S$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

IO $ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

OI $F(G)$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

$IO \quad ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

$OI \quad F(G)$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

IO $ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

OI $G\#G\#G$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

IO $ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

OI $G\#G\#G$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

IO $ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

OI $G\#aGbG\#G$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

IO $ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

OI $G\#aGbG\#G$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

$IO \quad ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

$OI \quad \#aGbG\#G$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

$IO \quad ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

$OI \quad \#aGbG\#G$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

$IO \quad ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

$OI \quad \#aaGbGbG\#G$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

IO $ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

OI $\#aaGbGbG\#G$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

IO $ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

OI $\#aaGbGb\#G$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ *and* $\mathcal{L}_{OI}$ *are incomparable.*
- $\mathcal{L}_{OI}$ = *Indexed languages.*
- *IO and OI coincide for non-duplicating grammars.*

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

IO $ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

OI $\#aaGbGb\#G$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

$IO \quad ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

$OI \quad \#aabGb\#G$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

$IO \quad ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

$OI \quad \#aabGb\#G$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

$IO \quad ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

$OI \quad \#aabGb\#$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

IO $ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

OI $\#aabGb\#$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

IO $ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

OI $\#aabb\#$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

IO $ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

OI $\#aabb\#$
$L_{OI}(S) = \{w_1\#w_2\#w_3 \mid w \in D_1^*\}$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ and $\mathcal{L}_{OI}$ are incomparable.
- $\mathcal{L}_{OI}$ = Indexed languages.
- IO and OI coincide for non-duplicating grammars.

IO vs. OI

$S \to F(G)$
$F(x) \to x\#x\#x$
$G \to aGbG$
$G \to \epsilon$

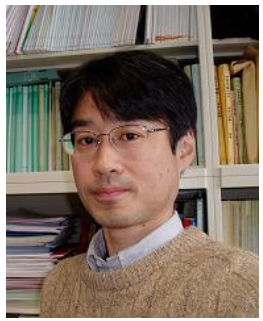
IO $ab\#ab\#ab$
$L_{IO}(S) = \{w\#w\#w \mid w \in D_1^*\}$

OI $\#aabb\#$
$L_{OI}(S) = \{w_1\#w_2\#w_3 \mid w \in D_1^*\}$

Theorem (Fischer 1968)

- $\mathcal{L}_{IO}$ *and* $\mathcal{L}_{OI}$ *are incomparable.*
- $\mathcal{L}_{OI} =$ *Indexed languages.*
- *IO and OI coincide for non-duplicating grammars.*

Non-duplicating macro grammars and $MCFG_{wn}$



- Kato, Seki (2008)

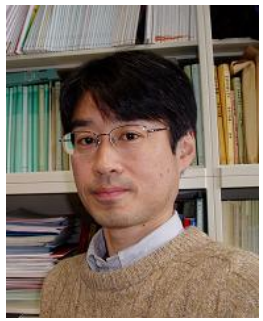
Theorem

$m+1$-$MCFL_{wn}$ = m-non-duplicating macro languages

Corollary

non-duplicating macro languages $\subsetneq$ MCFL

Non-duplicating macro grammars and $MCFG_{wn}$



- Kato, Seki (2008)

Theorem

$m+1$-$MCFL_{wn}$ = m-non-duplicating macro languages

Corollary

non-duplicating macro languages $\subsetneq$ MCFL

Going higher-order: string as λ-terms

Free monoid $\cong$ monoid of uninterpreted functions of type $o \to o$

- $aba \cong \lambda x^o.a(b(a\,x^o))$
- $w_1 \cdot w_2 \cong /w_1/ + /w_2/ = \lambda x^o./w_1/(/w_2/x^o)$
- $\epsilon = \lambda x^o.x^o$

Going higher-order: string as λ-terms

Free monoid $\cong$ monoid of uninterpreted functions of type $o \to o$

- $aba \cong \lambda x^o.a(b(a\,x^o))$
- $w_1 \cdot w_2 \cong /w_1/ + /w_2/ = \lambda x^o./w_1/(/w_2/x^o)$
- $\epsilon = \lambda x^o.x^o$

Going higher-order: string as λ-terms

Free monoid $\cong$ monoid of uninterpreted functions of type $o \to o$

- $aba \cong \lambda x^o.a(b(a\,x^o))$
- $w_1 \cdot w_2 \cong /w_1/ + /w_2/ = \lambda x^o./w_1/(/w_2/x^o)$
- $\epsilon = \lambda x^o.x^o$

Going higher-order: string as λ-terms

Free monoid $\cong$ monoid of uninterpreted functions of type $o \to o$

- $aba \cong \lambda x^o.a(b(a\,x^o))$
- $w_1 \cdot w_2 \cong /w_1/ + /w_2/ = \lambda x^o./w_1/(/w_2/x^o)$
- $\epsilon = \lambda x^o.x^o$

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO S

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO S

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $G(\lambda fx.f(f\,x))A$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $G(\lambda fx.f(f\,x))A$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda hl.G(\lambda f.h(hf))(hl))(\lambda fx.f(f\,x))A$

OI $\lambda x.a(b(b(a(a(b(bx))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda hl.G(\lambda f.h(hf))(hl))(\lambda fx.f(f\,x))A$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda hl.(\lambda hl.hl)(\lambda f.h(hf))(hl))(\lambda fx.f(f\,x))A$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda hl.(\lambda hl.hl)(\lambda f.h(hf))(hl))(\lambda fx.f(f\,x))A$

OI $\lambda x.a(b(b(a(a(b(bx))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda hl.(\lambda hl.hl)(\lambda f.h(hf))(hl))(\lambda fx.f(f\,x))a$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda hl.(\lambda hl.hl)(\lambda f.h(hf))(hl))(\lambda fx.f(f\,x))a$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda l.(\lambda hl.hl)(\lambda f.(\lambda fx.f(f\ x))((\lambda fx.f(f\ x))f))((\lambda fx.f(f\ x))l)a$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda l.(\lambda hl.hl)(\lambda f.(\lambda fx.f(f\,x))((\lambda fx.f(f\,x))f))((\lambda fx.f(f\,x))l)a$

OI $\lambda x.a(b(b(a(a(b(bx))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda hl.hl)(\lambda f.(\lambda fx.f(f\,x))((\lambda fx.f(f\,x))f))((\lambda fx.f(f\,x))a)$

OI $\lambda x.a(b(b(a(a(b(bx))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda hl.hl)(\lambda f.(\lambda fx.f(f\,x))((\lambda fx.f(f\,x))f))((\lambda fx.f(f\,x))a)$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda hl.hl)(\lambda f.(\lambda fx.f(f\,x))((\lambda fx.f(f\,x))f))(\lambda x.a(a\,x))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda hl.hl)(\lambda f.(\lambda fx.f(f\,x))((\lambda fx.f(f\,x))f))(\lambda x.a(a\,x))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda hl.hl)(\lambda f.(\lambda fx.f(f\,x))((\lambda x.f(f\,x))))(\lambda x.a(ax))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda hl.hl)(\lambda f.(\lambda fx.f(f\,x))((\lambda x.f(f\,x))))(\lambda x.a(ax))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda hl.hl)(\lambda fx.(\lambda x.f(f\,x))((\lambda x.f(f\,x))x))(\lambda x.a(ax))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda hl.hl)(\lambda fx.(\lambda x.f(f\,x))((\lambda x.f(f\,x))x))(\lambda x.a(ax))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda hl.hl)(\lambda fx.(\lambda x.f(f\,x))(f(f\,x)))(\lambda x.a(ax))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda hl.hl)(\lambda fx.(\lambda x.f(f\,x))(f(f\,x)))(\lambda x.a(ax))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda hl.hl)(\lambda fx.(f(f(f(f\,x)))))(\lambda x.a(ax))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda hl.hl)(\lambda fx.(f(f(f(f\,x)))))(\lambda x.a(ax))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda l.(\lambda fx.(f(f(f(f\,x)))))l)(\lambda x.a(ax))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda l.(\lambda fx.(f(f(f(f\,x)))))l)(\lambda x.a(ax))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda fx.(f(f(f(f\,x)))))(\lambda x.a(ax))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $(\lambda fx.(f(f(f(f\,x)))))(\lambda x.a(ax))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda x.(\lambda x.a(ax))((\lambda x.a(ax))((\lambda x.a(ax))((\lambda x.a(ax))x)))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda x.(\lambda x.a(ax))((\lambda x.a(ax))((\lambda x.a(ax))((\lambda x.a(ax))x)))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda x.(\lambda x.a(ax))((\lambda x.a(ax))((\lambda x.a(ax))a(ax)))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda x.(\lambda x.a(ax))((\lambda x.a(ax))((\lambda x.a(ax))a(ax)))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda x.(\lambda x.a(ax))((\lambda x.a(ax))(a(a(a(ax)))))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda x.(\lambda x.a(ax))((\lambda x.a(ax))(a(a(a(ax)))))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda x.(\lambda x.a(ax))(a(a(a(a(ax)))))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda x.(\lambda x.a(ax))(a(a(a(a(a(ax))))))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda x.a(a(a(a(a(a(ax))))))$

OI $\lambda x.a(b(b(a(a(a(b(bx)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad \lambda x.a(b(b(a(a(b(bx))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$
$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\}$

$OI \quad S$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\}$

OI $\quad S$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\}$

OI $\quad G(\lambda fx.f(f\,x))A$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\}$

$OI \quad G(\lambda fx.f(f\,x))A$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad (\lambda hl.G(\lambda f.h(hf))(hl)(\lambda fx.f(f\ x))A$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad (\lambda hl.G(\lambda f.h(hf))(hl)(\lambda fx.f(f\ x))A$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad (\lambda I.G(\lambda f.(\lambda fx.f(f\ x))((\lambda fx.f(f\ x))f))((\lambda fx.f(f\ x))I))A$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad (\lambda l.G(\lambda f.(\lambda fx.f(f\ x))((\lambda fx.f(f\ x))f))((\lambda fx.f(f\ x))l))A$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad (\lambda l.G(\lambda f.(\lambda fx.f(f\,x))((\lambda fx.f(f\,x))f))(\lambda x.l(l\,x)))A$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad (\lambda I.G(\lambda f.(\lambda fx.f(f\,x))((\lambda fx.f(f\,x))f))(\lambda x.I(I\,x)))A$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\}$

$OI \quad (\lambda l.G(\lambda f.(\lambda fx.f(f\ x))(\lambda x.f(f\ x)))(\lambda x.l(lx)))A$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

OI $\quad (\lambda l.G(\lambda f.(\lambda fx.f(f\ x))(\lambda x.f(f\ x)))(\lambda x.l(lx)))A$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad (\lambda I.G(\lambda fx.(\lambda x.f(f\ x))((\lambda x.f(f\ x))\ x)))(\lambda x.I(Ix)))A$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad (\lambda l.G(\lambda fx.(\lambda x.f(f\,x))((\lambda x.f(f\,x))\,x)))(\lambda x.l(lx)))A$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad (\lambda l.G(\lambda fx.(\lambda x.f(f\,x))(f(f\,x))))(\lambda x.l(lx)))A$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad (\lambda l.G(\lambda fx.(\lambda x.f(f\ x))(f(f\ x))))(\lambda x.l(lx)))A$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

OI $\quad (\lambda I.G(\lambda fx.f(f(f(f\ x))))(\lambda x.I(Ix)))A$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad (\lambda I.G(\lambda fx.f(f(f(f\,x))))(\lambda x.I(Ix)))A$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

OI $\quad G(\lambda fx.f(f(f(f\,x))))(\lambda x.A(Ax))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

OI $G(\lambda fx.f(f(f(f\ x))))(\lambda x.A(Ax))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$
$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad (\lambda hl.hl)(\lambda fx.f(f(f(f\ x))))(\lambda x.A(Ax))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

OI $\quad (\lambda hl.hl)(\lambda fx.f(f(f(f\,x))))(\lambda x.A(Ax))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

OI $(\lambda l.(\lambda fx.f(f(f(f\ x))))l)(\lambda x.A(Ax))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

OI $(\lambda l.(\lambda fx.f(f(f(f\ x))))l)(\lambda x.A(Ax))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad (\lambda Ix.I(I(I(I\,x))))(\lambda x.A(Ax))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

OI $\quad (\lambda Ix.I(I(I(I\,x))))(\lambda x.A(Ax))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

OI $\lambda x.(\lambda x.A(Ax))((\lambda x.A(Ax))((\lambda x.A(Ax))((\lambda x.A(A)x)\,x)))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad \lambda x.(\lambda x.A(Ax))((\lambda x.A(Ax))((\lambda x.A(Ax))((\lambda x.A(A)x)\,x)))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad \lambda x.(\lambda x.A(Ax))((\lambda x.A(Ax))((\lambda x.A(Ax))(A(Ax))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\,x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad \lambda x.(\lambda x.A(Ax))((\lambda x.A(Ax))((\lambda x.A(Ax))(A(Ax))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad \lambda x.(\lambda x.A(Ax))((\lambda x.A(Ax))(A(A(A(Ax)))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad \lambda x.(\lambda x.A(Ax))((\lambda x.A(Ax))(A(A(A(Ax)))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

OI $\lambda x.(\lambda x.A(Ax))(A(A(A(A(A(Ax))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad \lambda x.(\lambda x.A(Ax))(A(A(A(A(A(Ax))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad \lambda x.A(A(A(A(A(A(Ax))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad \lambda x.A(A(A(A(A(A(Ax))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\}$

$OI \quad \lambda x.a(A(A(A(A(A(A(Ax))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad \lambda x.a(A(A(A(A(A(Ax))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad \lambda x.a(b(A(A(A(A(A(Ax)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad \lambda x.a(b(A(A(A(A(A(Ax)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad \lambda x.a(b(b(A(A(A(A(Ax)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad \lambda x.a(b(b(A(A(A(A(Ax)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\}$

OI $\lambda x.a(b(b(a(A(A(A(Ax)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad \lambda x.a(b(b(a(A(A(A(Ax)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

OI $\lambda x.a(b(b(a(a(A(A(Ax)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

$OI \quad \lambda x.a(b(b(a(a(A(A(Ax)))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\}$

OI $\lambda x.a(b(b(a(a(A(Ax))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\}$

OI $\lambda x.a(b(b(a(a(A(Ax))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}} / \mid n > 0\}$

OI $\lambda x.a(b(b(a(a(b(Ax))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

$IO \quad \lambda x.a(a(a(a(a(a(ax))))))$

$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\}$

$OI \quad \lambda x.a(b(b(a(a(b(Ax))))))$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$S \to G(\lambda fx.f(f\ x))A$
$G \to \lambda hl.G(\lambda f.h(hf))(hl)$
$G \to \lambda hl.hl$
$A \to a$
$A \to b$

IO $\quad \lambda x.a(a(a(a(a(a(ax))))))$

$$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\}$$

OI $\quad \lambda x.a(b(b(a(a(b(bx))))))$

$$L_{OI}(S) = \{/w/ \in \{a; b\}^* \mid |w| = 2^{\frac{n(n+1)}{2}} \wedge n > 0\}$$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

Going higher-order: IO and OI hierarchy

$$S \to G(\lambda fx.f(f\,x))A$$
$$G \to \lambda hl.G(\lambda f.h(hf))(hl)$$
$$G \to \lambda hl.hl$$
$$A \to a$$
$$A \to b$$

IO $\quad \lambda x.a(a(a(a(a(a(ax))))))$

$$L_{IO}(S) = \{/a^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\} \cup \{/b^{2^{\frac{n(n+1)}{2}}}/ \mid n > 0\}$$

OI $\quad \lambda x.a(b(b(a(a(b(bx))))))$

$$L_{OI}(S) = \{/w/ \in \{a; b\}^* \mid |w| = 2^{\frac{n(n+1)}{2}} \wedge n > 0\}$$

Theorem (Damm 1982)

- *the expressive power of IO and OI grammars increase strictly with the order,*
- *IO and OI give rise to incomparable hierarchies,*
- *non-duplicating (non-deleting) IO and OI grammars define the same languages.*

IO, OI and MCFL: tuples of strings as λ-terms

$$(ab, cd)$$

$$\lambda t.t(\underset{/ab/}{\lambda z.a(b\,z)})(\underset{/cd/}{\lambda z.c(d\,z)})$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$Q \xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z))$$
$$R \xrightarrow{*} \lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z)) \end{aligned}$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z)) \end{aligned}$$

$$M[Q \leftarrow \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z))]$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))[Q \leftarrow \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z))]$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))[Q \leftarrow \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z))]$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.\lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z))(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.\lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z))(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))(\lambda z.a(b\,z))(\lambda z.c(d\,z))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))(\lambda z.a(b\,z))(\lambda z.c(d\,z))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.(\lambda x_2.R(\lambda y_1 y_2.t(\lambda z.(\lambda z.a(bz))(y_1\ z))(\lambda z.x_2(y_2\ z))))(\lambda z.c(d\,z))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.(\lambda x_2.R(\lambda y_1 y_2.t(\lambda z.(\lambda z.a(bz))(y_1\ z))(\lambda z.x_2(y_2\ z))))(\lambda z.c(d\,z))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.(\lambda x_2.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.x_2(y_2\ z))))(\lambda z.c(d\,z))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.(\lambda x_2.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.x_2(y_2\ z))))(\lambda z.c(d\,z))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(\lambda z.c(d\,z))(y_2\ z)))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(\lambda z.c(d\,z))(y_2\ z)))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(d\,y_2\ z))))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\ z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(d y_2\ z))))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\ z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(dy_2\ z))))$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(d(y_2\ z)))))[R \leftarrow \lambda t.t(\lambda z.e(f\ z))(\lambda z.g(h\,z))]$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\ z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(d y_2\ z))))$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(d(y_2\ z)))))[R \leftarrow \lambda t.t(\lambda z.e(f\ z))(\lambda z.g(h\,z))]$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\ z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(dy_2\ z))))$$

$$\lambda t.\lambda t.t(\lambda z.e(f\ z))(\lambda z.g(h\,z))(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(d(y_2\ z)))))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(dy_2\ z))))$$

$$\lambda t.\lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z))(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(d(y_2\ z)))))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\ z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(d y_2\ z))))$$

$$\lambda t.(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(d(y_2\ z)))))(\lambda z.e(f\ z))(\lambda z.g(h\,z))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\ z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(d y_2\ z))))$$

$$\lambda t.(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(d(y_2\ z)))))(\lambda z.e(f\ z))(\lambda z.g(h\,z))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\ z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(d y_2\ z))))$$

$$\lambda t.\lambda y_2.t(\lambda z.(a(b((\lambda z.e(f\ z))\ z))))(\lambda z.(c(d(y_2\ z))))(\lambda z.g(h\,z))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\ z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(d y_2\ z))))$$

$$\lambda t.\lambda y_2.t(\lambda z.(a(b((\lambda z.e(f\ z))\ z))))(\lambda z.(c(d(y_2\ z))))(\lambda z.g(h\,z))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\ z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(d y_2\ z))))$$

$$\lambda t.\lambda y_2.t(\lambda z.(a(b(e(f\ z)))))(\lambda z.(c(d(y_2\ z))))(\lambda z.g(h\,z))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\ z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(d y_2\ z))))$$

$$\lambda t.\lambda y_2.t(\lambda z.(a(b(e(f\ z)))))(\lambda z.(c(d(y_2\ z))))(\lambda z.g(h\,z))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\ z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(d y_2\ z))))$$

$$\lambda t.t(\lambda z.(a(b(e(f\ z)))))(\lambda z.(c(d((\lambda z.g(h\,z))\ z))))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(dy_2\ z))))$$

$$\lambda t.t(\lambda z.(a(b(e(f\ z)))))(\lambda z.(c(d((\lambda z.g(h\,z))\ z))))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\ z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(d y_2\ z))))$$

$$\lambda t.t(\lambda z.(a(b(e(f\ z)))))(\lambda z.(c(d(g(h\,z))))))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\xrightarrow{*} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\xrightarrow{*} \lambda t.t(\lambda z.e(f\ z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(d y_2\ z))))$$

$$\lambda t.t(\lambda z.(a(b(e(f\ z)))))(\lambda z.(c(d(g(h\,z))))))$$

Representing MCFG rules as level 3 IO and OI grammar

The rule $P(x_1 y_1, x_2 y_2) \leftarrow Q(x_1, x_2),\ R(y_1, y_2)$:

$$P \rightarrow \underbrace{\lambda t.Q(\lambda x_1 x_2.R(\lambda y_1 y_2.t(\lambda z.x_1(y_1\ z))(\lambda z.x_2(y_2\ z))))}_{M}$$

$$\begin{aligned} Q &\stackrel{*}{\rightarrow} \lambda t.t(\lambda z.a(b\,z))(\lambda z.c(d\,z)) \\ R &\stackrel{*}{\rightarrow} \lambda t.t(\lambda z.e(f\,z))(\lambda z.g(h\,z)) \end{aligned}$$

$$\lambda t.R(\lambda y_1 y_2.t(\lambda z.(a(b(y_1\ z))))(\lambda z.(c(dy_2\ z))))$$

$$\lambda t.t(\lambda z.(a(b(e(f\ z)))))(\lambda z.(c(d(g(h\,z))))))$$

$$\begin{aligned} Q &\approx (/ab/, /cd/) \\ R &\approx (/ef/, /gh/) \\ M[Q \leftarrow (/ab/, /cd/), R \leftarrow (/ef/./gh/)] &\approx (/abef/, /cdgh/) \end{aligned}$$

String languages of λ-CFG

- de Groote, Pogodalla (2004)

$$\text{MCFG} \subseteq \text{non-duplicating IO/OI languages}$$

- Yoshinaka (2007)

non-duplicating IO/OI languages = non-duplicating non-deleting IO/OI languages

String languages of λ-CFG

- de Groote, Pogodalla (2004)

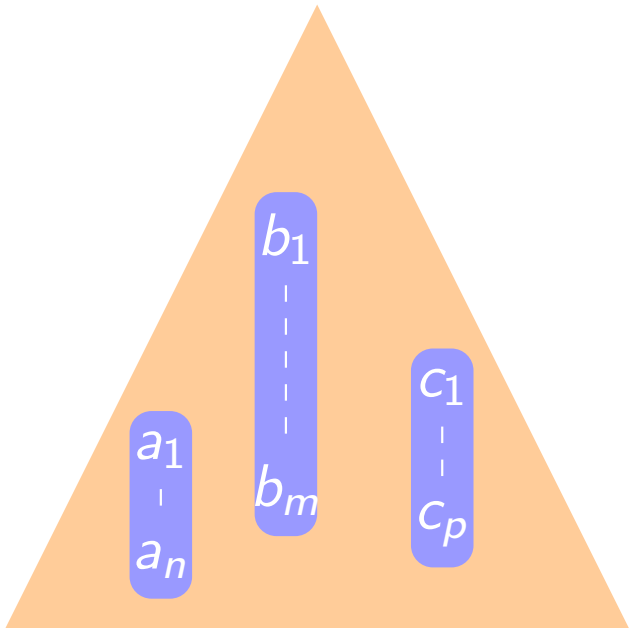
$$\text{MCFG} \subseteq \text{non-duplicating IO/OI languages}$$

- Yoshinaka (2007)

$$\text{non-duplicating IO/OI languages} = \text{non-duplicating non-deleting IO/OI languages}$$

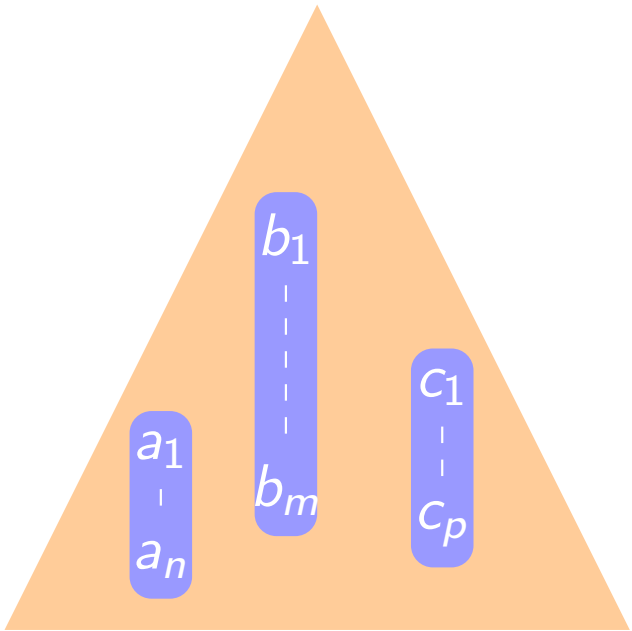
From non-duplicating non-deleting IO/OI grammars to MCFG

$A \rightarrow$



From non-duplicating non-deleting IO/OI grammars to MCFG

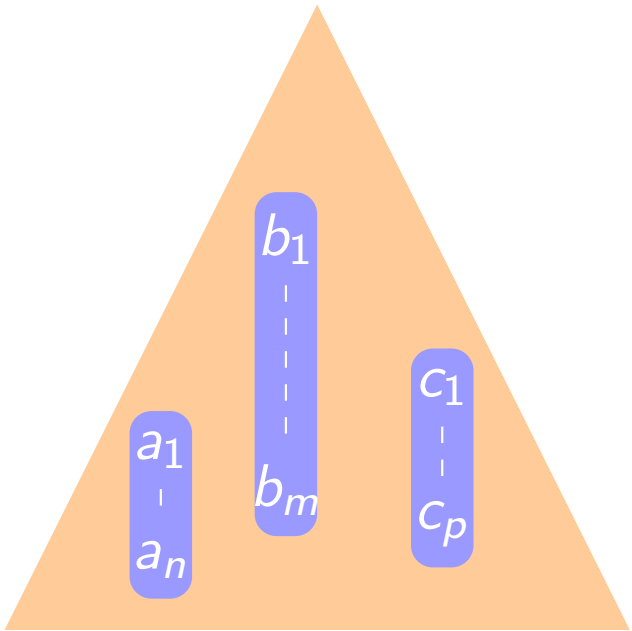
$A \rightarrow$



$((o \rightarrow o) \rightarrow (o \rightarrow o)) \rightarrow (o \rightarrow o) \rightarrow (o \rightarrow o)$

From non-duplicating non-deleting IO/OI grammars to MCFG

$A \rightarrow$

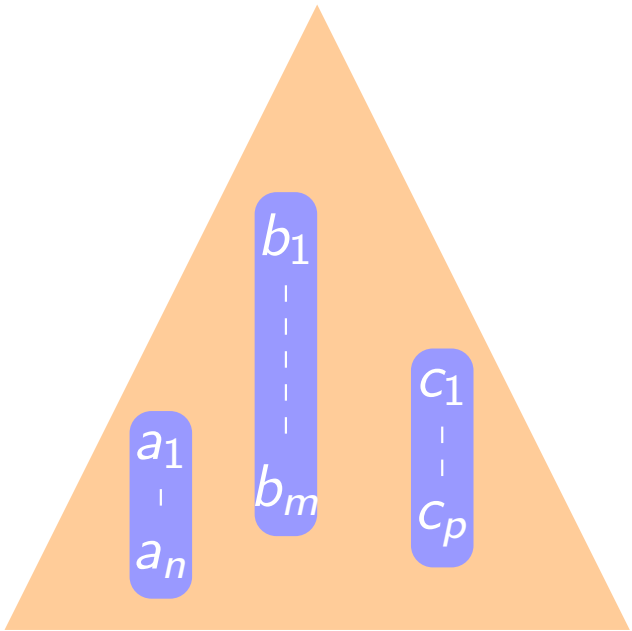


$((o \rightarrow o) \rightarrow (o \rightarrow o)) \rightarrow (o \rightarrow o) \rightarrow (o \rightarrow o)$

$\lambda f\, s\, x.$

From non-duplicating non-deleting IO/OI grammars to MCFG

$A \rightarrow$

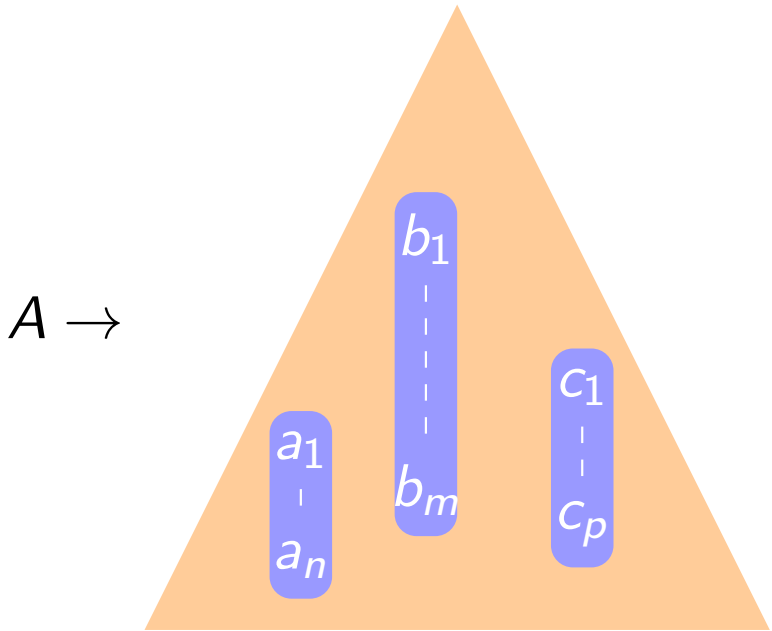


$((o \rightarrow o) \rightarrow (o \rightarrow o)) \rightarrow (o \rightarrow o) \rightarrow (o \rightarrow o)$

$\lambda f\, s\, x.$

f

From non-duplicating non-deleting IO/OI grammars to MCFG



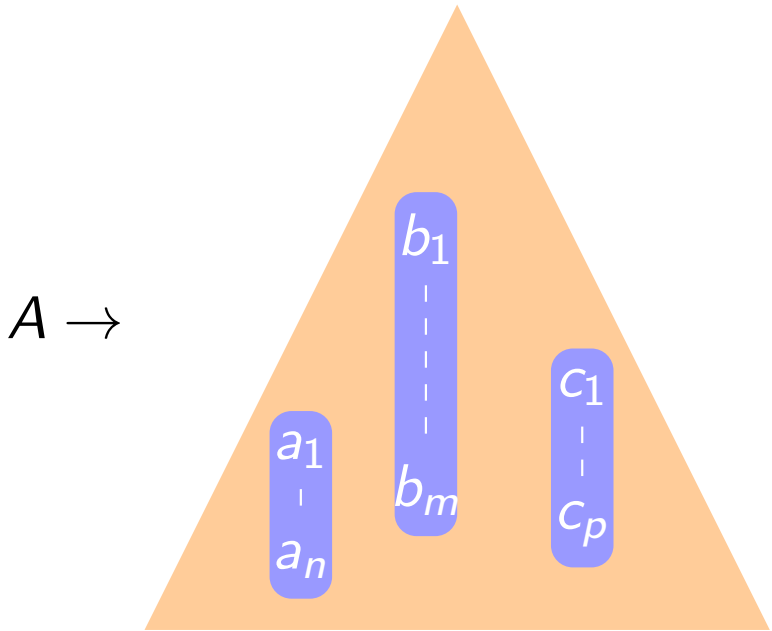
$((o \to o) \to (o \to o)) \to (o \to o) \to (o \to o)$

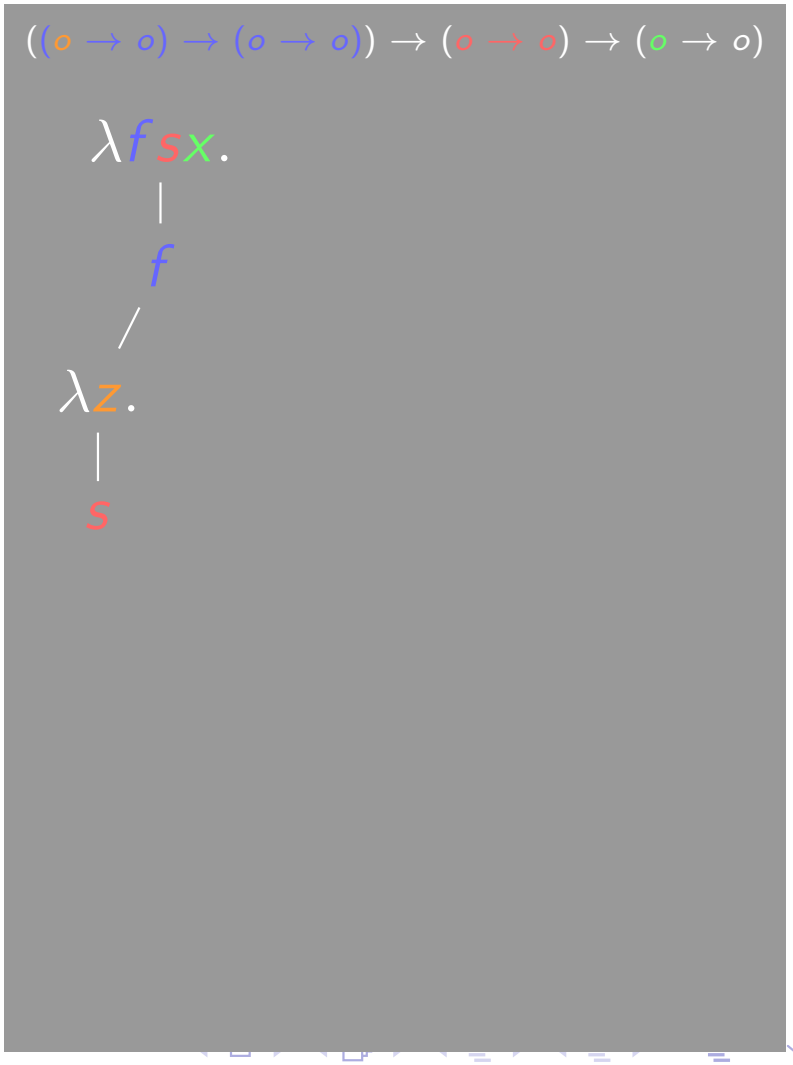
$\lambda f\, s\, x.$

f

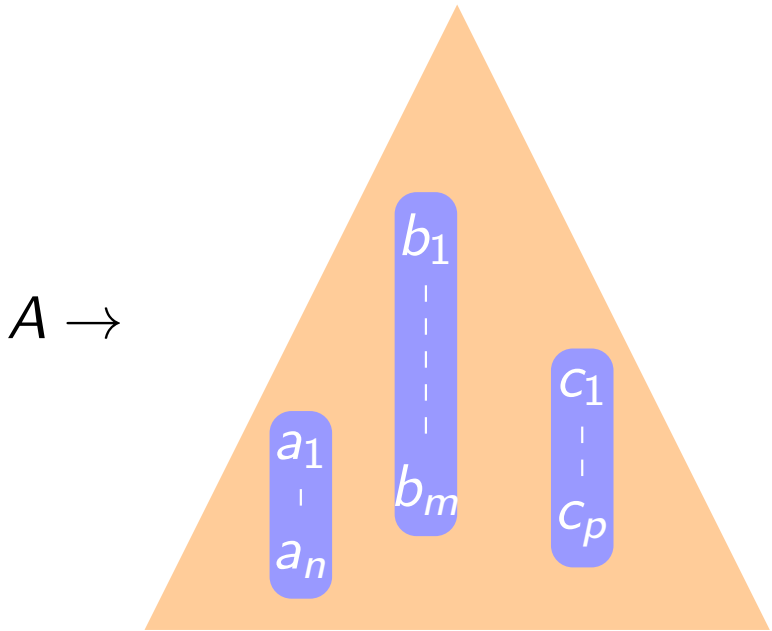
$\lambda z.$

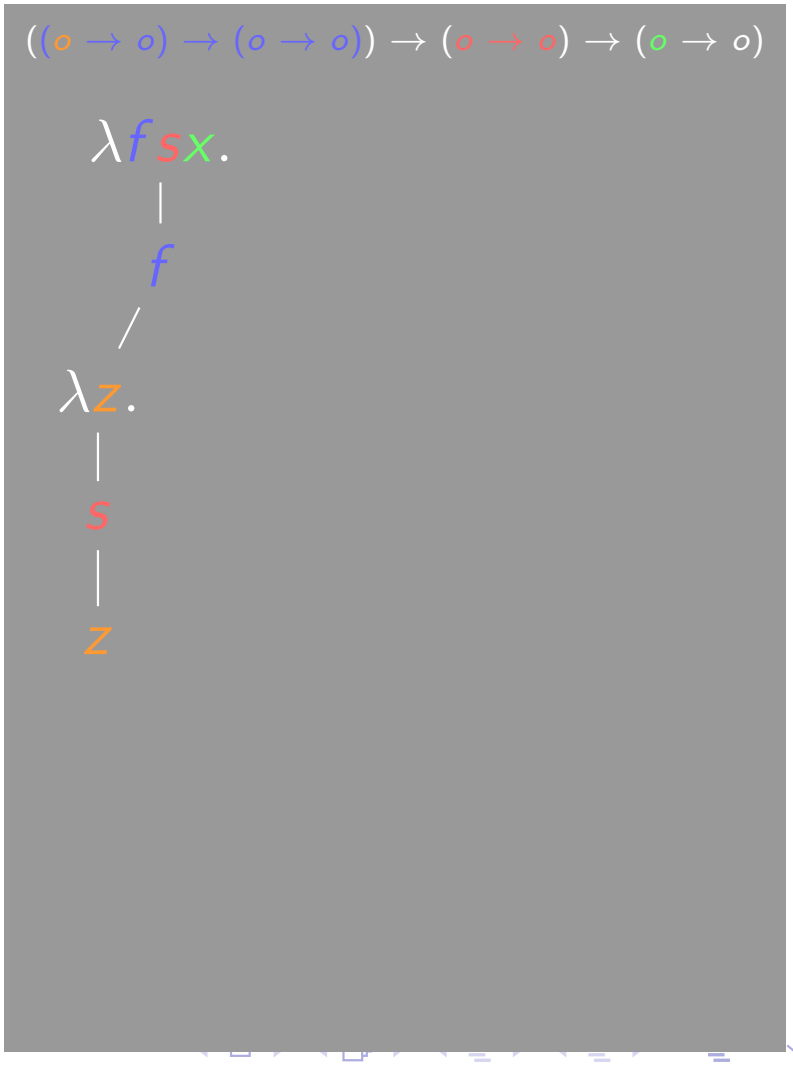
From non-duplicating non-deleting IO/OI grammars to MCFG



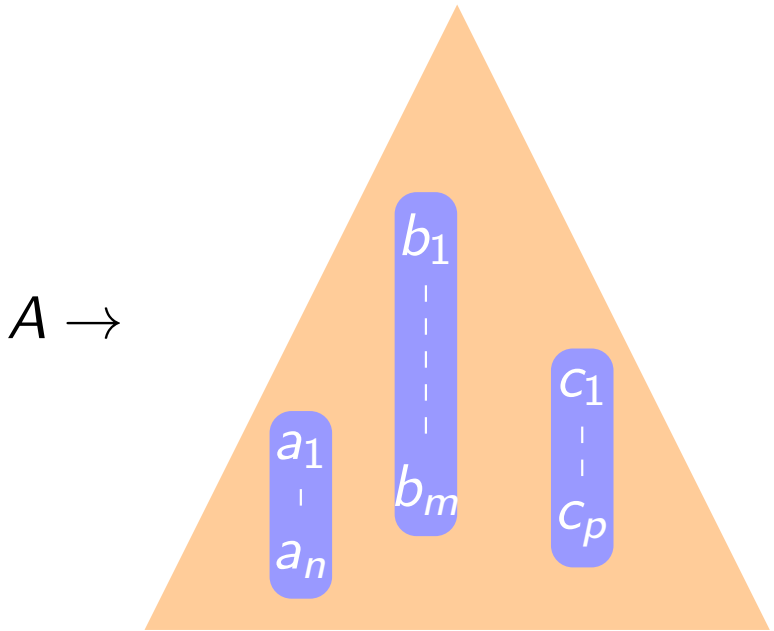


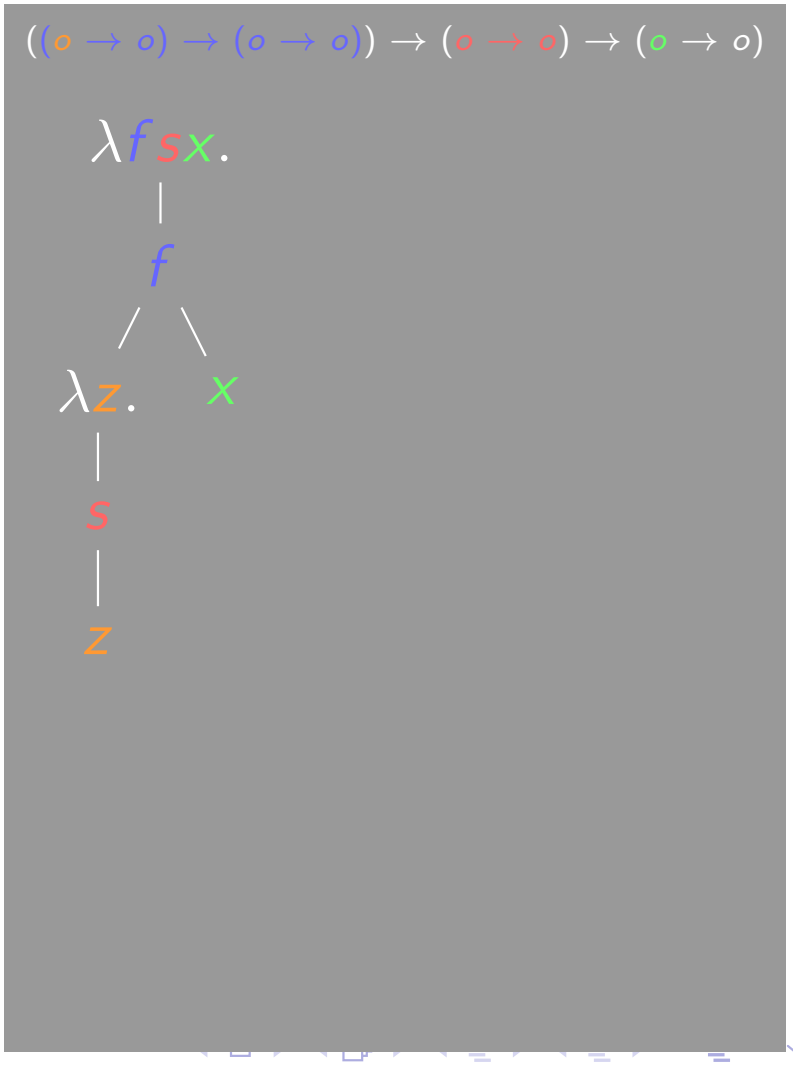
From non-duplicating non-deleting IO/OI grammars to MCFG



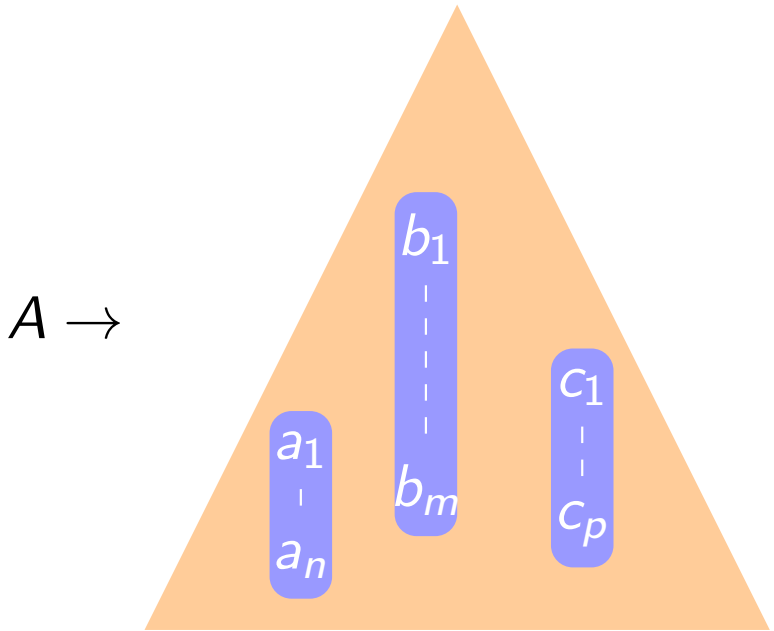


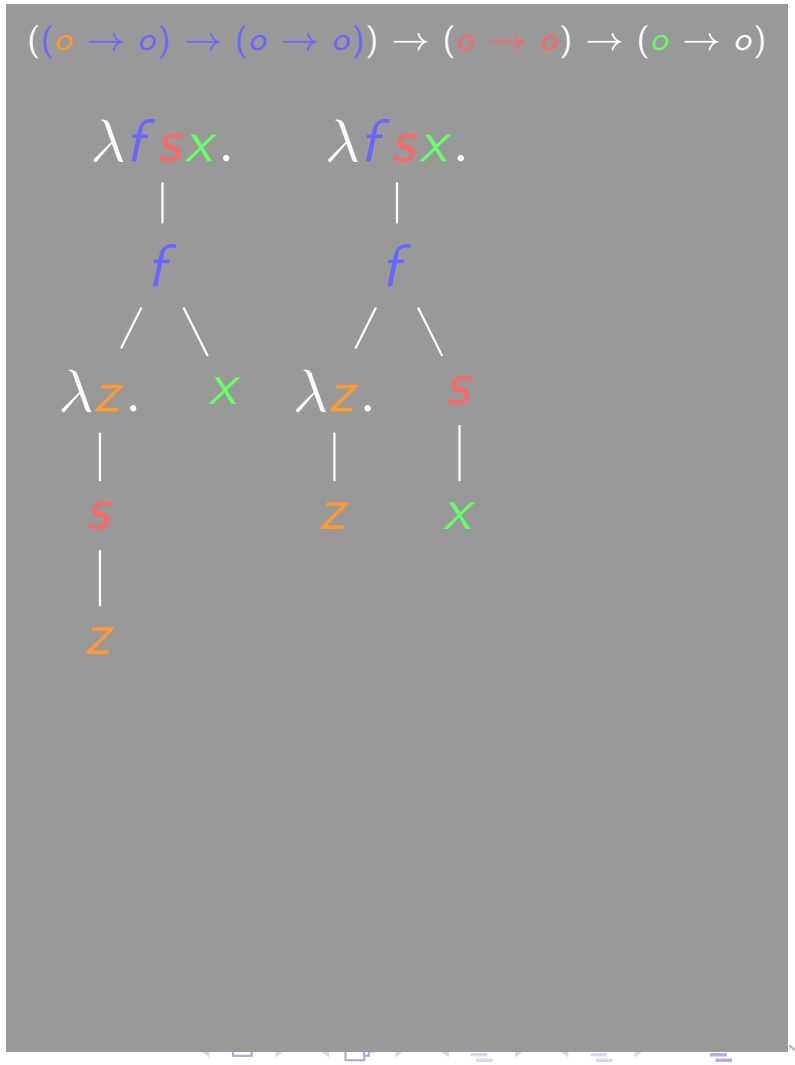
From non-duplicating non-deleting IO/OI grammars to MCFG



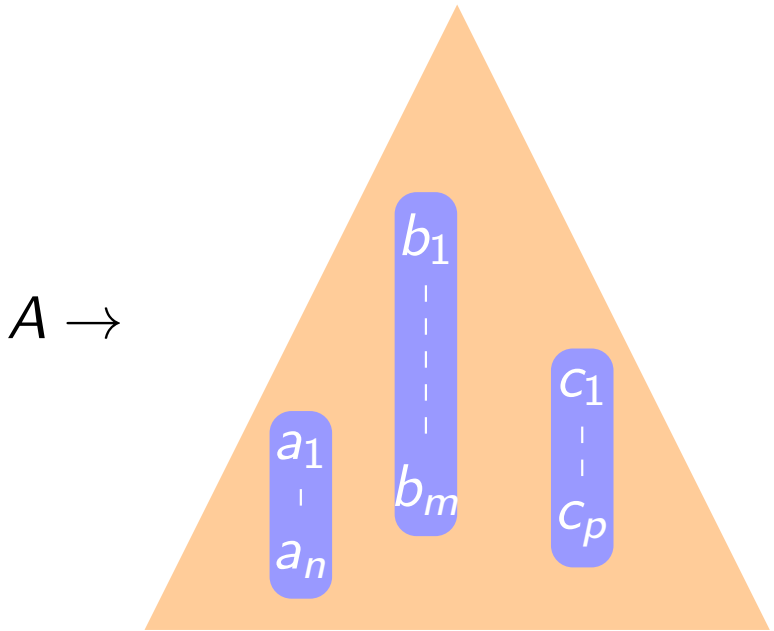


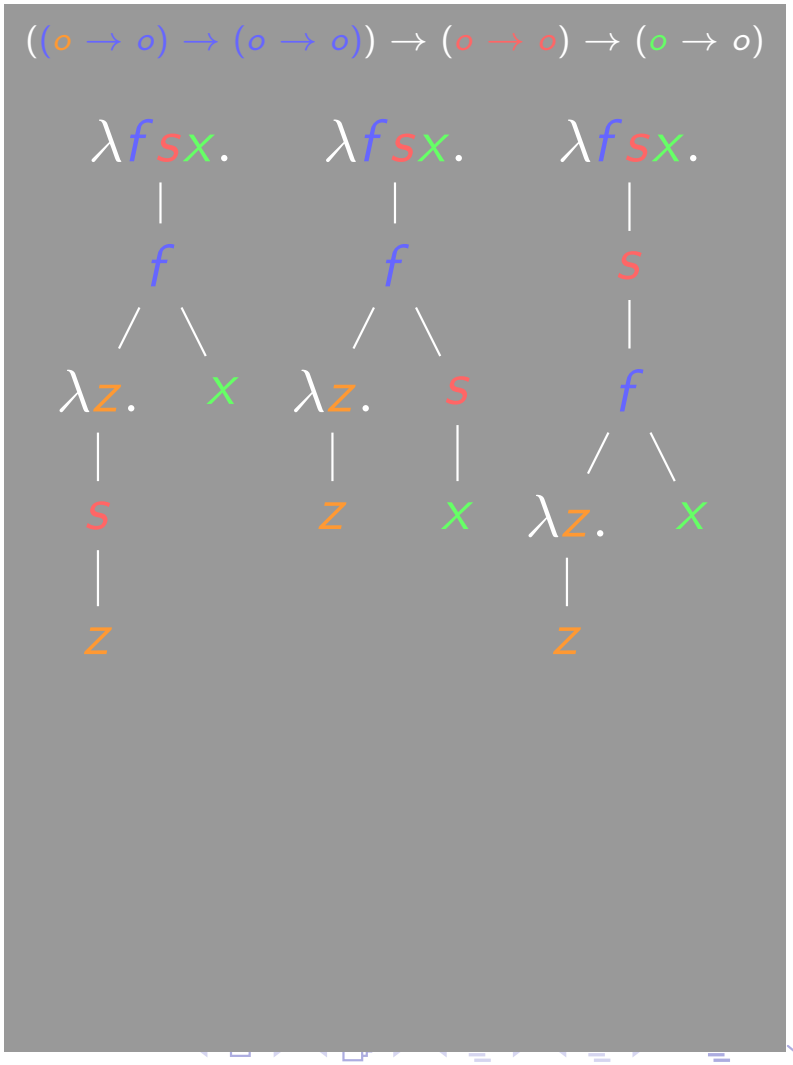
From non-duplicating non-deleting IO/OI grammars to MCFG



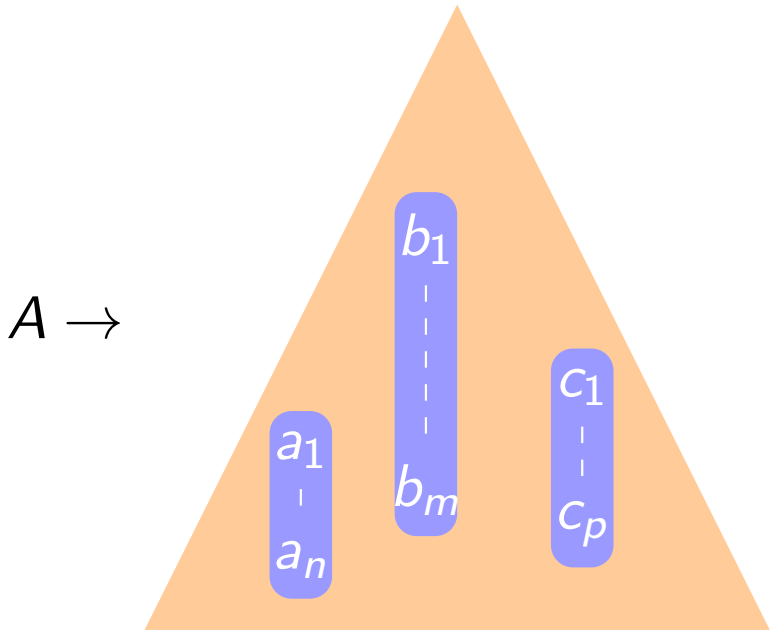


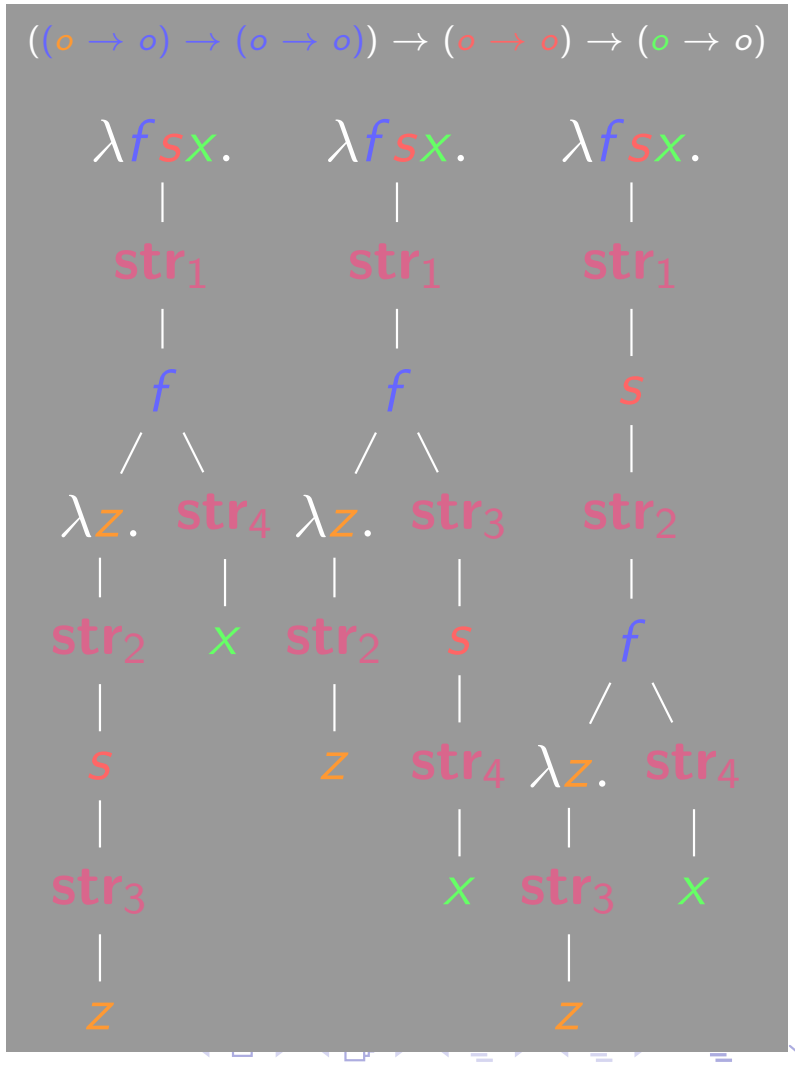
From non-duplicating non-deleting IO/OI grammars to MCFG



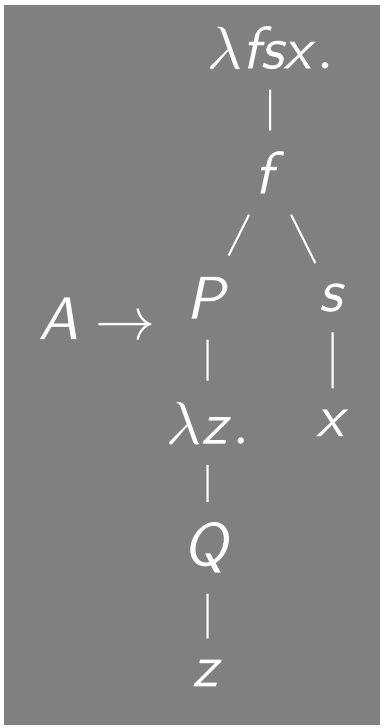


From non-duplicating non-deleting IO/OI grammars to MCFG



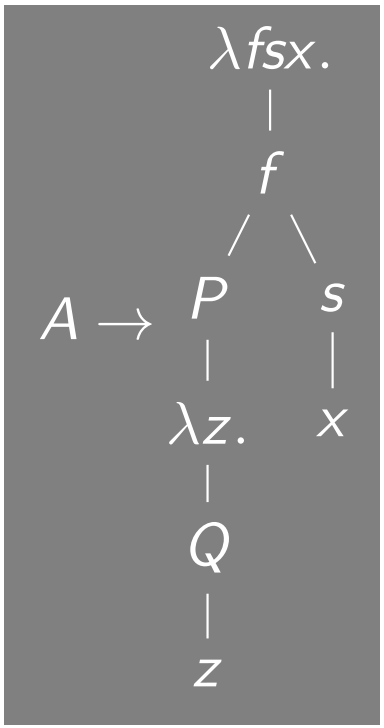


From non-duplicating IO/OI grammars to MCFG



$$
\begin{array}{lcl}
A & : & ((o \to o) \to (o \to o)) \to (o \to o) \to (o \to o) \\
P & : & (o \to o) \to (o \to o) \\
Q & : & (o \to o) \\
A & \to & \lambda fsx.f(P(\lambda z.Qz))(sx)
\end{array}
$$

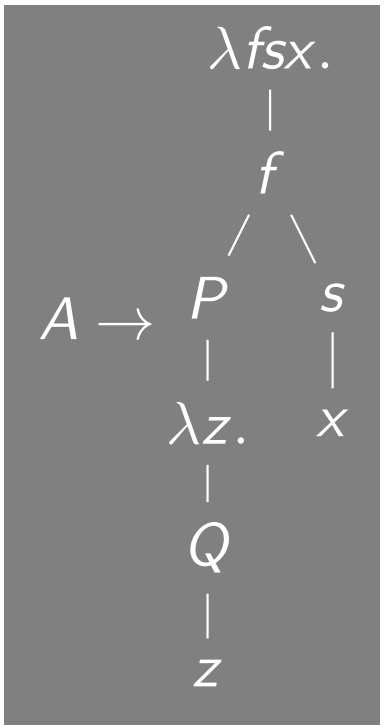
From non-duplicating IO/OI grammars to MCFG



$$
\begin{array}{lcl}
A & : & ((o \to o) \to (o \to o)) \to (o \to o) \to (o \to o) \\
P & : & (o \to o) \to (o \to o) \\
Q & : & (o \to o) \\
A & \to & \lambda fsx.f(P(\lambda z.Qz))(sx)
\end{array}
$$

$\langle A, N_1\rangle(\ ,\qquad ,\ ,\ ) \leftarrow \langle P, N_2\rangle(x_1, x_2)\,\langle Q, N_3\rangle(y)$

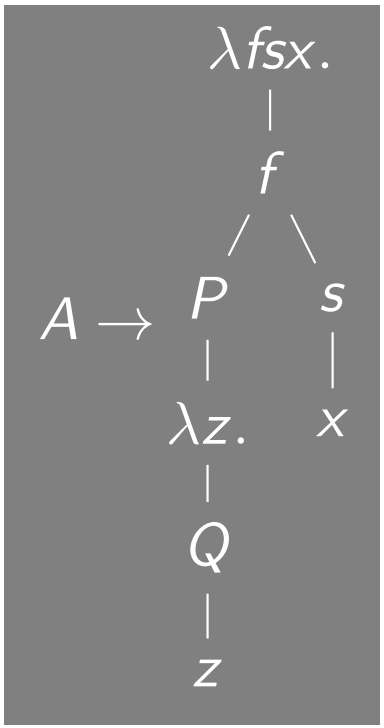
From non-duplicating IO/OI grammars to MCFG



$$
\begin{array}{lcl}
A & : & ((o \to o) \to (o \to o)) \to (o \to o) \to (o \to o) \\
P & : & (o \to o) \to (o \to o) \\
Q & : & (o \to o) \\
A & \to & \lambda fsx.f(P(\lambda z.Qz))(sx)
\end{array}
$$

$\langle A, N_1\rangle(\ ,\qquad ,\ ,\ ) \leftarrow \langle P, N_2\rangle(x_1, x_2)\,\langle Q, N_3\rangle(y)$
$N_2 = \lambda gz.\mathbf{x_1}(g(\mathbf{x_2}z))$

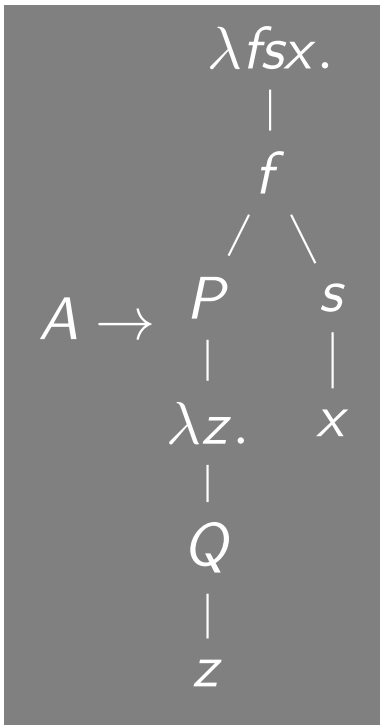
From non-duplicating IO/OI grammars to MCFG



$$
\begin{array}{lcl}
A & : & ((o \to o) \to (o \to o)) \to (o \to o) \to (o \to o) \\
P & : & (o \to o) \to (o \to o) \\
Q & : & (o \to o) \\
A & \to & \lambda fsx.f(P(\lambda z.Qz))(sx)
\end{array}
$$

$\langle A, N_1\rangle(\ ,\qquad ,\ ,\ ) \leftarrow \langle P, N_2\rangle(x_1, x_2)\,\langle Q, N_3\rangle(y)$
$N_2 = \lambda gz.\mathbf{x_1}(g(\mathbf{x_2}z))$
$N_3 = \lambda z.\mathbf{y}z$

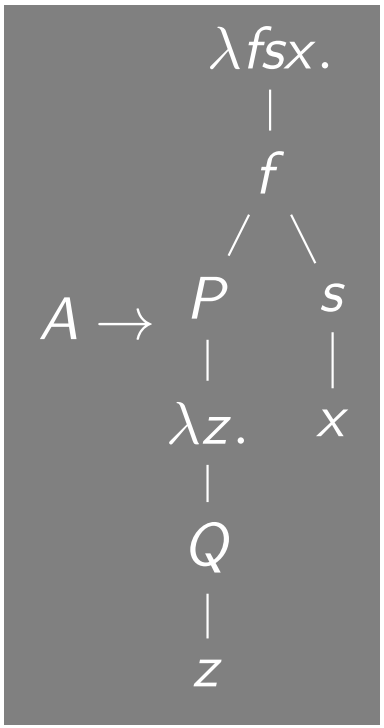
From non-duplicating IO/OI grammars to MCFG



$$
\begin{array}{lcl}
A & : & ((o \to o) \to (o \to o)) \to (o \to o) \to (o \to o) \\
P & : & (o \to o) \to (o \to o) \\
Q & : & (o \to o) \\
A & \to & \lambda fsx.f(P(\lambda z.Qz))(sx)
\end{array}
$$

$$
\begin{array}{l}
\langle A, N_1 \rangle(\ ,\qquad ,\ ,\ ) \leftarrow \langle P, N_2 \rangle(x_1, x_2)\,\langle Q, N_3 \rangle(y) \\
N_2 = \lambda gz.\mathbf{x_1}(g(\mathbf{x_2}z)) \\
N_3 = \lambda z.\mathbf{y}z \\
\lambda fsx.f(P(\lambda z.Qz))(sx)[P \leftarrow N_2, Q \leftarrow N_3]
\end{array}
$$

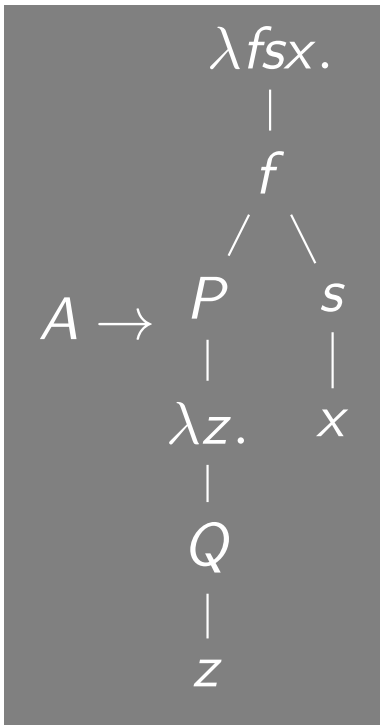
From non-duplicating IO/OI grammars to MCFG



$$
\begin{array}{lcl}
A & : & ((o \to o) \to (o \to o)) \to (o \to o) \to (o \to o) \\
P & : & (o \to o) \to (o \to o) \\
Q & : & (o \to o) \\
A & \to & \lambda fsx.f(P(\lambda z.Qz))(sx)
\end{array}
$$

$$
\begin{array}{l}
\langle A, N_1\rangle(\ ,\quad\ ,\ ,\ ) \leftarrow \langle P, N_2\rangle(x_1, x_2)\,\langle Q, N_3\rangle(y) \\
N_2 = \lambda gz.\mathbf{x_1}(g(\mathbf{x_2}z)) \\
N_3 = \lambda z.\mathbf{y}z \\
\lambda fsx.f(P(\lambda z.Qz))(sx)[P \leftarrow N_2, Q \leftarrow N_3] \\
\quad =_\beta \lambda fsx.f(\lambda z.\mathbf{x_1}(\mathbf{y}(\mathbf{x_2}z)))(sx)
\end{array}
$$

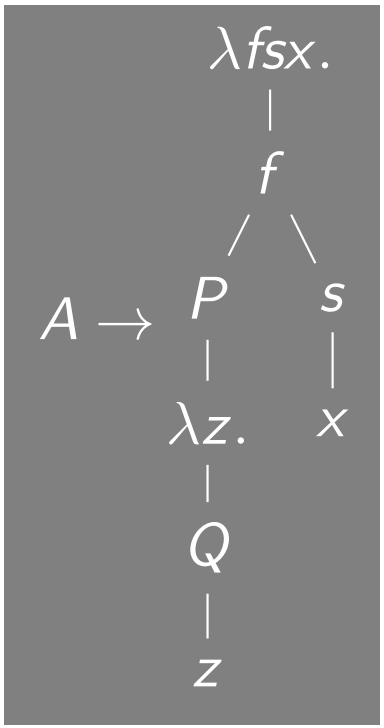
From non-duplicating IO/OI grammars to MCFG



$$
\begin{array}{lcl}
A & : & ((o \to o) \to (o \to o)) \to (o \to o) \to (o \to o) \\
P & : & (o \to o) \to (o \to o) \\
Q & : & (o \to o) \\
A & \to & \lambda fsx.f(P(\lambda z.Qz))(sx)
\end{array}
$$

$$
\begin{array}{l}
\langle A, N_1\rangle(\ ,\quad\ ,\ ,\ ) \leftarrow \langle P, N_2\rangle(x_1, x_2)\,\langle Q, N_3\rangle(y) \\
N_2 = \lambda gz.\mathbf{x_1}(g(\mathbf{x_2}z)) \\
N_3 = \lambda z.\mathbf{y}z \\
\lambda fsx.f(P(\lambda z.Qz))(sx)[P \leftarrow N_2, Q \leftarrow N_3] \\
\quad =_\beta \lambda fsx.f(\lambda z.\mathbf{x_1}(\mathbf{y}(\mathbf{x_2}z)))(sx) \\
N_1 = \lambda fsx.\mathbf{z_1}f(\lambda z.\mathbf{z_2}z)(\mathbf{z_3}(s(\mathbf{z_4}x)))
\end{array}
$$

From non-duplicating IO/OI grammars to MCFG



$$
\begin{array}{lcl}
A & : & ((o \to o) \to (o \to o)) \to (o \to o) \to (o \to o) \\
P & : & (o \to o) \to (o \to o) \\
Q & : & (o \to o) \\
A & \to & \lambda fsx.f(P(\lambda z.Qz))(sx)
\end{array}
$$

$$\langle A, N_1\rangle(\epsilon, \quad , \ , \ ) \leftarrow \langle P, N_2\rangle(x_1, x_2)\,\langle Q, N_3\rangle(y)$$

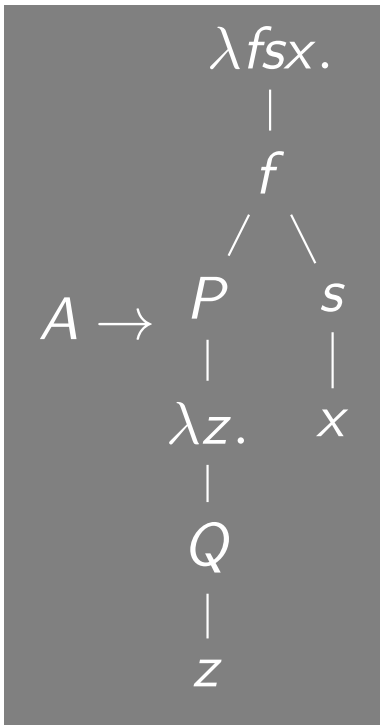
$$N_2 = \lambda gz.\mathbf{x_1}(g(\mathbf{x_2}z))$$

$$N_3 = \lambda z.\mathbf{y}z$$

$$\lambda fsx.f(P(\lambda z.Qz))(sx)[P \leftarrow N_2, Q \leftarrow N_3]$$
$$=_\beta \lambda fsx.f(\lambda z.\mathbf{x_1}(\mathbf{y}(\mathbf{x_2}z)))(sx)$$

$$N_1 = \lambda fsx.\mathbf{z_1}f(\lambda z.\mathbf{z_2}z)(\mathbf{z_3}(s(\mathbf{z_4}x)))$$

From non-duplicating IO/OI grammars to MCFG



$$
\begin{array}{lcl}
A & : & ((o \to o) \to (o \to o)) \to (o \to o) \to (o \to o) \\
P & : & (o \to o) \to (o \to o) \\
Q & : & (o \to o) \\
A & \to & \lambda fsx.f(P(\lambda z.Qz))(sx)
\end{array}
$$

$$\langle A, N_1\rangle(\epsilon, \mathbf{x_1}\mathbf{y}\mathbf{x_2}, \ , \ ) \leftarrow \langle P, N_2\rangle(x_1, x_2)\,\langle Q, N_3\rangle(y)$$

$$N_2 = \lambda gz.\mathbf{x_1}(g(\mathbf{x_2}z))$$

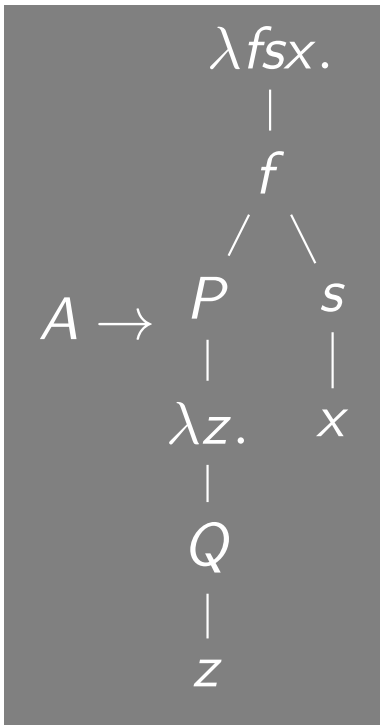
$$N_3 = \lambda z.\mathbf{y}z$$

$$\lambda fsx.f(P(\lambda z.Qz))(sx)[P \leftarrow N_2, Q \leftarrow N_3]$$

$$=_\beta \lambda fsx.f(\lambda z.\mathbf{x_1}(\mathbf{y}(\mathbf{x_2}z)))(sx)$$

$$N_1 = \lambda fsx.\mathbf{z_1}f(\lambda z.\mathbf{z_2}z)(\mathbf{z_3}(s(\mathbf{z_4}x)))$$

From non-duplicating IO/OI grammars to MCFG



$$
\begin{array}{lcl}
A & : & ((o \to o) \to (o \to o)) \to (o \to o) \to (o \to o) \\
P & : & (o \to o) \to (o \to o) \\
Q & : & (o \to o) \\
A & \to & \lambda fsx.f(P(\lambda z.Qz))(sx)
\end{array}
$$

$$\langle A, N_1\rangle(\epsilon, \mathbf{x_1}\mathbf{y}\mathbf{x_2}, \epsilon, \ ) \leftarrow \langle P, N_2\rangle(x_1, x_2)\,\langle Q, N_3\rangle(y)$$

$$N_2 = \lambda gz.\mathbf{x_1}(g(\mathbf{x_2}z))$$

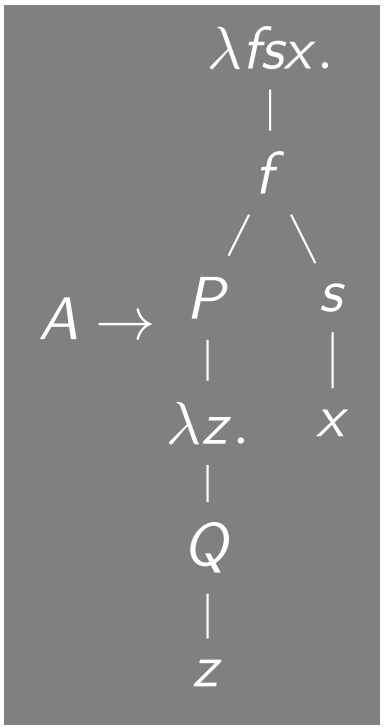
$$N_3 = \lambda z.\mathbf{y}z$$

$$\lambda fsx.f(P(\lambda z.Qz))(sx)[P \leftarrow N_2, Q \leftarrow N_3]$$

$$=_\beta \lambda fsx.f(\lambda z.\mathbf{x_1}(\mathbf{y}(\mathbf{x_2}z)))(sx)$$

$$N_1 = \lambda fsx.\mathbf{z_1}f(\lambda z.\mathbf{z_2}z)(\mathbf{z_3}(s(\mathbf{z_4}x)))$$

From non-duplicating IO/OI grammars to MCFG



$$
\begin{array}{lcl}
A & : & ((o \to o) \to (o \to o)) \to (o \to o) \to (o \to o) \\
P & : & (o \to o) \to (o \to o) \\
Q & : & (o \to o) \\
A & \to & \lambda fsx.f(P(\lambda z.Qz))(sx)
\end{array}
$$

$$\langle A, N_1\rangle(\epsilon, \mathbf{x_1}\mathbf{y}\mathbf{x_2}, \epsilon, \epsilon) \leftarrow \langle P, N_2\rangle(x_1, x_2)\,\langle Q, N_3\rangle(y)$$

$$N_2 = \lambda gz.\mathbf{x_1}(g(\mathbf{x_2}z))$$

$$N_3 = \lambda z.\mathbf{y}z$$

$$\lambda fsx.f(P(\lambda z.Qz))(sx)[P \leftarrow N_2, Q \leftarrow N_3]$$

$$=_\beta \lambda fsx.f(\lambda z.\mathbf{x_1}(\mathbf{y}(\mathbf{x_2}z)))(sx)$$

$$N_1 = \lambda fsx.\mathbf{z_1}f(\lambda z.\mathbf{z_2}z)(\mathbf{z_3}(s(\mathbf{z_4}x)))$$

The equivalence

- S. (2007)

$$\text{MCFL} = \text{non-duplicating IO/OI languages}$$

- Kanazawa (2010)

$$\text{tree(HR)} = \text{non-duplicating IO/OI languages}$$

The equivalence

- S. (2007)

$$\text{MCFL} = \text{non-duplicating IO/OI languages}$$

- Kanazawa (2010)

$$\text{tree(HR)} = \text{non-duplicating IO/OI languages}$$

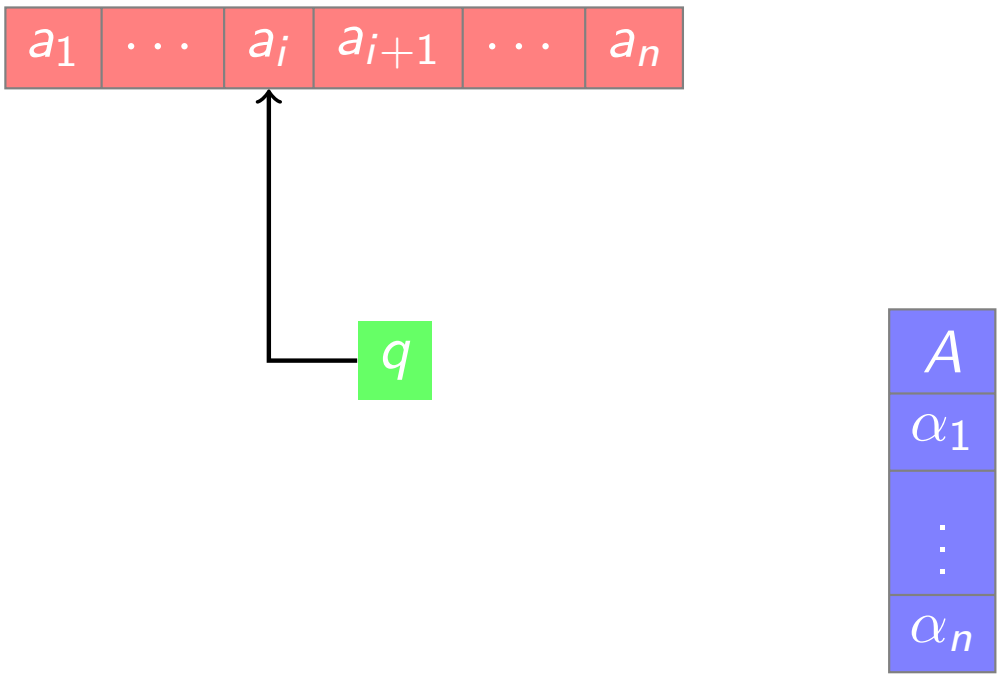
The ingredients for being inside MCFL

- context-freeness
- non-copyings operations
- operation derived from the free monoid

Outline

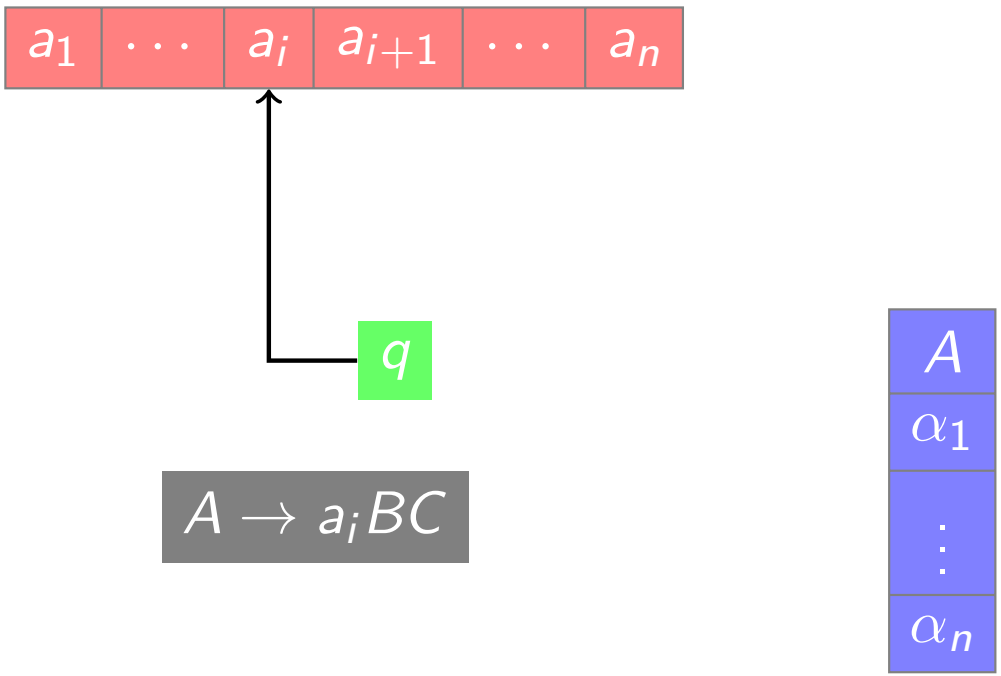
Pushdown automata

CFL = languages recognized by pushdown automata



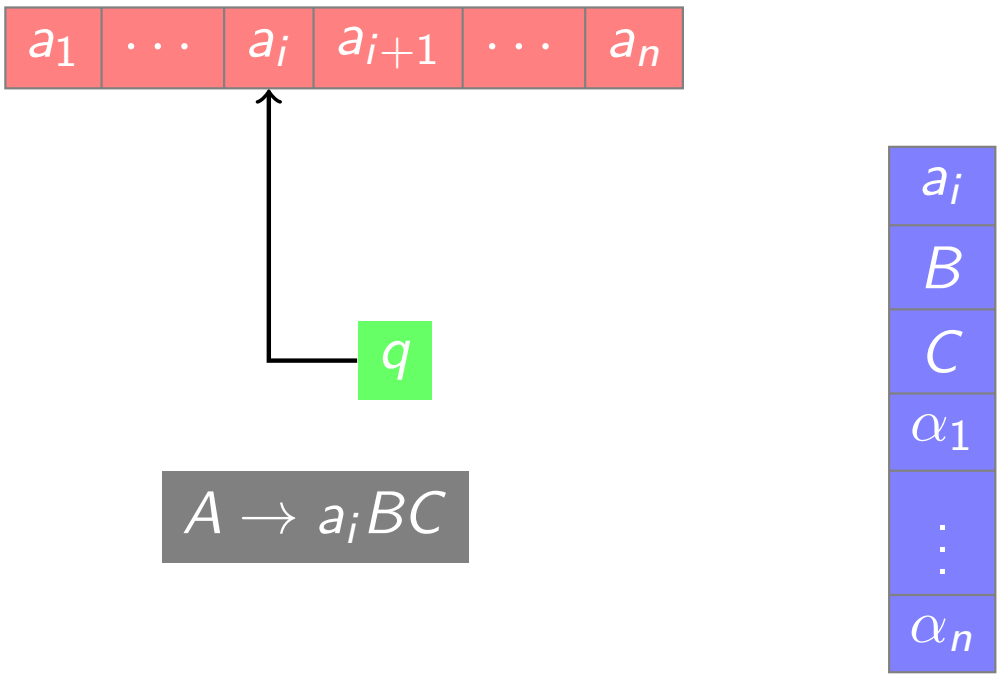
Pushdown automata

CFL = languages recognized by pushdown automata



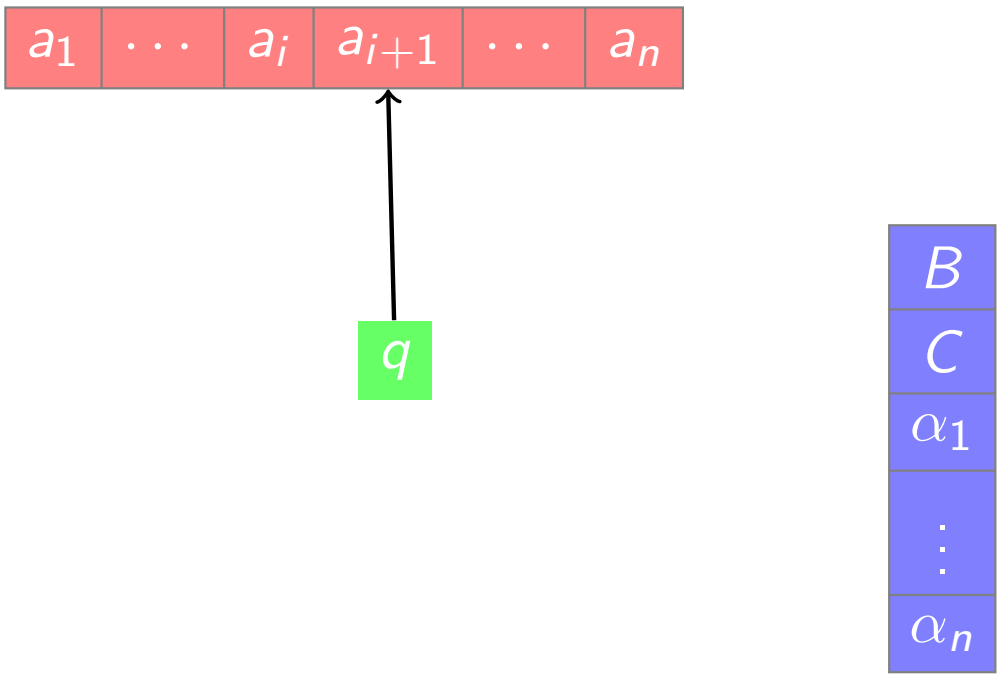
Pushdown automata

CFL = languages recognized by pushdown automata



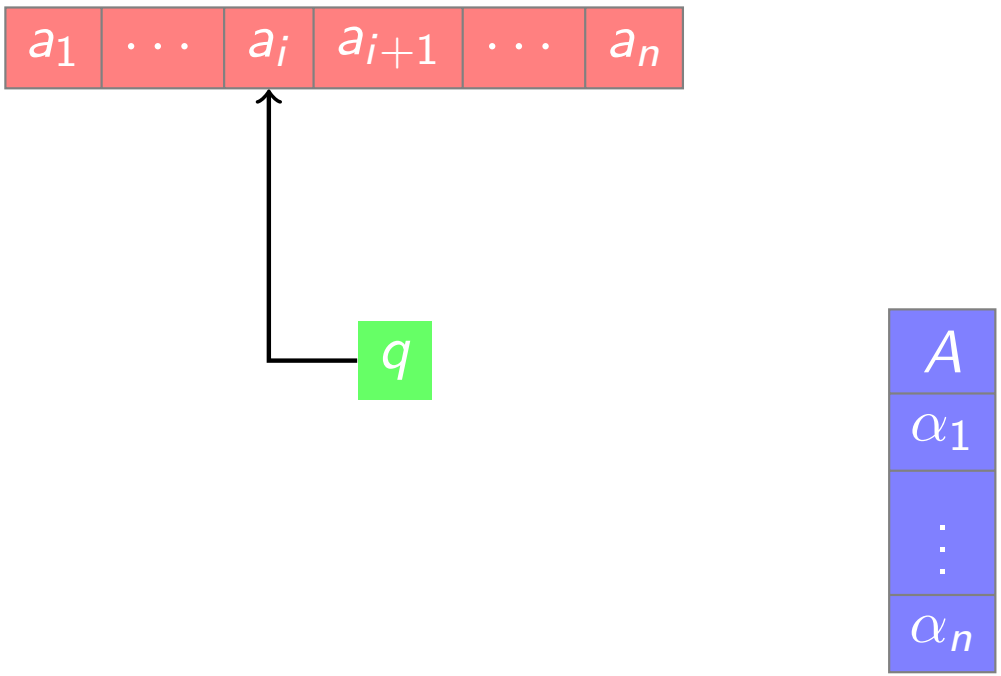
Pushdown automata

CFL = languages recognized by pushdown automata



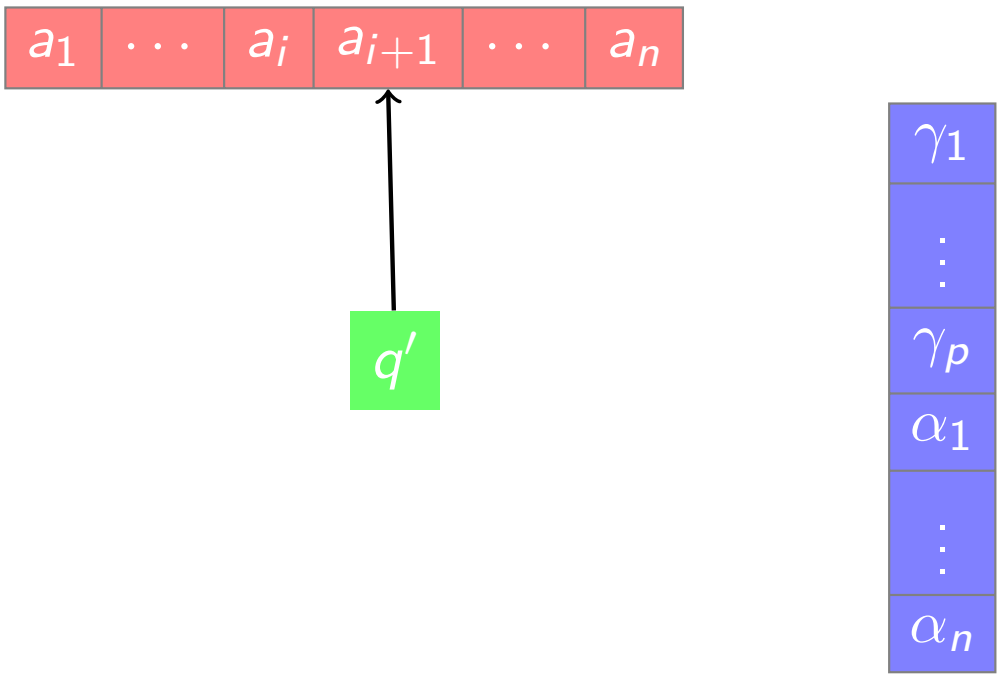
Pushdown automata

CFL = languages recognized by pushdown automata



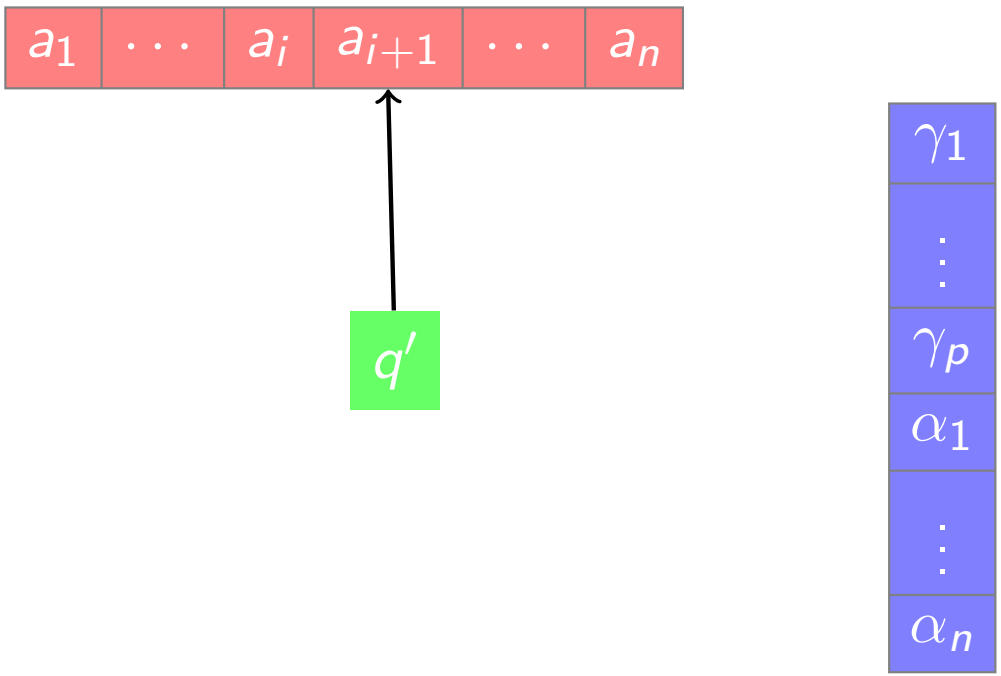
Pushdown automata

CFL = languages recognized by pushdown automata



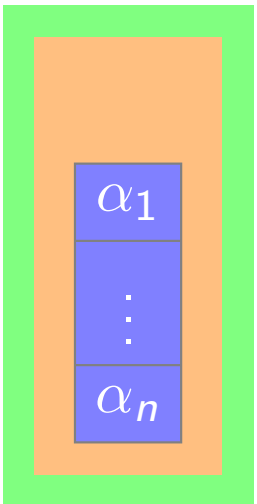
Pushdown automata

CFL = languages recognized by pushdown automata



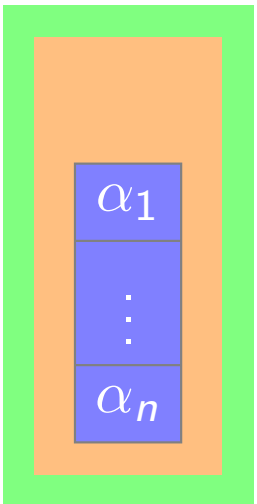
$$A \to a_i(q, \gamma_1, q_1) \dots (q_{i-1}, \gamma_i, q_i) \dots (q_{p-1}, \gamma_p, q')$$

Higher-order pushdown automata



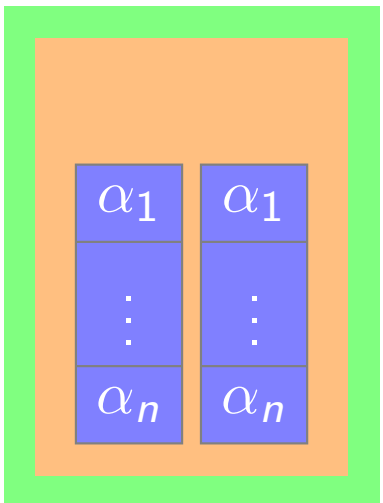
Higher-order pushdown automata

$copy_1$



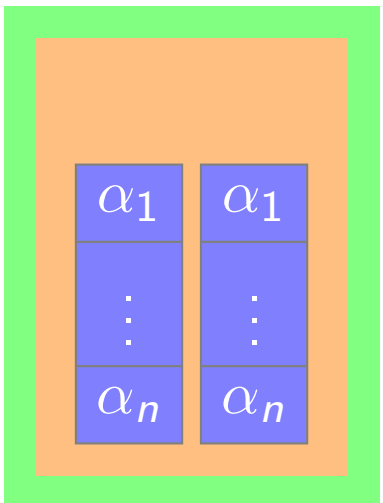
Higher-order pushdown automata

$copy_1$



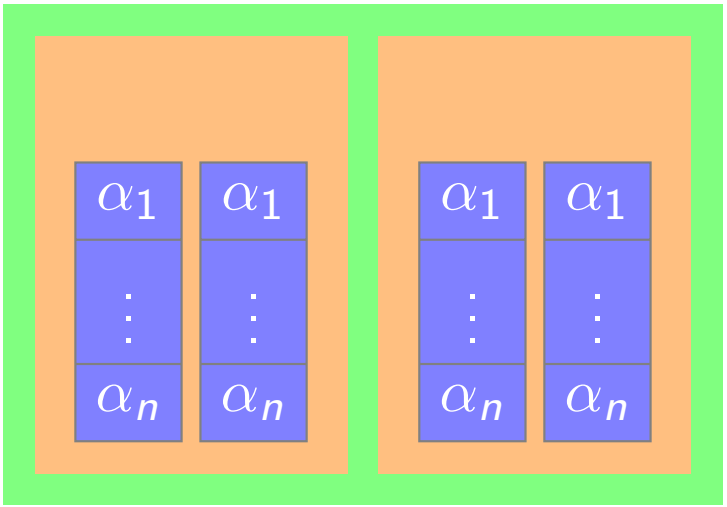
Higher-order pushdown automata

$copy_2$



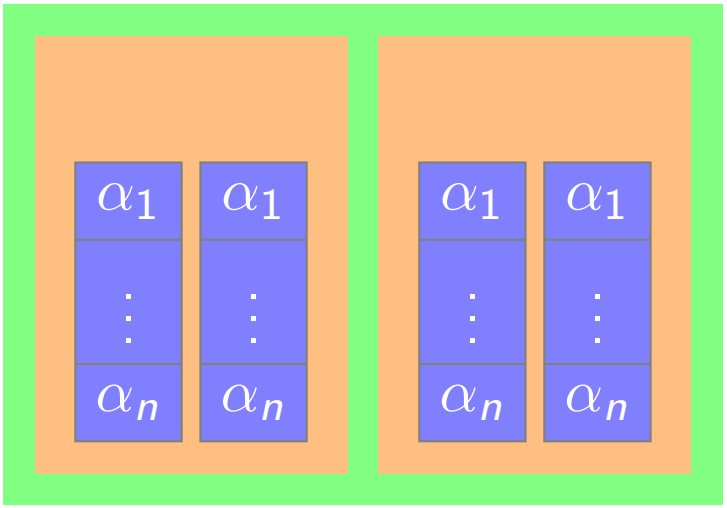
Higher-order pushdown automata

$copy_2$



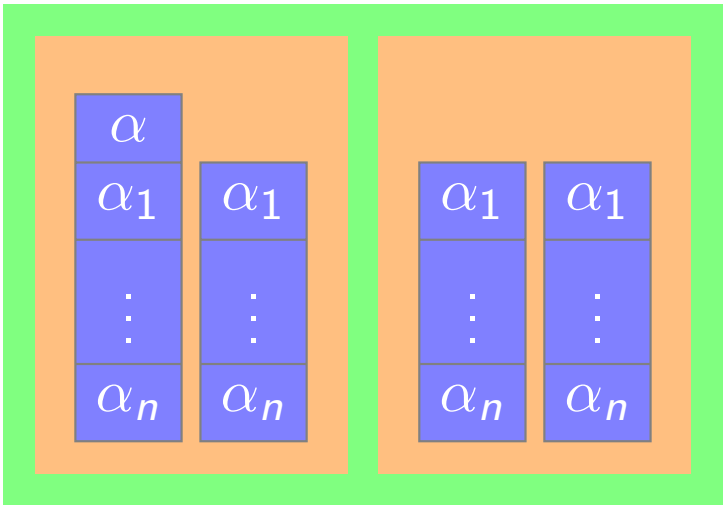
Higher-order pushdown automata

$push(\alpha)$



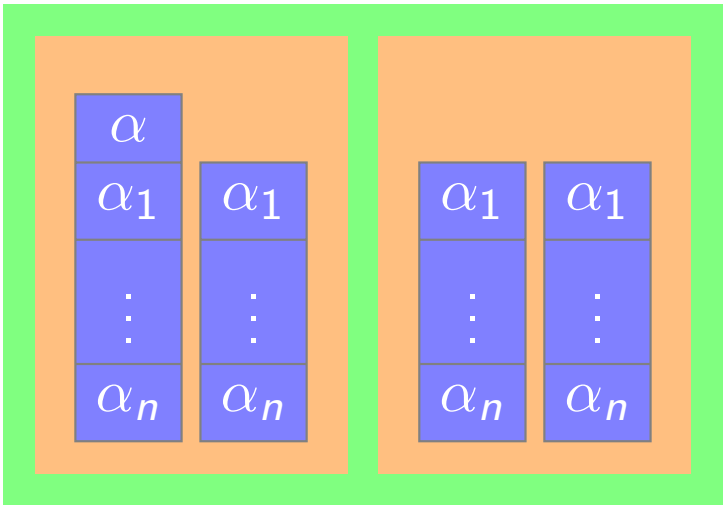
Higher-order pushdown automata

$push(\alpha)$



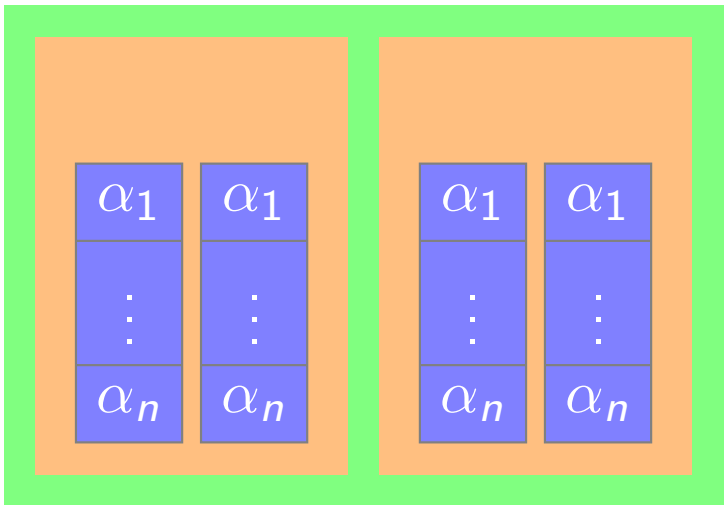
Higher-order pushdown automata

pop_0



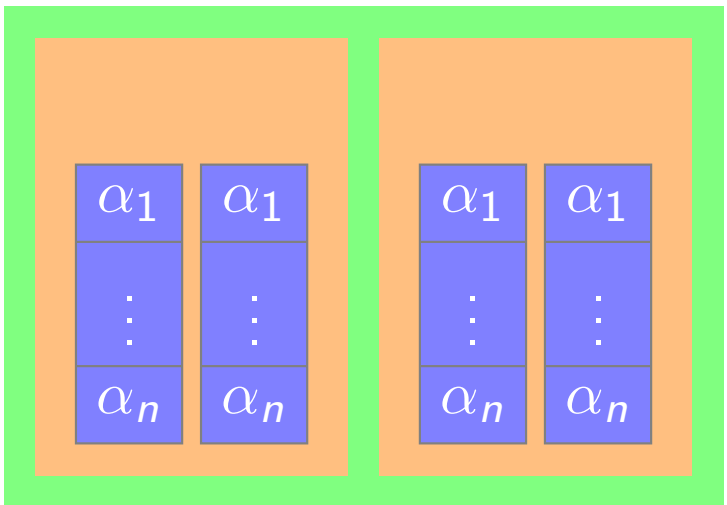
Higher-order pushdown automata

pop_0



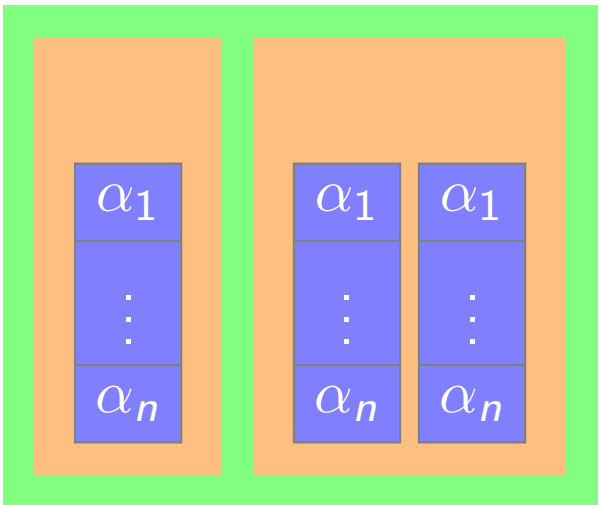
Higher-order pushdown automata

pop_1



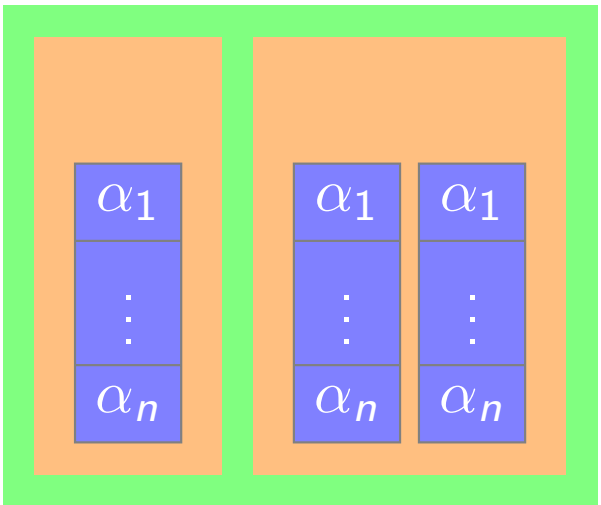
Higher-order pushdown automata

pop_1

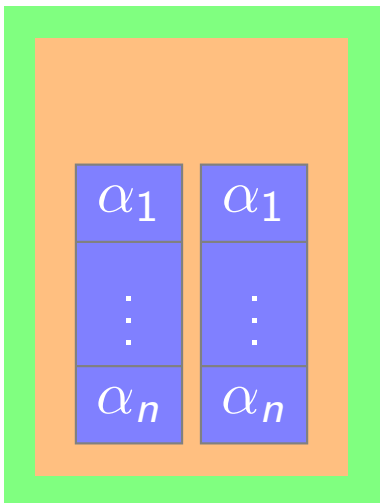


Higher-order pushdown automata

pop_2



Higher-order pushdown automata



Recognition power of higher-order pushown automata

- Damm, Goerdt (1986), Knapik, Niwinski, Urzyczyn, Walukiewicz (2005)

  level n PDA languages = level n *safe* OI languages

The safety condition

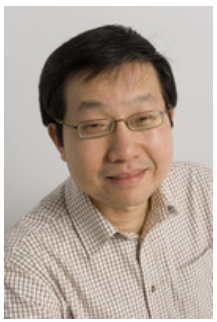
$$(\lambda x.M)N \rightarrow_\beta M[x \leftarrow N]$$

$$(\lambda y.M')[x \leftarrow N] = \lambda y.(M[x \leftarrow N]) \text{ if } y \notin FV(N)$$

The safety condition

$$(\lambda x.M)N \to_\beta M[x \leftarrow N]$$

$$(\lambda y.M')[x \leftarrow N] = \lambda y.(M[x \leftarrow N]) \text{ if } y \notin FV(N)$$



- Blum, Ong (2009)

Theorem

When a term is safe, it can be reduced without renaming variables.

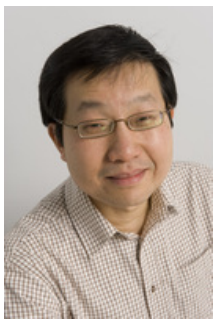
Theorem

The simply typed safe λ-calculus is strictly less expressive than the simply typed λ-calculus.

The safety condition

$$(\lambda x.M)N \to_\beta M[x \leftarrow N]$$

$$(\lambda y.M')[x \leftarrow N] = \lambda y.(M[x \leftarrow N]) \text{ if } y \notin FV(N)$$



- Blum, Ong (2009)

Theorem

When a term is safe, it can be reduced without renaming variables.

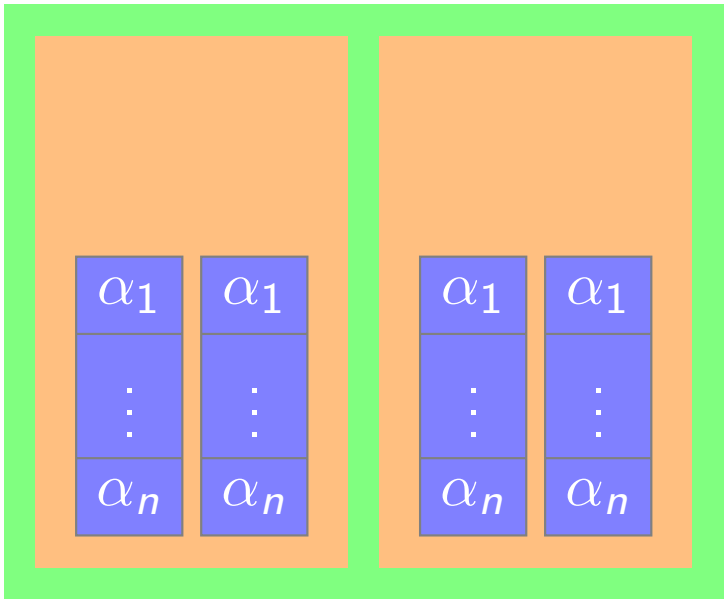
Theorem

The simply typed safe λ-calculus is strictly less expressive than the simply typed λ-calculus.

Problems

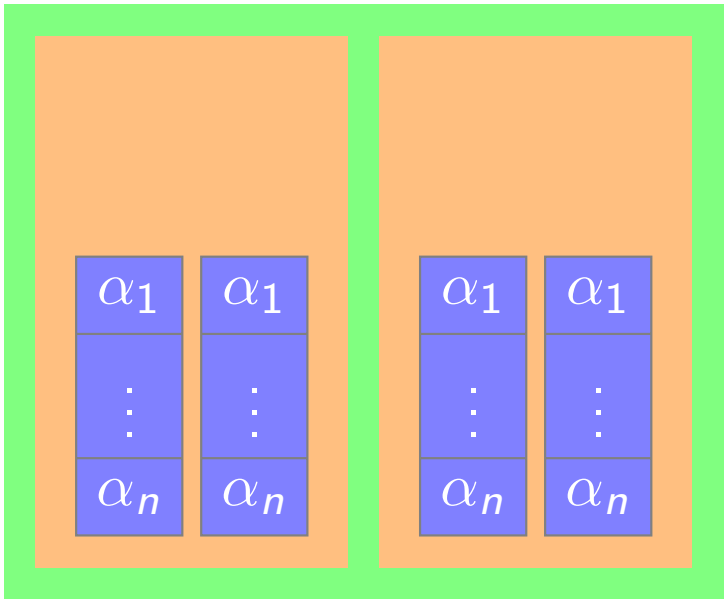
- Is the safe linear λ-calculus strictly less expressive than the linear λ-calculus?
- safe non-duplicating IO/OI languages $\stackrel{?}{=}$ non-duplicating IO/OI languages

Higher-order collapsible pushdown automata



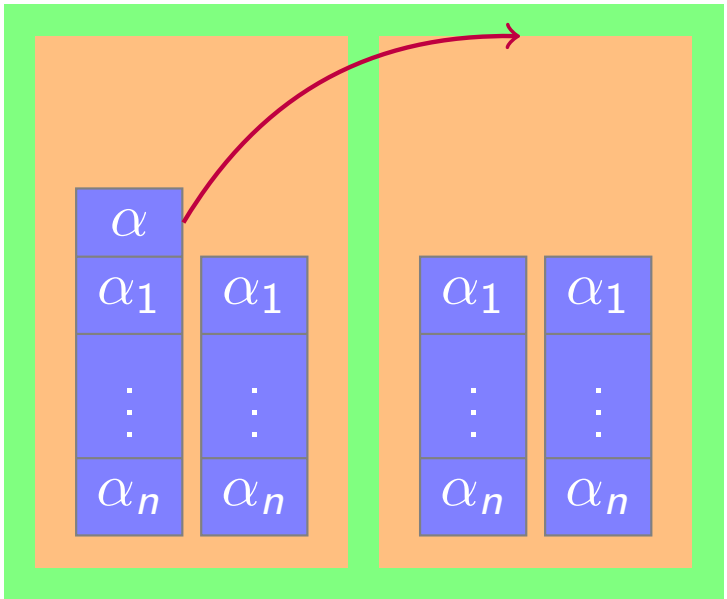
Higher-order collapsible pushdown automata

$push_2$



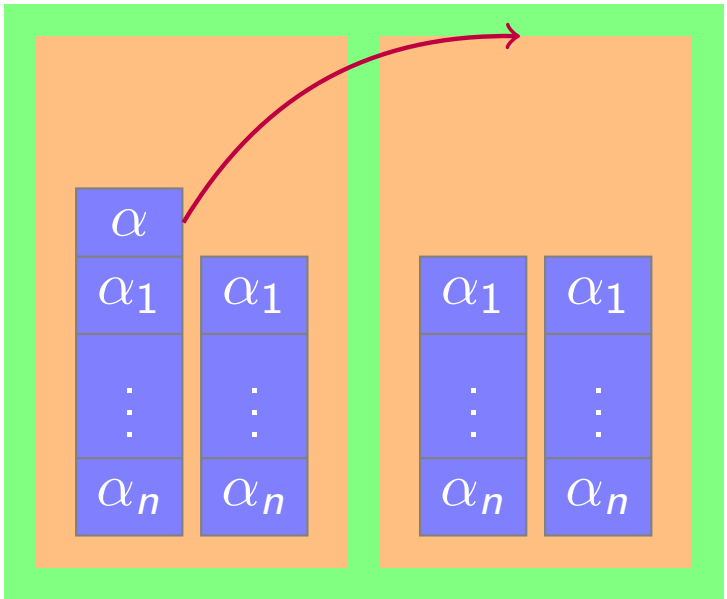
Higher-order collapsible pushdown automata

$push_2$



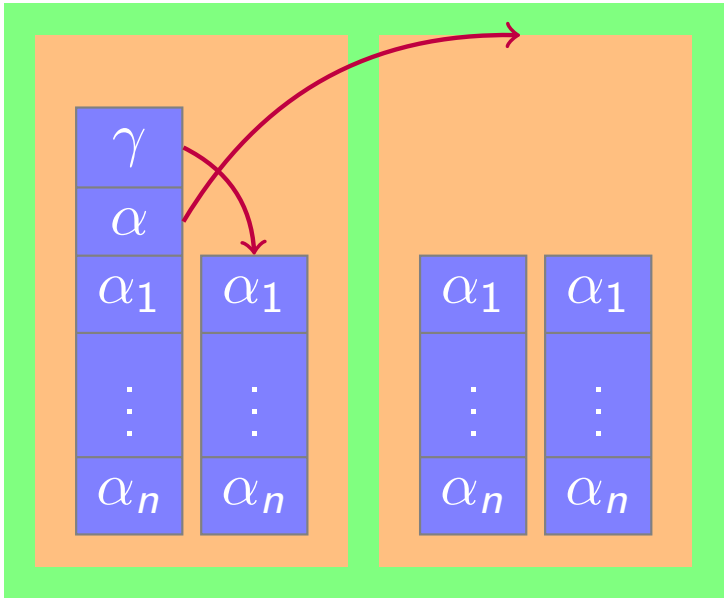
Higher-order collapsible pushdown automata

$push_1$



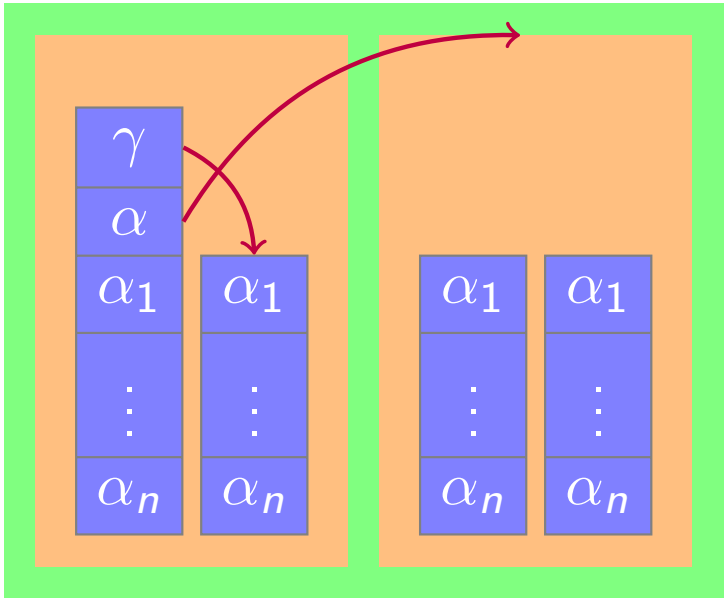
Higher-order collapsible pushdown automata

$push_1$



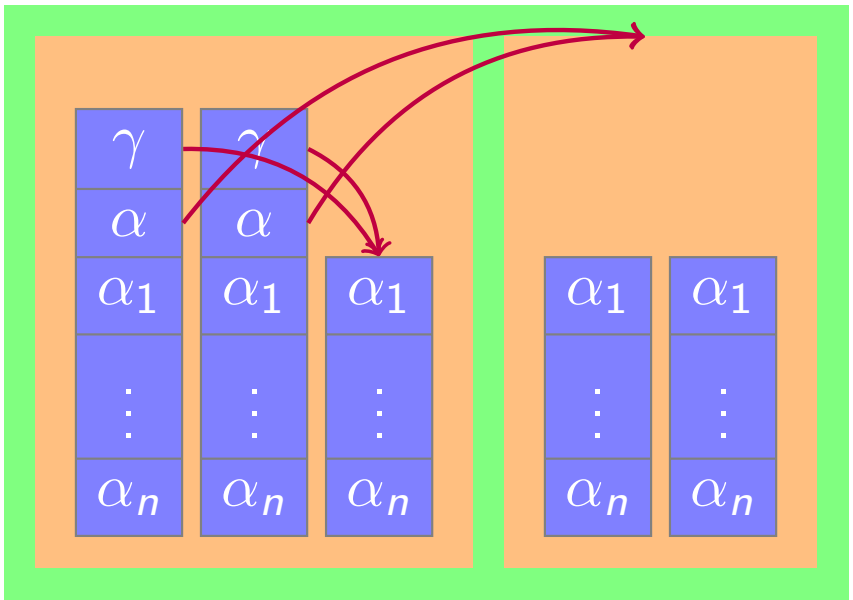
Higher-order collapsible pushdown automata

$copy_1$



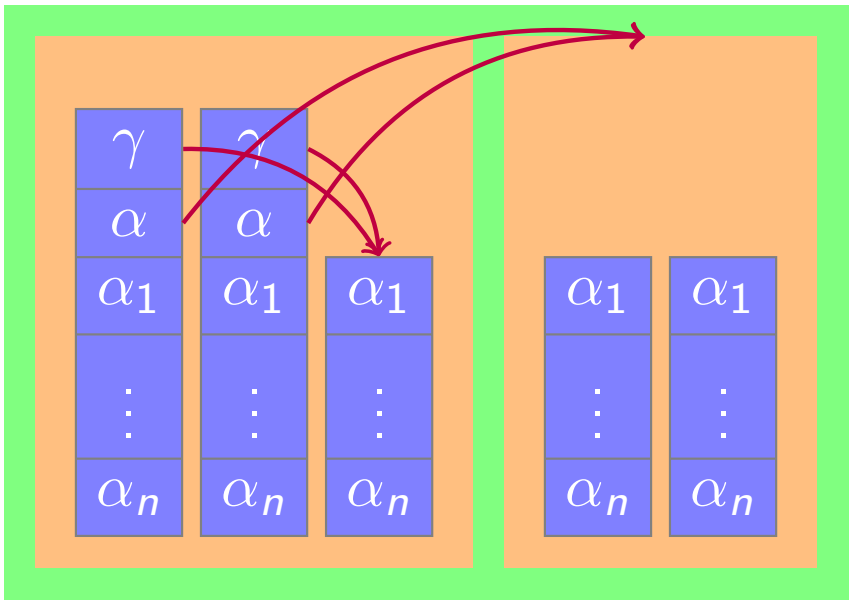
Higher-order collapsible pushdown automata

$copy_1$



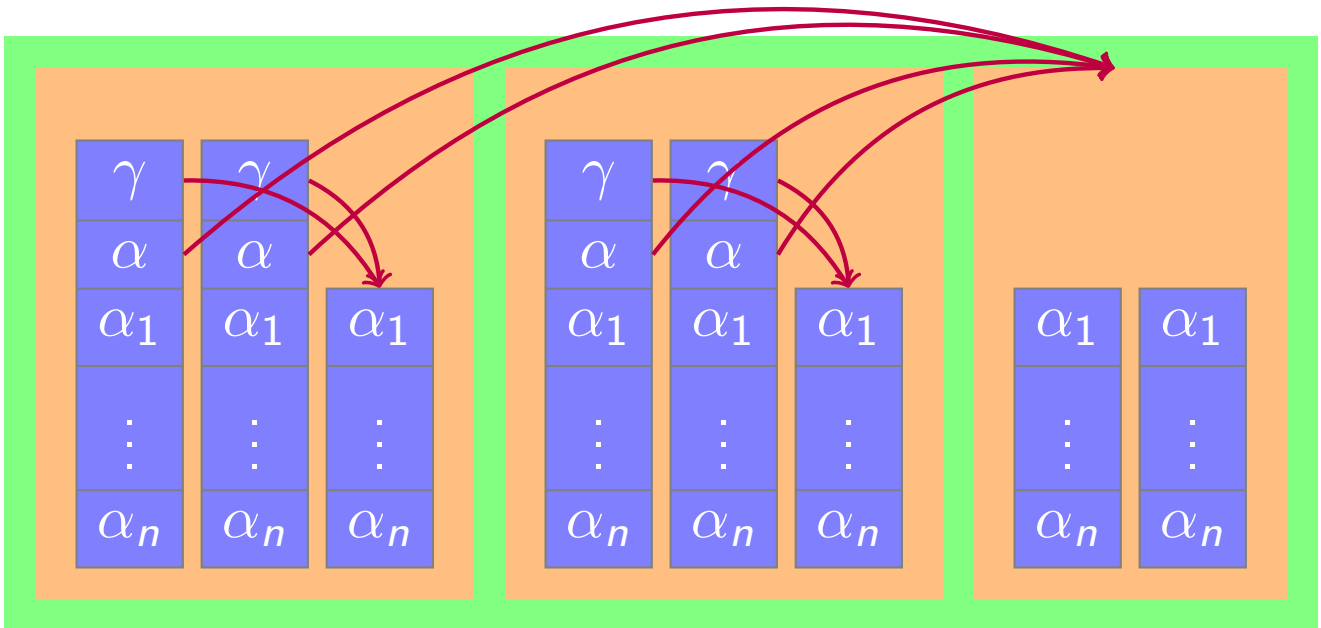
Higher-order collapsible pushdown automata

$copy_2$



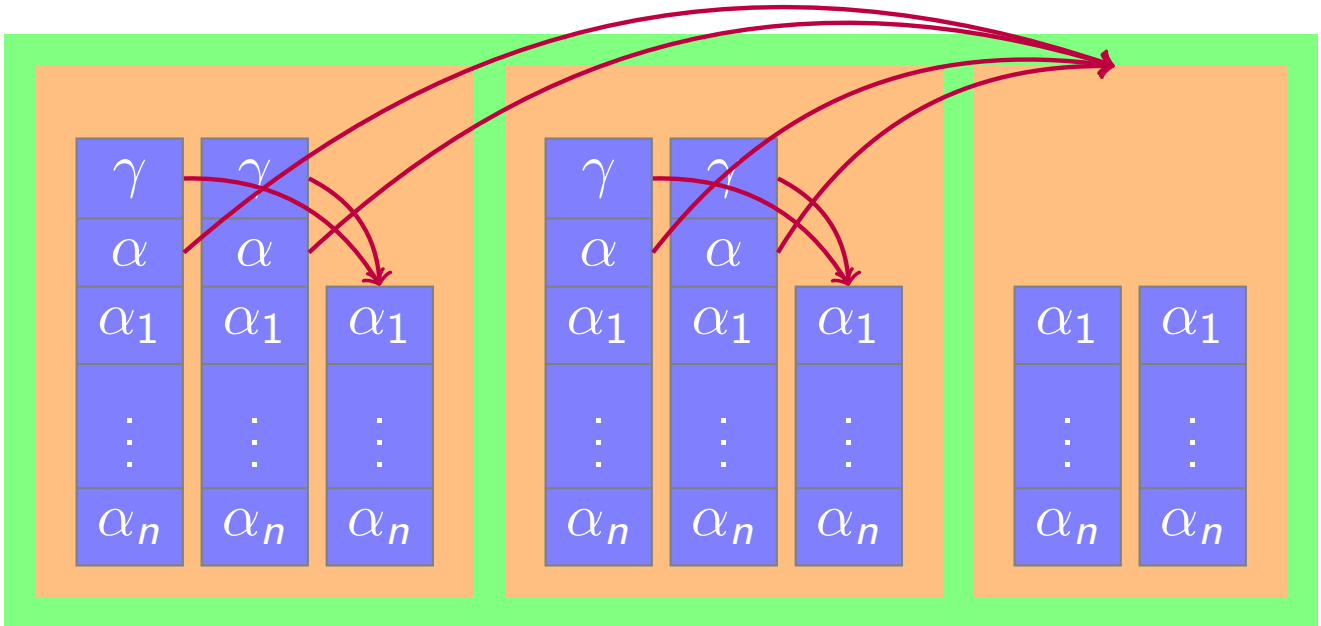
Higher-order collapsible pushdown automata

$copy_2$



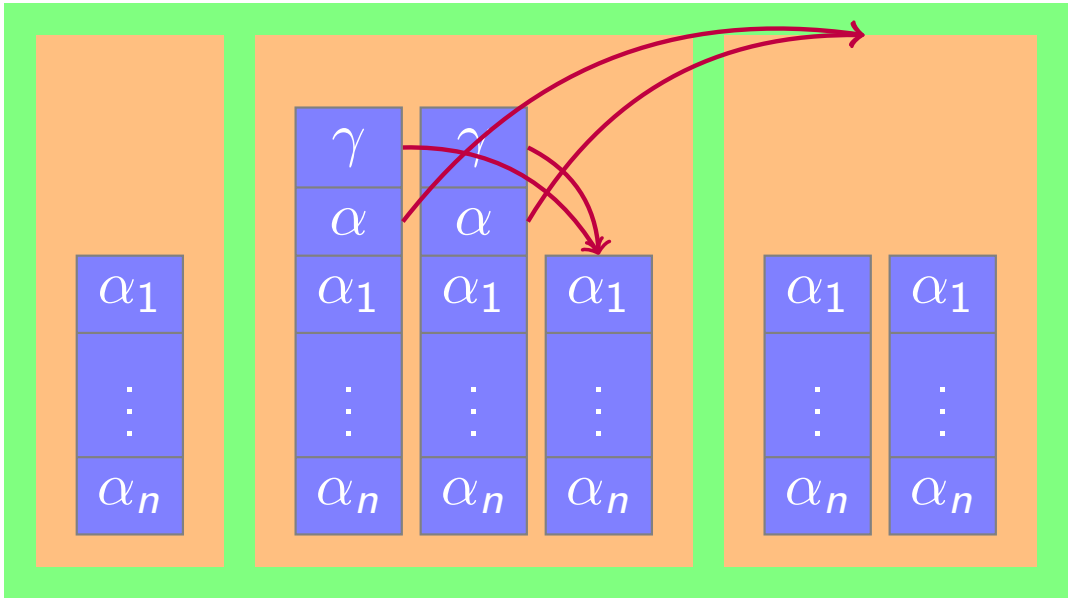
Higher-order collapsible pushdown automata

collapse



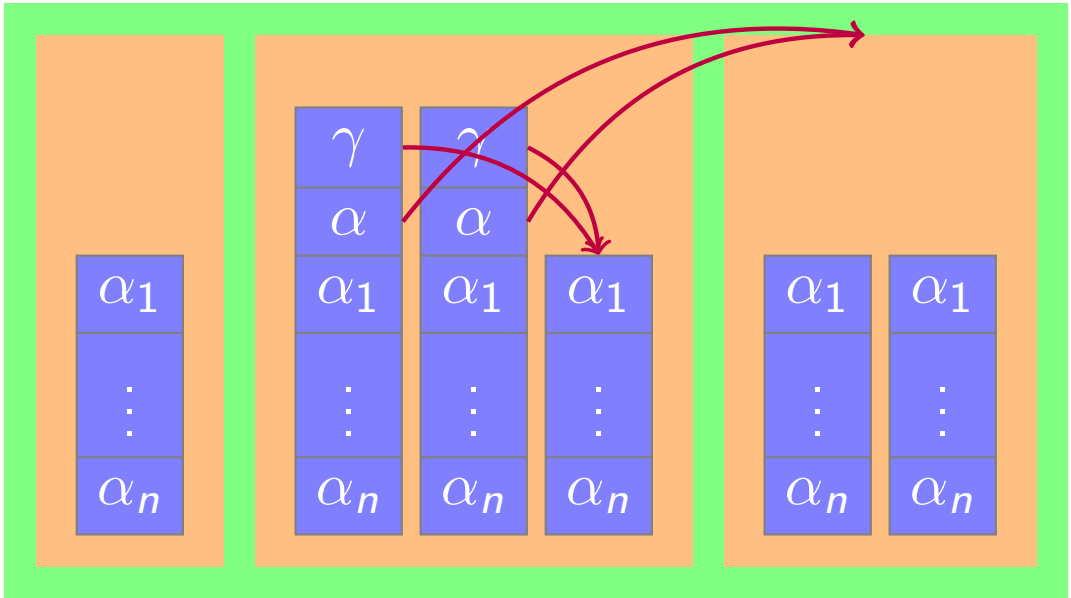
Higher-order collapsible pushdown automata

collapse



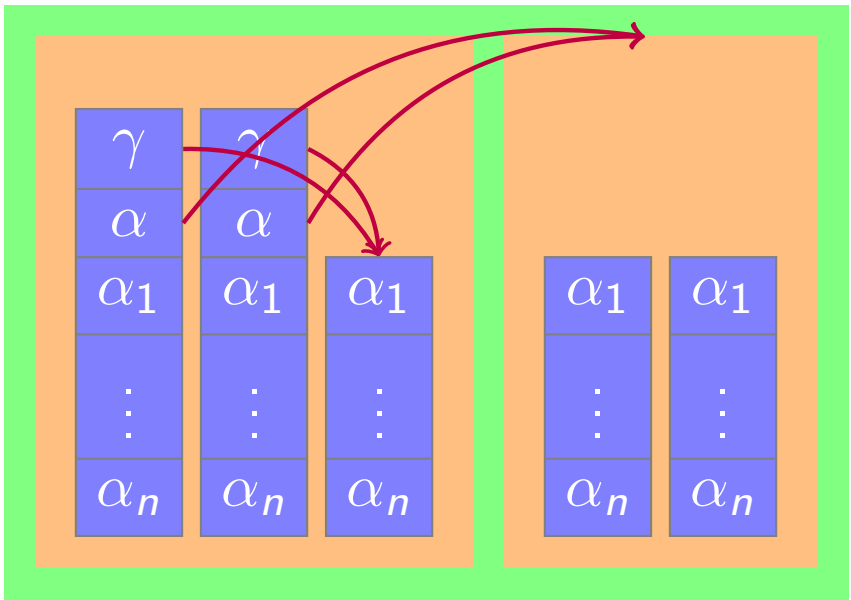
Higher-order collapsible pushdown automata

pop_2



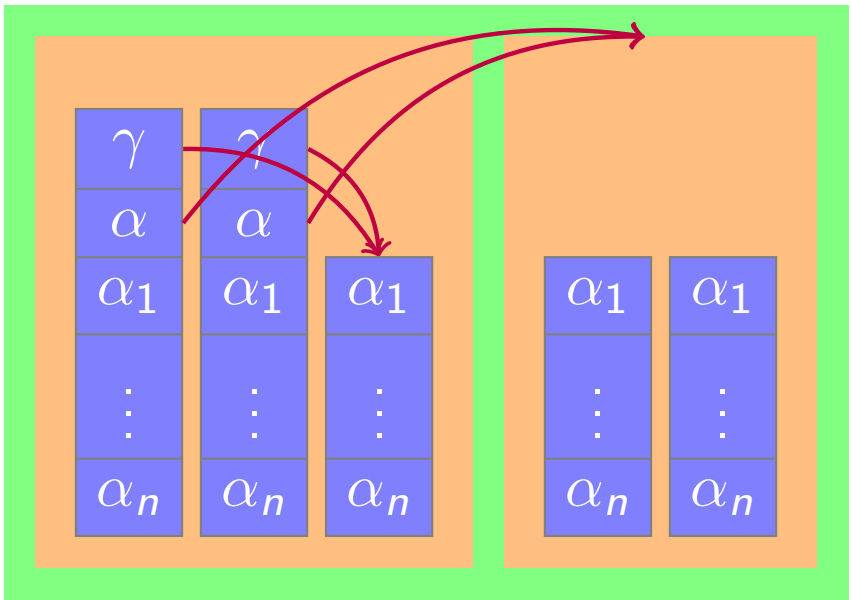
Higher-order collapsible pushdown automata

pop_2



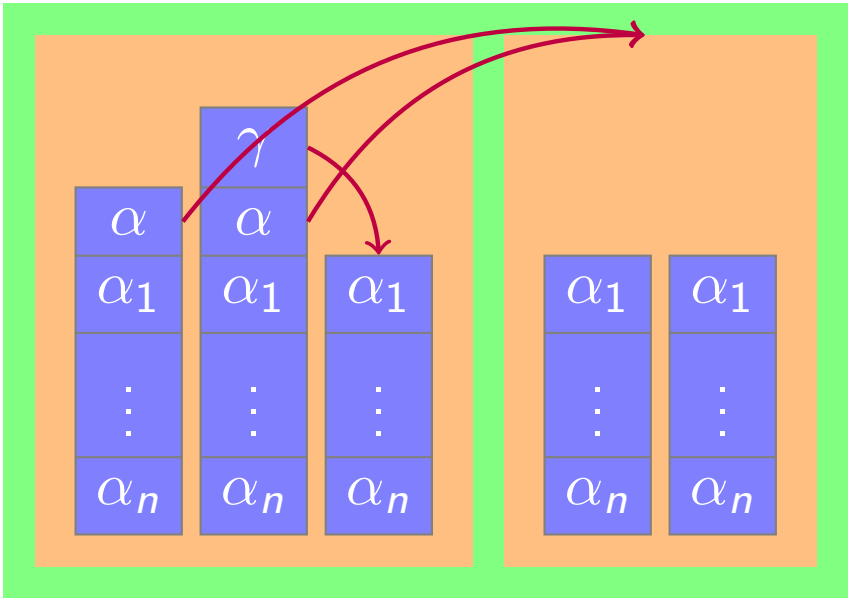
Higher-order collapsible pushdown automata

pop_0



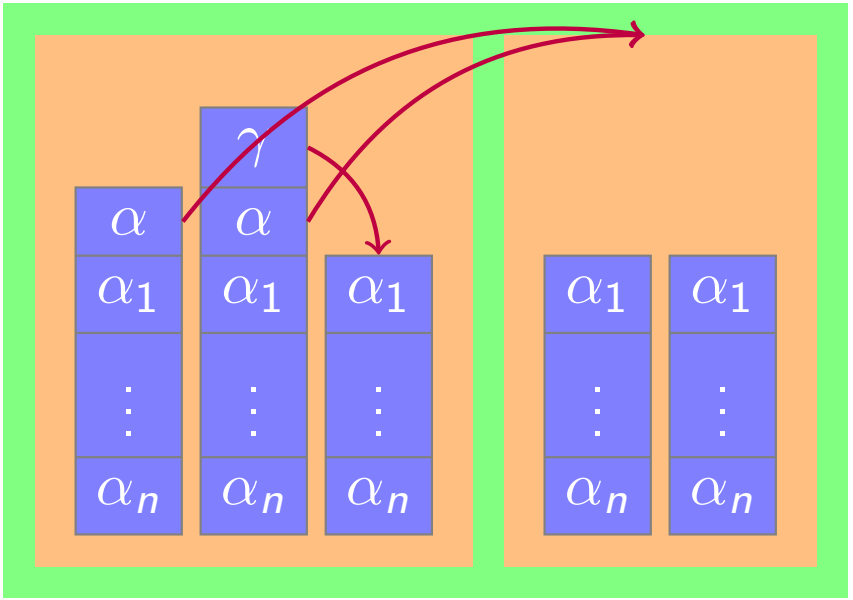
Higher-order collapsible pushdown automata

pop_0



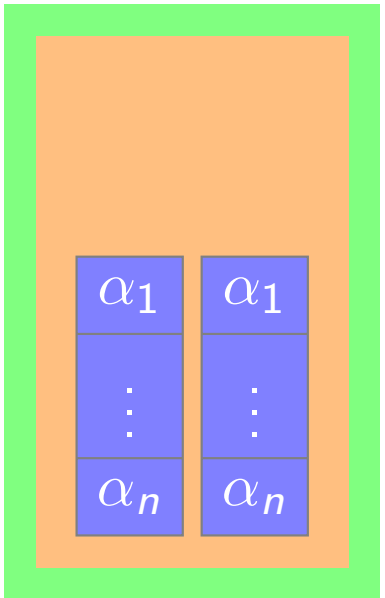
Higher-order collapsible pushdown automata

collapse

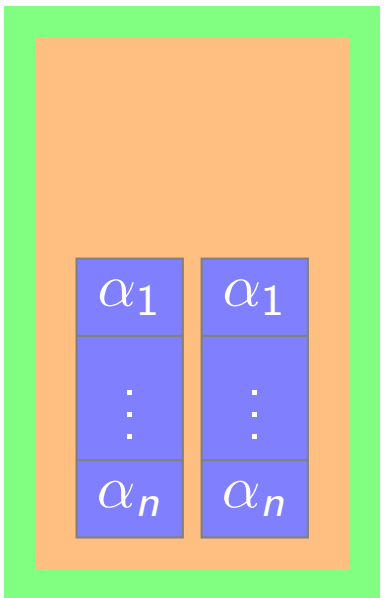


Higher-order collapsible pushdown automata

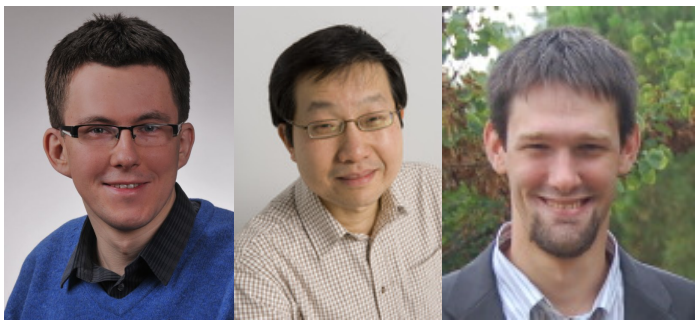
collapse



Higher-order collapsible pushdown automata



HO collapsible PDA and MCFL



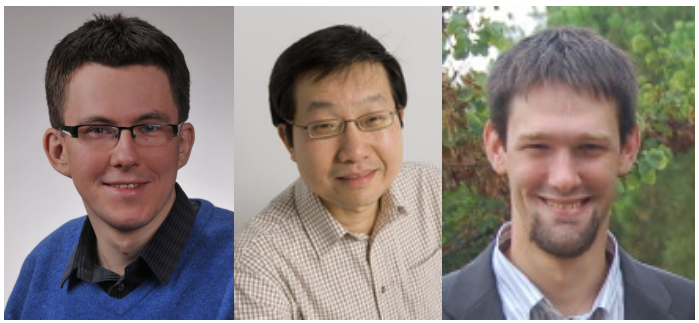
- Hague, Murawski, Ong, Serre

level n collapsible PDA languages = level n OI languages

Corollary

Level 3 collapsible PDA recognize MCFL and PMCFL.

HO collapsible PDA and MCFL



- Hague, Murawski, Ong, Serre

  level n collapsible PDA languages = level n OI languages

Corollary

Level 3 collapsible PDA recognize MCFL and PMCFL.

HO collapsible PDA and the Krivine machine

HO collapsible PDA:

- top-down evaluation of rules

HO collapsible PDA and the Krivine machine

HO collapsible PDA:

- top-down evaluation of rules $\equiv$ weak head normalisation of λ-terms

HO collapsible PDA and the Krivine machine

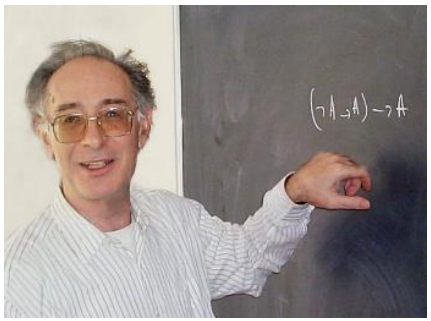
HO collapsible PDA:

- top-down evaluation of rules $\equiv$ weak head normalisation of λ-terms

the Krivine machine:

- weak head normalisation of λ-terms

The Krivine machine



- Krivine (198?/2006)

$$\langle M, \sigma, S \rangle$$

- M is the term evaluated
- σ is the substitution providing values to the free variables of M
- S is the stack of arguments of M

$$\begin{aligned}
\langle \lambda x.M, \sigma, (N, \tau) :: S \rangle &\rightarrow \langle M, (x, N, \tau) :: \sigma, S \rangle \\
\langle (MN), \sigma, S \rangle &\rightarrow \langle M, \sigma, (N, \sigma) :: S \rangle \\
\langle x, \sigma, S \rangle &\rightarrow \langle M, \tau, S \rangle \text{ if } \sigma(x) = (M, \tau)
\end{aligned}$$

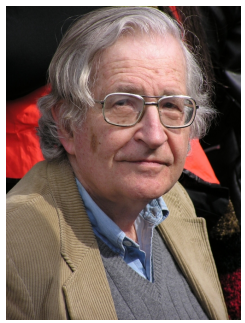
Conclusion

- there is a tight relation between language definition and linear λ-caclulus
- MCFL and simply typed λ-calculus with fixed-point provide a nice example

Conclusion

- there is a tight relation between language definition and linear λ-caclulus
- MCFL and simply typed λ-calculus with fixed-point provide a nice example

MCFL: the missing link?



- Chomsky (2004)

pened. The systems that capture other properties of language, for example transformational grammar, hold no interest for mathematics. But I do not think that that is a necessary truth. It could turn out that there would be richer or more appropriate mathematical ideas that would capture other, maybe deeper properties of language than context-free grammars do. In that case you have another branch of applied mathematics which might have linguistic consequences. That could be exciting.