Generating Control Languages with Abstract Categorial Grammars

Makoto Kanazawa, NII, Tokyo
Sylvain Salvati, LaBRI, Bordeaux

This talk

$$\mathbf{C}_k \subsetneq 2^{k-1}\text{-MCFL}$$

- $\mathbf{C}_k$: k-th level of the control language hierarchy (Weir 1992)
- m-MCFL: m-multiple context-free grammars (Seki et al. 1991)

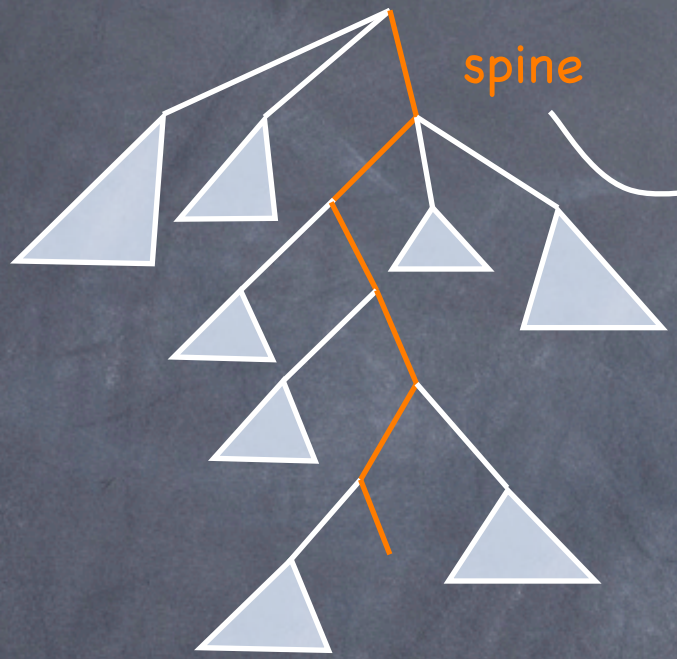
This talk

$$\mathbf{C}_k \subsetneq \mathit{STR}(\mathbf{A}_{2^k}) \subseteq 2^{k-1}\text{-MCFL}$$

- $\mathbf{C}_k$: k-th level of the control language hierarchy (Weir 1992)
- m-MCFL: m-multiple context-free grammars (Seki et al. 1991)
- $\mathit{STR}(\mathbf{A}_m)$: the class of string languages generated by second-order abstract categorial grammars (de Groote 2001) of width m

Control languages

CFG derivation tree T



head

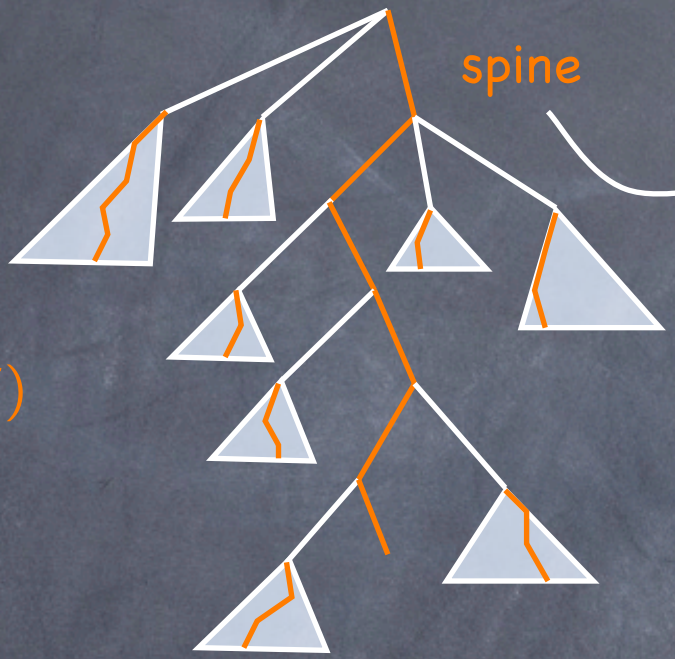
$\pi : A \to B_1 \ldots B_h \ldots B_n$

control string = sequence of productions

$\pi_1 \ldots \pi_n$

Control languages

CFG derivation tree T



head

$\pi : A \to B_1 \ldots B_h \ldots B_n$

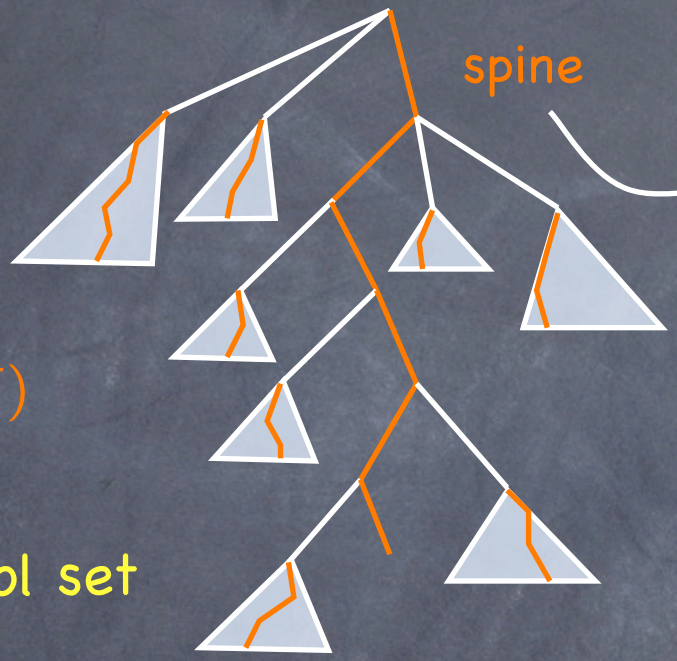
control string = sequence of productions

$\pi_1 \ldots \pi_n$

Spines(T)

Control languages

CFG derivation tree T



head

$\pi : A \to B_1 \ldots B_h \ldots B_n$

control string = sequence of productions

$\pi_1 \ldots \pi_n$

Spines(T)

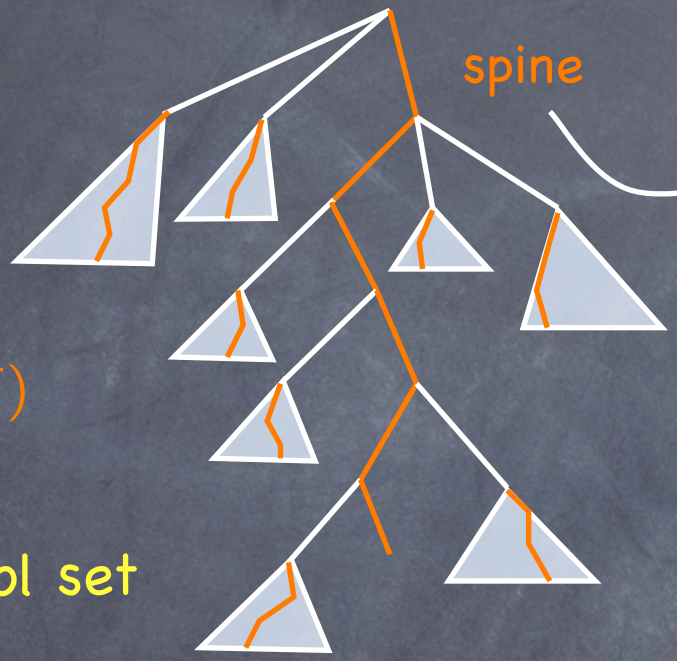
control set

$$DT(G, C) = \{\, T \mid T \text{ is a complete derivation tree of } G \text{ and } \text{Spines}(T) \subseteq C \,\}$$

$$L(G, C) = \{\, \text{yield}(T) \mid T \in DT(G, C) \,\}$$

Control languages

CFG derivation tree T



head

$\pi : A \to B_1 \ldots B_h \ldots B_n$

control string = sequence of productions

$\pi_1 \ldots \pi_n$

$\mathrm{Spines}(T)$

control set

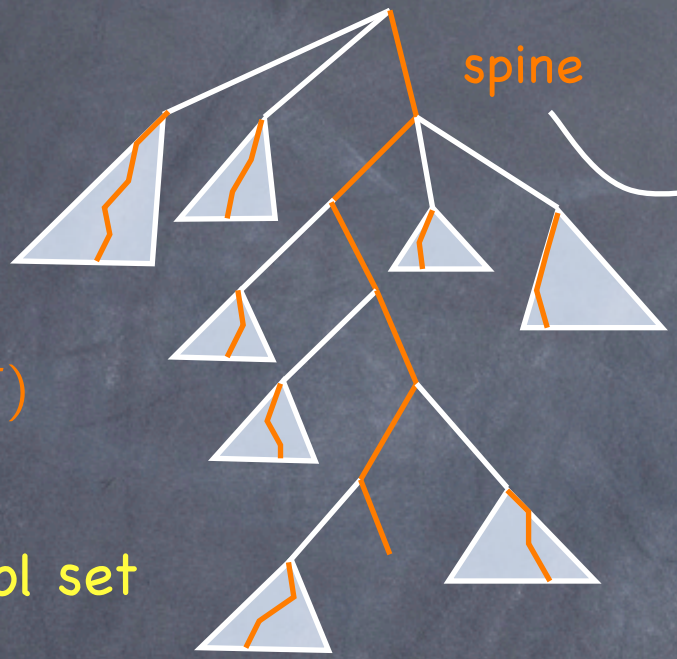
$$DT(G, C) = \{\, T \mid T \text{ is a complete derivation tree of } G \text{ and } \mathrm{Spines}(T) \subseteq C \,\}$$

$$L(G, C) = \{\, \mathrm{yield}(T) \mid T \in DT(G, C) \,\}$$

$$\mathbf{C}_1 = \mathrm{CFL} \quad \mathbf{C}_{k+1} = \{\, L(G, C) \mid G \text{ is an LDG and } C \in \mathbf{C}_k \,\}$$

Control languages

CFG derivation tree T



head

$\pi : A \to B_1 \ldots B_h \ldots B_n$

control string = sequence of productions

$\pi_1 \ldots \pi_n$

Spines(T)

control set

non-regular tree language

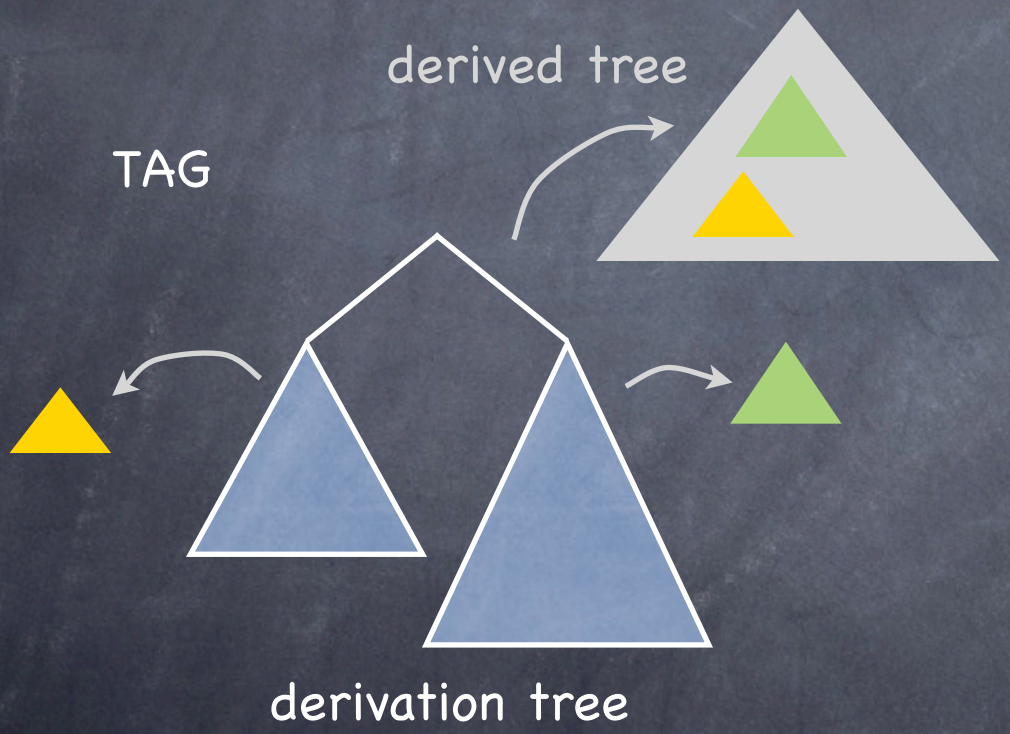
$$DT(G, C) = \{\, T \mid T \text{ is a complete derivation tree of } G \text{ and } \text{Spines}(T) \subseteq C \,\}$$

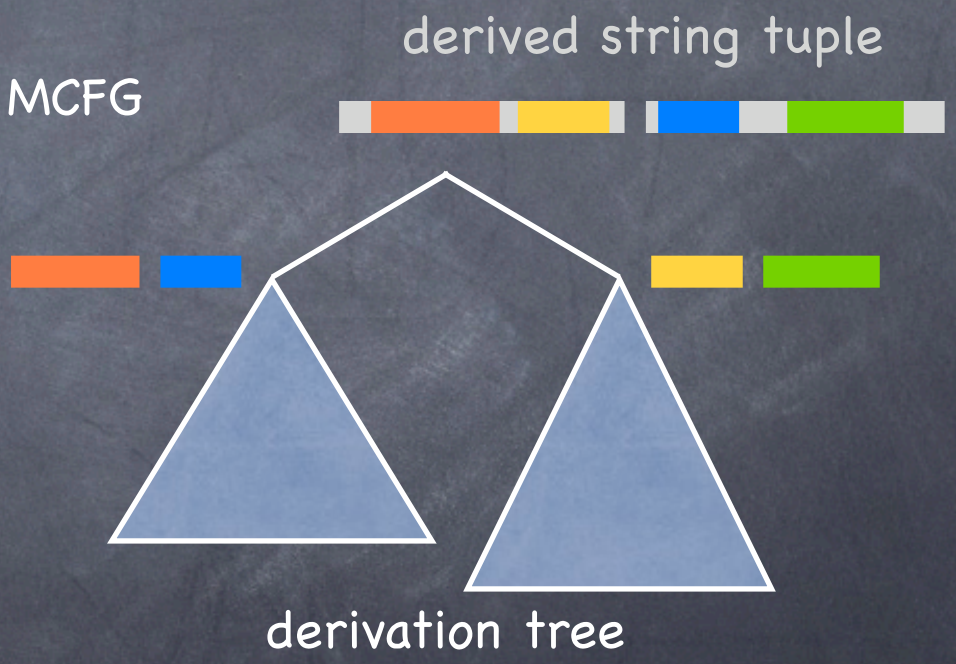
$$L(G, C) = \{\, \text{yield}(T) \mid T \in DT(G, C) \,\}$$

$$\mathbf{C}_1 = \text{CFL} \quad \mathbf{C}_{k+1} = \{\, L(G, C) \mid G \text{ is an LDG and } C \in \mathbf{C}_k \,\}$$

Grammars with context-free derivations

- tree-adjoining grammars
- multiple context-free grammars





complex yields
set of derivation trees = local set

Relationship

$\mathbf{C}_1 = \mathrm{CFL} = 1\text{-MCFL}$

$\mathbf{C}_2 = \mathrm{TAL} \subsetneq 2\text{-MCFL}$

$\mathbf{C}_3$

$\mathbf{C}_4$

$\vdots$

Relationship

$\mathbf{C}_1 = \mathrm{CFL} = 1\text{-MCFL}$

$\mathbf{C}_2 = \mathrm{TAL} \subsetneq 2\text{-MCFL}$

$\mathbf{C}_3$

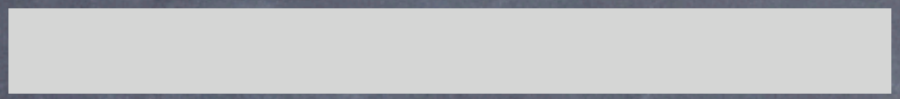
$\mathbf{C}_4$

$\vdots$

?

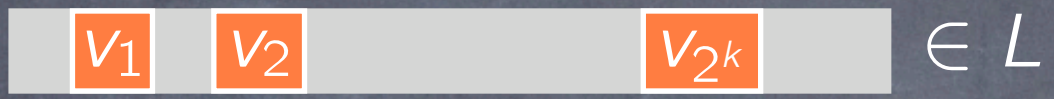
Pumping Lemmas for CLH and MCFL

$L \in \mathbf{C}_k \Rightarrow$ every long string in L can be pumped at 2^k intervals (Palis and Shende 1995)

 $\in L$

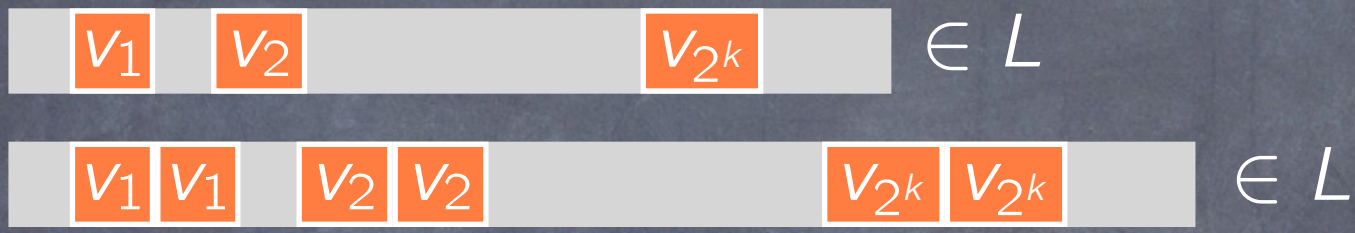
Pumping Lemmas for CLH and MCFL

$L \in \mathbf{C}_k \Rightarrow$ every long string in L can be pumped at 2^k intervals (Palis and Shende 1995)



Pumping Lemmas for CLH and MCFL

$L \in \mathbf{C}_k \Rightarrow$ every long string in L can be pumped at 2^k intervals (Palis and Shende 1995)

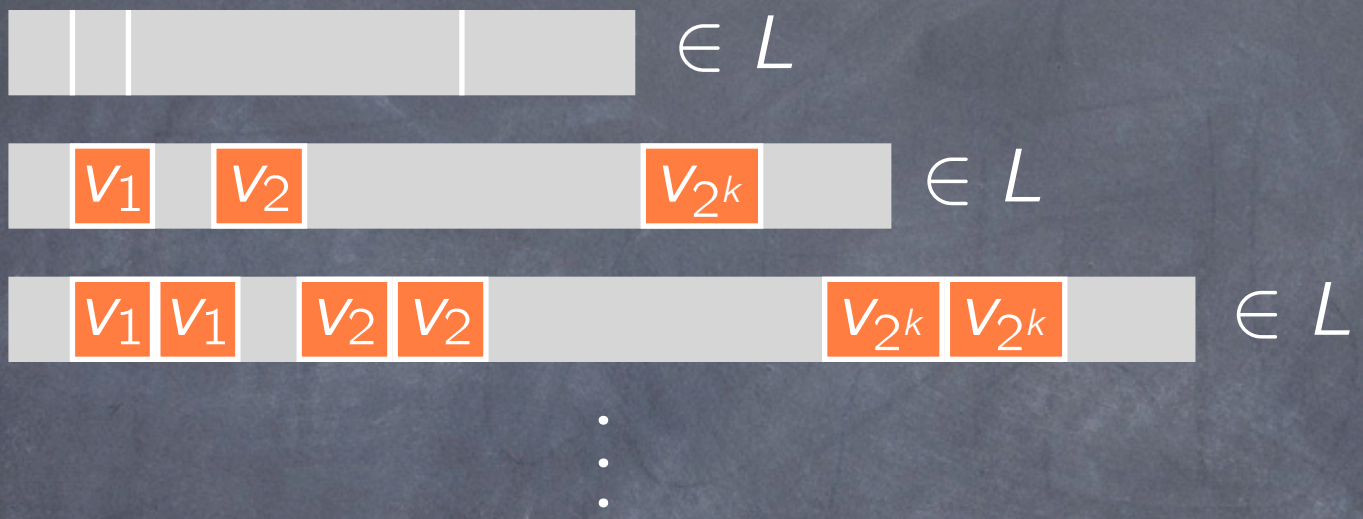


Pumping Lemmas for CLH and MCFL

$L \in \mathbf{C}_k \Rightarrow$ every long string in L can be pumped at 2^k intervals (Palis and Shende 1995)

Pumping Lemmas for CLH and MCFL

$L \in \mathbf{C}_k \Rightarrow$ every long string in L can be pumped at 2^k intervals (Palis and Shende 1995)



Pumping Lemmas for CLH and MCFL

$L \in \mathbf{C}_k \Rightarrow$ every long string in L can be pumped at 2^k intervals (Palis and Shende 1995)

$$\{a_1^n \ldots a_p^n \mid n \geq 0\} \in \mathbf{C}_k \Leftrightarrow p \leq 2^k$$

Pumping Lemmas for CLH and MCFL

$L \in \mathbf{C}_k \Rightarrow$ every long string in L can be pumped at 2^k intervals (Palis and Shende 1995)

$$\{a_1^n \ldots a_p^n \mid n \geq 0\} \in \mathbf{C}_k \Leftrightarrow p \leq 2^k$$

$L \in m$-MCFL $\Rightarrow$ some long string in L can be pumped at $2m$ intervals (Seki et al. 1991)

Pumping Lemmas for CLH and MCFL

$L \in \mathbf{C}_k \Rightarrow$ every long string in L can be pumped at 2^k intervals (Palis and Shende 1995)

$$\{a_1^n \ldots a_p^n \mid n \geq 0\} \in \mathbf{C}_k \Leftrightarrow p \leq 2^k$$

$L \in m$-MCFL $\Rightarrow$ some long string in L can be pumped at $2m$ intervals (Seki et al. 1991)

$$\{a_1^n \ldots a_p^n \mid n \geq 0\} \in m\text{-MCFL} \Leftrightarrow p \leq 2m$$

Pumping Lemmas for CLH and MCFL

$L \in \mathbf{C}_k \Rightarrow$ every long string in L can be pumped at 2^k intervals (Palis and Shende 1995)

$$\{a_1^n \ldots a_p^n \mid n \geq 0\} \in \mathbf{C}_k \Leftrightarrow p \leq 2^k$$

$L \in m$-MCFL $\Rightarrow$ some long string in L can be pumped at $2m$ intervals (Seki et al. 1991)

$$\{a_1^n \ldots a_p^n \mid n \geq 0\} \in m\text{-MCFL} \Leftrightarrow p \leq 2m$$

$$\mathbf{C}_k \subsetneq 2^{k-1}\text{-MCFL}\ ?$$

Proof of inclusion

$$\mathbf{C}_k \subseteq STR(\mathbf{A}_{2^k}) \subseteq 2^{k-1}\text{-MCFL}$$

string-generating power of second-order ACGs of width 2^k

Proof of inclusion

$$\mathbf{C}_k \subseteq STR(\mathbf{A}_{2^k}) \subseteq 2^{k-1}\text{-MCFL}$$

string-generating power of second-order ACGs of width 2^k

ACGs generate

- ✓ strings
- ✓ trees
- ✓ logical formulas

Proof of inclusion

$$\mathbf{C}_k \subseteq STR(\mathbf{A}_{2^k}) \subseteq 2^{k-1}\text{-MCFL}$$

string-generating power of second-order ACGs of width 2^k

ACGs generate

- ✓ strings
- ✓ trees
- ✓ logical formulas

} represented by linear λ-terms

Proof of inclusion

$$\mathbf{C}_k \subseteq STR(\mathbf{A}_{2^k}) \subseteq 2^{k-1}\text{-MCFL}$$

string-generating power of second-order ACGs of width 2^k

ACGs generate

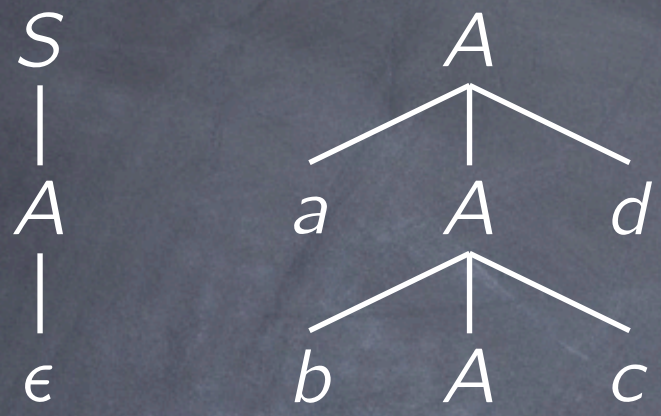
- ✓ strings
- ✓ trees
- ✓ logical formulas

} represented by linear λ-terms

$$C \in STR(\mathbf{A}_m) \Rightarrow DT(G, C) \in TR(\mathbf{A}_m) \Rightarrow L(G, C) \in STR(\mathbf{A}_{2m})$$

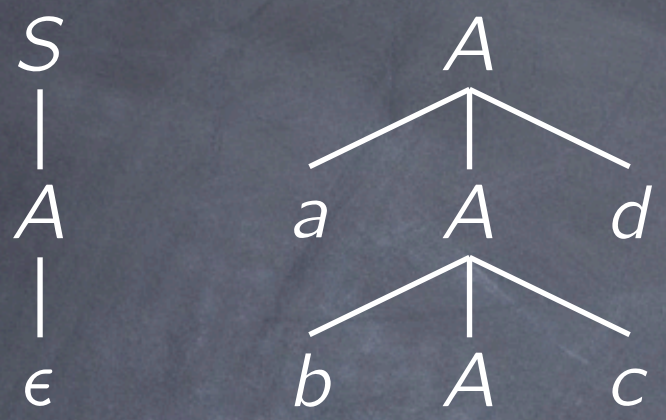
tree-generating power

Trees and strings as linear λ-terms



trees

Trees and strings as linear λ-terms



trees

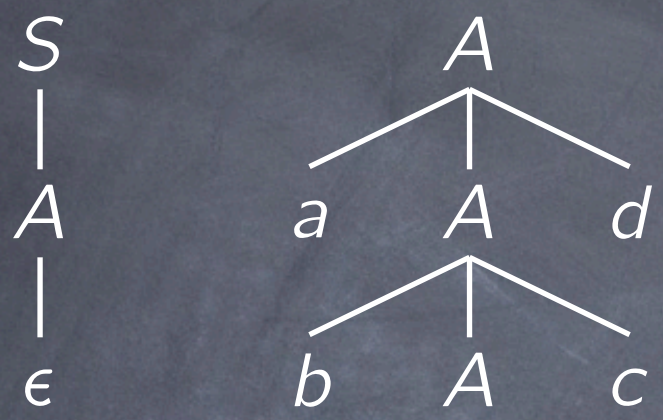
rank 1

$S(A^{(1)}(\epsilon))$

$A^{(3)}(a, A^{(3)}(b, A^{(0)}, c), d)$

terms

Trees and strings as linear λ-terms



trees

rank 1

$S(A^{(1)}(\epsilon))$

$A^{(3)}(a, A^{(3)}(b, A^{(0)}, c), d)$

terms

$S^{o\to o}\ (A^{(1)^{o\to o}}\ \epsilon^o)$

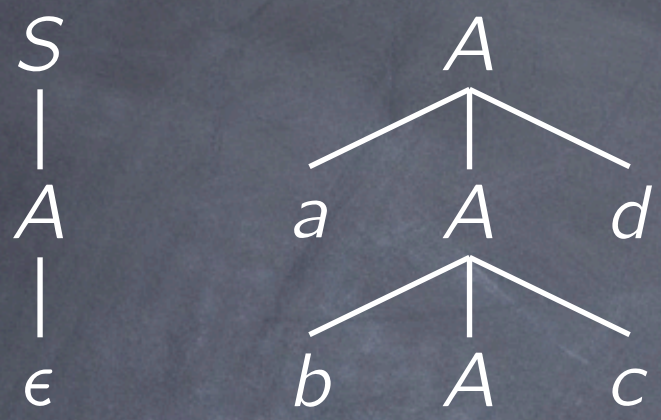
$A^{(3)^{o\to o\to o\to o}}\ a^o\ (A^{(3)^{o\to o\to o\to o}}\ b^o\ A^{(1)^o}\ c^o)\ d^o$

λ-terms

$(M^{\alpha\to\beta}\ N^\alpha)^\beta$

application

Trees and strings as linear λ-terms



trees

rank 1

$S(A^{(1)}(\epsilon))$

$A^{(3)}(a, A^{(3)}(b, A^{(0)}, c), d)$

terms

$S^{o\to o}\ (A^{(1)^{o\to o}}\ \epsilon^o)$

$A^{(3)^{o\to o\to o\to o}}\ a^o\ (A^{(3)^{o\to o\to o\to o}}\ b^o\ A^{(1)^o}\ c^o)\ d^o$

λ-terms

$(M^{\alpha\to\beta}\ N^\alpha)^\beta$

application

$(\lambda x^\alpha . M^\beta)^{\alpha\to\beta}$

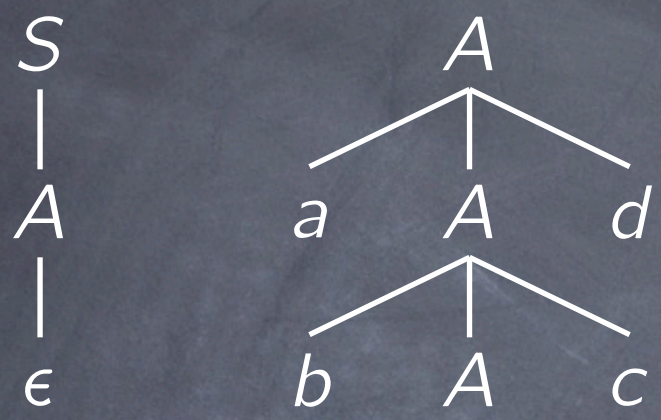
λ-abstraction

strings

$/a_1 \ldots a_n/ = \lambda z^o . a_1^{o\to o}(\ldots(a_n^{o\to o}\ z)\ldots)$

concatenation = function composition

Trees and strings as linear λ-terms



trees

rank 1

$S(A^{(1)}(\epsilon))$

$A^{(3)}(a, A^{(3)}(b, A^{(0)}, c), d)$

terms

$S^{o\to o}\ (A^{(1)^{o\to o}}\ \epsilon^o)$

$A^{(3)^{o\to o\to o\to o}}\ a^o\ (A^{(3)^{o\to o\to o\to o}}\ b^o\ A^{(1)^o}\ c^o)\ d^o$

λ-terms

$(M^{\alpha\to\beta}\ N^\alpha)^\beta$

application

$(\lambda x^\alpha . M^\beta)^{\alpha\to\beta}$

λ-abstraction

strings

$/a_1 \dots a_n/ = \lambda z^o . a_1^{o\to o}(\dots(a_n^{o\to o}\ z)\dots)$

$\ulcorner a_1 \dots a_n \urcorner = a_1^{(1)}(\dots(a_{n-1}^{(1)}(a_n^{(0)}))\dots)$

monadic tree

concatenation = function composition

Formal definition of ACGs

ACG (de Groote 2001)

$$\mathscr{G} = (\Sigma, \Sigma', \mathscr{L}, s)$$

Formal definition of ACGs

ACG (de Groote 2001)

$$\mathscr{G} = (\Sigma, \Sigma', \mathscr{L}, s)$$

atomic types — constants — typing of constants

$\Sigma = (A, C, \tau)$ abstract vocabulary

$\Sigma' = (A', C', \tau')$ object vocabulary

$\mathscr{L}$: $A \rightarrow$ types over A' lexicon

$C \rightarrow$ linear λ-terms over Σ' where $\vdash_{\Sigma'} \mathscr{L}(c) : \mathscr{L}(\tau(c))$

$s \in A$ distinguished type

Formal definition of ACGs

ACG (de Groote 2001)

$$\mathscr{G} = (\Sigma, \Sigma', \mathscr{L}, s)$$

atomic types — constants — typing of constants

$\Sigma = (A, C, \tau)$ abstract vocabulary

$\Sigma' = (A', C', \tau')$ object vocabulary

$\mathscr{L} : A \to$ types over A' lexicon

$C \to$ linear λ-terms over Σ' where $\vdash_{\Sigma'} \mathscr{L}(c) : \mathscr{L}(\tau(c))$

$s \in A$ distinguished type

$$\mathcal{A}(\mathscr{G}) = \{ P \mid \vdash_{\Sigma} P : s \}$$ abstract language

$$\mathcal{O}(\mathscr{G}) = \{ \mathscr{L}(P) \mid P \in \mathcal{A}(\mathscr{G}) \}$$ object language

Example: TAG as tree-generating ACG

abstract constant	type		object term	type
C	$p \to s$	$\overset{\mathscr{L}}{\mapsto}$	$\lambda y.S(y(A^{(1)}\,\epsilon))$	$(o \to o) \to o$
D	$p \to p$	$\overset{\mathscr{L}}{\mapsto}$	$\lambda y.\lambda x.A^{(3)}a(y(A^{(3)}bxc))d$	$(o \to o) \to o \to o$
E	p	$\overset{\mathscr{L}}{\mapsto}$	$\lambda x.x$	$o \to o$

Example: TAG as tree-generating ACG

lexicon

abstract constant	type		object term	type
C	$p \to s$	$\overset{\mathscr{L}}{\mapsto}$	$\lambda y.S(y(A^{(1)}\,\epsilon))$	$(o \to o) \to o$
D	$p \to p$	$\overset{\mathscr{L}}{\mapsto}$	$\lambda y.\lambda x.A^{(3)}a(y(A^{(3)}bxc))d$	$(o \to o) \to o \to o$
E	p	$\overset{\mathscr{L}}{\mapsto}$	$\lambda x.x$	$o \to o$

Example: TAG as tree-generating ACG

lexicon

abstract constant	type		object term	type
C	$p \to s$	$\overset{\mathscr{L}}{\mapsto}$	$\lambda y.S(y(A^{(1)}\,\epsilon))$	$(o \to o) \to o$
D	$p \to p$	$\overset{\mathscr{L}}{\mapsto}$	$\lambda y.\lambda x.A^{(3)}a(y(A^{(3)}bxc))d$	$(o \to o) \to o \to o$
E	p	$\overset{\mathscr{L}}{\mapsto}$	$\lambda x.x$	$o \to o$

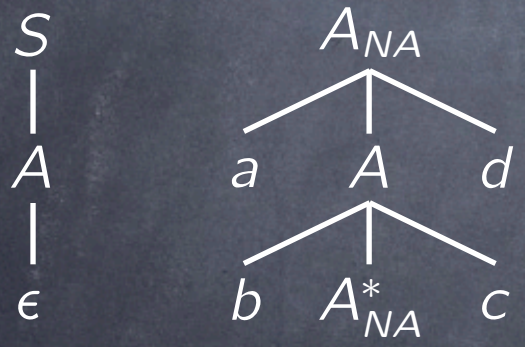
$$\mathscr{L}(p) = o \to o$$
$$\mathscr{L}(s) = o$$

Example: TAG as tree-generating ACG

lexicon

abstract constant	type		object term	type
C	$p \to s$	$\overset{\mathscr{L}}{\mapsto}$	$\lambda y.S(y(A^{(1)}\,\epsilon))$	$(o \to o) \to o$
D	$p \to p$	$\overset{\mathscr{L}}{\mapsto}$	$\lambda y.\lambda x.A^{(3)}a(y(A^{(3)}bxc))d$	$(o \to o) \to o \to o$
E	p	$\overset{\mathscr{L}}{\mapsto}$	$\lambda x.x$	$o \to o$

$$\mathscr{L}(p) = o \to o$$
$$\mathscr{L}(s) = o$$

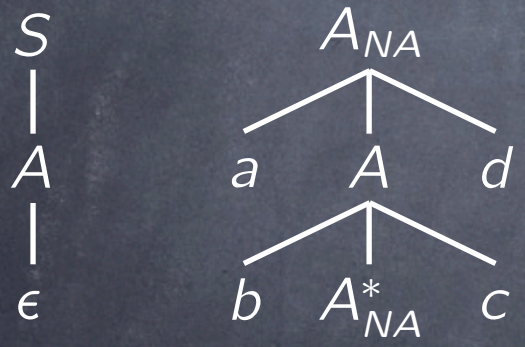


Example: TAG as tree-generating ACG

lexicon

abstract constant	type		object term	type
C	$p \to s$	$\overset{\mathscr{L}}{\mapsto}$	$\lambda y.S(y(A^{(1)}\,\epsilon))$	$(o \to o) \to o$
D	$p \to p$	$\overset{\mathscr{L}}{\mapsto}$	$\lambda y.\lambda x.A^{(3)}a(y(A^{(3)}bxc))d$	$(o \to o) \to o \to o$
E	p	$\overset{\mathscr{L}}{\mapsto}$	$\lambda x.x$	$o \to o$

$$\mathscr{L}(p) = o \to o$$
$$\mathscr{L}(s) = o$$



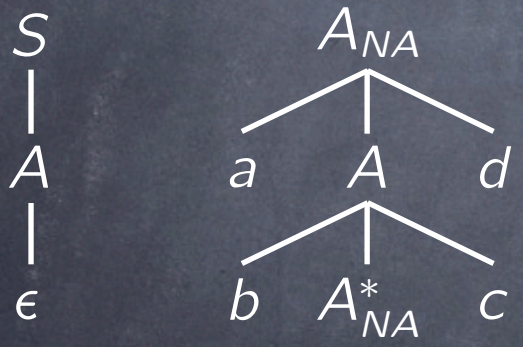
$$\mathscr{L}(\mathsf{C}(\mathsf{DE})) = S(A^{(3)}a(A^{(3)}b(A^{(1)}\epsilon)c)d)$$

Example: TAG as tree-generating ACG

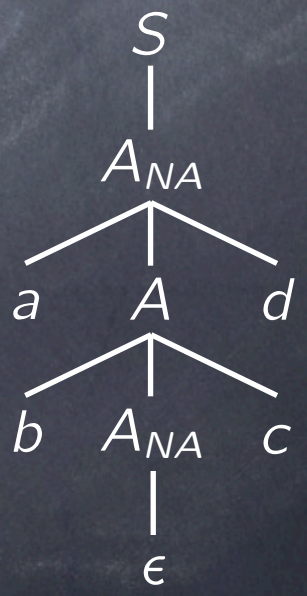
lexicon

abstract constant	type		object term	type
C	$p \to s$	$\overset{\mathscr{L}}{\mapsto}$	$\lambda y.S(y(A^{(1)}\,\epsilon))$	$(o \to o) \to o$
D	$p \to p$	$\overset{\mathscr{L}}{\mapsto}$	$\lambda y.\lambda x.A^{(3)}a(y(A^{(3)}bxc))d$	$(o \to o) \to o \to o$
E	p	$\overset{\mathscr{L}}{\mapsto}$	$\lambda x.x$	$o \to o$

$$\mathscr{L}(p) = o \to o$$
$$\mathscr{L}(s) = o$$



$$\mathscr{L}(\mathsf{C}(\mathsf{DE})) = S(A^{(3)}a(A^{(3)}b(A^{(1)}\epsilon)c)d)$$



From trees to strings

S	$o \to o$	$\overset{\mathscr{L}_{\text{yield}}}{\mapsto}$	$\lambda x.\lambda z.xz$	$(o \to o) \to (o \to o)$
$A^{(1)}$	$o \to o$	$\overset{\mathscr{L}_{\text{yield}}}{\mapsto}$	$\lambda x.\lambda z.xz$	$(o \to o) \to o \to o$
$A^{(3)}$	$o^3 \to o$	$\overset{\mathscr{L}_{\text{yield}}}{\mapsto}$	$\lambda x_1.\lambda x_2.\lambda x_3.\lambda z.x_1(x_2(x_3 z))$	$(o \to o)^3 \to o \to o$
ϵ	o	$\overset{\mathscr{L}_{\text{yield}}}{\mapsto}$	$\lambda z.z$	$o \to o$
a	o	$\overset{\mathscr{L}_{\text{yield}}}{\mapsto}$	$\lambda z.az$	$o \to o$
b	o	$\overset{\mathscr{L}_{\text{yield}}}{\mapsto}$	$\lambda z.bz$	$o \to o$
c	o	$\overset{\mathscr{L}_{\text{yield}}}{\mapsto}$	$\lambda z.cz$	$o \to o$
d	o	$\overset{\mathscr{L}_{\text{yield}}}{\mapsto}$	$\lambda z.dz$	$o \to o$

From trees to strings

S	$o \to o$	$\overset{\mathscr{L}_{\text{yield}}}{\mapsto}$	$\lambda x.\lambda z.xz$	$(o \to o) \to (o \to o)$
$A^{(1)}$	$o \to o$	$\overset{\mathscr{L}_{\text{yield}}}{\mapsto}$	$\lambda x.\lambda z.xz$	$(o \to o) \to o \to o$
$A^{(3)}$	$o^3 \to o$	$\overset{\mathscr{L}_{\text{yield}}}{\mapsto}$	$\lambda x_1.\lambda x_2.\lambda x_3.\lambda z.x_1(x_2(x_3 z))$	$(o \to o)^3 \to o \to o$
ϵ	o	$\overset{\mathscr{L}_{\text{yield}}}{\mapsto}$	$\lambda z.z$	$o \to o$
a	o	$\overset{\mathscr{L}_{\text{yield}}}{\mapsto}$	$\lambda z.az$	$o \to o$
b	o	$\overset{\mathscr{L}_{\text{yield}}}{\mapsto}$	$\lambda z.bz$	$o \to o$
c	o	$\overset{\mathscr{L}_{\text{yield}}}{\mapsto}$	$\lambda z.cz$	$o \to o$
d	o	$\overset{\mathscr{L}_{\text{yield}}}{\mapsto}$	$\lambda z.dz$	$o \to o$

$$\mathscr{L}_{\text{yield}}(o) = o \to o$$

From trees to strings

$$
\begin{array}{lll}
S & o \to o & \overset{\mathscr{L}_{\text{yield}}}{\mapsto} \quad \lambda x.\lambda z.xz & (o \to o) \to (o \to o) \\
A^{(1)} & o \to o & \overset{\mathscr{L}_{\text{yield}}}{\mapsto} \quad \lambda x.\lambda z.xz & (o \to o) \to o \to o \\
A^{(3)} & o^3 \to o & \overset{\mathscr{L}_{\text{yield}}}{\mapsto} \quad \lambda x_1.\lambda x_2.\lambda x_3.\lambda z.x_1(x_2(x_3 z)) & (o \to o)^3 \to o \to o \\
\epsilon & o & \overset{\mathscr{L}_{\text{yield}}}{\mapsto} \quad \lambda z.z & o \to o \\
a & o & \overset{\mathscr{L}_{\text{yield}}}{\mapsto} \quad \lambda z.az & o \to o \\
b & o & \overset{\mathscr{L}_{\text{yield}}}{\mapsto} \quad \lambda z.bz & o \to o \\
c & o & \overset{\mathscr{L}_{\text{yield}}}{\mapsto} \quad \lambda z.cz & o \to o \\
d & o & \overset{\mathscr{L}_{\text{yield}}}{\mapsto} \quad \lambda z.dz & o \to o
\end{array}
$$

$$\mathscr{L}_{\text{yield}}(o) = o \to o$$

$$
\begin{aligned}
(\mathscr{L}_{\text{yield}} \circ \mathscr{L})(\mathsf{C}(\mathsf{DE})) &= \mathscr{L}_{\text{yield}}(S(A^{(3)}a(A^{(3)}b(A^{(1)}\epsilon)c)d)) \\
&= \lambda z.a(b(c(dz))) \\
&= /abcd/
\end{aligned}
$$

Tree- and string-generating ACGs

$$\Sigma = \{\mathsf{C} : p \to s,\ \mathsf{D} : p \to p,\ \mathsf{E} : p\}$$

$$\Sigma' = \{S : o \to o,\ A^{(1)} : o \to o,\ A^{(3)} : o^3 \to o,\ \epsilon : o,\ a : o,\ b : o,\ c : o,\ d : o\}$$

$$\Sigma'' = \{a : o \to o,\ b : o \to o,\ c : o \to o,\ d : o \to o\}$$

$$\mathscr{L} = \{\mathsf{C} \mapsto \lambda y.S(y(A^{(1)}\epsilon)),\ \mathsf{D} \mapsto \lambda y.\lambda x.A^{(3)}a(y(A^{(3)}bxc)d,\ \mathsf{E} \mapsto \lambda x.x\}$$

$\mathscr{L}(p) = o \to o$

$\mathscr{L}(s) = o$

$$\mathscr{L}_{\text{yield}} = \{S \mapsto \lambda x.\lambda z.xz,\ A^{(1)} \mapsto \lambda x.\lambda z.xz,\ A^{(3)} \mapsto \lambda x_1.\lambda x_2.\lambda x_3.\lambda z.x_1(x_2(x_3 z)),\ \epsilon \mapsto \lambda z.z,\ a \mapsto \lambda z.az,\ b \mapsto \lambda z.bz,\ c \mapsto \lambda z.cz,\ d \mapsto \lambda z.dz\}$$

$\mathscr{L}_{\text{yield}}(o) = o \to o$

$$\mathscr{G} = (\Sigma, \Sigma', \mathscr{L}, s)$$

$$\mathscr{G}' = (\Sigma, \Sigma'', \mathscr{L}_{\text{yield}} \circ \mathscr{L}, s)$$

Tree- and string-generating ACGs

$$\Sigma = \{\mathsf{C} : p \to s,\ \mathsf{D} : p \to p,\ \mathsf{E} : p\}$$

$$\Sigma' = \{S : o \to o,\ A^{(1)} : o \to o,\ A^{(3)} : o^3 \to o,\ \epsilon : o,\ a : o,\ b : o,\ c : o,\ d : o\}$$

$$\Sigma'' = \{a : o \to o,\ b : o \to o,\ c : o \to o,\ d : o \to o\}$$

$$\mathscr{L} = \{\mathsf{C} \mapsto \lambda y.S(y(A^{(1)}\epsilon)),\ \mathsf{D} \mapsto \lambda y.\lambda x.A^{(3)}a(y(A^{(3)}bxc)d,\ \mathsf{E} \mapsto \lambda x.x\}$$

$$\mathscr{L}_{\text{yield}} = \{S \mapsto \lambda x.\lambda z.xz,\ A^{(1)} \mapsto \lambda x.\lambda z.xz,\ A^{(3)} \mapsto \lambda x_1.\lambda x_2.\lambda x_3.\lambda z.x_1(x_2(x_3 z)),\ \epsilon \mapsto \lambda z.z,\ a \mapsto \lambda z.az,\ b \mapsto \lambda z.bz,\ c \mapsto \lambda z.cz,\ d \mapsto \lambda z.dz\}$$

$\mathscr{L}(p) = o \to o$

$\mathscr{L}(s) = o$

$\mathscr{L}_{\text{yield}}(o) = o \to o$

$$\mathscr{G} = (\Sigma, \Sigma', \mathscr{L}, s)$$

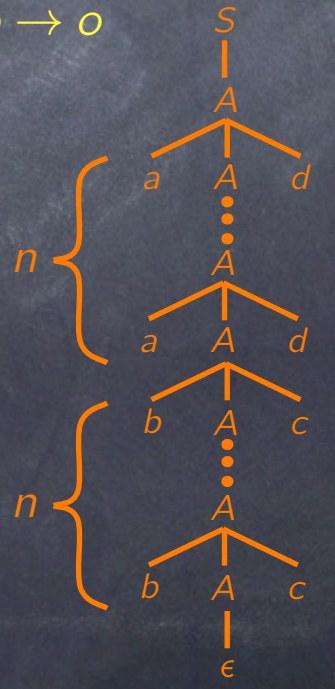
$$\mathscr{G}' = (\Sigma, \Sigma'', \mathscr{L}_{\text{yield}} \circ \mathscr{L}, s)$$

$$\mathcal{A}(\mathscr{G}) = \mathcal{A}(\mathscr{G}') = \{\mathsf{C}(\mathsf{D}^n\mathsf{E}) \mid n \geq 0\}$$

$$\mathcal{O}(\mathscr{G}) = \{T_n \mid n \geq 0\}$$

$$\mathcal{O}(\mathscr{G}') = \{/a^n b^n c^n d^n/ \mid n \geq 0\}$$

$T_n =$



Tree- and string-generating ACGs

$$\Sigma = \{\mathsf{C} : p \to s,\ \mathsf{D} : p \to p,\ \mathsf{E} : p\}$$

$$\Sigma' = \{S : o \to o,\ A^{(1)} : o \to o,\ A^{(3)} : o^3 \to o,\ \epsilon : o,\ a : o,\ b : o,\ c : o,\ d : o\}$$

second-order types

$$\Sigma'' = \{a : o \to o,\ b : o \to o,\ c : o \to o,\ d : o \to o\}$$

$$\mathscr{L} = \{\mathsf{C} \mapsto \lambda y.S(y(A^{(1)}\epsilon)),\ \mathsf{D} \mapsto \lambda y.\lambda x.A^{(3)}a(y(A^{(3)}bxc)d,\ \mathsf{E} \mapsto \lambda x.x\}$$

$$\mathscr{L}_{\text{yield}} = \{S \mapsto \lambda x.\lambda z.xz,\ A^{(1)} \mapsto \lambda x.\lambda z.xz,\ A^{(3)} \mapsto \lambda x_1.\lambda x_2.\lambda x_3.\lambda z.x_1(x_2(x_3 z)),\ \epsilon \mapsto \lambda z.z,\ a \mapsto \lambda z.az,\ b \mapsto \lambda z.bz,\ c \mapsto \lambda z.cz,\ d \mapsto \lambda z.dz\}$$

$\mathscr{L}(p) = o \to o$

$\mathscr{L}(s) = o$

$\mathscr{L}_{\text{yield}}(o) = o \to o$

$$\mathscr{G} = (\Sigma, \Sigma', \mathscr{L}, s)$$

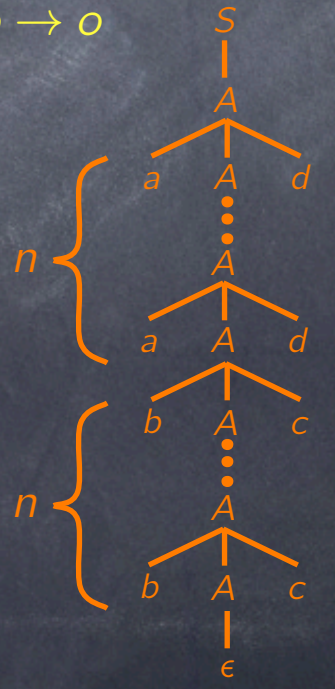
$$\mathscr{G}' = (\Sigma, \Sigma'', \mathscr{L}_{\text{yield}} \circ \mathscr{L}, s)$$

$$\mathcal{A}(\mathscr{G}) = \mathcal{A}(\mathscr{G}') = \{\mathsf{C}(\mathsf{D}^n\mathsf{E}) \mid n \geq 0\}$$

$$\mathcal{O}(\mathscr{G}) = \{T_n \mid n \geq 0\}$$

$$\mathcal{O}(\mathscr{G}') = \{/a^n b^n c^n d^n/ \mid n \geq 0\}$$

$T_n =$



Tree- and string-generating ACGs

$$\Sigma = \{\mathsf{C} : p \to s,\ \mathsf{D} : p \to p,\ \mathsf{E} : p\}$$

$$\Sigma' = \{S : o \to o,\ A^{(1)} : o \to o,\ A^{(3)} : o^3 \to o,\ \epsilon : o,\ a : o,\ b : o,\ c : o,\ d : o\}$$

second-order types

$$\Sigma'' = \{a : o \to o,\ b : o \to o,\ c : o \to o,\ d : o \to o\}$$

$$\mathscr{L} = \{\mathsf{C} \mapsto \lambda y.S(y(A^{(1)}\epsilon)),\ \mathsf{D} \mapsto \lambda y.\lambda x.A^{(3)}a(y(A^{(3)}bxc)d,\ \mathsf{E} \mapsto \lambda x.x\}$$

$$\mathscr{L}_{\text{yield}} = \{S \mapsto \lambda x.\lambda z.xz,\ A^{(1)} \mapsto \lambda x.\lambda z.xz,\ A^{(3)} \mapsto \lambda x_1.\lambda x_2.\lambda x_3.\lambda z.x_1(x_2(x_3 z)),\ \epsilon \mapsto \lambda z.z,\ a \mapsto \lambda z.az,\ b \mapsto \lambda z.bz,\ c \mapsto \lambda z.cz,\ d \mapsto \lambda z.dz\}$$

$\mathscr{L}(p) = o \to o$

$\mathscr{L}(s) = o$

$\mathscr{L}_{\text{yield}}(o) = o \to o$

$\mathscr{G} = (\Sigma, \Sigma', \mathscr{L}, s)$ 2nd-order, width 2

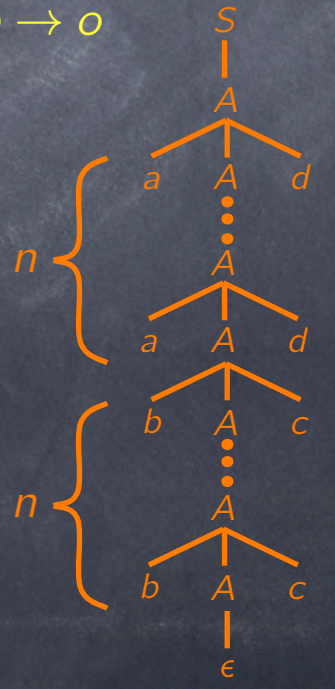
$\mathscr{G}' = (\Sigma, \Sigma'', \mathscr{L}_{\text{yield}} \circ \mathscr{L}, s)$ 2nd-order, width 4

$$\mathcal{A}(\mathscr{G}) = \mathcal{A}(\mathscr{G}') = \{\mathsf{C}(\mathsf{D}^n\mathsf{E}) \mid n \geq 0\}$$

$$\mathcal{O}(\mathscr{G}) = \{T_n \mid n \geq 0\}$$

$$\mathcal{O}(\mathscr{G}') = \{/a^n b^n c^n d^n/ \mid n \geq 0\}$$

$T_n =$



Second-order ACG hierarchy

- G is n-th order $\Leftrightarrow \max_{c \in C} \text{order}(\tau(c)) \leq n$
- G is of width m $\Leftrightarrow \max_{p \in A} |\mathscr{L}(p)| \leq m$

Second-order ACG hierarchy

- G is n-th order $\Leftrightarrow \max_{c \in C} \mathrm{order}(\tau(c)) \leq n$
- G is of width m $\Leftrightarrow \max_{p \in A} |\mathscr{L}(p)| \leq m$

# of atomic type occurrences

$\mathbf{A}_m$: the class of second-order ACGs of width m

Second-order ACG hierarchy

- G is n-th order $\Leftrightarrow \max_{c \in C} \mathrm{order}(\tau(c)) \leq n$
- G is of width m $\Leftrightarrow \max_{p \in A} |\mathscr{L}(p)| \leq m$

# of atomic type occurrences

$\mathbf{A}_m$: the class of second-order ACGs of width m

string generating power

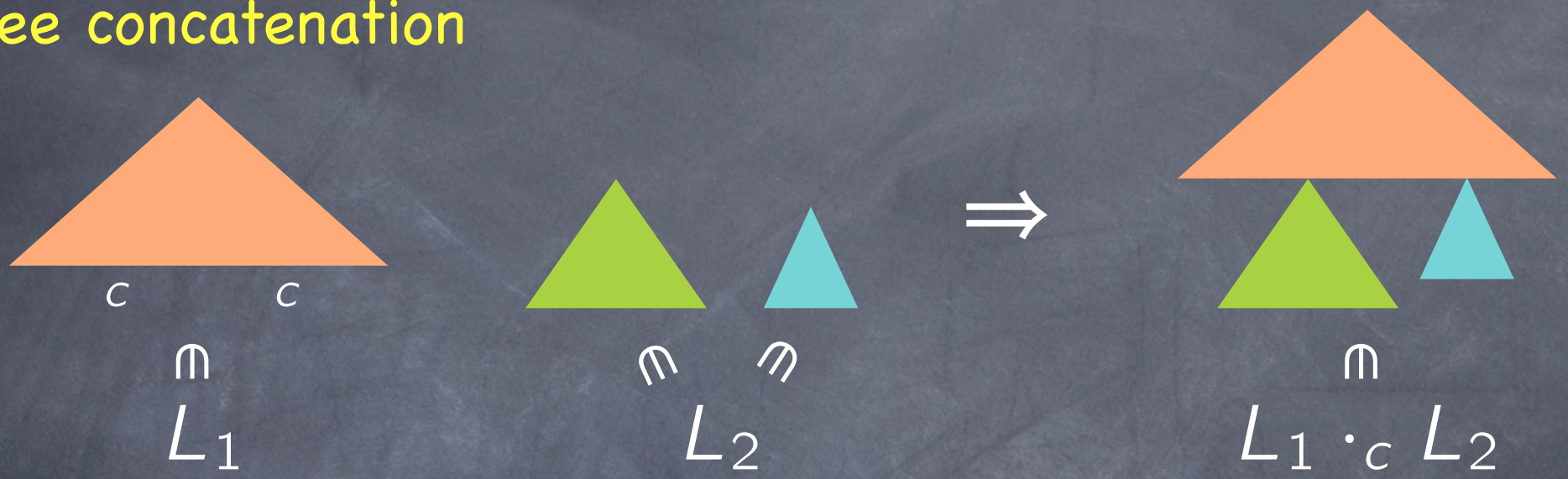
- $STR(\mathbf{A}_m)$ $(m \geq 2)$ is a substitution-closed full AFL (Kanazawa 2006)

tree generating power

- $TR(\mathbf{A}_m)$ is closed under union, tree concatenation, linear non-deleting tree homomorphism, intersection with regular tree languages, and Kleene star (Kanazawa 2006)

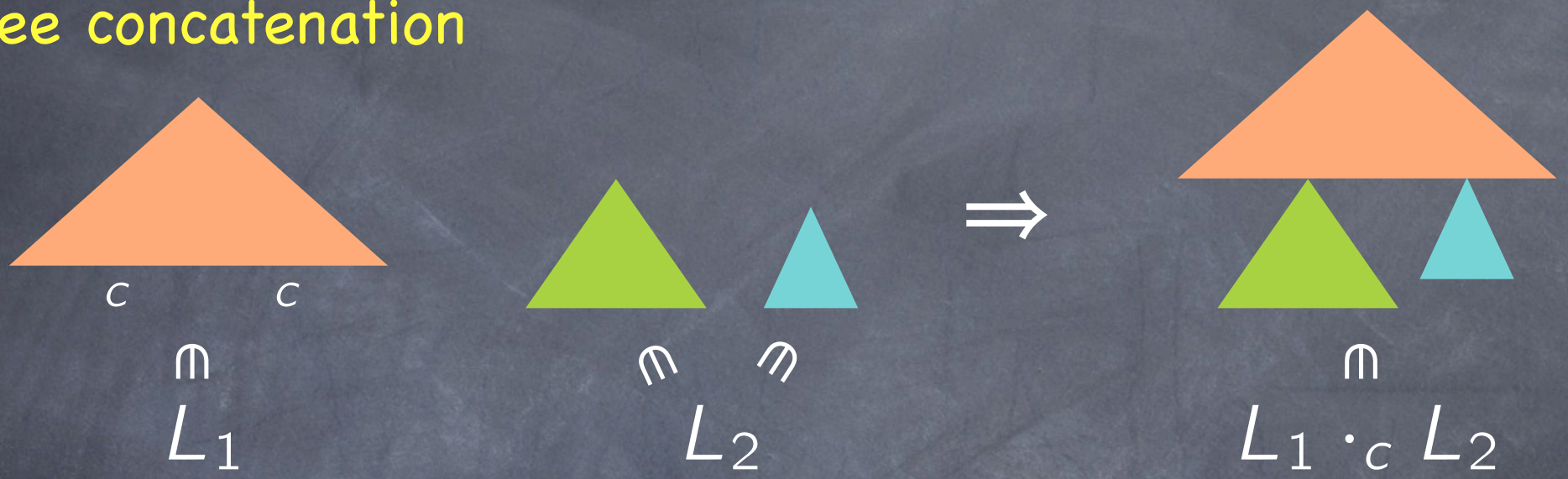
Tree language operations

tree concatenation



Tree language operations

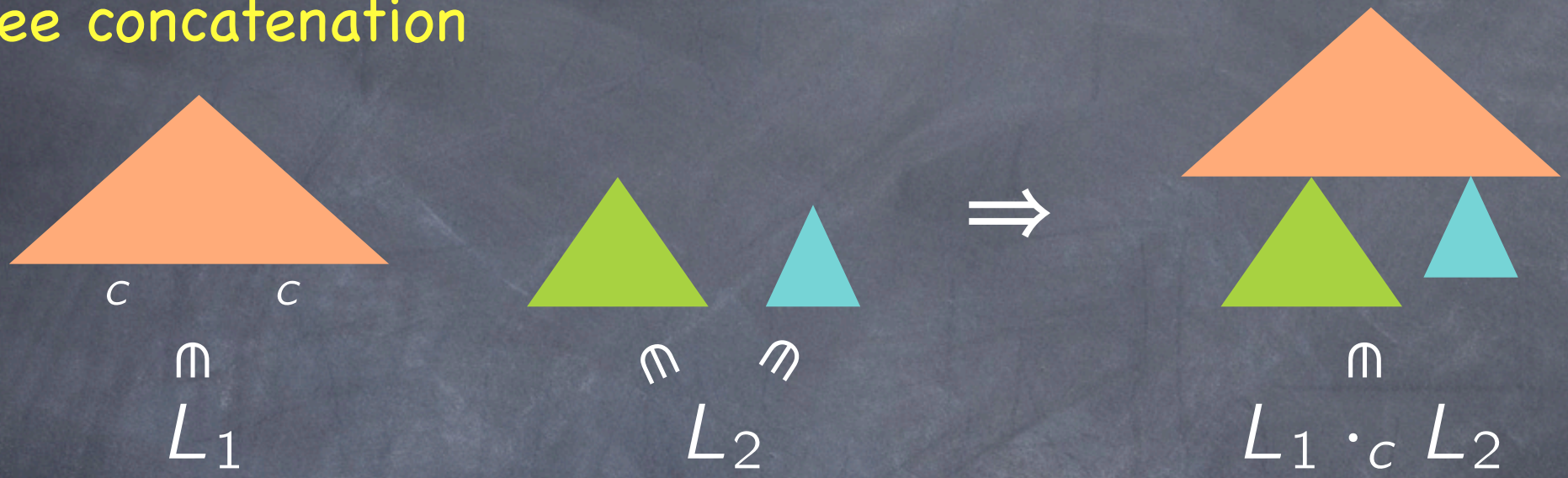
tree concatenation



Kleene star $L^{*,c} = \{c\} \cup L \cdot_c L^{*,c}$

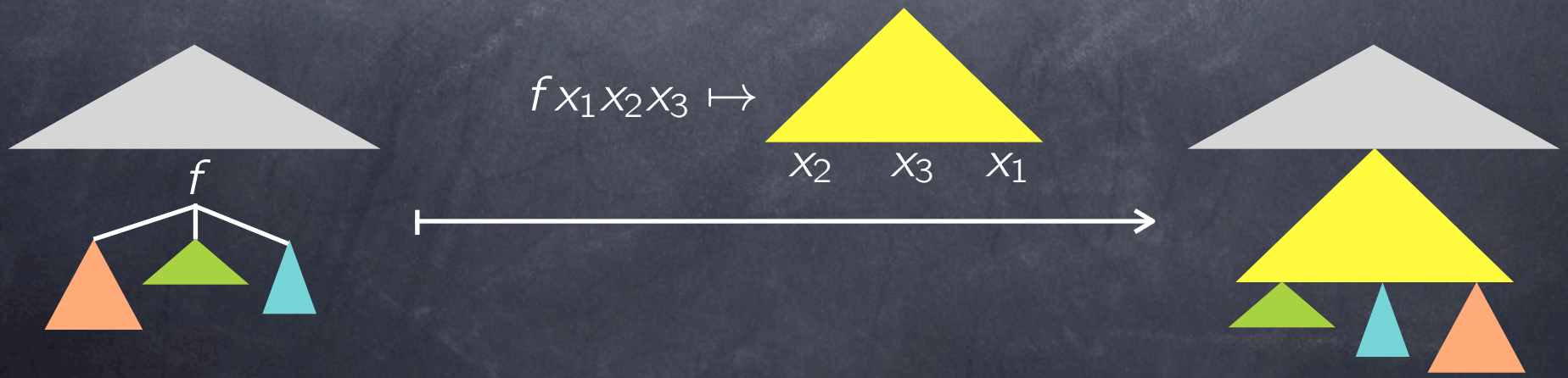
Tree language operations

tree concatenation



Kleene star $L^{*,c} = \{c\} \cup L \cdot_c L^{*,c}$

linear non-deleting tree homomorphism



From control sets to 2nd-order ACGs

$$C \in STR(\mathbf{A}_m) \Rightarrow \ulcorner C \urcorner \in TR(\mathbf{A}_m)$$

$= \{\ulcorner w \urcorner \mid w \in C\}$

spine as monadic tree

From control sets to 2nd-order ACGs

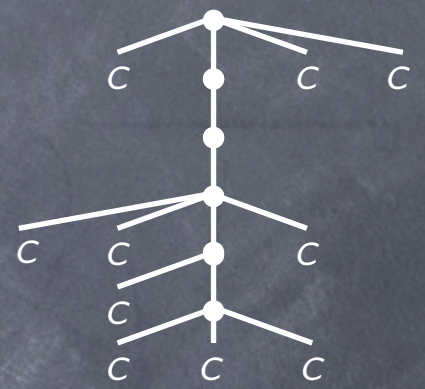
$= \{ \ulcorner w \urcorner \mid w \in C \}$

$C \in STR(\mathbf{A}_m) \Rightarrow \ulcorner C \urcorner \in TR(\mathbf{A}_m)$

spine as monadic tree

linear non-deleting tree homomorphism

$\pi \colon A \to B_1 \dots B_h \dots B_n \qquad \phi \colon \pi^{(1)} x \mapsto \pi c \dots c x c \dots c$



From control sets to 2nd-order ACGs

$= \{ \ulcorner w \urcorner \mid w \in C \}$

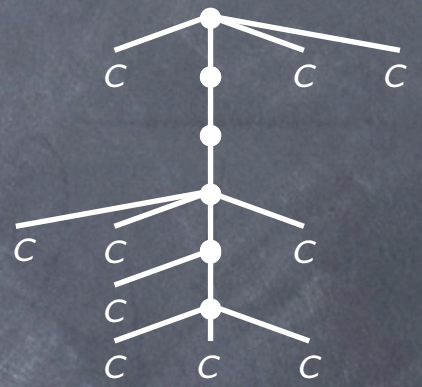
$C \in STR(\mathbf{A}_m) \Rightarrow \ulcorner C \urcorner \in TR(\mathbf{A}_m)$

spine as monadic tree

linear non-deleting tree homomorphism

$\pi \colon A \to B_1 \dots B_h \dots B_n \qquad \phi \colon \pi^{(1)} x \mapsto \pi c \dots c x c \dots c$

$\phi(\ulcorner C \urcorner) \in TR(\mathbf{A}_m)$



From control sets to 2nd-order ACGs

$= \{ \ulcorner w \urcorner \mid w \in C \}$

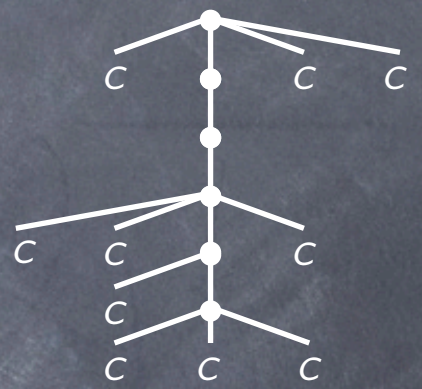
$C \in STR(\mathbf{A}_m) \Rightarrow \ulcorner C \urcorner \in TR(\mathbf{A}_m)$

spine as monadic tree

linear non-deleting tree homomorphism

$\pi \colon A \to B_1 \dots B_h \dots B_n \qquad \phi \colon \pi^{(1)} x \mapsto \pi c \dots c x c \dots c$

$\phi(\ulcorner C \urcorner) \in TR(\mathbf{A}_m)$



$TR(\mathbf{A}_m)$

$\cup\!\!\!\!-$ regular

$DT(G, C) = \phi(\ulcorner C \urcorner)^{*,c} \cap DT(G) \in TR(\mathbf{A}_m)$

From control sets to 2nd-order ACGs

$= \{ \ulcorner w \urcorner \mid w \in C \}$

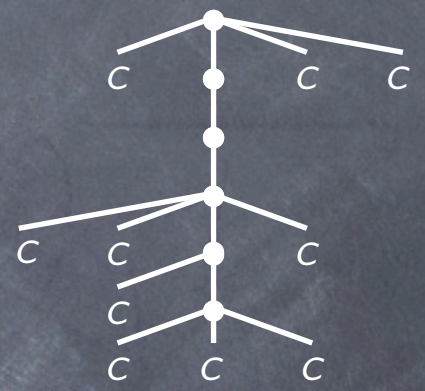
$C \in STR(\mathbf{A}_m) \Rightarrow \ulcorner C \urcorner \in TR(\mathbf{A}_m)$

spine as monadic tree

linear non-deleting tree homomorphism

$\pi \colon A \to B_1 \dots B_h \dots B_n \qquad \phi \colon \pi^{(1)} x \mapsto \pi c \dots c x c \dots c$

$\phi(\ulcorner C \urcorner) \in TR(\mathbf{A}_m)$



$TR(\mathbf{A}_m)$

$\cup$

regular

$DT(G, C) = \phi(\ulcorner C \urcorner)^{*,c} \cap DT(G) \in TR(\mathbf{A}_m)$

$L(G, C) = \{ \mathrm{yield}(T) \mid T \in DT(G, C) \} \in STR(\mathbf{A}_{2m})$

From control sets to 2nd-order ACGs

$= \{ \ulcorner w \urcorner \mid w \in C \}$

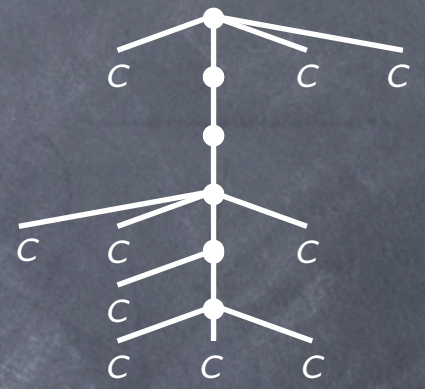
$C \in STR(\mathbf{A}_m) \Rightarrow \ulcorner C \urcorner \in TR(\mathbf{A}_m)$

spine as monadic tree

linear non-deleting tree homomorphism

$\pi \colon A \to B_1 \dots B_h \dots B_n \qquad \phi \colon \pi^{(1)} x \mapsto \pi c \dots c x c \dots c$

$\phi(\ulcorner C \urcorner) \in TR(\mathbf{A}_m)$



$TR(\mathbf{A}_m)$

$\cup$

regular

$DT(G, C) = \phi(\ulcorner C \urcorner)^{*,c} \cap DT(G) \in TR(\mathbf{A}_m)$

$L(G, C) = \{ \mathsf{yield}(T) \mid T \in DT(G, C) \} \in STR(\mathbf{A}_{2m})$

$C \in STR(\mathbf{A}_m) \Rightarrow L(G, C) \in STR(\mathbf{A}_{2m})$

From control sets to 2nd-order ACGs

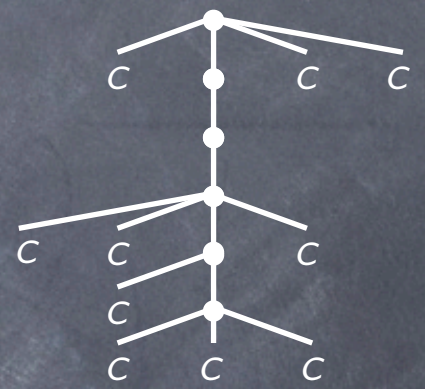
$= \{ \ulcorner w \urcorner \mid w \in C \}$

$C \in STR(\mathbf{A}_m) \Rightarrow \ulcorner C \urcorner \in TR(\mathbf{A}_m)$

spine as monadic tree

linear non-deleting tree homomorphism

$\pi \colon A \to B_1 \dots B_h \dots B_n \qquad \phi \colon \pi^{(1)} x \mapsto \pi c \dots c x c \dots c$



$\phi(\ulcorner C \urcorner) \in TR(\mathbf{A}_m)$

$TR(\mathbf{A}_m)$ regular

$DT(G, C) = \phi(\ulcorner C \urcorner)^{*,c} \cap DT(G) \in TR(\mathbf{A}_m)$

$L(G, C) = \{ \mathsf{yield}(T) \mid T \in DT(G, C) \} \in STR(\mathbf{A}_{2m})$

$C \in STR(\mathbf{A}_m) \Rightarrow L(G, C) \in STR(\mathbf{A}_{2m})$

$\mathbf{C}_1 = \mathsf{CFL} = STR(\mathbf{A}_2) \qquad \mathbf{C}_k \subseteq STR(\mathbf{A}_{2^k})$

From 2nd-order ACGs to MCFGs

C: $p \to s$ $\quad \lambda y.y(\lambda z.z)$
D: $p \to p$ $\quad \lambda yxz.a(y(\lambda z.b(x(cz)))(dz))$
E: p $\quad \lambda xz.xz$

$\mathsf{D}(\mathsf{DE}) : p$ $\quad \lambda xz.a(a(b(b(x(c(c(d(dz))))))))$

$\mathsf{D} : p \to p$ $\quad \mathsf{DE} : p$ $\quad \lambda xz.a(b(x(c(dz))))$

$\mathsf{D} : p \to p$ $\quad \mathsf{E} : p$ $\quad \lambda xz.xz$

From 2nd-order ACGs to MCFGs

C : $p \to s$ $\quad \lambda y.y(\lambda z.z)$

D : $p \to p$ $\quad \lambda yxz.a(y(\lambda z.b(x(cz)))(dz))$

E : p $\quad \lambda xz.xz$

tuple of strings

$(aabb, ccdd)$

D(DE) : p $\quad \lambda xz.a(a(b(b(x(c(c(d(dz))))))))$

D : $p \to p$ $\quad$ DE : p $\quad \lambda xz.a(b(x(c(dz))))$

D : $p \to p$ $\quad$ E : p $\quad \lambda xz.xz$

From 2nd-order ACGs to MCFGs

C : $p \to s$ $\quad \lambda y.y(\lambda z.z)$

D : $p \to p$ $\quad \lambda yxz.a(y(\lambda z.b(x(cz)))(dz))$

E : p $\quad \lambda xz.xz$

D(DE) : p $\quad \lambda xz.a(a(b(b(x(c(c(d(dz))))))))$

tuple of strings

$(aabb, ccdd)$

$\lambda xz.z_1(x(z_2z))$

pure λ-term

D : $p \to p$ $\quad$ DE : p $\quad \lambda xz.a(b(x(c(dz))))$

D : $p \to p$ $\quad$ E : p $\quad \lambda xz.xz$

From 2nd-order ACGs to MCFGs

C : $p \to s$ $\quad \lambda y.y(\lambda z.z)$

D : $p \to p$ $\quad \lambda yxz.a(y(\lambda z.b(x(cz)))(dz))$

E : p $\quad \lambda xz.xz$

D(DE) : p $\quad \lambda xz.a(a(b(b(x(c(c(d(dz))))))))$

D : $p \to p$ $\quad$ DE : p $\quad \lambda xz.a(b(x(c(dz))))$

D : $p \to p$ $\quad$ E : p $\quad \lambda xz.xz$

tuple of strings

$(aabb, ccdd)$

(ab, cd)

$\lambda xz.z_1(x(z_2z))$

pure λ-term

From 2nd-order ACGs to MCFGs

C : $p \to s$ $\quad \lambda y.y(\lambda z.z)$

D : $p \to p$ $\quad \lambda yxz.a(y(\lambda z.b(x(cz)))(dz))$

E : p $\quad \lambda xz.xz$

D(DE) : p $\quad \lambda xz.a(a(b(b(x(c(c(d(dz))))))))$

D : $p \to p$ $\quad$ DE : p $\quad \lambda xz.a(b(x(c(dz))))$

D : $p \to p$ $\quad$ E : p $\quad \lambda xz.xz$

tuple of strings

$(aabb, ccdd)$

$\lambda xz.z_1(x(z_2z))$

pure λ-term

(ab, cd)

$\lambda xz.z_1(x(z_2z))$

From 2nd-order ACGs to MCFGs

$\mathsf{C} : p \to s \quad \lambda y.y(\lambda z.z)$

$\mathsf{D} : p \to p \quad \lambda yxz.a(y(\lambda z.b(x(cz)))(dz))$

$\mathsf{E} : p \quad \lambda xz.xz$

tuple of strings

$\mathsf{D}(\mathsf{DE}) : p \quad \lambda xz.a(a(b(b(x(c(c(d(dz)))))))) \quad (aabb, ccdd)$

$\lambda xz.z_1(x(z_2z))$

pure λ-term

$\mathsf{D} : p \to p \quad \mathsf{DE} : p \quad \lambda xz.a(b(x(c(dz)))) \quad (ab, cd)$

$\lambda xz.z_1(x(z_2z))$

$\mathsf{D} : p \to p \quad \mathsf{E} : p \quad \lambda xz.xz$

$\mathsf{D}N : p \quad \mathscr{L}(\mathsf{D}N) \quad (ax_1b, cx_2d)$

$Q[z_1, z_2]$

$\mathsf{D} : p \to p \quad N : p \quad \mathscr{L}(N) \quad (x_1, x_2)$

$P[z_1, z_2]$

From 2nd-order ACGs to MCFGs

$\mathsf{C} : p \to s \quad \lambda y.y(\lambda z.z)$

$\mathsf{D} : p \to p \quad \lambda yxz.a(y(\lambda z.b(x(cz)))(dz))$

$\mathsf{E} : p \quad \lambda xz.xz$

tuple of strings

$\mathsf{D}(\mathsf{DE}) : p \quad \lambda xz.a(a(b(b(x(c(c(d(dz)))))))) \to (aabb, ccdd)$

$\lambda xz.z_1(x(z_2z))$

pure λ-term

$\mathsf{D} : p \to p \quad \mathsf{DE} : p \quad \lambda xz.a(b(x(c(dz)))) \to (ab, cd)$

$\lambda xz.z_1(x(z_2z))$

$\mathsf{D} : p \to p \quad \mathsf{E} : p \quad \lambda xz.xz$

$\mathsf{D}N : p \quad \mathscr{L}(\mathsf{D})P[x_1, x_2] \to (ax_1b, cx_2d)$

$Q[z_1, z_2]$

$\mathsf{D} : p \to p \quad N : p \to (x_1, x_2)$

$P[z_1, z_2]$

From 2nd-order ACGs to MCFGs

$\mathsf{C} : p \to s \quad \lambda y.y(\lambda z.z)$
$\mathsf{D} : p \to p \quad \lambda yxz.a(y(\lambda z.b(x(cz)))(dz))$
$\mathsf{E} : p \quad \lambda xz.xz$

tuple of strings

$\mathsf{D}(\mathsf{DE}) : p \quad \lambda xz.a(a(b(b(x(c(c(d(dz)))))))) \to (aabb, ccdd)$

$\lambda xz.z_1(x(z_2z))$

pure λ-term

$\mathsf{D} : p \to p \quad \mathsf{DE} : p \quad \lambda xz.a(b(x(c(dz)))) \to (ab, cd)$

$\lambda xz.z_1(x(z_2z))$

$\mathsf{D} : p \to p \quad \mathsf{E} : p \quad \lambda xz.xz$

$\mathsf{D}N : p \quad \mathscr{L}(\mathsf{D})P[x_1, x_2] \to (ax_1b, cx_2d)$

$Q[z_1, z_2]$

$\mathsf{D} : p \to p \quad N : p \to (x_1, x_2)$

$P[z_1, z_2]$

MCFG rule

$(p, Q[z_1, z_2])(ax_1b, cx_2d) :- (p, P[z_1, z_2])(x_1, x_2)$

Relating width with tuple length

$$z_1 : o \to o,\ z_2 : o \to o \vdash \lambda xz.z_1(x(z_2 z)) : (o \to o) \to o \to o$$

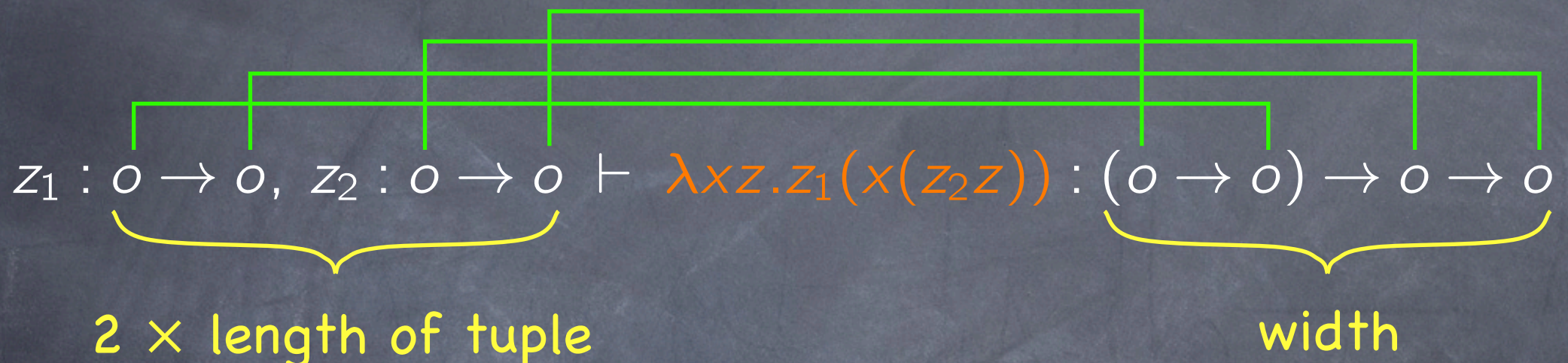
Relating width with tuple length

$$z_1 : \underbrace{o \to o,\ z_2 : o \to o}_{2 \times \text{length of tuple}} \vdash \lambda x z.z_1(x(z_2 z)) : (o \to o) \to o \to o$$

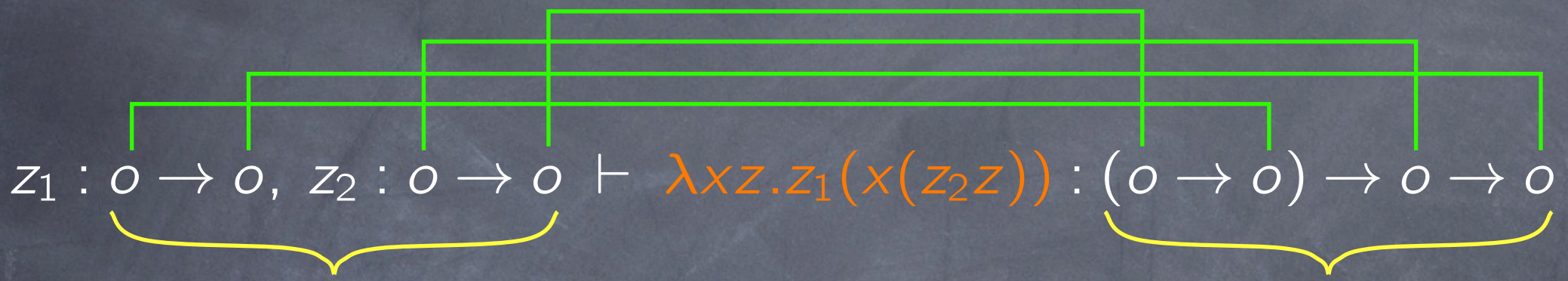
Relating width with tuple length

$$z_1 : \underbrace{o \to o,\ z_2 : o \to o}_{2 \times \text{length of tuple}} \vdash \lambda x z.z_1(x(z_2 z)) : \underbrace{(o \to o) \to o \to o}_{\text{width}}$$

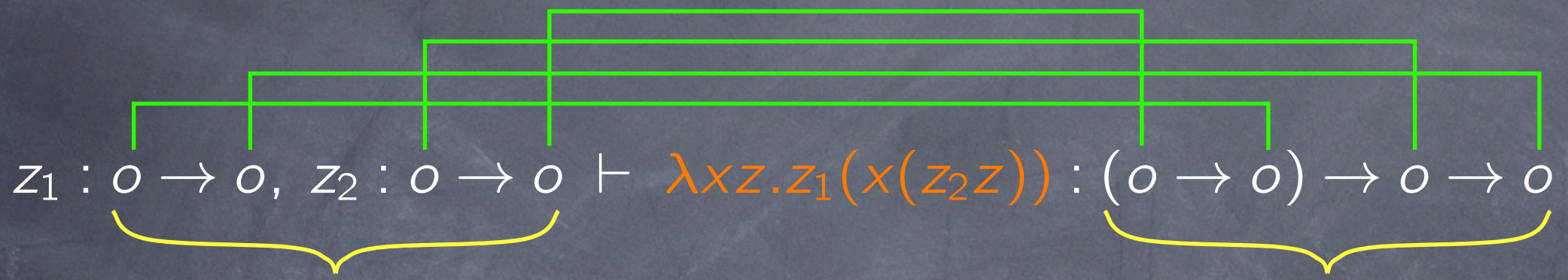
Relating width with tuple length



Relating width with tuple length

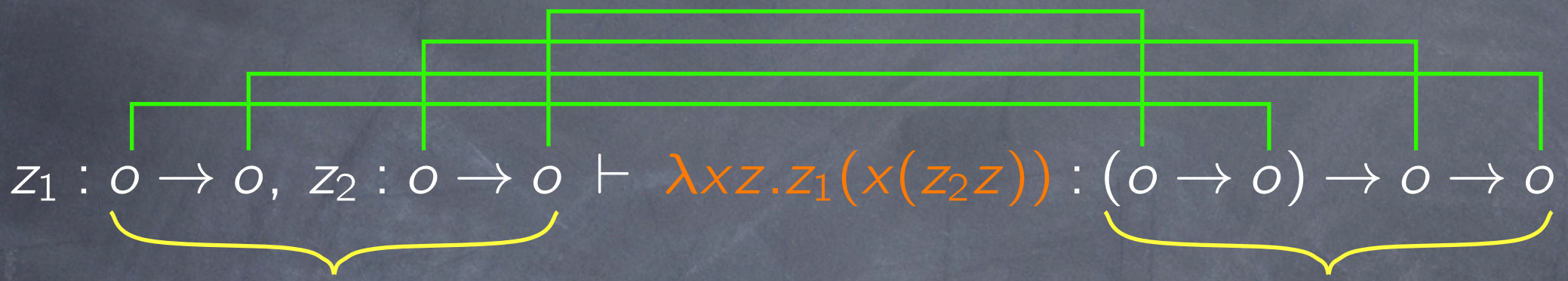


Relating width with tuple length



$$STR(\mathbf{A}_{2m}) \subseteq m\text{-MCFL}$$

Relating width with tuple length



$$STR(\mathbf{A}_{2m}) \subseteq m\text{-MCFL}$$

- $\bigcup_m STR(\mathbf{A}_m) \subseteq$ MCFL was shown by Salvati at FG 2006
- m-MCFL $\subseteq STR(\mathbf{A}_{2m+2})$ was shown by de Groote and Pogodalla (2004)

Separation

Seki et al. (1991):

$$\mathrm{RESP} = \{\, a_1^m a_2^m b_1^n b_2^n a_3^m a_4^m b_3^n b_4^n \mid m, n \geq 0 \,\} \in 2\text{-MCFL} - \mathbf{C}_2$$

Separation

Seki et al. (1991):

$$\mathrm{RESP} = \{\, a_1^m a_2^m b_1^n b_2^n a_3^m a_4^m b_3^n b_4^n \mid m, n \geq 0 \,\} \in \text{2-MCFL} - \mathbf{C}_2$$

$$\mathrm{RESP}_k = \{\, a_1^m a_2^m b_1^n b_2^n \ldots a_{2k-1}^m a_{2k}^m b_{2k-1}^n b_{2k}^n \mid m, n \geq 0 \,\}$$

Separation

Seki et al. (1991):

$$\mathrm{RESP} = \{\, a_1^m a_2^m b_1^n b_2^n a_3^m a_4^m b_3^n b_4^n \mid m, n \geq 0 \,\} \in 2\text{-MCFL} - \mathbf{C}_2$$

$$\mathrm{RESP}_k = \{\, a_1^m a_2^m b_1^n b_2^n \ldots a_{2k-1}^m a_{2k}^m b_{2k-1}^n b_{2k}^n \mid m, n \geq 0 \,\}$$

Easy: $\mathrm{RESP}_k \in k\text{-MCFL} \subseteq STR(\mathbf{A}_{2k+2})$

Separation

Seki et al. (1991):

$$\mathrm{RESP} = \{\, a_1^m a_2^m b_1^n b_2^n a_3^m a_4^m b_3^n b_4^n \mid m, n \geq 0 \,\} \in 2\text{-MCFL} - \mathbf{C}_2$$

$$\mathrm{RESP}_k = \{\, a_1^m a_2^m b_1^n b_2^n \ldots a_{2k-1}^m a_{2k}^m b_{2k-1}^n b_{2k}^n \mid m, n \geq 0 \,\}$$

Easy: $\mathrm{RESP}_k \in k\text{-MCFL} \subseteq STR(\mathbf{A}_{2k+2})$

But also: $\mathrm{RESP}_k \in STR(\mathbf{A}_{2k})$

Separation

Seki et al. (1991):

$$\mathrm{RESP} = \{\, a_1^m a_2^m b_1^n b_2^n a_3^m a_4^m b_3^n b_4^n \mid m, n \geq 0 \,\} \in 2\text{-MCFL} - \mathbf{C}_2$$

$$\mathrm{RESP}_k = \{\, a_1^m a_2^m b_1^n b_2^n \ldots a_{2k-1}^m a_{2k}^m b_{2k-1}^n b_{2k}^n \mid m, n \geq 0 \,\}$$

Easy: $\mathrm{RESP}_k \in k\text{-MCFL} \subseteq STR(\mathbf{A}_{2k+2})$

But also: $\mathrm{RESP}_k \in STR(\mathbf{A}_{2k})$

For k≥2: $\mathrm{RESP}_{2^{k-1}} \notin \mathbf{C}_k$ by Palis and Shende's Pumping Lemma

$$\mathrm{RESP}_{2^{k-1}} \in STR(\mathbf{A}_{2^k}) - \mathbf{C}_k$$

Separation

Seki et al. (1991):

$$\mathrm{RESP} = \{\, a_1^m a_2^m b_1^n b_2^n a_3^m a_4^m b_3^n b_4^n \mid m, n \geq 0 \,\} \in 2\text{-MCFL} - \mathbf{C}_2$$

$$\mathrm{RESP}_k = \{\, a_1^m a_2^m b_1^n b_2^n \ldots a_{2k-1}^m a_{2k}^m b_{2k-1}^n b_{2k}^n \mid m, n \geq 0 \,\}$$

Easy: $\mathrm{RESP}_k \in k\text{-MCFL} \subseteq STR(\mathbf{A}_{2k+2})$

But also: $\mathrm{RESP}_k \in STR(\mathbf{A}_{2k})$

For k≥2: $\mathrm{RESP}_{2^{k-1}} \notin \mathbf{C}_k$ by Palis and Shende's Pumping Lemma

$$\mathrm{RESP}_{2^{k-1}} \in STR(\mathbf{A}_{2^k}) - \mathbf{C}_k$$

$$\mathbf{C}_k \subsetneq STR(\mathbf{A}_{2^k}) \subseteq 2^{k-1}\text{-MCFL}$$

Questions

Questions

$$STR(\mathbf{A}_{2^k}) \stackrel{?}{\subsetneq} 2^{k-1}\text{-MCFL} \ (k \geq 2)$$

Questions

$$STR(\mathbf{A}_{2^k}) \overset{?}{\subsetneq} 2^{k-1}\text{-MCFL} \ (k \geq 2)$$

$$\bigcup_k \mathbf{C}_k \overset{?}{\subsetneq} \text{MCFL} \ \left(= \bigcup_m STR(\mathbf{A}_m)\right)$$

Questions

$$STR(\mathbf{A}_{2^k}) \overset{?}{\subsetneq} 2^{k-1}\text{-MCFL} \ (k \geq 2)$$

$$\bigcup_k \mathbf{C}_k \overset{?}{\subsetneq} \text{MCFL} \ \left(= \bigcup_m STR(\mathbf{A}_m)\right)$$

Note: $\text{RESP}_k \in \bigcup_k \mathbf{C}_k$