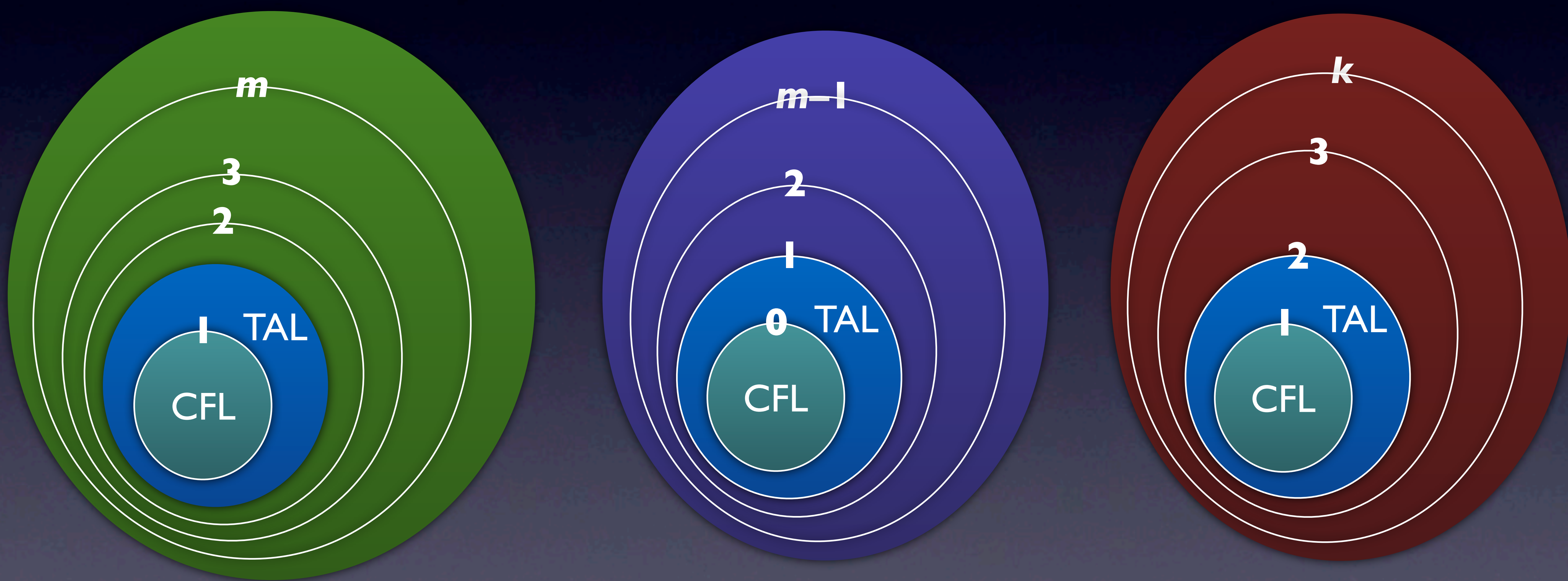
Two Subhierarchies Inside the MCFLs

Makoto Kanazawa
National Institute of Informatics

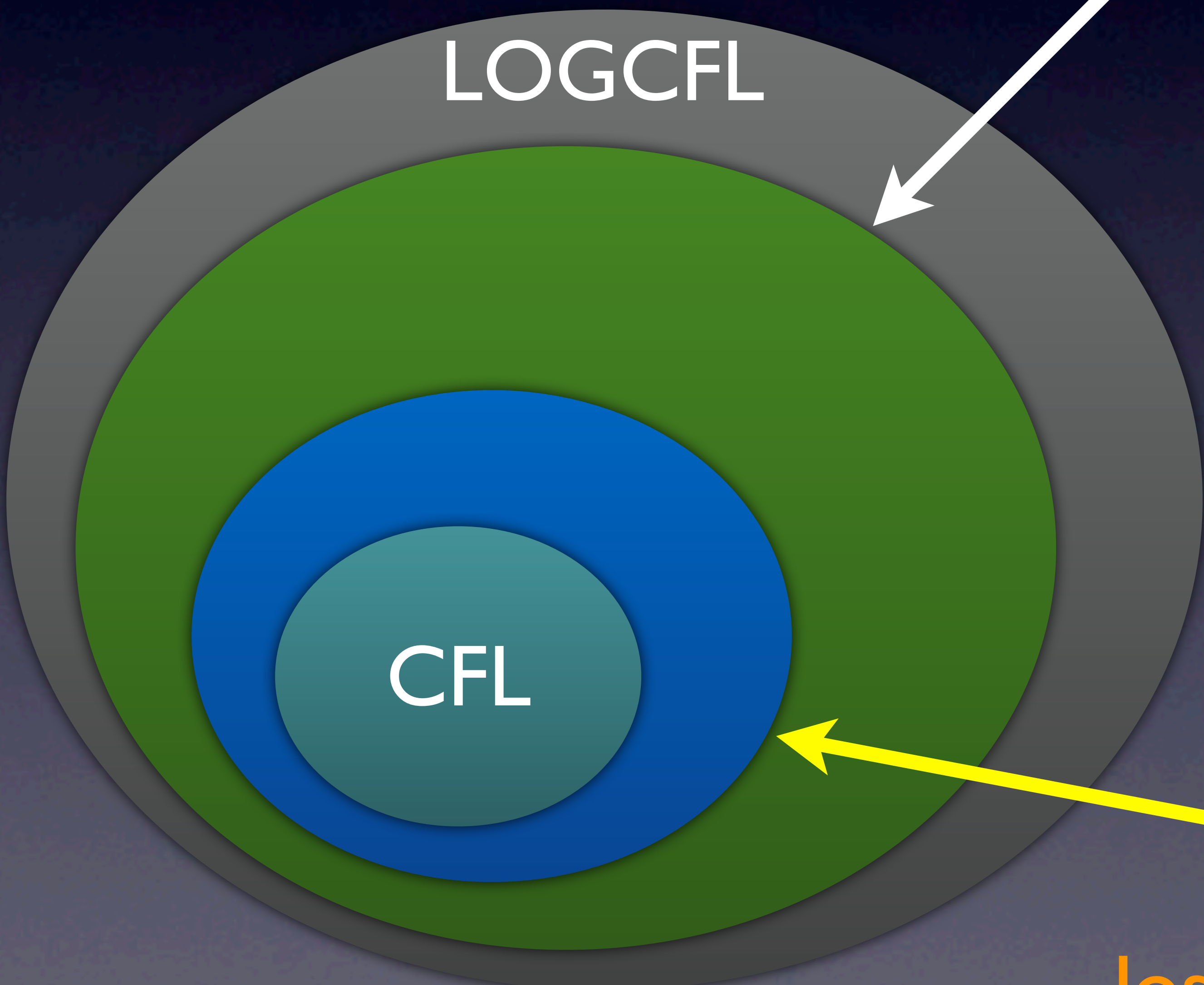
Three Infinite Hierarchies



$$\mathrm{MCFL} = \bigcup_{m \geq 1} m\text{-}\mathrm{MCFL} \qquad y\mathrm{CFT}_{\mathrm{sp}} = \bigcup_{m \geq 1} y\mathrm{CFT}_{\mathrm{sp}}(m-1) \qquad \mathbf{C} = \bigcup_{k \geq 1} \mathbf{C}_k$$

$y\mathrm{CFT}_{\mathrm{sp}}(1)$ and C_2 coincide with TAL, the class of languages generated by Tree Adjoining Grammars.

Convergence of Mildly Context-Sensitive Grammar Formalisms



$\mathrm{MCFG} \equiv \mathrm{MCTAG} \equiv \mathrm{HR}$

$\equiv \mathrm{OUT}(\mathrm{DTWT})$

$\equiv \mathrm{yDT}_{\mathrm{fc}}(\mathrm{REGT}) \equiv \mathrm{LUSCG}$

$\equiv \mathrm{MG} \equiv \mathrm{ACG}_2$

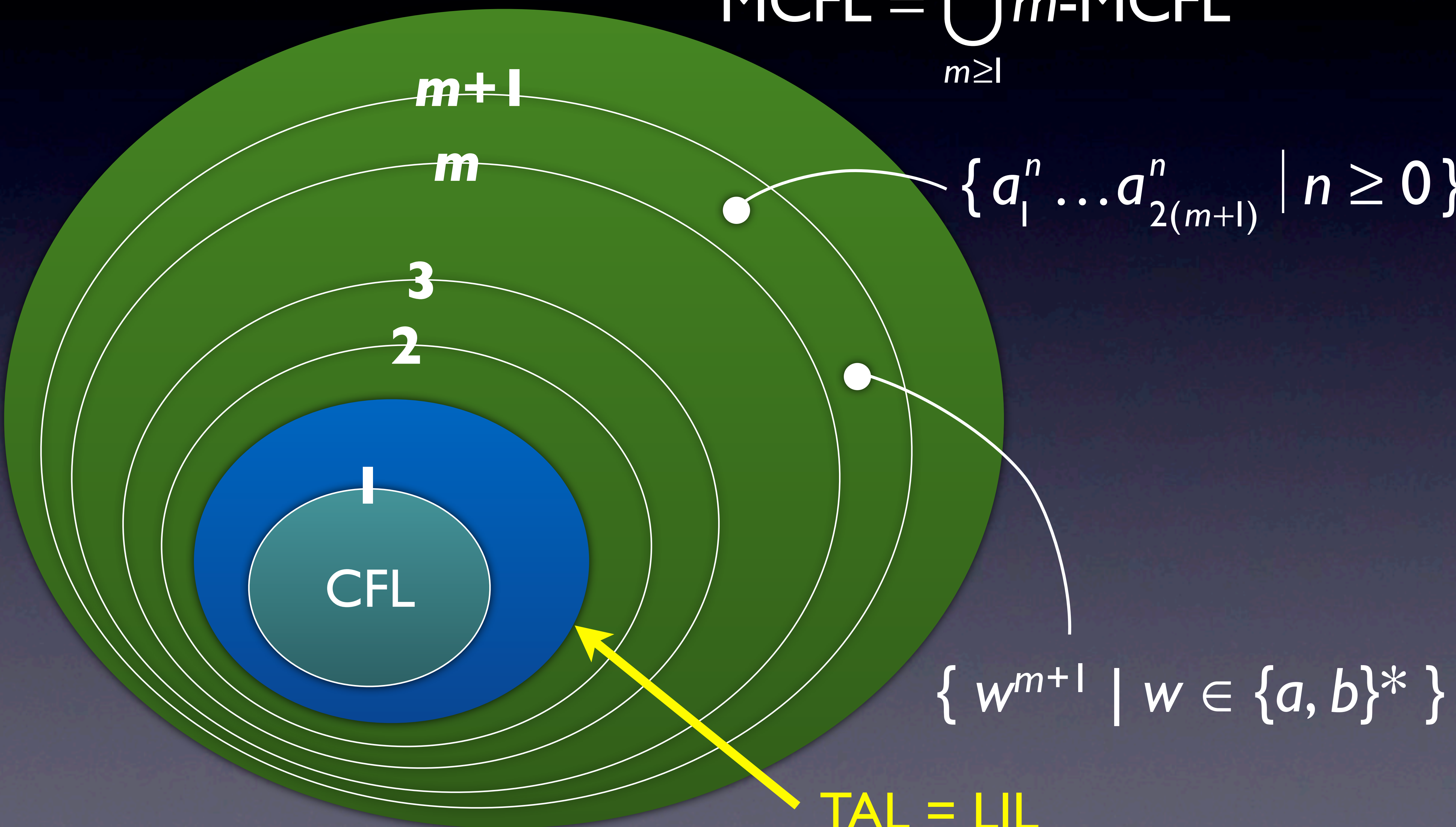
$\mathrm{TAG} \equiv \mathrm{CCG} \equiv \mathrm{LIG} \equiv \mathrm{HG}$

Joshi, Vijay-Shanker, and Weir 1991

The “convergence of mildly context-sensitive ...” originally referred to TAG, CCG, LIG, HG.
CCG is rather ad hoc. TAG and HG are not very different.

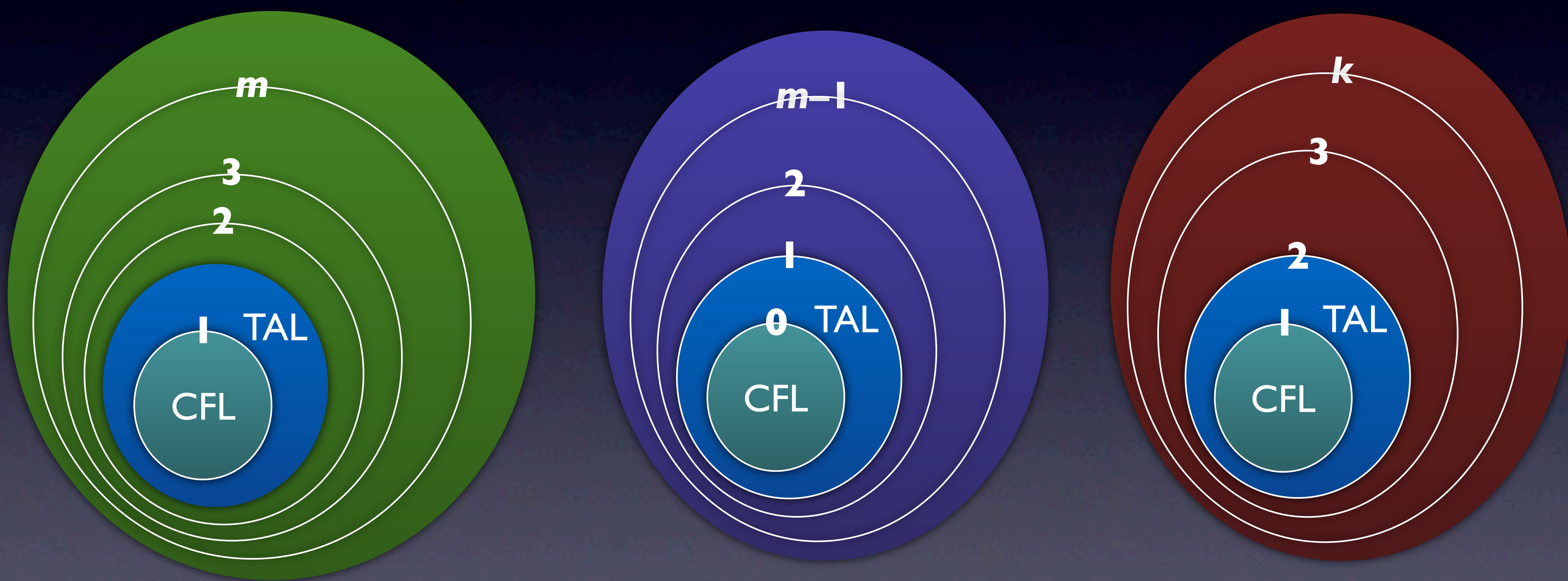
MCFL Hierarchy

$$\mathrm{MCFL} = \bigcup_{m \geq 1} m\text{-MCFL}$$



This is an infinite hierarchy.

Three Infinite Hierarchies



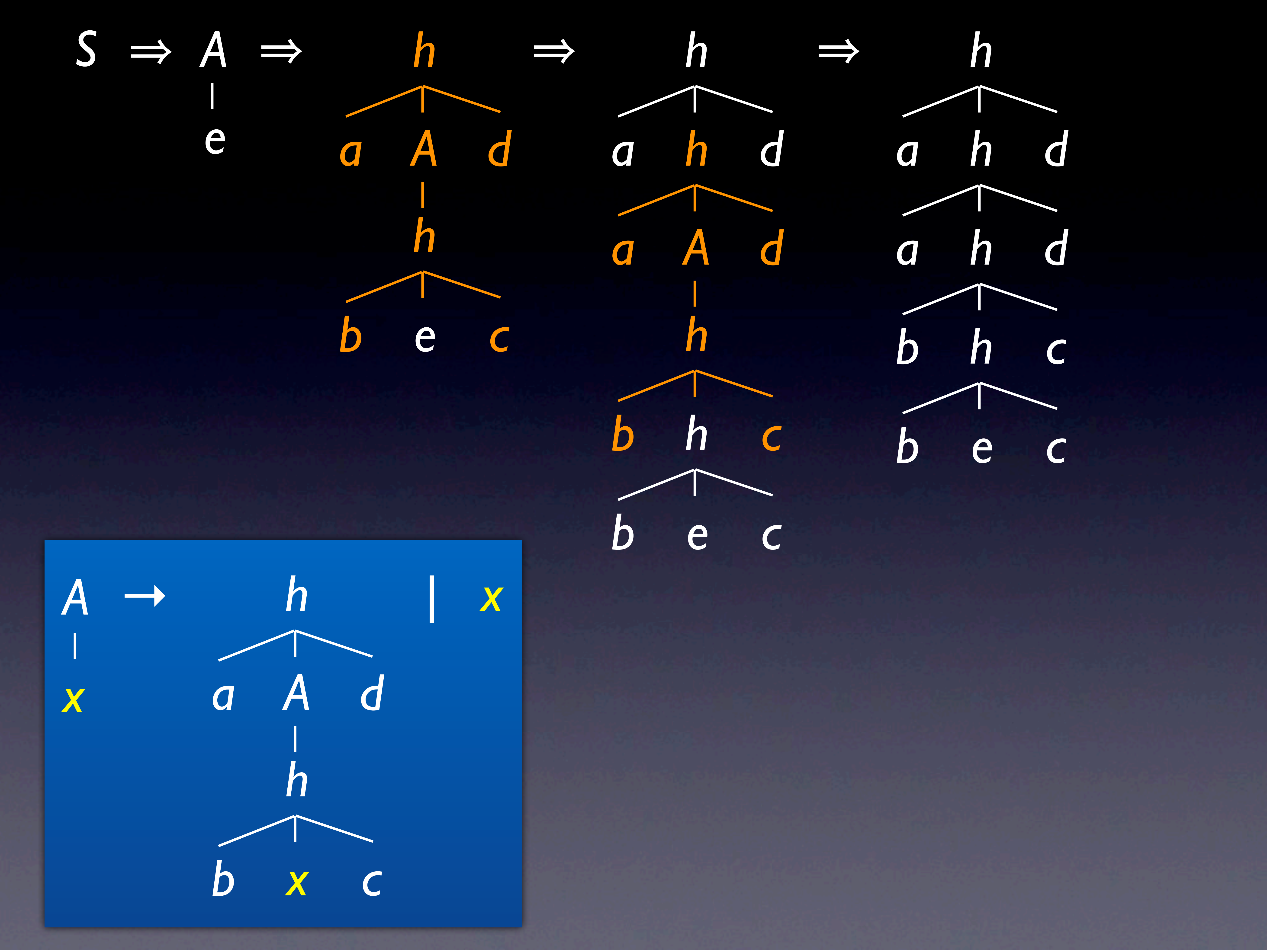
$$\mathrm{MCFL} = \bigcup_{m\geq 1} m\text{-}\mathrm{MCFL} \qquad y\mathrm{CFT}_{\mathrm{sp}} = \bigcup_{m\geq 1} y\mathrm{CFT}_{\mathrm{sp}}(m-1) \qquad \mathbf{C} = \bigcup_{k\geq 1} \mathbf{C}_k$$

TAG ≡ $\mathrm{CFT}_{sp}(1)$

$$N = N^{(0)} \cup N^{(1)} = \{S\} \cup \{A\}$$

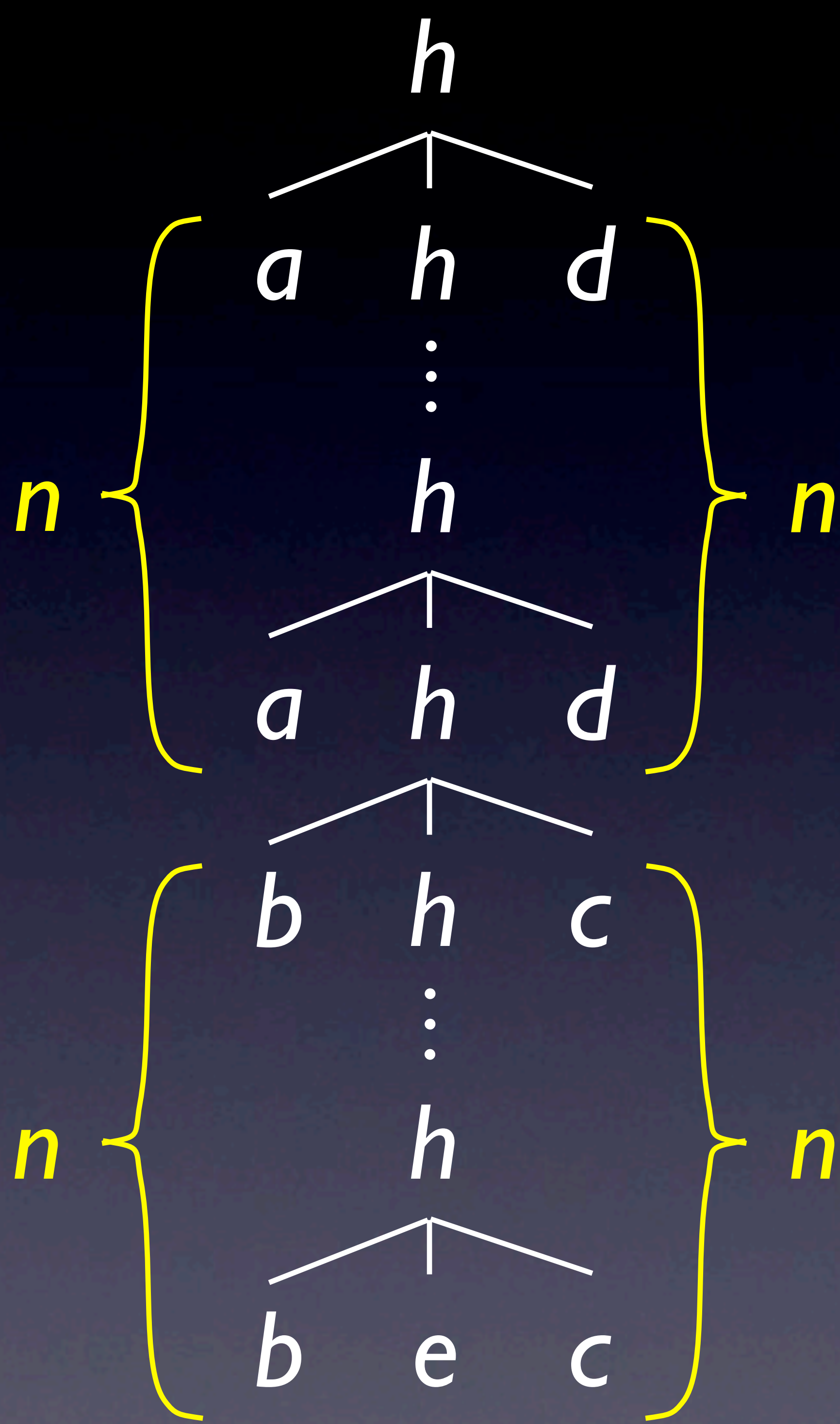
$$\Sigma = \Sigma^{(0)} \cup \Sigma^{(1)} = \{a, b, c, d, e\} \cup \{h\}$$

A CFT grammar generates a ranked tree language. Nonterminals and terminals come with ranks.
This grammar is “monadic” because the maximal rank of a nonterminal is 1.



7

An example of a derivation.

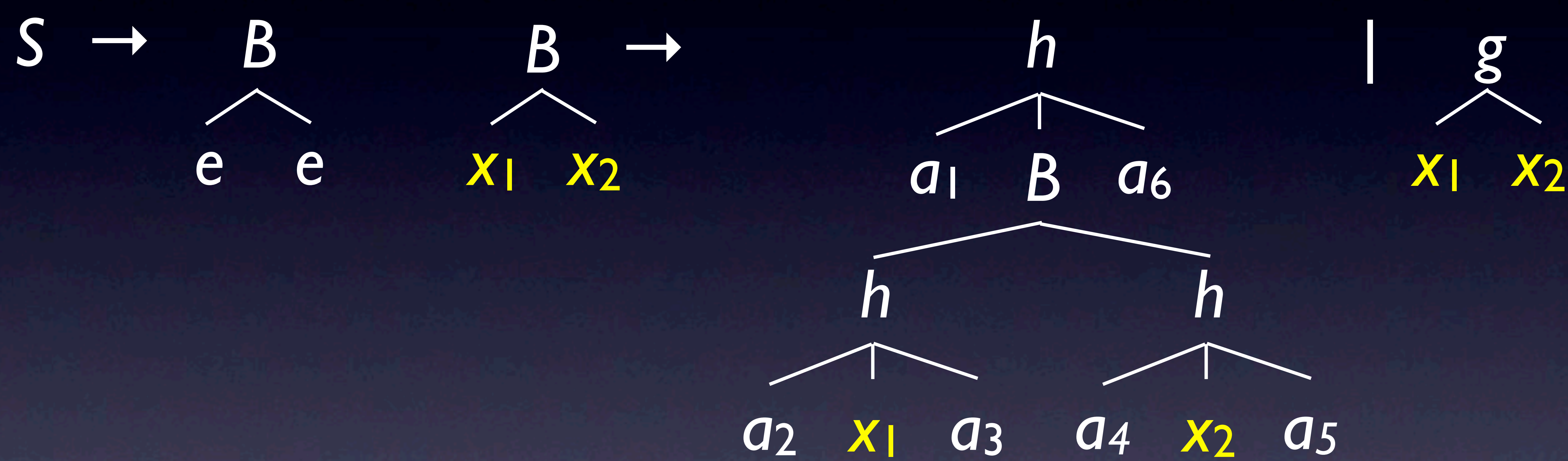


$a^n b^n e c^n d^n$

yield

This example grammar generates all trees of this form.

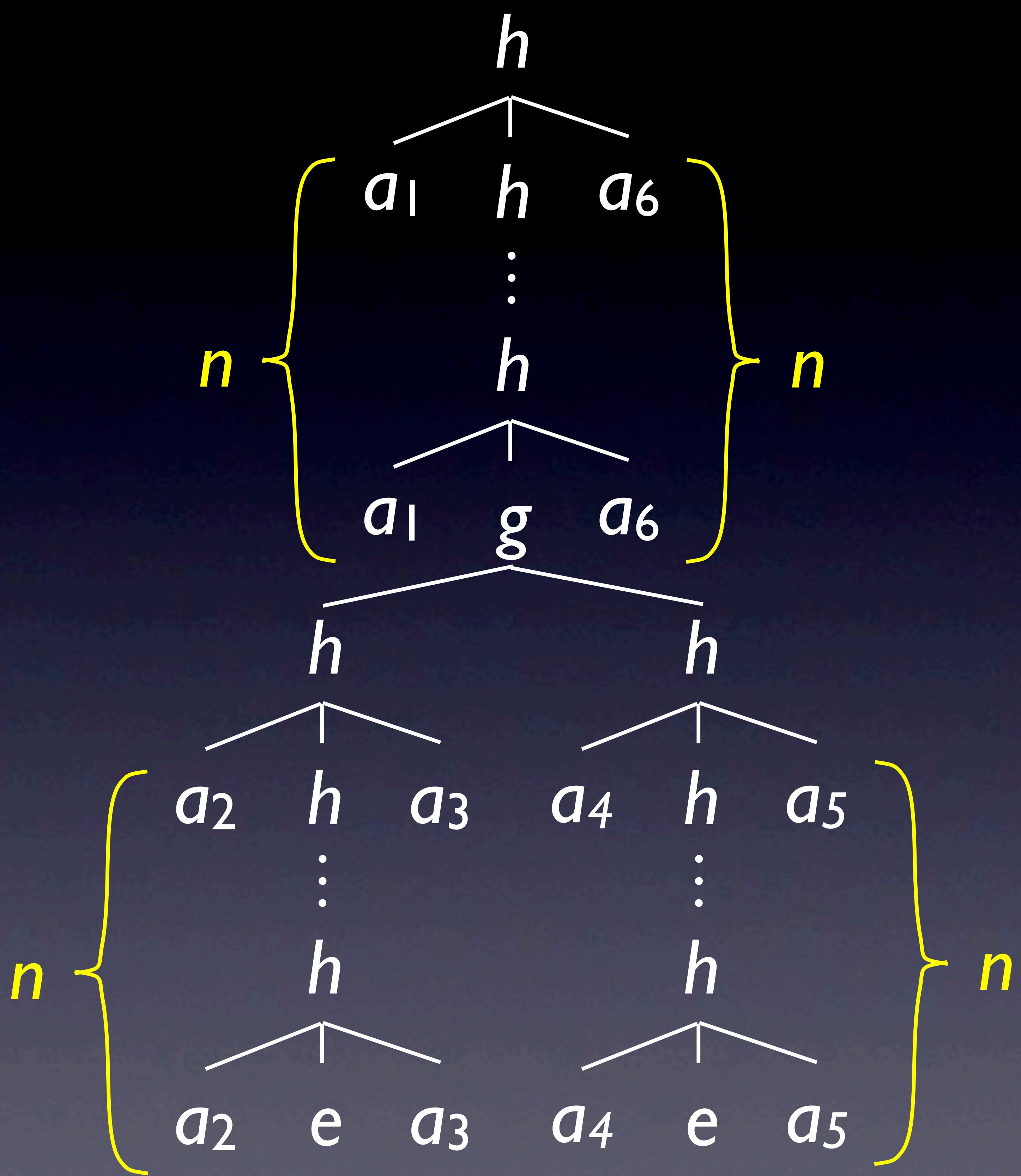
$\mathrm{CFT}_{sp}(2)$



$$N = N^{(0)} \cup N^{(2)} = \{S\} \cup \{B\}$$

$$\Sigma = \Sigma^{(0)} \cup \Sigma^{(2)} \cup \Sigma^{(3)} = \{a_1, a_2, a_3, a_4, a_5, a_6, e\} \cup \{g\} \cup \{h\}$$

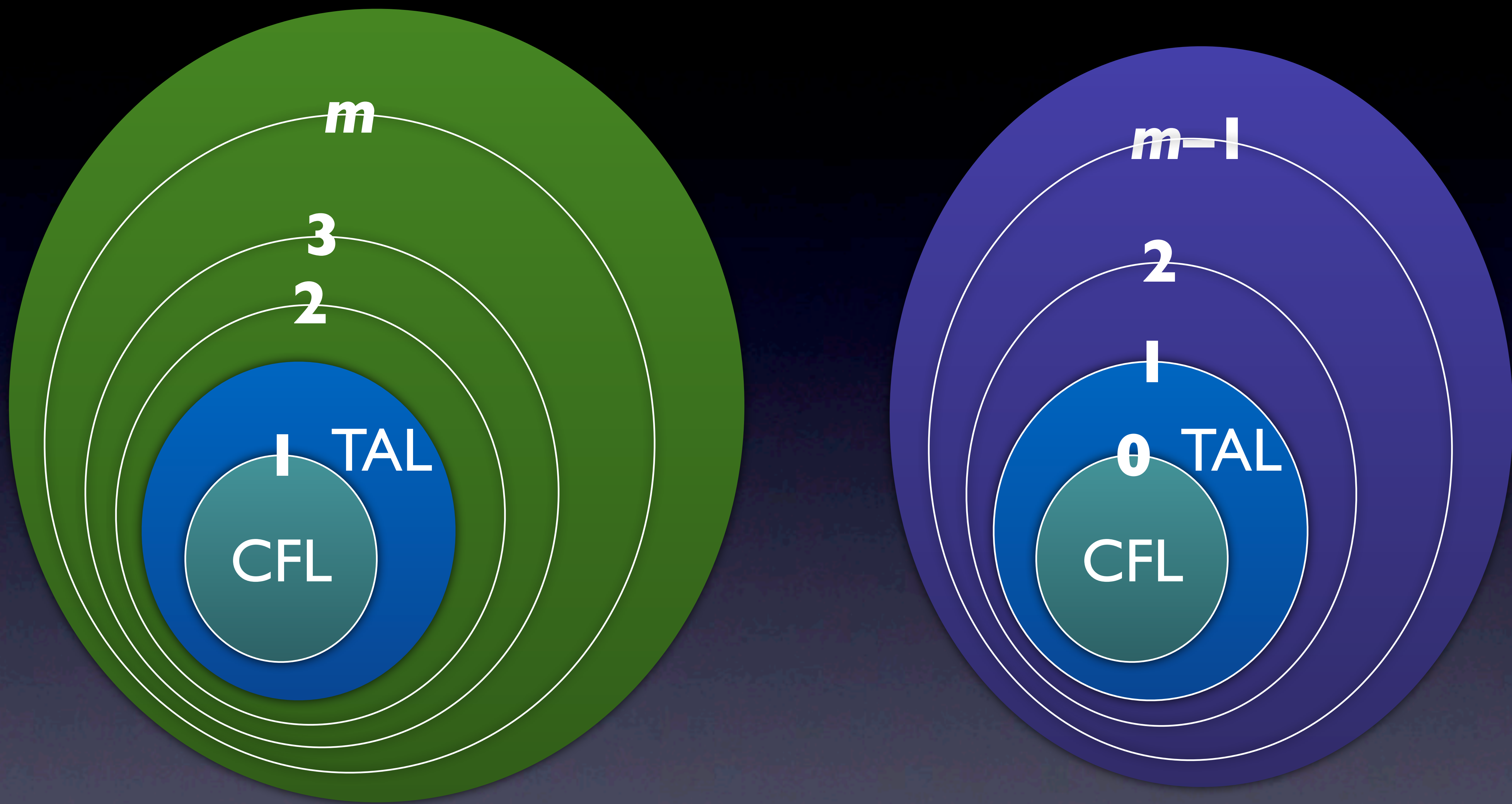
This grammar has a rank 2 nonterminal.



$a_1^n a_2^n e a_3^n a_4^n e a_5^n a_6^n$

yield

The yield image of the language of this grammar is not a TAL.



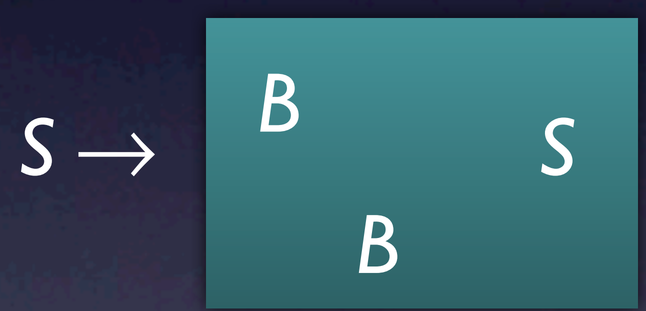
$$\mathrm{MCFL} = \bigcup_{m \geq 1} m\text{-MCFL} \qquad y\mathrm{CFT}_{\mathrm{sp}} = \bigcup_{m \geq 1} y\mathrm{CFT}_{\mathrm{sp}}(m-1)$$

Let’s look at how corresponding levels of these two hierarchies compare with each other.

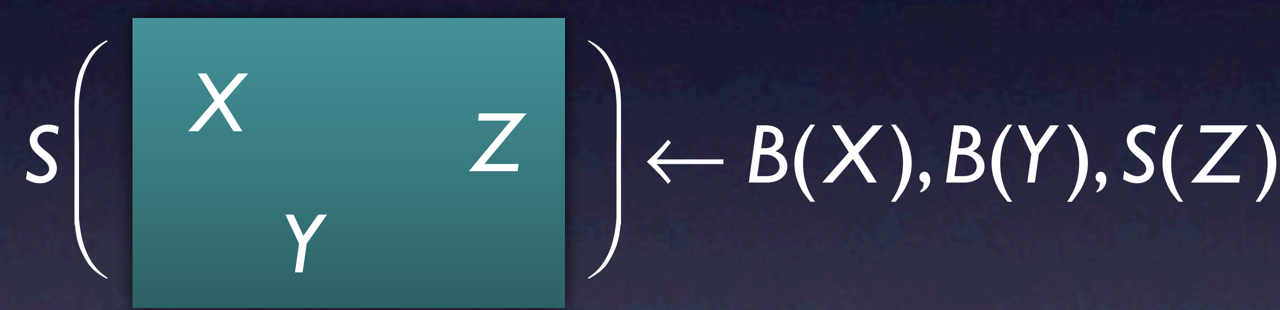
$$y\mathrm{CFT}_{\mathrm{sp}}(m-1) \subseteq m\text{-MCFL}$$

Seki and Kato 2008
de Groote and Pogodalla 2004, Salvati 2007

“Context-Free” Grammar Formalisms



top-down rewriting of sentential forms



bottom-up construction of derived objects

In order to convert a CFT_{sp} to an MCFG, we take a bottom-up view of the former, because the latter is a bottom-up formalism.

Top-down vs. bottom-up

Type 0	EFS	Smullyan 1961
Type 1 = CSG	l.b. EFS ≡ CSG	Arikawa et al. 1989
	simple LMG ≡ P	Groenink 1997
	PMCFG	Seki et al. 1991, Groenink 1997
	MCFG	Seki et al. 1991, Groenink 1997
Type 2 = CFG	simple EFS ≡ CFG	Arikawa 1970
Type 3		
rewriting systems	logic programs on strings	

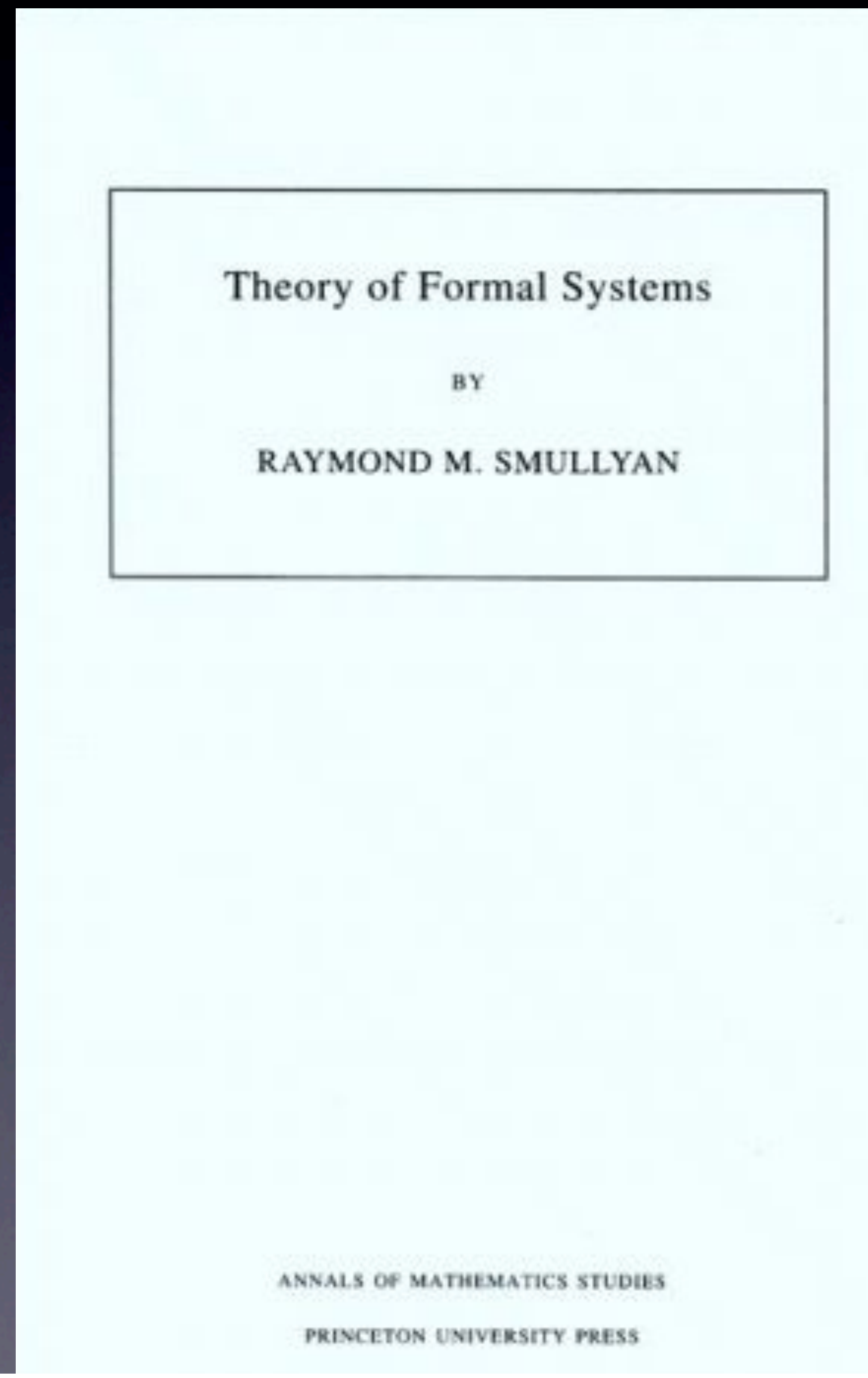
The bottom-up formalisms give you a richer picture. Note that almost all formalisms that were not defined by Chomsky do not fit within the Chomsky Hierarchy.
For example, an indexed grammar is *not* an instance of a Type 0 grammar.
An MCFG is an instance of an Elementary Formal System of Smullyan, which was rediscovered by Groenink.

1956

Thue ⟶ Post ⟶ Chomsky ⟶

Smullyan

1961


Theory of Formal Systems

BY

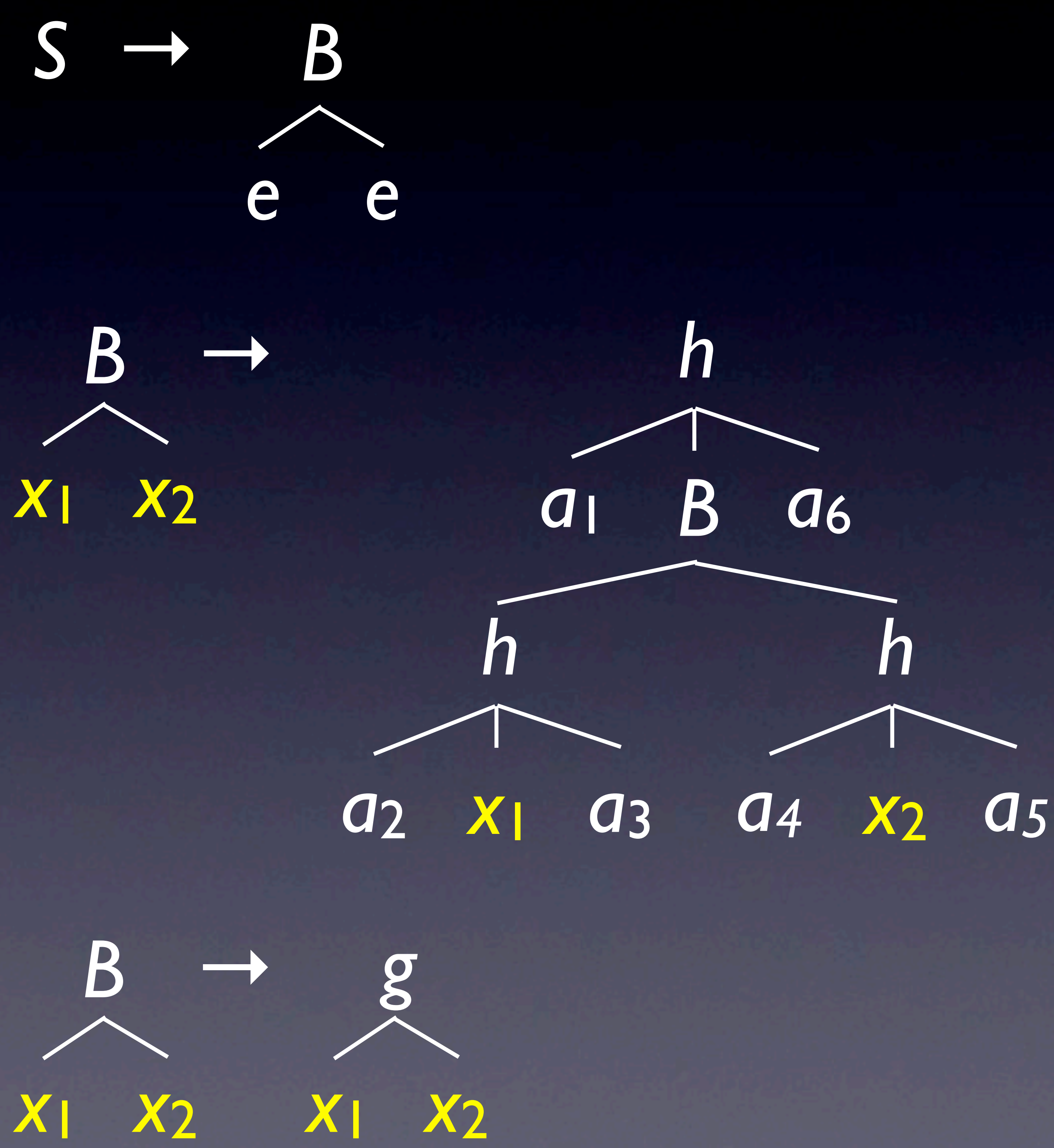
RAYMOND M. SMULLYAN

ANNALS OF MATHEMATICS STUDIES

PRINCETON UNIVERSITY PRESS

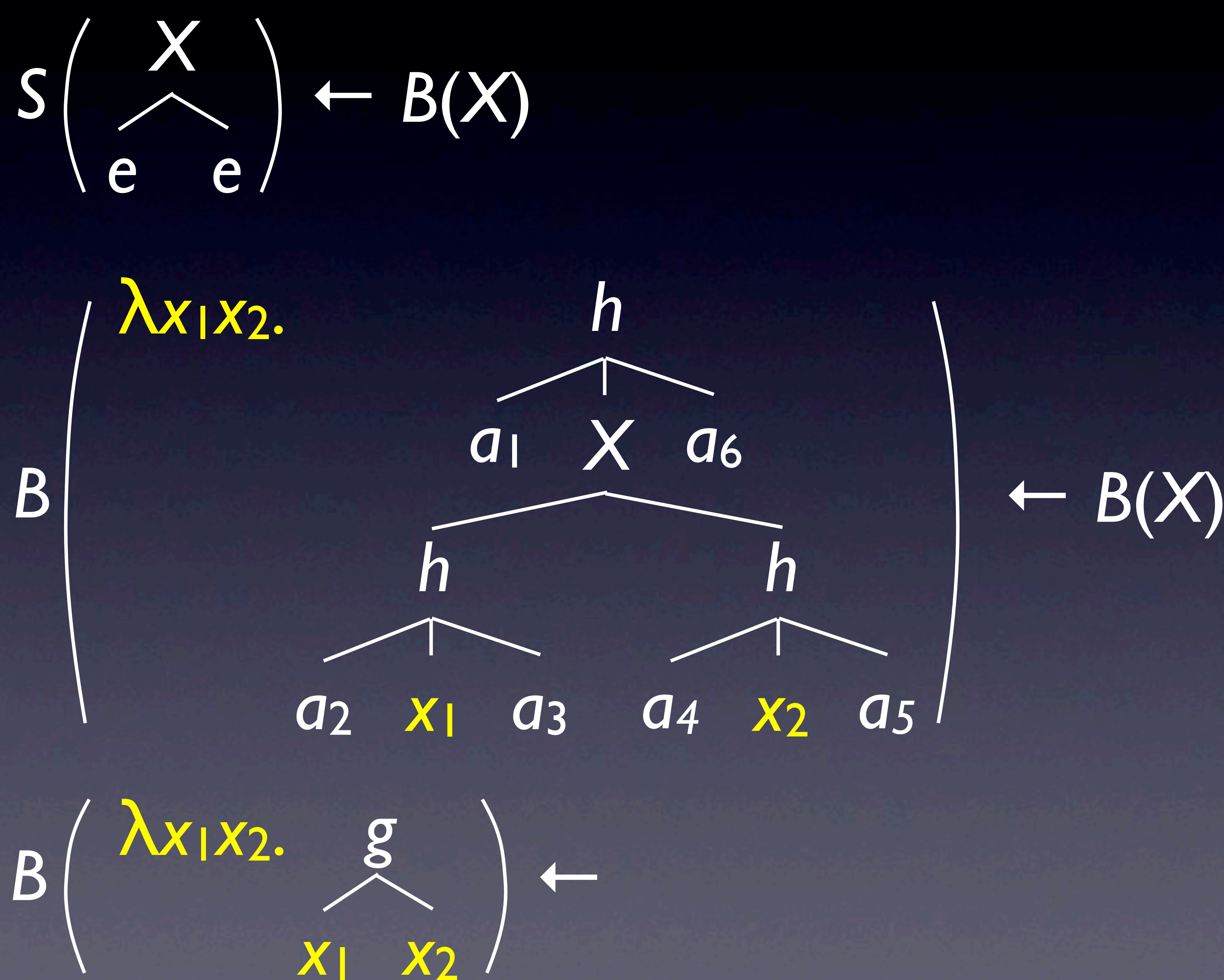
It’s too bad that Smullyan’s work came a little too late for Chomsky to take notice. The world would have been a better place if Chomsky had based his theory of formal grammar on Smullyan’s work, rather than the work of Thue and Post.

Top-down View

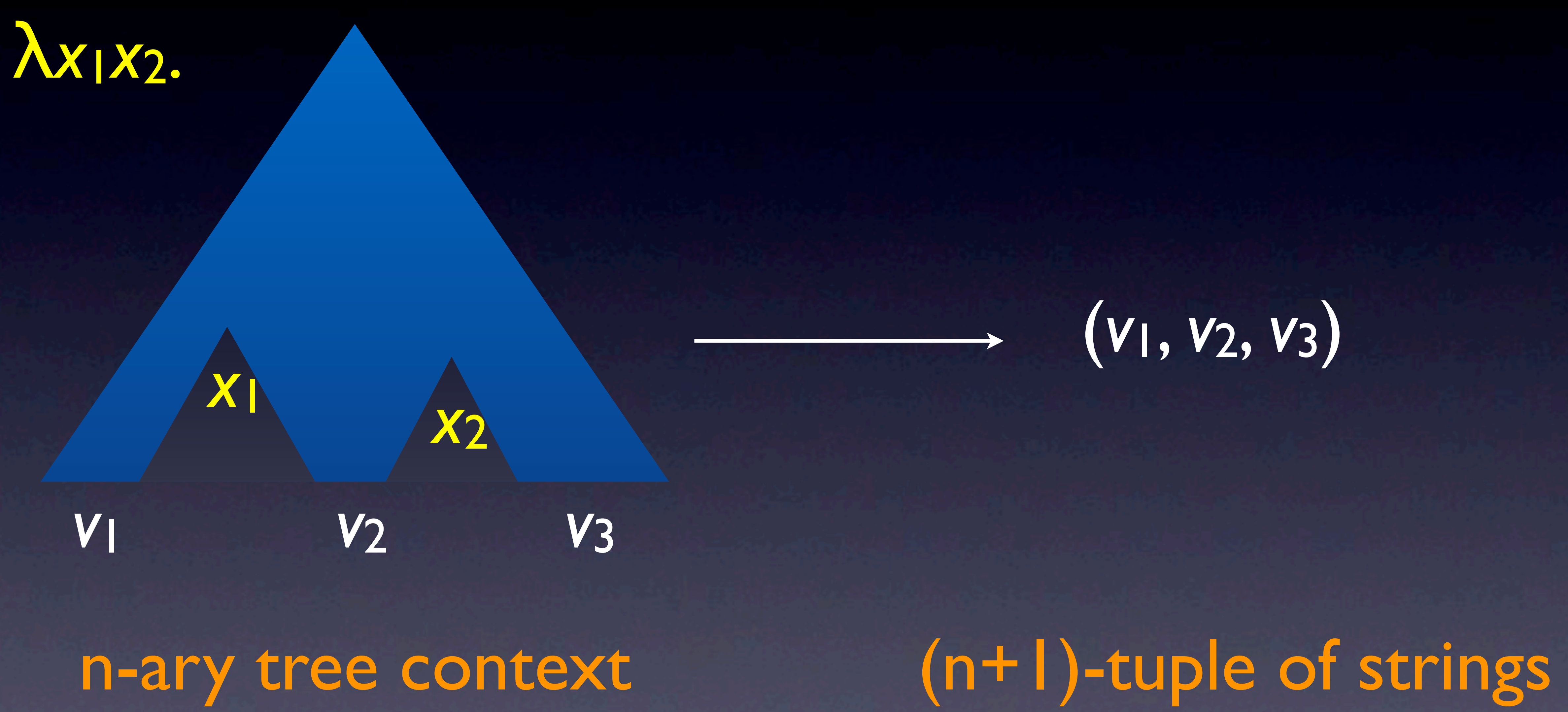


Standard, top-down view of CFT grammar rules.

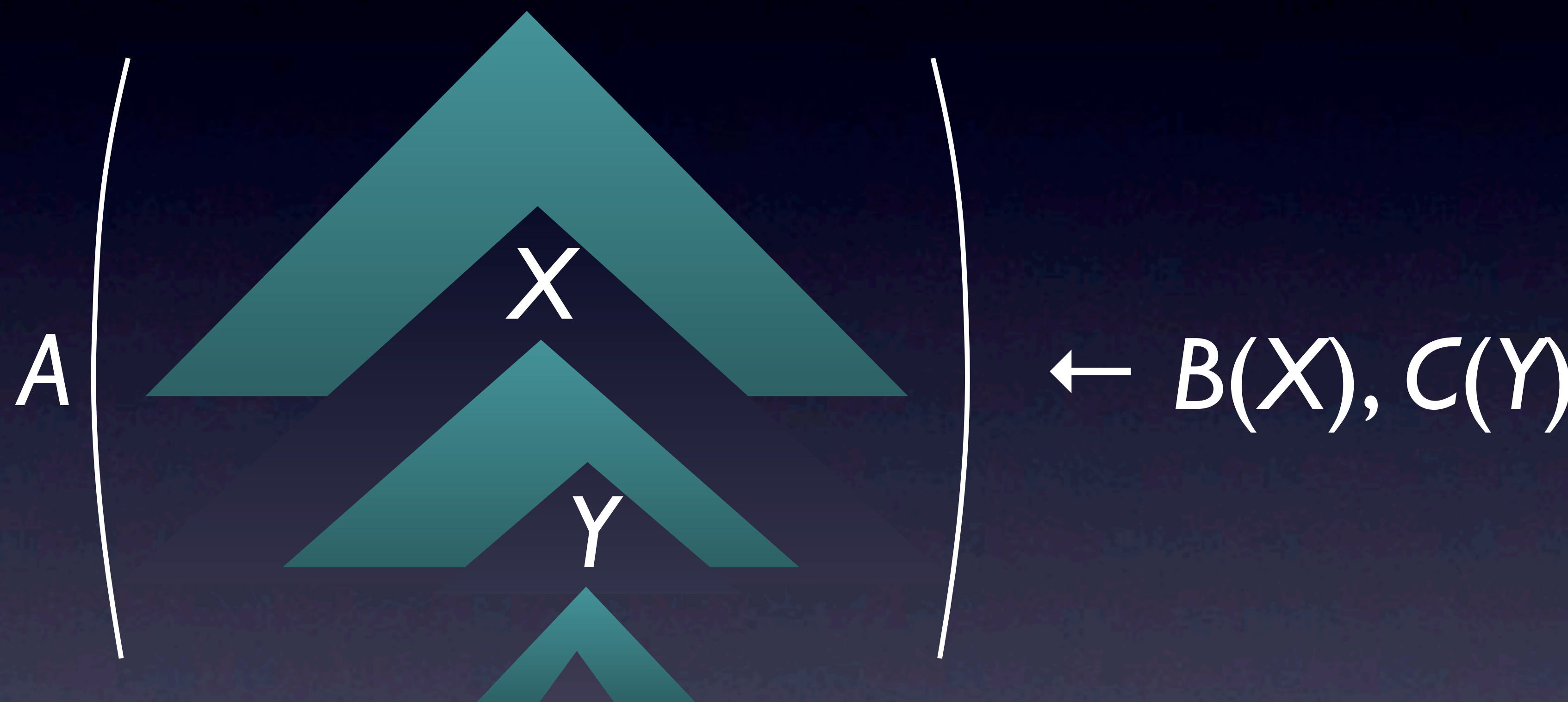
Bottom-up View



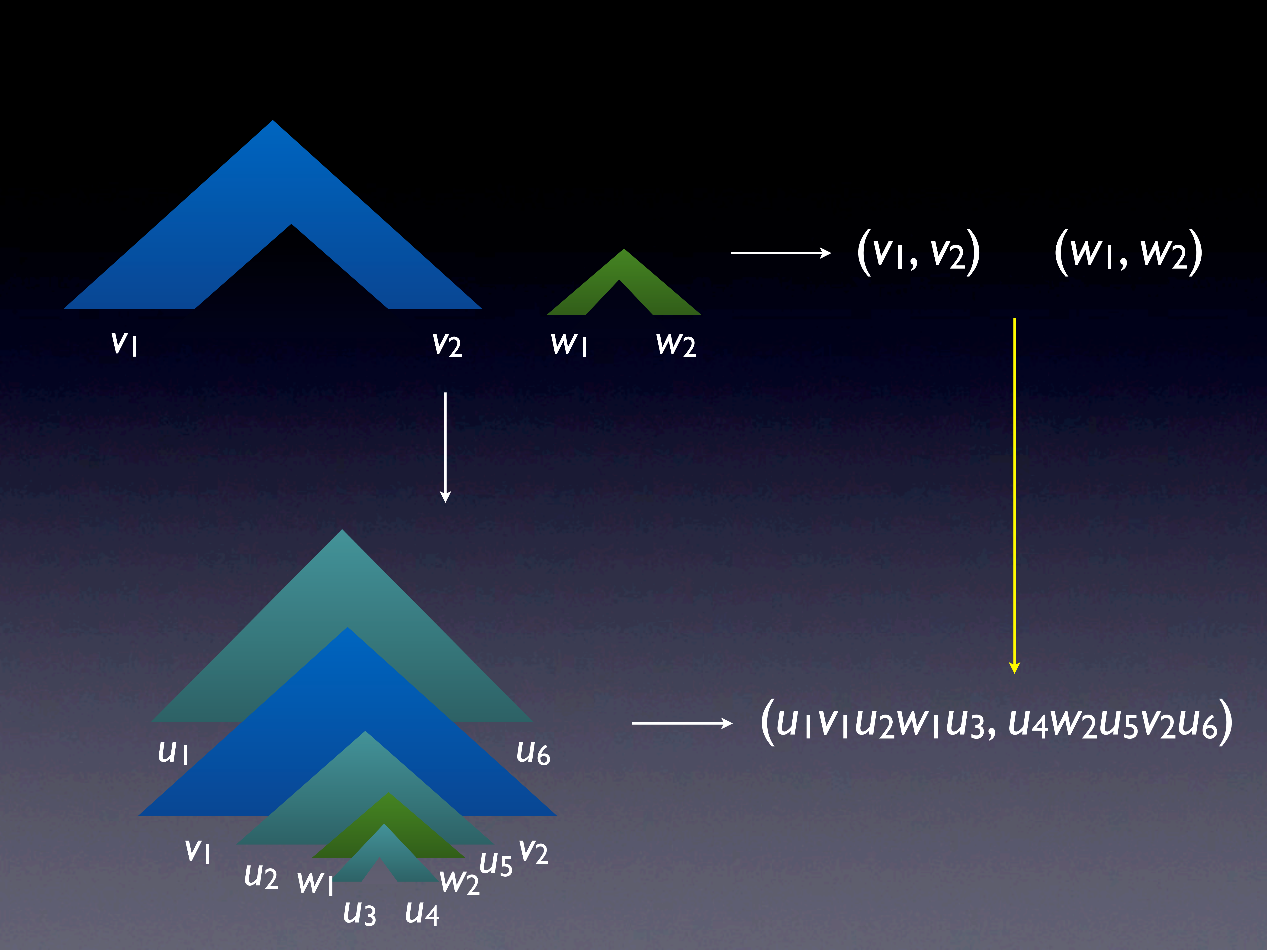
An alternative bottom-up view.



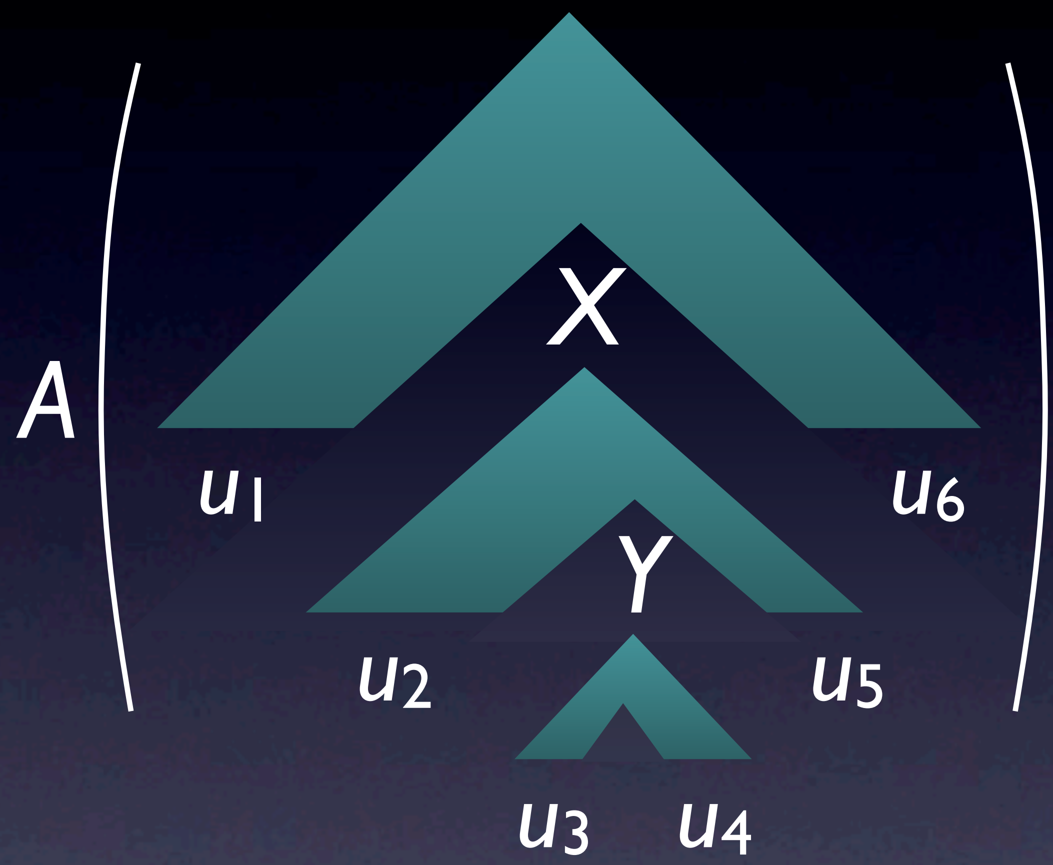
There's a natural mapping form n-ary tree contexts to (n+1)-tuples of strings.



What about the operation in a CFT_{sp} rule? Does it naturally correspond to an operation on tuples of strings?



Yes, it does. This is a commutative diagram!



$\leftarrow B(X), C(Y)$

$$A(u_1\,\boldsymbol{x}_1\,u_2\,\boldsymbol{y}_1\,u_3,\ \ u_4\,\boldsymbol{y}_2\,u_5\,\boldsymbol{x}_2\,u_6) \leftarrow B(\boldsymbol{x}_1, \boldsymbol{x}_2), C(\boldsymbol{y}_1, \boldsymbol{y}_2)$$

This is how to translate a $\mathrm{CFT}_{sp}(1)$ rule to a 2-MCFG rule.

$$y\mathrm{CFT}_{\mathrm{sp}}(m-1) \subseteq m\text{-MCFL}$$

Seki and Kato 2008
de Groote and Pogodalla 2004, Salvati 2007

The translation establishes this inclusion.

Well-nested MCFGs

✗ $S(x_1 y_1 x_2 y_2) \leftarrow A(x_1, x_2), B(y_1, y_2)$

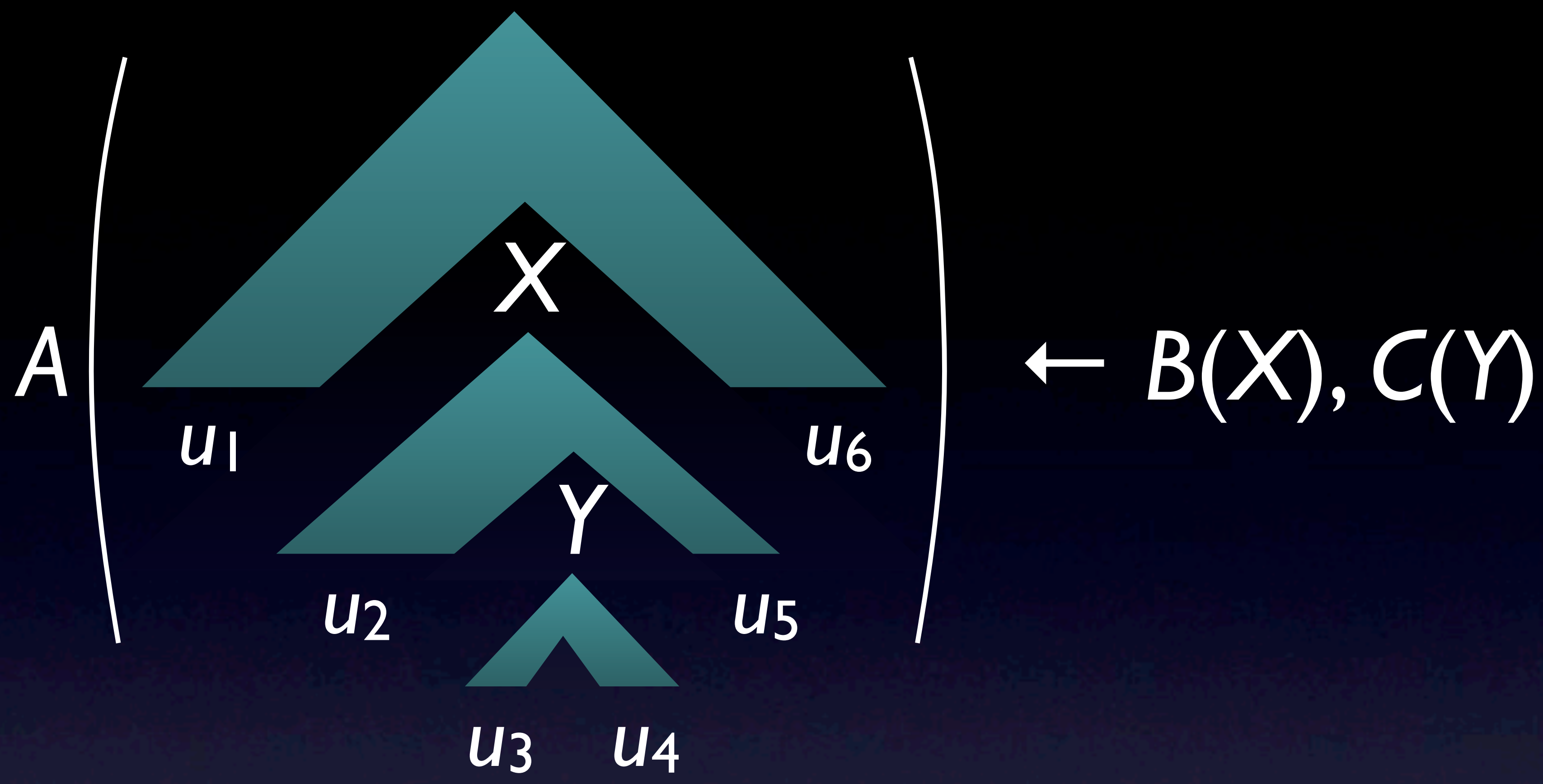
✓ $S(x_1 y_1 y_2 x_2) \leftarrow A(x_1, x_2), B(y_1, y_2)$

✗ $C(x_1 y_1, y_2 z_1, z_2 x_2 z_3) \leftarrow A(x_1, x_2), B(y_1, y_2), C(z_1, z_2, z_3)$

✓ $C(z_1 x_1, x_2 z_2, y_1 y_2 z_3) \leftarrow A(x_1, x_2), B(y_1, y_2), C(z_1, z_2, z_3)$

Cf. Kuhlmann 2007

Assume all rules are “non-permuting” (x_i appears before x_j if $i < j$).



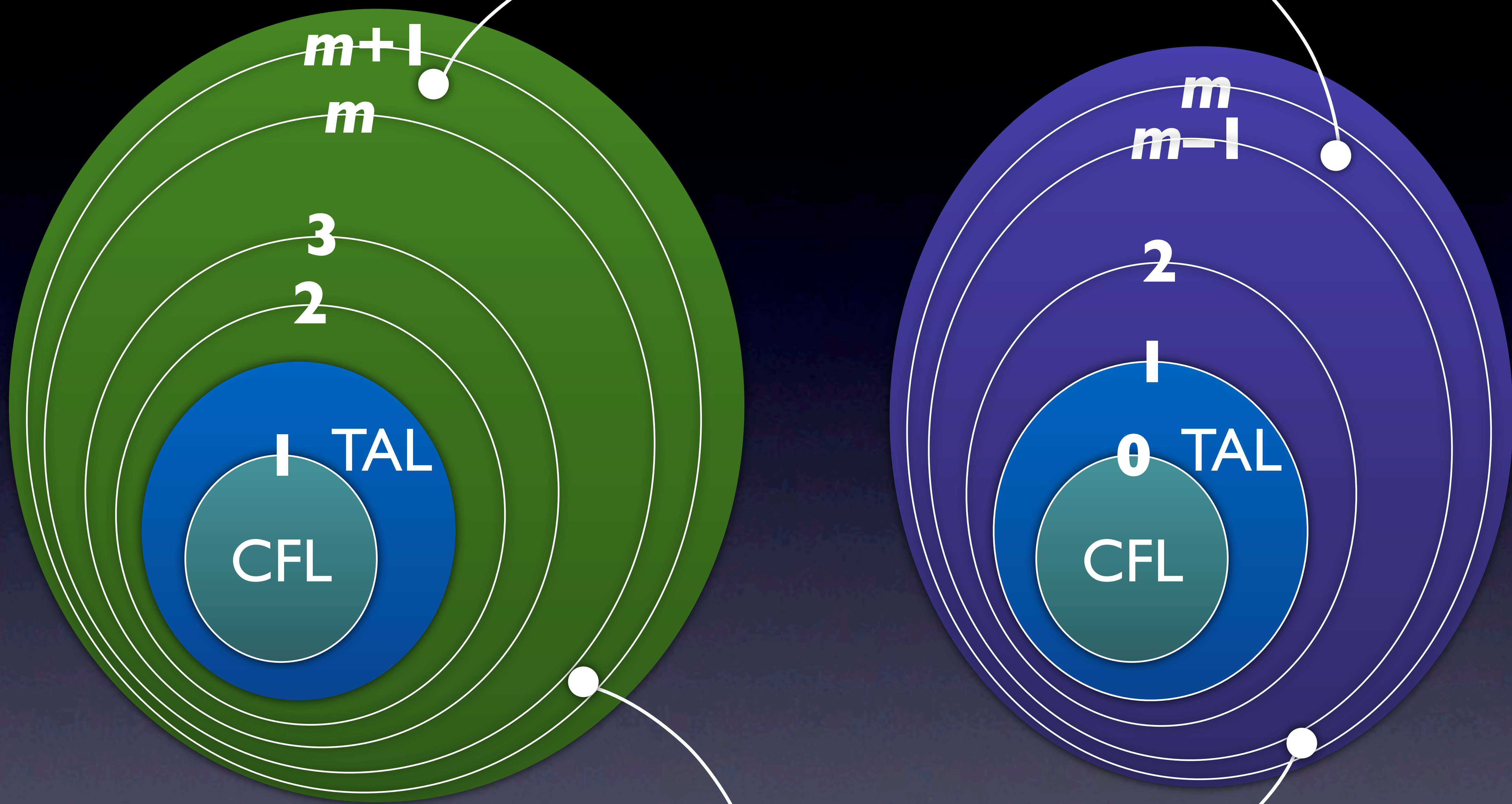
$$A(u_1\, \mathbf{x}_1\, u_2\, \mathbf{y}_1\, u_3,\ \ u_4\, \mathbf{y}_2\, u_5\, \mathbf{x}_2\, u_6) \leftarrow B(\mathbf{x}_1, \mathbf{x}_2), C(\mathbf{y}_1, \mathbf{y}_2)$$

$$y\mathrm{CFT}_{\mathrm{sp}}(m-1) \subseteq m\text{-}\mathrm{MCFL}_{\mathrm{wn}}$$

$$y\mathrm{CFT}_{\mathrm{sp}}(m-1) \supseteq m\text{-}\mathrm{MCFL}_{\mathrm{wn}}$$

The translation of a $\mathrm{CFT}_{\mathrm{sp}}$ rule gives you a well-nested MCFG rule.

$$\{\, a_1^n \ldots a_{2(m+1)}^n \mid n \geq 0 \,\}$$



$$\{\, w^{m+1} \mid w \in \{a, b\}^* \,\}$$

$$\mathrm{MCFL} = \bigcup_{m \geq 1} m\text{-}\mathrm{MCFL} \qquad y\mathrm{CFT}_{sp} = \bigcup_{m \geq 1} y\mathrm{CFT}_{sp}(m-1)$$

What about separation of the corresponding levels of the two hierarchies? These languages do not separate them.

m-MCFL vs. $y\mathrm{CFT}_{\mathrm{sp}}(m-1)$

$$\mathrm{RESP}_2 = \{ a_1^i a_2^i b_1^j b_2^j a_3^i a_4^i b_3^j b_4^j \mid i, j \geq 0 \}$$ Weir 1989

$$\mathrm{RESP}_2 \in 2\text{-MCFL} - y\mathrm{CFT}_{\mathrm{sp}}(1)$$ Seki et al. 1991

$$\mathrm{RESP}_m = \{ a_1^i a_2^i b_1^j b_2^j \ldots a_{2m-1}^i a_{2m}^i b_{2m-1}^j b_{2m}^j \mid i, j \geq 0 \}$$

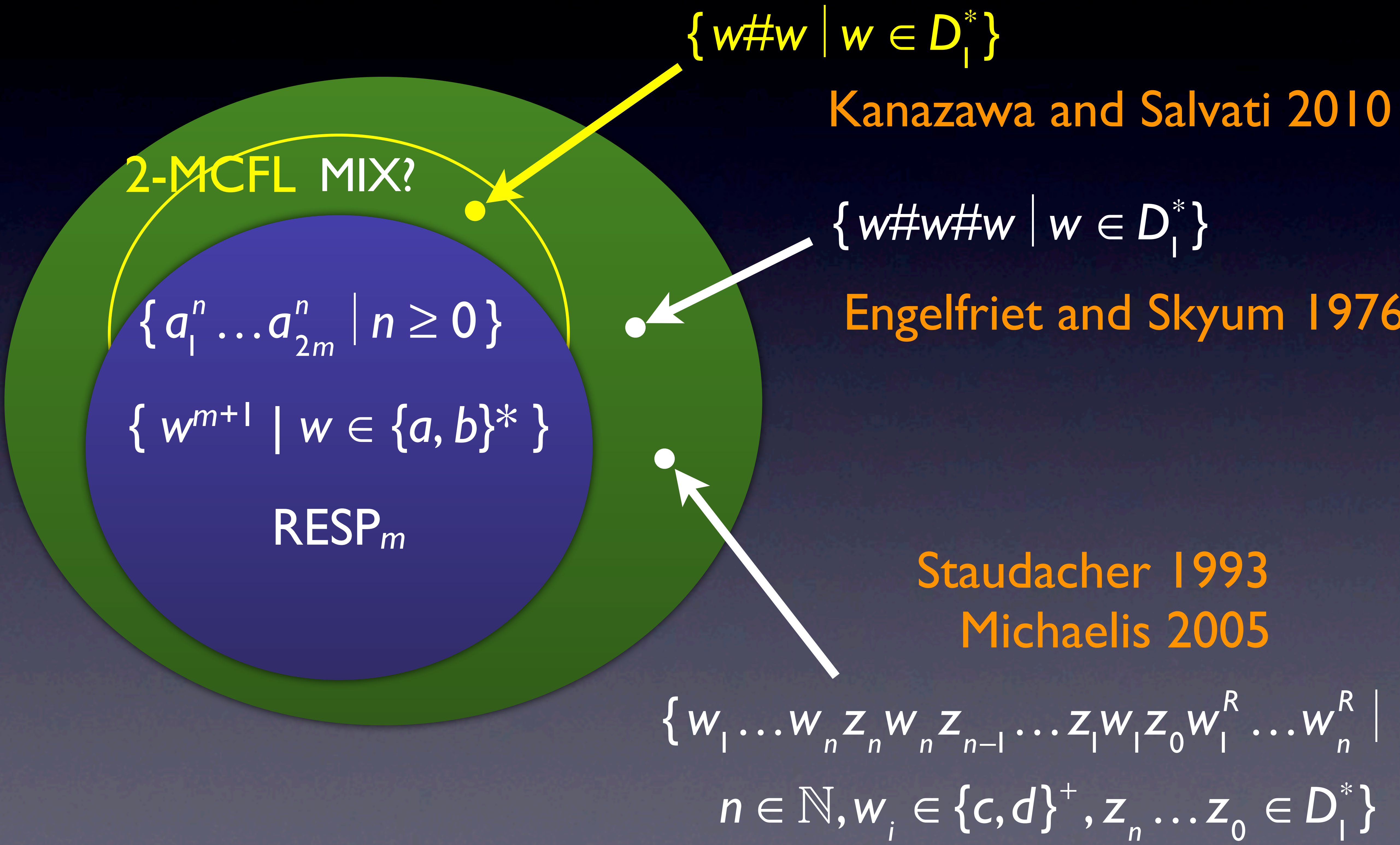
$$\mathrm{RESP}_m \in m\text{-MCFL} - y\mathrm{CFT}_{\mathrm{sp}}(m-1) \quad \text{for } m \geq 2$$

Seki and Kato 2008

$$\mathrm{RESP}_m \in y\mathrm{CFT}_{\mathrm{sp}}(2m-1)$$

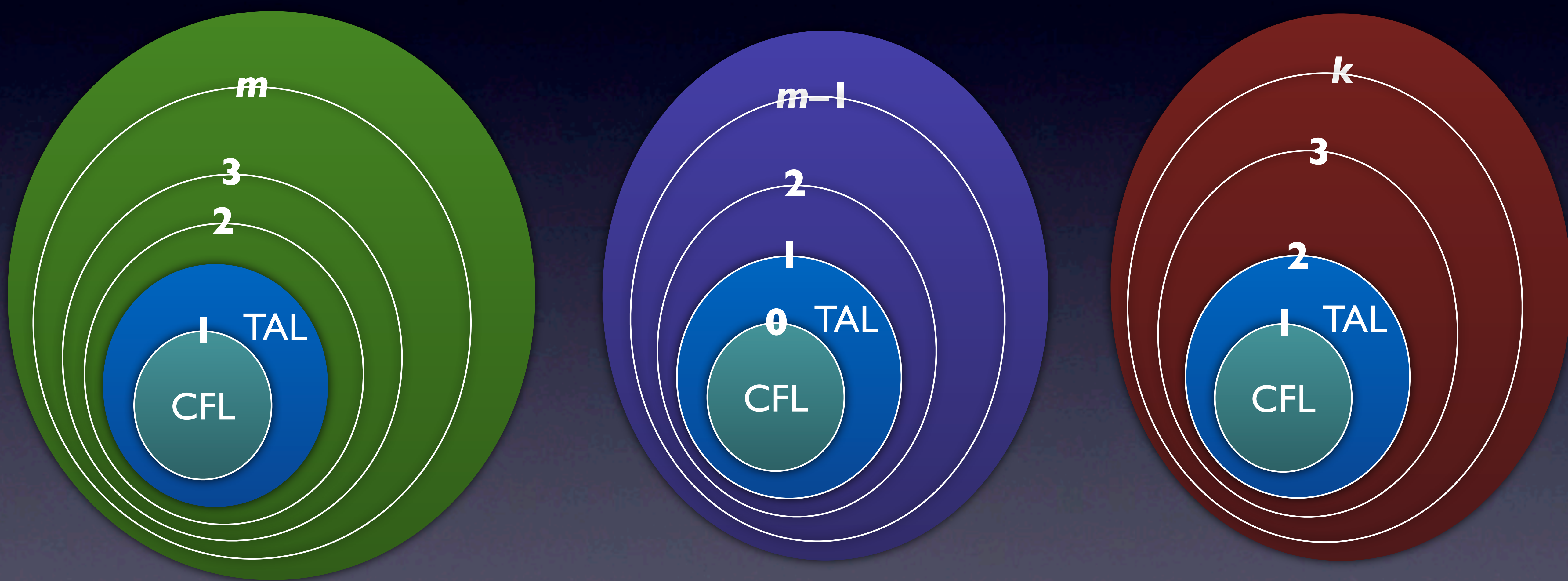
m−MCFL and m−MCFL$_{\mathrm{wn}}$ have many languages in common, but are of course different.
Separation at each level is witnessed by RESP$_m$.
But RESP$_m$ is inside the entire yCFT$_{\mathrm{sp}}$ hierarchy.

MCFL vs. $yCFT_{sp}$



With Kanazawa and Salvati's (2010) theorem, we can see 2-MCFL − $MCFL_{wn} \neq \varnothing$.
Improves known results.

Three Infinite Hierarchies



$$\mathrm{MCFL} = \bigcup_{m \geq 1} m\text{-}\mathrm{MCFL} \qquad y\mathrm{CFT}_{\mathrm{sp}} = \bigcup_{m \geq 1} y\mathrm{CFT}_{\mathrm{sp}}(m-1) \qquad \mathbf{C} = \bigcup_{k \geq 1} \mathbf{C}_k$$

Let's look at the third hierarchy.

Controlled Tree Languages

$$\mathrm{CT}(L_1, L_2) = L_1 \cap (\widehat{L_2})^{*,c}$$

$$L_1 \subseteq T_\Delta$$

$$L_2 \subseteq \Delta^*$$

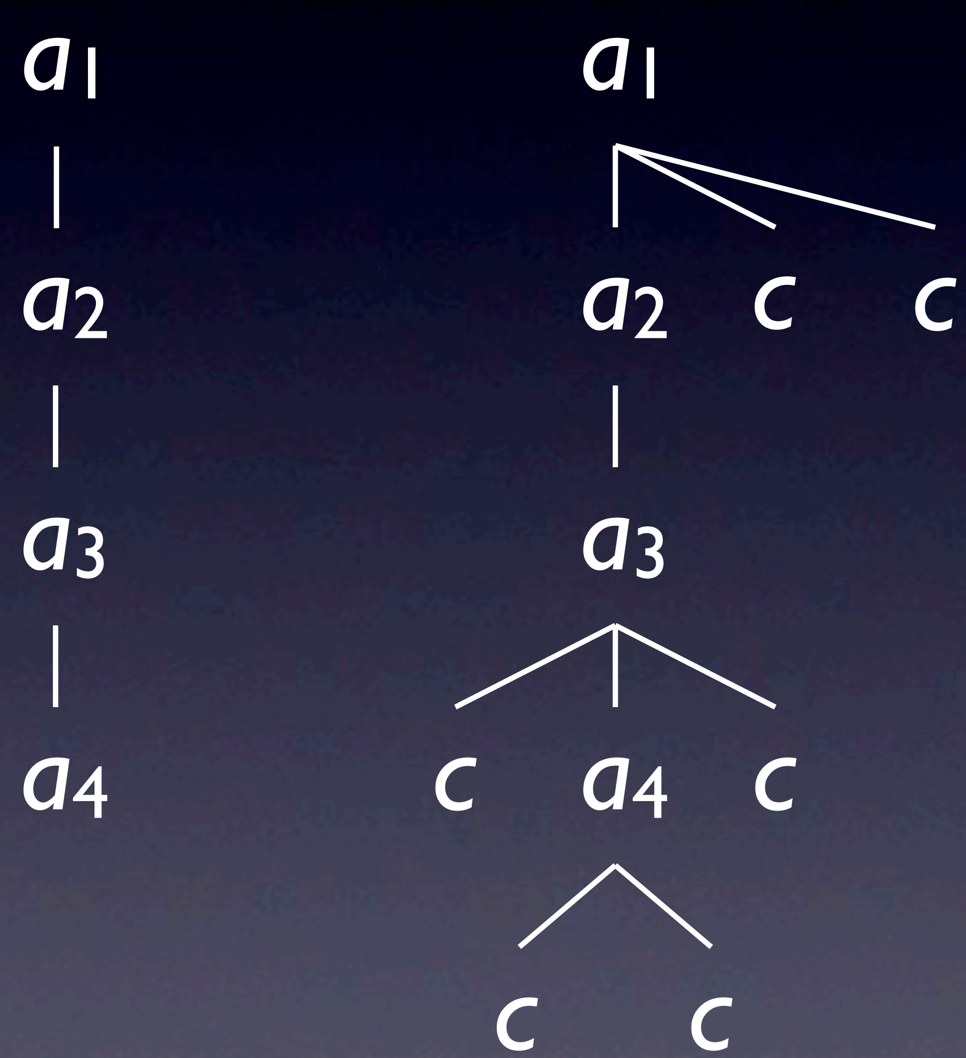
$$h : \Delta \to \mathbb{N}$$

$$d \in \Delta^{(r)} \Rightarrow h(d) \in \{0, \ldots, r\}$$

Let's define an operation CT, which takes a tree language L_1 and a string language L_2 and returns a subset of L_1. This operation is parametric on a function h.
Defined in terms of three operations: the hat operation, the tree analogue of the Kleene star operation, and intersection.

$L \mapsto \hat{L}$

$a_1a_2a_3a_4$



$a_1 \in \Delta^{(3)}, h(a_1) = 1$

$a_2 \in \Delta^{(1)}, h(a_2) = 1$

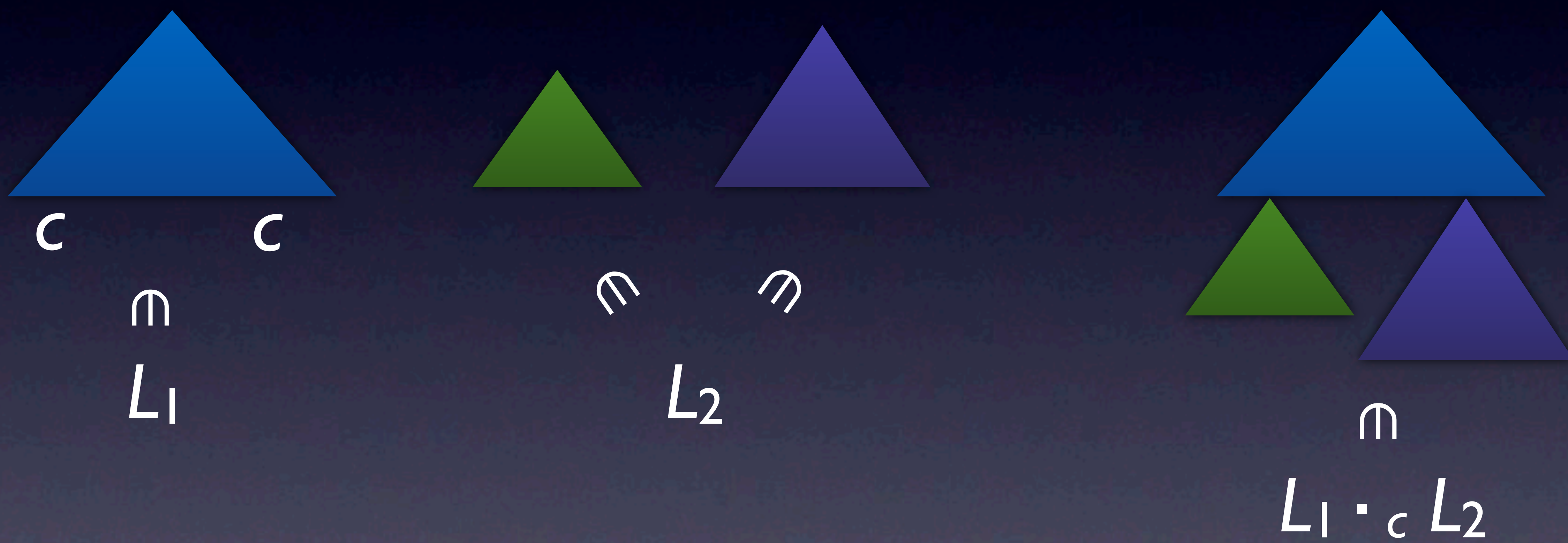
$a_3 \in \Delta^{(3)}, h(a_3) = 2$

$a_4 \in \Delta^{(2)}, h(a_4) = 0$

The hat operation turns a string into a tree.

Regular Tree Operations

Concatenation

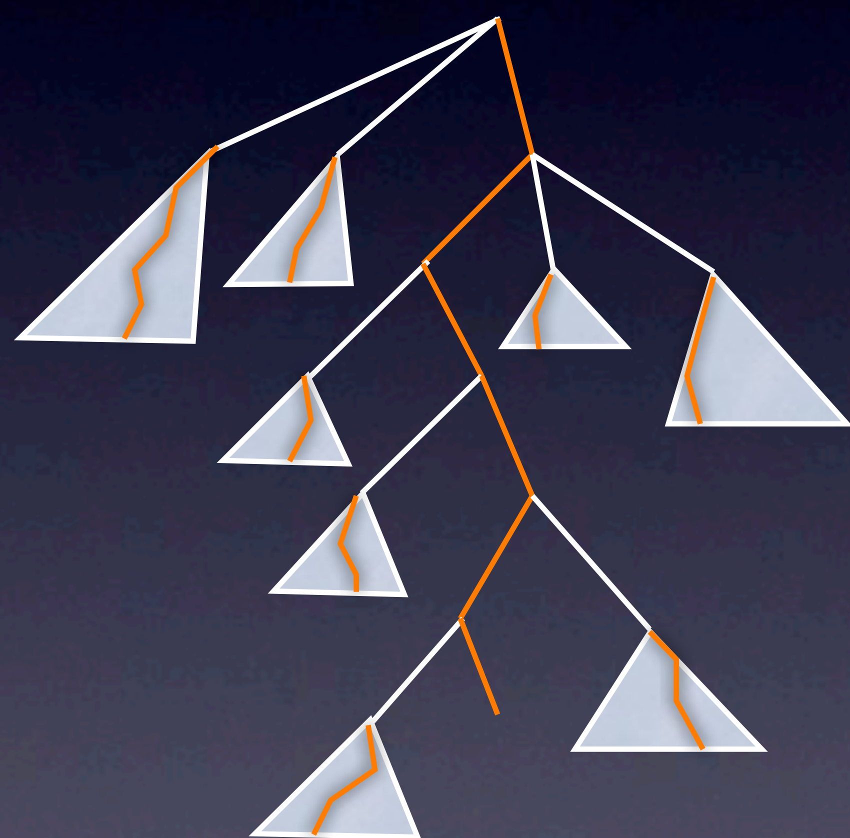


Kleene star

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

The Kleene star operation is a standard notion in tree automata theory.

$$\mathrm{CT}(L_1, L_2) = L_1 \cap (\widehat{L_2})^{*,c}$$



This depicts the CT operation. You can observe elements of L_2 along some paths in a tree in $\mathrm{CT}(L_1,L_2)$. These paths are determined by the h function.

$$\mathbf{CT}_1 = \mathrm{LOC}$$

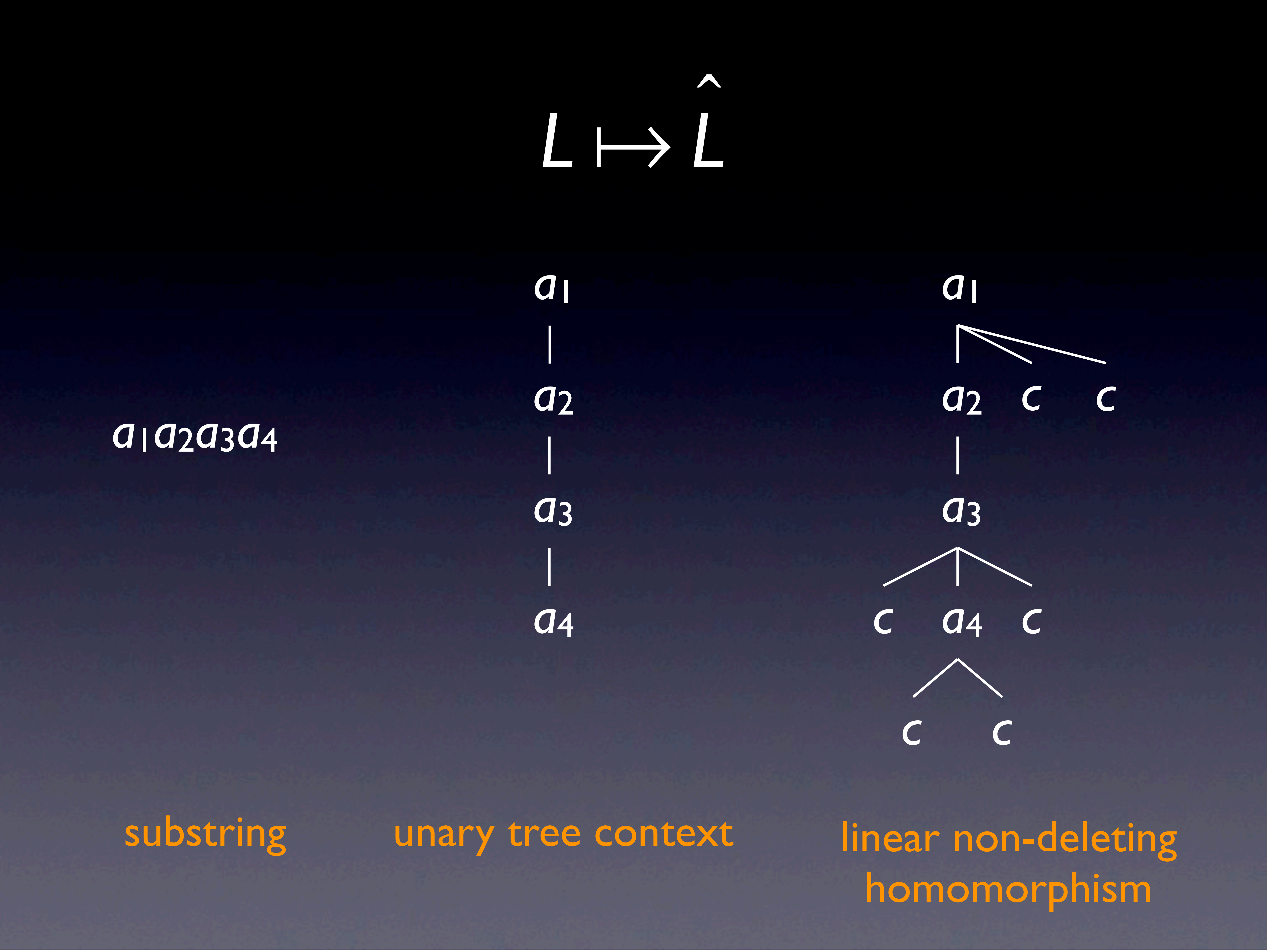
$$\mathbf{CT}_{k+1} = \mathrm{CT}(\mathrm{LOC}, y\mathbf{CT}_k)$$

$$\mathbf{C}_k = y\mathbf{CT}_k$$

$$\mathbf{C}_1 = \mathrm{CFL}$$

$$\mathbf{CT}_2 \subseteq \mathrm{CFT}_{sp}(1) \qquad \mathbf{C}_2 = y\mathrm{CFT}_{sp}(1)$$

Define the Control Tree Language Hierarchy in terms of CT, starting from the class of local tree languages.
Weir's Control Language Hierarchy is the yield image of this hierarchy.
Let's show CT_2 is included in $CFT_{sp}(1)$.



Going from strings to monadic trees, substrings are mapped to unary tree contexts, so a CFG is mapped to a $\mathrm{CFT}_{sp}(1)$. Addition of nodes labeled by c is a simple case of a linear non-deleting homomorphism.

$$\mathbf{CT}_2 = \{ L_1 \cap \widehat{(L_2)}^{*,c} \mid L_1 \in \mathrm{LOC},\ L_2 \in \mathrm{CFL} \}$$

$$\subseteq \mathrm{CFT}_{sp}(1)$$

$$\mathbf{C}_2 \subseteq y\mathrm{CFT}_{sp}(1)$$

$$\subseteq \text{2-MCFL}$$

$\mathrm{CFT}_{sp}(1)$ is closed under linear non-deleting homomorphism, Kleene star, and intersection with regular sets. This shows that CT_2 is included in $\mathrm{CFT}_{sp}(1)$.

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

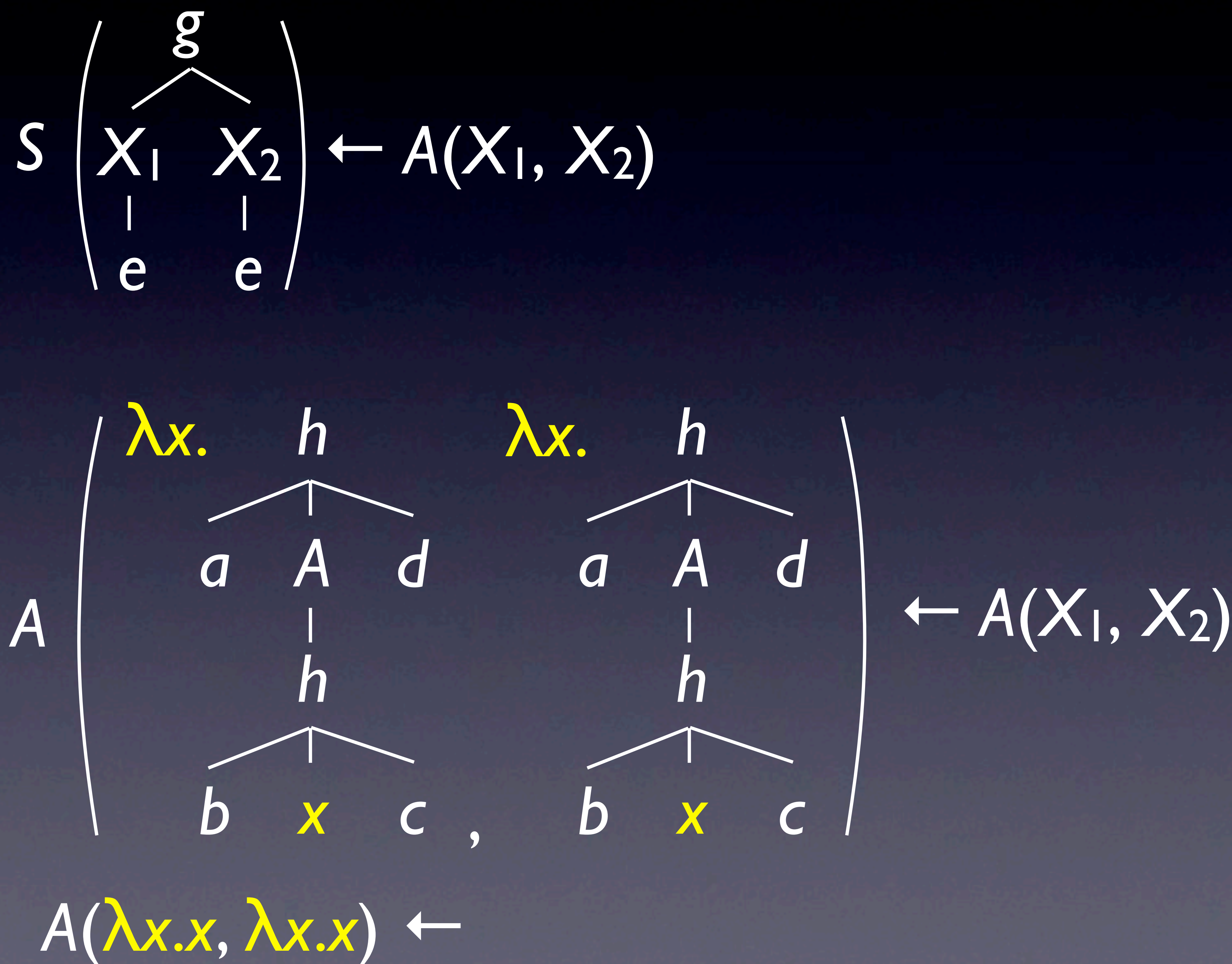
Kanazawa and Salvati 2007

$\mathbf{CT}_k \subseteq 2^{k-2}$-MCFT$_{sp}(1)$ $(k \geq 2)$

$\approx$ MCTAG (Weir 1988)

The induction step is similar. Going form strings to monadic trees, substrings are mapped to unary tree contexts, and an MCFG is mapped to a “multiple monadic simple context-free tree grammar”. The latter is basically the same as a multicomponent TAG.

2-MCFT$_{sp}$(1)



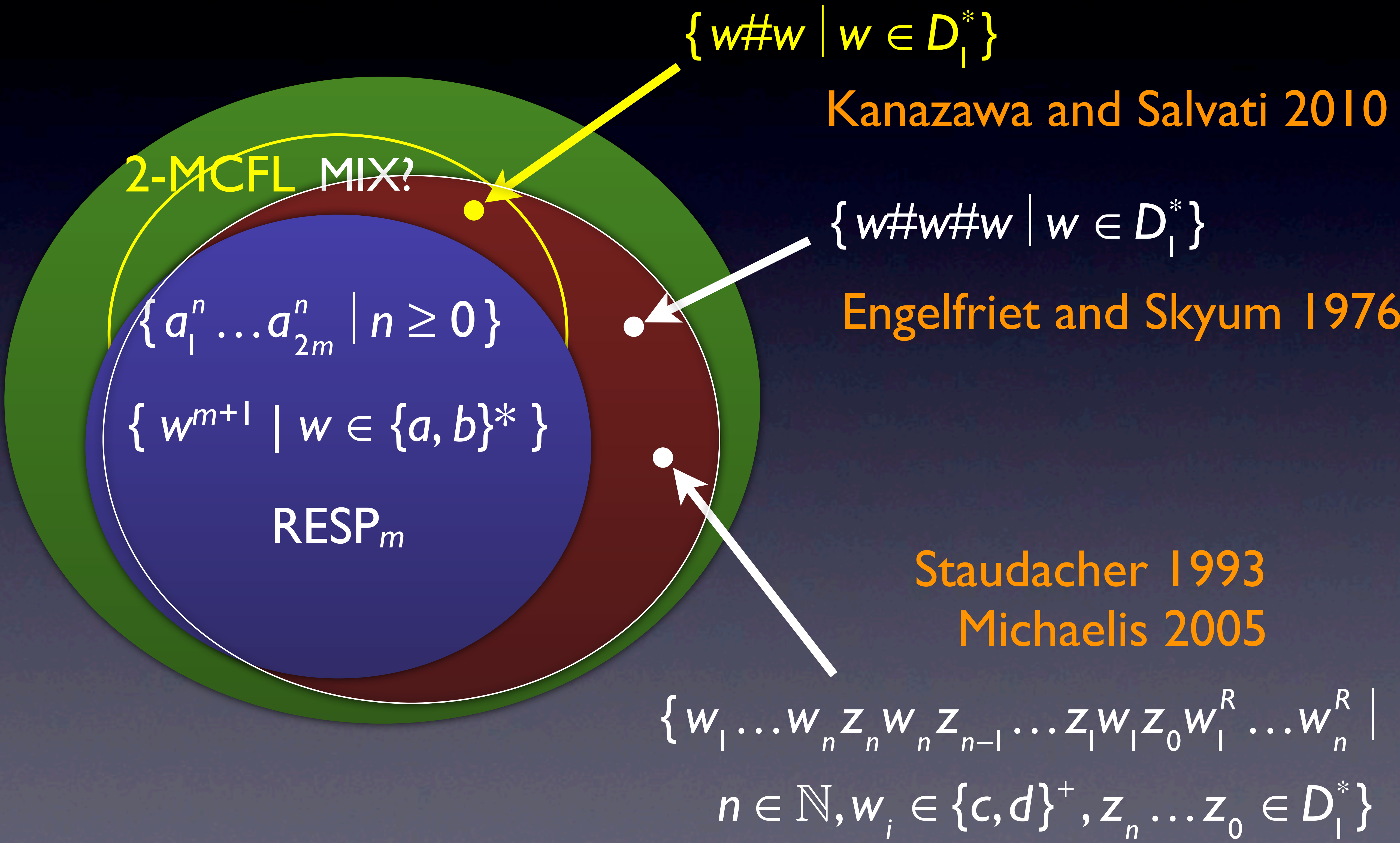
Here’s an example of a 2-MCFT$_{sp}$(1).

$$\mathbf{CT}_{k+1} = \{ L_1 \cap (\widehat{L_2})^{*,c} \mid L_1 \in \mathrm{LOC}, L_2 \in \mathbf{C}_k \}$$
$$\subseteq \{ L_1 \cap (\widehat{L_2})^{*,c} \mid L_1 \in \mathrm{LOC}, L_2 \in y(2^{k-2}\text{-MCFT}_{\mathrm{sp}}(1)) \}$$
$$\subseteq \{ L_1 \cap (\widehat{L_2})^{*,c} \mid L_1 \in \mathrm{LOC}, L_2 \in 2^{k-1}\text{-MCFL} \}$$
$$\subseteq \{ L_1 \cap L_2^{*,c} \mid L_1 \in \mathrm{LOC}, L_2 \in 2^{k-1}\text{-MCFT}_{\mathrm{sp}}(1) \}$$
$$\subseteq 2^{k-1}\text{-MCFT}_{\mathrm{sp}}(1)$$

$$\mathbf{C}_k = y\mathbf{CT}_k$$
$$\subseteq y(2^{k-2}\text{-MCFT}_{\mathrm{sp}}(1))$$
$$\subseteq 2^{k-1}\text{-MCFL}$$

The induction step goes like this.
The yield image of the tree language of an m-MCFT$_{\mathrm{sp}}$(1) is the language of a 2m-MCFL.

MCFL vs. **C**



C is closed under copying.
It’s harder to separate MCFL and C.