Making copies: Insights from computation

Greg Kobele

Reduplicative Processes in Grammar

CTM

provides means of understanding copy constructions

Makes particularly salient

External syntax

- distribution in clause

Doesn't have much to say

about internal syntax of copies, or about how copies are made

The question

What is the internal syntax of copies?

- what do copies look like
- what kinds of things can be copied
- . . . and what kind of mechanism makes copies

Not all formalisms can describe copying

In phonology

- FSOT
- SPE (à la *Kaplan & Kay*)

In syntax

- GPSG
- GB (à la *Rogers*)
- WG

Copying doesn't come *for free*

- What mechanisms can generate copies at all?
- separate this question from:
  - hierarchies of functional projections
  - the 'right' structure

How to tell?

wcw

$$\{wcw : w \in \{a, b\}^*\}$$

a sentence is in this language iff

1. it has exactly one c
2. the material to the left of c is identical to the material to its right

if you cannot describe this language

you cannot describe copying

if you can describe this language

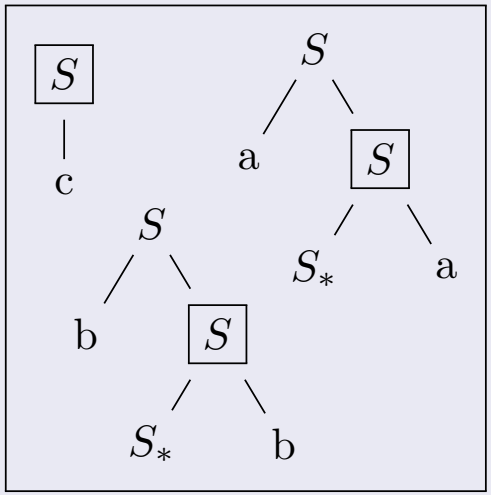
I'm interested in what you can do

(Some) mechanisms

- Adjunction
- Indexed rewriting
- Movement
- Copy movement

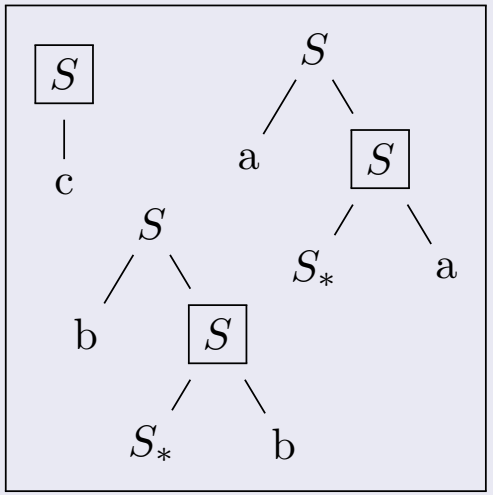
Adjunction

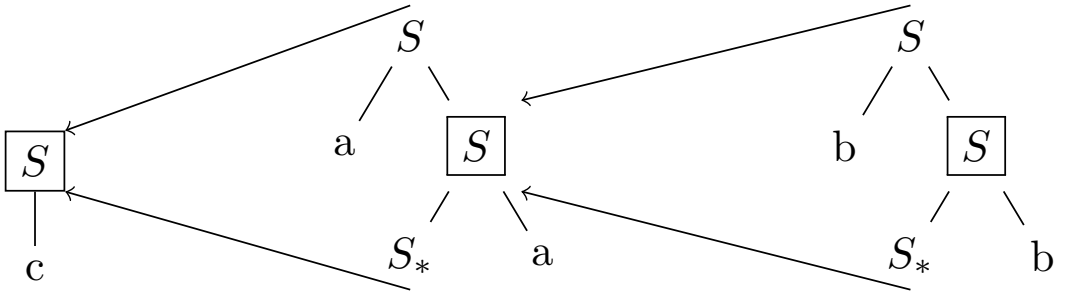
Lexicon



Adjunction

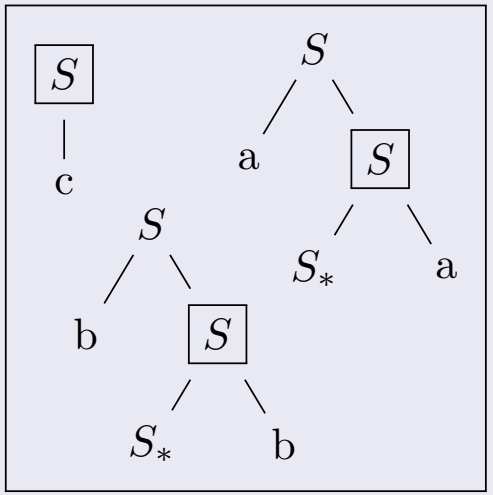
Lexicon

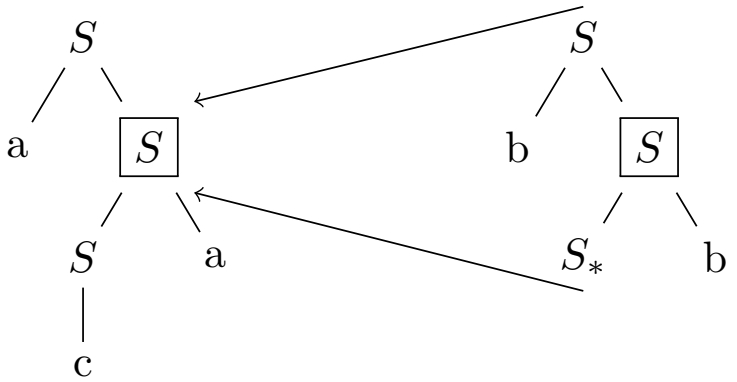




Adjunction

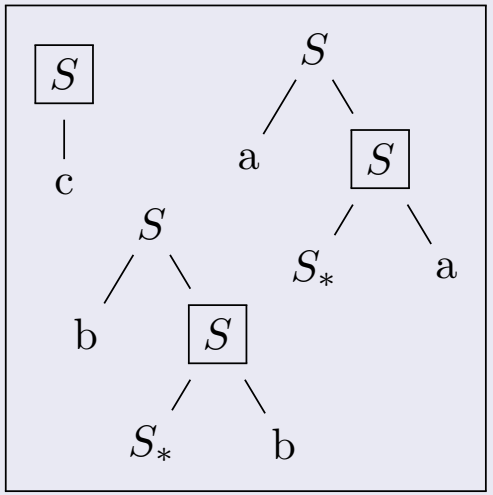
Lexicon

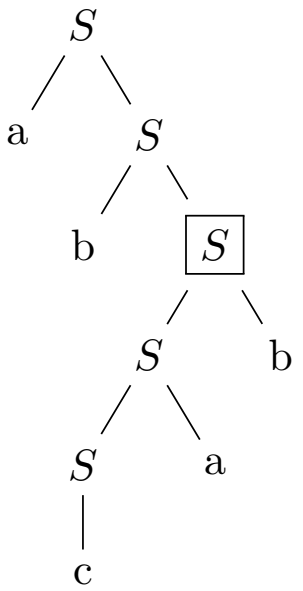




Adjunction

Lexicon

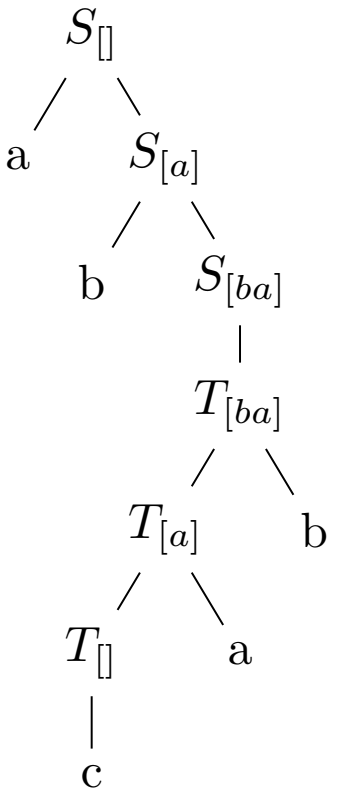




Indexed rewriting

Rules

$$S_{[\alpha]} \rightarrow xS_{[x\alpha]}$$
$$S_{[\alpha]} \rightarrow T_{[\alpha]}$$
$$T_{[x\alpha]} \rightarrow T_{[\alpha]}x$$
$$T_{[]} \rightarrow c$$



Derivation

$$S_{[]} \Rightarrow aS_{[a]} \Rightarrow abS_{[ba]} \Rightarrow abT_{[ba]} \Rightarrow abT_{[a]}b \Rightarrow abT_{[]}ab \Rightarrow abcab$$

Movement

Lexicon

a :: *x *z A z a :: *A *y x y b :: *x *z B z b :: *B *y x y	ϵ :: *x *z w c :: *w *y s ϵ :: x z y

Movement

Lexicon

a :: *x *z A z a :: *A *y x y b :: *x *z B z b :: *B *y x y	ϵ :: *x *z w c :: *w *y s ϵ :: x z y

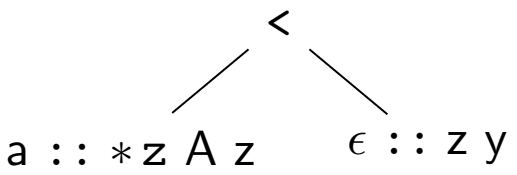
a :: *x *z A z

ϵ :: x z y

Movement

Lexicon

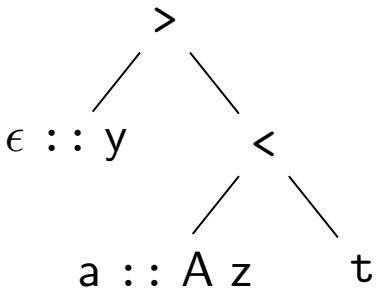
a :: ∗x ∗z A z a :: ∗A ∗y x y b :: ∗x ∗z B z b :: ∗B ∗y x y	ϵ :: ∗x ∗z w c :: ∗w ∗y s ϵ :: x z y



Movement

Lexicon

a :: *x *z A z a :: *A *y x y b :: *x *z B z b :: *B *y x y	ϵ :: *x *z w c :: *w *y s ϵ :: x z y

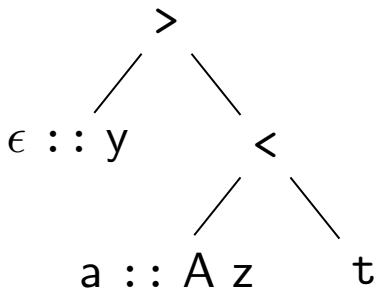


Movement

Lexicon

a :: *x *z A z a :: *A *y x y b :: *x *z B z b :: *B *y x y	ϵ :: *x *z w c :: *w *y s ϵ :: x z y

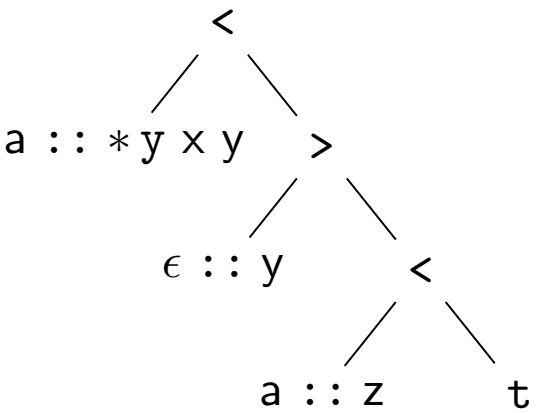
a :: *A *y x y



Movement

Lexicon

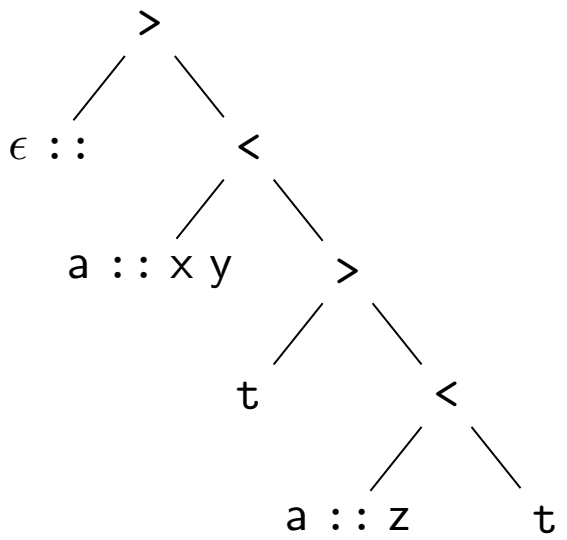
a :: *x *z A z a :: *A *y x y b :: *x *z B z b :: *B *y x y	ϵ :: *x *z w c :: *w *y s ϵ :: x z y



Movement

Lexicon

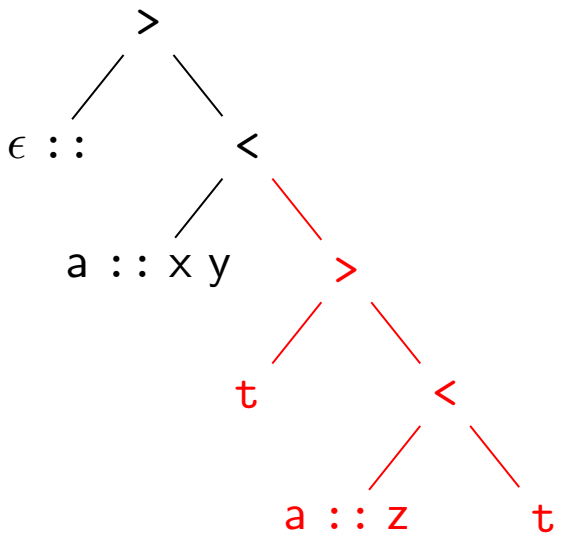
a :: *x *z A z a :: *A *y x y b :: *x *z B z b :: *B *y x y	ϵ :: *x *z w c :: *w *y s ϵ :: x z y



Movement

Lexicon

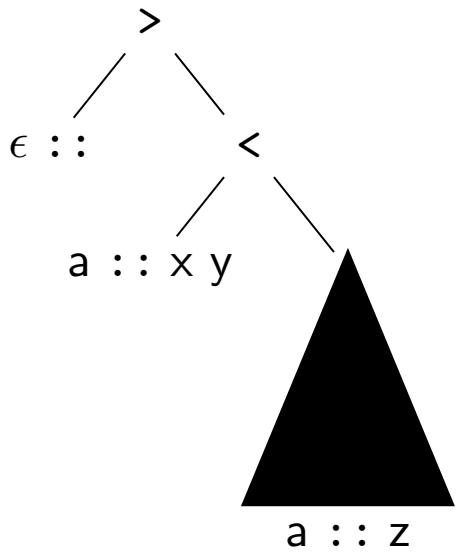
a :: *x *z A z a :: *A *y x y b :: *x *z B z b :: *B *y x y	ϵ :: *x *z w c :: *w *y s ϵ :: x z y



Movement

Lexicon

a :: =x =z A z a :: =A =y x y b :: =x =z B z b :: =B =y x y	ϵ :: =x =z w c :: =w =y s ϵ :: x z y

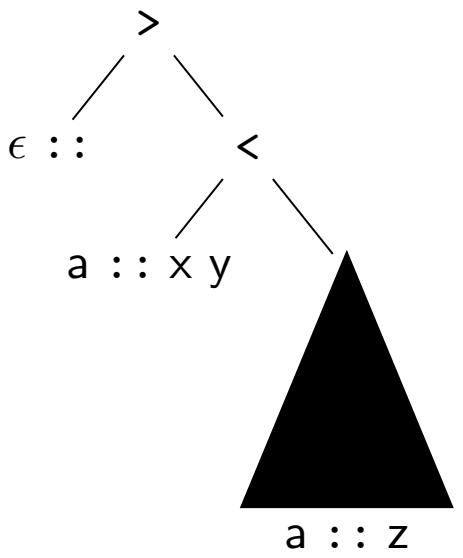


Movement

Lexicon

a :: ∗x ∗z A z a :: ∗A ∗y x y b :: ∗x ∗z B z b :: ∗B ∗y x y	ϵ :: ∗x ∗z w c :: ∗w ∗y s ϵ :: x z y

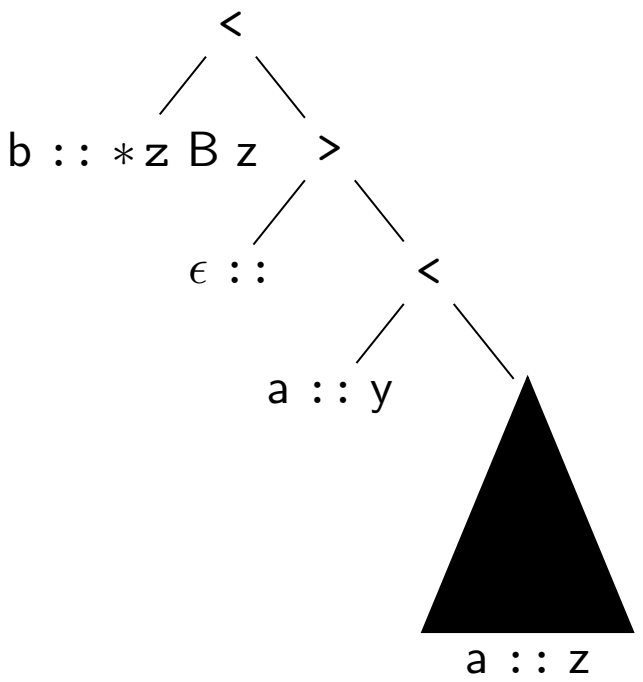
b :: ∗x ∗z B z



Movement

Lexicon

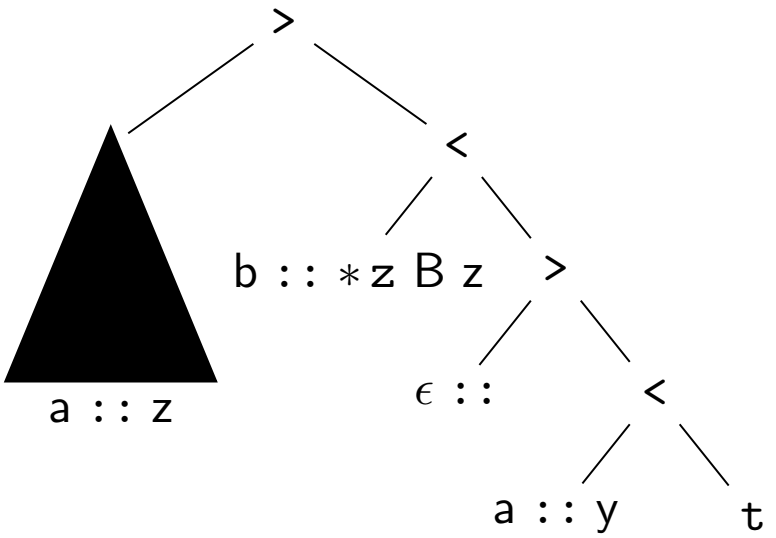
a :: *x *z A z a :: *A *y x y b :: *x *z B z b :: *B *y x y	ϵ :: *x *z w c :: *w *y s ϵ :: x z y



Movement

Lexicon

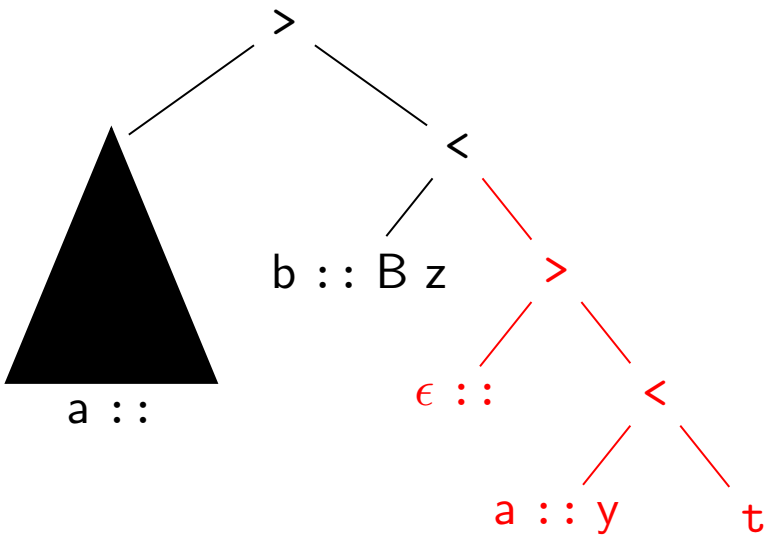
a :: =x =z A z a :: =A =y x y b :: =x =z B z b :: =B =y x y	ϵ :: =x =z w c :: =w =y s ϵ :: x z y



Movement

Lexicon

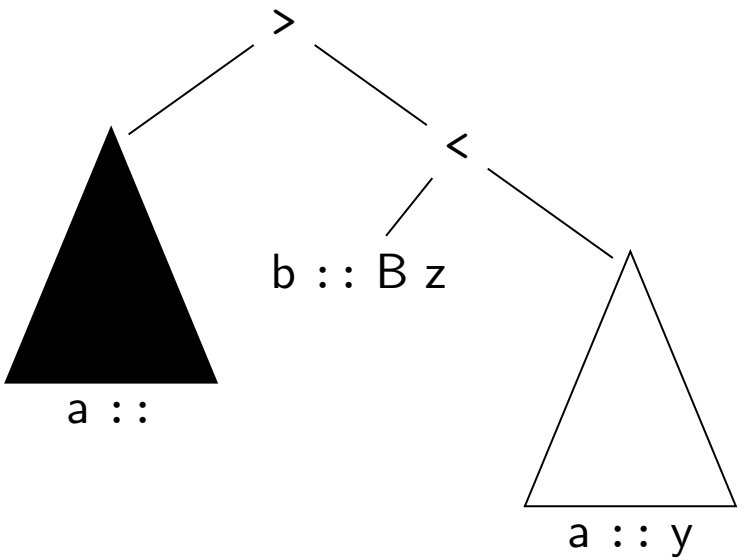
a :: *x *z A z a :: *A *y x y b :: *x *z B z b :: *B *y x y	ϵ :: *x *z w c :: *w *y s ϵ :: x z y



Movement

Lexicon

a :: *x *z A z a :: *A *y x y b :: *x *z B z b :: *B *y x y	ϵ :: *x *z w c :: *w *y s ϵ :: x z y

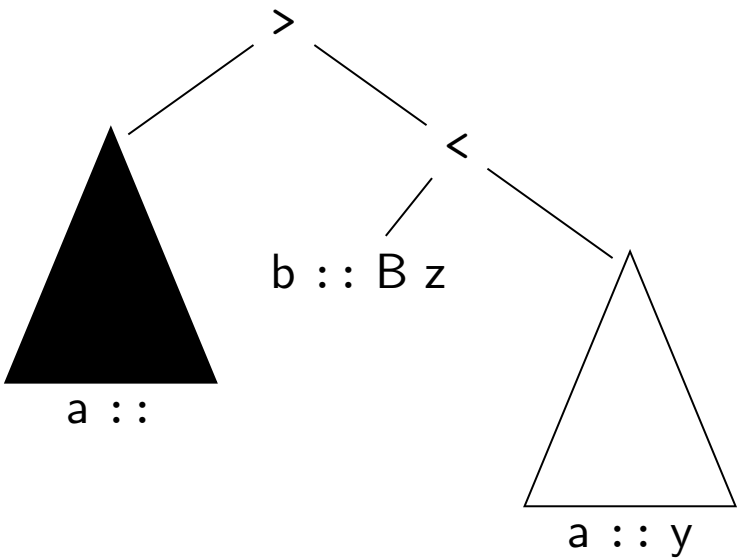


Movement

Lexicon

a :: =x =z A z a :: =A =y x y b :: =x =z B z b :: =B =y x y	ϵ :: =x =z w c :: =w =y s ϵ :: x z y

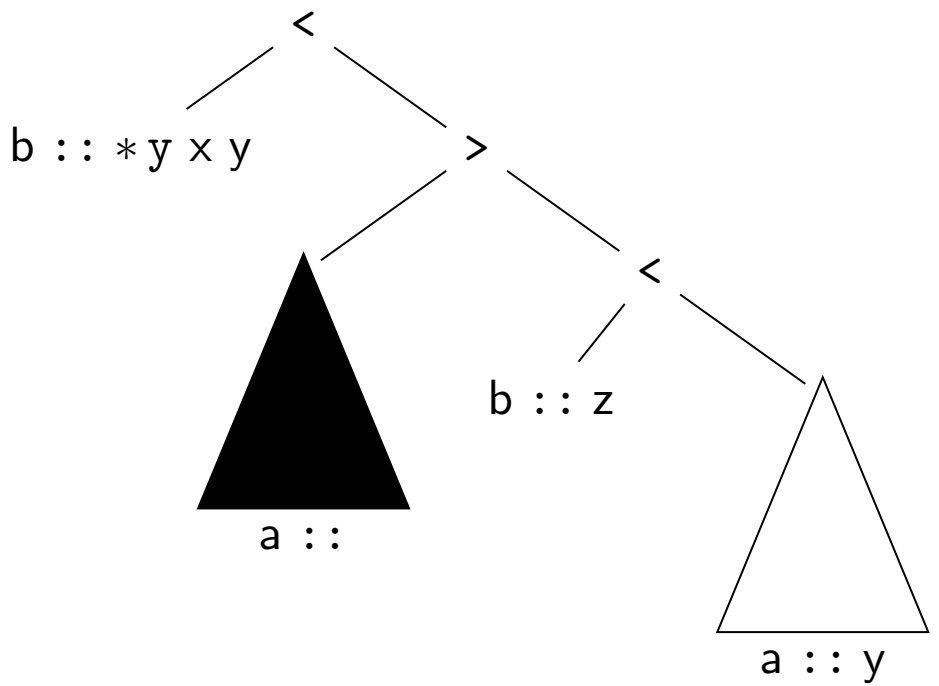
b :: =B =y x y



Movement

Lexicon

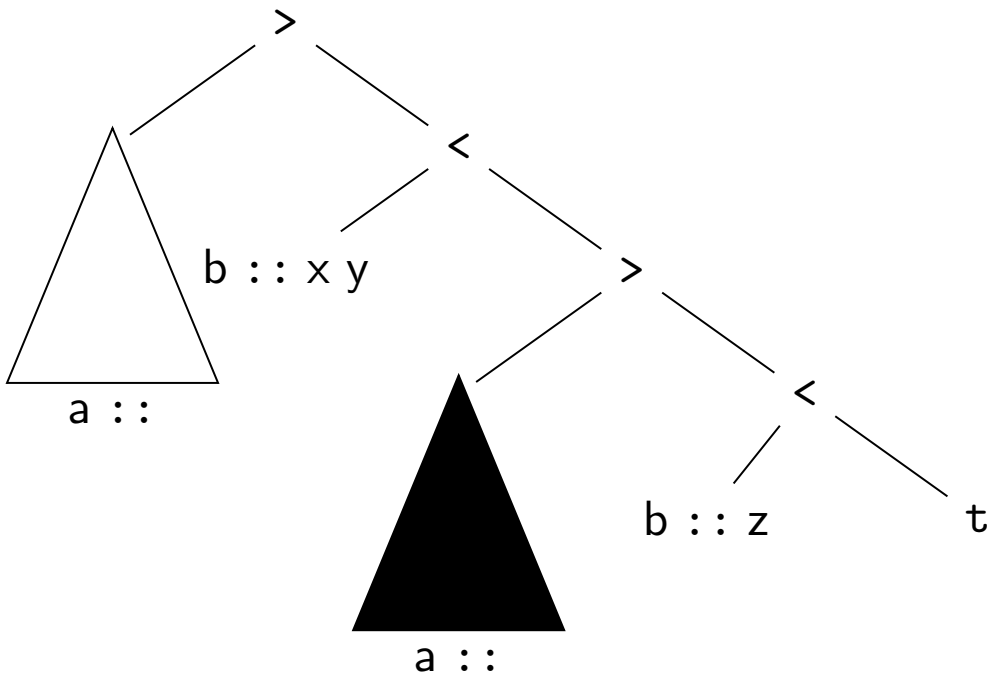
a :: ∗x ∗z A z a :: ∗A ∗y x y b :: ∗x ∗z B z b :: ∗B ∗y x y	ϵ :: ∗x ∗z w c :: ∗w ∗y s ϵ :: x z y



Movement

Lexicon

a :: *x *z A z a :: *A *y x y b :: *x *z B z b :: *B *y x y	ϵ :: *x *z w c :: *w *y s ϵ :: x z y

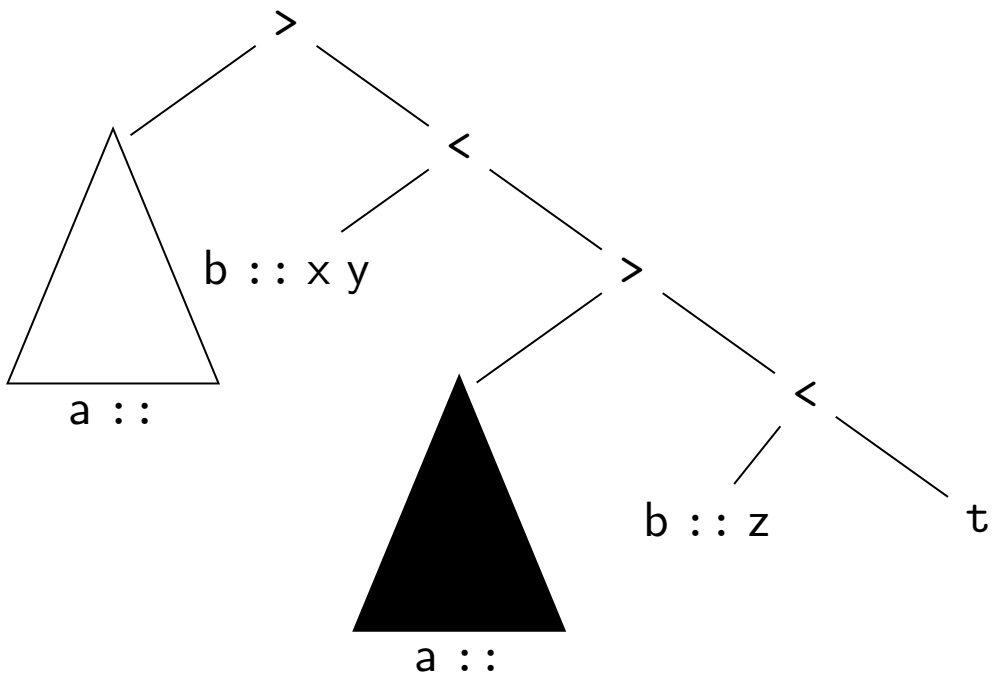


Movement

Lexicon

a :: *x *z A z a :: *A *y x y b :: *x *z B z b :: *B *y x y	ϵ :: *x *z w c :: *w *y s ϵ :: x z y

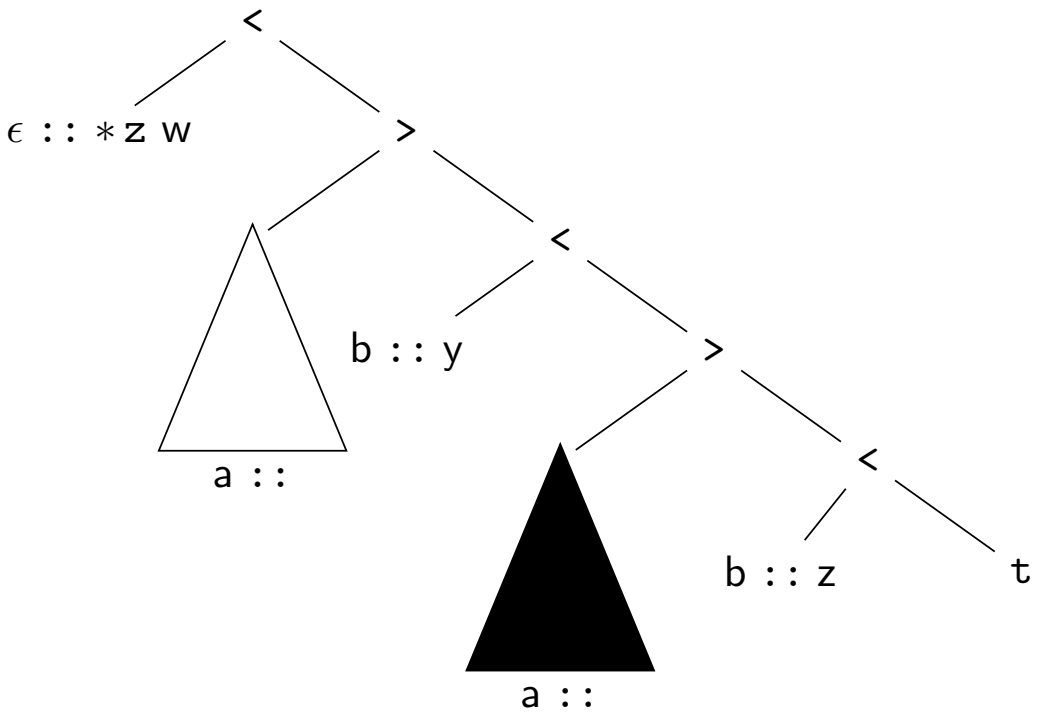
ϵ :: *x *z w



Movement

Lexicon

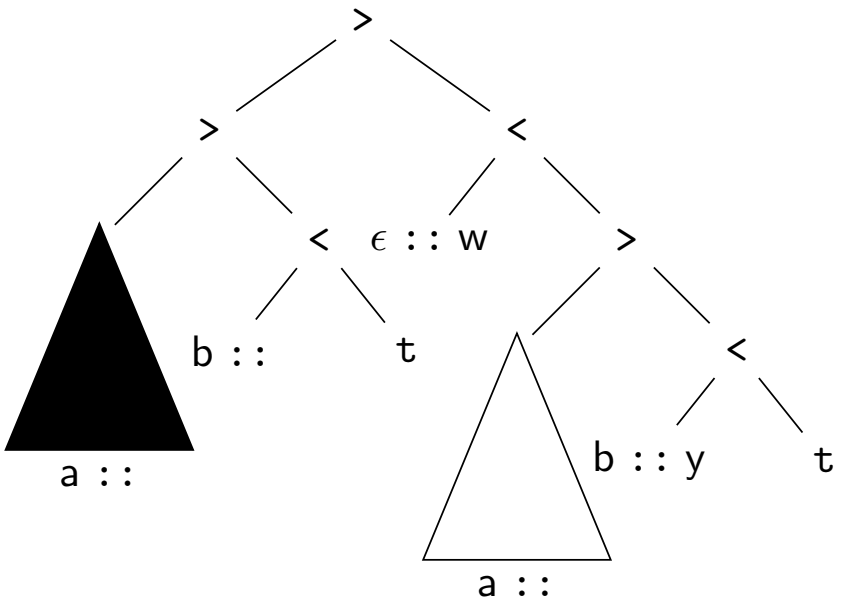
a :: *x *z A z a :: *A *y x y b :: *x *z B z b :: *B *y x y	ϵ :: *x *z w c :: *w *y s ϵ :: x z y



Movement

Lexicon

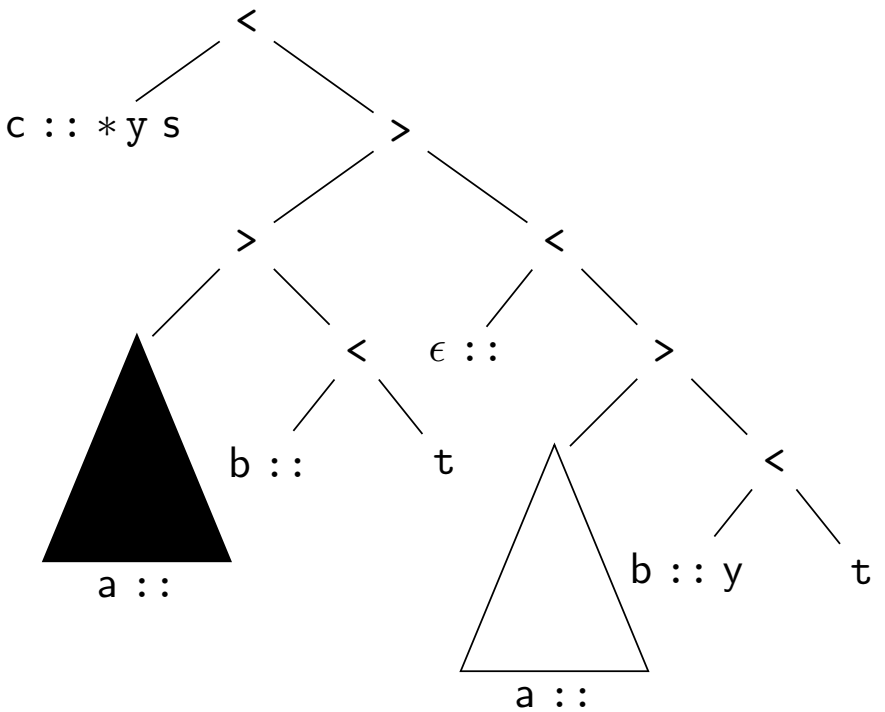
a :: =x =z A z a :: =A =y x y b :: =x =z B z b :: =B =y x y	ϵ :: =x =z w c :: =w =y s ϵ :: x z y



Movement

Lexicon

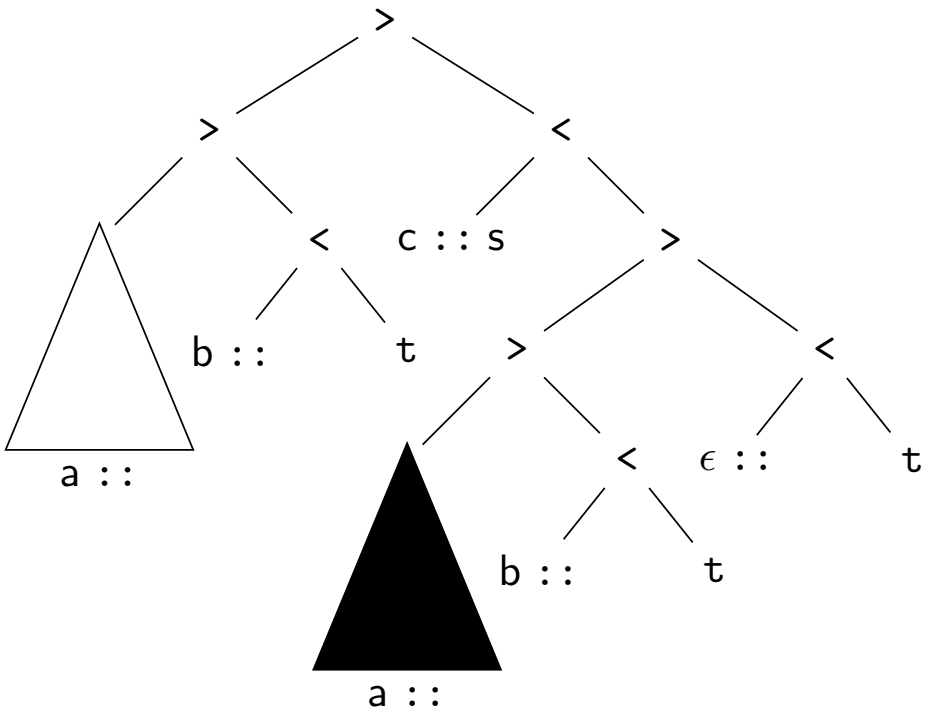
a :: *x *z A z a :: *A *y x y b :: *x *z B z b :: *B *y x y	ϵ :: *x *z w c :: *w *y s ϵ :: x z y



Movement

Lexicon

a :: *x *z A z a :: *A *y x y b :: *x *z B z b :: *B *y x y	ϵ :: *x *z w c :: *w *y s ϵ :: x z y



Copying

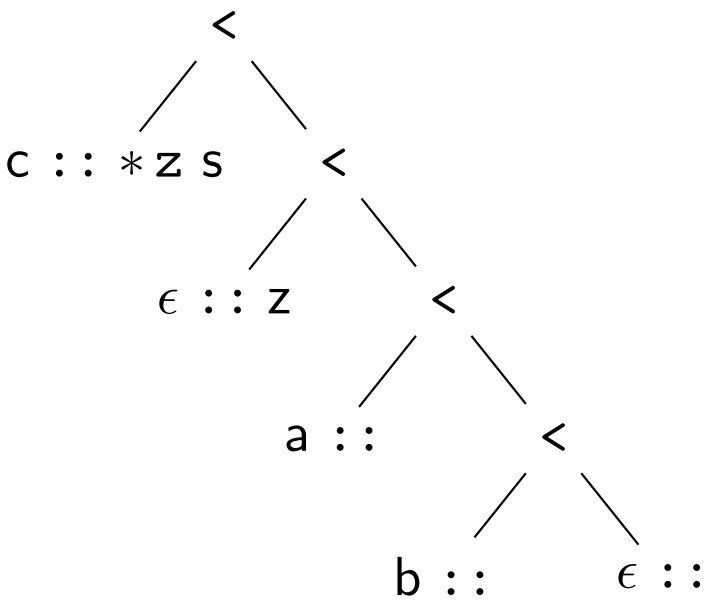
Lexicon

a :: $*\mathrm{x}\ \mathrm{x}$	ϵ :: x	
b :: $*\mathrm{x}\ \mathrm{x}$	ϵ :: $*\mathrm{x}\ \mathrm{s}\ \mathrm{z}$	
	c :: $*\hat{\mathrm{s}}\ *\mathrm{z}\ \mathrm{s}$	

Copying

Lexicon

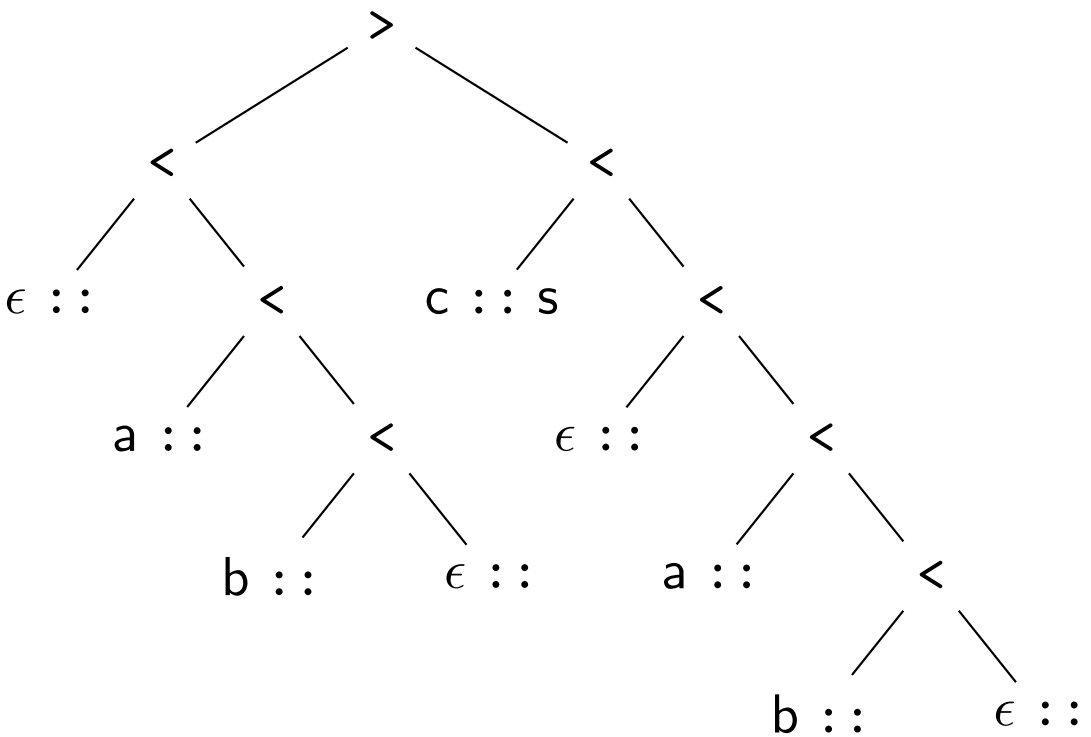
a :: ∗x x ϵ :: x
b :: ∗x x ϵ :: ∗x s z
c ::∗ŝ ∗z s



Copying

Lexicon

a :: $*\mathrm{x}\,\mathrm{x}$	ϵ :: x	
b :: $*\mathrm{x}\,\mathrm{x}$	ϵ :: $*\mathrm{x}\,\mathrm{s}\,\mathrm{z}$	
	c :: $*\hat{\mathrm{s}}\,*\mathrm{z}\,\mathrm{s}$	



2-MCFL$_{wn}$

Different mechanisms, *same patterns*

- TAG
- CCG
- Tree-local MCTAG
- MCFTG
- LIG

$\mathrm{MCFL}_{\mathrm{wn}}$

Different mechanisms, *same patterns*

- ACG(2,3)
- CFTG

MCFL

Different mechanisms, *same patterns*

- MG
- Set-local MCTAG
- ACG(2,>3)
- MCFG
- STR(CFHG)
- MSOT(REG)

PMCFL

Different mechanisms, *same patterns*

- CMG
- almost-linear ACG(2,>3)
- PMCFG

Take home message

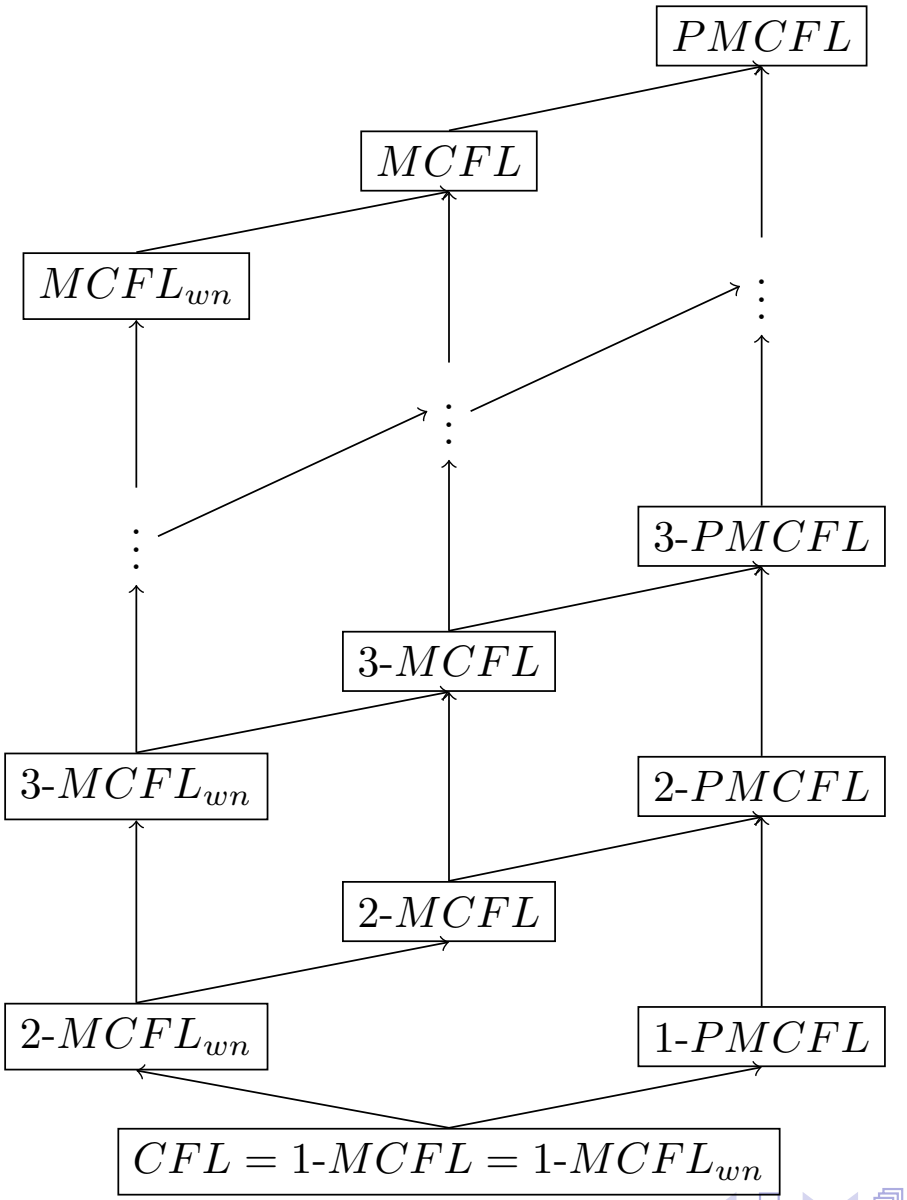
Patterns are real

grammars are a *mode of presentation* of underlying patterns

Meaningful to ask

which patterns are instantiated in NL?

Inclusion diagram



Separating MCFL_{wn} from MCFL

The copying theorem (Kanazawa & Salvati, 2010)

if $L = \{wcw \mid w \in L_0\}$ is a MCFL_{wn} then L_0 is generable by a *non-branching* MCFG

Translation: only simple things (L_0) can be copied

L_0 cannot necessarily involve branching structures

- $*$give [a good book] [to a nice student]

The first question

Question 1

Are only structurally trivial things copyable?

- gk conjectures: no!

A *no* answer means

TAG, CCG, etc are wrong

A *yes* answer means

Huge confirmation of a deep prediction of TAG, CCG

Separating MCFL from PMCFL

Semilinearity

intuition: Operations add a fixed, finite amount of structural elements

Translation: copies cannot be copied

$$a \Rightarrow aa \not\Rightarrow \mathbf{aa}aa$$

Exponential growth is just recursive doubling

The second question

Question 2

Can copies be made of copies (of copies . . .)?

- gk conjectures: yes!

A *no* answer means

Huge confirmation of mild-context sensitivity hypothesis

A *yes* answer means

Hypothesis of mild-context sensitivity (Joshi, 85) is wrong
. . . and with it most restrictive grammar formalisms

The limits of full copying

Non-semilinear patterns

must involve exact copying

Translation: recursive copying must be exact

- only recursive copying is truly copying
- non-recursive copying can just as easily generate *opposite* 'copies'

$$wc\overline{w} = \{ac\overline{a}, abc\overline{a}\overline{b}, abbc\overline{a}\overline{bb}, \ldots\}$$

- let $\overline{a} = b$ and $\overline{b} = a$; can't have

$$a \rightarrow \underline{b}a \rightarrow \underline{ab}ba \rightarrow \underline{baab}abba \rightarrow \underline{abbabaab}baababba$$

The third question

Question 3

Can we have non-exact (i.e. structure-only) copying in recursive copy constructions?

- gk conjectures: No!

Conclusions

- There are deep and fundamental questions that we need to ask about copying
  - is copying always small?
  - is copying only of simplexes?
  - is copying recursive?
- The answers to these will sink various approaches to grammar
- Once we know what patterns there are, we can focus on coming up with an elegant description language

Thank you