Concise representations of regular languages

Bruno Guillon

INRIA Lille, équipe Links

December 6, 2018

— *Journées DeLTA – Bordeaux* —

mainly joint work with Giovanni Pighizzini and Luca Prigioniero, University of Milan

Descriptional complexity

- study of size of models recognizing languages

  e.g., number of transitions of an automaton

Descriptional complexity

- study of size of models recognizing languages

  e.g., number of transitions of an automaton

- study of transformations,

  e.g., determinization of 1NFA costs exponential
  cannot be avoided in the worst case

Descriptional complexity

- study of size of models recognizing languages

  e.g., number of transitions of an automaton

- study of transformations,

  e.g., determinization of 1NFA costs exponential
  cannot be avoided in the worst case

- study of natural operations,

  e.g., transforming a 1DFA in order to recognize the reverse of
  its accepted language has exponential size cost as well

Descriptional complexity

- study of size of models recognizing languages

  e.g., number of transitions of an automaton

- study of transformations,

  e.g., determinization of 1NFA costs exponential
  cannot be avoided in the worst case

- study of natural operations,

  e.g., transforming a 1DFA in order to recognize the reverse of
  its accepted language has exponential size cost as well

This talk:

- focus on regular languages

Descriptional complexity

- study of size of models recognizing languages

  e.g., number of transitions of an automaton

- study of transformations,

  e.g., determinization of 1NFA costs exponential cannot be avoided in the worst case

- study of natural operations,

  e.g., transforming a 1DFA in order to recognize the reverse of its accepted language has exponential size cost as well

This talk:

- focus on regular languages
- particular attention paid to 1-*limited automata*

Context-free ability: describe recursive structure

Context-free

CFG PDA

Context-free ability: describe recursive structure

Context-free

CFG PDA

$\mathbf{G}: S \to (S) | SS | \varepsilon$

Context-free ability: describe recursive structure

Context-free

CFG

PDA

G : $S \to (S) \mid SS \mid \varepsilon$

Definition (NSE [Chomsky 1959])

G is *self-embedding* if for some X, $X \overset{*}{\Rightarrow} \alpha X \beta$ with both α, β nonempty.

Otherwise, **G** is non-self-embedding.

Context-free ability: describe recursive structure

Context-free

CFG

$\downarrow$

NSEG

PDA

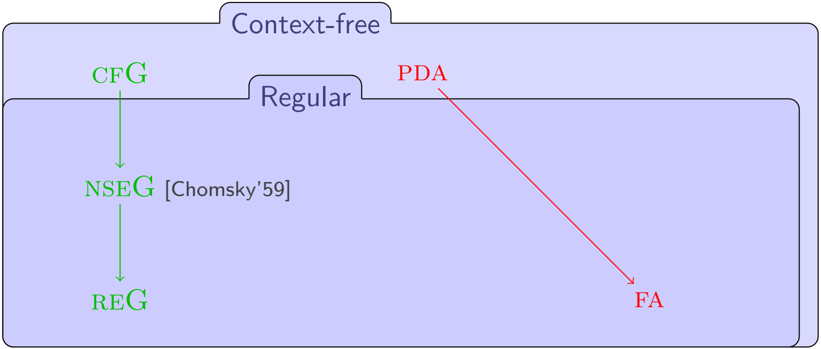
G : $S \to (S) \mid SS \mid \varepsilon$

Definition (NSE [Chomsky 1959])

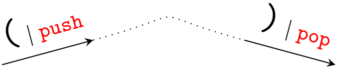
G is *self-embedding* if for some X, $X \stackrel{*}{\Rightarrow} \alpha X \beta$ with both α, β nonempty.

Otherwise, **G** is non-self-embedding.

Context-free ability: describe recursive structure



G : $S \to (S)\,|\,SS\,|\,\varepsilon$

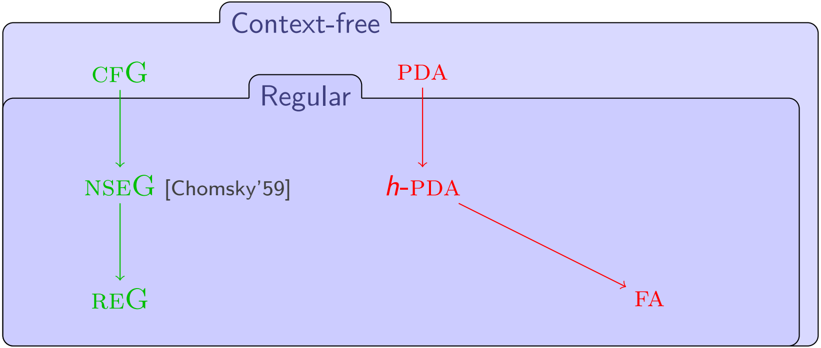


Definition (NSE [Chomsky 1959])

G is *self-embedding* if for some X,
$X \overset{*}{\Rightarrow} \alpha X \beta$ with both α, β nonempty.

Otherwise, **G** is non-self-embedding.

Context-free ability: describe recursive structure



$\mathbf{G}:\ S \to (S)\,|\,SS\,|\,\varepsilon$

Definition (NSE [Chomsky 1959])

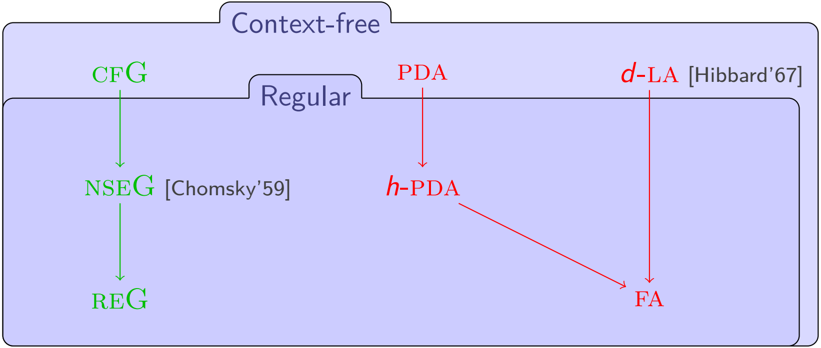
G is *self-embedding* if for some X, $X \overset{*}{\Rightarrow} \alpha X \beta$ with both α, β nonempty.

Otherwise, **G** is non-self-embedding.

Definition (h-PDA)

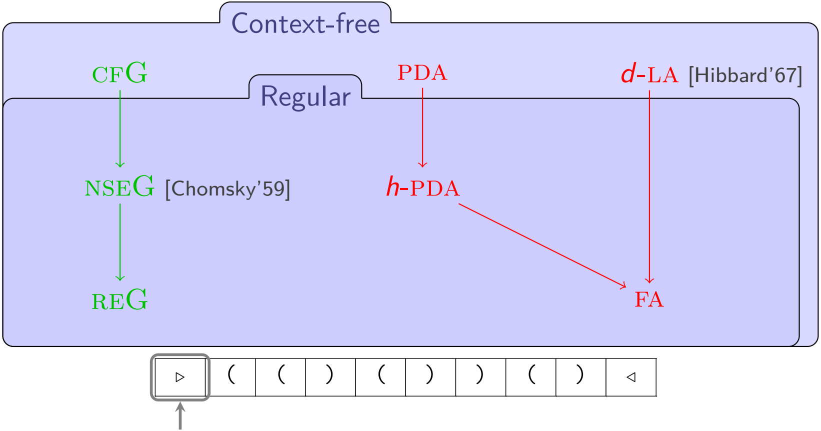
An *h-height* PDA is a PDA with stack size $\leq h \in \mathbb{N}$.

Context-free ability: describe recursive structure



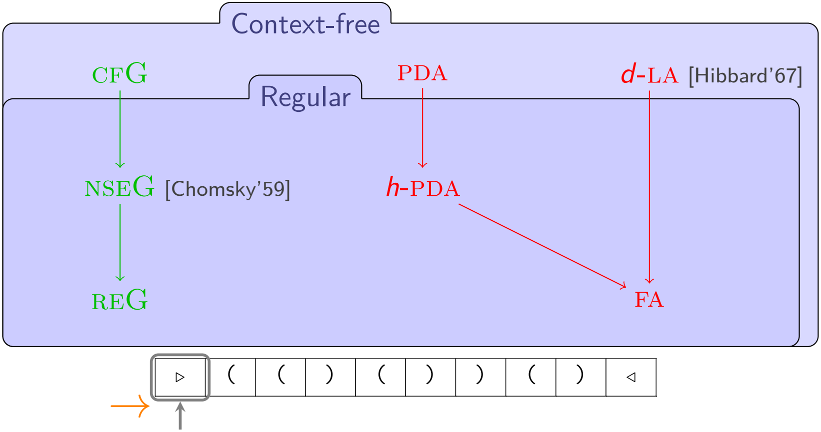
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



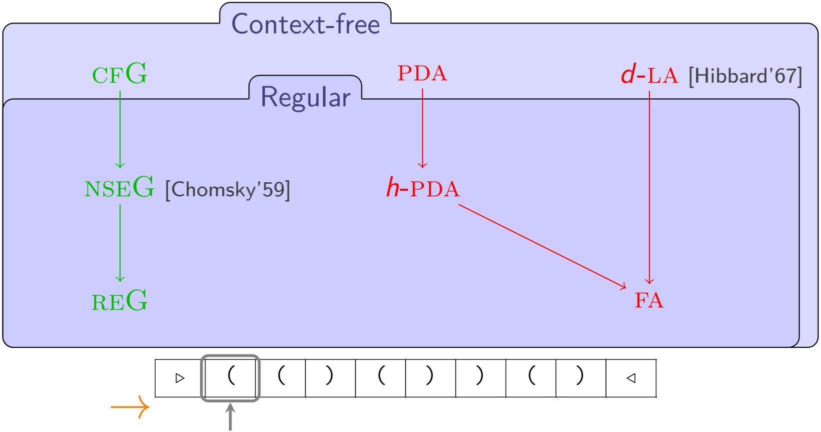
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



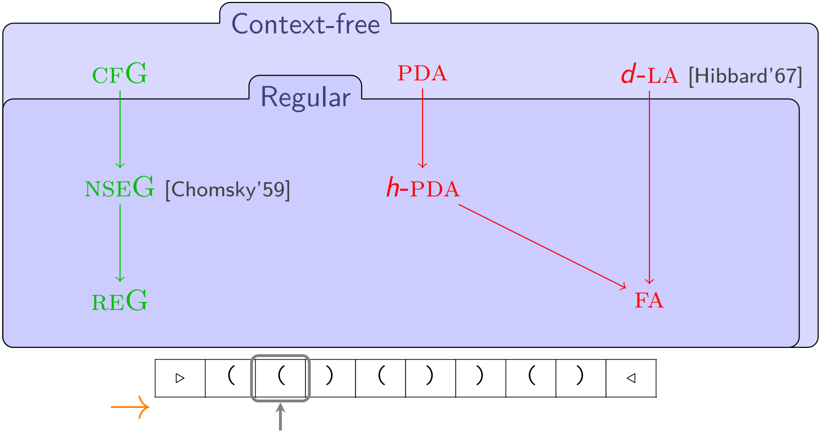
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



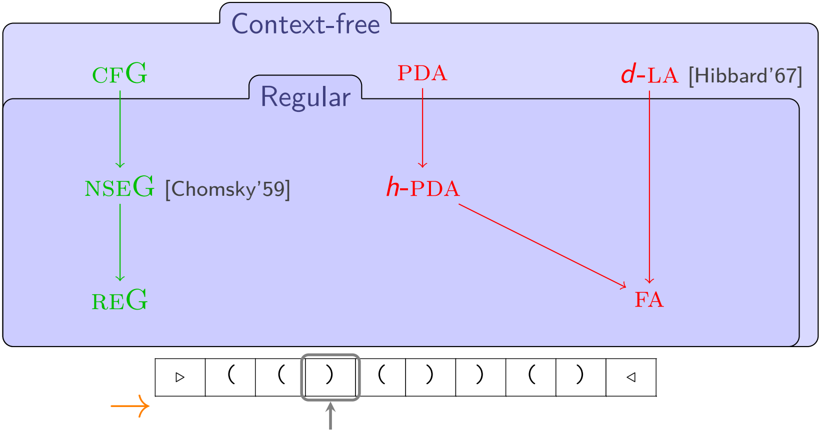
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



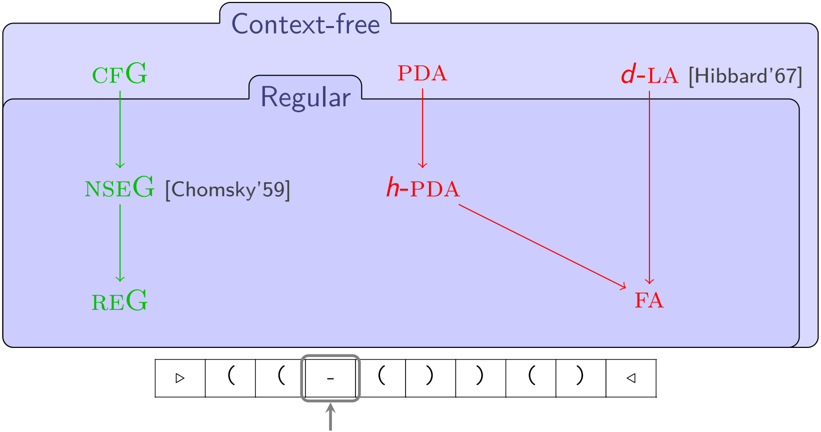
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



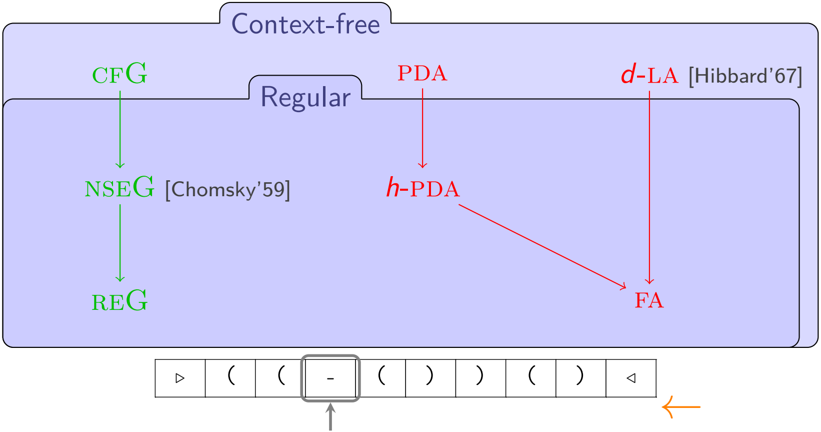
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



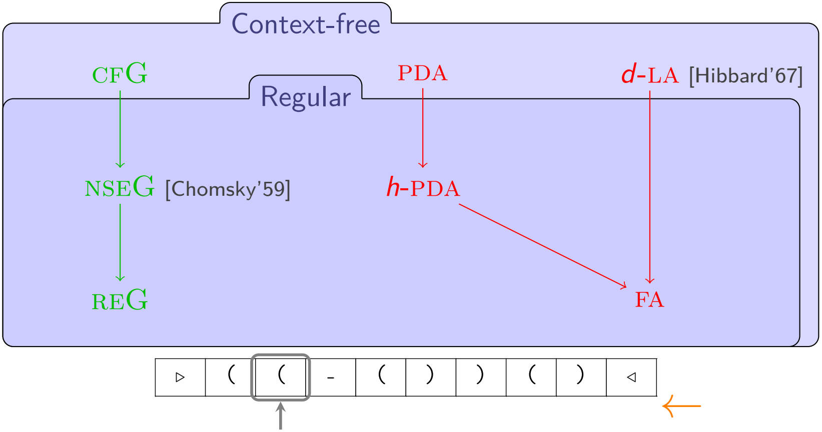
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



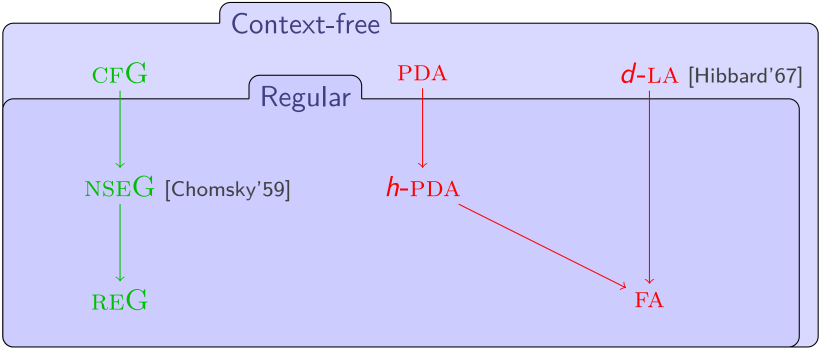
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

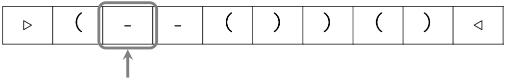
Context-free ability: describe recursive structure



Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

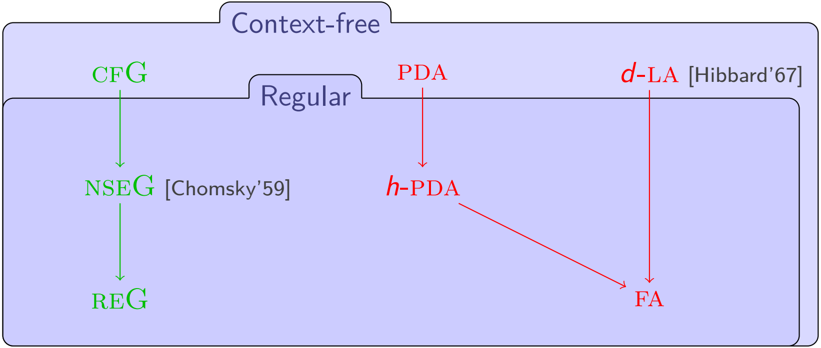
Context-free ability: describe recursive structure

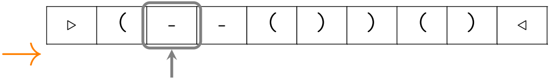




Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

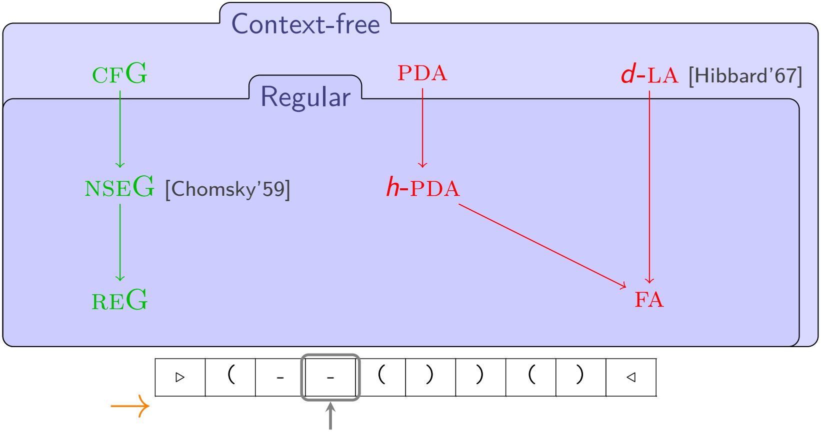
Context-free ability: describe recursive structure





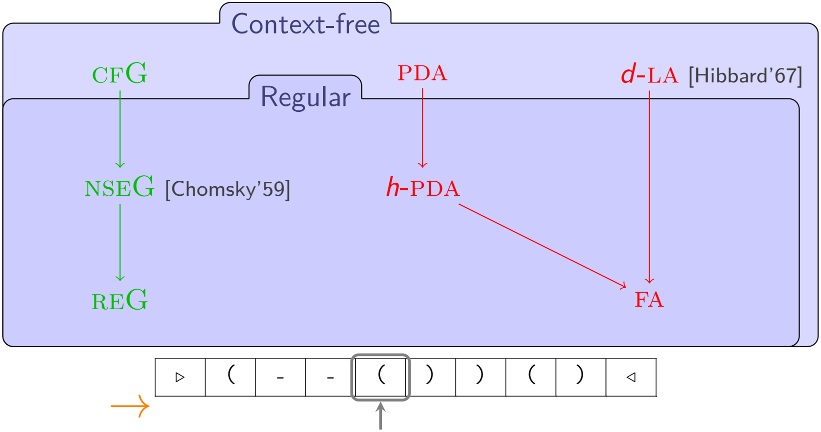
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



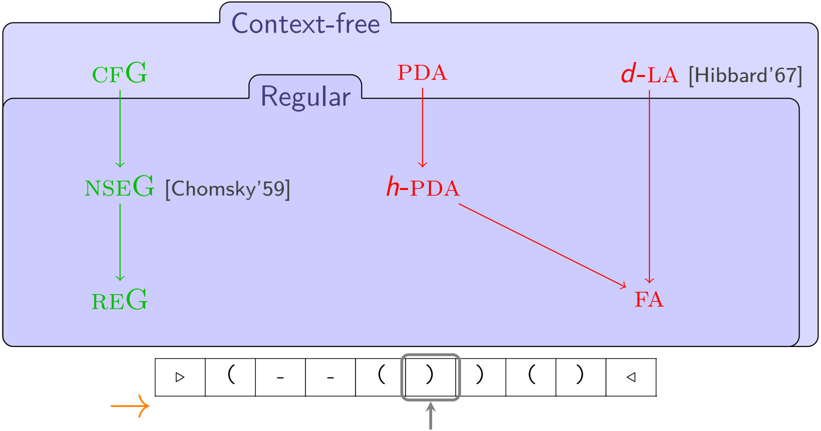
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



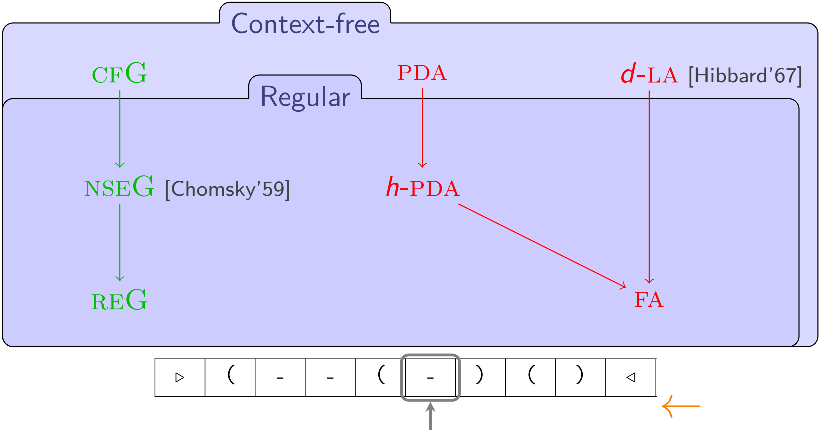
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



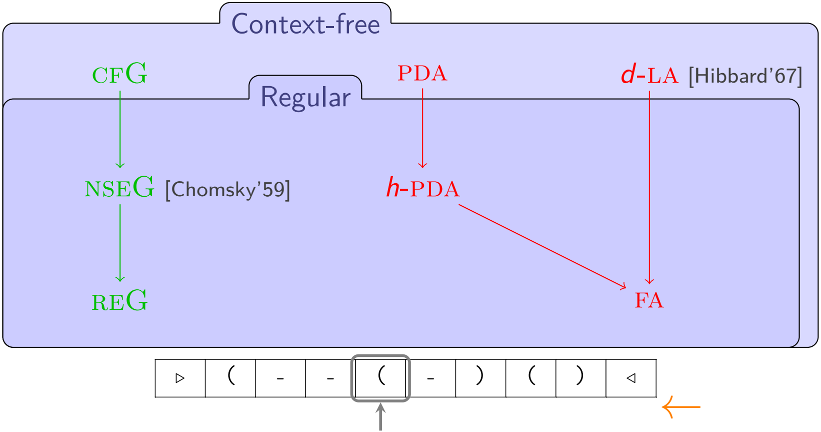
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



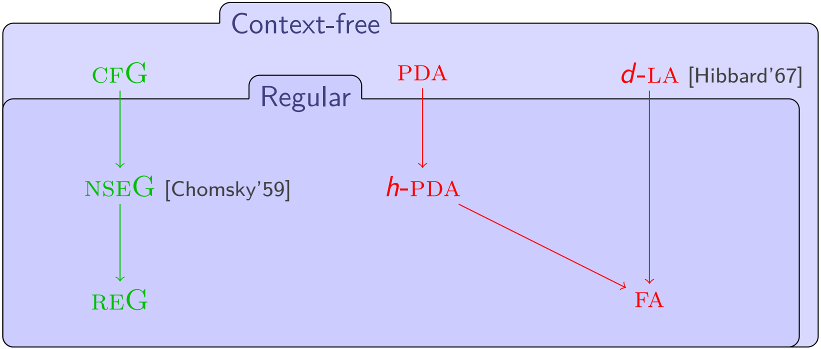
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

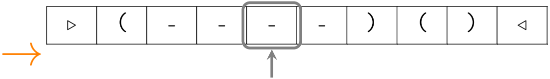
Context-free ability: describe recursive structure



Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

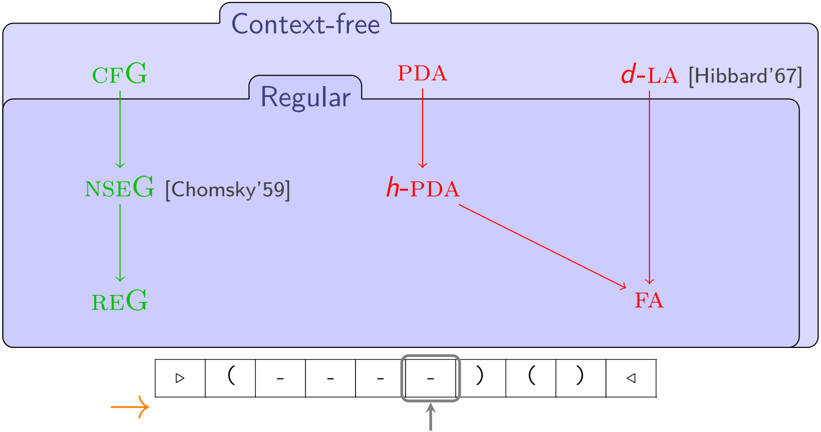
Context-free ability: describe recursive structure





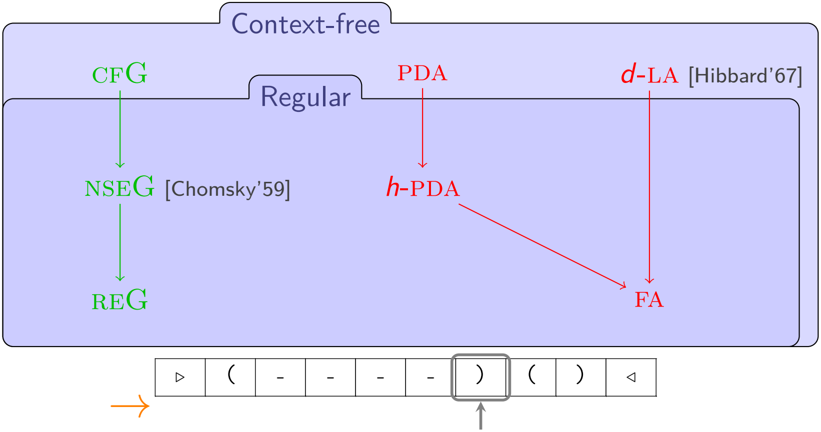
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



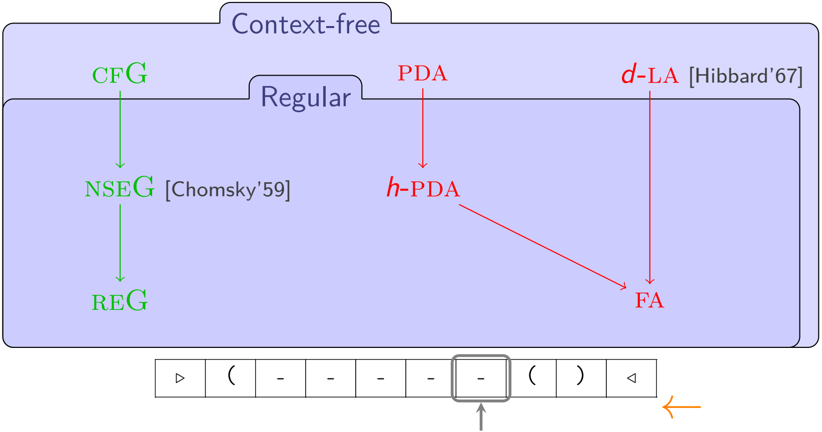
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



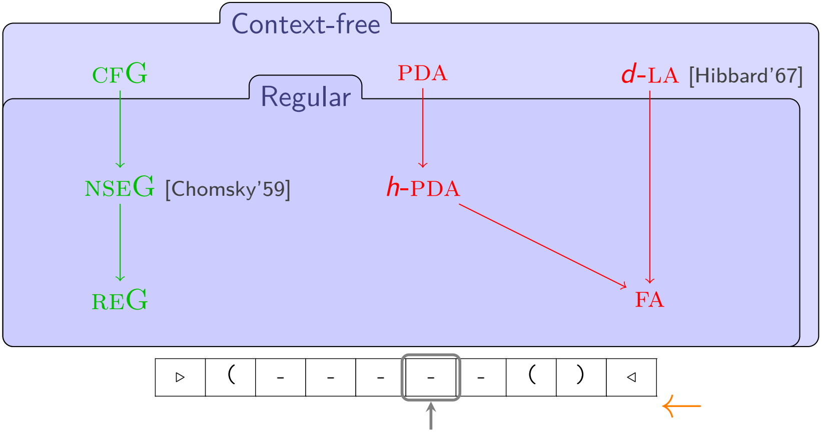
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



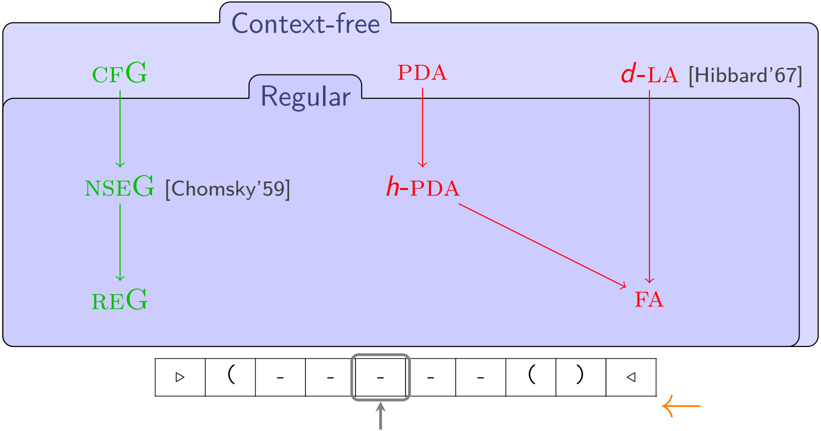
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



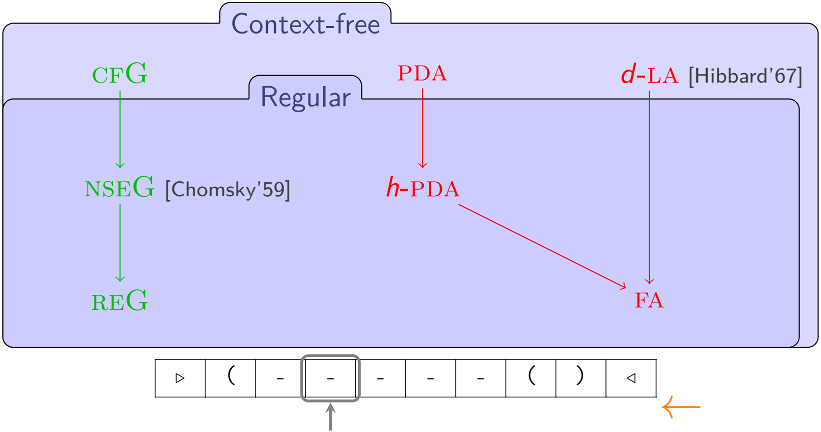
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



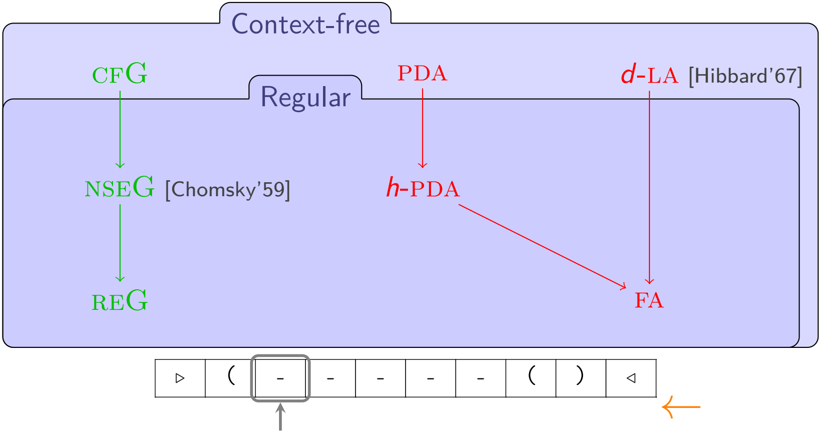
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



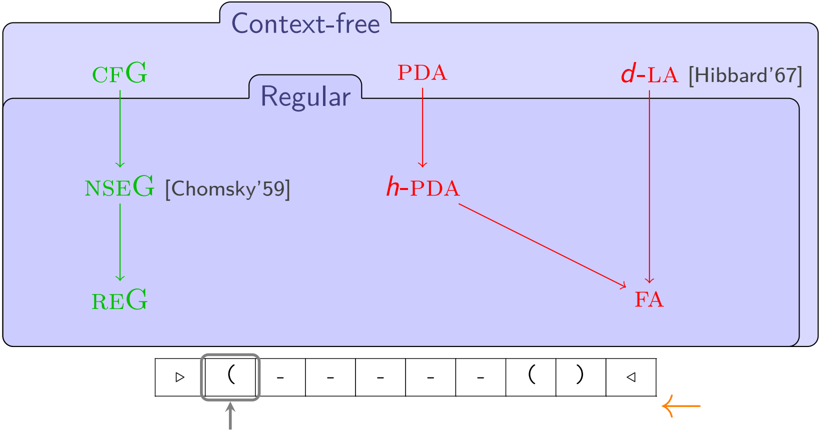
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



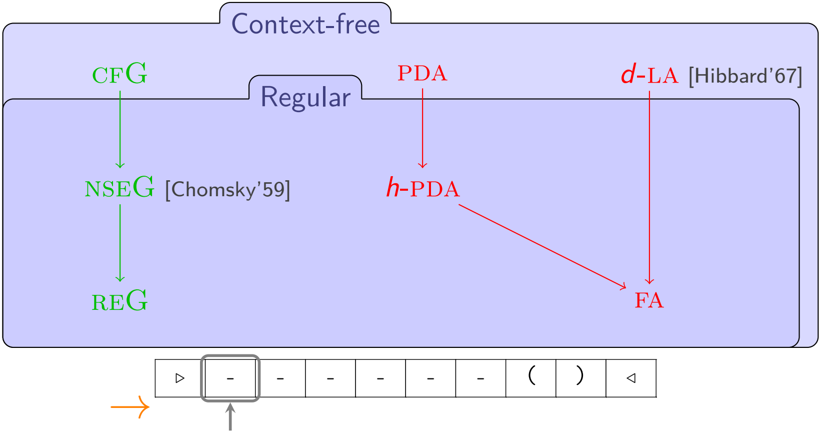
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



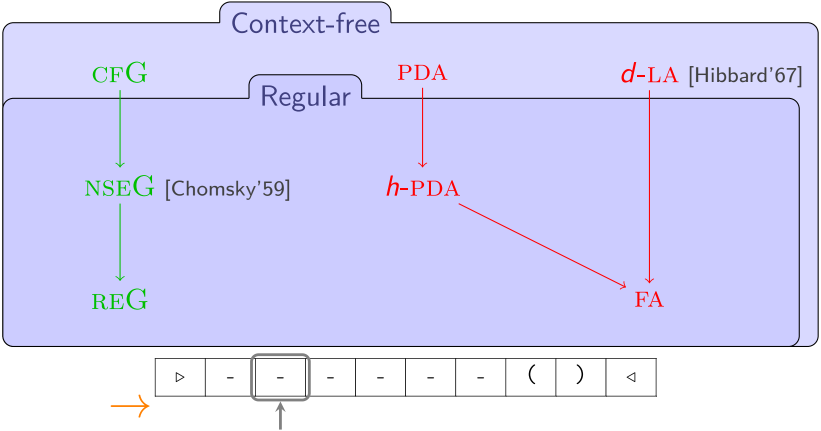
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



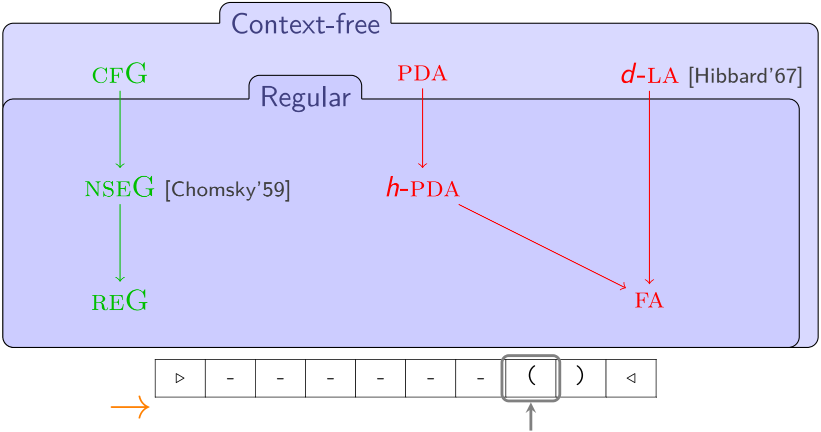
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



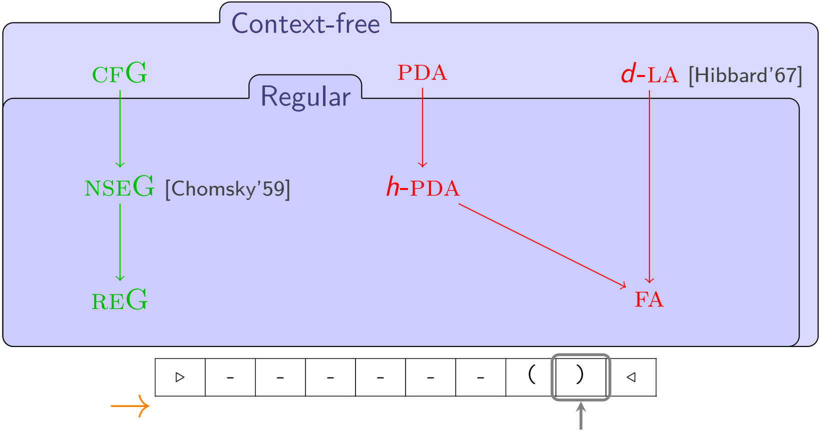
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



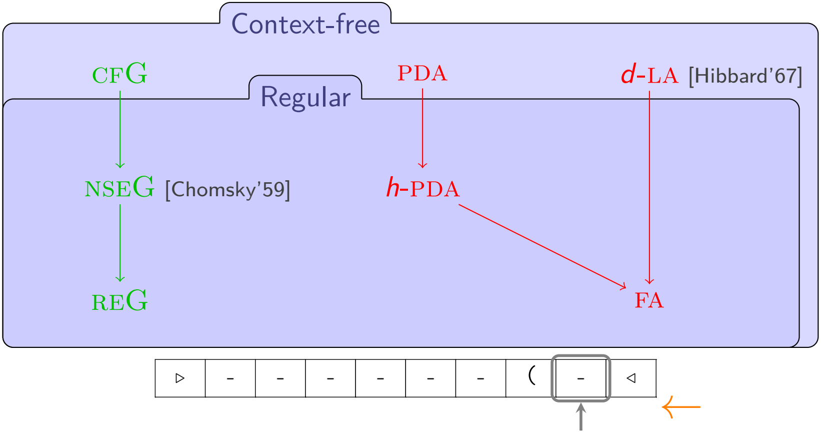
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



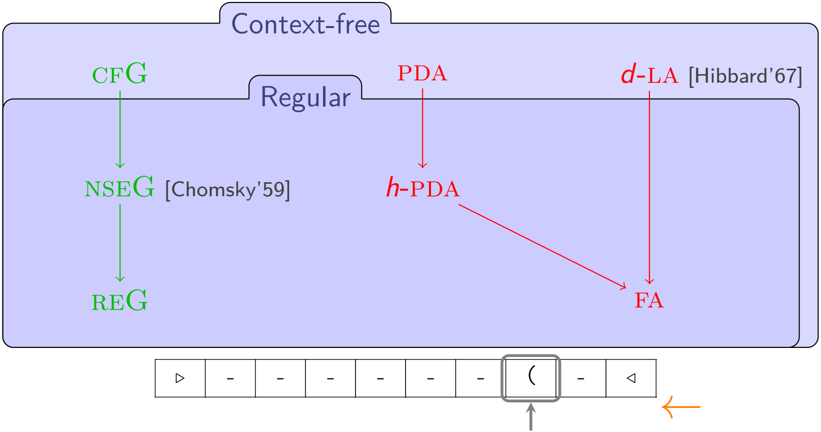
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



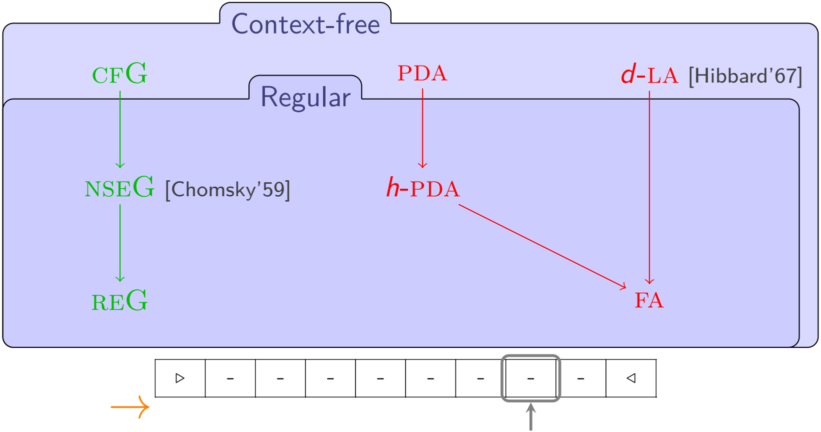
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



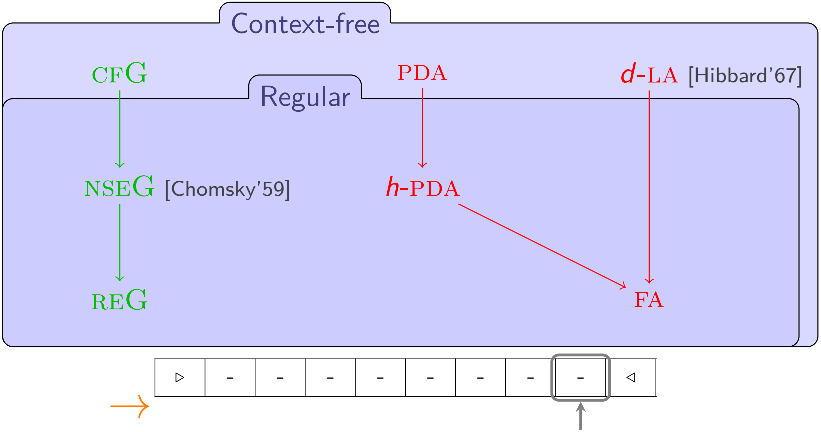
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



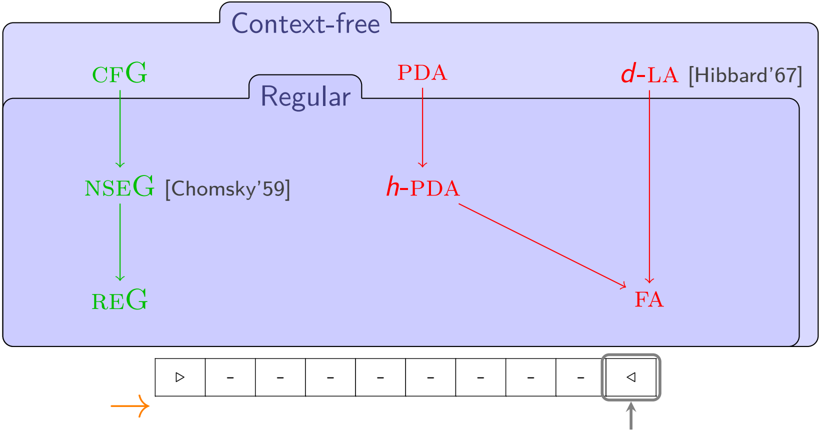
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



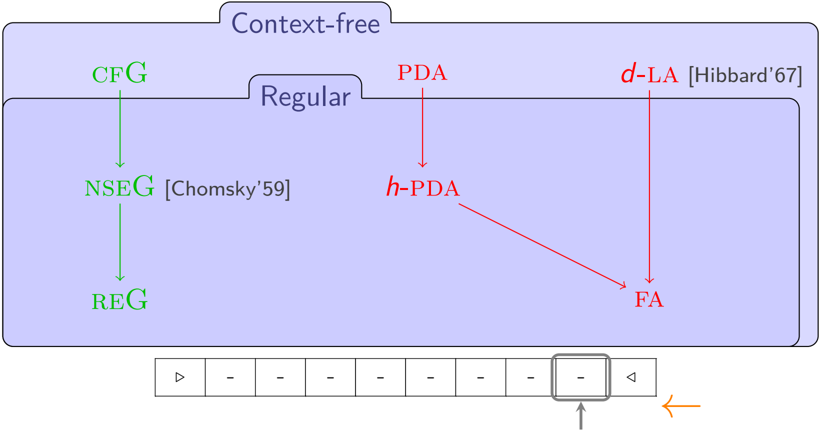
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



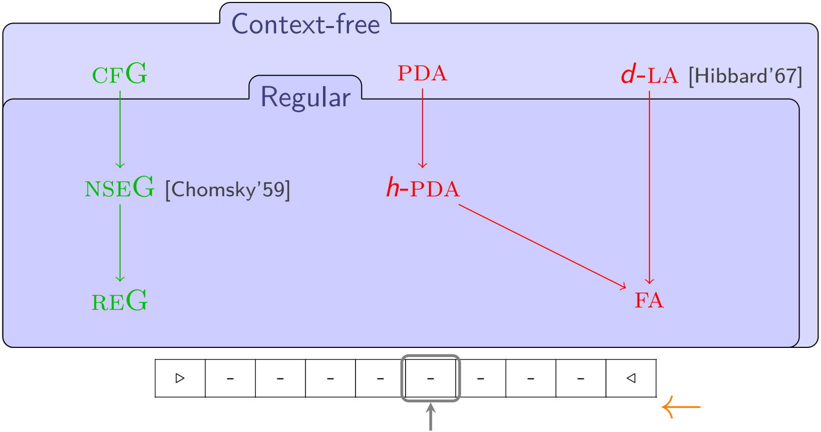
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



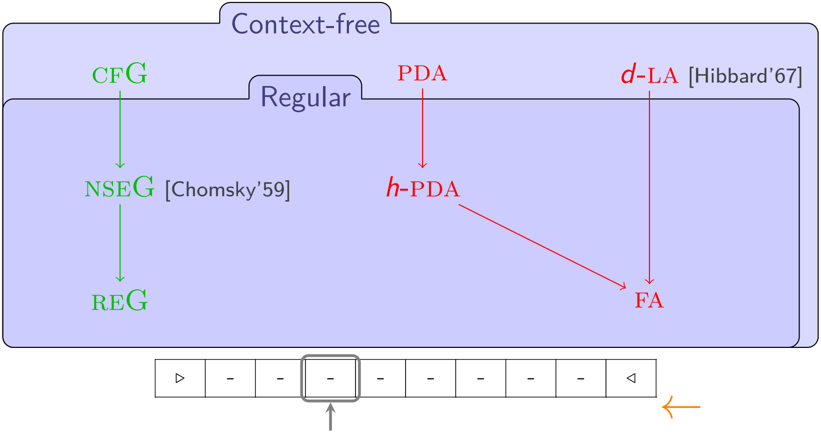
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



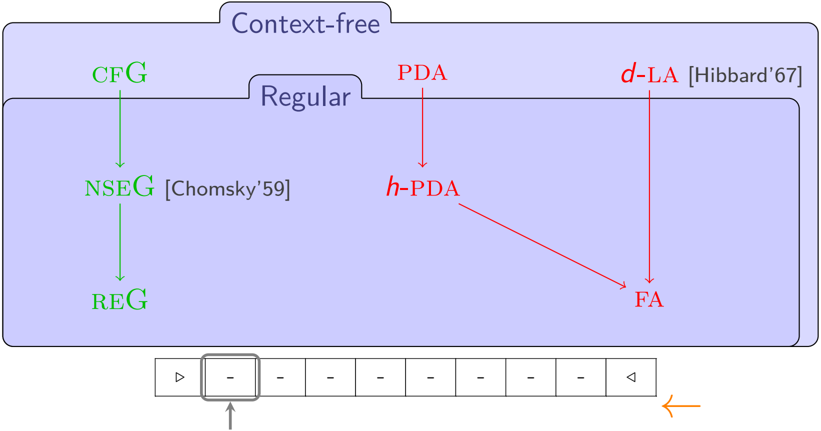
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



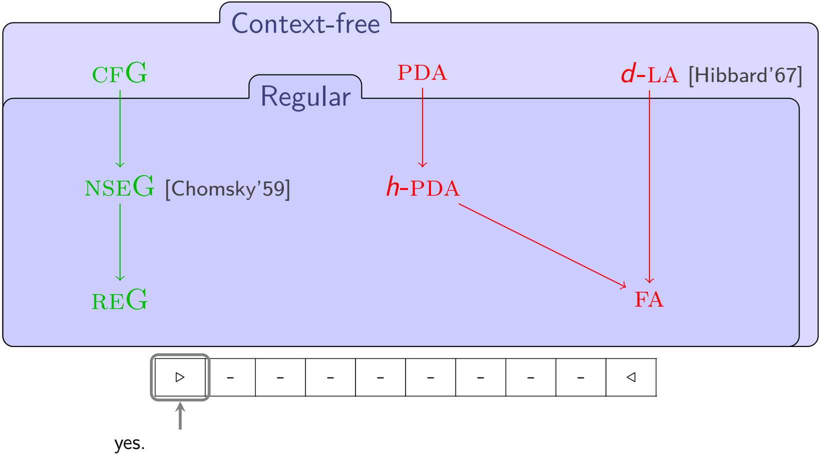
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



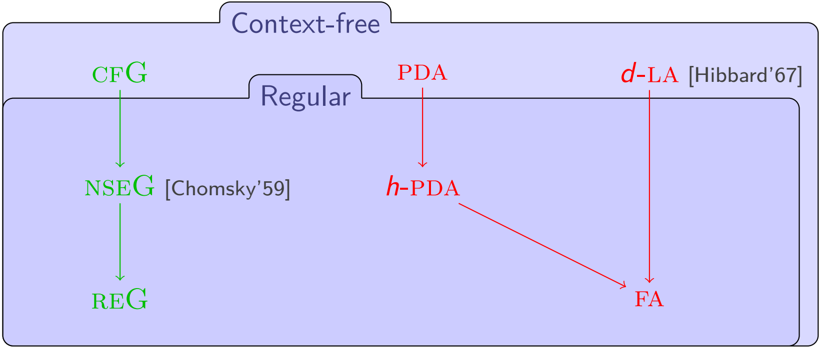
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

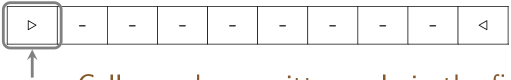
Context-free ability: describe recursive structure



Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure

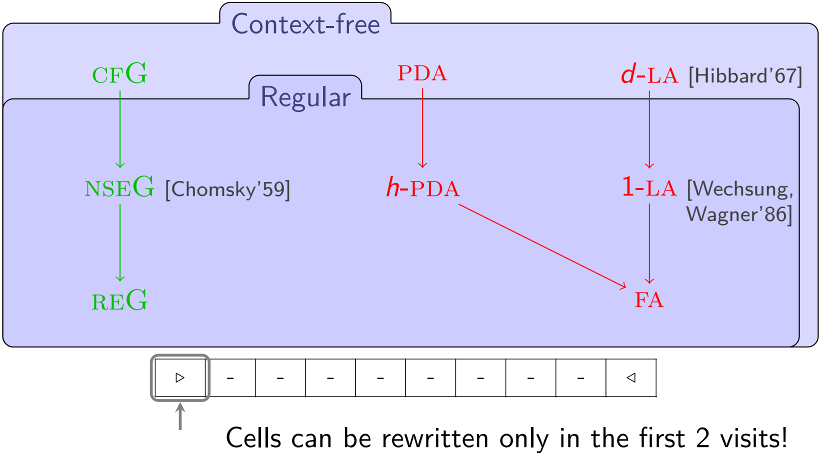




Cells can be rewritten only in the first 2 visits!

Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Context-free ability: describe recursive structure



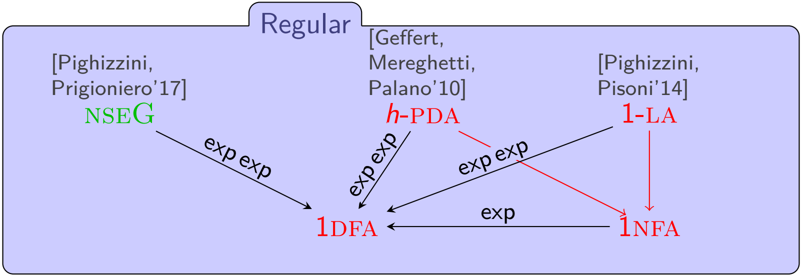
Definition (Hibbard 1967) For $d \in \mathbb{N}$, a d-LA is a 1-TAPE TM allowed to rewrite a cell content only during its first d visits.

Concise representations of regular languages

Regular

NSEG h-PDA 1-LA

Concise representations of regular languages



Definition: Sizes of models:

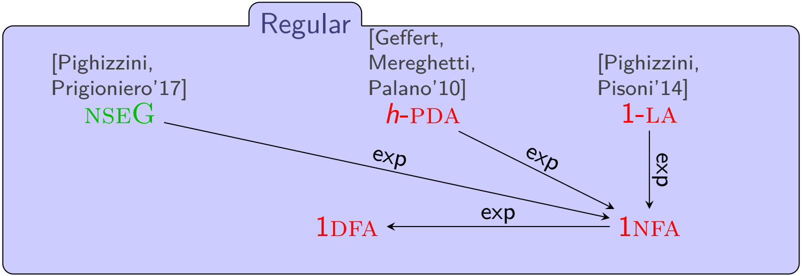
grammars
$$\sum_{X \to \alpha \in P} (2 + |\alpha|)$$

h-PDA
poly in $\#Q$, $\#\Delta$, h

1-LA
poly in $\#Q$, $\#\Gamma$

FA: poly in $\#Q$

Concise representations of regular languages



Definition: Sizes of models:

grammars
$$\sum_{X \to \alpha \in P} (2 + |\alpha|)$$

h-PDA
poly in $\#Q$, $\#\Delta$, h

1-LA
poly in $\#Q$, $\#\Gamma$

FA: poly in $\#Q$

Concise representations of regular languages

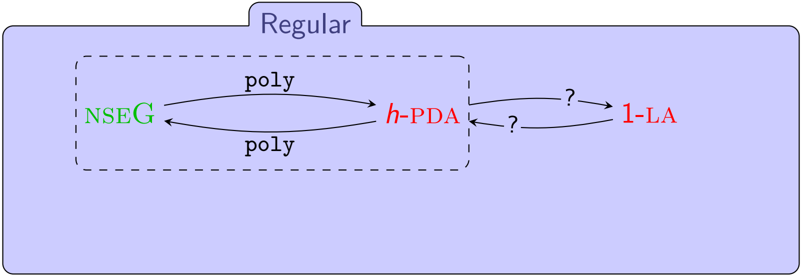
Regular

$$\begin{tikzcd} \textsc{nseG} \arrow[r, bend left, "?"] & h\text{-}\textsc{pda} \arrow[l, bend left, "?"] \arrow[r, bend left, "?"] & 1\text{-}\textsc{la} \arrow[l, bend left, "?"] \end{tikzcd}$$

Definition: Sizes of models:

grammars	h-PDA	1-LA
$\sum_{X \to \alpha \in P} (2 + \lvert\alpha\rvert)$	poly in $\#Q$, $\#\Delta$, h	poly in $\#Q$, $\#\Gamma$

Concise representations of regular languages



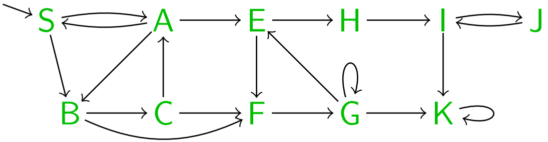
Definition: Sizes of models:

grammars	h-PDA	1-LA
$\sum_{X \to \alpha \in P} (2 + \lvert\alpha\rvert)$	poly in $\#Q$, $\#\Delta$, h	poly in $\#Q$, $\#\Gamma$

From NSEG to h-PDA and back

Production graph

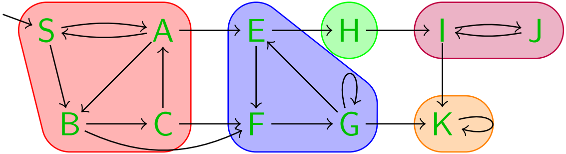
There is an edge $X \to Y$ if there is a production $X \to \alpha Y \beta$



From NSEG to h-PDA and back

Production graph

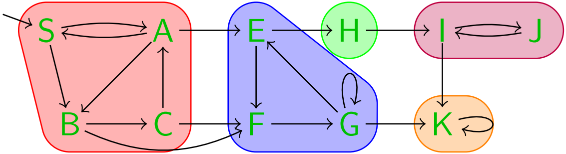
There is an edge $X \rightarrow Y$ if there is a production $X \rightarrow \alpha Y \beta$



From NSEG to h-PDA and back

Production graph

There is an edge $X \to Y$ if there is a production $X \to \alpha Y \beta$



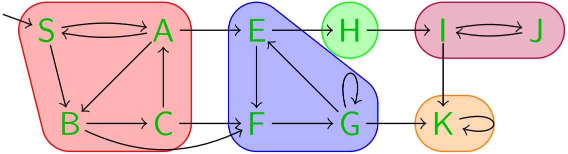
- each SCC defines a left- or right-linear grammar

[Anselmo, Giammarresi, Varricchio 2002]

From NSEG to h-PDA and back

Production graph

There is an edge $X \to Y$ if there is a production $X \to \alpha Y \beta$



- each SCC defines a left- or right-linear grammar

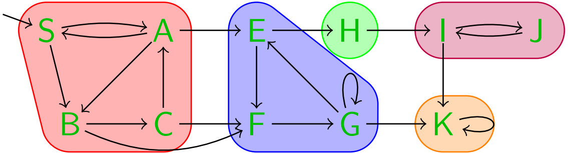
  [Anselmo, Giammarresi, Varricchio 2002]
- with a polynomial size increase,

  we can assume that each such SCC-grammar is right-linear

From NSEG to *h*-PDA and back

Production graph

There is an edge
$X \to Y$ if there is a
production $X \to \alpha Y \beta$

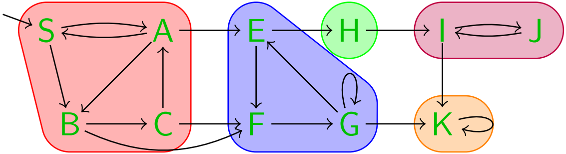


- each SCC defines a left- or right-linear grammar
  [Anselmo, Giammarresi, Varricchio 2002]
- with a polynomial size increase,
  we can assume that each such SCC-grammar is right-linear
- the resulting grammar can in turn be transformed
  into an *h*-PDA of polynomial size (*adapting* [AGV02])

From NSEG to h-PDA and back

Production graph

There is an edge $X \to Y$ if there is a production $X \to \alpha Y \beta$



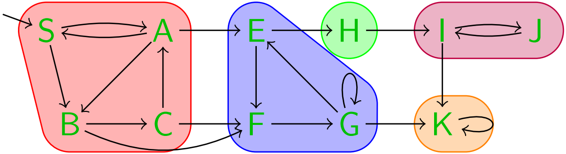
- each SCC defines a left- or right-linear grammar [Anselmo, Giammarresi, Varricchio 2002]
- with a polynomial size increase, we can assume that each such SCC-grammar is right-linear
- the resulting grammar can in turn be transformed into an h-PDA of polynomial size (*adapting* [AGV02])

- Conversely, we can transform each h-PDA into a poly-size NSEG

From NSEG to h-PDA and back

Production graph

There is an edge $X \to Y$ if there is a production $X \to \alpha Y \beta$



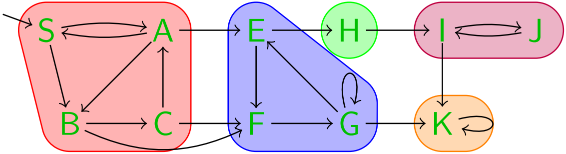
- each SCC defines a left- or right-linear grammar [Anselmo, Giammarresi, Varricchio 2002]
- with a polynomial size increase, we can assume that each such SCC-grammar is right-linear
- the resulting grammar can in turn be transformed into an h-PDA of polynomial size (*adapting* [AGV02])

- Conversely, we can transform each h-PDA into a poly-size NSEG of particular form:

From NSEG to h-PDA and back

Production graph

There is an edge $X \to Y$ if there is a production $X \to \alpha Y \beta$



- each SCC defines a left- or right-linear grammar
  [Anselmo, Giammarresi, Varricchio 2002]
- with a polynomial size increase,
  we can assume that each such SCC-grammar is right-linear
- the resulting grammar can in turn be transformed
  into an h-PDA of polynomial size (*adapting* [AGV02])

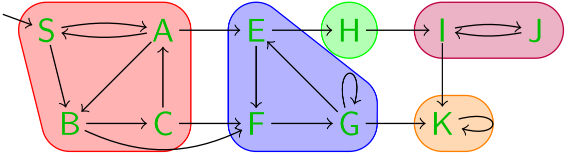
- Conversely, we can transform each h-PDA into a poly-size NSEG of particular form:

  CNF in which each production $X \to YZ$ is such that $Y > X$

From NSEG to h-PDA and back

Production graph

There is an edge $X \to Y$ if there is a production $X \to \alpha Y \beta$

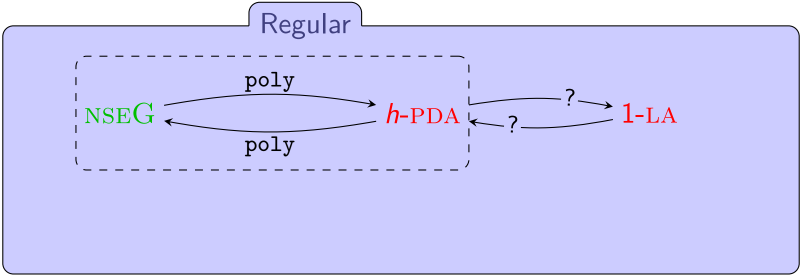


- each SCC defines a left- or right-linear grammar
  [Anselmo, Giammarresi, Varricchio 2002]
- with a polynomial size increase,
  we can assume that each such SCC-grammar is right-linear
- the resulting grammar can in turn be transformed
  into an h-PDA of polynomial size (*adapting* [AGV02])

- Conversely, we can transform each h-PDA into a poly-size NSEG of particular form:

  CNF in which each production $X \to YZ$ is such that $Y > X$

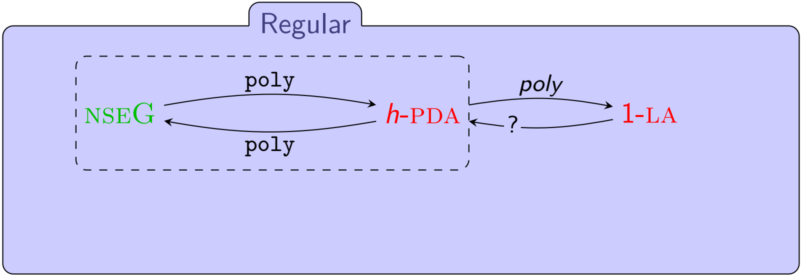
Concise representations of regular languages



Definition: Sizes of models:

grammars	h-PDA	1-LA
$\sum_{X \to \alpha \in P} (2 + \lvert\alpha\rvert)$	poly in $\#Q$, $\#\Delta$, h	poly in $\#Q$, $\#\Gamma$

Concise representations of regular languages

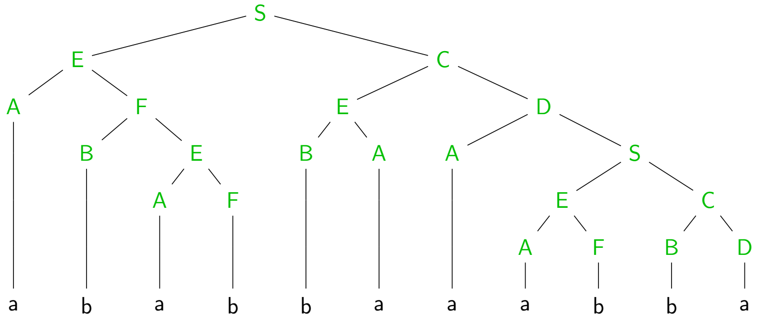


Definition: Sizes of models:

grammars	h-PDA	1-LA
$\sum_{X \to \alpha \in P} (2 + \lvert\alpha\rvert)$	poly in $\#Q$, $\#\Delta$, h	poly in $\#Q$, $\#\Gamma$

From NSEG to 1-LA

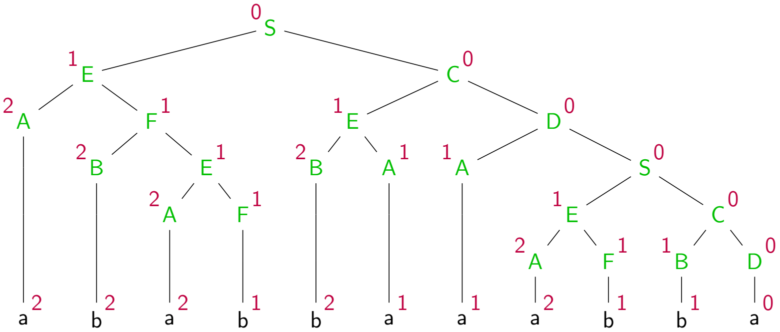
from NSEG in CNF with
$X \to YZ \implies Y > X$



▷	a	b	a	b	b	a	a	a	b	b	a	◁

From NSEG to 1-LA

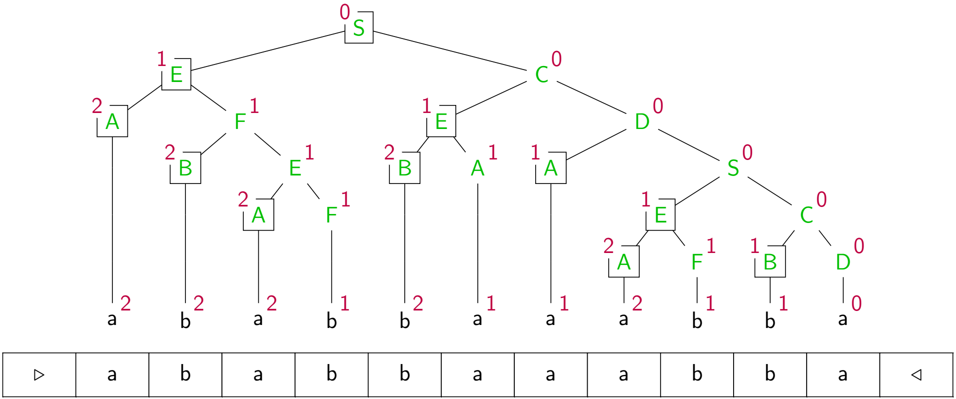
from NSEG in CNF with
$X \to YZ \implies Y > X$



▷	a	b	a	b	b	a	a	a	b	b	a	◁

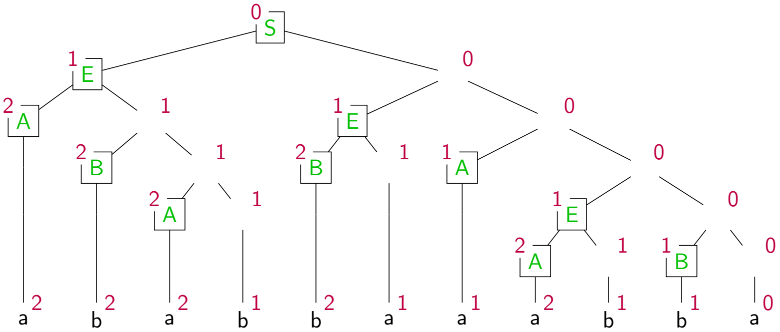
From NSEG to 1-LA

from NSEG in CNF with $X \to YZ \implies Y > X$



From NSEG to 1-LA

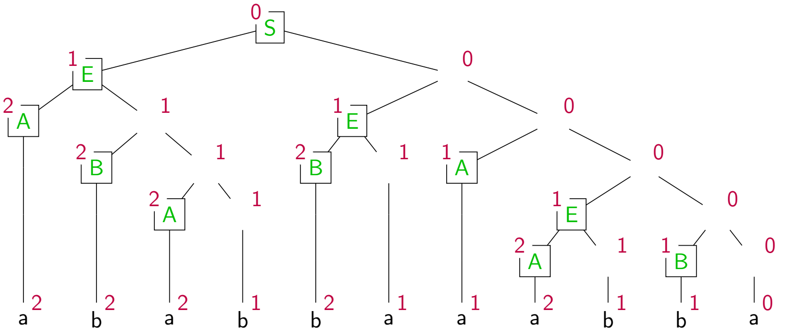
from NSEG in CNF with
$X \to YZ \implies Y > X$



$\triangleright$	$a, A, 2$	$b, B, 2$	$a, A, 2$	$b, E, 1$	$b, B, 2$	$a, E, 1$	$a, A, 1$	$a, A, 2$	$a, E, 1$	$b, B, 1$	$a, S, 0$	$\triangleleft$

From NSEG to 1-LA

from NSEG in CNF with
$X \to YZ \implies Y > X$

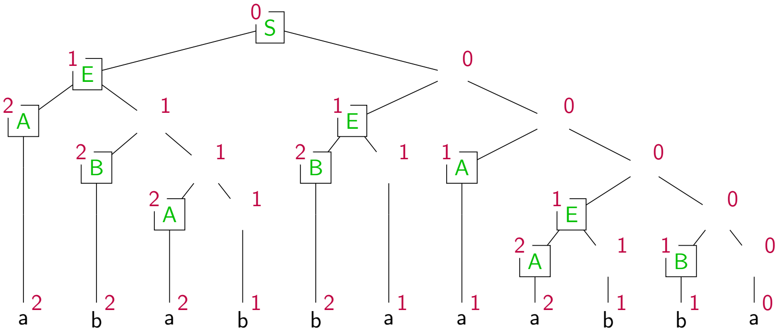


$\triangleright$	$a, A, 2$	$b, B, 2$	$a, A, 2$	$b, E, 1$	$b, B, 2$	$a, E, 1$	$a, A, 1$	$a, A, 2$	$a, E, 1$	$b, B, 1$	$\boxed{a, S, 0}$	$\triangleleft$

the root

From NSEG to 1-LA

from NSEG in CNF with
$X \to YZ \implies Y > X$

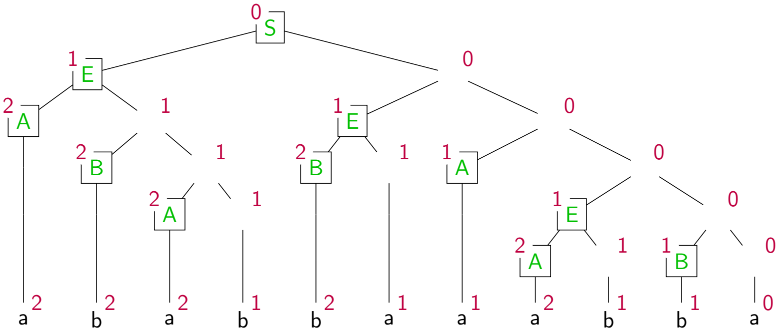


$\triangleright$	$a, A, 2$	$b, B, 2$	$a, A, 2$	$b, E, 1$	$b, B, 2$	$a, E, 1$	$a, A, 1$	$a, A, 2$	$a, E, 1$	$b, B, 1$	$\boxed{a, S, 0}$	$\triangleleft$

the root

From NSEG to 1-LA

from NSEG in CNF with
$X \to YZ \implies Y > X$



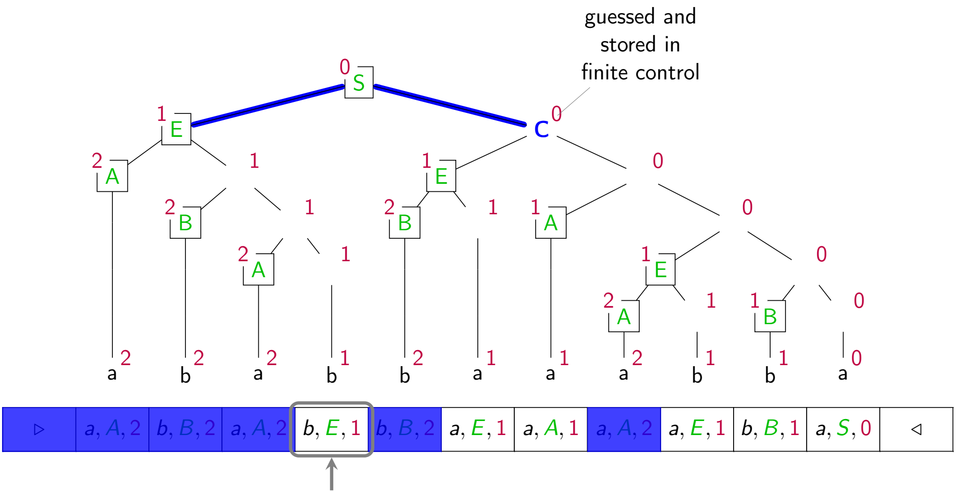
$\triangleright$	$a, A, 2$	$b, B, 2$	$a, A, 2$	$b, E, 1$	$b, B, 2$	$a, E, 1$	$a, A, 1$	$a, A, 2$	$a, E, 1$	$b, B, 1$	$a, S, 0$	$\triangleleft$

its left child

the root

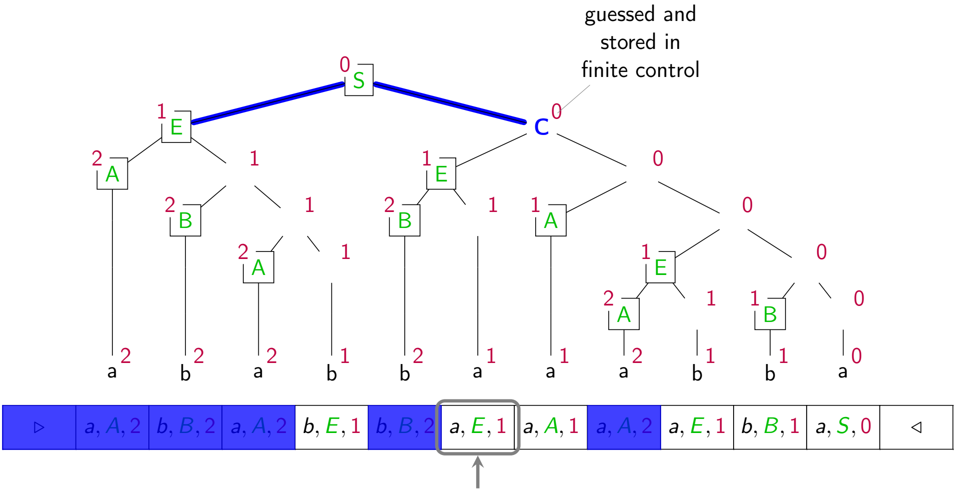
From NSEG to 1-LA

from NSEG in CNF with
$X \to YZ \implies Y > X$



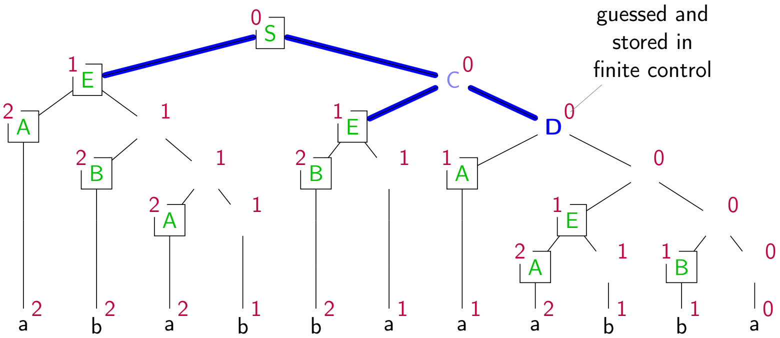
From NSEG to 1-LA

from NSEG in CNF with
$X \to YZ \implies Y > X$



From NSEG to 1-LA

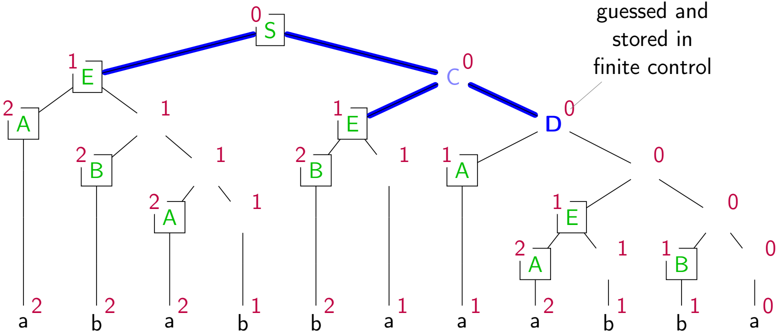
from NSEG in CNF with
$X \to YZ \implies Y > X$



$\triangleright$	$a, A, 2$	$b, B, 2$	$a, A, 2$	$b, E, 1$	$b, B, 2$	$a, E, 1$	$a, A, 1$	$a, A, 2$	$a, E, 1$	$b, B, 1$	$a, S, 0$	$\triangleleft$

From NSEG to 1-LA

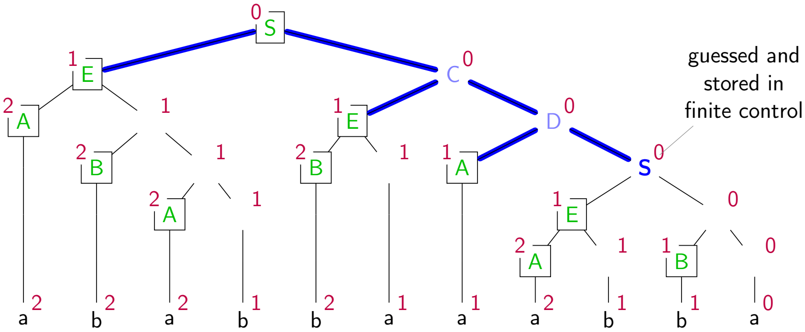
from NSEG in CNF with
$X \to YZ \implies Y > X$



$\triangleright$	$a, A, 2$	$b, B, 2$	$a, A, 2$	$b, E, 1$	$b, B, 2$	$a, E, 1$	$a, A, 1$	$a, A, 2$	$a, E, 1$	$b, B, 1$	$a, S, 0$	$\triangleleft$

From NSEG to 1-LA

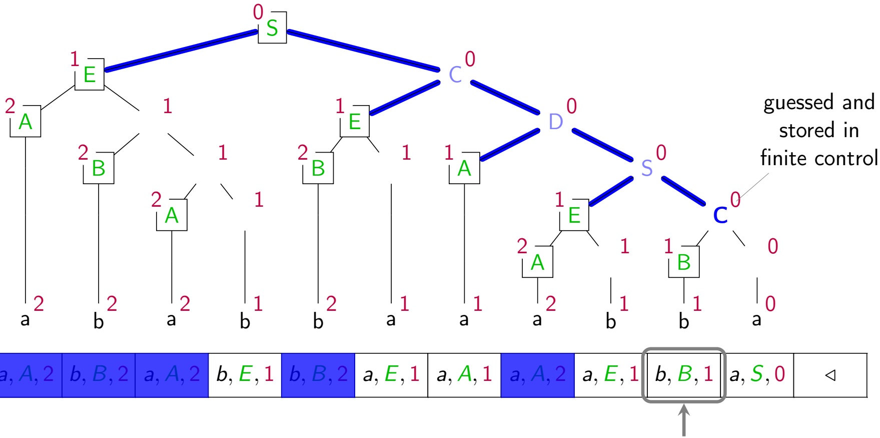
from NSEG in CNF with
$X \to YZ \implies Y > X$



$\triangleright$	$a, A, 2$	$b, B, 2$	$a, A, 2$	$b, E, 1$	$b, B, 2$	$a, E, 1$	$a, A, 1$	$a, A, 2$	$a, E, 1$	$b, B, 1$	$a, S, 0$	$\triangleleft$

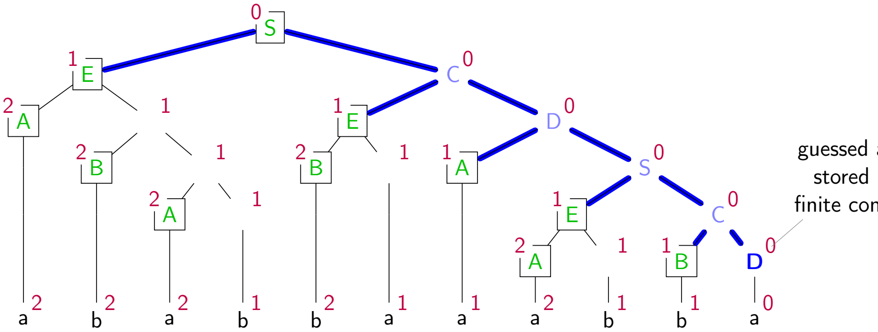
From NSEG to 1-LA

from NSEG in CNF with
$X \to YZ \implies Y > X$



From NSEG to 1-LA

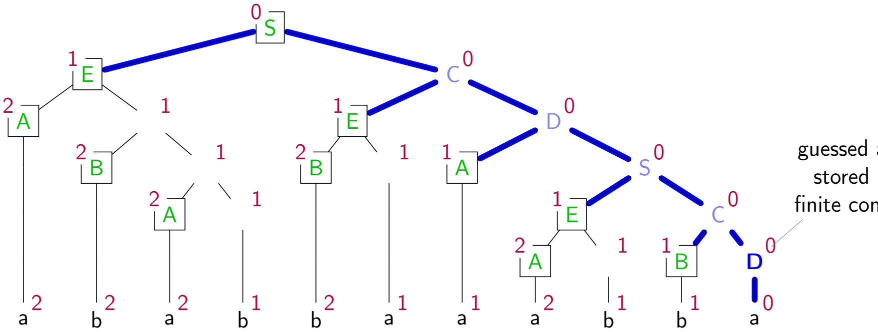
from NSEG in CNF with
$X \to YZ \implies Y > X$



$\triangleright$	$a, A, 2$	$b, B, 2$	$a, A, 2$	$b, E, 1$	$b, B, 2$	$a, E, 1$	$a, A, 1$	$a, A, 2$	$a, E, 1$	$b, B, 1$	$a, S, 0$	$\triangleleft$

From NSEG to 1-LA

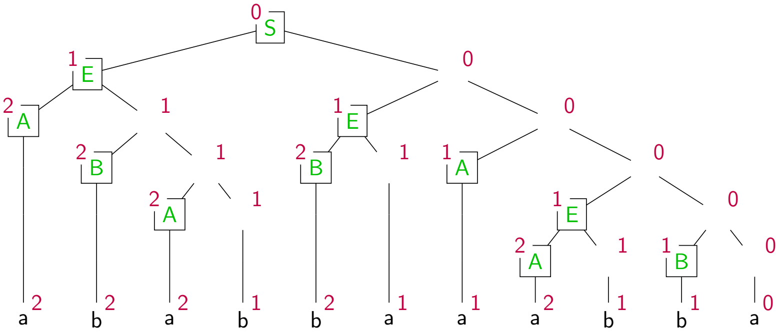
from NSEG in CNF with
$X \to YZ \implies Y > X$



$\triangleright$	$a, A, 2$	$b, B, 2$	$a, A, 2$	$b, E, 1$	$b, B, 2$	$a, E, 1$	$a, A, 1$	$a, A, 2$	$a, E, 1$	$b, B, 1$	$a, S, 0$	$\triangleleft$

From NSEG to 1-LA

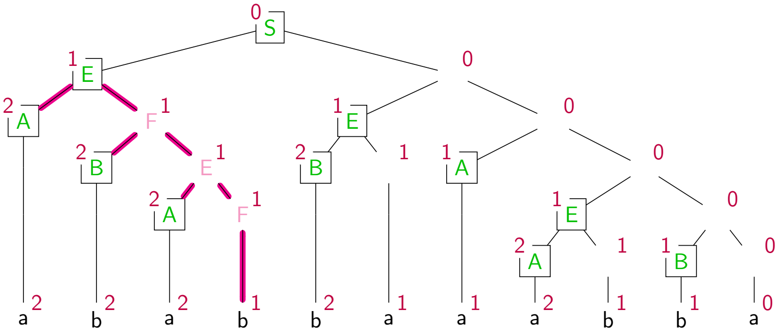
from NSEG in CNF with
$X \to YZ \implies Y > X$



$\triangleright$	$a, A, 2$	$b, B, 2$	$a, A, 2$	$b, E, 1$	$b, B, 2$	$a, E, 1$	$a, A, 1$	$a, A, 2$	$a, E, 1$	$b, B, 1$	$a, S, 0$	$\triangleleft$

From NSEG to 1-LA

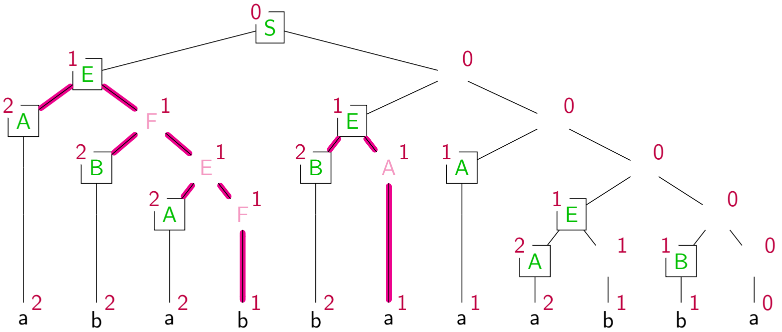
from NSEG in CNF with
$X \to YZ \implies Y > X$



$\triangleright$	$a, A, 2$	$b, B, 2$	$a, A, 2$	$b, E, 1$	$b, B, 2$	$a, E, 1$	$a, A, 1$	$a, A, 2$	$a, E, 1$	$b, B, 1$	$a, S, 0$	$\triangleleft$

From NSEG to 1-LA

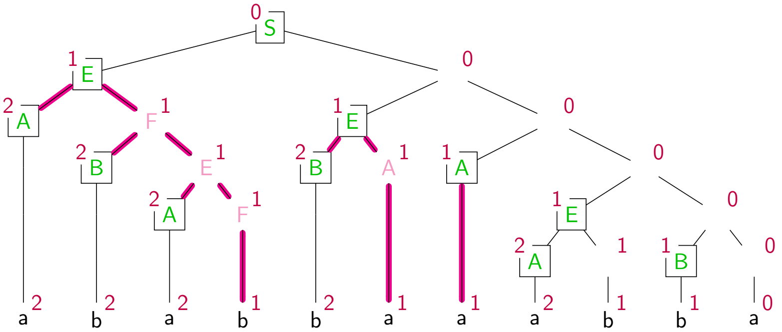
from NSEG in CNF with
$X \to YZ \implies Y > X$



▷	$a, A, 2$	$b, B, 2$	$a, A, 2$	$b, E, 1$	$b, B, 2$	$a, E, 1$	$a, A, 1$	$a, A, 2$	$a, E, 1$	$b, B, 1$	$a, S, 0$	◁

From NSEG to 1-LA

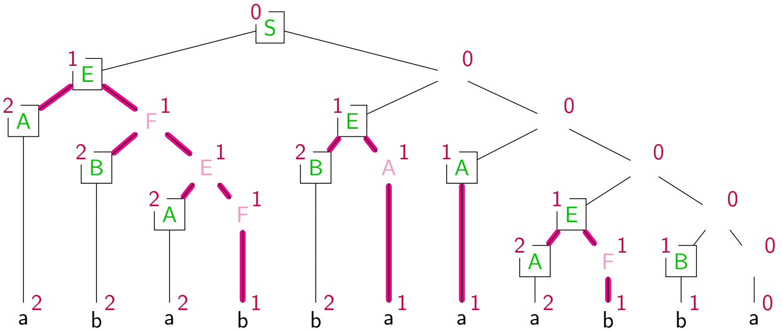
from NSEG in CNF with
$X \to YZ \implies Y > X$



$\triangleright$	$a, A, 2$	$b, B, 2$	$a, A, 2$	$b, E, 1$	$b, B, 2$	$a, E, 1$	$\boxed{a, A, 1}$	$a, A, 2$	$a, E, 1$	$b, B, 1$	$a, S, 0$	$\triangleleft$

From NSEG to 1-LA

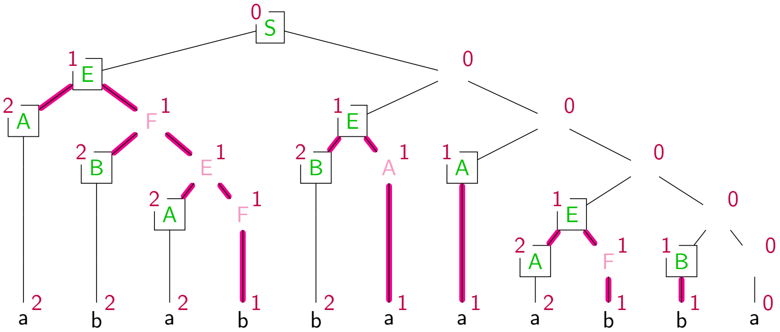
from NSEG in CNF with
$X \to YZ \implies Y > X$



$\triangleright$	$a, A, 2$	$b, B, 2$	$a, A, 2$	$b, E, 1$	$b, B, 2$	$a, E, 1$	$a, A, 1$	$a, A, 2$	$a, E, 1$	$b, B, 1$	$a, S, 0$	$\triangleleft$

From NSEG to 1-LA

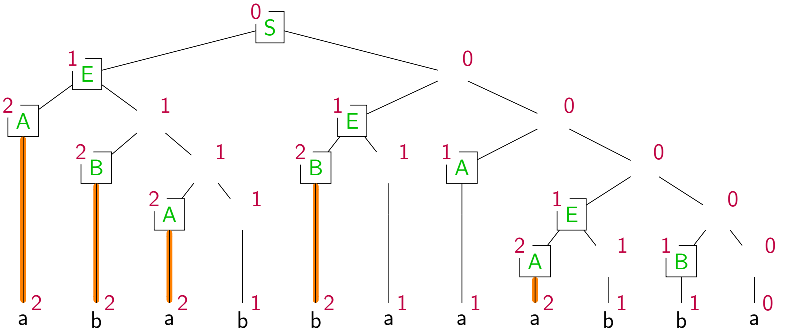
from NSEG in CNF with
$X \to YZ \implies Y > X$



$\triangleright$	$a, A, 2$	$b, B, 2$	$a, A, 2$	$b, E, 1$	$b, B, 2$	$a, E, 1$	$a, A, 1$	$a, A, 2$	$a, E, 1$	$\boxed{b, B, 1}$	$a, S, 0$	$\triangleleft$

From NSEG to 1-LA

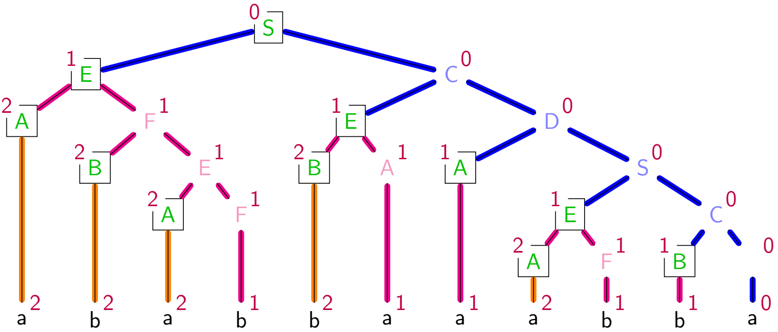
from NSEG in CNF with
$X \to YZ \implies Y > X$



$\triangleright$	$\boxed{a, A, 2}$	$\boxed{b, B, 2}$	$\boxed{a, A, 2}$	$b, E, 1$	$\boxed{b, B, 2}$	$a, E, 1$	$a, A, 1$	$\boxed{a, A, 2}$	$a, E, 1$	$b, B, 1$	$a, S, 0$	$\triangleleft$

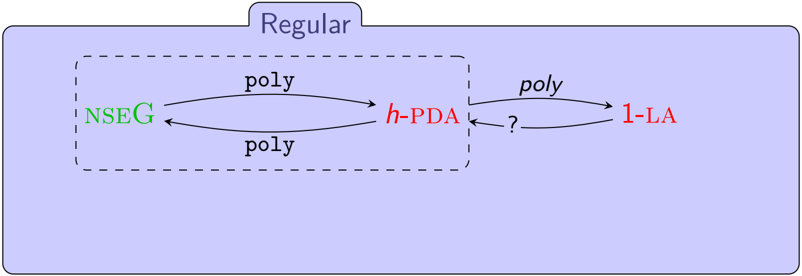
From NSEG to 1-LA

from NSEG in CNF with
$X \to YZ \implies Y > X$

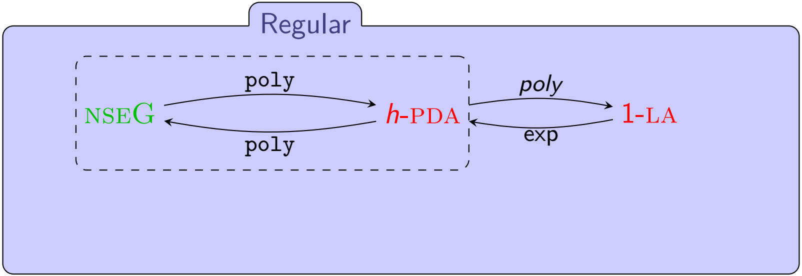


▷	$a, A, 2$	$b, B, 2$	$a, A, 2$	$b, E, 1$	$b, B, 2$	$a, E, 1$	$a, A, 1$	$a, A, 2$	$a, E, 1$	$b, B, 1$	$a, S, 0$	◁

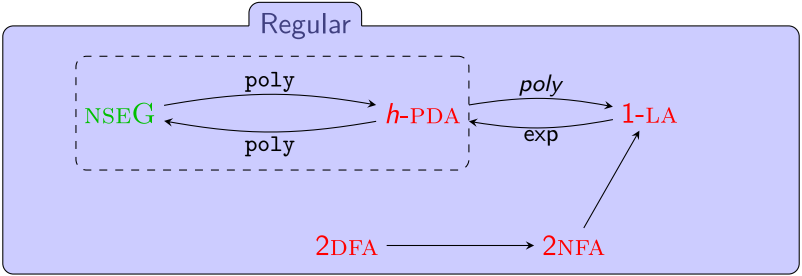
Simulation of 1-LA: an exponential gap



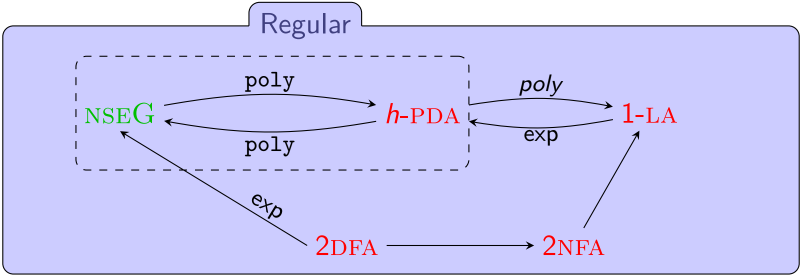
Simulation of 1-LA: an exponential gap



Simulation of 1-LA: an exponential gap



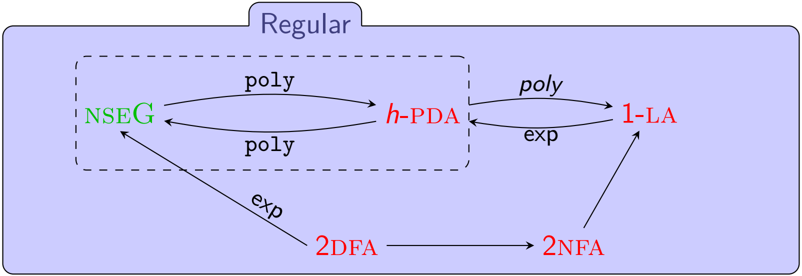
Simulation of 1-LA: an exponential gap



Theorem: $$L_k = \{u^n \mid n \in \mathbb{N},\ u \in \{a,b\}^k\}$$

- accepted by a 2DFA with $\mathcal{O}(n)$ states
- for which a PDA or a CFG requires a size exponential in n

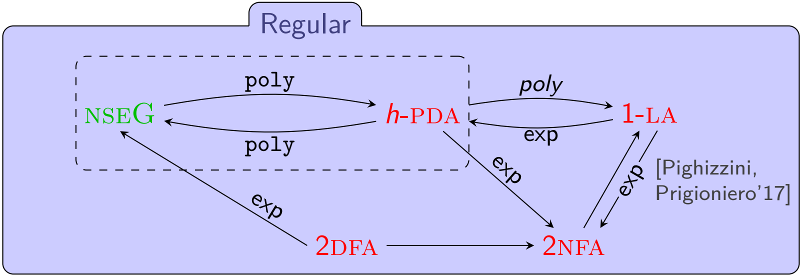
Simulation of 1-LA: an exponential gap



Theorem: $$L_k = \{u^n \mid n \in \mathbb{N},\ u \in \{a,b\}^k\}$$

- accepted by a 2DFA with $\mathcal{O}(n)$ states
- for which a PDA or a CFG requires a size exponential in n

Simulation of 1-LA: an exponential gap



Theorem: $L_k = \{u^n \mid n \in \mathbb{N},\ u \in \{a, b\}^k\}$

- accepted by a 2DFA with $\mathcal{O}(n)$ states
- for which a PDA or a CFG requires a size exponential in n

Time complexity of 1-LA

- Remark: As d-LA, 1-LA may use quadratic time.

Time complexity of 1-LA

- Remark: As d-LA, 1-LA may use quadratic time.

This contrasts with Hennie's result:

Theorem: lin-time 1-tape TM recognize regular languages only.

Time complexity of 1-LA

- Remark: As d-LA, 1-LA may use quadratic time.

This contrasts with Hennie's result:

Theorem: lin-time 1-tape TM recognize regular languages only.

Is there some 1-LA more succinct
than any equivalent lin-time 1-tape TM?

Time complexity of 1-LA

- Remark: As d-LA, 1-LA may use quadratic time.

This contrasts with Hennie's result:

Theorem: lin-time 1-tape TM recognize regular languages only.

Is there some 1-LA more succinct
than any equivalent lin-time 1-tape TM?

Theorem: No.

Time complexity of 1-LA

- Remark: As d-LA, 1-LA may use quadratic time.

This contrasts with Hennie's result:

Theorem: lin-time 1-tape TM recognize regular languages only.

Is there some 1-LA more succinct
than any equivalent lin-time 1-tape TM?

Theorem: No. Each 1-LA admits an equivalent linear-time 1-LA of polynomial size.

Time complexity of 1-LA

- Remark: As d-LA, 1-LA may use quadratic time.

This contrasts with Hennie's result:

Theorem: lin-time 1-tape TM recognize regular languages only.

Is there some 1-LA more succinct
than any equivalent lin-time 1-tape TM?

Theorem: No. Each 1-LA admits an equivalent linear-time 1-LA of polynomial size.
Rewriting can be pushed to an initial phase in the resulting 1-LA.

1-LA versus common guess

Definition: 2NFA (resp. 2DFA) with common guess:

- first annotate the input with some guessed symbols from a finite alphabet
- then perform a read-only computation over the enriched input

1-LA versus common guess

Definition: 2NFA (resp. 2DFA) with common guess:

- first annotate the input with some guessed symbols from a finite alphabet
- then perform a read-only computation over the enriched input

Theorem:
Each 1-LA admits an equivalent 2NFA+CG of poly size.

1-LA versus common guess

Definition: 2NFA (resp. 2DFA) with common guess:

- first annotate the input with some guessed symbols from a finite alphabet
- then perform a read-only computation over the enriched input

Theorem:

Each 1-LA admits an equivalent 2NFA+CG of poly size.

Each deterministic 1-LA admits an equivalent 2DFA+CG of poly size.

1-LA versus common guess

Definition: 2NFA (resp. 2DFA) with common guess:

- first annotate the input with some guessed symbols from a finite alphabet
- then perform a read-only computation over the enriched input

Theorem:

Each 1-LA admits an equivalent 2NFA+CG of poly size.

Each deterministic 1-LA admits an equivalent 2DFA+CG of poly size.

Remark:

2NFAs+CG are particular 1-LAs

1-LA versus common guess

Definition: 2NFA (resp. 2DFA) with common guess:

- first annotate the input with some guessed symbols from a finite alphabet
- then perform a read-only computation over the enriched input

Theorem:

Each 1-LA admits an equivalent 2NFA+CG of poly size.

Each deterministic 1-LA admits an equivalent 2DFA+CG of poly size.

Remark:

2NFAs+CG are particular 1-LAs, contrary to 2DFAs+CG wrt det 1-LAs.

1-LA versus common guess

Definition: 2NFA (resp. 2DFA) with common guess:

- first annotate the input with some guessed symbols from a finite alphabet
- then perform a read-only computation over the enriched input

Theorem:

Each 1-LA admits an equivalent 2NFA+CG of poly size.

Each deterministic 1-LA admits an equivalent 2DFA+CG of poly size.

Remark:

2NFAs+CG are particular 1-LAs, contrary to 2DFAs+CG wrt det 1-LAs.

Theorem:

Exponential lower bound for the simulation of 2DFA+CG by det 1-LA.

1-LA versus common guess

Definition: 2NFA (resp. 2DFA) with common guess:

- first annotate the input with some guessed symbols from a finite alphabet
- then perform a read-only computation over the enriched input

Theorem:

Each 1-LA admits an equivalent 2NFA+CG of poly size.

Each deterministic 1-LA admits an equivalent 2DFA+CG of poly size.

Remark:

2NFAs+CG are particular 1-LAs, contrary to 2DFAs+CG wrt det 1-LAs.

Theorem:

Exponential lower bound for the simulation of 2DFA+CG by det 1-LA.

Proof: cost of reversal

Conclusion

Open problem [Sakoda and Sipser'78]

What is the size cost of the simulation of 2NFAs by 2DFAs?

Conclusion

Open problem [Sakoda and Sipser'78]

What is the size cost of the simulation of 2NFAs by 2DFAs?

- hard (connexions with LogSpace versus NLogSpace)

Conclusion

Open problem [Sakoda and Sipser'78]

What is the size cost of the simulation of 2NFAs by 2DFAs?

- hard (connexions with LogSpace versus NLogSpace)

Sub-goal

Is there a poly-size simulation of 2NFA by 2DFA+CG?

Conclusion

Open problem [Sakoda and Sipser'78]

What is the size cost of the simulation of 2NFAs by 2DFAs?

- hard (connexions with LogSpace versus NLogSpace)

Sub-goal

Is there a poly-size simulation of 2NFA by 2DFA+CG?

- poly-size simulation of 2NFA by lin-time Turing Machines [G., Pighizzini, Prigioniero, Průša'18]

Conclusion

Open problem [Sakoda and Sipser'78]

What is the size cost of the simulation of 2NFAs by 2DFAs?

- hard (connexions with LogSpace versus NLogSpace)

Sub-goal

Is there a poly-size simulation of 2NFA by 2DFA+CG?

- poly-size simulation of 2NFA by lin-time Turing Machines [G., Pighizzini, Prigioniero, Průša'18]
- sub-exponential simulation of 2NFA by 2DFA+CG (in progress)

Conclusion

Open problem [Sakoda and Sipser'78]

What is the size cost of the simulation of 2NFAs by 2DFAs?

- hard (connexions with LogSpace versus NLogSpace)

Sub-goal

Is there a poly-size simulation of 2NFA by 2DFA+CG?

- poly-size simulation of 2NFA by lin-time Turing Machines [G., Pighizzini, Prigioniero, Průša'18]
- sub-exponential simulation of 2NFA by 2DFA+CG (in progress)

Thank you for your attention.