Limited Automata and Context-Free Languages

Giovanni Pighizzini   Andrea Pisoni

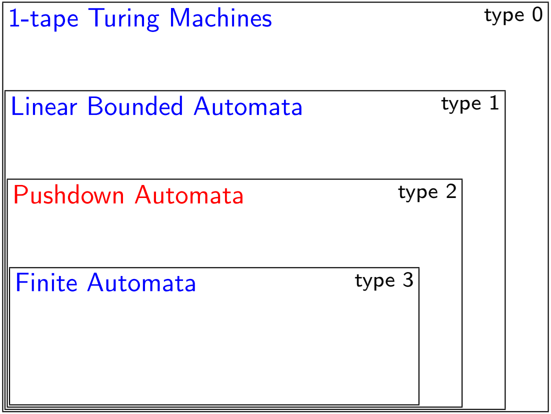
Dipartimento di Informatica
Università degli Studi di Milano, Italy

NCMA 2013

Umeå, Sweden
August 13–14, 2013

UNIVERSITÀ DEGLI STUDI
DI MILANO

The Chomsky Hierarchy



Limited Automata [Hibbard'67]

One-tape Turing machines with restricted rewritings

Definition

Fixed an integer $d \geq 1$, a *d-limited automaton* is

- a one-tape Turing machine
- which is allowed to rewrite the content of each tape cell *only in the first d visits*

Limited Automata [Hibbard'67]

One-tape Turing machines with restricted rewritings

Definition

Fixed an integer $d \geq 1$, a *d-limited automaton* is

- a one-tape Turing machine
- which is allowed to rewrite the content of each tape cell *only in the first d visits*

Limited Automata [Hibbard'67]

One-tape Turing machines with restricted rewritings

Definition

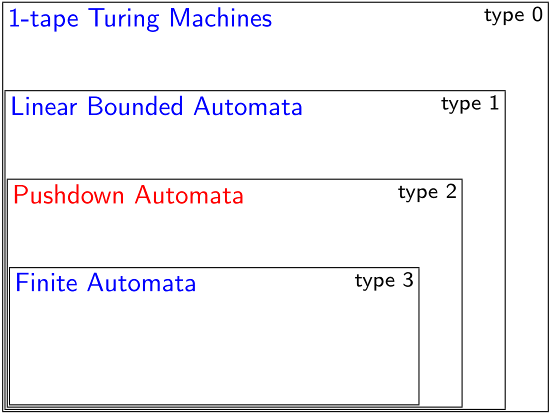
Fixed an integer $d \geq 1$, a *d-limited automaton* is

- a one-tape Turing machine
- which is allowed to rewrite the content of each tape cell *only in the first d visits*

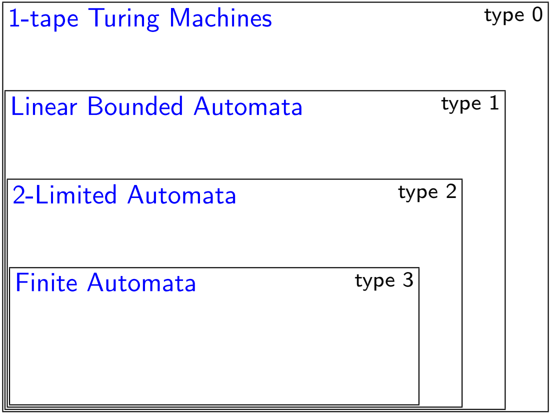
Computational power

- For each $d \geq 2$, d-limited automata characterize context-free languages [Hibbard'67]
- 1-limited automata characterize regular languages [Wagner&Wechsung'86]

The Chomsky Hierarchy



The Chomsky Hierarchy



Our Contributions

- 2-Limited Automata ≡ Pushdown Automata:
  descriptional complexity point of view

- Determinism vs Nondeterminism

Our Contributions

- 2-Limited Automata ≡ Pushdown Automata: descriptional complexity point of view

2-LAs → PDAs

Exponential gap

- Determinism vs Nondeterminism

Our Contributions

- 2-Limited Automata $\equiv$ Pushdown Automata: descriptional complexity point of view

2-LAs $\rightarrow$ PDAs

Exponential gap

PDAs $\rightarrow$ 2-LAs

Polynomial upper bound

- Determinism vs Nondeterminism

Our Contributions

- 2-Limited Automata $\equiv$ Pushdown Automata: descriptional complexity point of view

2-LAs $\rightarrow$ PDAs

Exponential gap

PDAs $\rightarrow$ 2-LAs

Polynomial upper bound

- Determinism vs Nondeterminism

Our Contributions

- 2-Limited Automata ≡ Pushdown Automata: descriptional complexity point of view

2-LAs → PDAs

Exponential gap

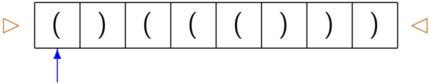
PDAs → 2-LAs

Polynomial upper bound

- Determinism vs Nondeterminism

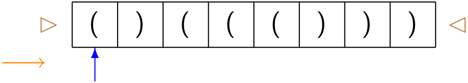
Deterministic Context-Free Languages ≡ Deterministic 2-LAs

Example: Balanced Parentheses



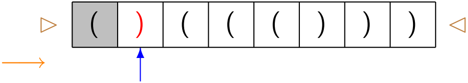
(i) Move to the right to search a closed parenthesis
(ii) Rewrite it by #
(iii) Move to the left to search an open parenthesis
(iv) Rewrite it by #
(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis
(ii) Rewrite it by #
(iii) Move to the left to search an open parenthesis
(iv) Rewrite it by #
(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

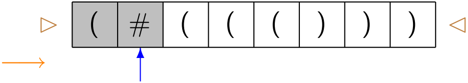
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

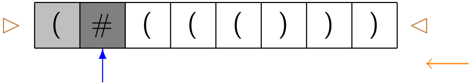
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

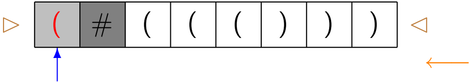
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

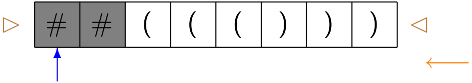
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

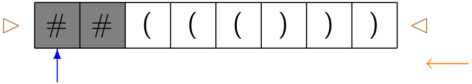
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

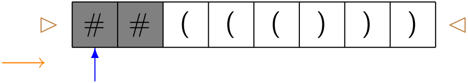
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

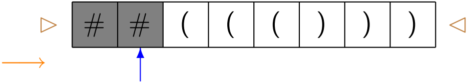
(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis
(ii) Rewrite it by #
(iii) Move to the left to search an open parenthesis
(iv) Rewrite it by #
(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

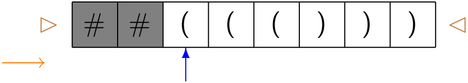
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

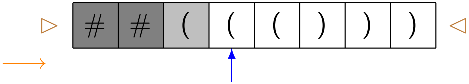
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

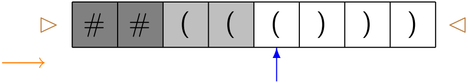
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

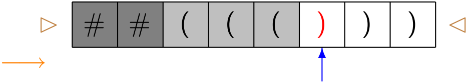
(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis
(ii) Rewrite it by #
(iii) Move to the left to search an open parenthesis
(iv) Rewrite it by #
(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

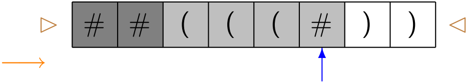
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

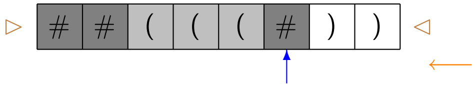
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

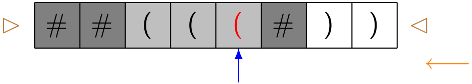
(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis
(ii) Rewrite it by #
(iii) Move to the left to search an open parenthesis
(iv) Rewrite it by #
(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

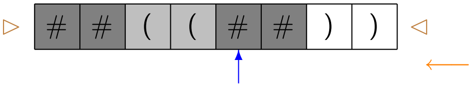
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

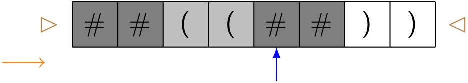
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses

(i) Move to the right to search a closed parenthesis

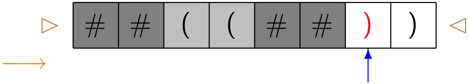
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

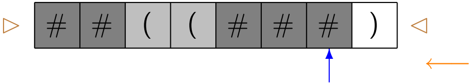
(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses

(i) Move to the right to search a closed parenthesis
(ii) Rewrite it by #
(iii) Move to the left to search an open parenthesis
(iv) Rewrite it by #
(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

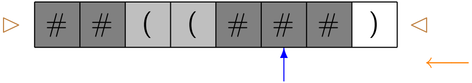
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

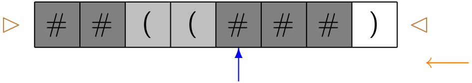
(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis
(ii) Rewrite it by #
(iii) Move to the left to search an open parenthesis
(iv) Rewrite it by #
(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

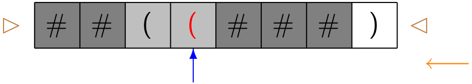
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

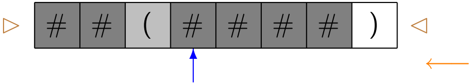
(v) Repeat from the beginning

Example: Balanced Parentheses



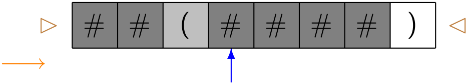
(i) Move to the right to search a closed parenthesis
(ii) Rewrite it by #
(iii) Move to the left to search an open parenthesis
(iv) Rewrite it by #
(v) Repeat from the beginning

Example: Balanced Parentheses



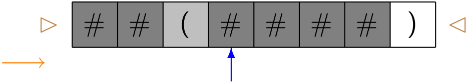
1. (i) Move to the right to search a closed parenthesis
2. (ii) Rewrite it by #
3. (iii) Move to the left to search an open parenthesis
4. (iv) Rewrite it by #
5. (v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis
(ii) Rewrite it by #
(iii) Move to the left to search an open parenthesis
(iv) Rewrite it by #
(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

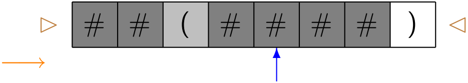
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses

(i) Move to the right to search a closed parenthesis

(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses

(i) Move to the right to search a closed parenthesis

(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses

(i) Move to the right to search a closed parenthesis

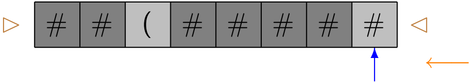
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

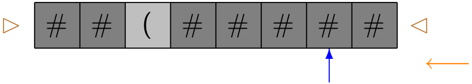
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

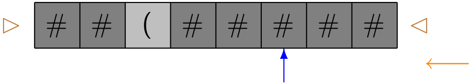
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

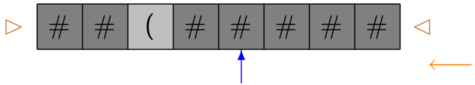
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

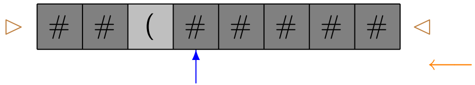
(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis
(ii) Rewrite it by #
(iii) Move to the left to search an open parenthesis
(iv) Rewrite it by #
(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

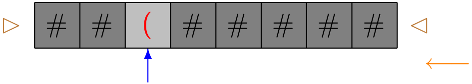
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

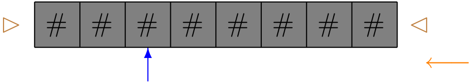
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

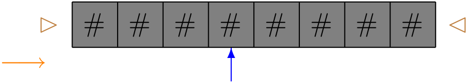
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

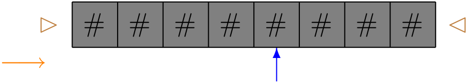
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

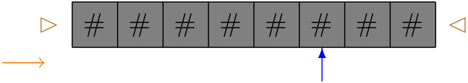
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

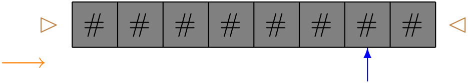
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

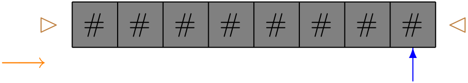
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

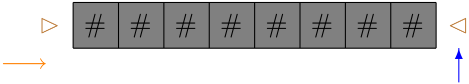
(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis
(ii) Rewrite it by #
(iii) Move to the left to search an open parenthesis
(iv) Rewrite it by #
(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

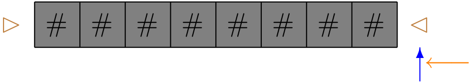
(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

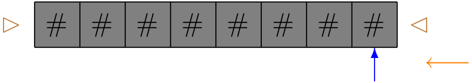
(v) Repeat from the beginning

Special cases:

(i') If in (i) the right end of the tape is reached then scan all the tape and *accept* iff all tape cells contain #

(iii') If in (iii) the left end of the tape is reached then *reject*

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

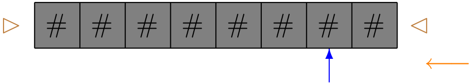
(v) Repeat from the beginning

Special cases:

(i') If in (i) the right end of the tape is reached then scan all the tape and *accept* iff all tape cells contain #

(iii') If in (iii) the left end of the tape is reached then *reject*

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

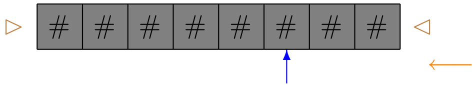
(v) Repeat from the beginning

Special cases:

(i') If in (i) the right end of the tape is reached then scan all the tape and *accept* iff all tape cells contain #

(iii') If in (iii) the left end of the tape is reached then *reject*

Example: Balanced Parentheses



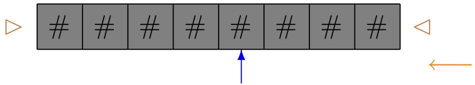
(i) Move to the right to search a closed parenthesis
(ii) Rewrite it by #
(iii) Move to the left to search an open parenthesis
(iv) Rewrite it by #
(v) Repeat from the beginning

Special cases:

(i') If in (i) the right end of the tape is reached then scan all the tape and *accept* iff all tape cells contain #

(iii') If in (iii) the left end of the tape is reached then *reject*

Example: Balanced Parentheses



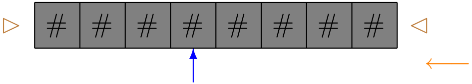
(i) Move to the right to search a closed parenthesis
(ii) Rewrite it by #
(iii) Move to the left to search an open parenthesis
(iv) Rewrite it by #
(v) Repeat from the beginning

Special cases:

(i') If in (i) the right end of the tape is reached then scan all the tape and *accept* iff all tape cells contain #

(iii') If in (iii) the left end of the tape is reached then *reject*

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

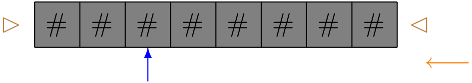
(v) Repeat from the beginning

Special cases:

(i') If in (i) the right end of the tape is reached then scan all the tape and *accept* iff all tape cells contain #

(iii') If in (iii) the left end of the tape is reached then *reject*

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

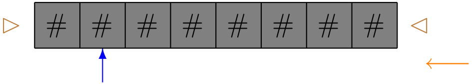
(v) Repeat from the beginning

Special cases:

(i') If in (i) the right end of the tape is reached then scan all the tape and *accept* iff all tape cells contain #

(iii') If in (iii) the left end of the tape is reached then *reject*

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

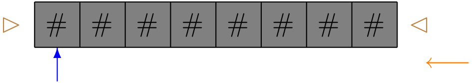
(v) Repeat from the beginning

Special cases:

(i') If in (i) the right end of the tape is reached then scan all the tape and *accept* iff all tape cells contain #

(iii') If in (iii) the left end of the tape is reached then *reject*

Example: Balanced Parentheses



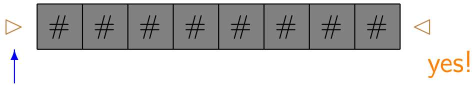
(i) Move to the right to search a closed parenthesis
(ii) Rewrite it by #
(iii) Move to the left to search an open parenthesis
(iv) Rewrite it by #
(v) Repeat from the beginning

Special cases:

(i') If in (i) the right end of the tape is reached then
scan all the tape and *accept* iff all tape cells contain #

(iii') If in (iii) the left end of the tape is reached then *reject*

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Special cases:

(i') If in (i) the right end of the tape is reached then scan all the tape and *accept* iff all tape cells contain #

(iii') If in (iii) the left end of the tape is reached then *reject*

Example: Balanced Parentheses

▷	#	#	#	#	#	#	#	#	◁

(i) Move to the right to search a closed parenthesis

(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

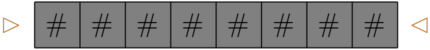
(v) Repeat from the beginning

Special cases:

(i') If in (i) the right end of the tape is reached then scan all the tape and *accept* iff all tape cells contain #

(iii') If in (iii) the left end of the tape is reached then *reject*

Example: Balanced Parentheses



(i) Move to the right to search a closed parenthesis

(ii) Rewrite it by #

(iii) Move to the left to search an open parenthesis

(iv) Rewrite it by #

(v) Repeat from the beginning

Special cases:

(i') If in (i) the right end of the tape is reached then scan all the tape and *accept* iff all tape cells contain #

(iii') If in (iii) the left end of the tape is reached then *reject*

Each cell is rewritten only in the first 2 visits!

Simulation of 2-Limited Automata by Pushdown Automata

Problem

How much it costs, in the description size, the simulation of 2-LAs *by* PDAs*?*

This work

Exponential cost!

Simulation of 2-Limited Automata by Pushdown Automata

Problem

How much it costs, in the description size, the simulation of 2-LAs *by* PDAs*?*

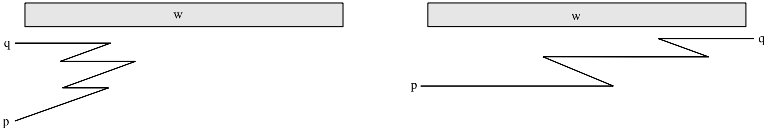
This work

Exponential cost!

Transition Tables of 2-LAs

- Fixed a 2-limited automaton
- *Transition table* τ_w $\qquad$ w is a "frozen" string

$$\tau_w \subseteq Q \times \{-1, +1\} \times Q \times \{-1, +1\}$$



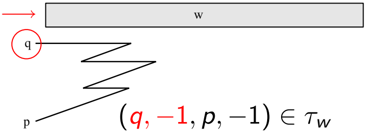
$(q, d', p, d'') \in \tau_w$ iff M on a tape segment containing w has a computation path:

- entering the segment in q from d'
- exiting the segment in p from d''
- left $= -1$, right $= +1$

Transition Tables of 2-LAs

- Fixed a 2-limited automaton
- *Transition table* τ_w — w is a "frozen" string

$$\tau_w \subseteq Q \times \{-1,+1\} \times Q \times \{-1,+1\}$$



$$(q, -1, p, -1) \in \tau_w$$

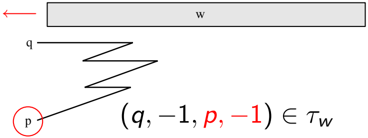
$(q, d', p, d'') \in \tau_w$ iff M on a tape segment containing w has a computation path:

- entering the segment in q from d'
- exiting the segment in p from d''
- left $= -1$, right $= +1$

Transition Tables of 2-LAs

- Fixed a 2-limited automaton
- *Transition table* τ_w $\qquad$ w is a "frozen" string

$$\tau_w \subseteq Q \times \{-1, +1\} \times Q \times \{-1, +1\}$$



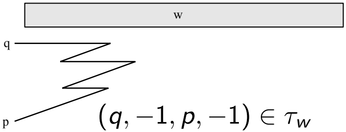
$(q, d', p, d'') \in \tau_w$ iff M on a tape segment containing w has a computation path:

- entering the segment in q from d'
- exiting the segment in p from d''
- left $= -1$, right $= +1$

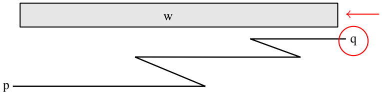
Transition Tables of 2-LAs

- Fixed a 2-limited automaton
- *Transition table* τ_w — w is a "frozen" string

$$\tau_w \subseteq Q \times \{-1,+1\} \times Q \times \{-1,+1\}$$



$(q,-1,p,-1) \in \tau_w$



$(q,+1,p,-1) \in \tau_w$

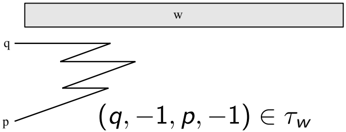
$(q,d',p,d'') \in \tau_w$ iff M on a tape segment containing w has a computation path:

- entering the segment in q from d'
- exiting the segment in p from d''
- left $= -1$, right $= +1$

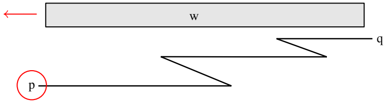
Transition Tables of 2-LAs

- Fixed a 2-limited automaton
- *Transition table* τ_w $\qquad$ w is a "frozen" string

$$\tau_w \subseteq Q \times \{-1, +1\} \times Q \times \{-1, +1\}$$



$(q, -1, p, -1) \in \tau_w$



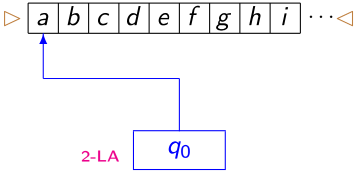
$(q, +1, p, -1) \in \tau_w$

$(q, d', p, d'') \in \tau_w$ iff M on a tape segment containing w has a computation path:

- entering the segment in q from d'
- exiting the segment in p from d''
- left $= -1$, right $= +1$

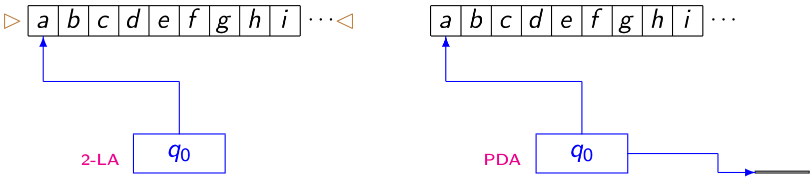
Simulation of 2-LAs by PDAs

Initial configuration



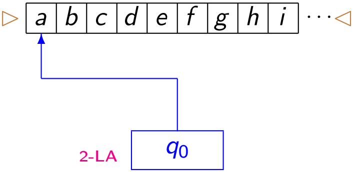
Simulation of 2-LAs by PDAs

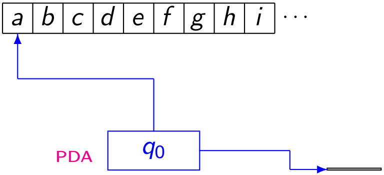
Initial configuration



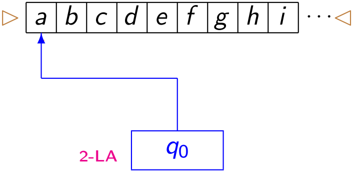
Simulation of 2-LAs by PDAs

Initial configuration





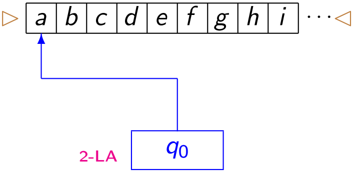
Some computation steps...

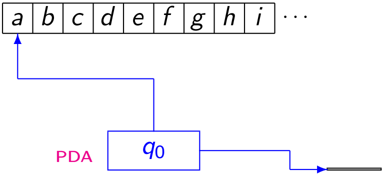


. . .

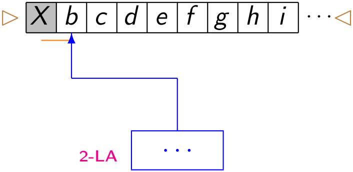
Simulation of 2-LAs by PDAs

Initial configuration





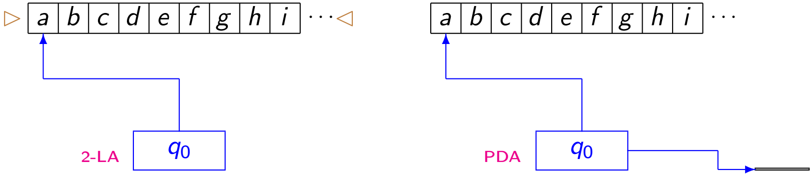
Some computation steps...



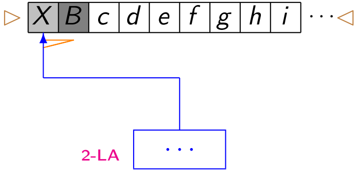
...

Simulation of 2-LAs by PDAs

Initial configuration



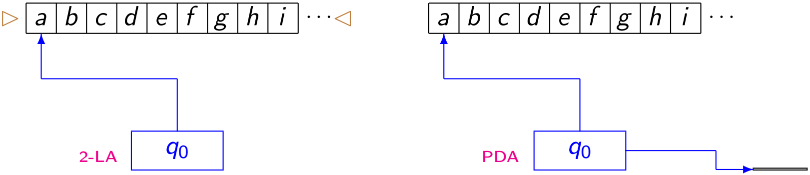
Some computation steps...



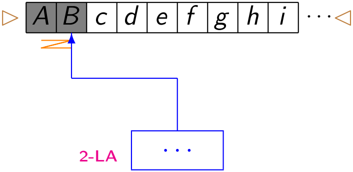
. . .

Simulation of 2-LAs by PDAs

Initial configuration



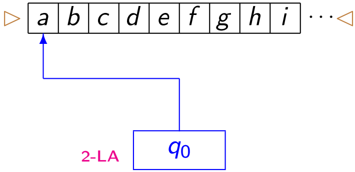
Some computation steps...

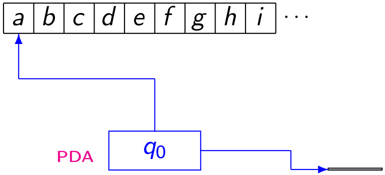


. . .

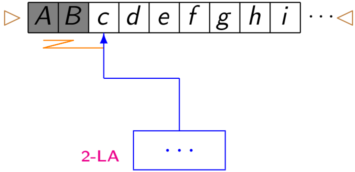
Simulation of 2-LAs by PDAs

Initial configuration





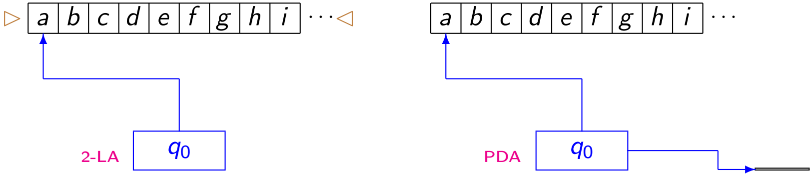
Some computation steps...



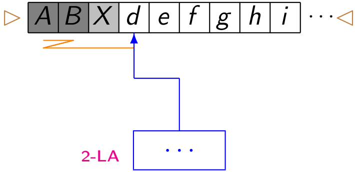
...

Simulation of 2-LAs by PDAs

Initial configuration



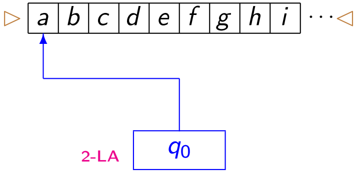
Some computation steps...

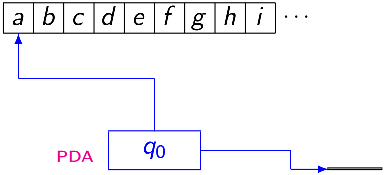


...

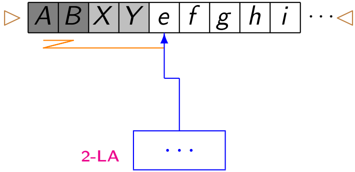
Simulation of 2-LAs by PDAs

Initial configuration





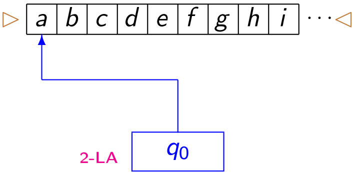
Some computation steps...

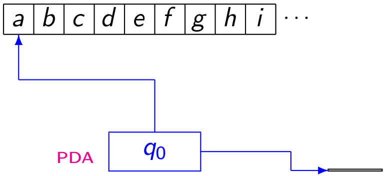


...

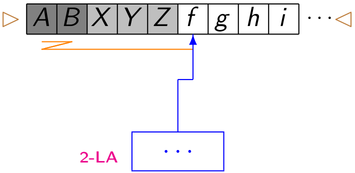
Simulation of 2-LAs by PDAs

Initial configuration





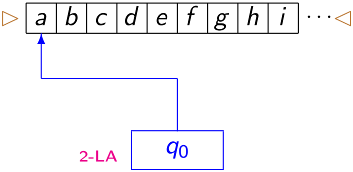
Some computation steps...

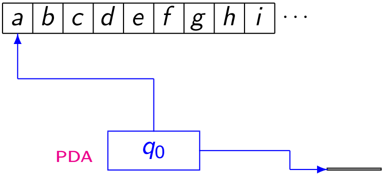


...

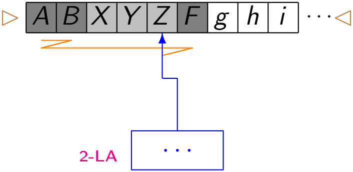
Simulation of 2-LAs by PDAs

Initial configuration





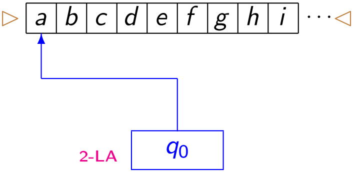
Some computation steps...

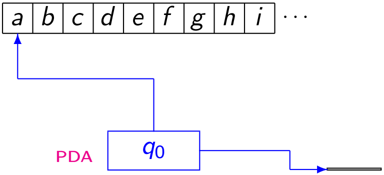


...

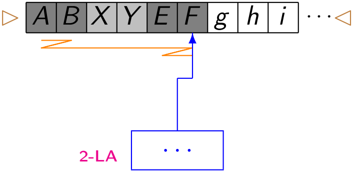
Simulation of 2-LAs by PDAs

Initial configuration





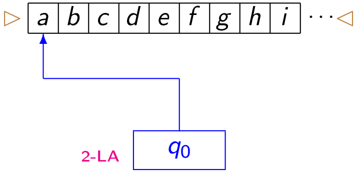
Some computation steps...

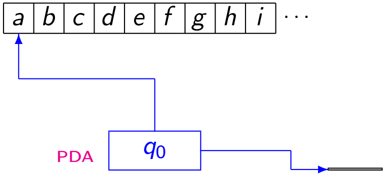


. . .

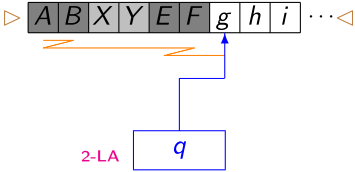
Simulation of 2-LAs by PDAs

Initial configuration





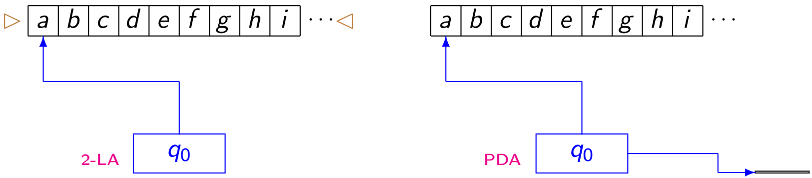
Some computation steps...



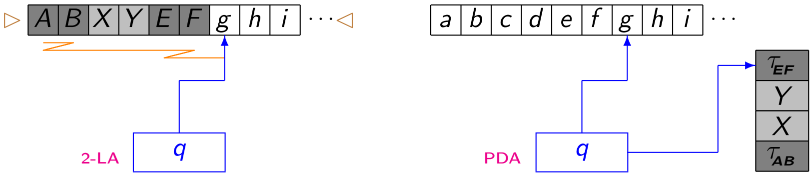
...

Simulation of 2-LAs by PDAs

Initial configuration

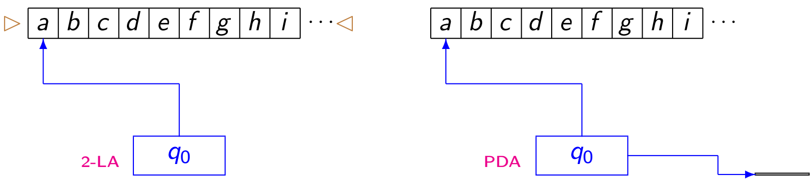


After some steps...

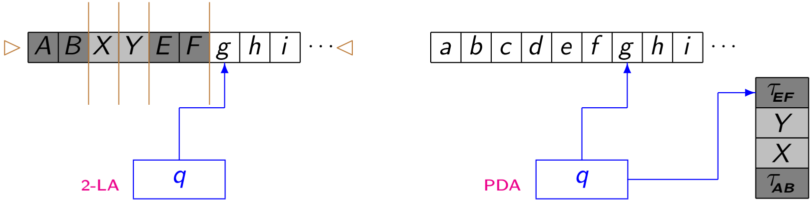


Simulation of 2-LAs by PDAs

Initial configuration

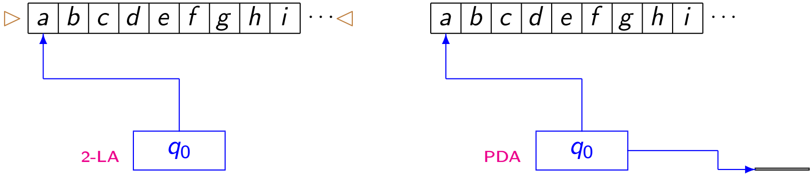


After some steps...

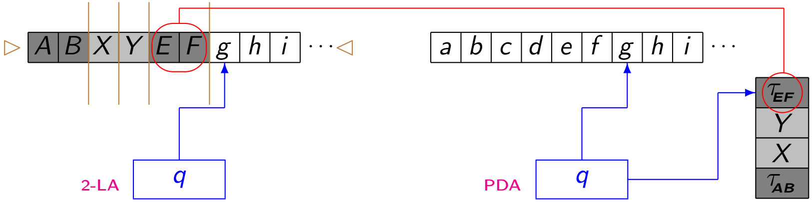


Simulation of 2-LAs by PDAs

Initial configuration

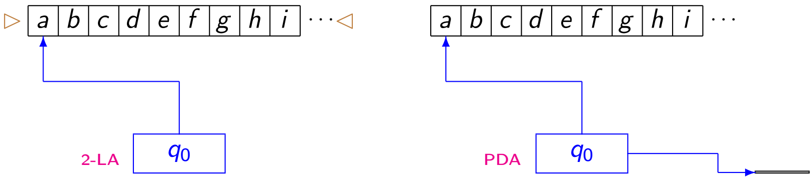


After some steps...

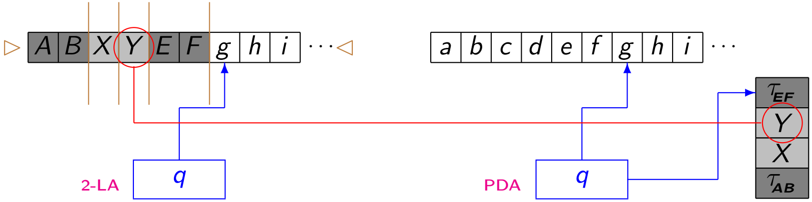


Simulation of 2-LAs by PDAs

Initial configuration

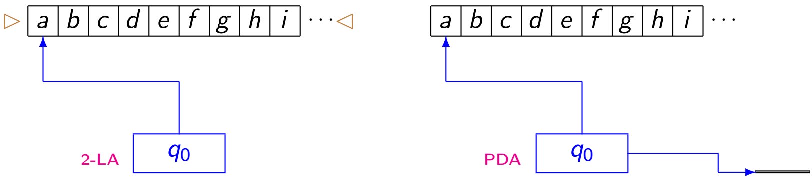


After some steps...

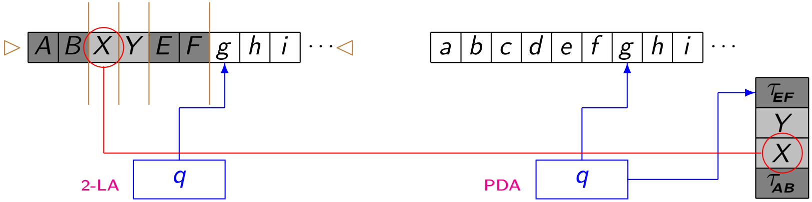


Simulation of 2-LAs by PDAs

Initial configuration

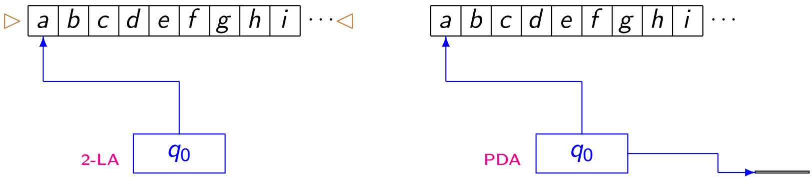


After some steps...

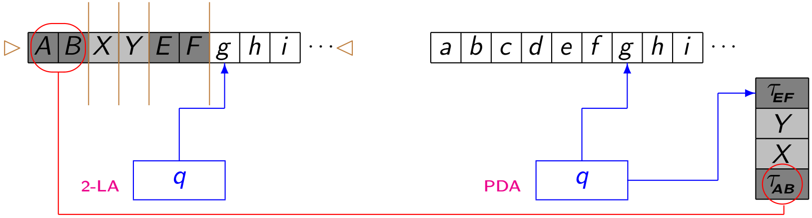


Simulation of 2-LAs by PDAs

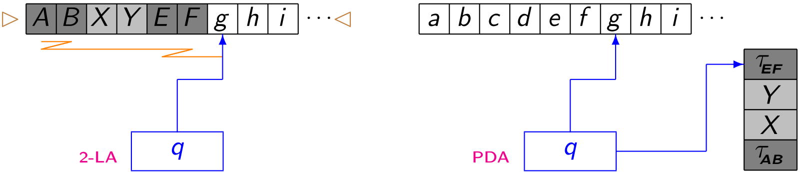
Initial configuration



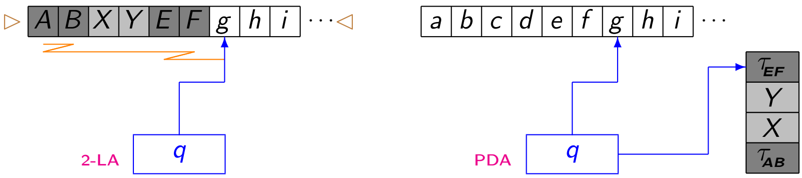
After some steps...



Simulation of 2-LAs by PDAs



Simulation of 2-LAs by PDAs

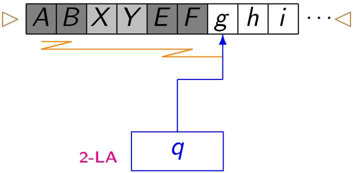


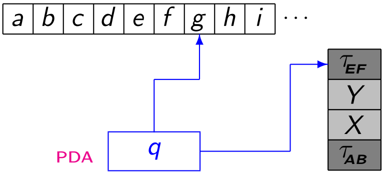
$\delta(q, g) \ni (p, Z, +1)$

move to the right

$\Downarrow$

Simulation of 2-LAs by PDAs

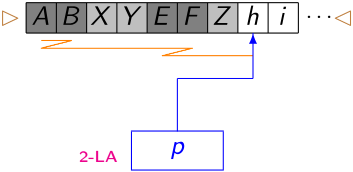




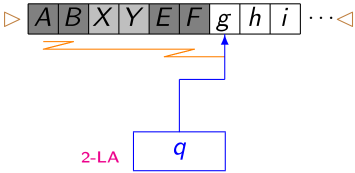
$\delta(q, g) \ni (p, Z, +1)$

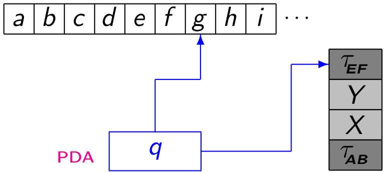
move to the right

$\Downarrow$



Simulation of 2-LAs by PDAs





$\delta(q, g) \ni (p, Z, +1)$

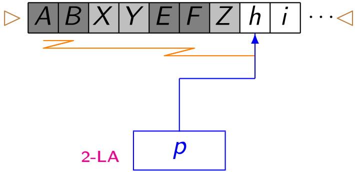
move to the right

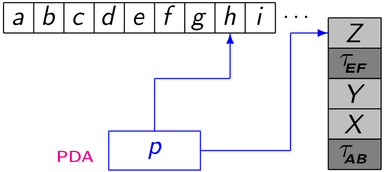
$\Downarrow$

normal mode

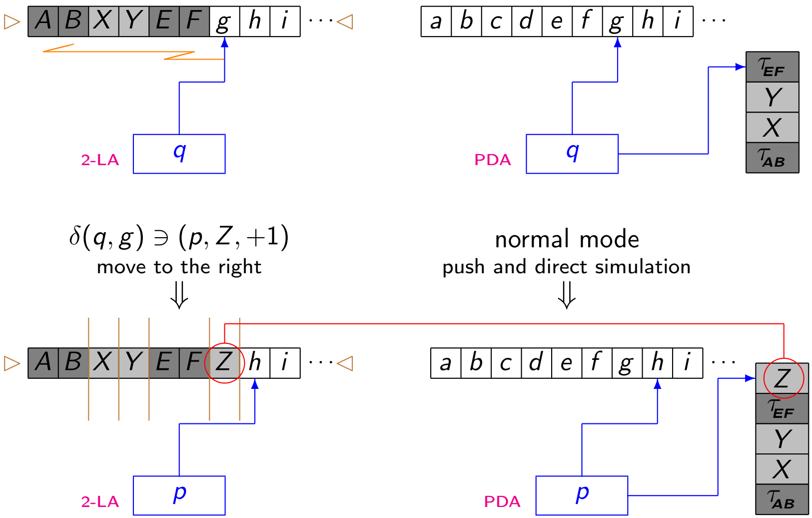
push and direct simulation

$\Downarrow$

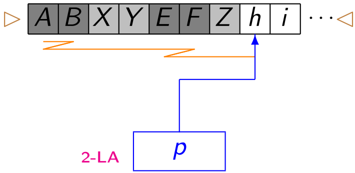


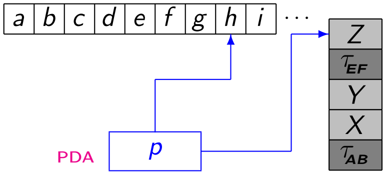


Simulation of 2-LAs by PDAs

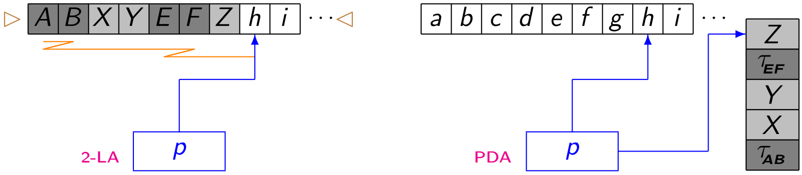


Simulation of 2-LAs by PDAs





Simulation of 2-LAs by PDAs

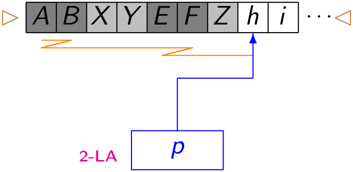


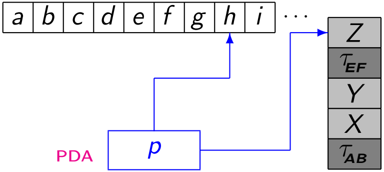
$\delta(p, h) \ni (r, H, -1)$

move to the left

$\Downarrow$

Simulation of 2-LAs by PDAs

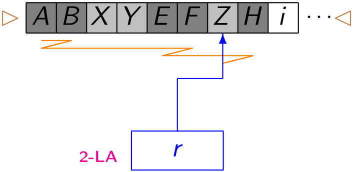




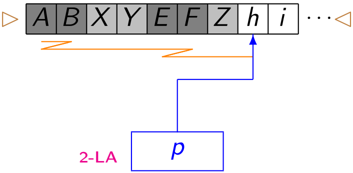
$\delta(p, h) \ni (r, H, -1)$

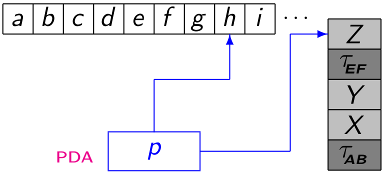
move to the left

$\Downarrow$



Simulation of 2-LAs by PDAs





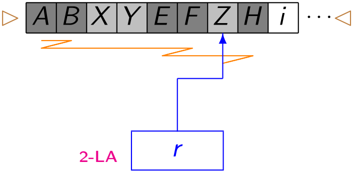
$\delta(p, h) \ni (r, H, -1)$

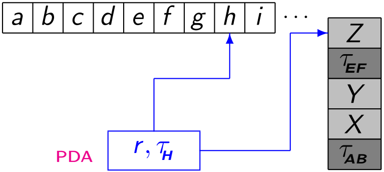
move to the left

$\Downarrow$

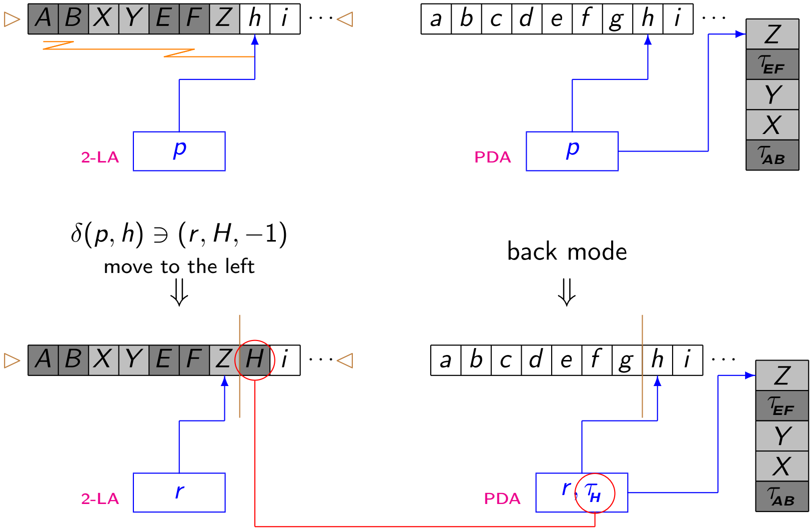
back mode

$\Downarrow$

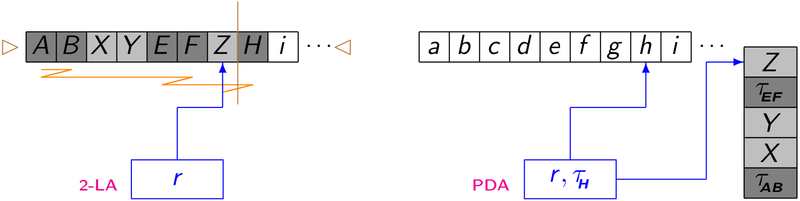




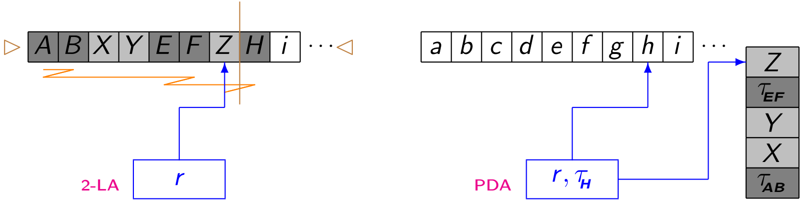
Simulation of 2-LAs by PDAs



Simulation of 2-LAs by PDAs



Simulation of 2-LAs by PDAs

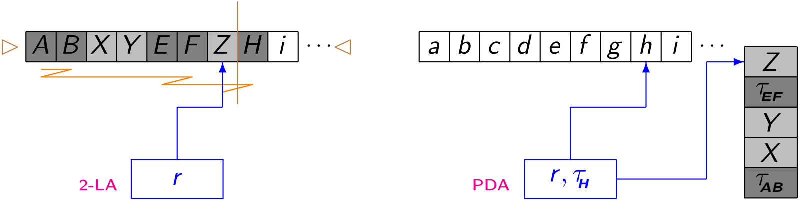


$\delta(r, Z) \ni (q, G, -1)$

move to the left

$\Downarrow$

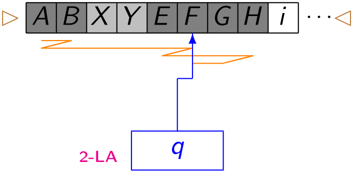
Simulation of 2-LAs by PDAs



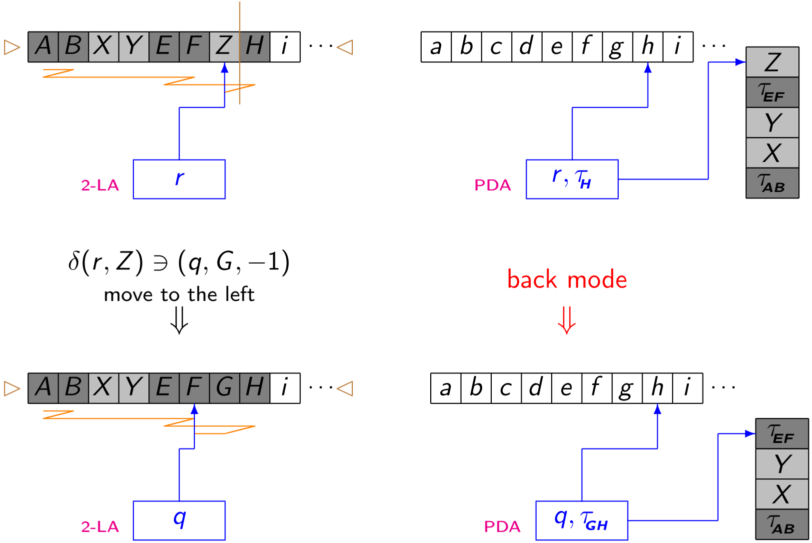
$\delta(r, Z) \ni (q, G, -1)$

move to the left

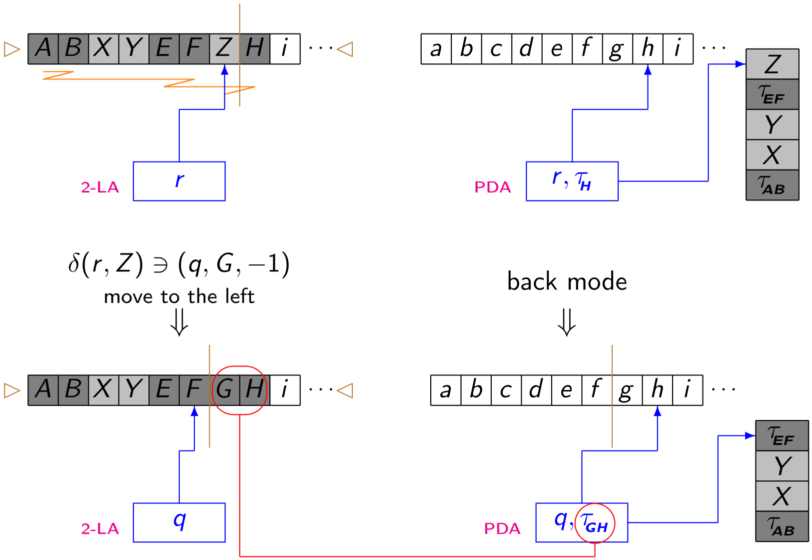
$\Downarrow$



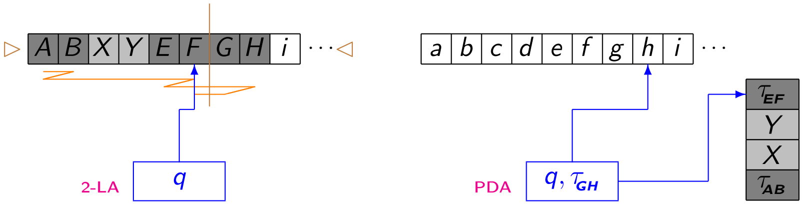
Simulation of 2-LAs by PDAs



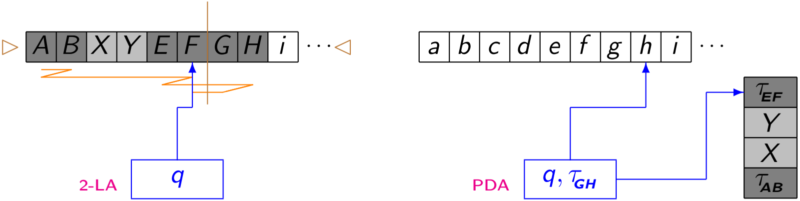
Simulation of 2-LAs by PDAs



Simulation of 2-LAs by PDAs



Simulation of 2-LAs by PDAs

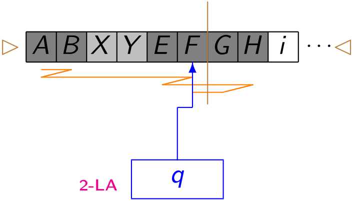


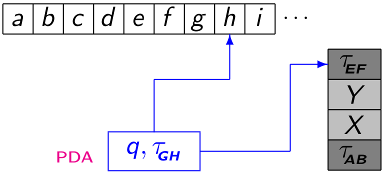
$(q, +1, s, -1) \in \mathcal{T}_{EF}$

exit to the left

$\Downarrow$

Simulation of 2-LAs by PDAs

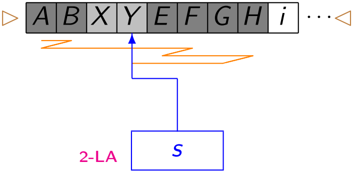




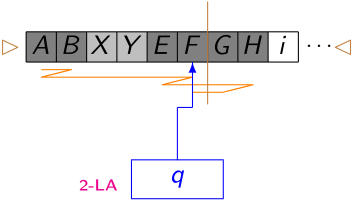
$(q, +1, s, -1) \in \mathcal{T}_{EF}$

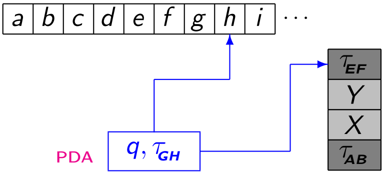
exit to the left

$\Downarrow$



Simulation of 2-LAs by PDAs





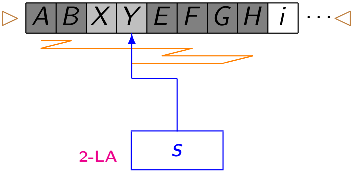
$(q, +1, s, -1) \in \mathcal{T}_{EF}$

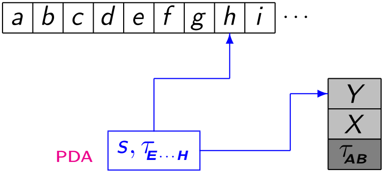
exit to the left

$\Downarrow$

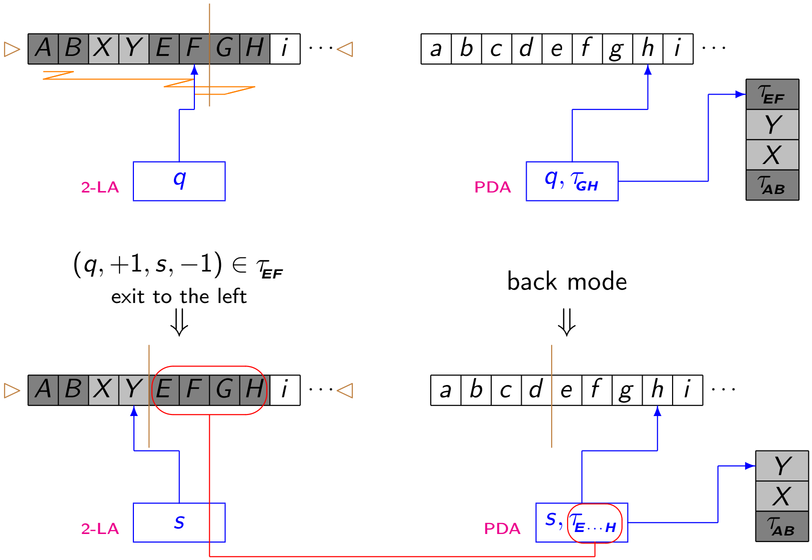
back mode

$\Downarrow$

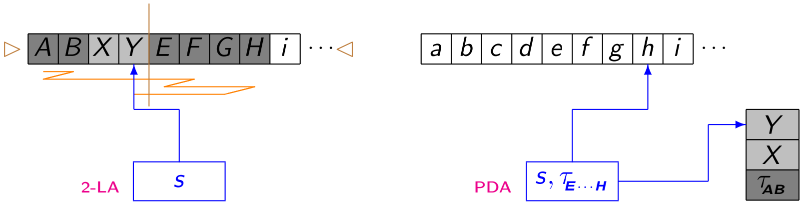




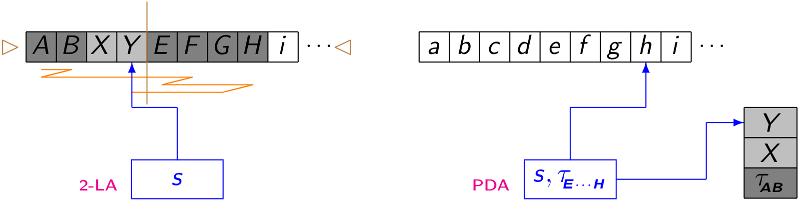
Simulation of 2-LAs by PDAs



Simulation of 2-LAs by PDAs



Simulation of 2-LAs by PDAs

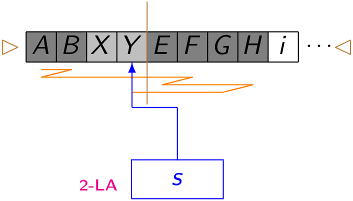


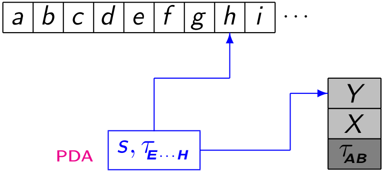
$$\delta(s, Y) \ni (p, D, +1)$$

move to the right

$\Downarrow$

Simulation of 2-LAs by PDAs

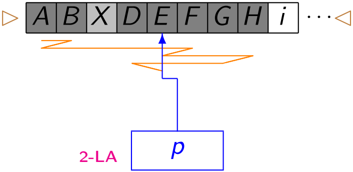




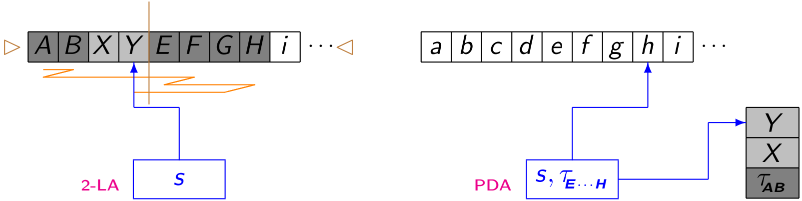
$\delta(s, Y) \ni (p, D, +1)$

move to the right

$\Downarrow$



Simulation of 2-LAs by PDAs



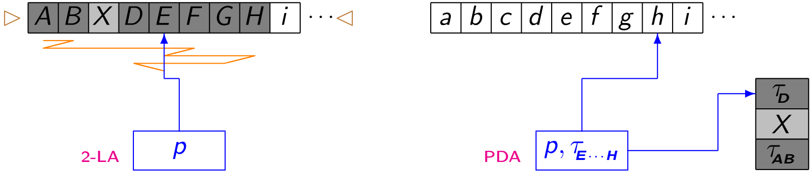
$\delta(s, Y) \ni (p, D, +1)$

move to the right

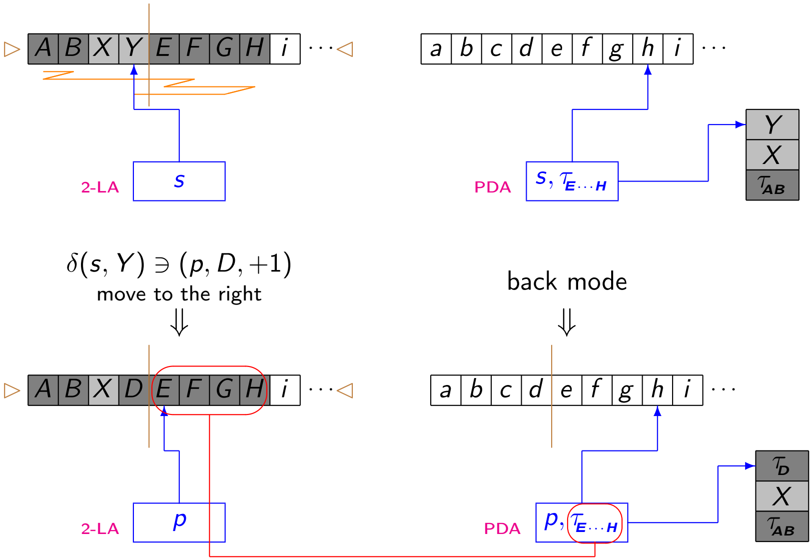
$\Downarrow$

back mode

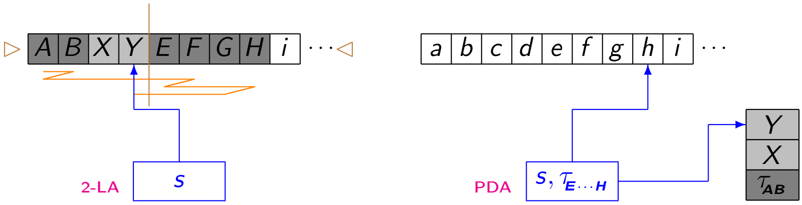
$\Downarrow$



Simulation of 2-LAs by PDAs



Simulation of 2-LAs by PDAs

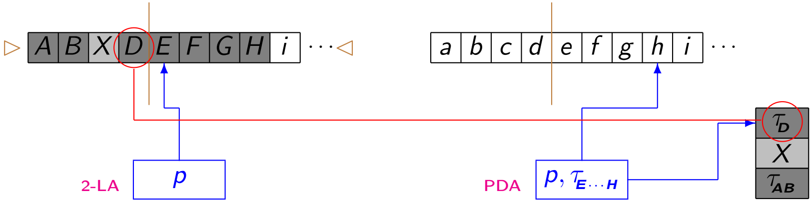


$\delta(s, Y) \ni (p, D, +1)$

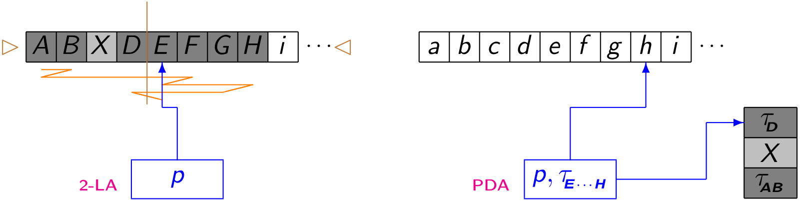
move to the right

back mode

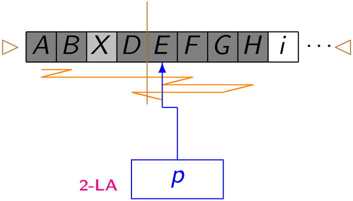
$\Downarrow$

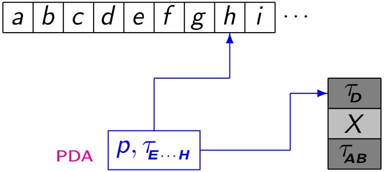


Simulation of 2-LAs by PDAs



Simulation of 2-LAs by PDAs



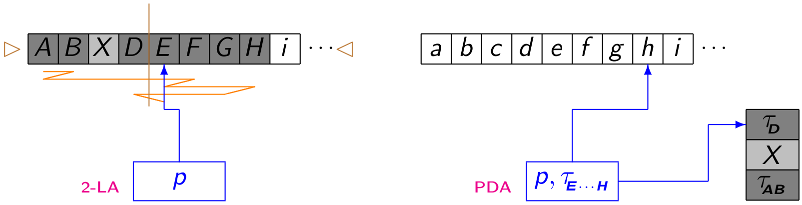


$(p, -1, r, +1) \in \tau_{E\cdots H}$

exit to the right

$\Downarrow$

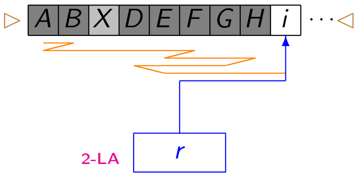
Simulation of 2-LAs by PDAs



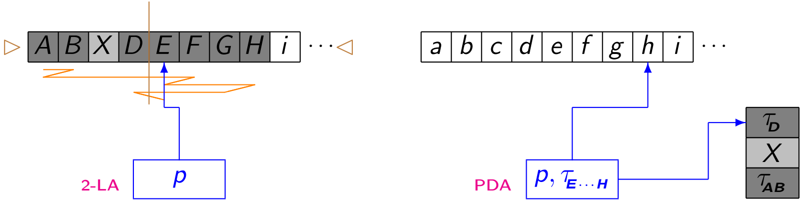
$(p, -1, r, +1) \in \tau_{E\cdots H}$

exit to the right

$\Downarrow$



Simulation of 2-LAs by PDAs



$(p, -1, r, +1) \in \mathcal{T}_{E\cdots H}$

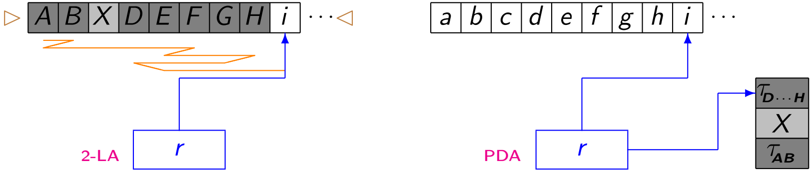
exit to the right

$\Downarrow$

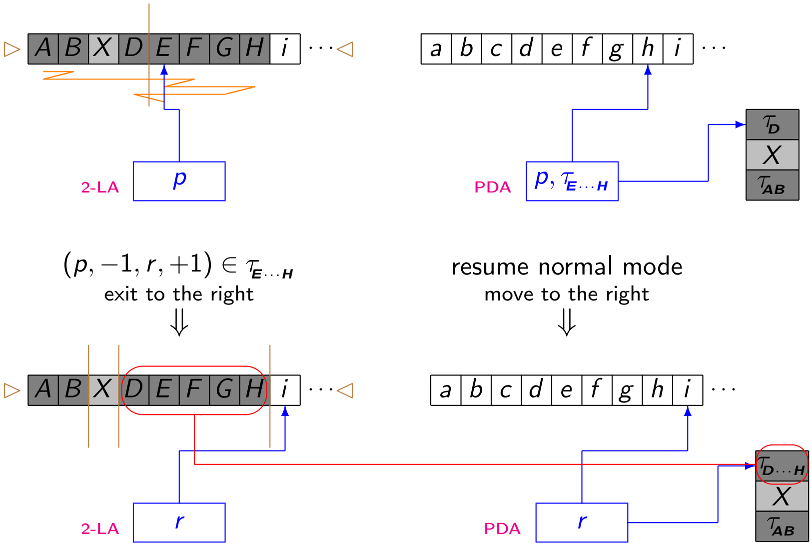
resume normal mode

move to the right

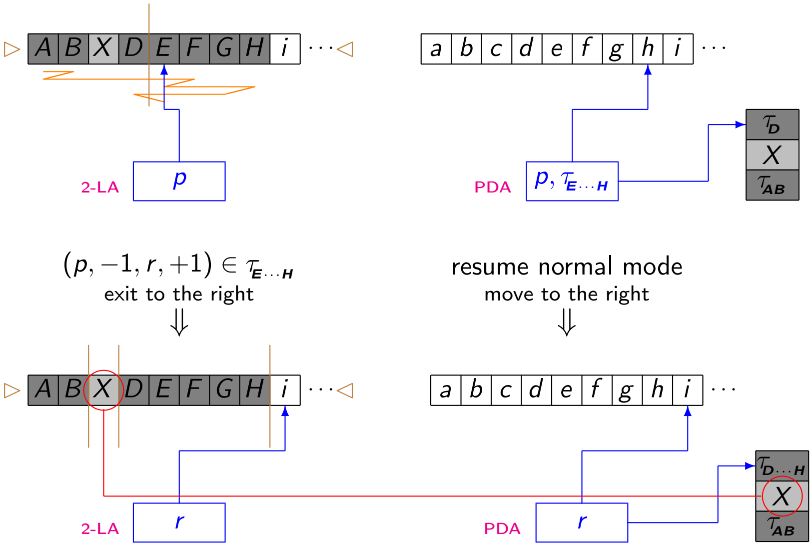
$\Downarrow$



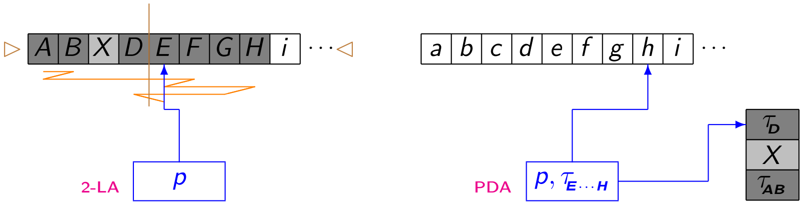
Simulation of 2-LAs by PDAs



Simulation of 2-LAs by PDAs



Simulation of 2-LAs by PDAs



$(p, -1, r, +1) \in \mathcal{T}_{E\cdots H}$

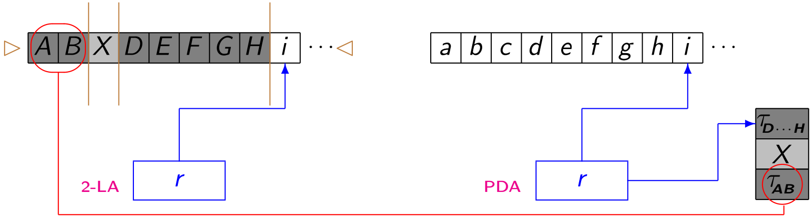
exit to the right

$\Downarrow$

resume normal mode

move to the right

$\Downarrow$



Simulation of 2-LAs by PDAs

Summing up...

Cost of the simulation

- In the resulting PDA transition tables are used for
  - states
  - pushdown alphabet
- *Exponential upper bound* for the size of the resulting PDA
- Optimal

Simulation of 2-LAs by PDAs

Summing up...

Cost of the simulation

- In the resulting PDA transition tables are used for
  - states
  - pushdown alphabet
- *Exponential upper bound* for the size of the resulting PDA
- Optimal

Cost of the simulation

- In the resulting PDA transition tables are used for
  - states
  - pushdown alphabet
- *Exponential upper bound* for the size of the resulting PDA
- Optimal

Simulation of 2-LAs by PDAs

Summing up...

Cost of the simulation

- In the resulting PDA transition tables are used for
  - states
  - pushdown alphabet
- *Exponential upper bound* for the size of the resulting PDA
- Optimal

Determinism vs nondeterminism

- Determinism is preserved by the simulation
  provided that the input of the PDA *is right end-marked*
- *Double exponential* size for the simulation of D2-LAs by DPDAs
- Conjecture: this cost cannot be reduced

Simulation of 2-LAs by PDAs

Summing up...

Cost of the simulation

- In the resulting PDA transition tables are used for
  - states
  - pushdown alphabet
- *Exponential upper bound* for the size of the resulting PDA
- Optimal

Determinism vs nondeterminism

- Determinism is preserved by the simulation
  provided that the input of the PDA *is right end-marked*
- *Double exponential* size for the simulation of D2-LAs by DPDAs
- Conjecture: this cost cannot be reduced

Simulation of 2-LAs by PDAs

Summing up...

Cost of the simulation

- In the resulting PDA transition tables are used for
  - states
  - pushdown alphabet
- *Exponential upper bound* for the size of the resulting PDA
- Optimal

Determinism vs nondeterminism

- Determinism is preserved by the simulation
  provided that the input of the PDA *is right end-marked*
- *Double exponential* size for the simulation of D2-LAs by DPDAs
- Conjecture: this cost cannot be reduced

Simulation of Pushdown Automata by 2-Limited Automata

PDAs $\longrightarrow$ 2-LAs

Polynomial cost!

(in the description size)

Simulation of Pushdown Automata by 2-Limited Automata

PDAs $\rightarrow$ 2-LAs

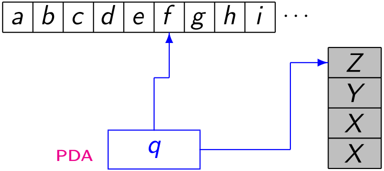
Polynomial cost!

DPDAs $\rightarrow$ D2-LAs

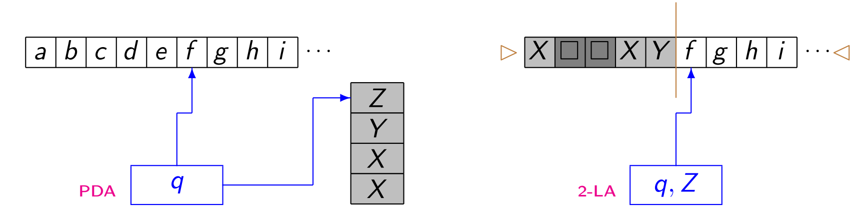
Polynomial cost!

(in the description size)

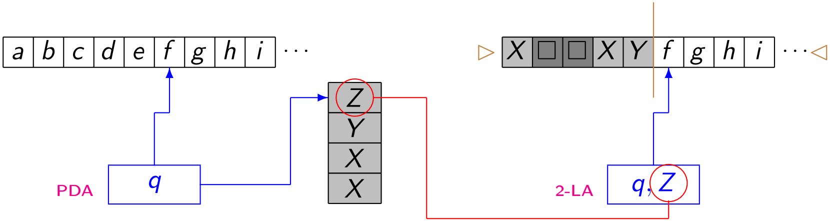
Simulation of PDAs by 2-LAs



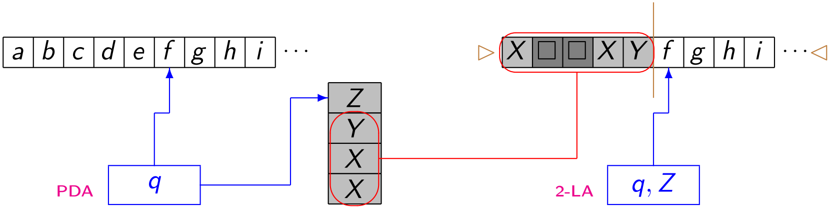
Simulation of PDAs by 2-LAs



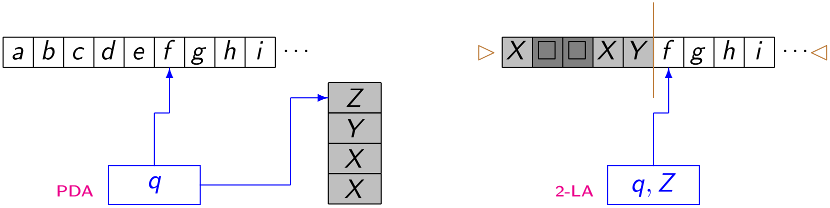
Simulation of PDAs by 2-LAs



Simulation of PDAs by 2-LAs



Simulation of PDAs by 2-LAs



Normal form for (D)PDAs:

- at each step, the stack height increases at most by 1
- ϵ-moves cannot push on the stack

Each (D)PDA can be simulated by an equivalent (D)2-LA of polynomial size

Determinism vs Nondeterminism in Limited Automata

Corollary of the simulations

Deterministic 2-LAs $\equiv$ Deterministic Context-Free Languages

On the other hand, the language

$$L = \{a^n b^n c \mid n \geq 0\} \cup \{a^n b^{2n} d \mid n \geq 0\}$$

is accepted by a *deterministic* 3-LA, but it is not a DCFL

Infinite hierarchy [Hibbard'67]

> *For each $d \geq 2$ there is a language which is accepted by a deterministic d-limited automaton and that cannot be accepted by any deterministic $(d-1)$-limited automaton*

Determinism vs Nondeterminism in Limited Automata

Corollary of the simulations

Deterministic 2-LAs $\equiv$ Deterministic Context-Free Languages

On the other hand, the language

$$L = \{a^n b^n c \mid n \geq 0\} \cup \{a^n b^{2n} d \mid n \geq 0\}$$

is accepted by a *deterministic* 3-LA, but it is not a DCFL

Infinite hierarchy [Hibbard'67]

For each $d \geq 2$ there is a language which is accepted by a deterministic d-limited automaton and that cannot be accepted by any deterministic $(d-1)$-limited automaton

Determinism vs Nondeterminism in Limited Automata

Corollary of the simulations

Deterministic 2-LAs $\equiv$ Deterministic Context-Free Languages

On the other hand, the language

$$L = \{a^n b^n c \mid n \geq 0\} \cup \{a^n b^{2n} d \mid n \geq 0\}$$

is accepted by a *deterministic* 3-LA, but it is not a DCFL

Infinite hierarchy [Hibbard'67]

> *For each $d \geq 2$ there is a language which is accepted by a deterministic d-limited automaton and that cannot be accepted by any deterministic $(d-1)$-limited automaton*

Futher Investigations

- Descriptional complexity aspects for $d > 2$

  We conjecture that for $d > 2$ the size gap from d-limited automata to PDAs remains exponential

- Descriptional complexity aspects in the unary case
  - Unary context-free language are regular [Ginbsurg&Rice'62]
  - Ex: $L_n = (a^{2^n})^*$

	size
2-LA	$O(n)$
DPDA	$O(n)$
minimal DFA	2^n
minimal 2NFA	2^n

Futher Investigations

- Descriptional complexity aspects for $d > 2$

  We conjecture that for $d > 2$ the size gap from d-limited automata to PDAs remains exponential

- Descriptional complexity aspects in the unary case
  - Unary context-free language are regular [Ginbsurg&Rice'62]
  - Ex: $L_n = (a^{2^n})^*$

	size
2-LA	$O(n)$
DPDA	$O(n)$
minimal DFA	2^n
minimal 2NFA	2^n

Futher Investigations

- Descriptional complexity aspects for $d > 2$

  We conjecture that for $d > 2$ the size gap from d-limited automata to PDAs remains exponential

- Descriptional complexity aspects in the unary case
  - Unary context-free language are regular [Ginbsurg&Rice'62]
  - Ex: $L_n = (a^{2^n})^*$

	size
2-LA	$O(n)$
DPDA	$O(n)$
minimal DFA	2^n
minimal 2NFA	2^n

Futher Investigations

- Descriptional complexity aspects for $d > 2$

  We conjecture that for $d > 2$ the size gap from d-limited automata to PDAs remains exponential

- Descriptional complexity aspects in the unary case
  - Unary context-free language are regular [Ginbsurg&Rice'62]
  - Ex: $L_n = (a^{2^n})^*$

	size
2-LA	$O(n)$
DPDA	$O(n)$
minimal DFA	2^n
minimal 2NFA	2^n

Thank you for your attention!