Two-Way Automata and Descriptional Complexity

Giovanni Pighizzini

Dipartimento di Informatica
Università degli Studi di Milano

TCS 2014
Rome, Italy – September 1-3, 2014

IFIP TC1, Working Group 1.2, *Descriptional Complexity*

UNIVERSITÀ DEGLI STUDI DI MILANO
DIPARTIMENTO DI INFORMATICA

What is Descriptional Complexity?

A Classical Example: NFAs vs DFAs

Formal language point of view:

- The class of languages recognized by NFAs coincides with the class of languages recognized by DFAs

A Classical Example: NFAs vs DFAs

Formal language point of view:

- The class of languages recognized by NFAs coincides with the class of languages recognized by DFAs

A Classical Example: NFAs vs DFAs

Formal language point of view:

- The class of languages recognized by NFAs coincides with the class of languages recognized by DFAs

Descriptional complexity point of view:

- Each n-state NFA can be simulated by a DFA *with* 2^n *states* [Rabin&Scott '59]
- For each integer n there exists a language L_n s.t.:
  - L_n is accepted by an n-state NFA
  - the minimum DFA for L_n requires 2^n states

  [Meyer&Fischer '71]
- Hence:

The exact cost, in terms of states, of the simulation of NFAs by DFAs is 2^n

A Classical Example: NFAs vs DFAs

Formal language point of view:

- The class of languages recognized by NFAs coincides with the class of languages recognized by DFAs

Descriptional complexity point of view:

- Each n-state NFA can be simulated by a DFA *with* 2^n *states* [Rabin&Scott '59]
- For each integer n there exists a language L_n s.t.:
  - L_n is accepted by an n-state NFA
  - the minimum DFA for L_n requires 2^n states

  [Meyer&Fischer '71]
- Hence:

  The exact cost, in terms of states, of the simulation of NFAs by DFAs is 2^n

A Classical Example: NFAs vs DFAs

Formal language point of view:

- The class of languages recognized by NFAs coincides with the class of languages recognized by DFAs

Descriptional complexity point of view:

- Each n-state NFA can be simulated by a DFA *with* 2^n *states* [Rabin&Scott '59]
- For each integer n there exists a language L_n s.t.:
  - L_n is accepted by an n-state NFA
  - the minimum DFA for L_n requires 2^n states

  [Meyer&Fischer '71]
- Hence:

  The exact cost, in terms of states, of the simulation of NFAs by DFAs is 2^n

A Classical Example: NFAs vs DFAs

Formal language point of view:

- The class of languages recognized by NFAs coincides with the class of languages recognized by DFAs

Descriptional complexity point of view:

- Each n-state NFA can be simulated by a DFA *with* 2^n *states* [Rabin&Scott '59]
- For each integer n there exists a language L_n s.t.:
  - L_n is accepted by an n-state NFA
  - the minimum DFA for L_n requires 2^n states

  [Meyer&Fischer '71]
- Hence:

  The exact cost, in terms of states, of the simulation of NFAs by DFAs is 2^n

Descriptional Complexity

Given

- $\mathcal{C}$ a class of languages
- $\mathcal{S}$ a formal system (e.g., class of devices, class of grammars,...) able to represent all the languages in $\mathcal{C}$

What is the *size* of the representations of the languages in $\mathcal{C}$ by the system $\mathcal{S}$?

Descriptional complexity compares different descriptions of a same class of languages:

- $\mathcal{S}'$ another formal system able to represent all the languages in $\mathcal{C}$:

Question

Find the relationships between the sizes of the representations in the system $\mathcal{S}$ and in the system $\mathcal{S}'$ of the languages of $\mathcal{C}$

Descriptional Complexity

Given

- $\mathcal{C}$ a class of languages
- $\mathcal{S}$ a formal system (e.g., class of devices, class of grammars,...) able to represent all the languages in $\mathcal{C}$

What is the *size* of the representations of the languages in $\mathcal{C}$ by the system $\mathcal{S}$?

Descriptional complexity compares different descriptions of a same class of languages:

- $\mathcal{S}'$ another formal system able to represent all the languages in $\mathcal{C}$:

Question

Find the relationships between the sizes of the representations in the system $\mathcal{S}$ and in the system $\mathcal{S}'$ of the languages of $\mathcal{C}$

Descriptional Complexity

Given

- $\mathcal{C}$ a class of languages
- $\mathcal{S}$ a formal system (e.g., class of devices, class of grammars,...) able to represent all the languages in $\mathcal{C}$

What is the *size* of the representations of the languages in $\mathcal{C}$ by the system $\mathcal{S}$?

Descriptional complexity compares different descriptions of a same class of languages:

- $\mathcal{S}'$ another formal system able to represent all the languages in $\mathcal{C}$:

Question

Find the relationships between the sizes of the representations in the system $\mathcal{S}$ and in the system $\mathcal{S}'$ of the languages of $\mathcal{C}$

Descriptional Complexity

Given

- $\mathcal{C}$ a class of languages
- $\mathcal{S}$ a formal system (e.g., class of devices, class of grammars,...) able to represent all the languages in $\mathcal{C}$

What is the *size* of the representations of the languages in $\mathcal{C}$ by the system $\mathcal{S}$?

Descriptional complexity compares different descriptions of a same class of languages:

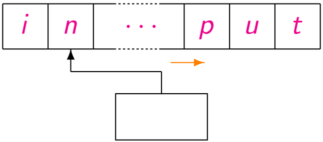
- $\mathcal{S}'$ another formal system able to represent all the languages in $\mathcal{C}$:

Question

Find the relationships between the sizes of the representations in the system $\mathcal{S}$ and in the system $\mathcal{S}'$ of the languages of $\mathcal{C}$

One-way and Two-way Automata

Finite State Automata



One-way version

At each step the input head is moved
one position to the right

- 1DFA: *deterministic* transitions
- 1NFA: *nondeterministic* transitions

A Very Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$H_n = (a + b)^{n-1} a (a + b)^*$$

A Very Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$H_n = (a+b)^{n-1} a (a+b)^*$$

Check the *n*th symbol from the left!

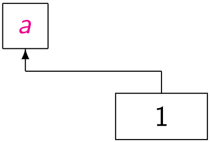
A Very Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$H_n = (a+b)^{n-1}a(a+b)^*$$

Check the *n*th symbol from the left!

Ex. $n = 4$



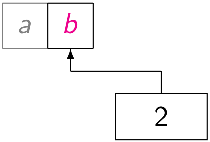
A Very Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$H_n = (a + b)^{n-1} a (a + b)^*$$

Check the nth symbol from the left!

Ex. $n = 4$



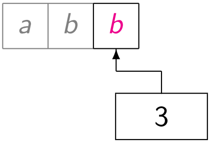
A Very Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$H_n = (a + b)^{n-1} a (a + b)^*$$

Check the *n*th symbol from the left!

Ex. $n = 4$



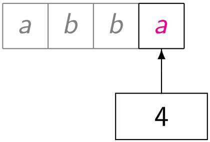
A Very Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$H_n = (a + b)^{n-1} a (a + b)^*$$

Check the nth symbol from the left!

Ex. $n = 4$



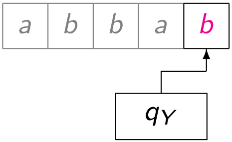
A Very Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$H_n = (a+b)^{n-1}a(a+b)^*$$

Check the *n*th symbol from the left!

Ex. $n = 4$



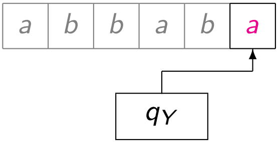
A Very Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$H_n = (a+b)^{n-1}a(a+b)^*$$

Check the nth symbol from the left!

Ex. $n = 4$



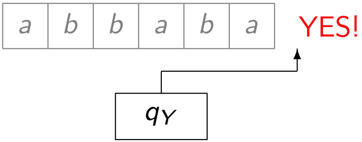
A Very Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$H_n = (a + b)^{n-1} a (a + b)^*$$

Check the nth symbol from the left!

Ex. $n = 4$



1DFA: $n + 2$ states

A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a + b)^* a(a + b)^{n-1}$$

A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a + b)^* a (a + b)^{n-1}$$

Check the *n*th symbol from the right!

A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a + b)^* a (a + b)^{n-1}$$

Check the *n*th symbol from the right!

How to locate it?

A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a + b)^* a (a + b)^{n-1}$$

Check the nth symbol from the right!

How to locate it?

Use nondeterminism!

A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a + b)^* a (a + b)^{n-1}$$

Check the *n*th symbol from the right!

How to locate it?

Use nondeterminism!

Guess Reading the symbol a the automaton can guess that it is the *n*th symbol from the right

Verify In the next steps the automaton verifies such a guess

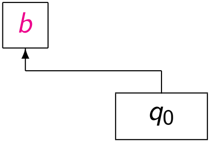
A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a+b)^* a (a+b)^{n-1}$$

Check the nth symbol from the right!

Ex. $n = 4$



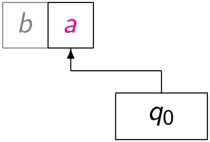
A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a + b)^* a (a + b)^{n-1}$$

Check the nth symbol from the right!

Ex. $n = 4$



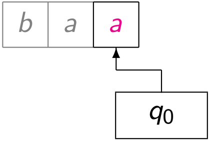
A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a + b)^* a (a + b)^{n-1}$$

Check the nth symbol from the right!

Ex. $n = 4$



guess
4th symbol from the right

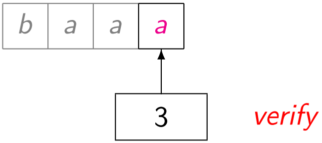
A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a + b)^* a (a + b)^{n-1}$$

Check the nth symbol from the right!

Ex. $n = 4$



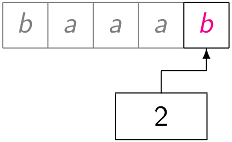
A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a+b)^* a (a+b)^{n-1}$$

Check the *n*th symbol from the right!

Ex. $n = 4$



verify

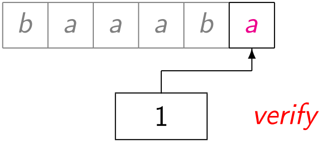
A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a+b)^* a (a+b)^{n-1}$$

Check the nth symbol from the right!

Ex. $n = 4$



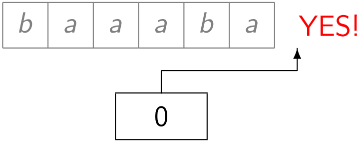
A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a+b)^* a (a+b)^{n-1}$$

Check the nth symbol from the right!

Ex. $n = 4$



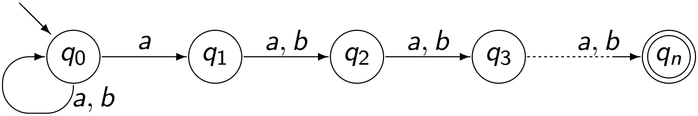
1NFA: $n + 1$ states

A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a + b)^* a (a + b)^{n-1}$$

Check the *n*th symbol from the right!

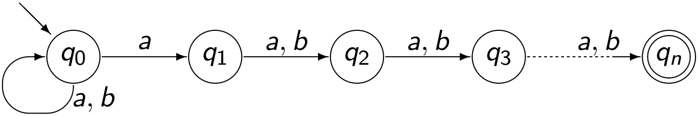


A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a+b)^* a (a+b)^{n-1}$$

Check the *n*th symbol from the right!



Very nice!

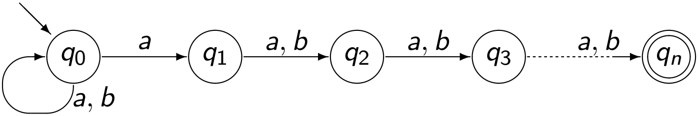
...but I need a *deterministic* automaton...

A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a + b)^* a (a + b)^{n-1}$$

Check the *n*th symbol from the right!



Very nice!

...but I need a *deterministic* automaton...

Remember the previous *n* input symbols!

A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a + b)^* a (a + b)^{n-1}$$

Check the nth symbol from the right!

Ex. $n = 4$

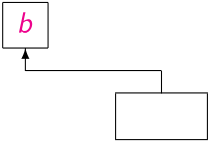
A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a+b)^* a (a+b)^{n-1}$$

Check the nth symbol from the right!

Ex. $n = 4$



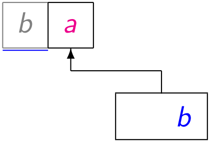
A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a+b)^* a (a+b)^{n-1}$$

Check the nth symbol from the right!

Ex. $n = 4$



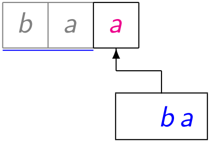
A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a+b)^* a (a+b)^{n-1}$$

Check the *n*th symbol from the right!

Ex. $n = 4$



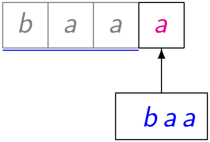
A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a+b)^* a (a+b)^{n-1}$$

Check the nth symbol from the right!

Ex. $n = 4$



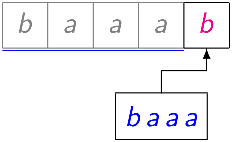
A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a+b)^* a (a+b)^{n-1}$$

Check the nth symbol from the right!

Ex. $n = 4$



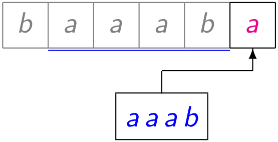
A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a+b)^* a (a+b)^{n-1}$$

Check the nth symbol from the right!

Ex. $n = 4$



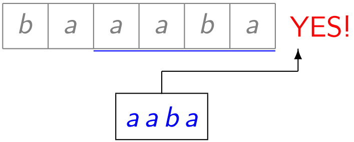
A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a + b)^* a (a + b)^{n-1}$$

Check the *n*th symbol from the right!

Ex. $n = 4$



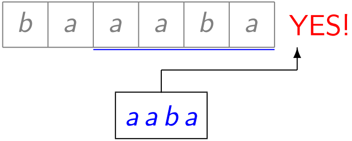
A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$I_n = (a + b)^* a (a + b)^{n-1}$$

Check the nth symbol from the right!

Ex. $n = 4$



1DFA: 2^n states

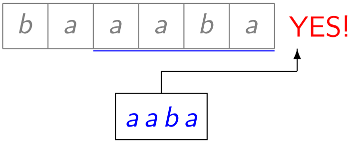
A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$I_n = (a+b)^* a (a+b)^{n-1}$$

Check the nth symbol from the right!

Ex. $n = 4$



1DFA: 2^n states

...but I need a smaller deterministic automaton...

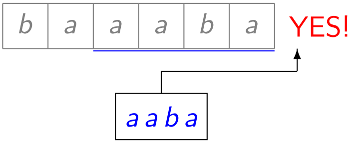
A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$I_n = (a+b)^* a (a+b)^{n-1}$$

Check the nth symbol from the right!

Ex. $n = 4$



1DFA: 2^n states

...but I need a smaller deterministic automaton...

This is the smallest one!

However...

A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a+b)^* a (a+b)^{n-1}$$

Check the *n*th symbol from the right!

...if the head can be moved back...

A Preliminary Example

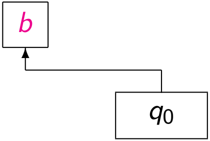
$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a+b)^* a (a+b)^{n-1}$$

Check the nth symbol from the right!

...if the head can be moved back...

Ex. $n = 4$



A Preliminary Example

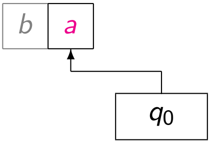
$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a+b)^* a (a+b)^{n-1}$$

Check the nth symbol from the right!

...if the head can be moved back...

Ex. $n = 4$



A Preliminary Example

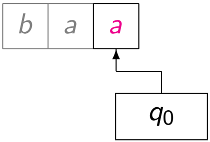
$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a+b)^* a (a+b)^{n-1}$$

Check the nth symbol from the right!

...if the head can be moved back...

Ex. $n = 4$



A Preliminary Example

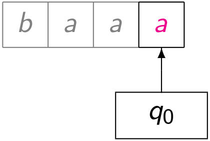
$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a + b)^* a (a + b)^{n-1}$$

Check the nth symbol from the right!

...if the head can be moved back...

Ex. $n = 4$



A Preliminary Example

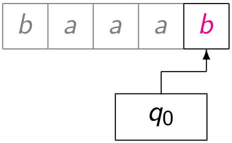
$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a+b)^* a (a+b)^{n-1}$$

Check the nth symbol from the right!

...if the head can be moved back...

Ex. $n = 4$



A Preliminary Example

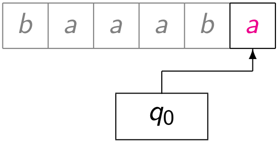
$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a+b)^* a (a+b)^{n-1}$$

Check the nth symbol from the right!

...if the head can be moved back...

Ex. $n = 4$



A Preliminary Example

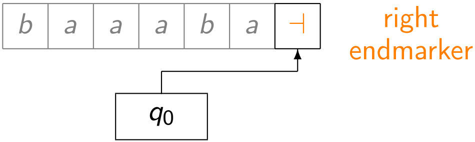
$\Sigma = \{a, b\}$, fixed $n > 0$:

$$I_n = (a+b)^* a (a+b)^{n-1}$$

Check the nth symbol from the right!

...if the head can be moved back...

Ex. $n = 4$



A Preliminary Example

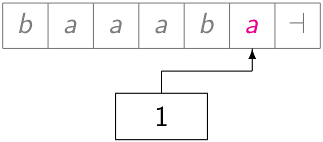
$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a + b)^* a (a + b)^{n-1}$$

Check the nth symbol from the right!

...if the head can be moved back...

Ex. $n = 4$



A Preliminary Example

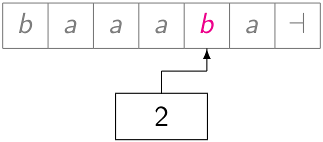
$\Sigma = \{a, b\}$, fixed $n > 0$:

$$I_n = (a + b)^* a (a + b)^{n-1}$$

Check the nth symbol from the right!

...if the head can be moved back...

Ex. $n = 4$



A Preliminary Example

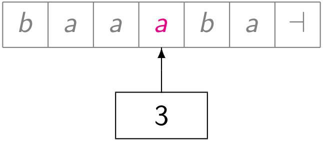
$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a + b)^* a (a + b)^{n-1}$$

Check the nth symbol from the right!

...if the head can be moved back...

Ex. $n = 4$



A Preliminary Example

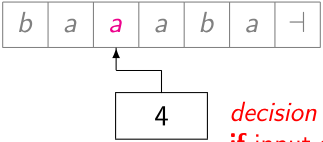
$\Sigma = \{a, b\}$, fixed $n > 0$:

$$l_n = (a + b)^* a (a + b)^{n-1}$$

Check the *n*th symbol from the right!

...if the head can be moved back...

Ex. $n = 4$



decision

if input symbol $= a$ **then accept**

else reject

A Preliminary Example

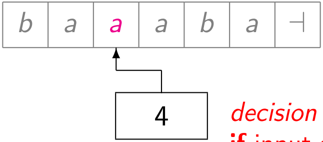
$\Sigma = \{a, b\}$, fixed $n > 0$:

$$I_n = (a+b)^* a (a+b)^{n-1}$$

Check the nth symbol from the right!

...if the head can be moved back...

Ex. $n = 4$



YES!

decision
if input symbol $= a$ **then accept**
else reject

A Preliminary Example

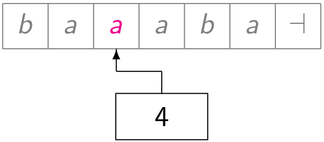
$\Sigma = \{a, b\}$, fixed $n > 0$:

$$I_n = (a+b)^* a (a+b)^{n-1}$$

Check the *n*th symbol from the right!

...if the head can be moved back...

Ex. $n = 4$



YES!

Two-way deterministic automaton (2DFA): $n + \ldots$ states

A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$I_n = (a + b)^* a (a + b)^{n-1}$$

Check the nth symbol from the right!

Summing up, I_n is accepted by

- a 1NFA and a 2DFA with approximatively the same number of states $n + \ldots$
- each 1DFA is exponentially larger ($\geq 2^n$ states)

A Preliminary Example

$\Sigma = \{a, b\}$, fixed $n > 0$:

$$I_n = (a+b)^* a (a+b)^{n-1}$$

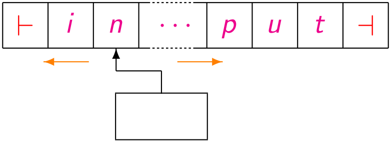
Check the nth symbol from the right!

Summing up, I_n is accepted by

- a 1NFA and a 2DFA with approximatively the same number of states $n + \ldots$
- each 1DFA is exponentially larger ($\geq 2^n$ states)

In this example,
nondeterminism can be removed using two-way motion keeping approximatively the same number of states

Two-Way Automata: Technical Details



- Input surrounded by the *endmarkers* $\vdash$ and $\dashv$
- Moves
  - to the *left*
  - to the *right*
  - *stationary*
- Initial configuration
- Accepting configuration
- Infinite computations are possible
- *Deterministic* (2DFA) and *nondeterministic* (2NFA) versions

1DFA, 1NFA, 2DFA, 2NFA

What about the power of these models?

1DFA, 1NFA, 2DFA, 2NFA

What about the power of these models?

They share the same computational power, namely they characterize the class of *regular languages*,

1DFA, 1NFA, 2DFA, 2NFA

What about the power of these models?

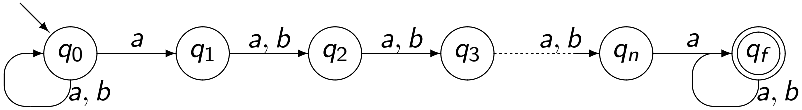
They share the same computational power, namely they characterize the class of *regular languages*, however...

...some of them are more succinct

Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$

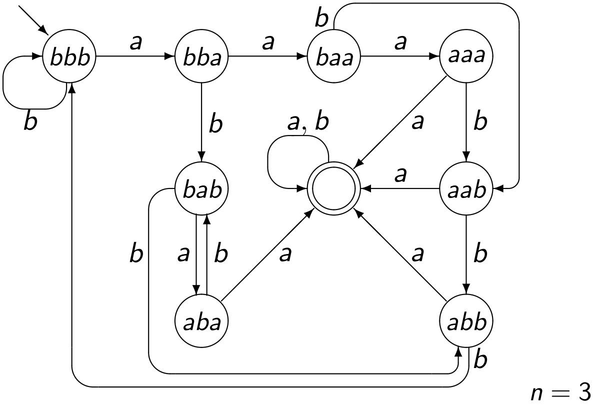
1NFA: $n+2$ states

Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$



1NFA: $n+2$ states

Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$



Minimum 1DFA: $2^n + 1$ states

Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$

2DFA ?

Even scanning from the right it seems that
we need to remember a “window” of n symbols

We use a different technique!

Main Example: $L_n = (a+b)^*a(a+b)^{n-1}a(a+b)^*$

2DFA ?

Even scanning from the right it seems that
we need to remember a "window" of n symbols

We use a different technique!

Main Example: $L_n = (a+b)^*a(a+b)^{n-1}a(a+b)^*$

2DFA ?

Even scanning from the right it seems that
we need to remember a "window" of n symbols

We use a different technique!

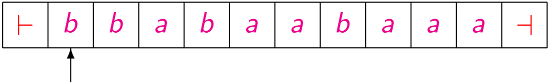
Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$

$\vdash$	b	b	a	b	a	a	b	a	a	a	$\dashv$

$n = 4$

while input symbol $\neq a$ **do** move to the right

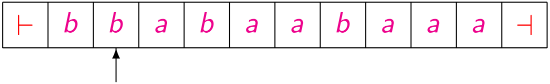
Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$



$n = 4$

while input symbol $\neq a$ **do** move to the right

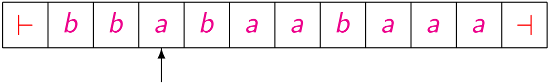
Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$



$n = 4$

while input symbol $\neq a$ **do** move to the right

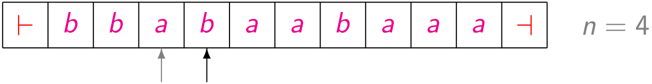
Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$



$n = 4$

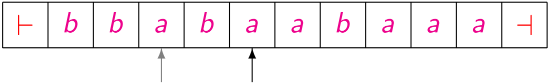
while input symbol $\neq a$ **do** move to the right
move n squares to the right

Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$



while input symbol $\neq a$ **do** move to the right
move n squares to the right

Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$

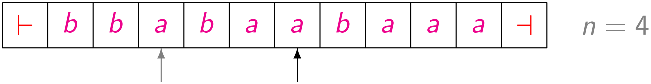


$n = 4$

while input symbol $\neq a$ **do** move to the right

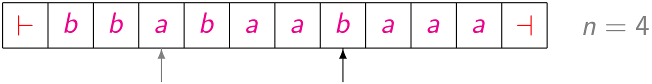
move n squares to the right

Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$



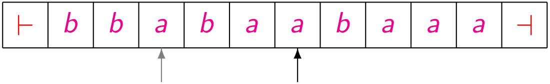
while input symbol $\neq a$ **do** move to the right
move n squares to the right

Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$



while input symbol $\neq a$ **do** move to the right
move n squares to the right
if input symbol $= a$ **then accept**
else move $n-1$ cells to the left
repeat from the first step

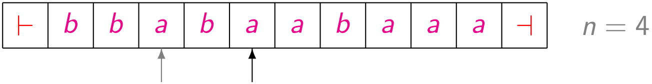
Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$



$n = 4$

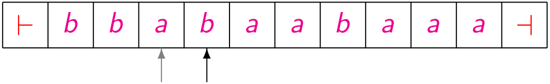
while input symbol $\neq a$ **do** move to the right
move n squares to the right
if input symbol $= a$ **then accept**
else move $n - 1$ cells to the left
repeat from the first step

Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$



while input symbol $\neq a$ **do** move to the right
move n squares to the right
if input symbol $= a$ **then accept**
else move $n-1$ cells to the left
repeat from the first step

Main Example: $L_n = (a+b)^*a(a+b)^{n-1}a(a+b)^*$



$n = 4$

while input symbol $\neq a$ **do** move to the right

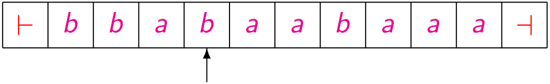
move n squares to the right

if input symbol $= a$ **then accept**

else move $n-1$ cells to the left

repeat from the first step

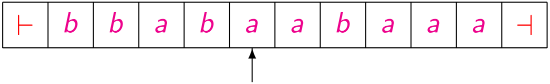
Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$



$n = 4$

while input symbol $\neq a$ **do** move to the right
move n squares to the right
if input symbol $= a$ **then accept**
else move $n-1$ cells to the left
repeat from the first step

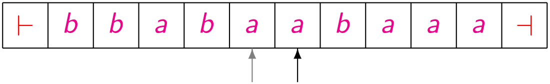
Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$



$n = 4$

while input symbol $\neq a$ **do** move to the right
move n squares to the right
if input symbol $= a$ **then accept**
else move $n-1$ cells to the left
repeat from the first step

Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$



$n = 4$

while input symbol $\neq a$ **do** move to the right

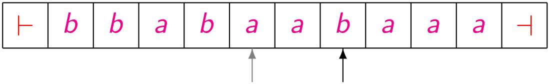
move n squares to the right

if input symbol $= a$ **then accept**

else move $n-1$ cells to the left

repeat from the first step

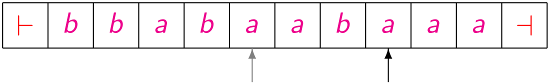
Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$



$n = 4$

while input symbol $\neq a$ **do** move to the right
move n squares to the right
if input symbol $= a$ **then accept**
else move $n-1$ cells to the left
repeat from the first step

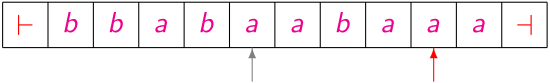
Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$



$n = 4$

while input symbol $\neq a$ **do** move to the right
move n squares to the right
if input symbol $= a$ **then accept**
else move $n-1$ cells to the left
repeat from the first step

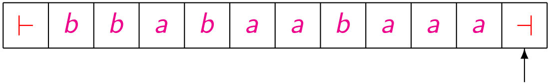
Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$



$n = 4$

while input symbol $\neq a$ **do** move to the right
move n squares to the right
if input symbol $= a$ **then accept**
else move $n-1$ cells to the left
repeat from the first step

Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$



$n = 4$

while input symbol $\neq a$ **do** move to the right
move n squares to the right
if input symbol $= a$ **then accept**
else move $n-1$ cells to the left
repeat from the first step
Exception: **if** input symbol $= \dashv$ **then reject**

Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$

$\vdash$	b	b	a	b	a	a	b	a	a	a	$\dashv$

$n = 4$

while input symbol $\neq a$ **do** move to the right
move n squares to the right
if input symbol $= a$ **then accept**
else move $n-1$ cells to the left
repeat from the first step
Exception: **if** input symbol $=\dashv$ **then reject**

2DFA: $2n+_{\ldots}$ states

Main Example: $L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$

Summing up,

- L_n is accepted by
  - a 1NFA
  - a 2DFA

  with $O(n)$ states
- Each 1DFA is exponentially larger

Also for this example,
nondeterminism can be removed using two-way motion
keeping a linear number of states

Is it always possible
to replace nondeterminism by two-way motion
without increasing too much the size?

Main Example: $L_n = (a+b)^*a(a+b)^{n-1}a(a+b)^*$

Summing up,

- L_n is accepted by
  - a 1NFA
  - a 2DFA

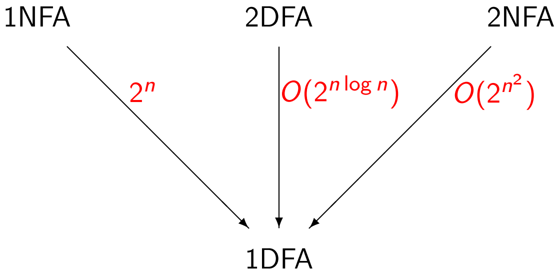
  with $O(n)$ states
- Each 1DFA is exponentially larger

Also for this example,
nondeterminism can be removed using two-way motion
keeping a linear number of states

Is it always possible
to replace nondeterminism by two-way motion
without increasing too much the size?

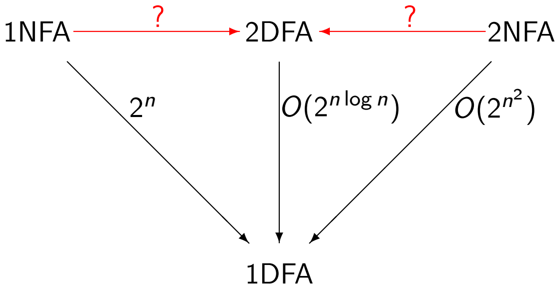
The Question of Sakoda and Sipser

Costs of the Optimal Simulations Between Automata



[Rabin&Scott '59, Shepardson '59, Meyer&Fischer '71, . . .]

Costs of the Optimal Simulations Between Automata

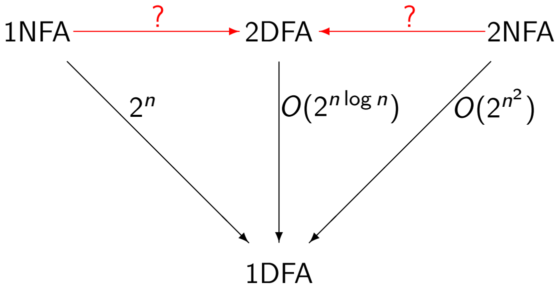


Problem ([Sakoda&Sipser '78])

Do there exist polynomial simulations of

- *1NFAs by 2DFAs*
- *2NFAs by 2DFAs ?*

Costs of the Optimal Simulations Between Automata



Problem ([Sakoda&Sipser '78])

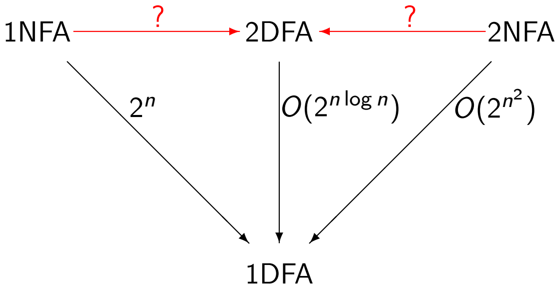
Do there exist polynomial simulations of

- *1NFAs by 2DFAs*
- *2NFAs by 2DFAs ?*

Conjecture

These simulations are not polynomial

Costs of the Optimal Simulations Between Automata



- Exponential upper bounds
  deriving from the simulations of 1NFAs and 2NFAs by 1DFAs
- Polynomial lower bound
  $\Omega(n^2)$ for the cost of the simulation of 1NFAs by 2DFAs
  [Chrobak '86]

Sakoda and Sipser Question

- Very difficult in its general form
- Not very encouraging obtained results:

  Lower and upper bounds too far
  (Polynomial vs exponential)
- Hence:

  Try to attack restricted versions of the problem!

NFAs vs 2DFAs: Restricted Versions

(i) Restrictions on the resulting machines (2DFAs)

- sweeping automata [Sipser '80]
- oblivious automata [Hromkovič&Schnitger '03]
- "few reversal" automata [Kapoutsis '11]

(ii) Restrictions on the languages

- unary regular languages [Geffert Mereghetti&P '03]

(iii) Restrictions on the starting machines (2NFAs)

- outer nondeterministic automata [Guillon Geffert&P '12]

NFAs vs 2DFAs: Restricted Versions

(i) Restrictions on the resulting machines (2DFAs)

- sweeping automata [Sipser '80]
- oblivious automata [Hromkovič&Schnitger '03]
- "few reversal" automata [Kapoutsis '11]

(ii) Restrictions on the languages

- unary regular languages [Geffert Mereghetti&P '03]

(iii) Restrictions on the starting machines (2NFAs)

- outer nondeterministic automata [Guillon Geffert&P '12]

NFAs vs 2DFAs: Restricted Versions

(i) Restrictions on the resulting machines (2DFAs)

- sweeping automata [Sipser '80]
- oblivious automata [Hromkovič&Schnitger '03]
- "few reversal" automata [Kapoutsis '11]

(ii) Restrictions on the languages

- unary regular languages [Geffert Mereghetti&P '03]

(iii) Restrictions on the starting machines (2NFAs)

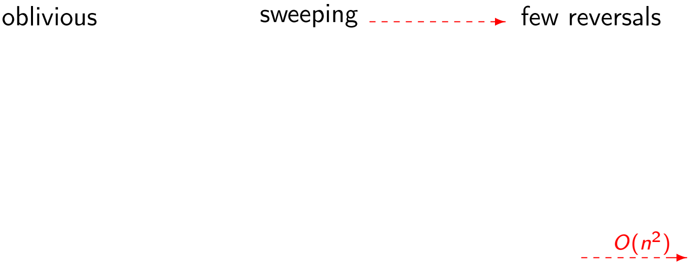
- outer nondeterministic automata [Guillon Geffert&P '12]

Restrictions on the resulting machines

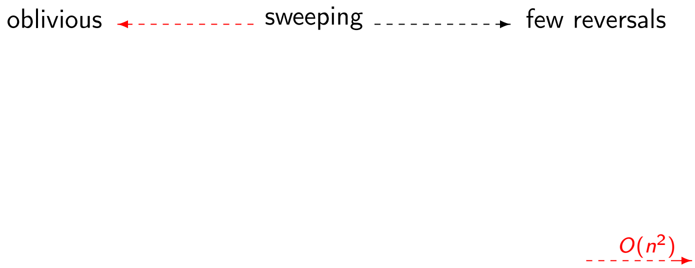
Restricted Models: Separations

oblivious sweeping few reversals

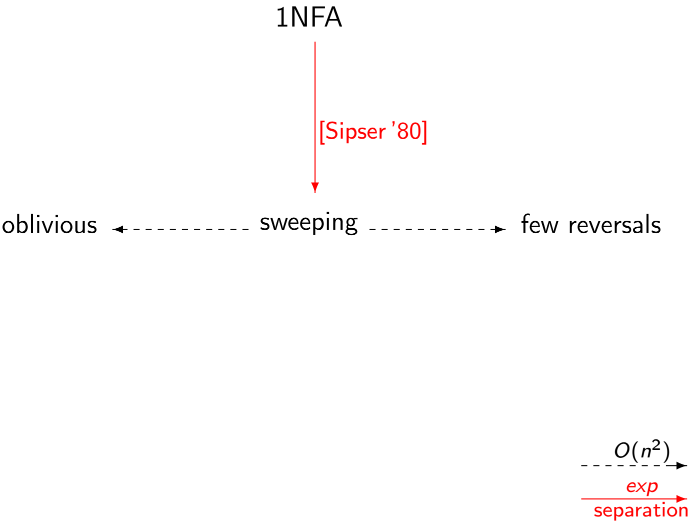
Restricted Models: Separations



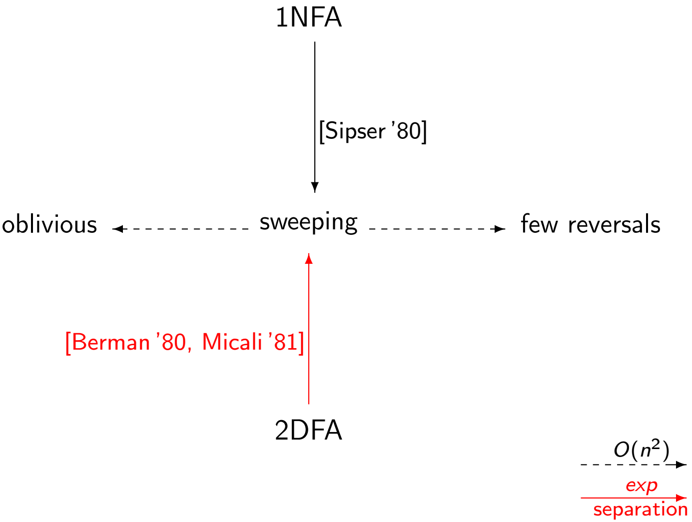
Restricted Models: Separations



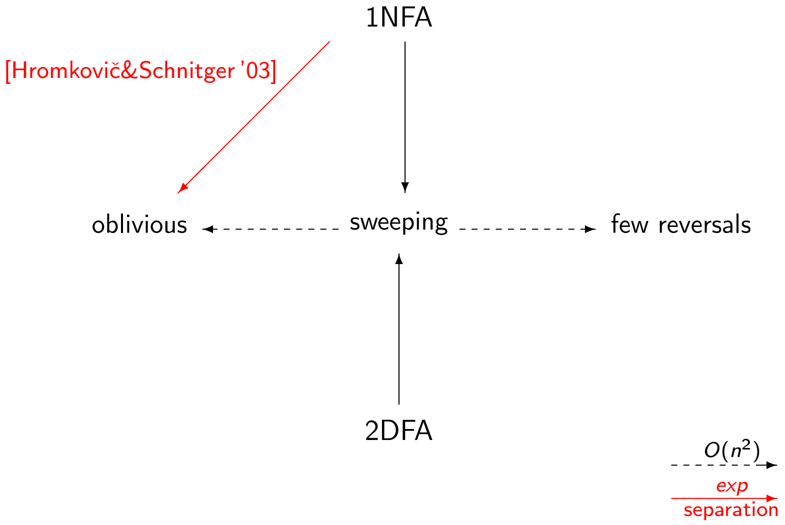
Restricted Models: Separations



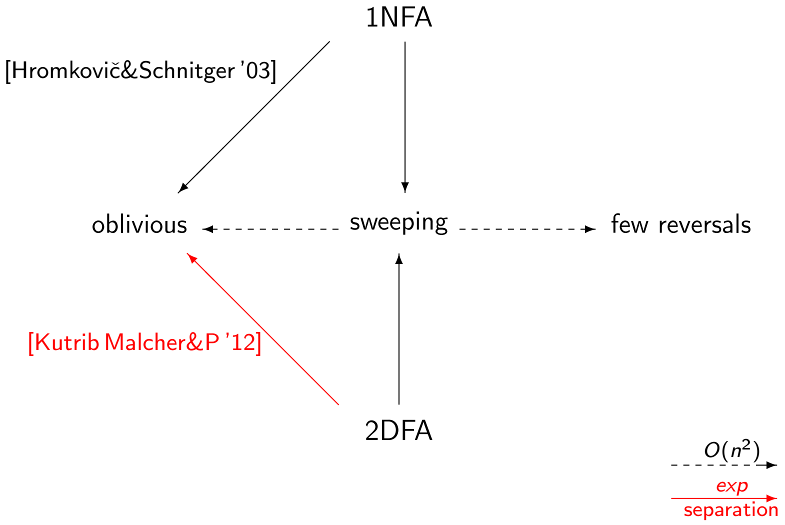
Restricted Models: Separations



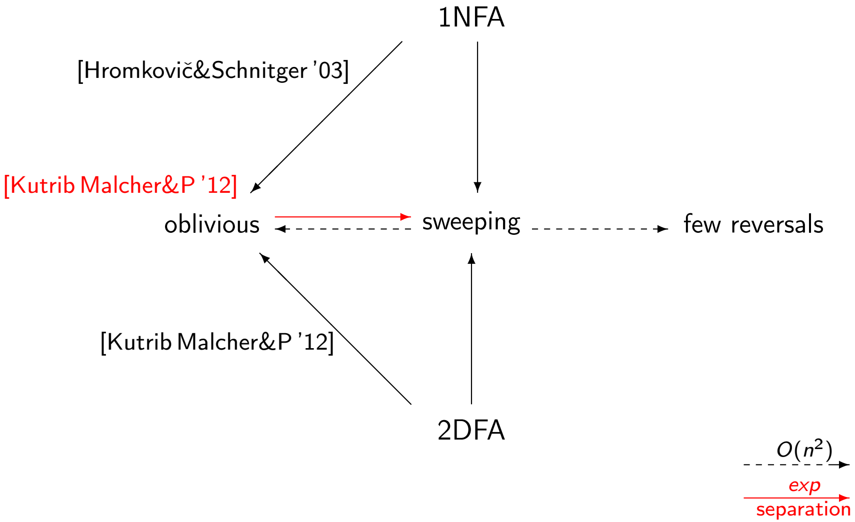
Restricted Models: Separations



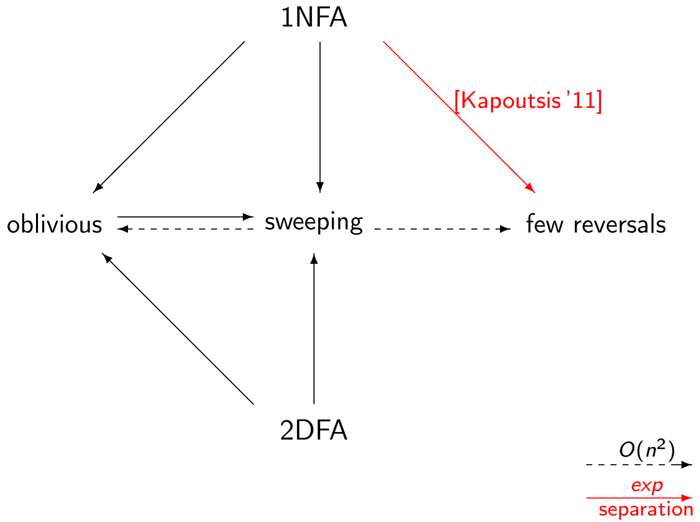
Restricted Models: Separations



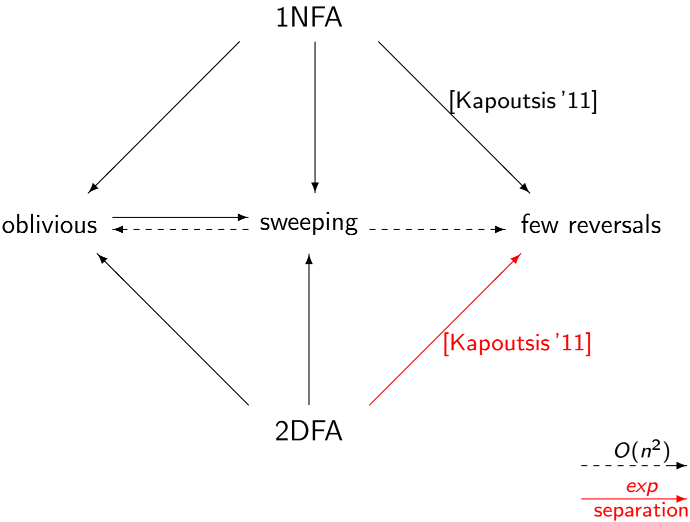
Restricted Models: Separations



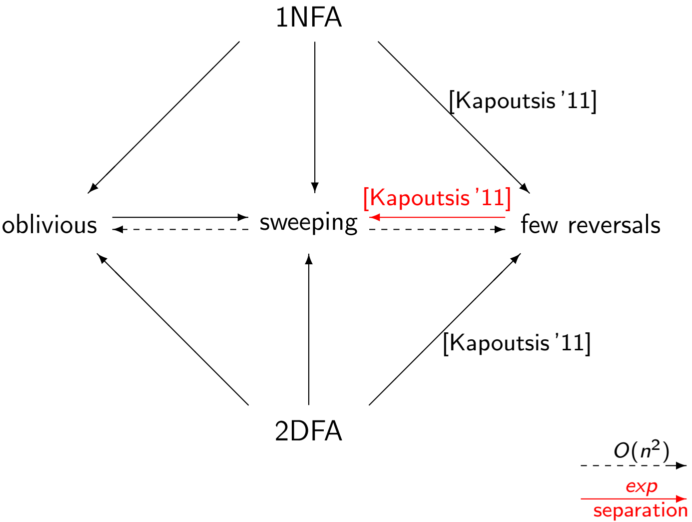
Restricted Models: Separations



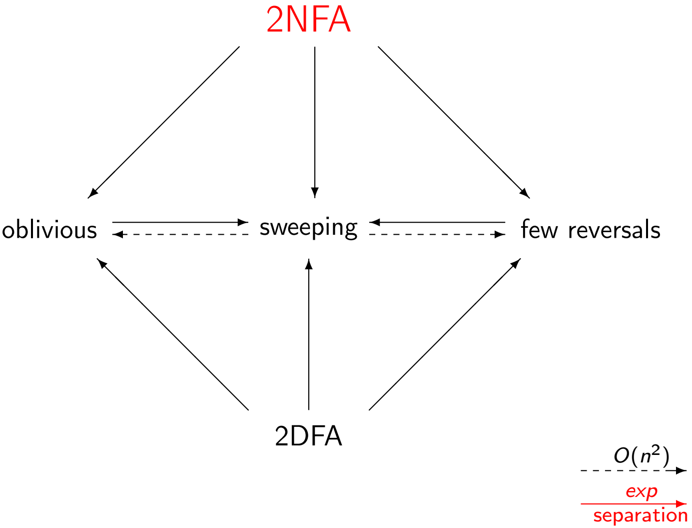
Restricted Models: Separations



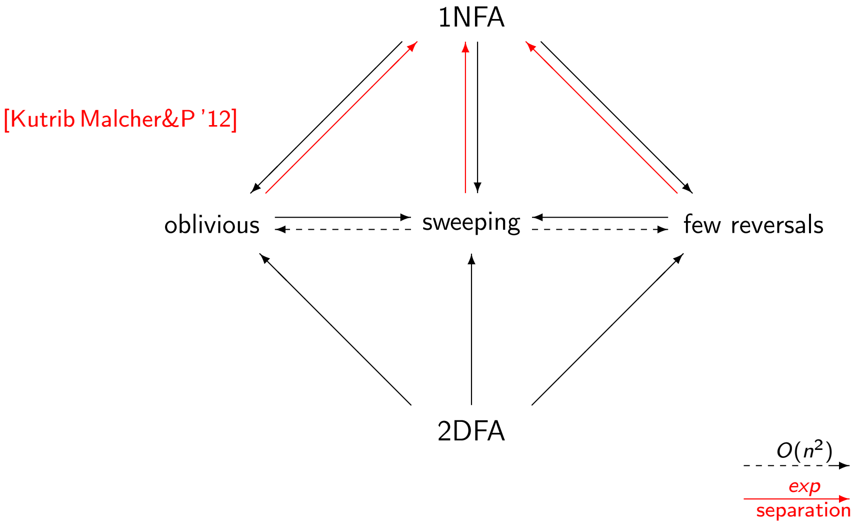
Restricted Models: Separations



Restricted Models: Separations



Restricted Models: Separations



The Unary Case

$$\#\Sigma = 1$$

Optimal Simulation Between Unary Automata

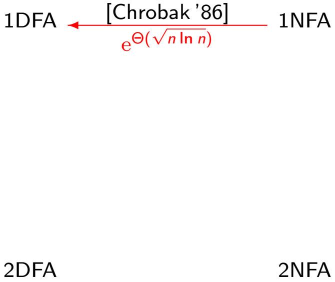
The costs of the optimal simulations between automata are different in the unary and in the general case

1DFA 1NFA

2DFA 2NFA

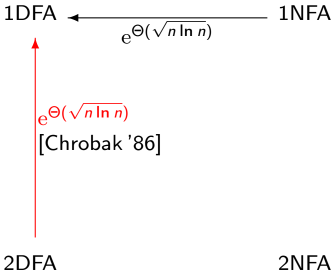
Optimal Simulation Between Unary Automata

The costs of the optimal simulations between automata are different in the unary and in the general case



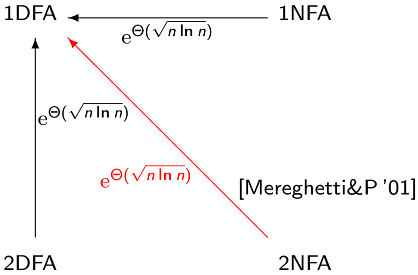
Optimal Simulation Between Unary Automata

The costs of the optimal simulations between automata are different in the unary and in the general case



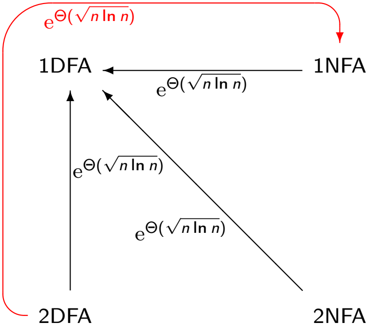
Optimal Simulation Between Unary Automata

The costs of the optimal simulations between automata are different in the unary and in the general case



Optimal Simulation Between Unary Automata

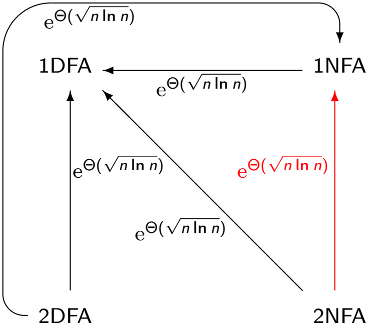
The costs of the optimal simulations between automata are different in the unary and in the general case



follows from 2DFA $\rightarrow$ 1DFA

Optimal Simulation Between Unary Automata

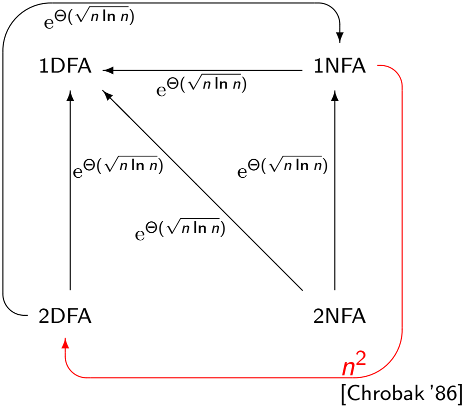
The costs of the optimal simulations between automata are different in the unary and in the general case



follows from 2NFA $\rightarrow$ 1DFA

Optimal Simulation Between Unary Automata

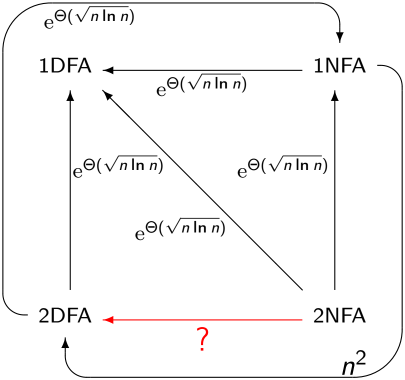
The costs of the optimal simulations between automata are different in the unary and in the general case



1NFA → 2DFA
In the unary case
this question is solved!
(polynomial conversion)

Optimal Simulation Between Unary Automata

The costs of the optimal simulations between automata are different in the unary and in the general case

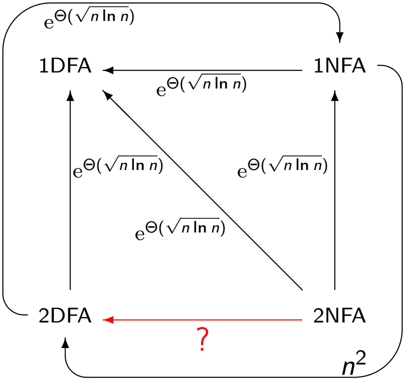


2NFA $\rightarrow$ 2DFA
Even in the unary case
this question is open!

- $e^{\Theta(\sqrt{n \ln n})}$ upper bound (from 2NFA $\rightarrow$ 1DFA)
- $\Omega(n^2)$ lower bound (from 1NFA $\rightarrow$ 2DFA)

Optimal Simulation Between Unary Automata

The costs of the optimal simulations between automata are different in the unary and in the general case

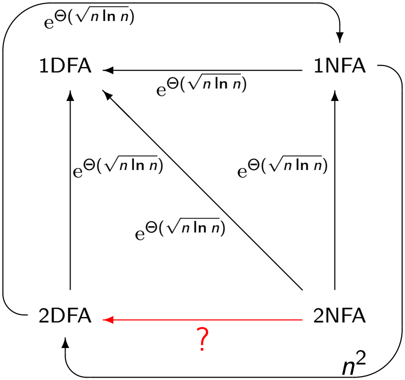


2NFA → 2DFA
Even in the unary case this question is open!

- $e^{\Theta(\sqrt{n \ln n})}$ upper bound (from 2NFA → 1DFA)
- $\Omega(n^2)$ lower bound (from 1NFA → 2DFA)

Optimal Simulation Between Unary Automata

The costs of the optimal simulations between automata are different in the unary and in the general case



2NFA → 2DFA
Even in the unary case this question is open!

- $e^{\Theta(\sqrt{n \ln n})}$ upper bound
  (from 2NFA → 1DFA)
- $\Omega(n^2)$ lower bound
  (from 1NFA → 2DFA)

A better upper bound $e^{O(\ln^2 n)}$ has been proved!

A Normal Form for Unary 2NFAs

Theorem ([Geffert Mereghetti&P '03])

For each unary n-state 2NFA A there exists an almost equivalent 2NFA M s.t.

- *M makes nondeterministic choices and changes the head direction only visiting the end-markers*
- *M has $N \leq 2n+2$ many states*
- *$L(A)$ and $L(M)$ can differ only on strings of length $\leq 5n^2$*

Normal Form for Unary 2NFAs: Consequences

(i) Subexponential simulation of unary 2NFAs by 2DFAs
Each unary n-state 2NFA can be simulated by a 2DFA with $e^{O(\ln^2 n)}$ states [Geffert Mereghetti&P '03]

(ii) Polynomial complementation of unary 2NFAs
Inductive counting argument [Geffert Mereghetti&P '07]

(iii) Polynomial simulation of unary 2NFAs by 2DFAs *under the condition* L = NL [Geffert&P '11]
Hence, proving that the upper bound $e^{O(\ln^2 n)}$ is tight would separate L and NL in the *general case*

(iv) Polynomial simulation of unary 2NFAs by unambiguous 2NFAs *(unconditional)* [Geffert&P '11]

Normal Form for Unary 2NFAs: Consequences

(i) Subexponential simulation of unary 2NFAs by 2DFAs
Each unary n-state 2NFA can be simulated by a 2DFA with $e^{O(\ln^2 n)}$ states [Geffert Mereghetti&P '03]

(ii) Polynomial complementation of unary 2NFAs
Inductive counting argument [Geffert Mereghetti&P '07]

(iii) Polynomial simulation of unary 2NFAs by 2DFAs *under the condition* $\mathrm{L} = \mathrm{NL}$ [Geffert&P '11]
Hence, proving that the upper bound $e^{O(\ln^2 n)}$ is tight would separate L and NL in the *general case*

(iv) Polynomial simulation of unary 2NFAs by unambiguous 2NFAs *(unconditional)* [Geffert&P '11]

Normal Form for Unary 2NFAs: Consequences

(i) Subexponential simulation of unary 2NFAs by 2DFAs
Each unary n-state 2NFA can be simulated by a 2DFA with $e^{O(\ln^2 n)}$ states [Geffert Mereghetti&P '03]

(ii) Polynomial complementation of unary 2NFAs
Inductive counting argument [Geffert Mereghetti&P '07]

(iii) Polynomial simulation of unary 2NFAs by 2DFAs *under the condition* L = NL [Geffert&P '11]
Hence, proving that the upper bound $e^{O(\ln^2 n)}$ is tight would separate L and NL in the *general case*

(iv) Polynomial simulation of unary 2NFAs by unambiguous 2NFAs *(unconditional)* [Geffert&P '11]

Normal Form for Unary 2NFAs: Consequences

(i) Subexponential simulation of unary 2NFAs by 2DFAs
Each unary n-state 2NFA can be simulated by a 2DFA with $e^{O(\ln^2 n)}$ states [Geffert Mereghetti&P '03]

(ii) Polynomial complementation of unary 2NFAs
Inductive counting argument [Geffert Mereghetti&P '07]

(iii) Polynomial simulation of unary 2NFAs by 2DFAs *under the condition* L = NL [Geffert&P '11]
Hence, proving that the upper bound $e^{O(\ln^2 n)}$ is tight would separate L and NL in the *general case*

(iv) Polynomial simulation of unary 2NFAs by unambiguous 2NFAs *(unconditional)* [Geffert&P '11]

Restricted 2NFAs

Restricted 2NFAs

Outer Nondeterministic Automata (OFAs) [Guillon Geffert&P '12]:

- *nondeterministic choices*

are *possible only* when the head is visiting the *endmarkers*

Restricted 2NFAs

Outer Nondeterministic Automata (OFAs) [Guillon Geffert&P '12]:

- *nondeterministic choices*

are *possible only* when the head is visiting the *endmarkers*

Hence:

- No restrictions on the *input alphabet*
- No restrictions on *head reversals*
- *Deterministic transitions* on "real" input symbols

Outer Nondeterministic Automata (OFAs)

The results we obtained for the unary case
can be extended to 2OFAs: [Guillon Geffert&P '12]

(i) Subexponential simulation of 2OFAs by 2DFAs
(ii) Polynomial complementation of 2OFAs
(iii) Polynomial simulation of 2OFAs by 2DFAs
under the condition L = NL
(iv) Polynomial simulation of 2OFAs by unambiguous 2OFAs

While in the unary case all the proofs rely
on the *normal form*,
for 2OFAs we do not have a similar tool!

Outer Nondeterministic Automata (OFAs)

The results we obtained for the unary case
can be extended to 2OFAs: [Guillon Geffert&P '12]

(i) Subexponential simulation of 2OFAs by 2DFAs
(ii) Polynomial complementation of 2OFAs
(iii) Polynomial simulation of 2OFAs by 2DFAs
under the condition L = NL
(iv) Polynomial simulation of 2OFAs by unambiguous 2OFAs

While in the unary case all the proofs rely
on the *normal form*,
for 2OFAs we do not have a similar tool!

Outer Nondeterministic Automata (OFAs)

The results we obtained for the unary case can be extended to 2OFAs: [Guillon Geffert&P '12]

(i) Subexponential simulation of 2OFAs by 2DFAs

(ii) Polynomial complementation of 2OFAs

(iii) Polynomial simulation of 2OFAs by 2DFAs *under the condition* L = NL

(iv) Polynomial simulation of 2OFAs by unambiguous 2OFAs

While in the unary case all the proofs rely
on the *normal form*,
for 2OFAs we do not have a similar tool!

Sakoda&Sipser Question: Current Knowledge

- Upper bounds

	1NFA→2DFA	2NFA→2DFA
unary case and OFAs	$O(n^2)$ optimal	$e^{O(\ln^2 n)}$
general case	exponential	exponential

Unary case [Chrobak '86, Geffert Mereghetti&P '03]
OFAs [Guillon Geffert&P '12]

- Lower Bounds
  In all the cases, the best known lower bound is $\Omega(n^2)$ [Chrobak '86]

Conclusion

Final Remarks

- The question of Sakoda and Sipser is very challenging
- In the investigation of restricted versions many interesting and not artificial models have been considered
- The results obtained under restrictions,
  even if not solving the full problem,
  are not trivial and, in many cases, very deep
- Connections with space and structural complexity
  - questions
  - techniques
- Connections with number theory (unary automata)

Final Remarks

- The question of Sakoda and Sipser is very challenging
- In the investigation of restricted versions many interesting and not artificial models have been considered
- The results obtained under restrictions, even if not solving the full problem, are not trivial and, in many cases, very deep
- Connections with space and structural complexity
  - questions
  - techniques
- Connections with number theory (unary automata)

Final Remarks

- The question of Sakoda and Sipser is very challenging
- In the investigation of restricted versions many interesting and not artificial models have been considered
- The results obtained under restrictions,
  even if not solving the full problem,
  are not trivial and, in many cases, very deep
- Connections with space and structural complexity
  - questions
  - techniques
- Connections with number theory (unary automata)

Final Remarks

- The question of Sakoda and Sipser is very challenging
- In the investigation of restricted versions many interesting and not artificial models have been considered
- The results obtained under restrictions,
  even if not solving the full problem,
  are not trivial and, in many cases, very deep
- Connections with space and structural complexity
  - questions
  - techniques
- Connections with number theory (unary automata)

Final Remarks

- The question of Sakoda and Sipser is very challenging
- In the investigation of restricted versions many interesting and not artificial models have been considered
- The results obtained under restrictions,
  even if not solving the full problem,
  are not trivial and, in many cases, very deep
- Connections with space and structural complexity
  - questions
  - techniques
- Connections with number theory (unary automata)

Thank you for your attention!