Two-way Automata and Related Models: Power and Complexity

Giovanni Pighizzini

Dipartimento di Informatica
Università degli Studi di Milano, Italy

La Primavera dell'Informatica Teorica
Italian Chapter of the EATCS
June 23, 2022

UNIVERSITÀ DEGLI STUDI
DI MILANO

Introduction

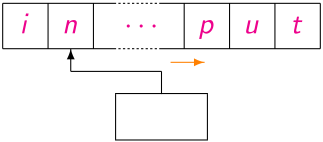
Descriptional Complexity

Investigation of formal models with respect to the sizes of their descriptions
(roughly: the number of symbols used to write down the description)

- *Relationships between the sizes* of the representations of a same class of objects (e.g., languages) by different formal systems (e.g., recognizers, grammars,...).
- Size costs of simulations
- ...

Two-Way Automata
and
Descriptional Complexity

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!

Ex. $n = 4$

a	b	b	a	b	a

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	b	b	a	b	a

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 nth symbol from the right!

Ex. $n = 4$

a	b	a	a	b	a

A Preliminary Example

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

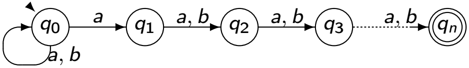
$$l_n = (a+b)^* a (a+b)^{n-1}$$

Check the nth symbol from the right!

Ex. $n = 4$

a	b	a	a	b	a

Nondeterminism!



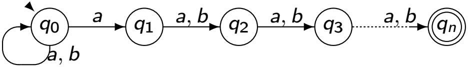
A Preliminary Example

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

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

Check the *n*th symbol from the right!

1NFA: $n+1$ states



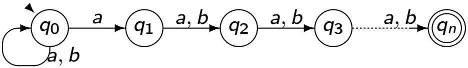
A Preliminary Example

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

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

Check the nth symbol from the right!

1NFA: $n + 1$ states



Miminal 1DFA: 2^n states!

Remember the last factor on length n
states $\equiv$ strings of length n over $\{a, b\}$

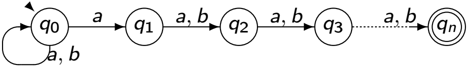
A Preliminary Example

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

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

Check the nth symbol from the right!

1NFA: $n+1$ states

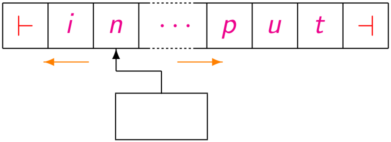


Miminal 1DFA: 2^n states!

Remember the last factor on length n
states $\equiv$ strings of length n over $\{a, b\}$

If we allow a DFA to reverse the head direction,
then $n + \ldots$ states are sufficient!

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
- *Deterministic* (2DFA) and *nondeterministic* (2NFA) versions
- Infinite computations are possible

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

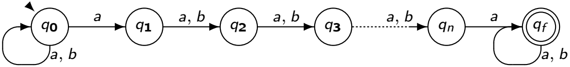
What about the power of these models?

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

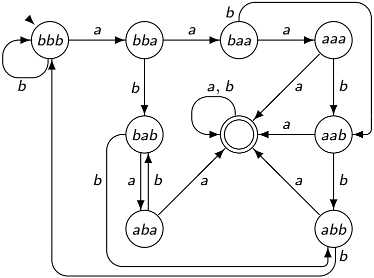
...some of them are more succinct

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

1NFA: $n+2$ states



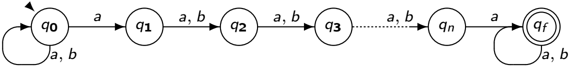
Minimum 1DFA: $2^n + 1$ states



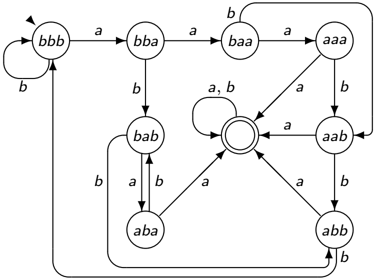
$n=3$

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

1NFA: $n + 2$ states



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



$n{=}3$

How many states on 2DFAs ?

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

A technique for 2DFA:

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

$n = 4$

- Move the head from left to right up to cell containing a
- Move n positions to the right
- If the symbol is a then accept
  else move $n - 1$ positions to the left and
  continue from the beginning

2DFA: $2n + \ldots$ states

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

A different technique for 2DFA:

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

$n = 4$

- Check positions k s.t. $k \equiv 1 \pmod{n}$
- Check positions k s.t. $k \equiv 2 \pmod{n}$

  ...
- Check positions k s.t. $k \equiv n \pmod{n}$

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

A different technique for 2DFA:

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

$n = 4$

- Check positions k s.t. $k \equiv 1 \pmod{n}$
- Check positions k s.t. $k \equiv 2 \pmod{n}$

  $\dots$
- Check positions k s.t. $k \equiv n \pmod{n}$

Even this strategy can be implemented using $O(n)$ states!

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

A different technique for 2DFA:

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

$n = 4$

- Check positions k s.t. $k \equiv 1 \pmod{n}$
- Check positions k s.t. $k \equiv 2 \pmod{n}$

  $\ldots$
- Check positions k s.t. $k \equiv n \pmod{n}$

Even this strategy can be implemented using $O(n)$ states!

Sweeping automata:

- Deterministic transitions
- Head reversals *only at the endmarkers*

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
  - a sweeping automaton

  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,

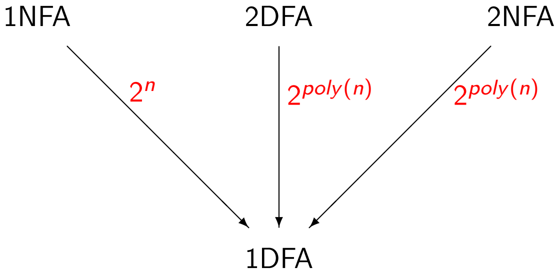
- L_n is accepted by
  - a 1NFA
  - a 2DFA
  - a sweeping automaton

  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

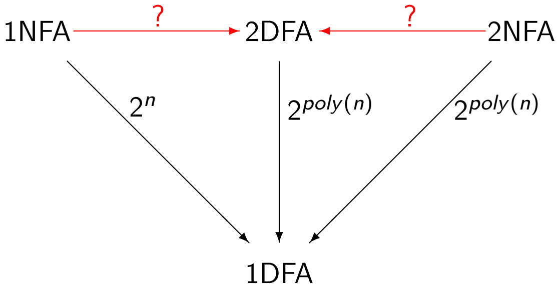
Is it always possible
to replace nondeterminism by two-way motion
without increasing too much the size?

Costs of the Optimal Simulations Between Automata



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

Costs of the Optimal Simulations Between Automata

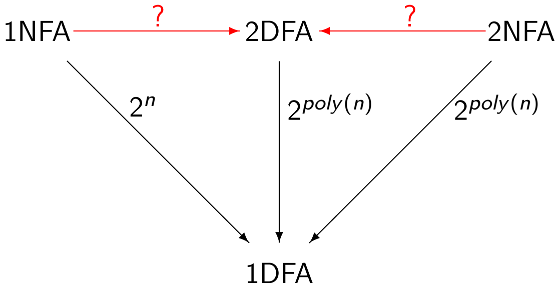


Problem ([Sakoda&Sipser '78])

Do there exist polynomial simulations of

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

Costs of the Optimal Simulations Between Automata



Problem ([Sakoda&Sipser '78])

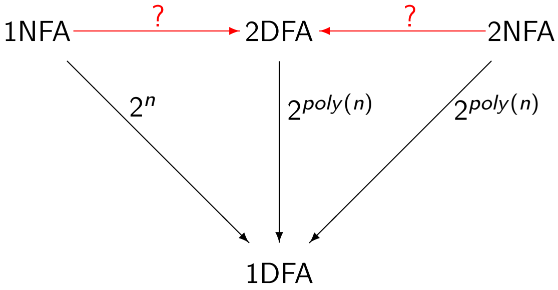
Do there exist polynomial simulations of

- *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]

$L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$ Again!

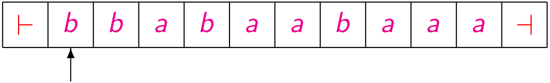
Naïf algorithm: compare input positions i and $i+n$, $i = 1, 2, \ldots$

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

$n = 4$

$L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$ Again!

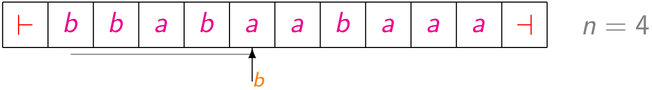
Naïf algorithm: compare input positions i and $i+n$, $i = 1, 2, \ldots$



$n = 4$

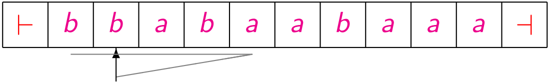
$L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$ Again!

Naïf algorithm: compare input positions i and $i+n$, $i = 1, 2, \ldots$



$L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$ Again!

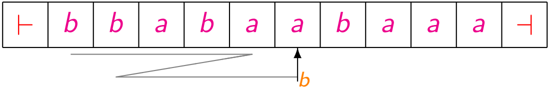
Naïf algorithm: compare input positions i and $i+n$, $i = 1, 2, \ldots$



$n = 4$

$L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$ Again!

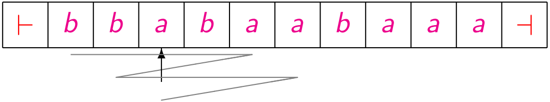
Naïf algorithm: compare input positions i and $i+n$, $i = 1, 2, \ldots$



$n = 4$

$L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$ Again!

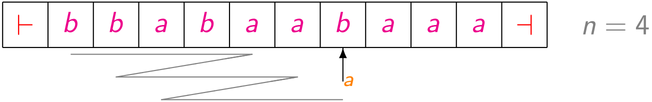
Naïf algorithm: compare input positions i and $i+n$, $i = 1, 2, \ldots$



$n = 4$

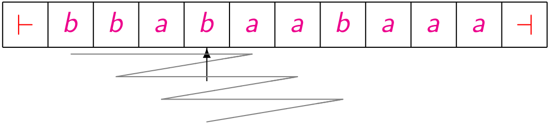
$L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$ Again!

Naïf algorithm: compare input positions i and $i+n$, $i = 1, 2, \ldots$



$L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$ Again!

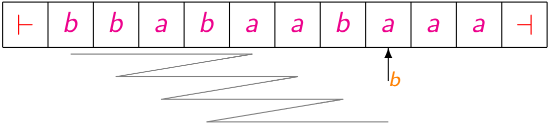
Naïf algorithm: compare input positions i and $i+n$, $i = 1, 2, \ldots$



$n = 4$

$L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$ Again!

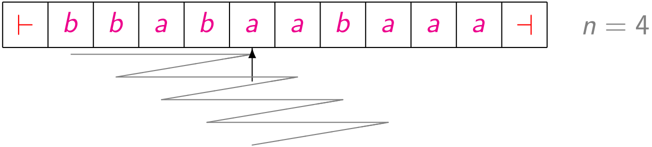
Naïf algorithm: compare input positions i and $i+n$, $i = 1, 2, \ldots$



$n = 4$

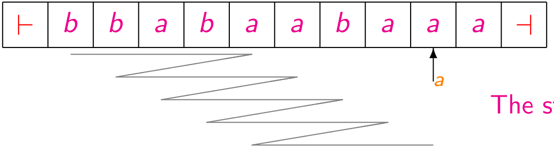
$L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$ Again!

Naïf algorithm: compare input positions i and $i+n$, $i = 1, 2, \ldots$



$L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$ Again!

Naïf algorithm: compare input positions i and $i+n$, $i = 1, 2, \ldots$

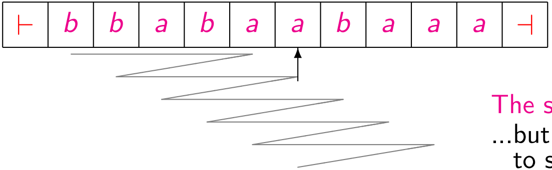


$n = 4$

The string can be accepted!

$L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$ Again!

Naïf algorithm: compare input positions i and $i+n$, $i = 1, 2, \ldots$



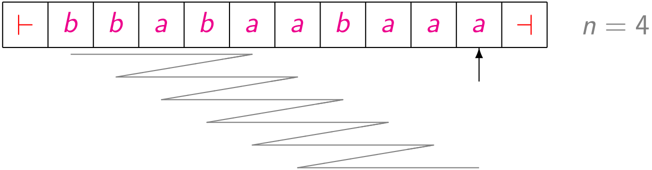
$n = 4$

The string can be accepted!

...but our automaton continues to scan

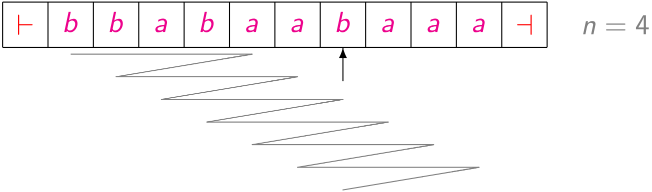
$L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$ Again!

Naïf algorithm: compare input positions i and $i+n$, $i = 1, 2, \ldots$



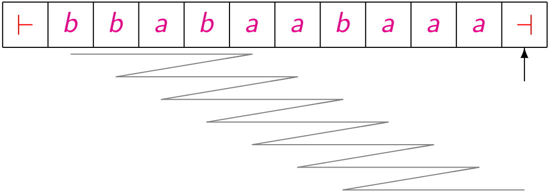
$L_n = (a+b)^*a(a+b)^{n-1}a(a+b)^*$ Again!

Naïf algorithm: compare input positions i and $i+n$, $i = 1, 2, \ldots$



$L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$ Again!

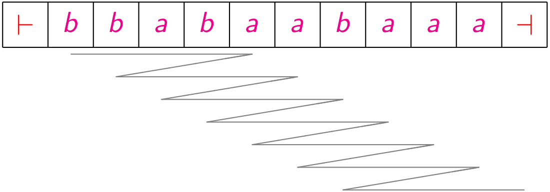
Naïf algorithm: compare input positions i and $i+n$, $i = 1, 2, \ldots$



$n = 4$

$L_n = (a+b)^*a(a+b)^{n-1}a(a+b)^*$ Again!

Naïf algorithm: compare input positions i and $i+n$, $i = 1, 2, \ldots$

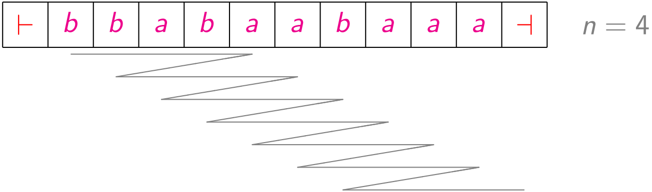


$n = 4$

Even in this case $O(n)$ states!

$L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$ Again!

Naïf algorithm: compare input positions i and $i+n$, $i = 1, 2, \ldots$



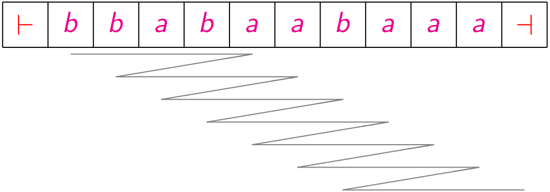
Even in this case $O(n)$ states!

Oblivious Automata:

- Deterministic transitions
- Same "trajectory" on all inputs of the same length

$L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$ Again!

Naïf algorithm: compare input positions i and $i+n$, $i = 1, 2, \ldots$



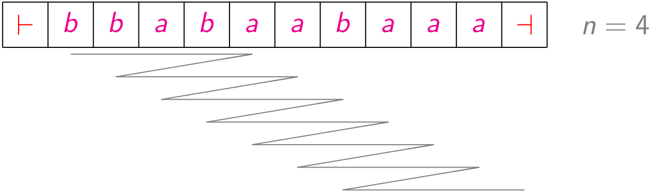
$n = 4$

Number of head reversals:
On input of length m:

- This technique uses about $2m$ reversals,
  a *linear number* in the input length
- The "sweeping" algorithm uses about $2n$ reversals,
  a *constant number* in the input length

$L_n = (a+b)^* a (a+b)^{n-1} a (a+b)^*$ Again!

Naïf algorithm: compare input positions i and $i+n$, $i = 1, 2, \dots$



Number of head reversals:
On input of length m:

- This technique uses about $2m$ reversals,
  a *linear number* in the input length
- The "sweeping" algorithm uses about $2n$ reversals,
  a *constant number* in the input length

Another Restricted Model

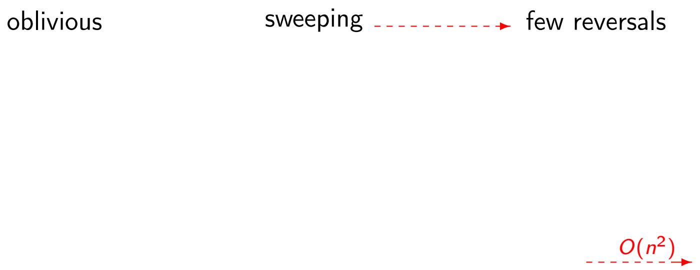
"Few Reversal" Automata [Kapoutsis '11]:

- On input of length m the number of reversals is $o(m)$, i.e., sublinear

Restricted Models: Separations

oblivious sweeping few reversals

Restricted Models: Separations

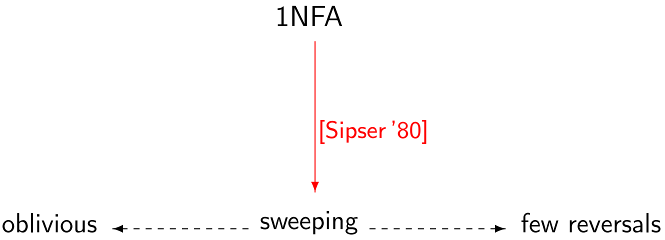


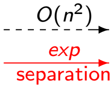
Restricted Models: Separations

oblivious ← - - - - - sweeping - - - - - → few reversals

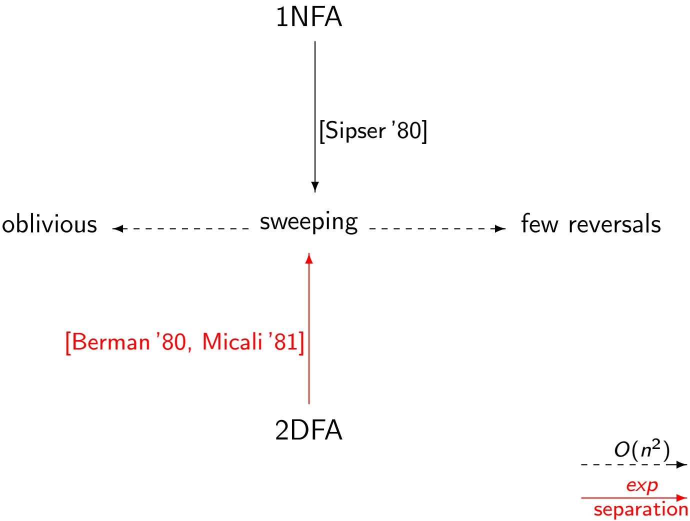
$O(n^2)$

Restricted Models: Separations

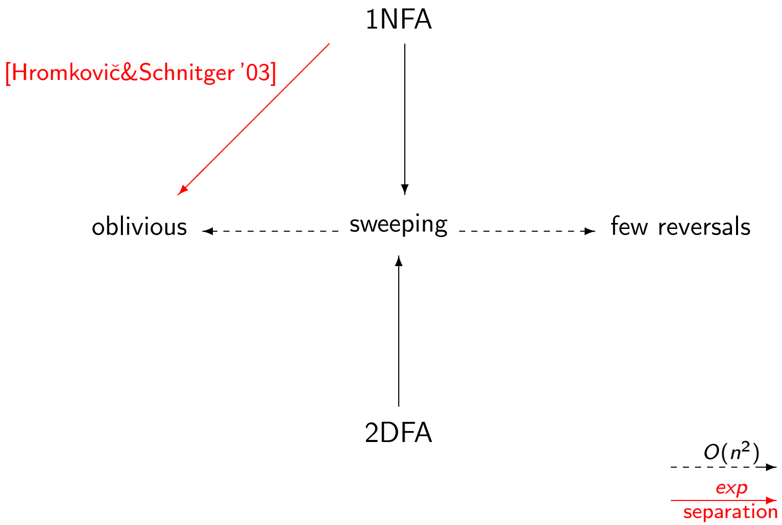




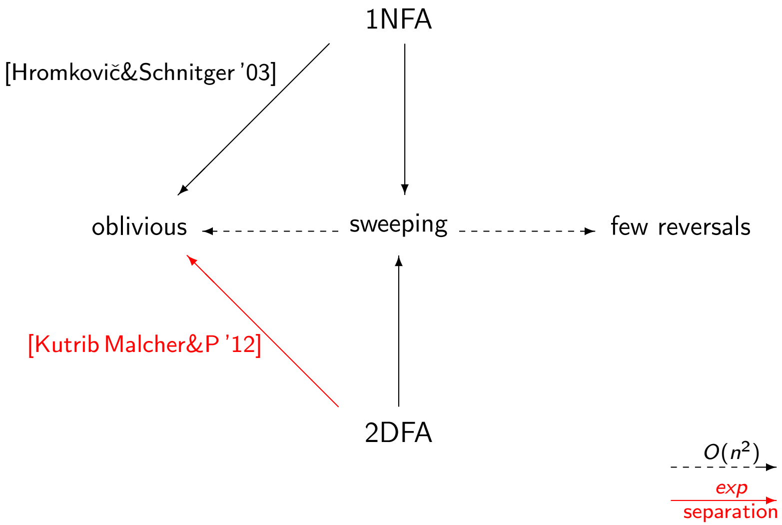
Restricted Models: Separations



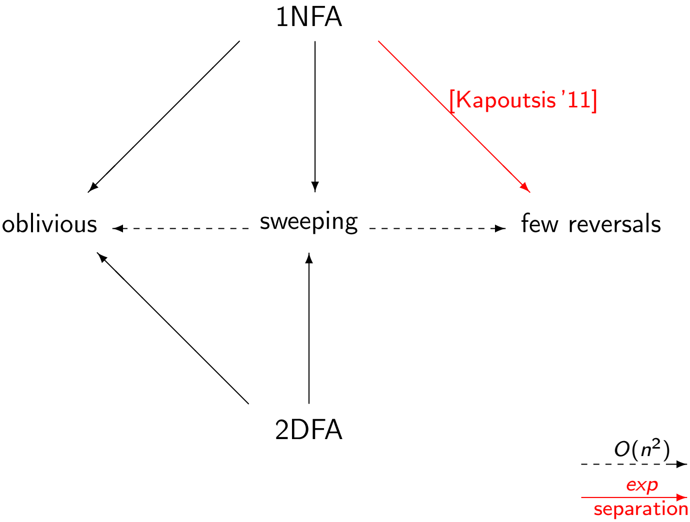
Restricted Models: Separations



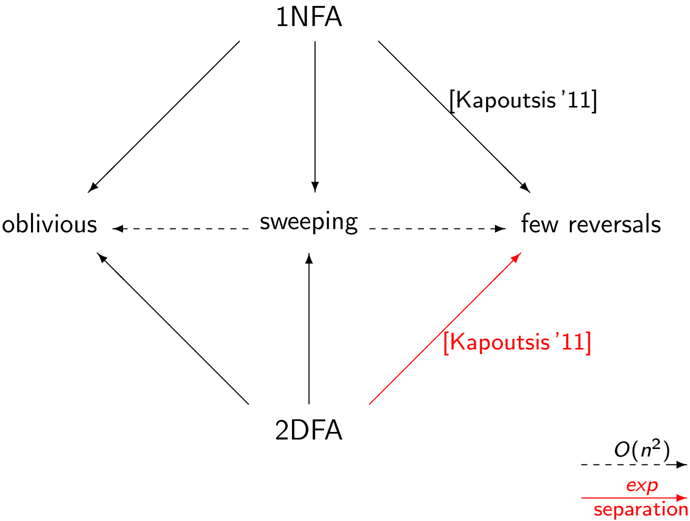
Restricted Models: Separations



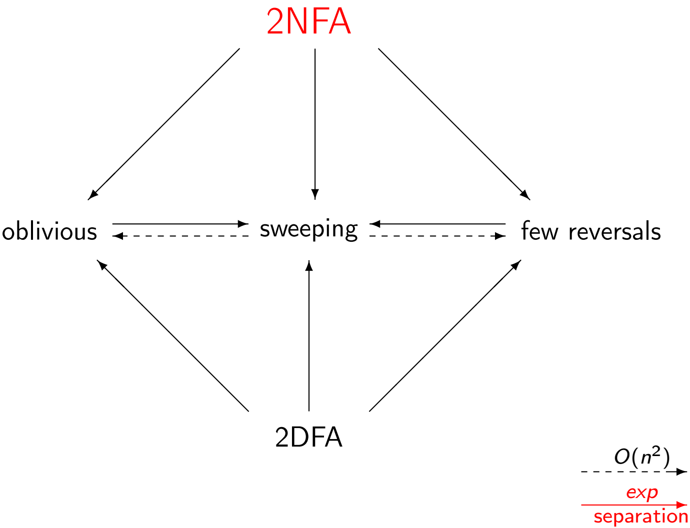
Restricted Models: Separations



Restricted Models: Separations



Restricted Models: Separations



Sakoda&Sipser Question

Problem ([Sakoda&Sipser '78])

Do there exist polynomial simulations of

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

Another possible restriction:

The unary case $\#\Sigma = 1$

A Normal Form for Unary 2NFAs

Theorem ([Geffert&Mereghetti&P.'03])

Each n-state unary 2NFA A can be transformed into a 2NFA M s.t.:

- *nondeterministic choices and head reversals are possibile only at the end-markers*
- *M has at most $2n + 2$ states*
- *M and A agrees on all inputs of length $> 5n^2$*

Normal Form for Unary 2NFAs: Some 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 simulation of unary 2NFAs by 2DFAs *under the condition* L = NL [Geffert&P.'11]

Normal Form for Unary 2NFAs: Some 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 simulation of unary 2NFAs by 2DFAs
under the condition L = NL [Geffert&P.'11]

Restricted 2NFAs

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

- *nondeterministic choices* are *possible only* when the head is visiting the *endmarkers*

Restricted 2NFAs

Outer Nondeterministic Automata [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

Restricted 2NFAs

Outer Nondeterministic Automata [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

Extensions of the results obtained for unary 2NFAs, in particular:

Subexponential simulation of outer NFAs by 2DFAs

Variants of the NFAs vs 2DFAs Question

(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]

Variants of the NFAs vs 2DFAs Question

(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]

(iv) Enlarge the family of simulating machines

- Hennie machines

[Guillon&P.&Prigioniero&Průša'18]

Variants of the NFAs vs 2DFAs Question

(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]

(iv) Enlarge the family of simulating machines

- Hennie machines

[Guillon&P.&Prigioniero&Průša'18]

Hennie Machines

One-tape *deterministic* Turing machines working in *linear time*

(extensions of 2DFAs)

Theorem ([Hennie '65])

Each language accepted by a Hennie machine is regular

Hennie Machines

One-tape *deterministic* Turing machines working in *linear time*

(extensions of 2DFAs)

Theorem ([Hennie '65])

Each language accepted by a Hennie machine is regular

Theorem ([Guillon&P.&Prigioniero&Průša'18])

Each n-state 2NFA can be simulated by a Hennie machine of size polynomial in n

Hennie Machines

One-tape *deterministic* Turing machines working in *linear time*

(extensions of 2DFAs)

Theorem ([Hennie '65])

Each language accepted by a Hennie machine is regular

Theorem ([Guillon&P.&Prigioniero&Průša'18])

Each n-state 2NFA *can be simulated by a Hennie machine of size polynomial in n*

Find a family of devices "between" 2DFAs and Hennie machines that can simulate 2NFAs using polynomial size

Limited Automata

Limited automata

- Model proposed by Hibbard in 1967
  (scan limited automata)
- One-tape Turing machines with rewriting restrictions
- Variants characterizing regular, context-free, deterministic context-free languages

A Classical Example: Balanced Brackets

([ ] [( )])

How to recognize if a sequence of brackets is correctly balanced?

A Classical Example: Balanced Brackets

([ ] [( )])

How to recognize if a sequence of brackets is correctly balanced?

- *For each opening bracket*
  locate its corresponding closing bracket

Use counters!

A Classical Example: Balanced Brackets

([ ] [( )])

How to recognize if a sequence of brackets is correctly balanced?

- *For each opening bracket*
  locate its corresponding closing bracket

Use counters!

- *For each closing bracket*
  locate its corresponding opening bracket

Limited automata!

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 overwrite 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 overwrite 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]

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 overwrite 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]

Descriptional Complexity of 1-Limited Automata

The Language B_n ($n > 0$)

$$B_n = \{x_1\, x_2 \cdots x_k\, x \in \{0,1\}^* \quad | \quad |x_1| = \cdots = |x_k| = |x| = n,\ k > 0,$$
$$\text{and } x_j = x,\ \text{for some } 1 \leq j \leq k\,\}$$

The Language B_n ($n > 0$)

$$B_n = \{x_1\, x_2 \cdots x_k\, x \in \{0,1\}^* \quad | \quad |x_1| = \cdots = |x_k| = |x| = n,\ k > 0, \text{ and } x_j = x,\ \text{for some } 1 \le j \le k\,\}$$

Example ($n = 3$):

0 0 1 0 1 0 1 1 0 0 1 0 1 0 0 1 1 1 1 0

The Language B_n $(n > 0)$

$$B_n = \{x_1\,x_2 \cdots x_k\,x \in \{0,1\}^* \quad | \quad |x_1| = \cdots = |x_k| = |x| = n,\ \ k > 0, \\ \text{and } x_j = x,\ \text{ for some } 1 \leq j \leq k\,\}$$

Example $(n = 3)$:

0 0 1|0 1 0|1 1 0|0 1 0|1 0 0|1 1 1|1 1 0

A Nondeterministic 1-Limited Automaton for B_n

$\triangleright$ 0 0 1 0 1 0 1 1 0 0 1 0 1 0 0 1 1 1 1 1 0 $\triangleleft$ $(n = 3)$

1. Scan all the tape from left to right
   and mark two nondeterministically chosen cells
2. Check that:
   - the input length is a multiple of n,
   - the last marked cell is the leftmost one of the last block, and
   - the other marked cell is the leftmost one of another block
3. Compare symbol by symbol the two blocks that start from the marked cells and accept if they are equal

A Nondeterministic 1-Limited Automaton for B_n

$\triangleright\ 0\ 0\ 1\ 0\ 1\ 0\ \hat{1}\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 1\ \hat{1}\ 1\ 0\ \triangleleft$ $(n = 3)$

1. Scan all the tape from left to right
   and mark two nondeterministically chosen cells
2. Check that:
   - the input length is a multiple of n,
   - the last marked cell is the leftmost one of the last block, and
   - the other marked cell is the leftmost one of another block
3. Compare symbol by symbol the two blocks that start from the marked cells and accept if they are equal

A Nondeterministic 1-Limited Automaton for B_n

$\triangleright$ 0 0 1 0 1 0 $\hat{1}$ 1 0 0 1 0 1 0 0 1 1 1 $\hat{1}$ 1 0 $\triangleleft$ $(n = 3)$

1. Scan all the tape from left to right
   and mark two nondeterministically chosen cells
2. Check that:
   - the input length is a multiple of n,
   - the last marked cell is the leftmost one of the last block, and
   - the other marked cell is the leftmost one of another block
3. Compare symbol by symbol the two blocks that start from the marked cells and accept if they are equal

A Nondeterministic 1-Limited Automaton for B_n

$\triangleright$ 0 0 1 0 1 0 $\hat{1}$ 1 0 0 1 0 1 0 0 1 1 1 $\hat{1}$ 1 0 $\triangleleft$ $(n = 3)$

1. Scan all the tape from left to right
   and mark two nondeterministically chosen cells
2. Check that:
   - the input length is a multiple of n,
   - the last marked cell is the leftmost one of the last block, and
   - the other marked cell is the leftmost one of another block
3. Compare symbol by symbol the two blocks that start from the marked cells and accept if they are equal

Complexity:

- $O(n)$ states
- Fixed working alphabet

  $\Rightarrow$ 1-LA of size $O(n)$

A Nondeterministic 1-Limited Automaton for B_n

$\triangleright\ 0\ 0\ 1\ 0\ 1\ 0\ \hat{1}\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 1\ \hat{1}\ 1\ 0\ \triangleleft$ $(n = 3)$

1. Scan all the tape from left to right and mark two nondeterministically chosen cells
2. Check that:
   - the input length is a multiple of n,
   - the last marked cell is the leftmost one of the last block, and
   - the other marked cell is the leftmost one of another block
3. Compare symbol by symbol the two blocks that start from the marked cells and accept if they are equal

Complexity:

- $O(n)$ states
- Fixed working alphabet

$\Rightarrow$ 1-LA of size $O(n)$

Finite automata

Each 1DFA accepting B_n needs a number of states *double exponential* in n

Size of Limited Automata vs Finite Automata

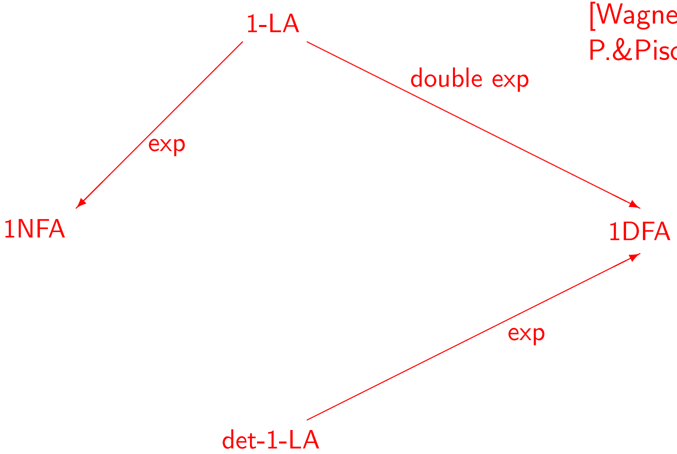
1-LA

1NFA

1DFA

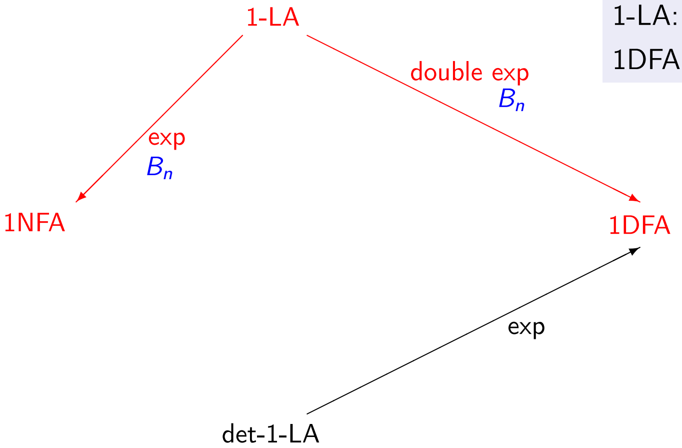
det-1-LA

Size of Limited Automata vs Finite Automata



Simulations
[Wagner&Wechsung '86, P.&Pisoni '14]

Size of Limited Automata vs Finite Automata

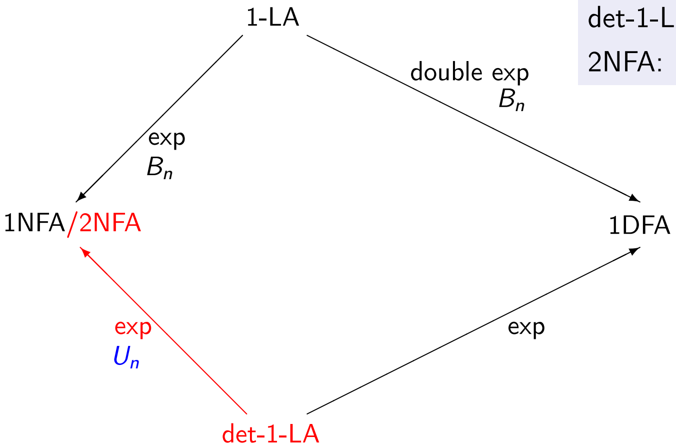


$B_n \subseteq \{0,1\}^*$

1-LA: size $O(n)$

1DFA: $\geq 2^{2^n}$ states

Size of Limited Automata vs Finite Automata

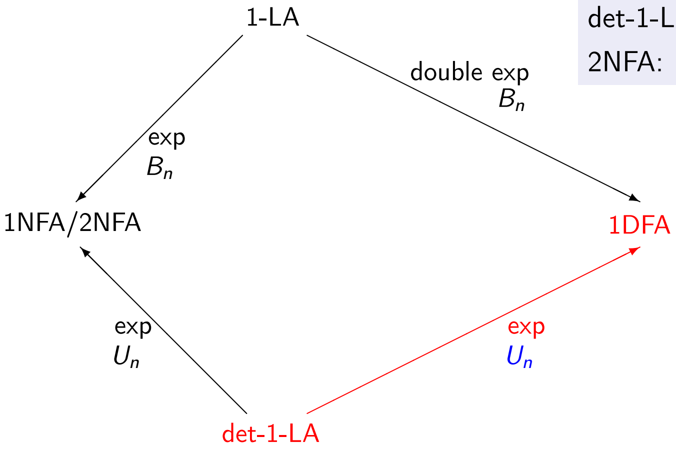


$U_n = \{a^{2^n}\}^*$

det-1-LA: size $O(n)$

2NFA: $\geq 2^n$ states

Size of Limited Automata vs Finite Automata

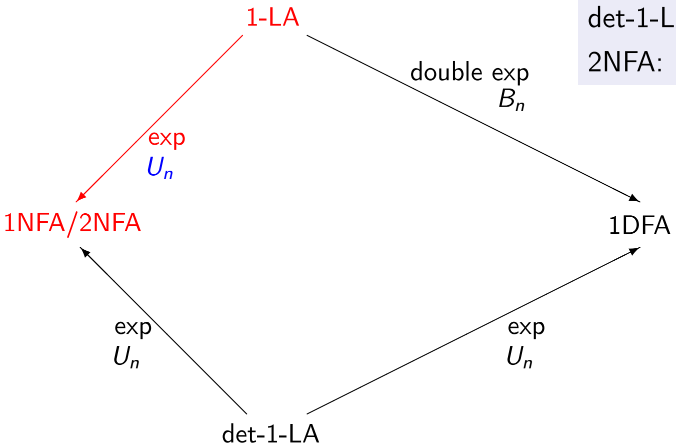


$U_n = \{a^{2^n}\}^*$

det-1-LA: size $O(n)$

2NFA: $\geq 2^n$ states

Size of Limited Automata vs Finite Automata

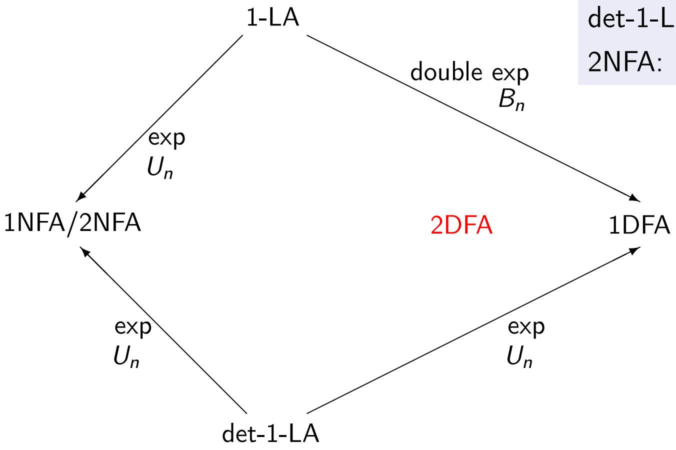


$U_n = \{a^{2^n}\}^*$

det-1-LA: size $O(n)$

2NFA: $\geq 2^n$ states

Size of Limited Automata vs Finite Automata

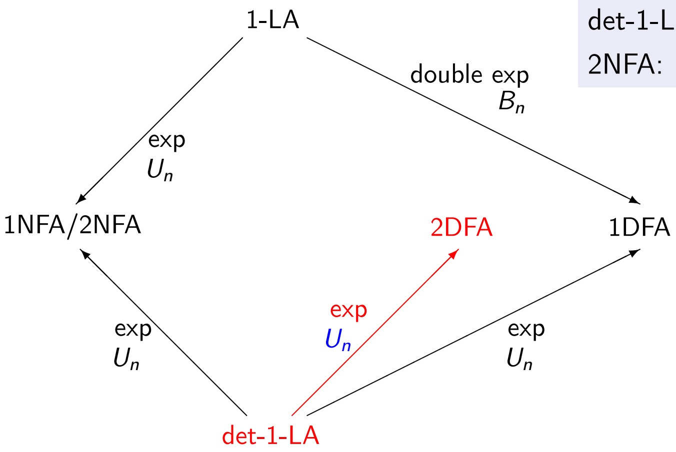


$U_n = \{a^{2^n}\}^*$

det-1-LA: size $O(n)$

2NFA: $\geq 2^n$ states

Size of Limited Automata vs Finite Automata



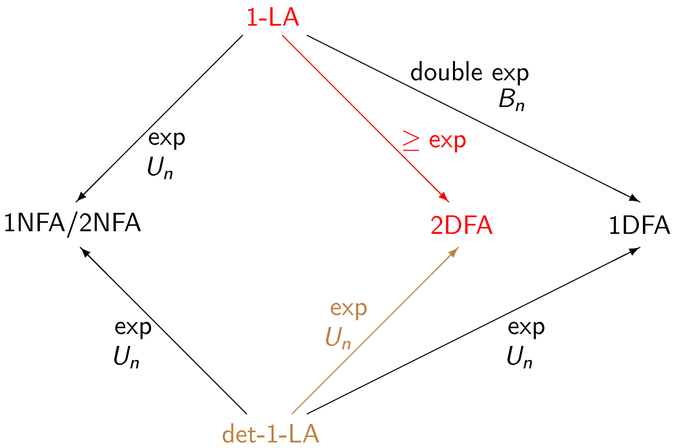
$U_n = \{a^{2^n}\}^*$

det-1-LA: size $O(n)$

2NFA: $\geq 2^n$ states

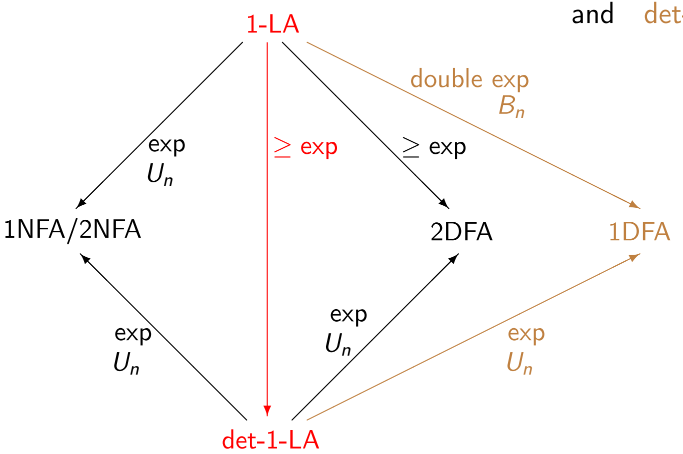
Size of Limited Automata vs Finite Automata

From det-1-LA→2DFA

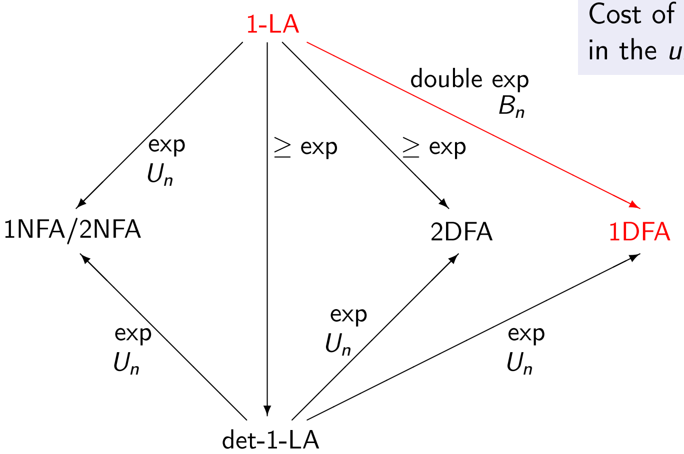


Size of Limited Automata vs Finite Automata

From 1-LA→1DFA
and det-1-LA→1DFA



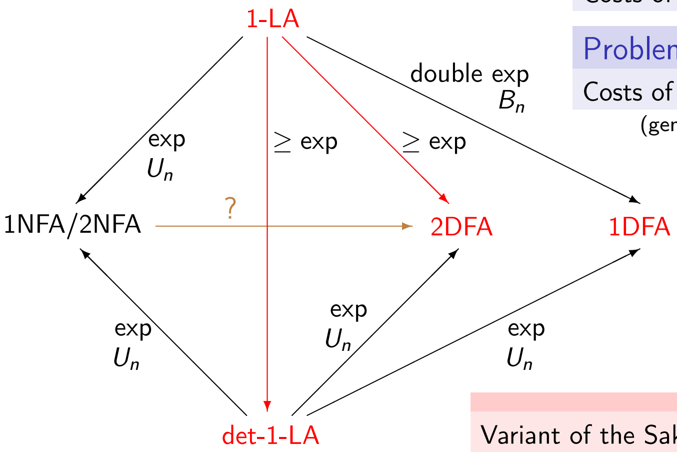
Size of Limited Automata vs Finite Automata



Problem 1

Cost of 1-LA → 1DFA in the *unary case*

Size of Limited Automata vs Finite Automata



Problem 2

Costs of 1-LA $\rightarrow$ det-1-LA

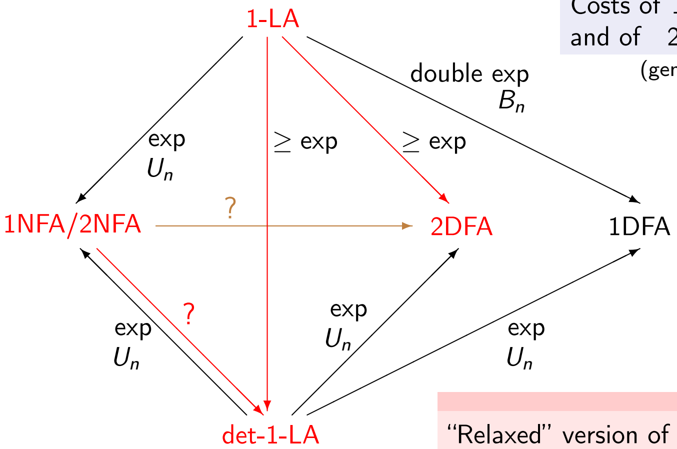
Problem 3

Costs of 1-LA $\rightarrow$ 2DFA

(general and unary case)

Variant of the Sakoda and Sipser question 1NFA/2NFA $\rightarrow$ 2DFA

Size of Limited Automata vs Finite Automata



Problem 4

Costs of 1NFA $\rightarrow$ det-1-LA
and of 2NFA $\rightarrow$ det-1-LA
(general and unary case)

"Relaxed" version of the Sakoda and Sipser question 1NFA/2NFA $\rightarrow$ 2DFA

Conclusion

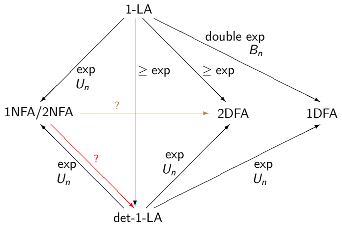
Final Remarks

- The question of Sakoda and Sipser is very challenging
- In the investigation of its variants, 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)

Possible lines of investigations

Find a family of devices "between" 2DFAs and Hennie machines that can simulate 2NFAs using polynomial size

- What is the cost of the simulation on 2NFAs by *deterministic* 1-limited automata?
- Any connections between descriptional complexity questions on variants of 1-limited automata and the Sakoda and Sipser question?



Thank you for your attention!