history-deterministic automata

Nicolas Markey
CNRS – IRISA (Univ. Rennes, France)

based on joint discussions with Uli, Sophie, Dylan, Hervé

history-deterministic automata

Nicolas Markey
CNRS – IRISA (Univ. Rennes, France)

based on joint discussions with Uli, Sophie, Dylan, Hervé
and on numerous external references...

history-deterministic automata

Nicolas Markey
CNRS – IRISA (Univ. Rennes, France)

based on joint discussions with Uli, Sophie, Dylan, Hervé
and on numerous external references...

Nondeterminism in the Presence of a Diverse or Unknown Future*

Udi Boker[1], Denis Kuperberg[2], Orna Kupferman[2], and Michał Skrzypczak[3]

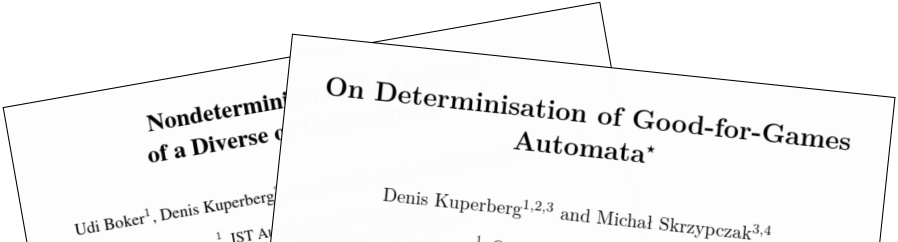
[1] IST Austria, Klosterneuburg, Austria
University, Jerusalem, Israel

history-deterministic automata

Nicolas Markey

CNRS – IRISA (Univ. Rennes, France)

based on joint discussions with Uli, Sophie, Dylan, Hervé
and on numerous external references...



history-deterministic automata

Nicolas Markey
CNRS – IRISA (Univ. Rennes, France)

based on joint discussions with Uli, Sophie, Dylan, Hervé
and on numerous external references...

history-deterministic automata

Nicolas Markey

CNRS – IRISA (Univ. Rennes, France)

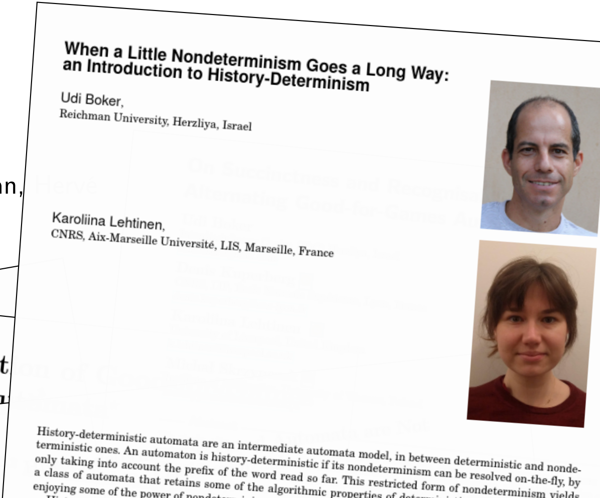
based on joint discussions with Uli, Sophie, Dylan, Hervé
and on numerous external references...

history-deterministic automata

Nicolas Markey

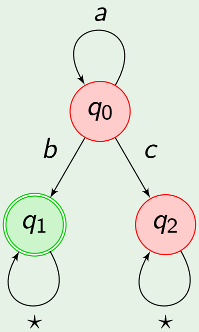
CNRS – IRISA (Univ. Rennes, France)

based on joint discussions with Uli, Sophie, Dylan,
and on numerous external references...



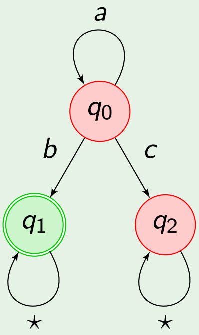
Automata on infinite words and trees

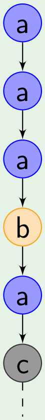
Example (Büchi word automata)



Automata on infinite words and trees

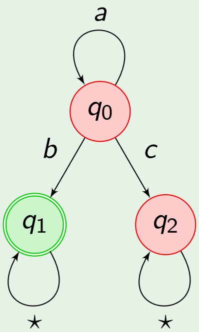
Example (Büchi word automata)

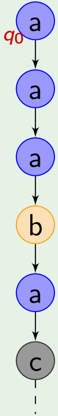




Automata on infinite words and trees

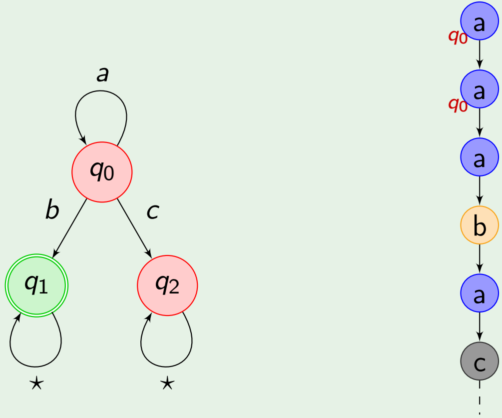
Example (Büchi word automata)





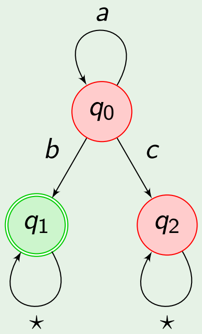
Automata on infinite words and trees

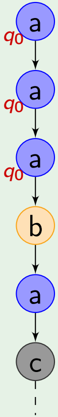
Example (Büchi word automata)



Automata on infinite words and trees

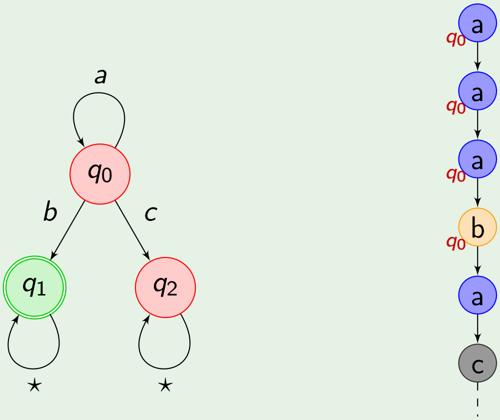
Example (Büchi word automata)





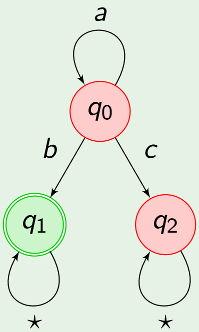
Automata on infinite words and trees

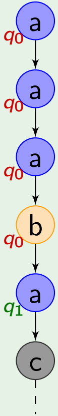
Example (Büchi word automata)



Automata on infinite words and trees

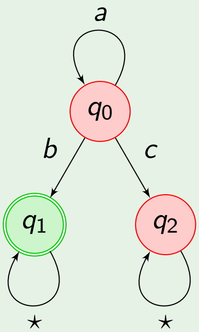
Example (Büchi word automata)

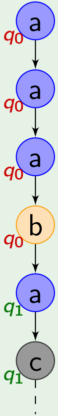




Automata on infinite words and trees

Example (Büchi word automata)





Automata on infinite words and trees

Example (Büchi tree automata)

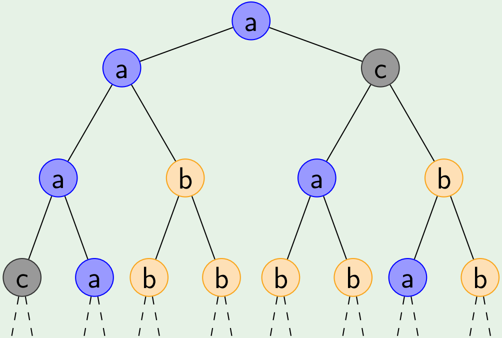
$\delta(q_0, a) = (q_0, q_1)$

$\delta(q_0, b) = (q_1, q_1)$

$\delta(q_0, c) = (q_2, q_2)$

$\delta(q_1, \star) = (q_1, q_1)$

$\delta(q_2, \star) = (q_2, q_2)$



Automata on infinite words and trees

Example (Büchi tree automata)

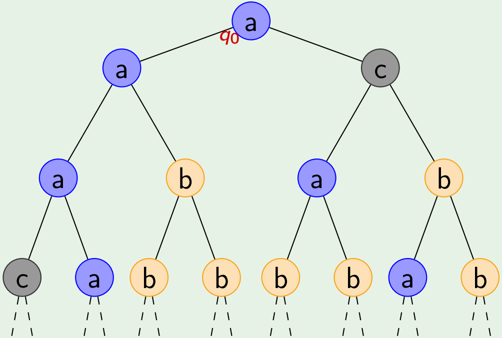
$\delta(q_0, a) = (q_0, q_1)$

$\delta(q_0, b) = (q_1, q_1)$

$\delta(q_0, c) = (q_2, q_2)$

$\delta(q_1, \star) = (q_1, q_1)$

$\delta(q_2, \star) = (q_2, q_2)$



Automata on infinite words and trees

Example (Büchi tree automata)

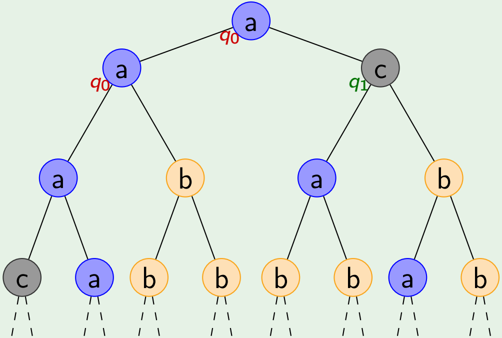
$\delta(q_0, a) = (q_0, q_1)$

$\delta(q_0, b) = (q_1, q_1)$

$\delta(q_0, c) = (q_2, q_2)$

$\delta(q_1, \star) = (q_1, q_1)$

$\delta(q_2, \star) = (q_2, q_2)$



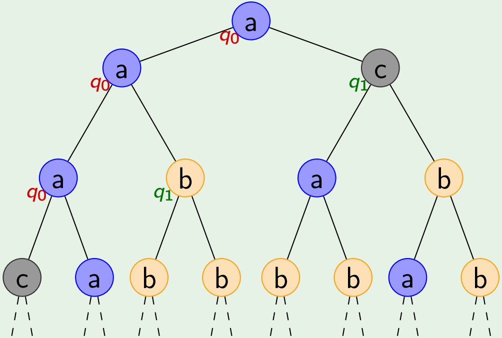
Automata on infinite words and trees

Example (Büchi tree automata)

$$\delta(q_0, a) = (q_0, q_1)$$
$$\delta(q_0, b) = (q_1, q_1)$$
$$\delta(q_0, c) = (q_2, q_2)$$

$$\delta(q_1, \star) = (q_1, q_1)$$

$$\delta(q_2, \star) = (q_2, q_2)$$



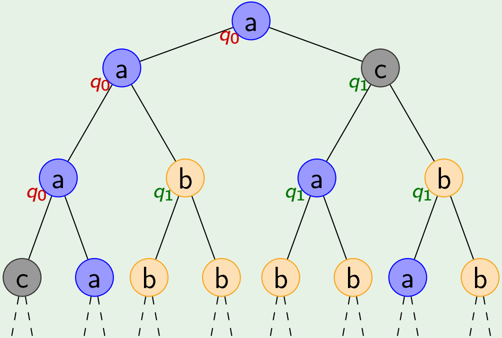
Automata on infinite words and trees

Example (Büchi tree automata)

$$\delta(q_0, a) = (q_0, q_1)$$
$$\delta(q_0, b) = (q_1, q_1)$$
$$\delta(q_0, c) = (q_2, q_2)$$

$$\delta(q_1, \star) = (q_1, q_1)$$

$$\delta(q_2, \star) = (q_2, q_2)$$



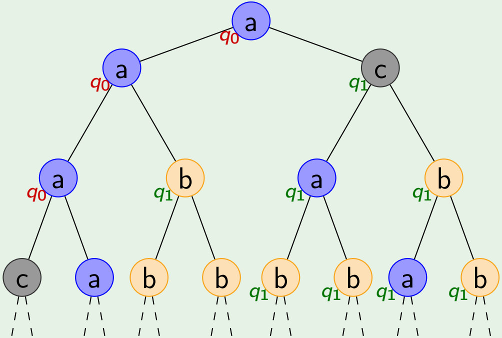
Automata on infinite words and trees

Example (Büchi tree automata)

$$\delta(q_0, a) = (q_0, q_1)$$
$$\delta(q_0, b) = (q_1, q_1)$$
$$\delta(q_0, c) = (q_2, q_2)$$

$$\delta(q_1, \star) = (q_1, q_1)$$

$$\delta(q_2, \star) = (q_2, q_2)$$



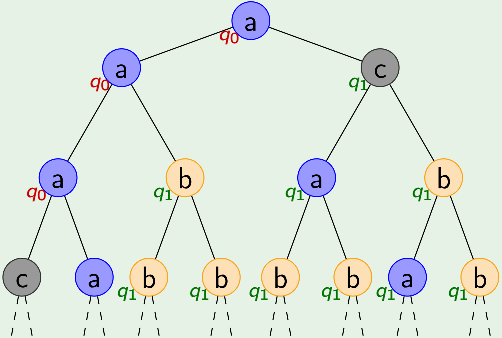
Automata on infinite words and trees

Example (Büchi tree automata)

$$\delta(q_0, a) = (q_0, q_1)$$
$$\delta(q_0, b) = (q_1, q_1)$$
$$\delta(q_0, c) = (q_2, q_2)$$

$$\delta(q_1, \star) = (q_1, q_1)$$

$$\delta(q_2, \star) = (q_2, q_2)$$



Automata on infinite words and trees

Example (Büchi tree automata)

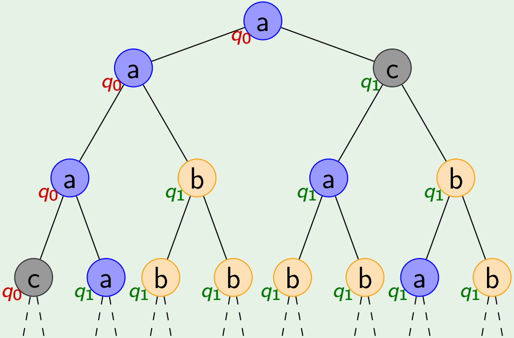
$\delta(q_0, a) = (q_0, q_1)$

$\delta(q_0, b) = (q_1, q_1)$

$\delta(q_0, c) = (q_2, q_2)$

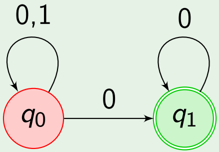
$\delta(q_1, \star) = (q_1, q_1)$

$\delta(q_2, \star) = (q_2, q_2)$



Deterministic vs. non-deterministic Büchi automata

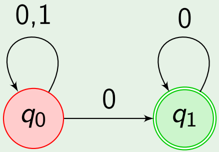
Example



accepts all infinite words
having finitely many 1s

Deterministic vs. non-deterministic Büchi automata

Example



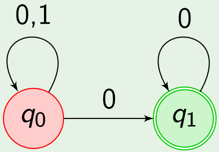
accepts all infinite words
having finitely many 1s

No deterministic Büchi automata accept this language

0 0 0 0 0 0 0 0 $\cdots$ is accepted

Deterministic vs. non-deterministic Büchi automata

Example



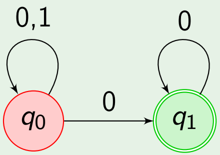
accepts all infinite words having finitely many 1s

No deterministic Büchi automata accept this language

0 0 0 0 0 0 0 0 $\cdots$ is accepted

Deterministic vs. non-deterministic Büchi automata

Example



accepts all infinite words having finitely many 1s

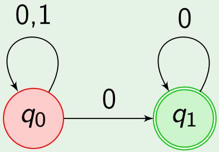
No deterministic Büchi automata accept this language

0 0 0 0 0 0 0 0 $\cdots$ is accepted

0 0 0 0 0 1 0 0 0 0 0 0 $\cdots$ is accepted

Deterministic vs. non-deterministic Büchi automata

Example



accepts all infinite words
having finitely many 1s

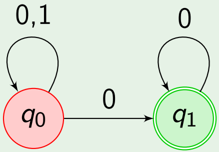
No deterministic Büchi automata accept this language

0 0 0 0 0 0 0 0 $\cdots$ is accepted

0 0 0 0 0 1 0 0 0 0 0 0 $\cdots$ is accepted

Deterministic vs. non-deterministic Büchi automata

Example



accepts all infinite words having finitely many 1s

No deterministic Büchi automata accept this language

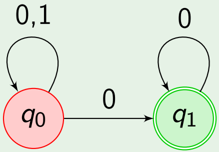
0 0 0 0 0 0 0 0 $\cdots$ is accepted

0 0 0 0 0 1 0 0 0 0 0 0 $\cdots$ is accepted

0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 $\cdots$ is accepted

Deterministic vs. non-deterministic Büchi automata

Example



accepts all infinite words having finitely many 1s

No deterministic Büchi automata accept this language

0 0 0 0 0 0 0 0 ⋯ is accepted

0 0 0 0 0 1 0 0 0 0 0 0 ⋯ is accepted

0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 ⋯ is accepted

From word to tree automata

Derived language [KSV06]

Given $L \subseteq \Sigma^\omega$, we let ΔL be the set of trees all of whose infinite branches are labelled with words in L.

[KSV06] Kupferman, Safra, Vardi. Relating word and tree automata. Annals Pure & Applied Logic, 138(1-3):126-146, 2006.
[BKKS13] Boker *et al.* Nondeterminism in the Presence of a Diverse or Unknown Future. In ICALP'13, p. 89-100. Springer, 2013.

From word to tree automata

Derived language [KSV06]

Given $L \subseteq \Sigma^\omega$, we let ΔL be the set of trees all of whose infinite branches are labelled with words in L.

If L is accepted by $\mathcal{B}$, is ΔL accepted by some tree automaton $\mathcal{C}$?

[KSV06] Kupferman, Safra, Vardi. Relating word and tree automata. Annals Pure & Applied Logic, 138(1-3):126-146, 2006.
[BKKS13] Boker *et al.* Nondeterminism in the Presence of a Diverse or Unknown Future. In ICALP'13, p. 89-100. Springer, 2013.

From word to tree automata

Derived language [KSV06]

Given $L \subseteq \Sigma^\omega$, we let ΔL be the set of trees all of whose infinite branches are labelled with words in L.

If L is accepted by $\mathcal{B}$, is ΔL accepted by some tree automaton $\mathcal{C}$?

Natural construction for deterministic automata

If $\delta_{\mathcal{B}}(q, \sigma) = q'$, we let $\delta_{\mathcal{C}}(q, \sigma) = (q', q')$. ✓

[KSV06] Kupferman, Safra, Vardi. Relating word and tree automata. Annals Pure & Applied Logic, 138(1-3):126-146, 2006.
[BKKS13] Boker *et al.* Nondeterminism in the Presence of a Diverse or Unknown Future. In ICALP'13, p. 89-100. Springer, 2013.

From word to tree automata

Derived language [KSV06]

Given $L \subseteq \Sigma^\omega$, we let ΔL be the set of trees all of whose infinite branches are labelled with words in L.

If L is accepted by $\mathcal{B}$, is ΔL accepted by some tree automaton $\mathcal{C}$?

Natural construction for non-deterministic automata

We let $\delta_{\mathcal{C}}(q, \sigma) = \{(q', q'') \mid q', q'' \in \delta_{\mathcal{B}}(q, \sigma)\}$. ✗

[KSV06] Kupferman, Safra, Vardi. Relating word and tree automata. Annals Pure & Applied Logic, 138(1-3):126-146, 2006.
[BKKS13] Boker *et al.* Nondeterminism in the Presence of a Diverse or Unknown Future. In ICALP'13, p. 89-100. Springer, 2013.

From word to tree automata

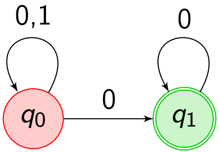
Derived language [KSV06]

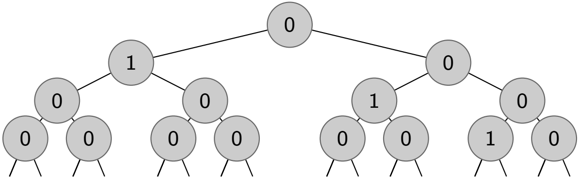
Given $L \subseteq \Sigma^\omega$, we let ΔL be the set of trees all of whose infinite branches are labelled with words in L.

If L is accepted by $\mathcal{B}$, is ΔL accepted by some tree automaton $\mathcal{C}$?

Natural construction for non-deterministic automata

We let $\delta_{\mathcal{C}}(q,\sigma) = \{(q',q'') \mid q',q'' \in \delta_{\mathcal{B}}(q,\sigma)\}$. ✗





[KSV06] Kupferman, Safra, Vardi. Relating word and tree automata. Annals Pure & Applied Logic, 138(1-3):126-146, 2006.
[BKKS13] Boker *et al.* Nondeterminism in the Presence of a Diverse or Unknown Future. In ICALP'13, p. 89-100. Springer, 2013.

From word to tree automata

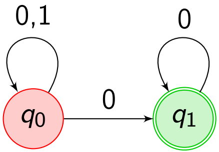
Derived language [KSV06]

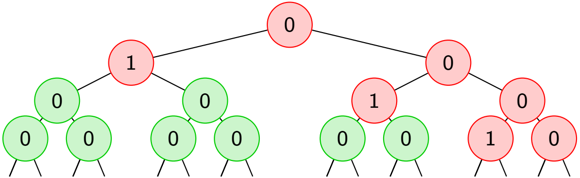
Given $L \subseteq \Sigma^\omega$, we let ΔL be the set of trees all of whose infinite branches are labelled with words in L.

If L is accepted by $\mathcal{B}$, is ΔL accepted by some tree automaton $\mathcal{C}$?

Natural construction for non-deterministic automata

We let $\delta_{\mathcal{C}}(q,\sigma) = \{(q',q'') \mid q',q'' \in \delta_{\mathcal{B}}(q,\sigma)\}$. ✗





[KSV06] Kupferman, Safra, Vardi. Relating word and tree automata. Annals Pure & Applied Logic, 138(1-3):126-146, 2006.
[BKKS13] Boker *et al.* Nondeterminism in the Presence of a Diverse or Unknown Future. In ICALP'13, p. 89-100. Springer, 2013.

From word to tree automata

Derived language [KSV06]

Given $L \subseteq \Sigma^\omega$, we let ΔL be the set of trees all of whose infinite branches are labelled with words in L.

If L is accepted by $\mathcal{B}$, is ΔL accepted by some tree automaton $\mathcal{C}$?

Natural construction for non-deterministic automata

We let $\delta_{\mathcal{C}}(q, \sigma) = \{(q', q'') \mid q', q'' \in \delta_{\mathcal{B}}(q, \sigma)\}$. ✗

Good-for-trees automata [BKKS13]

A word automaton $\mathcal{B}$ accepting some language L is good-for-trees if the associated tree automaton $\mathcal{C}$ accepts ΔL.

[KSV06] Kupferman, Safra, Vardi. Relating word and tree automata. Annals Pure & Applied Logic, 138(1-3):126-146, 2006.
[BKKS13] Boker *et al.* Nondeterminism in the Presence of a Diverse or Unknown Future. In ICALP'13, p. 89-100. Springer, 2013.

Reactive synthesis

Environment
Controller

Reactive synthesis

Environment $i_1 = a$

Controller

Reactive synthesis

Environment	$i_1 = a$
Controller	$o_1 = \alpha$

Reactive synthesis

Environment	$i_1 = a$	$i_2 = b$
Controller	$o_1 = \alpha$	

Reactive synthesis

Environment	$i_1 = a$	$i_2 = b$
Controller	$o_1 = \alpha$	$o_2 = \alpha$

Reactive synthesis

Environment	$i_1 = a$	$i_2 = b$	$i_3 = b$
Controller	$o_1 = \alpha$	$o_2 = \alpha$	

Reactive synthesis

Environment	$i_1 = a$	$i_2 = b$	$i_3 = b$
Controller	$o_1 = \alpha$	$o_2 = \alpha$	$o_3 = \beta$

Reactive synthesis

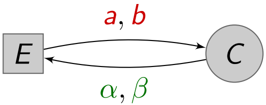
Environment	$i_1 = a$	$i_2 = b$	$i_3 = b$	$\cdots$
Controller	$o_1 = \alpha$	$o_2 = \alpha$	$o_3 = \beta$	$\cdots$

Reactive synthesis

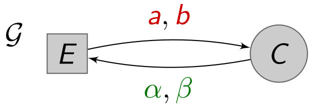
Environment $i_1 = a \quad i_2 = b \quad i_3 = b \quad \cdots$

Controller $o_1 = \alpha \quad o_2 = \alpha \quad o_3 = \beta \quad \cdots$

$\mathcal{G}$



Reactive synthesis

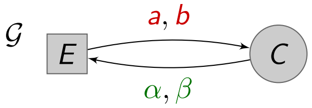


Environment	$i_1 = a$	$i_2 = b$	$i_3 = b$	$\cdots$
Controller	$o_1 = \alpha$	$o_2 = \alpha$	$o_3 = \beta$	$\cdots$

Controller wins if, and only if, $i_1 \; o_1 \; i_2 \; o_2 \; i_3 \; o_3 \; \ldots \in L$

Reactive synthesis

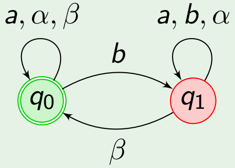
Environment	$i_1 = a$	$i_2 = b$	$i_3 = b$	$\cdots$
Controller	$o_1 = \alpha$	$o_2 = \alpha$	$o_3 = \beta$	$\cdots$



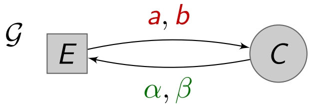
Controller wins if, and only if, $i_1\ o_1\ i_2\ o_2\ i_3\ o_3\ \ldots \in L$

Example

if L is defined by a determinstic automaton $\mathcal{D}$: $(\mathbf{G}(b \Rightarrow \mathbf{F}\,\beta))$



Reactive synthesis

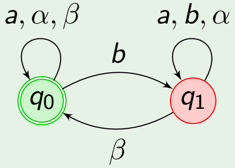


Environment	$i_1 = a$	$i_2 = b$	$i_3 = b$	$\cdots$
Controller	$o_1 = \alpha$	$o_2 = \alpha$	$o_3 = \beta$	$\cdots$

Controller wins if, and only if, $i_1\ o_1\ i_2\ o_2\ i_3\ o_3\ \ldots \in L$

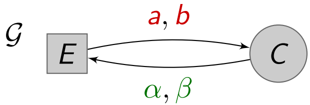
Example

if L is defined by a determinstic automaton $\mathcal{D}$: $(\mathbf{G}(b \Rightarrow \mathbf{F}\,\beta))$



- take product of $\mathcal{D}$ and $\mathcal{G}$
- solve the resulting game (in polynomial time).

Reactive synthesis

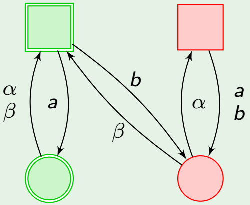


Environment	$i_1 = a$	$i_2 = b$	$i_3 = b$	$\cdots$
Controller	$o_1 = \alpha$	$o_2 = \alpha$	$o_3 = \beta$	$\cdots$

Controller wins if, and only if, $i_1\ o_1\ i_2\ o_2\ i_3\ o_3\ \ldots \in L$

Example

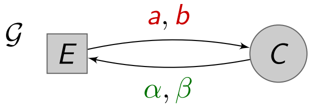
if L is defined by a determinstic automaton $\mathcal{D}$: $(\mathbf{G}(b \Rightarrow \mathbf{F}\,\beta))$



- take product of $\mathcal{D}$ and $\mathcal{G}$
- solve the resulting game (in polynomial time).

Reactive synthesis

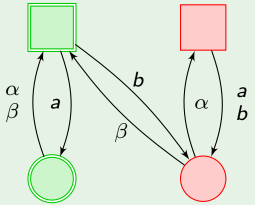
Environment	$i_1 = a$	$i_2 = b$	$i_3 = b$	$\cdots$
Controller	$o_1 = \alpha$	$o_2 = \alpha$	$o_3 = \beta$	$\cdots$



Controller wins if, and only if, $i_1\ o_1\ i_2\ o_2\ i_3\ o_3\ \ldots \in L$

Example

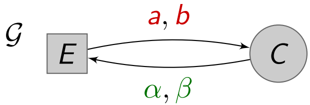
if L is defined by a determinstic automaton $\mathcal{D}$: $(\mathbf{G}(b \Rightarrow \mathbf{F}\,\beta))$



- take product of $\mathcal{D}$ and $\mathcal{G}$
- solve the resulting game (in polynomial time).

✓

Reactive synthesis

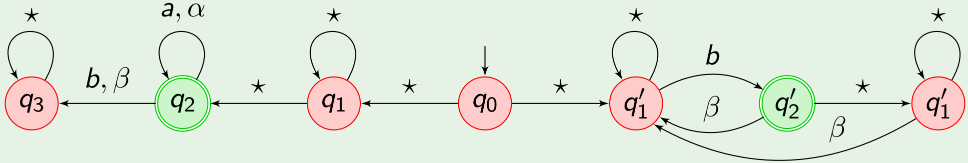


Environment	$i_1 = a$	$i_2 = b$	$i_3 = b$	$\cdots$
Controller	$o_1 = \alpha$	$o_2 = \alpha$	$o_3 = \beta$	$\cdots$

Controller wins if, and only if, $i_1\ o_1\ i_2\ o_2\ i_3\ o_3\ \ldots \in L$

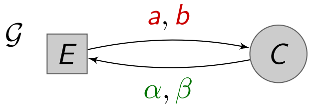
Example

if L is defined by non-determinstic automaton $\mathcal{N}$: $(\mathbf{G}\,\mathbf{F}\, b \iff \mathbf{G}\,\mathbf{F}\, \beta)$



Reactive synthesis

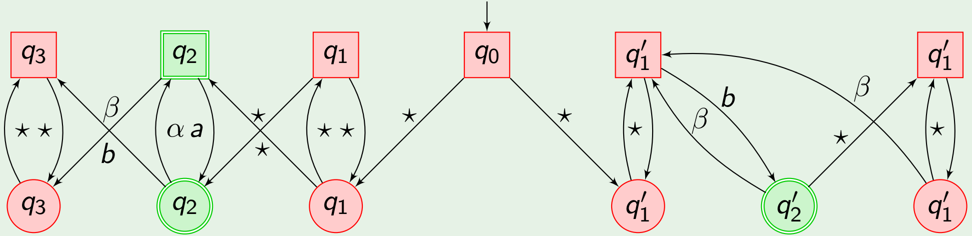
Environment	$i_1 = a$	$i_2 = b$	$i_3 = b$	$\cdots$
Controller	$o_1 = \alpha$	$o_2 = \alpha$	$o_3 = \beta$	$\cdots$



Controller wins if, and only if, $i_1\ o_1\ i_2\ o_2\ i_3\ o_3\ \ldots \in L$

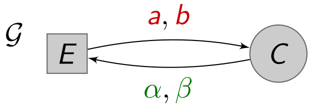
Example

if L is defined by non-determinstic automaton $\mathcal{N}$: $(\mathbf{G}\,\mathbf{F}\, b \iff \mathbf{G}\,\mathbf{F}\, \beta)$



Reactive synthesis

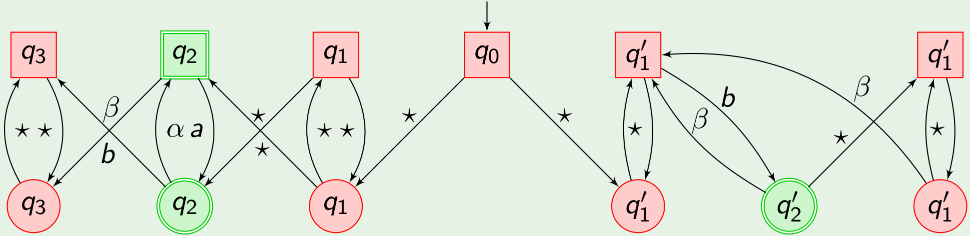
Environment	$i_1 = a$	$i_2 = b$	$i_3 = b$	$\cdots$
Controller	$o_1 = \alpha$	$o_2 = \alpha$	$o_3 = \beta$	$\cdots$



Controller wins if, and only if, $i_1\ o_1\ i_2\ o_2\ i_3\ o_3\ \ldots \in L$

Example

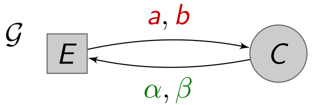
if L is defined by non-determinstic automaton $\mathcal{N}$: $(\mathbf{G}\,\mathbf{F}\, b \iff \mathbf{G}\,\mathbf{F}\,\beta)$



✗

Reactive synthesis

Environment	$i_1 = a$	$i_2 = b$	$i_3 = b$	$\cdots$
Controller	$o_1 = \alpha$	$o_2 = \alpha$	$o_3 = \beta$	$\cdots$



Controller wins if, and only if, $i_1\ o_1\ i_2\ o_2\ i_3\ o_3\ \ldots \in L$

Good-for-games automata [HP06]

An automaton $\mathcal{A}$ on alphabet Σ is good-for-games if for any two-player zero-sum game $\mathcal{G}$ with Σ-labelled transitions and winning condition $\mathcal{L}(\mathcal{A})$, the games $\mathcal{G}$ and $\mathcal{G} \times \mathcal{A}$ have the same winner.

[HP06] Henzinger, Piterman. Solving Games Without Determinization. In CSL 2006, p. 395-410. Springer, 2006.

History-deterministic automata

Letter game (aka. monitor game)

The letter game ona word automaton $\mathcal{A}$ is a 2-player game played as follows:

- initially, a token is placed on the (unique, wlog) initial state of $\mathcal{A}$;
- iteratively, and *ad infinitum*:
  - Adam proposes a letter σ in Σ;
  - Eve moves the token along a σ-transition in $\mathcal{A}$.

  In this process, Adam defines a word w, and Eve builds an run π of $\mathcal{A}$ on w.
- Adam wins if w is accepted by $\mathcal{A}$ and π is not an accepting run; Eve wins othw.

History-deterministic automata

Letter game (aka. monitor game)

The letter game ona word automaton $\mathcal{A}$ is a 2-player game played as follows:

- initially, a token is placed on the (unique, wlog) initial state of $\mathcal{A}$;
- iteratively, and *ad infinitum*:
  - Adam proposes a letter σ in Σ;
  - Eve moves the token along a σ-transition in $\mathcal{A}$.

  In this process, Adam defines a word w, and Eve builds an run π of $\mathcal{A}$ on w.
- Adam wins if w is accepted by $\mathcal{A}$ and π is not an accepting run; Eve wins othw.

History-deterministic automata [HP06,Col09]

Automaton $\mathcal{A}$ is history-deterministic if Eve has a winning strategy (called resolver) in the letter game.

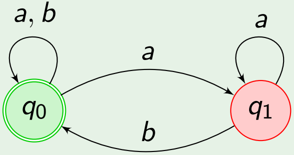
[HP06] Henzinger, Piterman. Solving Games Without Determinization. In CSL 2006, p. 395-410. Springer, 2006.

[Col09] Colcombet. The Theory of Stabilisation Monoids and Regular Cost Functions. In ICALP'09, p. 139-150. Springer, 2009.

History-deterministic automata

Example

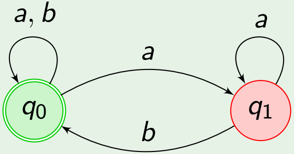
This co-Büchi automaton accepting $\{w \mid |w|_b < \infty\}$ is history-deterministic:



History-deterministic automata

Example

This co-Büchi automaton accepting $\{w \mid |w|_b < \infty\}$ is history-deterministic:

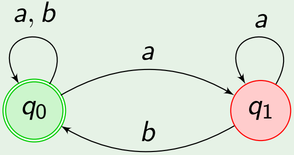


Resolver: always go to q_1 when reading a in q_0.

History-deterministic automata

Example

This co-Büchi automaton accepting $\{w \mid |w|_b < \infty\}$ is history-deterministic:



Resolver: always go to q_1 when reading a in q_0.

Determinizable-by-pruning automata [AKL10]

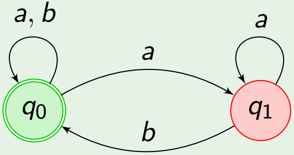
An automaton $\mathcal{A}$ if determinizable-by-pruning if a language-equivalent deterministic automaton can be obtained from $\mathcal{A}$ by removing some transitions.

[AKL10] Aminof, Kupferman, Lampert. Reasoning about online algorithms with weighted automata. ACM Trans. Alg. 6(2):28. ACM, 2010

History-deterministic automata

Example

This co-Büchi automaton accepting $\{w \mid |w|_b < \infty\}$ is history-deterministic:



Resolver: always go to q_1 when reading a in q_0.

Determinizable-by-pruning automata [AKL10]

An automaton $\mathcal{A}$ if determinizable-by-pruning if a language-equivalent deterministic automaton can be obtained from $\mathcal{A}$ by removing some transitions.

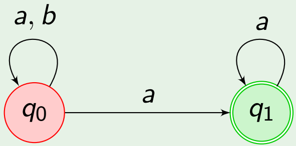
Obviously, determinizability-by-pruning implies history-determinism.

[AKL10] Aminof, Kupferman, Lampert. Reasoning about online algorithms with weighted automata. ACM Trans. Alg. 6(2):28. ACM, 2010

History-deterministic automata

Example

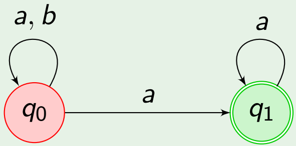
This Büchi automaton accepting $\{w \mid |w|_b < \infty\}$ is not history-deterministic:



History-deterministic automata

Example

This Büchi automaton accepting $\{w \mid |w|_b < \infty\}$ is not history-deterministic:



Eve does not win the letter game:

- Adam plays letter a as long as the token is in q_0;
- If Eve never move to q_1, she looses; otherwise, from q_1, Adam propose suffix ba^ω.

Results for ω-automata

Theorem ([HP06,BKKS13,BL19])

Good-for-treeness, good-for-gameness and history-determinism are equivalent.

Determinizability-by-pruning is stronger.

[HP06] Henzinger, Piterman. Solving Games Without Determinization. In CSL 2006, p. 395-410. Springer, 2006.
[BKKS13] Boker *et al.* Nondeterminism in the Presence of a Diverse or Unknown Future. In ICALP'13, p. 89-100. Springer, 2013.
[BL19] Boker, Lehtinen. Good for Games Automata: From Nondeterminism to Alternation. In CONCUR'19, p 19:1-19:16. LZI, 2019.

Results for ω-automata

Theorem ([HP06,BKKS13,BL19])

Good-for-treeness, good-for-gameness and history-determinism are equivalent.

Determinizability-by-pruning is stronger.

Proof

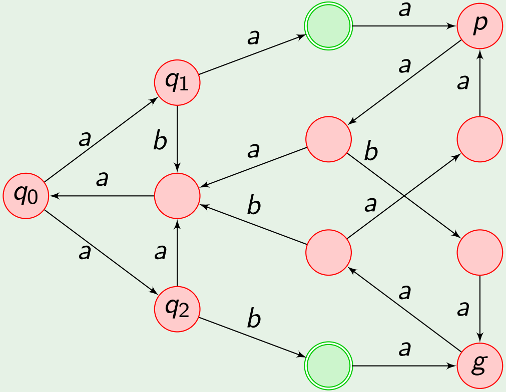
- history-determinism implies good-for-gameness and game-for-treeness: the resolver can be used to resolve non-determinism.
- The other direction is a bit more involved.
- That determinizability-by-pruning is stronger is shown with the next example.

[HP06] Henzinger, Piterman. Solving Games Without Determinization. In CSL 2006, p. 395-410. Springer, 2006.
[BKKS13] Boker *et al.* Nondeterminism in the Presence of a Diverse or Unknown Future. In ICALP'13, p. 89-100. Springer, 2013.
[BL19] Boker, Lehtinen. Good for Games Automata: From Nondeterminism to Alternation. In CONCUR'19, p 19:1-19:16. LZI, 2019.

History-deterministic automata

Example

This Büchi automaton accepting $[(aaa + aba)^* \cdot (aaa\ aaa + aba\ aba)]^\omega$ is history-deterministic, but not determinizable-by-pruning:

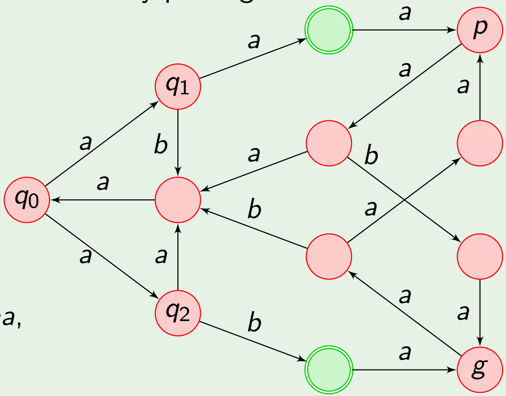


[BKKS13] Boker *et al.* Nondeterminism in the Presence of a Diverse or Unknown Future. In ICALP'13, p 89-100. Springer, 2013.

History-deterministic automata

Example

This Büchi automaton accepting $[(aaa + aba)^* \cdot (aaa\ aaa + aba\ aba)]^\omega$ is history-deterministic, but not determinizable-by-pruning:



Resolver:

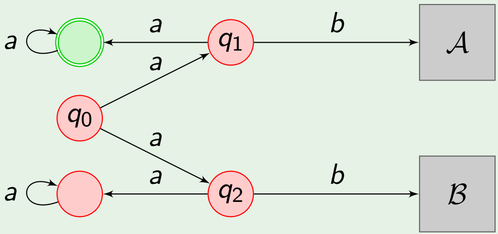
In q_0, go to q_1 if we just read aaa, otherwise, go to q_2.

[BKKS13] Boker *et al.* Nondeterminism in the Presence of a Diverse or Unknown Future. In ICALP'13, p 89-100. Springer, 2013.

Deciding history-determinism

Example (Hardness)

If $\mathcal{A}$ is deterministic, the automaton below is history-deterministic (and determinizable-by-pruning) if, and only if, $\mathcal{L}(\mathcal{B}) \subseteq \mathcal{L}(\mathcal{A})$.



[LZ22] Lehtinen, Zimmermann. Good-for-games ω-pushdown automata. LMCS 18(1):3. 2022.

Deciding history-determinism: k-token games

k-token games

Given an automaton $\mathcal{A}$, the k-token game $\mathcal{G}_k(\mathcal{A})$ on $\mathcal{A}$ runs as follows:

- initially, k tokens of Adam and one token of Eve are on the initial state of $\mathcal{A}$;
- repeatedly, *ad infinitum*:
  - Adam proposes a letter $\sigma \in \Sigma$;
  - Eve moves her token along a σ-transition;
  - Adam moves his k tokens along σ-transitions.
- Eve wins if either Adam fails to produce an accepting run, or if she produces an accepting run.

[BK18] Bagnol and Kuperberg. Büchi Good-for-game Automata are Efficiently Recognizable. In FSTTCS'18, p. 16:1-16:14. LZI, 2018.

Deciding history-determinism: k-token games

Lemma

If $\mathcal{A}$ is history-deterministic, then Eve wins $\mathcal{G}_k(\mathcal{A})$.

[BK18] Bagnol and Kuperberg. Büchi Good-for-game Automata are Efficiently Recognizable. In FSTTCS'18, p. 16:1-16:14. LZI, 2018.

Deciding history-determinism: k-token games

Lemma

If $\mathcal{A}$ is history-deterministic, then Eve wins $\mathcal{G}_k(\mathcal{A})$.

Lemma

There exists a non-history-deterministic automaton $\mathcal{B}$ for which Eve wins $\mathcal{G}_1(\mathcal{B})$.

[BK18] Bagnol and Kuperberg. Büchi Good-for-game Automata are Efficiently Recognizable. In FSTTCS'18, p. 16:1-16:14. LZI, 2018.

Deciding history-determinism: k-token games

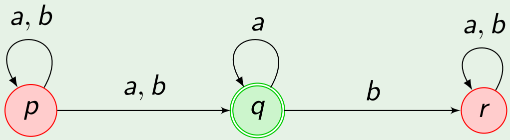
Lemma

If $\mathcal{A}$ is history-deterministic, then Eve wins $\mathcal{G}_k(\mathcal{A})$.

Lemma

There exists a non-history-deterministic automaton $\mathcal{B}$ for which Eve wins $\mathcal{G}_1(\mathcal{B})$.

Proof



- Adam wins the letter game: play a until Eve moves to q; then play ba^ω;
- Eve wins the 1-token game: follow Adam's token.

[BK18] Bagnol and Kuperberg. Büchi Good-for-game Automata are Efficiently Recognizable. In FSTTCS'18, p. 16:1-16:14. LZI, 2018.

Deciding history-determinism: k-token games

Lemma

Eve wins $\mathcal{G}_2(\mathcal{A})$ if, and only if, she wins $\mathcal{G}_k(\mathcal{A})$ for all $k \geq 2$.

[BK18] Bagnol and Kuperberg. Büchi Good-for-game Automata are Efficiently Recognizable. In FSTTCS'18, p. 16:1-16:14. LZI, 2018.

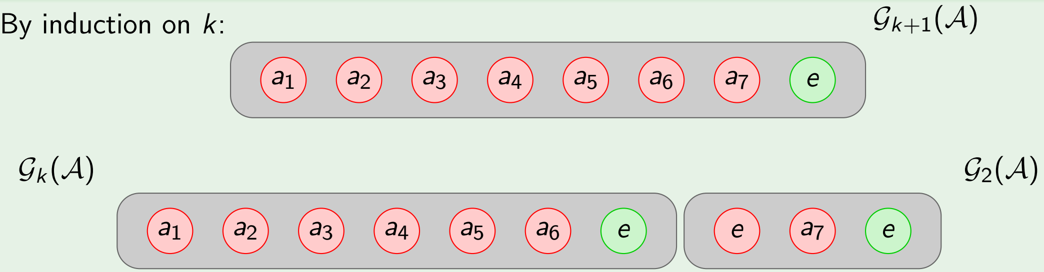
Deciding history-determinism: k-token games

Lemma

Eve wins $\mathcal{G}_2(\mathcal{A})$ if, and only if, she wins $\mathcal{G}_k(\mathcal{A})$ for all $k \geq 2$.

Proof

By induction on k:



[BK18] Bagnol and Kuperberg. Büchi Good-for-game Automata are Efficiently Recognizable. In FSTTCS'18, p. 16:1-16:14. LZI, 2018.

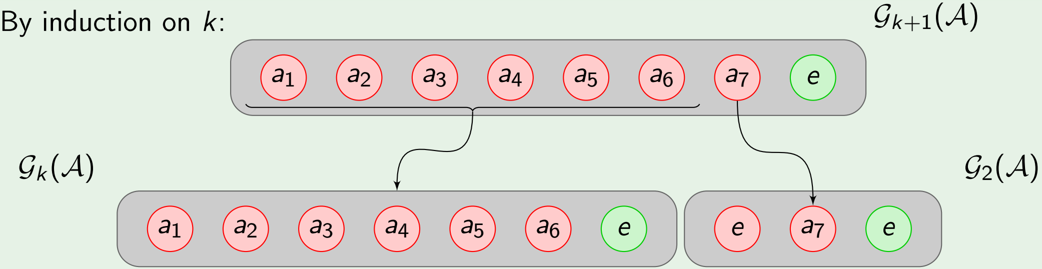
Deciding history-determinism: k-token games

Lemma

Eve wins $\mathcal{G}_2(\mathcal{A})$ if, and only if, she wins $\mathcal{G}_k(\mathcal{A})$ for all $k \geq 2$.

Proof

By induction on k:



[BK18] Bagnol and Kuperberg. Büchi Good-for-game Automata are Efficiently Recognizable. In FSTTCS'18, p. 16:1-16:14. LZI, 2018.

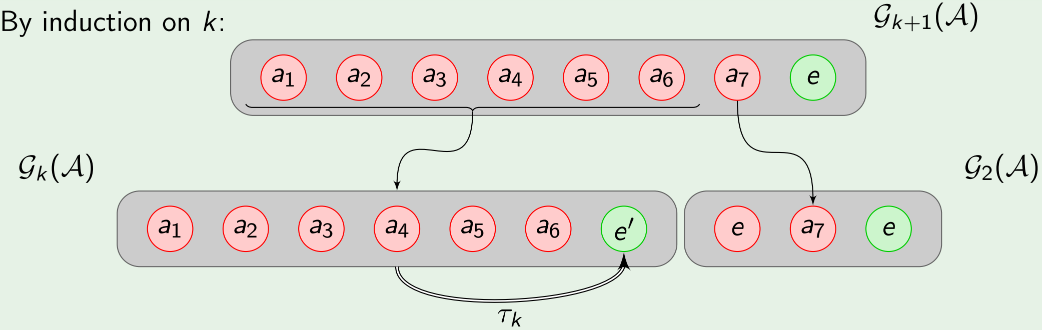
Deciding history-determinism: k-token games

Lemma

Eve wins $\mathcal{G}_2(\mathcal{A})$ if, and only if, she wins $\mathcal{G}_k(\mathcal{A})$ for all $k \geq 2$.

Proof

By induction on k:



[BK18] Bagnol and Kuperberg. Büchi Good-for-game Automata are Efficiently Recognizable. In FSTTCS'18, p. 16:1-16:14. LZI, 2018.

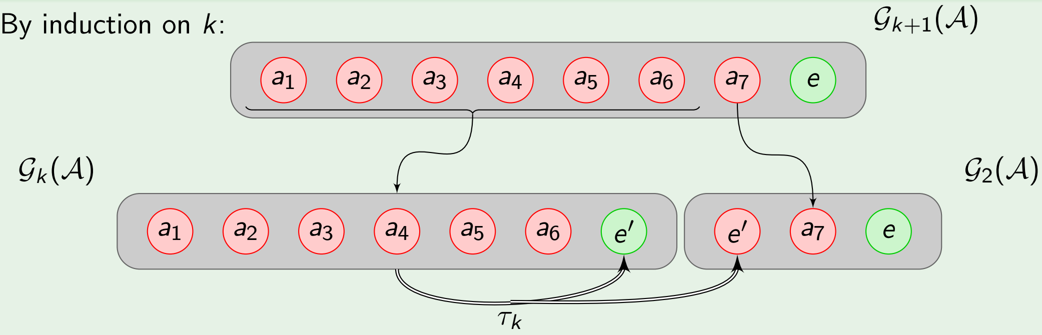
Deciding history-determinism: k-token games

Lemma

Eve wins $\mathcal{G}_2(\mathcal{A})$ if, and only if, she wins $\mathcal{G}_k(\mathcal{A})$ for all $k \geq 2$.

Proof

By induction on k:



[BK18] Bagnol and Kuperberg. Büchi Good-for-game Automata are Efficiently Recognizable. In FSTTCS'18, p. 16:1-16:14. LZI, 2018.

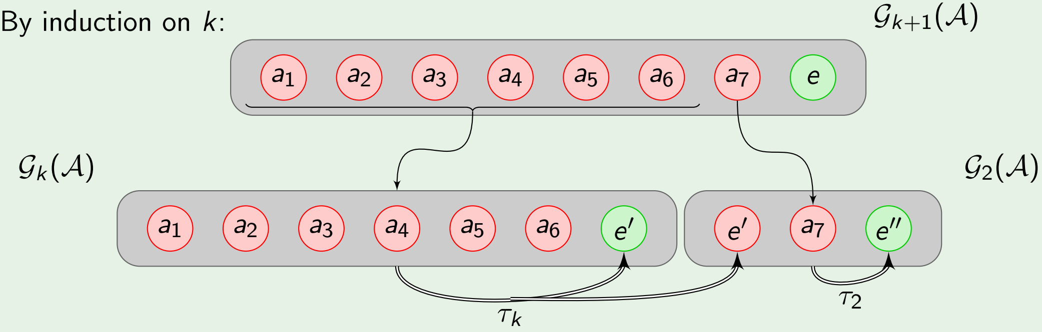
Deciding history-determinism: k-token games

Lemma

Eve wins $\mathcal{G}_2(\mathcal{A})$ if, and only if, she wins $\mathcal{G}_k(\mathcal{A})$ for all $k \geq 2$.

Proof

By induction on k:



[BK18] Bagnol and Kuperberg. Büchi Good-for-game Automata are Efficiently Recognizable. In FSTTCS'18, p. 16:1-16:14. LZI, 2018.

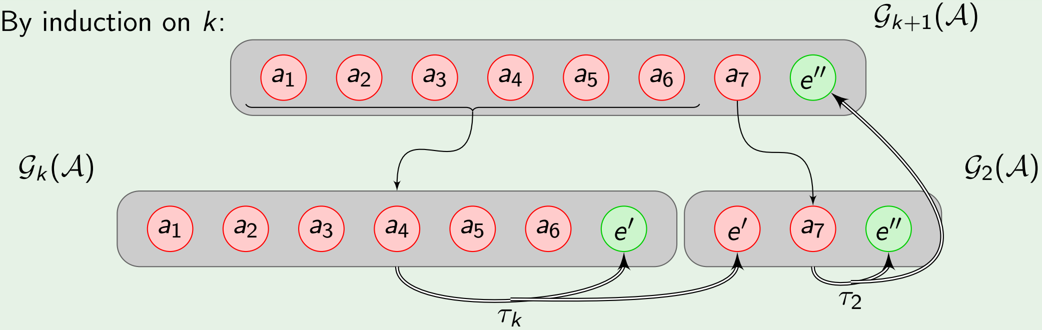
Deciding history-determinism: k-token games

Lemma

Eve wins $\mathcal{G}_2(\mathcal{A})$ if, and only if, she wins $\mathcal{G}_k(\mathcal{A})$ for all $k \geq 2$.

Proof

By induction on k:



[BK18] Bagnol and Kuperberg. Büchi Good-for-game Automata are Efficiently Recognizable. In FSTTCS'18, p. 16:1-16:14. LZI, 2018.

Deciding history-determinism: k-token games

Lemma

Eve wins $\mathcal{G}_2(\mathcal{A})$ if, and only if, she wins $\mathcal{G}_k(\mathcal{A})$ for all $k \geq 2$.

Theorem (BK19,BKLS20)

For Büchi and co-Büchi acceptance conditions, $\mathcal{A}$ is history-deterministic if, and only if, Eve wins the 2-token game.

[BK18] Bagnol and Kuperberg. Büchi Good-for-game Automata are Efficiently Recognizable. In FSTTCS'18, p. 16:1-16:14. LZI, 2018.
[BKLS20] Boker *et al.* On Succinctness and Recognisability of Alternating Good-for-Games Automata. arXiv preprint 2002.07278.

Deciding history-determinism: k-token games

Lemma

Eve wins $\mathcal{G}_2(\mathcal{A})$ if, and only if, she wins $\mathcal{G}_k(\mathcal{A})$ for all $k \geq 2$.

Theorem (BK19,BKLS20)

For Büchi and co-Büchi acceptance conditions, $\mathcal{A}$ is history-deterministic if, and only if, Eve wins the 2-token game.

Corollary

History-determinism is decidable in polynomial time for Büchi and co-Büchi automata.

[BK18] Bagnol and Kuperberg. Büchi Good-for-game Automata are Efficiently Recognizable. In FSTTCS'18, p. 16:1-16:14. LZI, 2018.
[BKLS20] Boker *et al.* On Succinctness and Recognisability of Alternating Good-for-Games Automata. arXiv preprint 2002.07278.

More results and open problems on history-determinism

Deciding history-determinism

- *does the 2-token game characterize history-determinism for parity automata?*

More results and open problems on history-determinism

Deciding history-determinism

- *does the 2-token game characterize history-determinism for parity automata?*

Expressiveness [KS15]

- ω-automata can be determinized, so on expressiveness gap;
- co-Büchi history-deterministic automata can be exponentially more succinct than deterministic ones.
- *is there an equivalent gap for Büchi automata?*

[KS15] Kuperberg and Skrzypczak. On determinisation of good-for-games automata. In ICALP'15, p. 299-310. Springer, 2015.

More results and open problems on history-determinism

Deciding history-determinism

- *does the 2-token game characterize history-determinism for parity automata?*

Expressiveness [KS15]

- ω-automata can be determinized, so on expressiveness gap;
- co-Büchi history-deterministic automata can be exponentially more succinct than deterministic ones.
- *is there an equivalent gap for Büchi automata?*

Extensions of automata [LZ20,BHL$^+$22]

- history-determinism is undecidable for pushdown automata;
- history-deterministic timed automata are not determinizable.

[KS15] Kuperberg and Skrzypczak. On determinisation of good-for-games automata. In ICALP'15, p. 299-310. Springer, 2015.
[LZ20] Lehtinen and Zimmermann. Good-for-games ω-pushdown automata. In LICS'20, p. 689-702. IEEE, 2020.
[BHL$^+$22] Bose *et al.* History-deterministic timed automata are not determinizable. In RP'22, p. 67-76. Springer, 2022.