On determinisation of Good-For-Games automata

Michał Skrzypczak
joint work with Denis Kuperberg

LIAFA, University of Warsaw

LIAFA
IV 2015 Paris

Synthesis

Synthesis

φ — specification (i.e. MSO)

Synthesis

φ — specification (i.e. MSO)

$\rightsquigarrow$

— machine

Synthesis

φ — specification (i.e. MSO)

$\rightsquigarrow$

— machine

Synthesis

φ — specification (i.e. MSO)

⇝

 — machine

Environment:

Synthesis

Environment: I_0,

φ — specification (i.e. MSO)

 — machine

Synthesis

φ — specification (i.e. MSO)

$\rightsquigarrow$

 — machine

Environment: $I_0,$

$\downarrow$

$\downarrow$

$O_0,$

Synthesis

φ — specification (i.e. MSO)

⇝

 — machine

Environment: $I_0, \ I_1,$

↓

↓

$O_0,$

Synthesis

φ — specification (i.e. MSO)

$\rightsquigarrow$

 — machine

Environment: I_0, I_1,

$\downarrow$

$\downarrow$

O_0, O_1,

Synthesis

φ — specification (i.e. MSO)

$\rightsquigarrow$

— machine

Environment: $I_0, \;\; I_1, \;\; I_2,$

$\downarrow$

$\downarrow$

$O_0, \; O_1,$

Synthesis

Environment: $I_0, \; I_1, \; I_2,$

$\downarrow$

$\downarrow$

$O_0, \; O_1, \; O_2,$

φ — specification (i.e. MSO)

$\rightsquigarrow$

 — machine

Synthesis

φ — specification (i.e. MSO)

$\rightsquigarrow$

 — machine

Environment: $I_0, \; I_1, \; I_2, \; \ldots$

$\downarrow$

$\downarrow$

$O_0, \; O_1, \; O_2, \; \ldots$

Synthesis

φ — specification (i.e. MSO)

$\rightsquigarrow$

 — machine

Environment: $I_0, \ I_1, \ I_2, \ \ldots$

$\downarrow$

$\downarrow$

$O_0, \ O_1, \ O_2, \ \ldots$

$$\big(I_0, O_0, \ I_1, O_1, \ \ldots\big) \models \varphi$$

Synthesis

φ — specification (i.e. MSO)

⇝

 — machine

Environment: $I_0, \; I_1, \; I_2, \; \ldots$

$\downarrow$

$\downarrow$

$O_0, \; O_1, \; O_2, \; \ldots$

$$\big(I_0, O_0, \; I_1, O_1, \; \ldots\big) \models \varphi$$

Game semantics

Synthesis

φ — specification (i.e. MSO)

$\rightsquigarrow$

 — machine

Environment: $I_0,\ I_1,\ I_2,\ \ldots$

$\downarrow$

$\downarrow$

$O_0,\ O_1,\ O_2,\ \ldots$

$$\big(I_0, O_0,\ I_1, O_1,\ \ldots\big) \models \varphi$$

Game semantics

Two players, perfect information

Synthesis

φ — specification (i.e. MSO)

$\rightsquigarrow$

 — machine

Environment: $I_0,\ I_1,\ I_2,\ \dots$

$\downarrow$

$\downarrow$

$O_0,\ O_1,\ O_2,\ \dots$

$$\big(I_0, O_0,\ I_1, O_1,\ \dots\big) \models \varphi$$

Game semantics

Two players, perfect information

$\forall$: I_i

$\exists$: O_i

Synthesis

φ — specification (i.e. MSO)

$\rightsquigarrow$

 — machine

Environment: $I_0,\ I_1,\ I_2,\ \dots$

$\downarrow$

$\downarrow$

$O_0,\ O_1,\ O_2,\ \dots$

$$\big(I_0, O_0,\ I_1, O_1,\ \dots\big) \models \varphi$$

Game semantics

Two players, perfect information

$\forall$: I_i

$\exists$: O_i

$\exists$ wins if $\big(I_0, O_0,\ \dots\big) \models \varphi$

Synthesis

φ — specification (i.e. MSO)

$\rightsquigarrow$

 — machine

Environment: $I_0, \; I_1, \; I_2, \; \ldots$

$\downarrow$

$\downarrow$

$O_0, \; O_1, \; O_2, \; \ldots$

$\big(I_0, O_0, \; I_1, O_1, \; \ldots\big) \models \varphi$

Game semantics

Two players, perfect information

$\forall$: I_i

$\exists$: O_i

$\exists$ wins if $\underbrace{\big(I_0, O_0, \; \ldots\big) \models \varphi}_{\omega\text{-regular winning condition}}$

Synthesis

φ — specification (i.e. MSO)

$\rightsquigarrow$

— machine

Environment: $I_0,\ I_1,\ I_2,\ \ldots$

$\downarrow$

$\downarrow$

$O_0,\ O_1,\ O_2,\ \ldots$

$\big(I_0, O_0,\ I_1, O_1,\ \ldots\big) \models \varphi$

Game semantics

Two players, perfect information

$\forall$: I_i

$\exists$: O_i

(Büchi, Landweber ['69])

$\exists$ wins if $\underbrace{\big(I_0, O_0,\ \ldots\big) \models \varphi}$

ω-regular winning condition

Synthesis

φ — specification (i.e. MSO)

⇝

— machine

Environment: $I_0,\ I_1,\ I_2,\ \ldots$

↓

↓

$O_0,\ O_1,\ O_2,\ \ldots$

$$\big(I_0, O_0,\ I_1, O_1,\ \ldots\big) \models \varphi$$

Game semantics

Two players, perfect information

$\forall$: I_i

$\exists$: O_i

(Büchi, Landweber ['69])

$\exists$ wins if $\underbrace{\big(I_0, O_0,\ \ldots\big) \models \varphi}_{\omega\text{-regular winning condition}}$

Solution

Synthesis

φ — specification (i.e. MSO)

$\rightsquigarrow$

 — machine

Environment: $I_0, \ I_1, \ I_2, \ \ldots$

$\downarrow$

$\downarrow$

$O_0, \ O_1, \ O_2, \ \ldots$

$$\big(I_0, O_0, \ I_1, O_1, \ \ldots\big) \models \varphi$$

Game semantics

Two players, perfect information

$\forall$: I_i

$\exists$: O_i

(Büchi, Landweber ['69])

$\exists$ wins if $\underbrace{\big(I_0, O_0, \ \ldots\big) \models \varphi}$

ω-regular winning condition

Solution

$\varphi \rightsquigarrow \mathcal{A}_{\mathbf{det}}$ — det. automaton

Synthesis

φ — specification (i.e. MSO)

$\rightsquigarrow$

— machine

Environment: $I_0, \ I_1, \ I_2, \ \ldots$

$\downarrow$

$\downarrow$

$O_0, \ O_1, \ O_2, \ \ldots$

$(I_0, O_0, \ I_1, O_1, \ \ldots) \models \varphi$

Game semantics

Two players, perfect information

$\forall$: I_i

$\exists$: O_i

(Büchi, Landweber ['69])

$\exists$ wins if $\underbrace{(I_0, O_0, \ \ldots) \models \varphi}_{\omega\text{-regular winning condition}}$

Solution

$\varphi \rightsquigarrow \mathcal{A}_{\mathbf{det}}$ — det. automaton

$\forall$: I_i

Synthesis

φ — specification (i.e. MSO)

$\rightsquigarrow$

 — machine

Environment: $I_0, \; I_1, \; I_2, \; \ldots$

$\downarrow$

$\downarrow$

$O_0, \; O_1, \; O_2, \; \ldots$

$$\big(I_0, O_0, \; I_1, O_1, \; \ldots\big) \models \varphi$$

Game semantics

Two players, perfect information

$\forall$: I_i

$\exists$: O_i

(Büchi, Landweber ['69])

$\exists$ wins if $\underbrace{\big(I_0, O_0, \; \ldots\big) \models \varphi}_{\omega\text{-regular winning condition}}$

Solution

$\varphi \rightsquigarrow \mathcal{A}_{\mathbf{det}}$ — det. automaton

$\forall$: I_i

$\exists$: O_i

Synthesis

φ — specification (i.e. MSO)

$\rightsquigarrow$

 — machine

Environment: $I_0, \ I_1, \ I_2, \ \ldots$

$\downarrow$

$\downarrow$

$O_0, \ O_1, \ O_2, \ \ldots$

$$\big(I_0, O_0, \ I_1, O_1, \ \ldots\big) \models \varphi$$

Game semantics

Two players, perfect information

$\forall$: I_i

$\exists$: O_i

(Büchi, Landweber ['69])

$\exists$ wins if $\underbrace{\big(I_0, O_0, \ \ldots\big) \models \varphi}_{\omega\text{-regular winning condition}}$

Solution

$\varphi \rightsquigarrow \mathcal{A}_{\mathbf{det}}$ — det. automaton

$\forall$: I_i

$\exists$: O_i

$\mathbf{det}$: transition of $\mathcal{A}_{\mathbf{det}}$

Synthesis

φ — specification (i.e. MSO)

$\rightsquigarrow$

 — machine

Game semantics

Two players, perfect information

$\forall$: I_i

$\exists$: O_i

(Büchi, Landweber ['69])

$\exists$ wins if $\underbrace{(I_0, O_0, \ \ldots) \models \varphi}$

ω-regular winning condition

Environment: $I_0, \ I_1, \ I_2, \ \ldots$

$\downarrow$

$\downarrow$

$O_0, \ O_1, \ O_2, \ \ldots$

$(I_0, O_0, \ I_1, O_1, \ \ldots) \models \varphi$

Solution

$\varphi \rightsquigarrow \mathcal{A}_{\mathbf{det}}$ — det. automaton

$\forall$: I_i

$\exists$: O_i

$\mathbf{det}$: transition of $\mathcal{A}_{\mathbf{det}}$

$\exists$ wins if

the run of $\mathcal{A}_{\mathbf{det}}$ is accepting

Complexity

Complexity

$\varphi \rightsquigarrow \mathcal{A}_{\mathbf{det}}$ — det. automaton

Complexity

$$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{det}} \text{ — det. automaton}}_{\textbf{expensive}!}$$

Complexity

$$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{det}} \text{ — det. automaton}}_{\textbf{expensive}!}$$

[e.g. 2-EXP for LTL]

Complexity

$$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{det}} \text{ — det. automaton}}_{\textbf{expensive}!}$$

[e.g. 2-EXP for LTL]

Idea

Complexity

$$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{det}} \text{ — det. automaton}}_{\textbf{expensive}!}$$

[e.g. 2-EXP for LTL]

Idea

$$\varphi \rightsquigarrow \mathcal{A}_{\mathbf{non-det}}$$

Complexity

$$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{det}} \text{ — det. automaton}}_{\textbf{expensive}!}$$

[e.g. 2-EXP for LTL]

Idea

$$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{non-det}}}_{\textbf{exponentially} \text{ more succinct!}}$$

Complexity

$$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{det}} \text{ — det. automaton}}_{\textbf{expensive}!}$$

[e.g. 2-EXP for LTL]

Idea

$$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{non-det}}}_{\textbf{exponentially} \text{ more succinct!}}$$

Game

Complexity

$$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{det}} \text{ — det. automaton}}_{\textbf{expensive}!}$$

[e.g. 2-EXP for LTL]

Idea

$$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{non-det}}}_{\textbf{exponentially} \text{ more succinct!}}$$

Game

$\forall$: I_i

Complexity

$$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{det}} \text{ — det. automaton}}_{\textbf{expensive}!}$$

[e.g. 2-EXP for LTL]

Idea

$$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{non-det}}}_{\textbf{exponentially} \text{ more succinct!}}$$

Game

$\forall$: I_i

$\exists$: O_i

Complexity

$$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{det}} \text{ — det. automaton}}_{\textbf{expensive}!}$$

[e.g. 2-EXP for LTL]

Idea

$$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{non-det}}}_{\textbf{exponentially} \text{ more succinct!}}$$

Game

$\forall$: I_i

$\exists$: O_i

$\exists$: transition of $\mathcal{A}_{\mathbf{non-det}}$

Complexity

$$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{det}} \text{ — det. automaton}}_{\textbf{expensive}!}$$

[e.g. 2-EXP for LTL]

Idea

$$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{non-det}}}_{\textbf{exponentially}\text{ more succinct!}}$$

Game

$\forall$: I_i

$\exists$: O_i

$\exists$: transition of $\mathcal{A}_{\mathbf{non-det}}$

$\forall$ may cheat!

Complexity

$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{det}} \text{ — det. automaton}}_{\textbf{expensive}!}$

[e.g. 2-EXP for LTL]

Idea

$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{non-det}}}$

exponentially more succinct!

Game

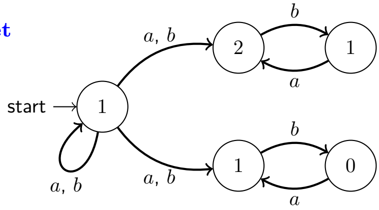
$\forall$: I_i

$\exists$: O_i

$\exists$: transition of $\mathcal{A}_{\mathbf{non-det}}$

$\forall$ may cheat!

$\mathcal{A}_{\mathbf{non-det}}$



Complexity

$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{det}} \text{ — det. automaton}}_{\textbf{expensive}!}$

[e.g. 2-EXP for LTL]

Idea

$\underbrace{\varphi \rightsquigarrow \mathcal{A}_{\mathbf{non-det}}}$

exponentially more succinct!

Game

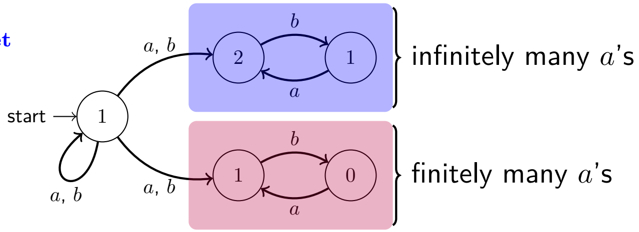
$\forall$: I_i

$\exists$: O_i

$\exists$: transition of $\mathcal{A}_{\mathbf{non-det}}$

$\forall$ may cheat!

$\mathcal{A}_{\mathbf{non-det}}$



Good-For-Games automata

Good-For-Games automata

$\mathcal{A}_{\mathbf{non-det}}$ is Good-For-Games (GFG) if

Good-For-Games automata

$\mathcal{A}_{\mathbf{non-det}}$ is Good-For-Games (GFG) if

$$\exists_{\sigma\colon \Sigma^* \to Q}$$

Good-For-Games automata

$\mathcal{A}_{\mathbf{non-det}}$ is Good-For-Games (GFG) if

$$\exists_{\sigma\colon \Sigma^* \to Q} \qquad \forall_{w \in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}})}$$

Good-For-Games automata

$\mathcal{A}_{\mathbf{non-det}}$ is Good-For-Games (GFG) if

$$\exists_{\sigma\colon \Sigma^* \to Q} \qquad \forall_{w \in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}})} \qquad \sigma(w) \text{ is accepting}$$

Good-For-Games automata

$\mathcal{A}_{\mathbf{non-det}}$ is Good-For-Games (GFG) if

$\Big(\sigma(), \sigma(w_0), \sigma(w_0 w_1), \ldots\Big)$

$$\exists_{\sigma\colon \Sigma^* \to Q} \qquad \forall_{w \in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}})} \qquad \sigma(w) \text{ is accepting}$$

Good-For-Games automata

$\mathcal{A}_{\mathbf{non-det}}$ is Good-For-Games (GFG) if

$$\big(\sigma(), \sigma(w_0), \sigma(w_0 w_1), \ldots\big)$$

$$\exists_{\underbrace{\sigma\colon \Sigma^* \to Q}_{\text{advice}}} \qquad \forall_{w \in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}})} \quad \sigma(w) \text{ is accepting}$$

Good-For-Games automata

$\mathcal{A}_{\mathbf{non-det}}$ is Good-For-Games (GFG) if

$$\big(\sigma(), \sigma(w_0), \sigma(w_0 w_1), \ldots\big)$$

$$\exists_{\underbrace{\sigma\colon \Sigma^* \to Q}_{\text{advice}}} \quad \underbrace{\forall_{w \in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}})} \quad \sigma(w) \text{ is accepting}}_{\sigma \text{ accepts whenever possible}}$$

Good-For-Games automata

$\mathcal{A}_{\mathbf{non-det}}$ is Good-For-Games (GFG) if

$$\big(\sigma(), \sigma(w_0), \sigma(w_0 w_1), \ldots\big)$$

$$\underbrace{\exists_{\sigma\colon \Sigma^* \to Q}}_{\text{advice}} \qquad \underbrace{\forall_{w \in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}})} \quad \sigma(w) \text{ is accepting}}_{\sigma \text{ accepts whenever possible}}$$

Synthesis (Henzinger, Piterman ['06])

Good-For-Games automata

$\mathcal{A}_{\mathbf{non-det}}$ is Good-For-Games (GFG) if

$$\underbrace{\exists_{\sigma\colon \Sigma^*\to Q}}_{\text{advice}} \qquad \underbrace{\forall_{w\in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}})} \quad \sigma(w) \text{ is accepting}}_{\sigma \text{ accepts whenever possible}}$$

$\sigma(w)$ ← $\big(\sigma(), \sigma(w_0), \sigma(w_0 w_1), \ldots\big)$

Synthesis (Henzinger, Piterman ['06])

$\varphi \rightsquigarrow \mathcal{A}_{\mathbf{GFG}}$ — GFG automaton

Good-For-Games automata

$\mathcal{A}_{\mathbf{non-det}}$ is Good-For-Games (GFG) if

$$\underbrace{\exists_{\sigma\colon \Sigma^*\to Q}}_{\text{advice}} \qquad \underbrace{\forall_{w\in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}})} \quad \sigma(w) \text{ is accepting}}_{\sigma \text{ accepts whenever possible}}$$

$\sigma(w)$: $\Big(\sigma(), \sigma(w_0), \sigma(w_0w_1), \ldots\Big)$

Synthesis (Henzinger, Piterman ['06])

$\varphi \rightsquigarrow \mathcal{A}_{\mathbf{GFG}}$ — GFG automaton

$\forall$: I_i

Good-For-Games automata

$\mathcal{A}_{\mathbf{non-det}}$ is Good-For-Games (GFG) if

$$\left(\sigma(), \sigma(w_0), \sigma(w_0 w_1), \ldots\right)$$

$$\underbrace{\exists_{\sigma\colon \Sigma^* \to Q}}_{\text{advice}} \qquad \underbrace{\forall_{w \in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}})} \quad \sigma(w) \text{ is accepting}}_{\sigma \text{ accepts whenever possible}}$$

Synthesis (Henzinger, Piterman ['06])

$\varphi \rightsquigarrow \mathcal{A}_{\mathbf{GFG}}$ — GFG automaton

$\forall$: I_i

$\exists$: O_i

Good-For-Games automata

$\mathcal{A}_{\mathbf{non-det}}$ is Good-For-Games (GFG) if

$\big(\sigma(), \sigma(w_0), \sigma(w_0 w_1), \ldots\big)$

$$\underbrace{\exists_{\sigma\colon \Sigma^* \to Q}}_{\text{advice}} \qquad \underbrace{\forall_{w \in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}})} \quad \sigma(w) \text{ is accepting}}_{\sigma \text{ accepts whenever possible}}$$

Synthesis (Henzinger, Piterman ['06])

$\varphi \rightsquigarrow \mathcal{A}_{\mathbf{GFG}}$ — GFG automaton

$\forall$: I_i

$\exists$: O_i

$\exists$: transition of $\mathcal{A}_{\mathbf{GFG}}$

Good-For-Games automata

$\mathcal{A}_{\mathbf{non-det}}$ is Good-For-Games (GFG) if

$$\underbrace{\exists_{\sigma\colon \Sigma^*\to Q}}_{\text{advice}} \quad \underbrace{\forall_{w\in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}})} \quad \sigma(w) \text{ is accepting}}_{\sigma \text{ accepts whenever possible}}$$

$\sigma(w)$ ↙ $\big(\sigma(), \sigma(w_0), \sigma(w_0w_1), \ldots\big)$

Synthesis (Henzinger, Piterman ['06])

$\varphi \rightsquigarrow \mathcal{A}_{\mathbf{GFG}}$ — GFG automaton

$\forall$: I_i

$\exists$: O_i

$\exists$: transition of $\mathcal{A}_{\mathbf{GFG}}$

History determinism (Colcombet, Löding ['08])

Good-For-Games automata

$\mathcal{A}_{\mathbf{non-det}}$ is Good-For-Games (GFG) if

$$\underbrace{\exists_{\sigma\colon \Sigma^* \to Q}}_{\text{advice}} \quad \underbrace{\forall_{w \in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}})} \quad \sigma(w) \text{ is accepting}}_{\sigma \text{ accepts whenever possible}}$$

where $\sigma(w)$ denotes $\big(\sigma(), \sigma(w_0), \sigma(w_0 w_1), \ldots\big)$

Synthesis (Henzinger, Piterman ['06])

$\varphi \rightsquigarrow \mathcal{A}_{\mathbf{GFG}}$ — GFG automaton

- $\forall$: I_i
- $\exists$: O_i
- $\exists$: transition of $\mathcal{A}_{\mathbf{GFG}}$

History determinism (Colcombet, Löding ['08])

[replaces determinism for counter automata]

Good-For-Games automata

$\mathcal{A}_{\mathbf{non-det}}$ is Good-For-Games (GFG) if

$$\underbrace{\exists_{\sigma\colon \Sigma^*\to Q}}_{\text{advice}} \qquad \underbrace{\forall_{w\in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}})} \quad \sigma(w) \text{ is accepting}}_{\sigma \text{ accepts whenever possible}}$$

$\sigma(w)$: $\big(\sigma(), \sigma(w_0), \sigma(w_0 w_1), \ldots\big)$

Synthesis (Henzinger, Piterman ['06])

$\varphi \rightsquigarrow \mathcal{A}_{\mathbf{GFG}}$ — GFG automaton

$\forall$: I_i

$\exists$: O_i

$\exists$: transition of $\mathcal{A}_{\mathbf{GFG}}$

History determinism (Colcombet, Löding ['08])

[replaces determinism for counter automata]

Theorem (Boker, Kuperberg, Kupferman, S. ['13])

GFG $\equiv$ Good-For-Trees (derived languages $\forall\textsc{path}(\mathrm{L})$)

Good-For-Games automata

$\mathcal{A}_{\mathbf{non-det}}$ is Good-For-Games (GFG) if

$\big(\sigma(), \sigma(w_0), \sigma(w_0 w_1), \ldots\big)$

$$\underbrace{\exists_{\sigma : \Sigma^* \to Q}}_{\text{advice}} \qquad \underbrace{\forall_{w \in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}})} \quad \sigma(w) \text{ is accepting}}_{\sigma \text{ accepts whenever possible}}$$

Synthesis (Henzinger, Piterman ['06])

$\varphi \rightsquigarrow \mathcal{A}_{\mathbf{GFG}}$ — GFG automaton

$\forall$: I_i

$\exists$: O_i

$\exists$: transition of $\mathcal{A}_{\mathbf{GFG}}$

What

is the blow-up of

$\mathcal{A}_{\mathbf{GFG}} \rightsquigarrow \mathcal{A}_{\mathbf{det}}$

?

History determinism (Colcombet, Löding ['08])

[replaces determinism for counter automata]

Theorem (Boker, Kuperberg, Kupferman, S. ['13])

GFG $\equiv$ Good-For-Trees (derived languages $\forall$PATH(L))

Blow-up of $\mathcal{A}_{\mathbf{GFG}} \rightsquigarrow \mathcal{A}_{\mathbf{det}}$

Blow-up of $\mathcal{A}_{\mathbf{GFG}} \rightsquigarrow \mathcal{A}_{\mathbf{det}}$

Theorem (Löding ['11])

Every GFG automaton over finite words contains an equivalent deterministic subautomaton.

Blow-up of $\mathcal{A}_{\mathbf{GFG}} \rightsquigarrow \mathcal{A}_{\mathbf{det}}$

Theorem (Löding ['11])

Every GFG automaton over finite words contains an equivalent deterministic subautomaton.

$\rightsquigarrow$ no blow-up over finite words

Blow-up of $\mathcal{A}_{\mathbf{GFG}} \rightsquigarrow \mathcal{A}_{\mathbf{det}}$

Theorem (Löding ['11])

Every GFG automaton over finite words contains an equivalent deterministic subautomaton.

$\rightsquigarrow$ no blow-up over finite words

Conjecture (Colcombet ['12])

The same holds for ω-words (parity automata).

Blow-up of $\mathcal{A}_{\mathbf{GFG}} \rightsquigarrow \mathcal{A}_{\mathbf{det}}$

Theorem (Löding ['11])

Every GFG automaton over finite words contains an equivalent deterministic subautomaton.

$\rightsquigarrow$ no blow-up over finite words

Conjecture (Colcombet ['12])

The same holds for ω-words (parity automata).

Theorem (Boker ['13])

There exists a Büchi GFG automaton with **no** equivalent deterministic subautomaton.

Blow-up of $\mathcal{A}_{\mathbf{GFG}} \rightsquigarrow \mathcal{A}_{\mathbf{det}}$

Theorem (Löding ['11])

Every GFG automaton over finite words contains an equivalent deterministic subautomaton.

$\rightsquigarrow$ no blow-up over finite words

Conjecture (Colcombet ['12])

The same holds for ω-words (parity automata).

Theorem (Boker ['13])

There exists a Büchi GFG automaton with **no** equivalent deterministic subautomaton.

Question (Kupferman et al. ['13])

Does every GFG automaton admit polynomial determinisation?

Blow-up of $\mathcal{A}_{\mathbf{GFG}} \rightsquigarrow \mathcal{A}_{\mathbf{det}}$

Theorem (Löding ['11])

Every GFG automaton over finite words contains an equivalent deterministic subautomaton.

$\rightsquigarrow$ no blow-up over finite words

Conjecture (Colcombet ['12])

The same holds for ω-words (parity automata).

Theorem (Boker ['13])

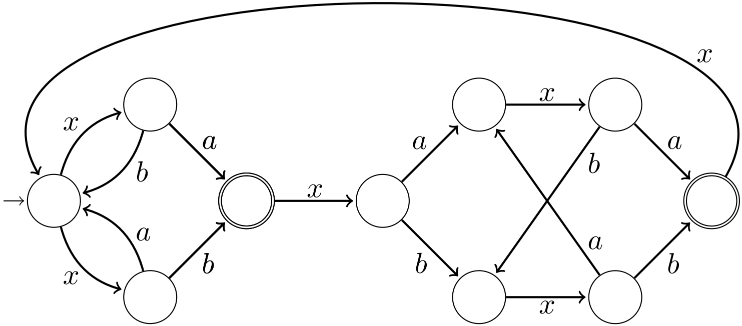
There exists a Büchi GFG automaton with **no** equivalent deterministic subautomaton.

Question (Kupferman et al. ['13])

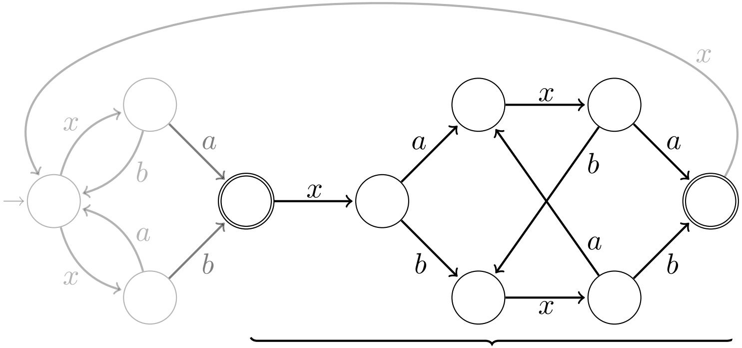
Does every GFG automaton admit polynomial determinisation?

Boker's example — Büchi GFG automaton w.o. det. subautomaton

Boker's example — Büchi GFG automaton w.o. det. subautomaton



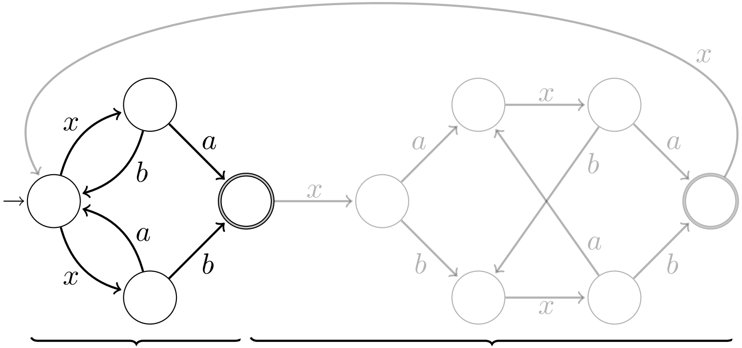
Boker's example — Büchi GFG automaton w.o. det. subautomaton



$$\mathrm{L} = \big(x(a,b)\big)^* \cdot \big(xaxa, xbxb\big)$$

deterministic

Boker's example — Büchi GFG automaton w.o. det. subautomaton



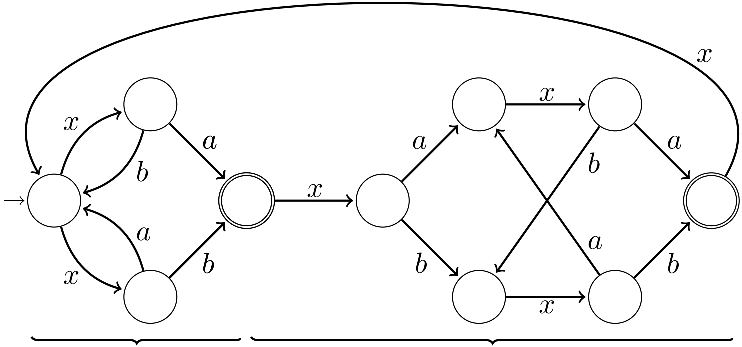
$\mathrm{L} = \big(x(a,b)\big)^*$

non-det.

$\mathrm{L} = \big(x(a,b)\big)^* \cdot \big(xaxa, xbxb\big)$

deterministic

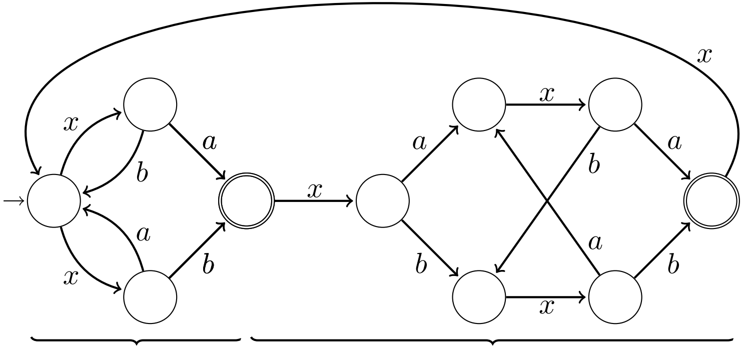
Boker's example — Büchi GFG automaton w.o. det. subautomaton



$L = \big(x(a,b)\big)^*$ $\qquad$ $L = \big(x(a,b)\big)^* \cdot \big(xaxa, xbxb\big)$

$L = \big(x(a,b)\big)^\omega \cap$ "infinitely many $xaxa$ or $xbxb$"

Boker's example — Büchi GFG automaton w.o. det. subautomaton

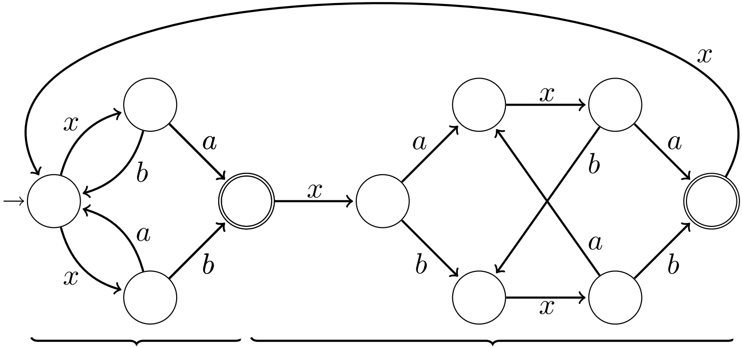


$\mathrm{L} = \big(x(a,b)\big)^*$ $\qquad$ $\mathrm{L} = \big(x(a,b)\big)^* \cdot \big(xaxa, xbxb\big)$

$\mathrm{L} = \big(x(a,b)\big)^\omega \cap$ "infinitely many $xaxa$ or $xbxb$"

$\sigma =$"guess that the last a/b will reappear"

Boker's example — Büchi GFG automaton w.o. det. subautomaton



$L = \left(x(a,b)\right)^*$ $\quad$ $L = \left(x(a,b)\right)^* \cdot \left(xaxa, xbxb\right)$

$L = \left(x(a,b)\right)^\omega \cap$ "infinitely many $xaxa$ or $xbxb$"

$\sigma =$"guess that the last a/b will reappear"

[two memory states]

Determinisation vs. memory

Determinisation vs. memory

When $\mathcal{A}_{\mathbf{non-det}}$ is GFG?

Determinisation vs. memory

When $\mathcal{A}_{\mathbf{non-det}}$ is GFG? $\left[\exists_{\sigma\colon \Sigma^{*}\to Q}\ \ \forall_{w\in \mathrm{L}(\mathcal{A})}\ \ \sigma(w) \text{ is accepting}\right]$

Determinisation vs. memory

When $\mathcal{A}_{\mathbf{non-det}}$ is GFG? $\left[\exists_{\sigma\colon \Sigma^{*}\to Q} \ \forall_{w\in \mathrm{L}(\mathcal{A})} \ \sigma(w) \text{ is accepting}\right]$

GFG game

Determinisation vs. memory

When $\mathcal{A}_{\mathbf{non-det}}$ is GFG? $\left[\exists_{\sigma\colon \Sigma^*\to Q}\ \ \forall_{w\in \mathrm{L}(\mathcal{A})}\ \ \sigma(w) \text{ is accepting}\right]$

GFG game

$\forall$: a_i

Determinisation vs. memory

When $\mathcal{A}_{\mathbf{non-det}}$ is GFG? $\left[\exists_{\sigma\colon \Sigma^* \to Q} \;\; \forall_{w \in \mathrm{L}(\mathcal{A})} \;\; \sigma(w) \text{ is accepting}\right]$

GFG game

$\forall$: a_i

$\exists$: q_i

Determinisation vs. memory

When $\mathcal{A}_{\mathbf{non-det}}$ is GFG? $\left[\exists_{\sigma\colon \Sigma^* \to Q} \ \forall_{w \in \mathrm{L}(\mathcal{A})} \ \sigma(w) \text{ is accepting}\right]$

GFG game

$\forall$: a_i

$\exists$: q_i

$\exists$ wins if $\begin{pmatrix} (a_0, a_1, \ldots) \in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}}) \\ \Downarrow \\ (q_0, q_1, \ldots) \text{ is accepting} \end{pmatrix}$

Determinisation vs. memory

When $\mathcal{A}_{\mathbf{non-det}}$ is GFG? $\left[\exists_{\sigma : \Sigma^* \to Q} \ \forall_{w \in \mathrm{L}(\mathcal{A})} \ \sigma(w) \text{ is accepting}\right]$

GFG game

$\forall$: a_i

$\exists$: q_i

$$\exists \text{ wins if } \begin{pmatrix} (a_0, a_1, \ldots) \in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}}) \\ \Downarrow \\ (q_0, q_1, \ldots) \text{ is accepting} \end{pmatrix}$$

Theorem (Büchi, Landweber ['69])

ω-regular games are finite memory determined.

Determinisation vs. memory

When $\mathcal{A}_{\mathbf{non-det}}$ is GFG? $\left[\exists_{\sigma\colon \Sigma^* \to Q} \ \forall_{w \in \mathrm{L}(\mathcal{A})} \ \sigma(w) \text{ is accepting}\right]$

GFG game

$\forall$: a_i

$\exists$: q_i

$\exists$ wins if $\begin{pmatrix} (a_0, a_1, \ldots) \in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}}) \\ \Downarrow \\ (q_0, q_1, \ldots) \text{ is accepting} \end{pmatrix}$

Theorem (Büchi, Landweber ['69])

ω-regular games are finite memory determined.

$\rightsquigarrow$ effective (EXPTIME) characterisation of GFG automata

Determinisation vs. memory

When $\mathcal{A}_{\mathbf{non-det}}$ is GFG? $\left[\exists_{\sigma\colon \Sigma^* \to Q}\ \ \forall_{w \in \mathrm{L}(\mathcal{A})}\ \ \sigma(w) \text{ is accepting}\right]$

GFG game

$\forall$: a_i

$\exists$: q_i

$\exists$ wins if $\left(\begin{array}{c}(a_0, a_1, \ldots) \in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}}) \\ \Downarrow \\ (q_0, q_1, \ldots) \text{ is accepting}\end{array}\right)$

Theorem (Büchi, Landweber ['69])

ω-regular games are finite memory determined.

⇝ effective (EXPTime) characterisation of GFG automata

Proposition (Boker, Kuperberg, Kupferman, S. ['13])

If M is a winning memory structure of $\exists$ then

Determinisation vs. memory

When $\mathcal{A}_{\mathbf{non-det}}$ is GFG? $\left[\exists_{\sigma\colon \Sigma^*\to Q} \ \forall_{w\in \mathrm{L}(\mathcal{A})} \ \sigma(w) \text{ is accepting}\right]$

GFG game

$\forall$: a_i

$\exists$: q_i

$$\exists \text{ wins if } \begin{pmatrix} (a_0, a_1, \ldots) \in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}}) \\ \Downarrow \\ (q_0, q_1, \ldots) \text{ is accepting} \end{pmatrix}$$

Theorem (Büchi, Landweber ['69])

ω-regular games are finite memory determined.

$\rightsquigarrow$ effective (EXPTIME) characterisation of GFG automata

Proposition (Boker, Kuperberg, Kupferman, S. ['13])

If M is a winning memory structure of $\exists$ then

$$|M| \leqslant |\mathcal{A}_{\mathbf{det}}| \leqslant |M \times \mathcal{A}_{\mathbf{GFG}}|$$

Determinisation vs. memory

When $\mathcal{A}_{\mathbf{non-det}}$ is GFG? $\left[\exists_{\sigma : \Sigma^* \to Q} \ \forall_{w \in \mathrm{L}(\mathcal{A})} \ \sigma(w) \text{ is accepting}\right]$

GFG game

$\forall$: a_i

$\exists$: q_i

$\exists$ wins if $\begin{pmatrix} (a_0, a_1, \ldots) \in \mathrm{L}(\mathcal{A}_{\mathbf{non-det}}) \\ \Downarrow \\ (q_0, q_1, \ldots) \text{ is accepting} \end{pmatrix}$

Theorem (Büchi, Landweber ['69])

ω-regular games are finite memory determined.

$\rightsquigarrow$ effective (EXPTIME) characterisation of GFG automata

Proposition (Boker, Kuperberg, Kupferman, S. ['13])

If M is a winning memory structure of $\exists$ then

$$|M| \leqslant |\mathcal{A}_{\mathbf{det}}| \leqslant |M \times \mathcal{A}_{\mathbf{GFG}}|$$

i.e. $|\mathcal{A}_{\mathbf{det}}| \sim |M|$

Co-Büchi case

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

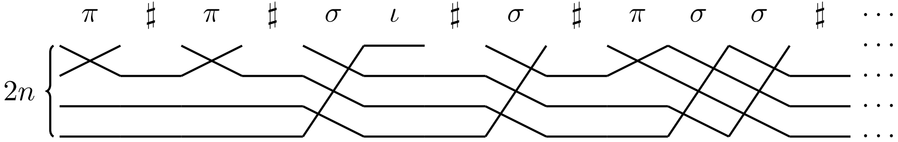
Plaits

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Plaits

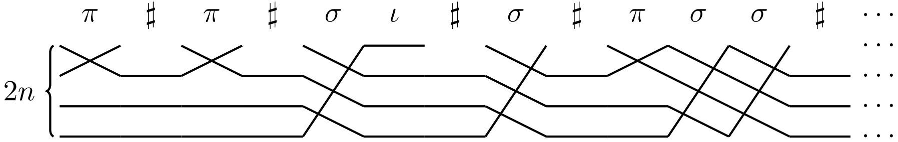


Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Plaits



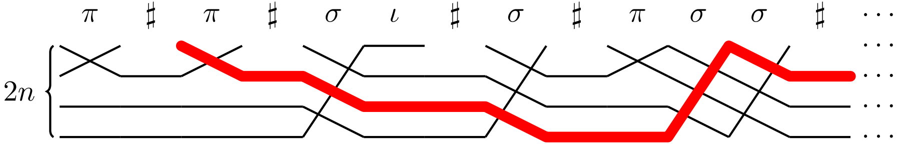
L ="exists some infinite path"

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $\left|\mathcal{A}_{\mathbf{det}}\right| \sim 2^{\left|\mathcal{A}_{\mathbf{GFG}}\right|}$

Plaits



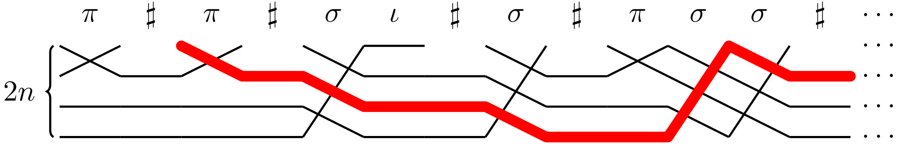
L =“exists some infinite path”

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Plaits



L =“exists some infinite path”

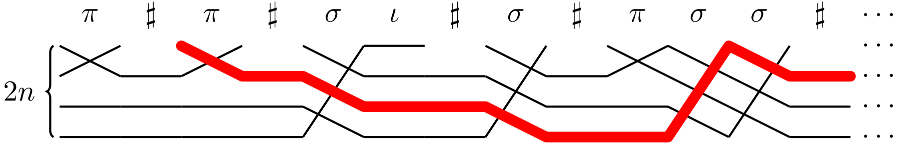
$\mathcal{A}_{\mathbf{GFG}}$: guess a path and follow

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $\left|\mathcal{A}_{\mathbf{det}}\right| \sim 2^{\left|\mathcal{A}_{\mathbf{GFG}}\right|}$

Plaits



L =“exists some infinite path”

$\mathcal{A}_{\mathbf{GFG}}$: guess a path and follow

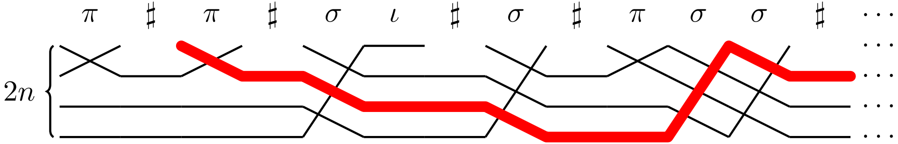
when reached a dead end, guess a new path (via a rejecting transition)

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Plaits



L ="exists some infinite path"

$\mathcal{A}_{\mathbf{GFG}}$: guess a path and follow

when reached a dead end, guess a new path (via a rejecting transition)

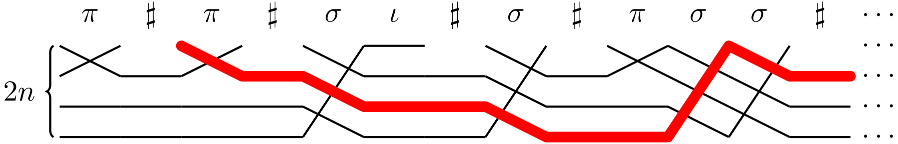
$|\mathcal{A}_{\mathbf{GFG}}| \sim 2n$

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Plaits



L ="exists some infinite path"

$\mathcal{A}_{\mathbf{GFG}}$: guess a path and follow

when reached a dead end, guess a new path (via a rejecting transition)

$|\mathcal{A}_{\mathbf{GFG}}| \sim 2n$

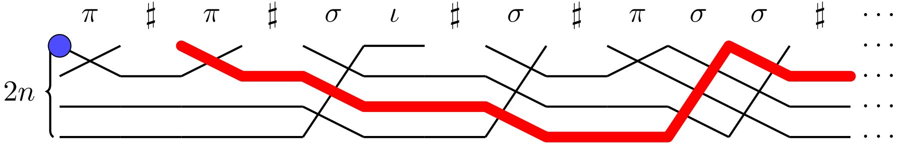
σ: try the oldest path available

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Plaits



L = "exists some infinite path"

$\mathcal{A}_{\mathbf{GFG}}$: guess a path and follow

when reached a dead end, guess a new path (via a rejecting transition)

$|\mathcal{A}_{\mathbf{GFG}}| \sim 2n$

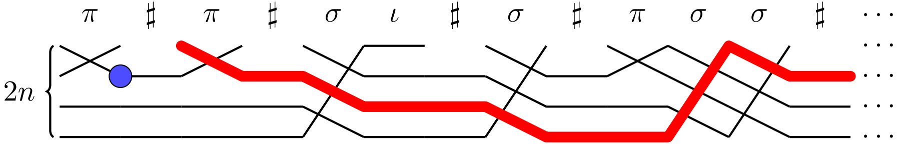
σ: try the oldest path available

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Plaits



L = "exists some infinite path"

$\mathcal{A}_{\mathbf{GFG}}$: guess a path and follow

when reached a dead end, guess a new path (via a rejecting transition)

$|\mathcal{A}_{\mathbf{GFG}}| \sim 2n$

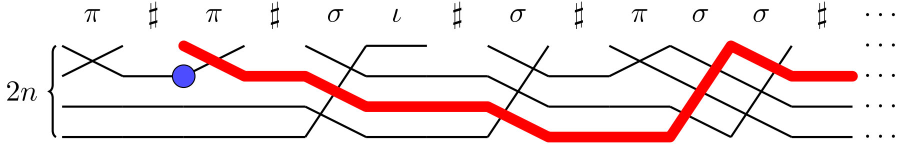
σ: try the oldest path available

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Plaits



L ="exists some infinite path"

$\mathcal{A}_{\mathbf{GFG}}$: guess a path and follow

when reached a dead end, guess a new path (via a rejecting transition)

$|\mathcal{A}_{\mathbf{GFG}}| \sim 2n$

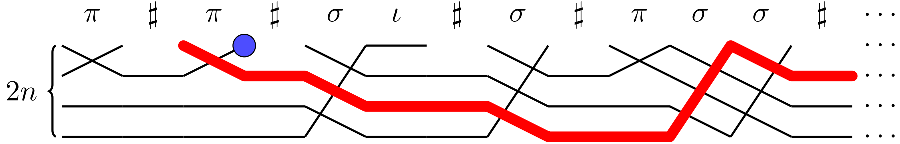
σ: try the oldest path available

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Plaits



L = "exists some infinite path"

$\mathcal{A}_{\mathbf{GFG}}$: guess a path and follow

when reached a dead end, guess a new path (via a rejecting transition)

$|\mathcal{A}_{\mathbf{GFG}}| \sim 2n$

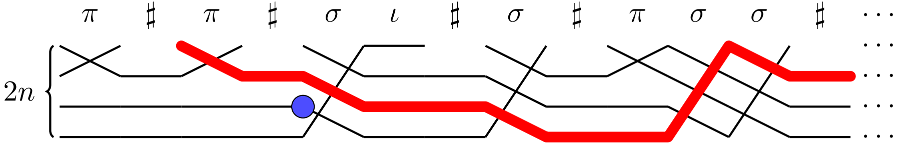
σ: try the oldest path available

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Plaits



L =“exists some infinite path”

$\mathcal{A}_{\mathbf{GFG}}$: guess a path and follow

when reached a dead end, guess a new path (via a rejecting transition)

$|\mathcal{A}_{\mathbf{GFG}}| \sim 2n$

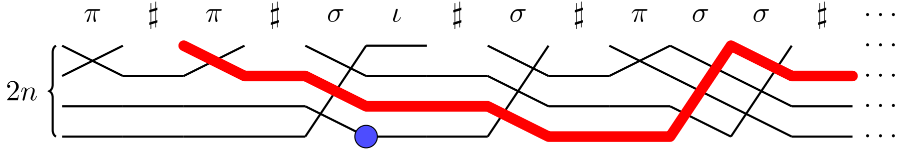
σ: try the oldest path available

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Plaits



L = "exists some infinite path"

$\mathcal{A}_{\mathbf{GFG}}$: guess a path and follow

when reached a dead end, guess a new path (via a rejecting transition)

$|\mathcal{A}_{\mathbf{GFG}}| \sim 2n$

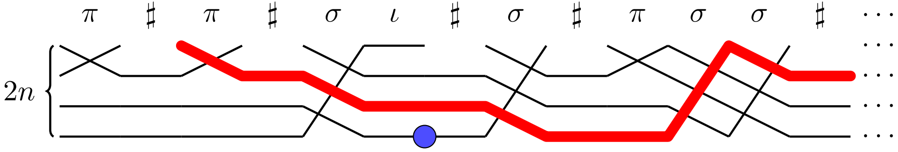
σ: try the oldest path available

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Plaits



$\mathrm{L} =$ "exists some infinite path"

$\mathcal{A}_{\mathbf{GFG}}$: guess a path and follow

when reached a dead end, guess a new path (via a rejecting transition)

$|\mathcal{A}_{\mathbf{GFG}}| \sim 2n$

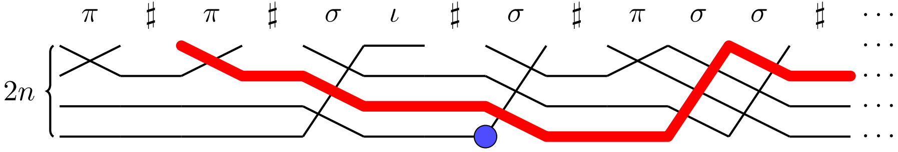
σ: try the oldest path available

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Plaits



L = "exists some infinite path"

$\mathcal{A}_{\mathbf{GFG}}$: guess a path and follow

when reached a dead end, guess a new path (via a rejecting transition)

$|\mathcal{A}_{\mathbf{GFG}}| \sim 2n$

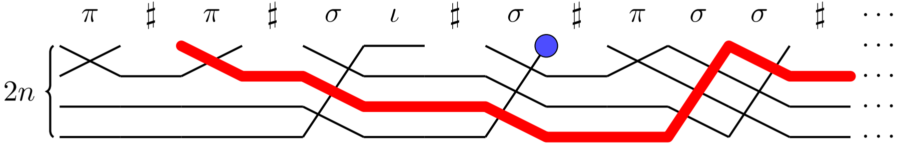
σ: try the oldest path available

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Plaits



L ="exists some infinite path"

$\mathcal{A}_{\mathbf{GFG}}$: guess a path and follow

when reached a dead end, guess a new path (via a rejecting transition)

$|\mathcal{A}_{\mathbf{GFG}}| \sim 2n$

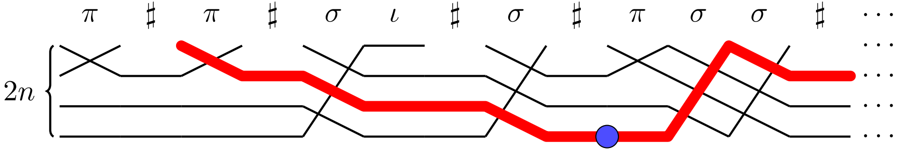
σ: try the oldest path available

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Plaits



L =“exists some infinite path”

$\mathcal{A}_{\mathbf{GFG}}$: guess a path and follow

when reached a dead end, guess a new path (via a rejecting transition)

$|\mathcal{A}_{\mathbf{GFG}}| \sim 2n$

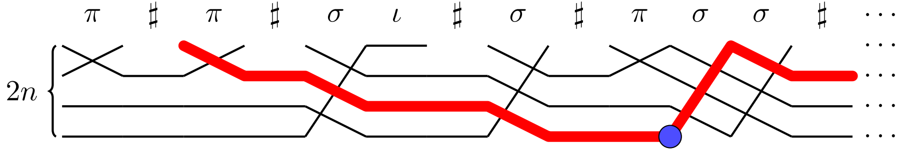
σ: try the oldest path available

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Plaits



L ="exists some infinite path"

$\mathcal{A}_{\mathbf{GFG}}$: guess a path and follow

when reached a dead end, guess a new path (via a rejecting transition)

$|\mathcal{A}_{\mathbf{GFG}}| \sim 2n$

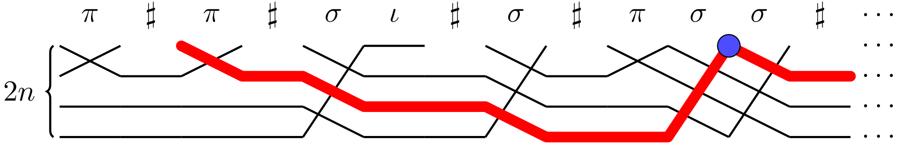
σ: try the oldest path available

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Plaits



L = "exists some infinite path"

$\mathcal{A}_{\mathbf{GFG}}$: guess a path and follow

when reached a dead end, guess a new path (via a rejecting transition)

$|\mathcal{A}_{\mathbf{GFG}}| \sim 2n$

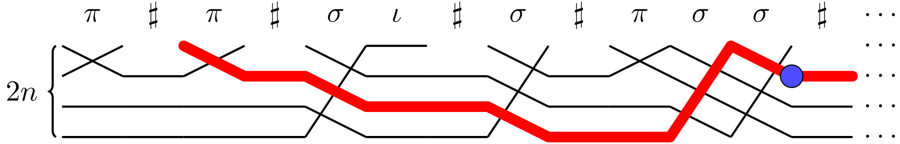
σ: try the oldest path available

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Plaits



$\mathrm{L} =$"exists some infinite path"

$\mathcal{A}_{\mathbf{GFG}}$: guess a path and follow

when reached a dead end, guess a new path (via a rejecting transition)

$|\mathcal{A}_{\mathbf{GFG}}| \sim 2n$

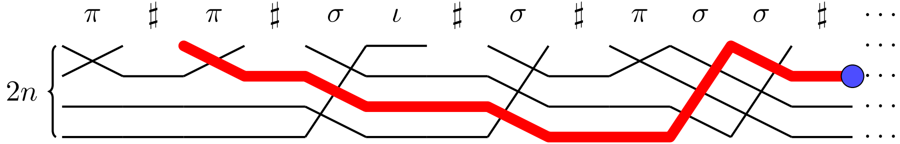
σ: try the oldest path available

Co-Büchi case

Theorem (S. ['13])

For co-Büchi automata $|\mathcal{A}_{\mathbf{det}}| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|}$

Plaits



L =“exists some infinite path”

$\mathcal{A}_{\mathbf{GFG}}$: guess a path and follow

when reached a dead end, guess a new path (via a rejecting transition)

$|\mathcal{A}_{\mathbf{GFG}}| \sim 2n$

σ: try the oldest path available

Exponential blow-up

Exponential blow-up

Assume M with $|M| < 2^n/2n$

Exponential blow-up

Assume M with $|M| < 2^n/2n$ $\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Exponential blow-up

$\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$

$\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Exponential blow-up

$\big[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\big]$

Assume M with $|M| < 2^n/2n$

$\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$

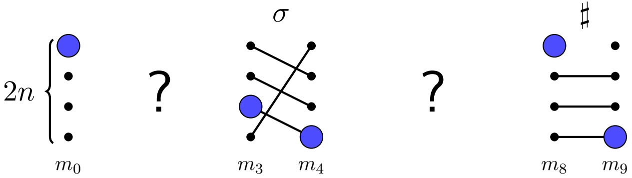
Exponential blow-up

$\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$

$\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



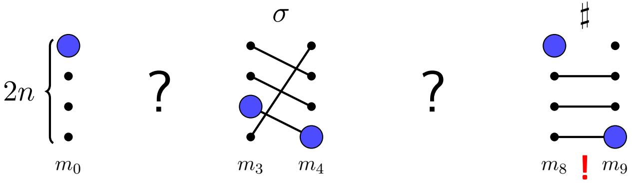
Exponential blow-up

$\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$

$\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



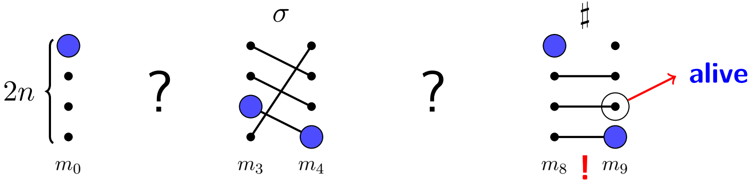
Exponential blow-up

$\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$

$\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$

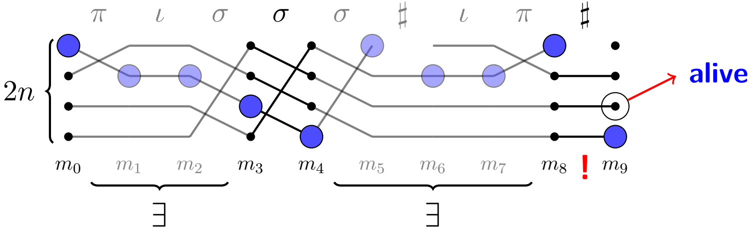


Exponential blow-up

$\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$ $\qquad$ $\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$

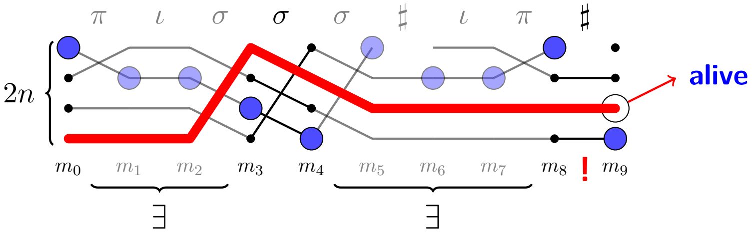


Exponential blow-up

$\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$ $\qquad$ $\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



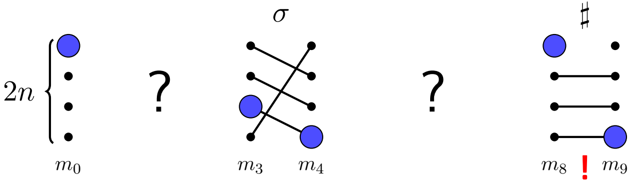
Exponential blow-up

$\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$

$\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



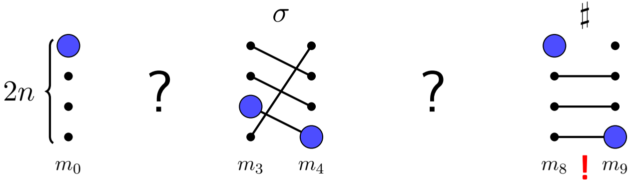
Aim: construct rejecting partial runs

Exponential blow-up

$\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$ $\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



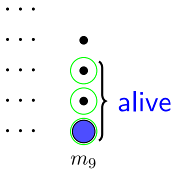
Aim: construct rejecting partial runs with $>n$ alive values

Exponential blow-up

$\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$ $\qquad$ $\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



Aim: construct rejecting partial runs with $>n$ alive values

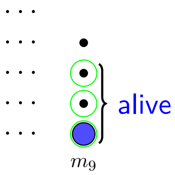
Exponential blow-up

$[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n]$

Assume M with $|M| < 2^n/2n$

$\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



Aim: construct rejecting partial runs with $>n$ alive values

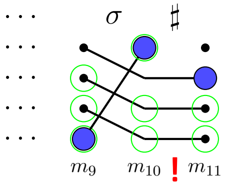
1. force a rejecting transition (using $\sharp$)

Exponential blow-up

$\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$ $\qquad$ $\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



Aim: construct rejecting partial runs with $>n$ alive values

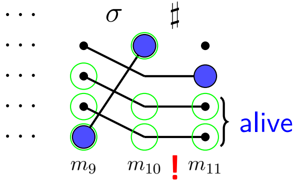
1. force a rejecting transition (using $\sharp$)

Exponential blow-up

$\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$ $\qquad$ $\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



Aim: construct rejecting partial runs with $>n$ alive values

1. force a rejecting transition (using $\sharp$)

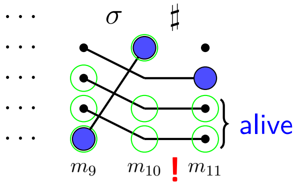
Exponential blow-up

$\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$

$\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



Aim: construct rejecting partial runs with $>n$ alive values

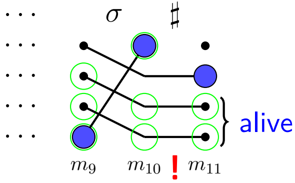
1. force a rejecting transition (using $\sharp$)
2. increase the number of alive values $n \mapsto n{+}1$

Exponential blow-up

$\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$ $\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



Aim: construct rejecting partial runs with $>n$ alive values

1. force a rejecting transition (using $\sharp$)

2. increase the number of alive values $n \mapsto n+1$

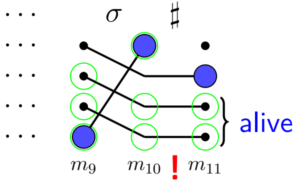
- consider 2^n permutations distinct on alive values

Exponential blow-up

$\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$ $\quad$ $\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



Aim: construct rejecting partial runs with $>n$ alive values

1. force a rejecting transition (using ♯)

2. increase the number of alive values $n \mapsto n+1$

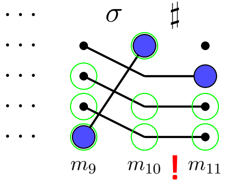
- consider 2^n permutations distinct on alive values
- $\pi \neq \pi'$ are indistinguishable by $\mathcal{A}_{\mathbf{GFG}} \times M$

Exponential blow-up

$\big[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\big]$

Assume M with $|M| < 2^n/2n$ $\qquad$ $\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



π/π'

Aim: construct rejecting partial runs with $>n$ alive values

1. force a rejecting transition (using ♯)

2. increase the number of alive values $n \mapsto n+1$

- consider 2^n permutations distinct on alive values
- $\pi \neq \pi'$ are indistinguishable by $\mathcal{A}_{\mathbf{GFG}} \times M$

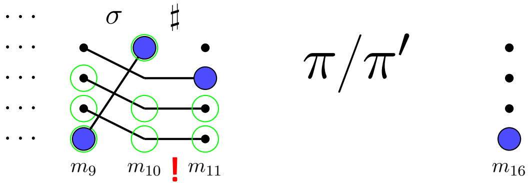
Exponential blow-up

$\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$

$\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



Aim: construct rejecting partial runs with $>n$ alive values

1. force a rejecting transition (using ♯)

2. increase the number of alive values $n \mapsto n+1$

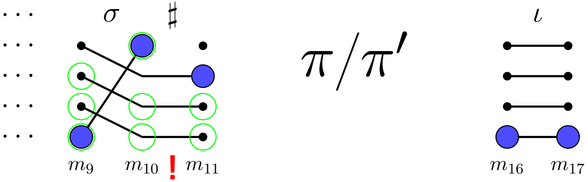
- consider 2^n permutations distinct on alive values
- $\pi \neq \pi'$ are indistinguishable by $\mathcal{A}_{\mathbf{GFG}} \times M$

Exponential blow-up

$\big[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\big]$

Assume M with $|M| < 2^n/2n$ $\qquad$ $\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



Aim: construct rejecting partial runs with $>n$ alive values

1. force a rejecting transition (using $\sharp$)

2. increase the number of alive values $n \mapsto n+1$

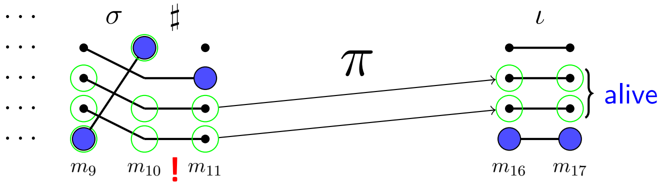
- consider 2^n permutations distinct on alive values
- $\pi \neq \pi'$ are indistinguishable by $\mathcal{A}_{\mathbf{GFG}} \times M$

Exponential blow-up

$\big[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\big]$

Assume M with $|M| < 2^n/2n$ $\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



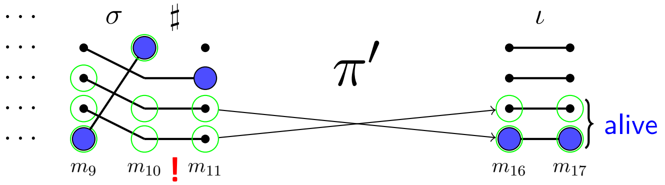
Aim: construct rejecting partial runs with $>n$ alive values

1. force a rejecting transition (using $\sharp$)
2. increase the number of alive values $n \mapsto n+1$
   - consider 2^n permutations distinct on alive values
   - $\pi \neq \pi'$ are indistinguishable by $\mathcal{A}_{\mathbf{GFG}} \times M$

Exponential blow-up $\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$ $\qquad$ $\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



Aim: construct rejecting partial runs with $>n$ alive values

1. force a rejecting transition (using ♯)

2. increase the number of alive values $n \mapsto n+1$

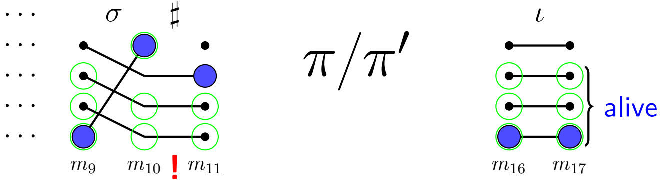
- consider 2^n permutations distinct on alive values
- $\pi \neq \pi'$ are indistinguishable by $\mathcal{A}_{\mathbf{GFG}} \times M$

Exponential blow-up

$\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$ $\qquad$ $\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



Aim: construct rejecting partial runs with $>n$ alive values

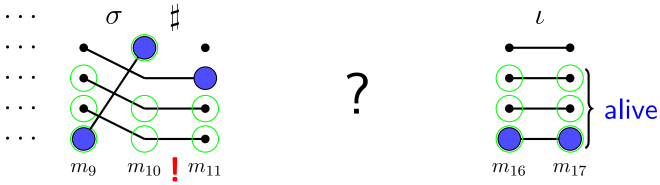
1. force a rejecting transition (using $\sharp$)
2. increase the number of alive values $n \mapsto n+1$
   - consider 2^n permutations distinct on alive values
   - $\pi \neq \pi'$ are indistinguishable by $\mathcal{A}_{\mathbf{GFG}} \times M$

Exponential blow-up

$\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$ $\qquad$ $\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



Aim: construct rejecting partial runs with $>n$ alive values

1. force a rejecting transition (using ♯)

2. increase the number of alive values $n \mapsto n+1$

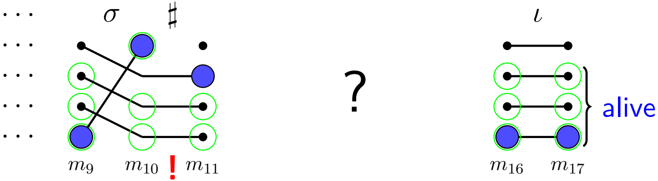
- consider 2^n permutations distinct on alive values
- $\pi \neq \pi'$ are indistinguishable by $\mathcal{A}_{\mathbf{GFG}} \times M$

Exponential blow-up

$\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$ $\qquad$ $\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$



Aim: construct rejecting partial runs with $>n$ alive values

1. force a rejecting transition (using $\sharp$)

2. increase the number of alive values $n \mapsto n+1$

- consider 2^n permutations distinct on alive values
- $\pi \neq \pi'$ are indistinguishable by $\mathcal{A}_{\mathbf{GFG}} \times M$

Finish: use compactness argument

Exponential blow-up

$\left[|\mathcal{A}_{\mathbf{GFG}} \times M| < 2^n\right]$

Assume M with $|M| < 2^n/2n$ $\mathcal{A}_{\mathbf{GFG}} \times M$ — det. aut. for L

Partial runs of $\mathcal{A}_{\mathbf{GFG}} \times M$

Aim: construct rejecting partial runs with $>n$ alive values

1. force a rejecting transition (using $\sharp$)
2. increase the number of alive values $n \mapsto n+1$
   - consider 2^n permutations distinct on alive values
   - $\pi \neq \pi'$ are indistinguishable by $\mathcal{A}_{\mathbf{GFG}} \times M$

Finish: use compactness argument

$\rightsquigarrow w \in L$ s.t. $\mathcal{A}_{\mathbf{GFG}} \times M$ rejects w

Büchi case

Büchi case

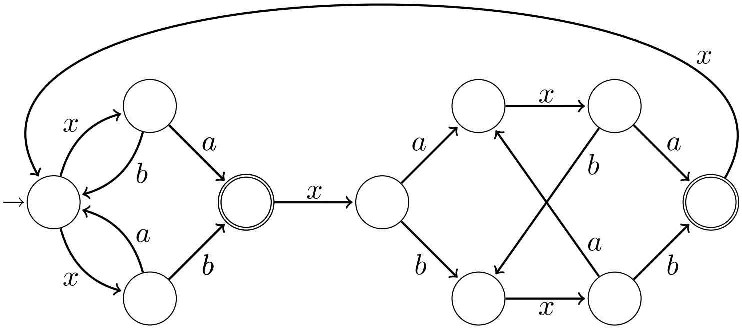
Theorem (Kuperberg, S. ['14])

For Büchi automata $|\mathcal{A}_{\mathbf{det}}| \leqslant |\mathcal{A}_{\mathbf{GFG}}|^2$

Büchi case

Theorem (Kuperberg, S. ['14])

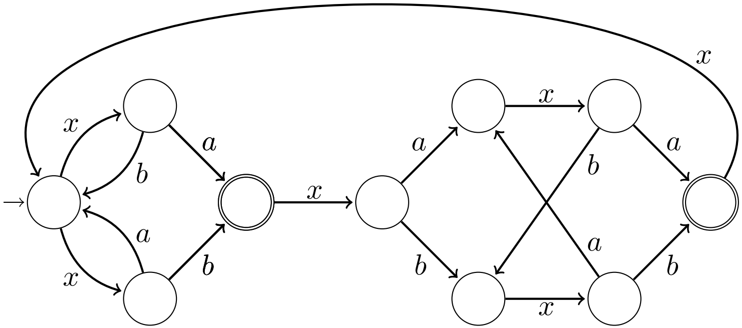
For Büchi automata $|\mathcal{A}_{\mathbf{det}}| \leqslant |\mathcal{A}_{\mathbf{GFG}}|^2$



Büchi case

Theorem (Kuperberg, S. ['14])

For Büchi automata $|\mathcal{A}_{\mathbf{det}}| \leqslant |\mathcal{A}_{\mathbf{GFG}}|^2$



Where the strategy comes from?

Büchi case

Theorem (Kuperberg, S. ['14])

For Büchi automata $\left|\mathcal{A}_{\mathbf{det}}\right| \leqslant \left|\mathcal{A}_{\mathbf{GFG}}\right|^2$

1. brutal determinisation of $\mathcal{A}_{\mathbf{GFG}} \rightsquigarrow D \quad (|D| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|})$

Büchi case

Theorem (Kuperberg, S. ['14])

For Büchi automata $|\mathcal{A}_{\mathbf{det}}| \leqslant |\mathcal{A}_{\mathbf{GFG}}|^2$

1. brutal determinisation of $\mathcal{A}_{\mathbf{GFG}} \rightsquigarrow D \quad (|D| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|})$

2. parity game on $\mathcal{A}_{\mathbf{GFG}} \times D$

Büchi case

Theorem (Kuperberg, S. ['14])

For Büchi automata $|\mathcal{A}_{\mathbf{det}}| \leqslant |\mathcal{A}_{\mathbf{GFG}}|^2$

1. brutal determinisation of $\mathcal{A}_{\mathbf{GFG}} \rightsquigarrow D \quad (|D| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|})$

2. parity game on $\mathcal{A}_{\mathbf{GFG}} \times D$

3. rank-optimal winning strategy $\sigma_\exists$

Büchi case

Theorem (Kuperberg, S. ['14])

For Büchi automata $\left|\mathcal{A}_{\mathbf{det}}\right| \leqslant \left|\mathcal{A}_{\mathbf{GFG}}\right|^2$

1. brutal determinisation of $\mathcal{A}_{\mathbf{GFG}} \rightsquigarrow D \quad (|D| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|})$

2. parity game on $\mathcal{A}_{\mathbf{GFG}} \times D$

3. rank-optimal winning strategy $\sigma_{\exists}$ $\quad$ [signatures (Walukiewicz ['02])]

Büchi case

Theorem (Kuperberg, S. ['14])

For Büchi automata $|\mathcal{A}_{\mathbf{det}}| \leqslant |\mathcal{A}_{\mathbf{GFG}}|^2$

1. brutal determinisation of $\mathcal{A}_{\mathbf{GFG}} \rightsquigarrow D \quad (|D| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|})$

2. parity game on $\mathcal{A}_{\mathbf{GFG}} \times D$

3. rank-optimal winning strategy $\sigma_\exists$ [signatures (Walukiewicz ['02])]

4. inductive normalisation of $\mathcal{A}_{\mathbf{GFG}}$ w.r.t $\sigma_\exists$

Büchi case

Theorem (Kuperberg, S. ['14])

For Büchi automata $|\mathcal{A}_{\mathbf{det}}| \leqslant |\mathcal{A}_{\mathbf{GFG}}|^2$

1. brutal determinisation of $\mathcal{A}_{\mathbf{GFG}} \rightsquigarrow D \quad (|D| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|})$
2. parity game on $\mathcal{A}_{\mathbf{GFG}} \times D$
3. rank-optimal winning strategy $\sigma_\exists$ [signatures (Walukiewicz ['02])]
4. inductive normalisation of $\mathcal{A}_{\mathbf{GFG}}$ w.r.t $\sigma_\exists$
5. monotonicity of ranks

Büchi case

Theorem (Kuperberg, S. ['14])

For Büchi automata $\left|\mathcal{A}_{\mathbf{det}}\right| \leqslant \left|\mathcal{A}_{\mathbf{GFG}}\right|^2$

1. brutal determinisation of $\mathcal{A}_{\mathbf{GFG}} \rightsquigarrow D \quad (|D| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|})$

2. parity game on $\mathcal{A}_{\mathbf{GFG}} \times D$

3. rank-optimal winning strategy $\sigma_\exists$ $\big[$signatures (Walukiewicz ['02])$\big]$

4. inductive normalisation of $\mathcal{A}_{\mathbf{GFG}}$ w.r.t $\sigma_\exists$

5. monotonicity of ranks

6. it suffices to use $\mathcal{A}_{\mathbf{GFG}} \subseteq D$ instead of D

Büchi case

Theorem (Kuperberg, S. ['14])

For Büchi automata $\left|\mathcal{A}_{\mathbf{det}}\right| \leqslant \left|\mathcal{A}_{\mathbf{GFG}}\right|^2$

1. brutal determinisation of $\mathcal{A}_{\mathbf{GFG}} \rightsquigarrow D \quad (|D| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|})$
2. parity game on $\mathcal{A}_{\mathbf{GFG}} \times D$
3. rank-optimal winning strategy $\sigma_\exists$ $\quad$ [signatures (Walukiewicz ['02])]
4. inductive normalisation of $\mathcal{A}_{\mathbf{GFG}}$ w.r.t $\sigma_\exists$
5. monotonicity of ranks
6. it suffices to use $\mathcal{A}_{\mathbf{GFG}} \subseteq D$ instead of D
7. winning strategy on $\mathcal{A}_{\mathbf{GFG}} \times \mathcal{A}_{\mathbf{GFG}}$

Büchi case

Theorem (Kuperberg, S. ['14])

For Büchi automata $\left|\mathcal{A}_{\mathbf{det}}\right| \leqslant \left|\mathcal{A}_{\mathbf{GFG}}\right|^2$

1. brutal determinisation of $\mathcal{A}_{\mathbf{GFG}} \rightsquigarrow D$ $\quad (|D| \sim 2^{|\mathcal{A}_{\mathbf{GFG}}|})$
2. parity game on $\mathcal{A}_{\mathbf{GFG}} \times D$
3. rank-optimal winning strategy $\sigma_\exists$ $\quad$ [signatures (Walukiewicz ['02])]
4. inductive normalisation of $\mathcal{A}_{\mathbf{GFG}}$ w.r.t $\sigma_\exists$
5. monotonicity of ranks
6. it suffices to use $\mathcal{A}_{\mathbf{GFG}} \subseteq D$ instead of D
7. winning strategy on $\mathcal{A}_{\mathbf{GFG}} \times \mathcal{A}_{\mathbf{GFG}}$
8. $\rightsquigarrow \mathcal{A}_{\mathbf{det}}$

Problem

Input: an automaton $\mathcal{A}_{\mathbf{non-det}}$
Output: is $\mathcal{A}_{\mathbf{non-det}}$ GFG?

Problem

Input: an automaton $\mathcal{A}_{\mathbf{non-det}}$
Output: is $\mathcal{A}_{\mathbf{non-det}}$ GFG?

[EXPTIME algorithm via the GFG game]

Problem

Input: an automaton $\mathcal{A}_{\mathbf{non-det}}$
Output: is $\mathcal{A}_{\mathbf{non-det}}$ GFG?

[EXPTIME algorithm via the GFG game]

Proposition (Kuperberg ['13])

For parity automata at least as hard as solving parity games

Problem

Input: an automaton $\mathcal{A}_{\mathbf{non-det}}$
Output: is $\mathcal{A}_{\mathbf{non-det}}$ GFG?

[EXPTIME algorithm via the GFG game]

Proposition (Kuperberg ['13])

For parity automata at least as hard as solving parity games

Corollary (Kuperberg, S. ['14])

NP algorithm for Büchi automata

Problem

Input: an automaton $\mathcal{A}_{\mathbf{non-det}}$
Output: is $\mathcal{A}_{\mathbf{non-det}}$ GFG?

[EXPTIME algorithm via the GFG game]

Proposition (Kuperberg ['13])

For parity automata at least as hard as solving parity games

Corollary (Kuperberg, S. ['14])

NP algorithm for Büchi automata

Theorem (Kuperberg, S. ['14])

PTIME algorithm for co-Büchi automata

Problem

Input: an automaton $\mathcal{A}_{\mathbf{non-det}}$
Output: is $\mathcal{A}_{\mathbf{non-det}}$ GFG?

[EXPTIME algorithm via the GFG game]

Proposition (Kuperberg ['13])

For parity automata at least as hard as solving parity games

Corollary (Kuperberg, S. ['14])

NP algorithm for Büchi automata

Theorem (Kuperberg, S. ['14])

PTIME algorithm for co-Büchi automata

Proof

Joker game. . .

Joker game

Joker game

Arena: $\mathcal{A}_{\mathbf{non-det}} \times \mathcal{A}_{\mathbf{non-det}}$

Joker game

Arena: $\mathcal{A}_{\mathbf{non-det}} \times \mathcal{A}_{\mathbf{non-det}}$

q_i

$\bar{q}_i$

Joker game

Arena: $\mathcal{A}_{\mathbf{non-det}} \times \mathcal{A}_{\mathbf{non-det}}$

$\forall$: a_i

q_i

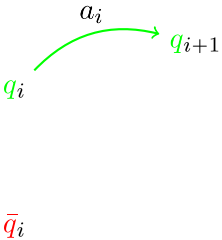
$\bar{q}_i$

Joker game

Arena: $\mathcal{A}_{\mathbf{non-det}} \times \mathcal{A}_{\mathbf{non-det}}$

$\forall$: a_i

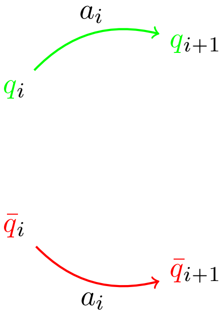
$\exists$: $q_i \xrightarrow{a_i} q_{i+1}$



Joker game

Arena: $\mathcal{A}_{\mathbf{non-det}} \times \mathcal{A}_{\mathbf{non-det}}$

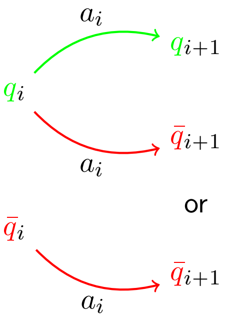
- $\forall$: a_i
- $\exists$: $q_i \xrightarrow{a_i} q_{i+1}$
- $\forall$: $\bar{q}_i \xrightarrow{a_i} \bar{q}_{i+1}$



Joker game

Arena: $\mathcal{A}_{\mathbf{non-det}} \times \mathcal{A}_{\mathbf{non-det}}$

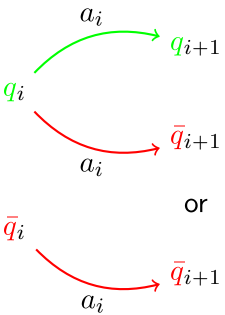
- $\forall$: a_i
- $\exists$: $q_i \xrightarrow{a_i} q_{i+1}$
- $\forall$: $\bar{q}_i \xrightarrow{a_i} \bar{q}_{i+1}$
- or: $q_i \xrightarrow{a_i} \bar{q}_{i+1}$ [**Joker**]



Joker game

Arena: $\mathcal{A}_{\mathbf{non-det}} \times \mathcal{A}_{\mathbf{non-det}}$

- $\forall$: a_i
- $\exists$: $q_i \xrightarrow{a_i} q_{i+1}$
- $\forall$: $\bar{q}_i \xrightarrow{a_i} \bar{q}_{i+1}$
- or: $q_i \xrightarrow{a_i} \bar{q}_{i+1}$ [**Joker**]



$\exists$ wins if either
- $(q_0, q_1, \ldots)$ is accepting
- $(\bar{q}_0, \bar{q}_1, \ldots)$ is rejecting
- $\forall$ played infinitely many **Jokers**

Joker game

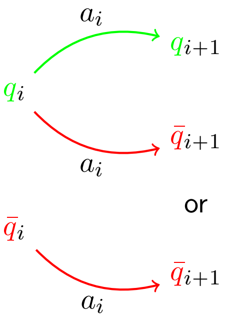
Arena: $\mathcal{A}_{\mathbf{non-det}} \times \mathcal{A}_{\mathbf{non-det}}$

$\forall$: a_i

$\exists$: $q_i \xrightarrow{a_i} q_{i+1}$

$\forall$: $\bar{q}_i \xrightarrow{a_i} \bar{q}_{i+1}$

or: $q_i \xrightarrow{a_i} \bar{q}_{i+1}$ [**Joker**]



$\exists$ wins if either $\begin{cases} (q_0, q_1, \ldots) \text{ is accepting} \\ (\bar{q}_0, \bar{q}_1, \ldots) \text{ is rejecting} \\ \forall \text{ played infinitely many } \textbf{Jokers} \end{cases}$

Conjecture (Kuperberg ['13])

For co-Büchi automata:
$\exists$ wins Joker game iff $\mathcal{A}_{\mathbf{non-det}}$ is GFG

Joker game

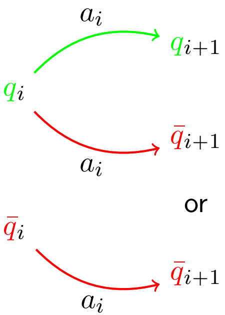
Arena: $\mathcal{A}_{\mathbf{non-det}} \times \mathcal{A}_{\mathbf{non-det}}$

$\forall$: a_i

$\exists$: $q_i \xrightarrow{a_i} q_{i+1}$

$\forall$: $\bar{q}_i \xrightarrow{a_i} \bar{q}_{i+1}$

or: $q_i \xrightarrow{a_i} \bar{q}_{i+1}$ [**Joker**]



$\exists$ wins if either
- $(q_0, q_1, \ldots)$ is accepting
- $(\bar{q}_0, \bar{q}_1, \ldots)$ is rejecting
- $\forall$ played infinitely many **Jokers**

Conjecture (Kuperberg ['13])

For co-Büchi automata:
$\exists$ wins Joker game iff $\mathcal{A}_{\mathbf{non-det}}$ is GFG

[for higher parities $\not\Rightarrow$]

Using Joker game

Using Joker game

1. If $\mathcal{A}$ is GFG then $\exists$ wins Joker game

Using Joker game

1. If $\mathcal{A}$ is GFG then $\exists$ wins Joker game

$\rightsquigarrow$ compute the winning region and a positional strategy of $\exists$

Using Joker game

1. If $\mathcal{A}$ is GFG then $\exists$ wins Joker game

$\rightsquigarrow$ compute the winning region and a positional strategy of $\exists$

$\rightsquigarrow$ if $\exists$ wins from $(q, \bar{q})$ then $\mathrm{L}(\mathcal{A}, q) \supseteq \mathrm{L}(\mathcal{A}, \bar{q})$

Using Joker game

1. If $\mathcal{A}$ is GFG then $\exists$ wins Joker game

$\rightsquigarrow$ compute the winning region and a positional strategy of $\exists$

$\rightsquigarrow$ if $\exists$ wins from $(q, \bar{q})$ then $\mathrm{L}(\mathcal{A}, q) \supseteq \mathrm{L}(\mathcal{A}, \bar{q})$

2. study the Joker **safety** game — rejecting transitions are loosing

Using Joker game

1. If $\mathcal{A}$ is GFG then $\exists$ wins Joker game

$\rightsquigarrow$ compute the winning region and a positional strategy of $\exists$

$\rightsquigarrow$ if $\exists$ wins from $(q, \bar{q})$ then $\mathrm{L}(\mathcal{A}, q) \supseteq \mathrm{L}(\mathcal{A}, \bar{q})$

2. study the Joker **safety** game — rejecting transitions are loosing

$\rightsquigarrow$ a new winning region of $\exists$

Using Joker game

1. If $\mathcal{A}$ is GFG then $\exists$ wins Joker game

$\rightsquigarrow$ compute the winning region and a positional strategy of $\exists$

$\rightsquigarrow$ if $\exists$ wins from $(q, \bar{q})$ then $\mathrm{L}(\mathcal{A}, q) \supseteq \mathrm{L}(\mathcal{A}, \bar{q})$

2. study the Joker **safety** game — rejecting transitions are loosing

$\rightsquigarrow$ a new winning region of $\exists$

3. construct a GFG automaton $\mathcal{B}$ for $\mathrm{L}(\mathcal{A})$

Using Joker game

1. If $\mathcal{A}$ is GFG then $\exists$ wins Joker game

$\rightsquigarrow$ compute the winning region and a positional strategy of $\exists$

$\rightsquigarrow$ if $\exists$ wins from $(q, \bar{q})$ then $\mathrm{L}(\mathcal{A}, q) \supseteq \mathrm{L}(\mathcal{A}, \bar{q})$

2. study the Joker **safety** game — rejecting transitions are loosing

$\rightsquigarrow$ a new winning region of $\exists$

3. construct a GFG automaton $\mathcal{B}$ for $\mathrm{L}(\mathcal{A})$

4. play the GFG game on $\mathcal{A} \times \mathcal{B}$

Using Joker game

1. If $\mathcal{A}$ is GFG then $\exists$ wins Joker game

$\rightsquigarrow$ compute the winning region and a positional strategy of $\exists$

$\rightsquigarrow$ if $\exists$ wins from $(q, \bar{q})$ then $\mathrm{L}(\mathcal{A}, q) \supseteq \mathrm{L}(\mathcal{A}, \bar{q})$

2. study the Joker **safety** game — rejecting transitions are loosing

$\rightsquigarrow$ a new winning region of $\exists$

3. construct a GFG automaton $\mathcal{B}$ for $\mathrm{L}(\mathcal{A})$

4. play the GFG game on $\mathcal{A} \times \mathcal{B}$

5. $\exists$ wins iff $\mathcal{A}$ is GFG

Summary

Summary

Two **positive** results:

Summary

Two **positive** results:

co-Büchi GFG automata
are exponentially succinct

Summary

Two **positive** results:

co-Büchi GFG automata are exponentially succinct

Büchi GFG automata can be efficiently determinised

Summary

Two **positive** results:

co-Büchi GFG automata are exponentially succinct

$\rightsquigarrow$

potential speed-up in synthesis

Büchi GFG automata can be efficiently determinised

Summary

Two **positive** results:

co-Büchi GFG automata are exponentially succinct

⇝

potential speed-up in synthesis

Büchi GFG automata can be efficiently determinised

⇝

fast complementation algorithm

Summary

Two **positive** results:

co-Büchi GFG automata are exponentially succinct

$\rightsquigarrow$

potential speed-up in synthesis

Büchi GFG automata can be efficiently determinised

$\rightsquigarrow$

fast complementation algorithm

Efficient characterisations:

Summary

Two **positive** results:

co-Büchi GFG automata are exponentially succinct

⇝

potential speed-up in synthesis

Büchi GFG automata can be efficiently determinised

⇝

fast complementation algorithm

Efficient characterisations:

NP for Büchi

Summary

Two **positive** results:

co-Büchi GFG automata are exponentially succinct

⇝

potential speed-up in synthesis

Büchi GFG automata can be efficiently determinised

⇝

fast complementation algorithm

Efficient characterisations:

NP for Büchi

PTIME for co-Büchi

Summary

Two **positive** results:

co-Büchi GFG automata are exponentially succinct

$\Downarrow$

potential speed-up in synthesis

Büchi GFG automata can be efficiently determinised

$\Downarrow$

fast complementation algorithm

Efficient characterisations:

NP for Büchi

PTIME for co-Büchi

On the way:

game theoretic arguments,

pumping techniques,

...