Revealing vs. Concealing: More Simulation Games for Büchi Inclusion

Milka Hutagalung, Martin Lange, Etienne Lozes

Universität Kassel

LATA 2013 - Bilbao, Spain
April 4, 2013

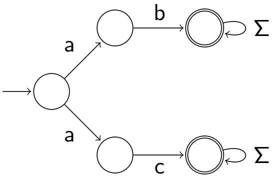
Table of contents

Büchi automata

- extends a finite automata to accept infinite sequences of letters
- represents a language over infinite words

Example

(Nondeterministic) Büchi automata $\mathcal{A}$ over $\Sigma = \{a, b, c\}$



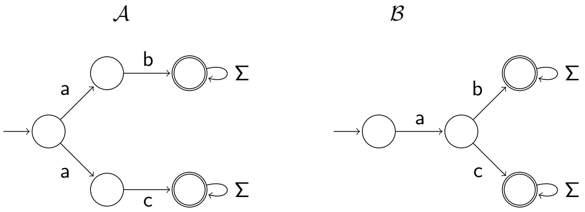
abaaa... $\in L(\mathcal{A})$
aaaaa... $\notin L(\mathcal{A})$

The language $L(\mathcal{A}) = a \cdot (b \cup c) \cdot (a \cup b \cup c)^{\omega}$.

Büchi inclusion problem

- Given : two Büchi automata $\mathcal{A}$, $\mathcal{B}$
  Question : Is $L(\mathcal{A}) \subseteq L(\mathcal{B})$?

Example

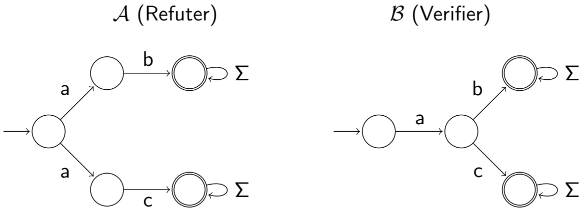


- PSPACE-complete

Büchi inclusion problem

- Given : two Büchi automata $\mathcal{A}$, $\mathcal{B}$
  Question : Is $L(\mathcal{A}) \subseteq L(\mathcal{B})$?

Example

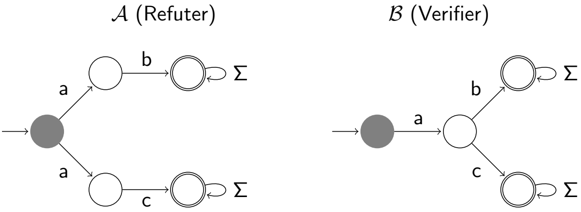


- PSPACE-complete
- fair simulation game approach:
  - "Refuter" plays against "Verifier"
  - each control a pebble on $\mathcal{A}$ and $\mathcal{B}$
  - Verifier tries to show language inclusion

Büchi inclusion problem

- Given : two Büchi automata $\mathcal{A}$, $\mathcal{B}$
  Question : Is $L(\mathcal{A}) \subseteq L(\mathcal{B})$?

Example

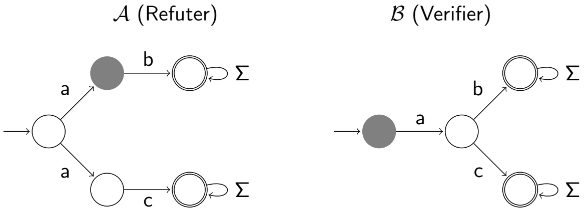


- PSPACE-complete
- fair simulation game approach:
  - "Refuter" plays against "Verifier"
  - each control a pebble on $\mathcal{A}$ and $\mathcal{B}$
  - Verifier tries to show language inclusion

Büchi inclusion problem

- Given : two Büchi automata $\mathcal{A}$, $\mathcal{B}$
  Question : Is $L(\mathcal{A}) \subseteq L(\mathcal{B})$?

Example

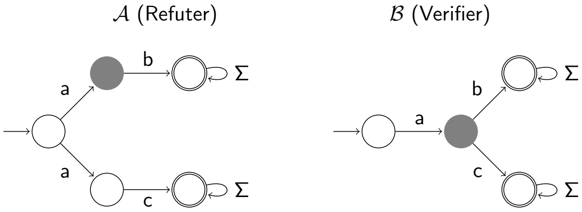


- PSPACE-complete
- fair simulation game approach:
  - "Refuter" plays against "Verifier"
  - each control a pebble on $\mathcal{A}$ and $\mathcal{B}$
  - Verifier tries to show language inclusion

Büchi inclusion problem

- Given : two Büchi automata $\mathcal{A}$, $\mathcal{B}$
  Question : Is $L(\mathcal{A}) \subseteq L(\mathcal{B})$?

Example

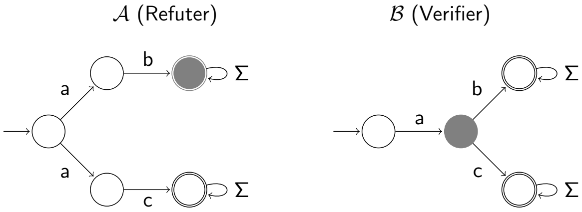


- PSPACE-complete
- fair simulation game approach:
  - "Refuter" plays against "Verifier"
  - each control a pebble on $\mathcal{A}$ and $\mathcal{B}$
  - Verifier tries to show language inclusion

Büchi inclusion problem

- Given : two Büchi automata $\mathcal{A}$, $\mathcal{B}$
  Question : Is $L(\mathcal{A}) \subseteq L(\mathcal{B})$?

Example

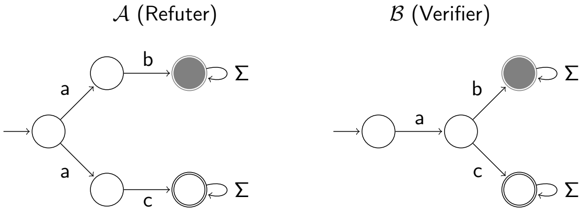


- PSPACE-complete
- fair simulation game approach:
  - "Refuter" plays against "Verifier"
  - each control a pebble on $\mathcal{A}$ and $\mathcal{B}$
  - Verifier tries to show language inclusion

Büchi inclusion problem

- Given : two Büchi automata $\mathcal{A}$, $\mathcal{B}$
  Question : Is $L(\mathcal{A}) \subseteq L(\mathcal{B})$?

Example

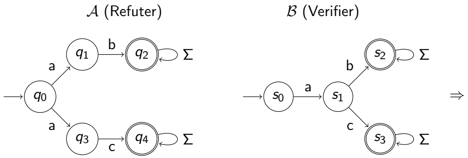


- PSPACE-complete
- fair simulation game approach:
  - "Refuter" plays against "Verifier"
  - each control a pebble on $\mathcal{A}$ and $\mathcal{B}$
  - Verifier tries to show language inclusion

Fair simulation game

- can be encoded to parity game with priorities: 0, 1, 2

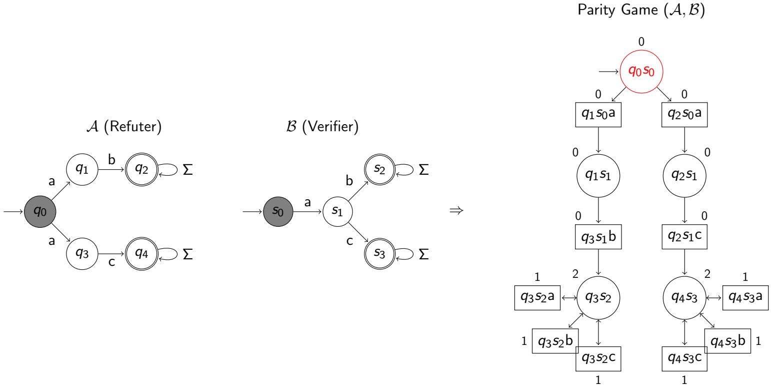
Example



Fair simulation game

- can be encoded to parity game with priorities: 0, 1, 2

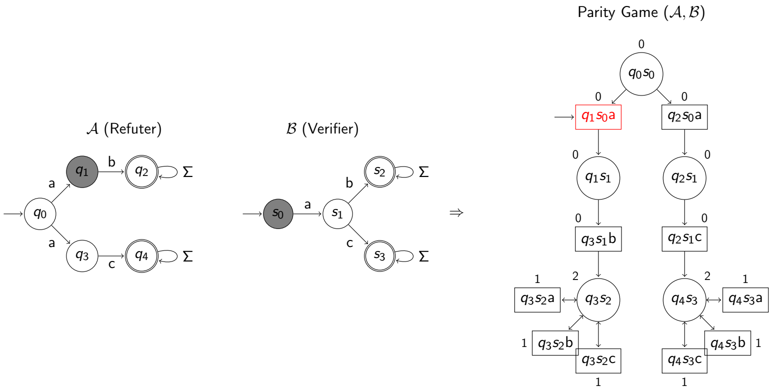
Example



Fair simulation game

- can be encoded to parity game with priorities: 0, 1, 2

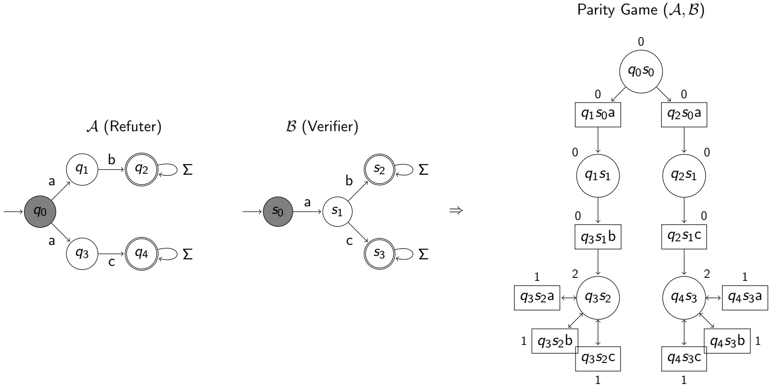
Example



Fair simulation game

- can be encoded to parity game with priorities: 0, 1, 2
- Verifier wins $\Leftrightarrow$ Player 0 wins $\quad \Rightarrow L(\mathcal{A}) \subseteq L(\mathcal{B})$
- parity game size $< |\mathcal{A}| \cdot |\mathcal{B}| \cdot |\Sigma|$

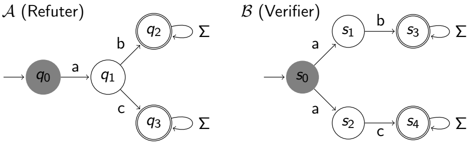
Example



Fair simulation game

- can be encoded to parity game with priorities: 0, 1, 2
- Verifier wins $\Leftrightarrow$ Player 0 wins $\quad \Rightarrow L(\mathcal{A}) \subseteq L(\mathcal{B})$
- parity game size $< |\mathcal{A}| \cdot |\mathcal{B}| \cdot |\Sigma|$

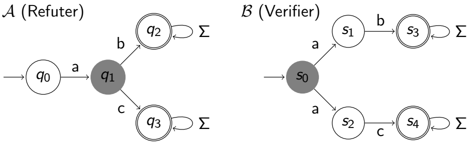
Example



Fair simulation game

- can be encoded to parity game with priorities: 0, 1, 2
- Verifier wins $\Leftrightarrow$ Player 0 wins $\quad \Rightarrow L(\mathcal{A}) \subseteq L(\mathcal{B})$
- parity game size $< |\mathcal{A}| \cdot |\mathcal{B}| \cdot |\Sigma|$

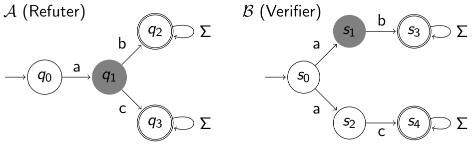
Example



Fair simulation game

- can be encoded to parity game with priorities: 0, 1, 2
- Verifier wins $\Leftrightarrow$ Player 0 wins $\quad \Rightarrow L(\mathcal{A}) \subseteq L(\mathcal{B})$
- parity game size $< |\mathcal{A}| \cdot |\mathcal{B}| \cdot |\Sigma|$

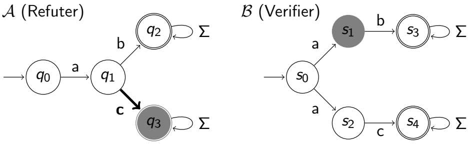
Example



Fair simulation game

- can be encoded to parity game with priorities: 0, 1, 2
- Verifier wins $\Leftrightarrow$ Player 0 wins $\quad \Rightarrow L(\mathcal{A}) \subseteq L(\mathcal{B})$
- parity game size $< |\mathcal{A}| \cdot |\mathcal{B}| \cdot |\Sigma|$

Example

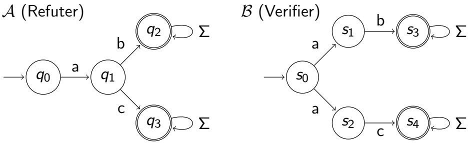


$L(\mathcal{A}) \subseteq L(\mathcal{B})$ but Verifier loss

Fair simulation game

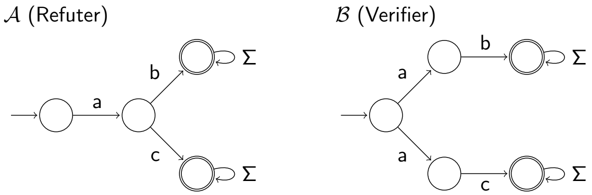
- can be encoded to parity game with priorities: 0, 1, 2
- Verifier wins $\Leftrightarrow$ Player 0 wins $\quad \Rightarrow L(\mathcal{A}) \subseteq L(\mathcal{B})$
- parity game size $< |\mathcal{A}| \cdot |\mathcal{B}| \cdot |\Sigma|$

Example



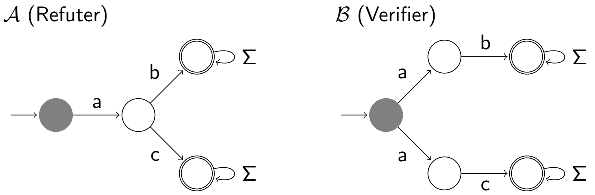
$L(\mathcal{A}) \subseteq L(\mathcal{B})$ but Verifier loss

k-pebble game [Etessami02]



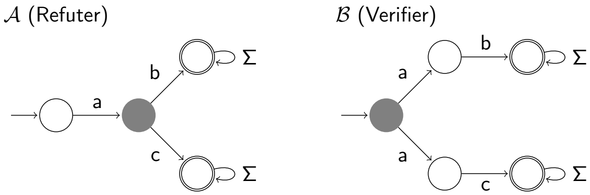
k-pebble game [Etessami02]

- Verifier controls k pebbles
- Verifier wins: Refuter forms an accepting run $\Rightarrow$ all infinite runs of Verifier are accepting



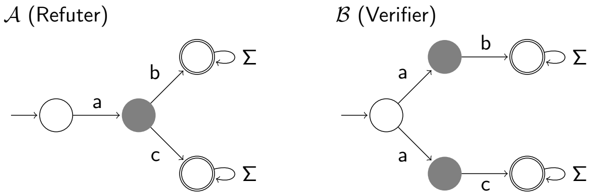
k-pebble game [Etessami02]

- Verifier controls k pebbles
- Verifier wins: Refuter forms an accepting run $\Rightarrow$ all infinite runs of Verifier are accepting



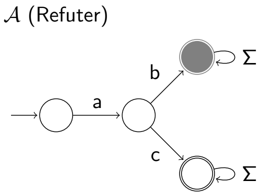
k-pebble game [Etessami02]

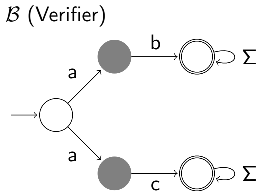
- Verifier controls k pebbles
- Verifier wins: Refuter forms an accepting run $\Rightarrow$ all infinite runs of Verifier are accepting



k-pebble game [Etessami02]

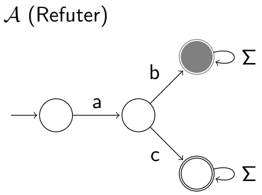
- Verifier controls k pebbles
- Verifier wins: Refuter forms an accepting run $\Rightarrow$ all infinite runs of Verifier are accepting

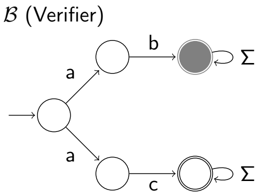




k-pebble game [Etessami02]

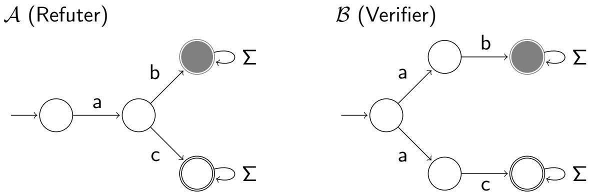
- Verifier controls k pebbles
- Verifier wins: Refuter forms an accepting run $\Rightarrow$ all infinite runs of Verifier are accepting





k-pebble game [Etessami02]

- Verifier controls k pebbles
- Verifier wins: Refuter forms an accepting run $\Rightarrow$ all infinite runs of Verifier are accepting

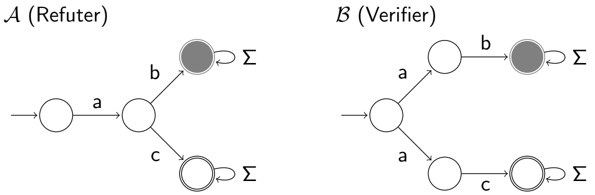


- forms a hierarchy:

$$\sqsubseteq^1_{\text{peb}} \subset \sqsubseteq^2_{\text{peb}} \subset \sqsubseteq^3_{\text{peb}} \subset \dots \subset \bigcup_{k\geq 1} \sqsubseteq^k_{\text{peb}} \subset \sqsubseteq_{\text{incl}}$$

k-pebble game [Etessami02]

- Verifier controls k pebbles
- Verifier wins: Refuter forms an accepting run $\Rightarrow$ all infinite runs of Verifier are accepting



- forms a hierarchy:

$$\sqsubseteq^1_{\text{peb}} \subset \sqsubseteq^2_{\text{peb}} \subset \sqsubseteq^3_{\text{peb}} \subset \dots \subset \bigcup_{k\geq 1} \sqsubseteq^k_{\text{peb}} \subset \sqsubseteq_{\text{incl}}$$

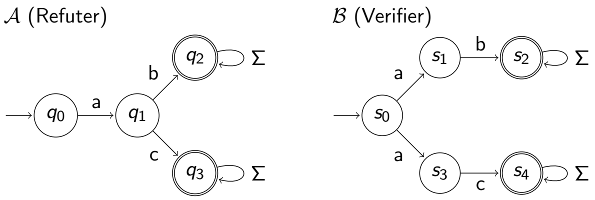
- parity game size $< |\mathcal{A}| \cdot (2 \cdot |\mathcal{B}| + 1)^k \cdot (|\Sigma| + 1)$

Static k-letter game

Static k-letter game

- both players control one pebble
- move k steps in each round

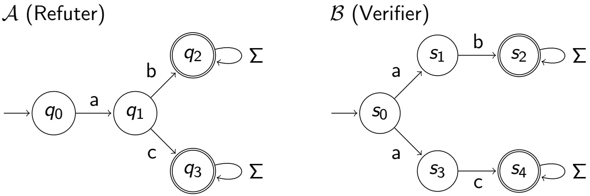
Example



Static k-letter game

- both players control one pebble
- move k steps in each round

Example

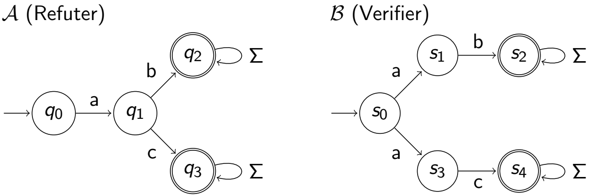


- parity game size $< |\mathcal{A}| \cdot |\mathcal{B}| \cdot (|\Sigma|^k + 1)$

Static k-letter game

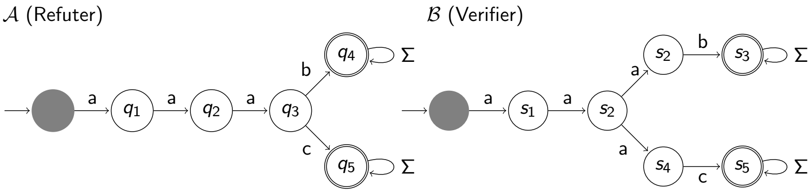
- both players control one pebble
- move k steps in each round

Example

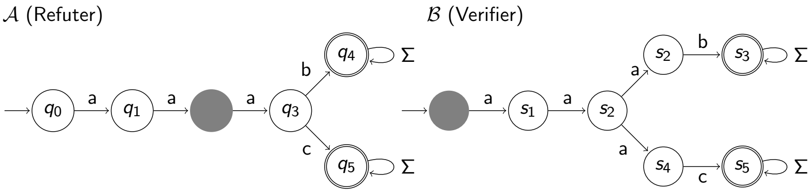


- parity game size $< |\mathcal{A}| \cdot |\mathcal{B}| \cdot (|\Sigma|^k + 1)$
- do not form a linear hierarchy

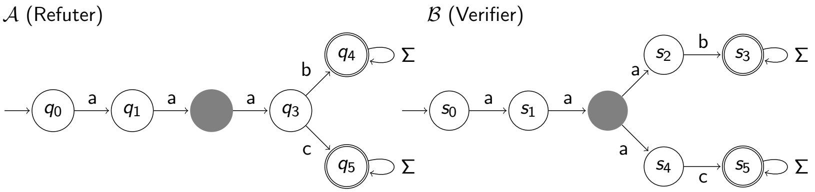
Example: static k-letter



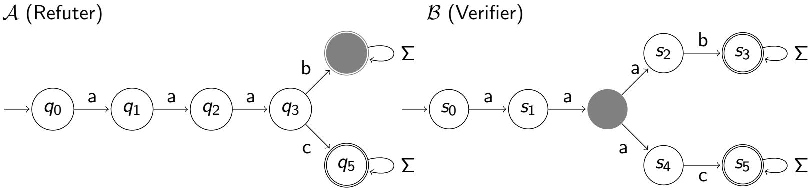
Example: static k-letter



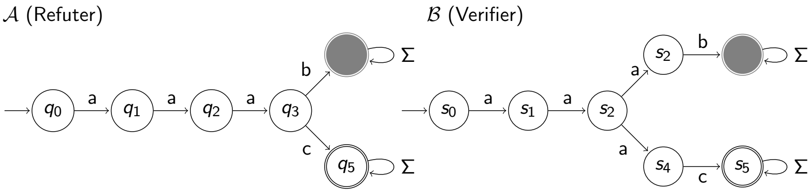
Example: static k-letter



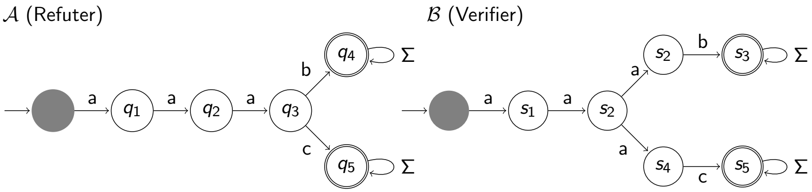
Example: static k-letter



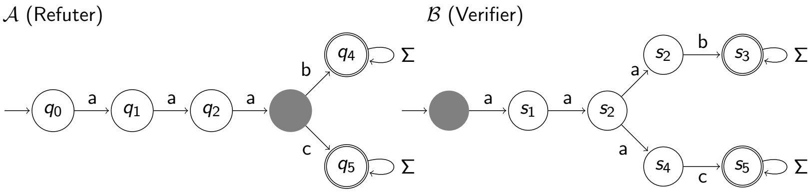
Example: static k-letter



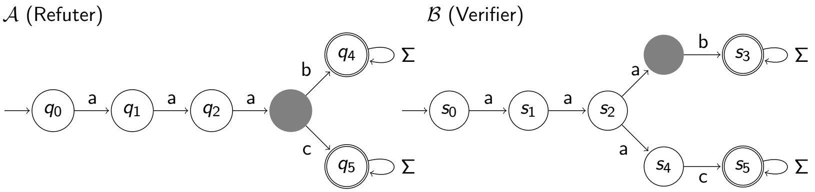
Example: static k-letter



Example: static k-letter

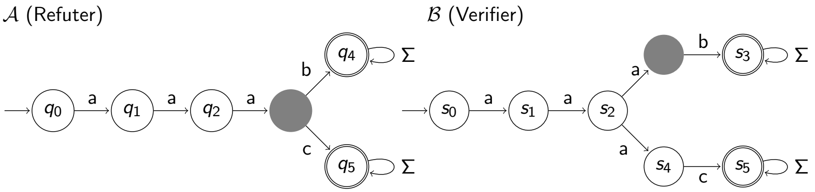


Example: static k-letter



- $\mathcal{A} \sqsubseteq^2_{\mathsf{stat}} \mathcal{B}$ but $\mathcal{A} \not\sqsubseteq^3_{\mathsf{stat}} \mathcal{B}$

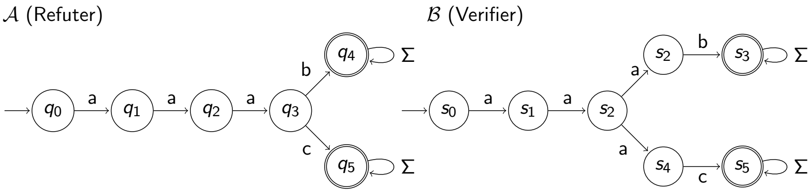
Example: static k-letter



- $\mathcal{A} \sqsubseteq^{2}_{\text{stat}} \mathcal{B}$ but $\mathcal{A} \not\sqsubseteq^{3}_{\text{stat}} \mathcal{B}$
- hierarchy:

$$\sqsubseteq^{\gcd(k,\ell)}_{\text{stat}} \subseteq \sqsubseteq^{k}_{\text{stat}} \;, \; \sqsubseteq^{\ell}_{\text{stat}} \subseteq \sqsubseteq^{\text{lcm}(k,\ell)}_{\text{stat}}$$

Example: static k-letter



- $\mathcal{A} \sqsubseteq^2_{\text{stat}} \mathcal{B}$ but $\mathcal{A} \not\sqsubseteq^3_{\text{stat}} \mathcal{B}$
- hierarchy:

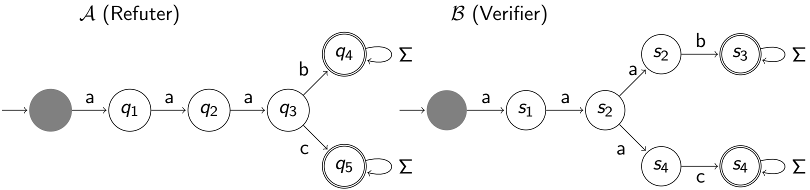
$$\sqsubseteq^{\gcd(k,\ell)}_{\text{stat}} \subseteq \sqsubseteq^{k}_{\text{stat}} , \sqsubseteq^{\ell}_{\text{stat}} \subseteq \sqsubseteq^{\text{lcm}(k,\ell)}_{\text{stat}}$$

- hard to determine a search is hopeless already
- can we refine the game?

Dynamic k-letter game

- verifier may choose how far both players move.
- both players move h_i steps in round i, $h_i < k$.

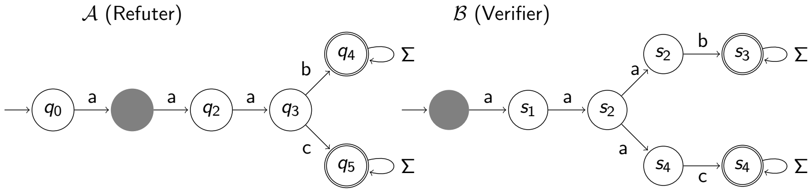
Example ($\mathcal{A} \sqsubseteq^3_{\mathrm{dyn}} \mathcal{B}$)



Dynamic k-letter game

- verifier may choose how far both players move.
- both players move h_i steps in round i, $h_i < k$.

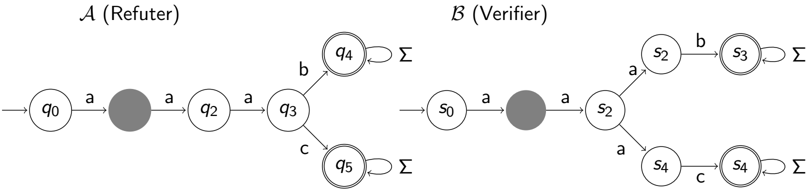
Example ($\mathcal{A} \sqsubseteq^3_{\text{dyn}} \mathcal{B}$)



Dynamic k-letter game

- verifier may choose how far both players move.
- both players move h_i steps in round i, $h_i < k$.

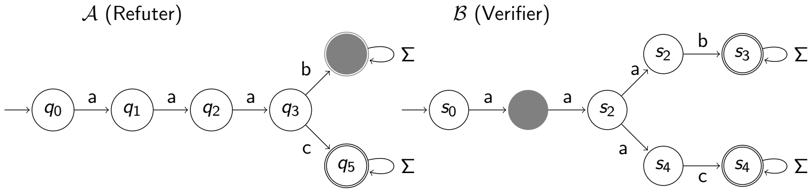
Example ($\mathcal{A} \sqsubseteq^3_{\text{dyn}} \mathcal{B}$)



Dynamic k-letter game

- verifier may choose how far both players move.
- both players move h_i steps in round i, $h_i < k$.

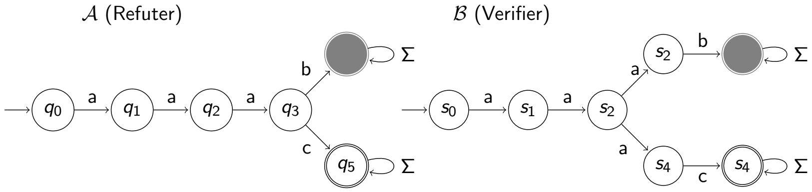
Example ($\mathcal{A} \sqsubseteq^3_{\text{dyn}} \mathcal{B}$)



Dynamic k-letter game

- verifier may choose how far both players move.
- both players move h_i steps in round i, $h_i < k$.

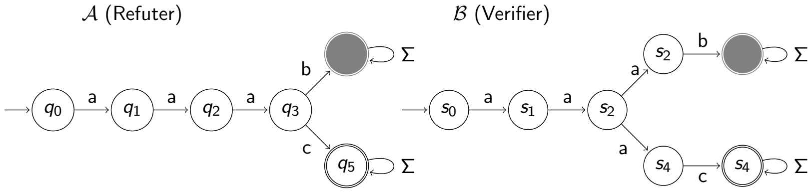
Example ($\mathcal{A} \sqsubseteq^3_{\text{dyn}} \mathcal{B}$)



Dynamic k-letter game

- verifier may choose how far both players move.
- both players move h_i steps in round i, $h_i < k$.

Example ($\mathcal{A} \sqsubseteq^3_{\text{dyn}} \mathcal{B}$)



- form a hierarchy:

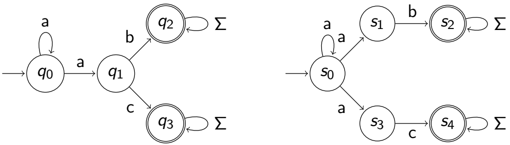
$$\sqsubseteq^1_{\text{dyn}} \subset \sqsubseteq^2_{\text{dyn}} \subset \sqsubseteq^3_{\text{dyn}} \subset \ldots \subset \bigcup_{k \geq 1} \sqsubseteq^k_{\text{dyn}} \subset \sqsubseteq_{\text{incl}}$$

- parity game size $< |\mathcal{A}| \cdot |\mathcal{B}| \cdot (|\Sigma|^{k+1} + k)$

Dynamic k-letter game

- more powerful than static game
- less powerful than pebble game

Example



- upper bound:

$$\mathcal{A} \sqsubseteq^{k}_{\text{dyn}} \mathcal{B} \quad \Rightarrow \quad \mathcal{A} \sqsubseteq^{k_0}_{\text{dyn}} \mathcal{B}, \quad k_0 := 2^{(|\mathcal{A}|+|\mathcal{B}|)^3}$$

Experiments

Experiments

mutual exclusion protocols							
	multi-letter				multi-pebble		Rabit
	k	dynamic	k	static	k	pebble	
Mcs	1	22.94s	1	23.48s	1	25.41s	39.00s
FischerV2	1	0.07s	1	0.07s	1	0.07s	0.09s
Peterson	1	0.01s	1	0.01s	1	0.01s	0.03s
Bakery	1	7.22s	1	7.16s	1	7.23s	4.43s
Phils	1	0.02s	1	0.02s	1	0.02s	0.11s
Fischer	1	40.80s	1	47.30s	1	47.37s	3.41s
FischerV3	-	>1h	-	>1h	-	>1h	7.63s
FischerV4	-	>1h	-	>1h	-	>1h	2136.70s
BakeryV2	2	36.01s	2	7.06s	-	>1h	>1h

random NBA								
	size $= 30$, $d_{\mathrm{acc}} = 0.1$, $d_{\mathrm{tr}} = 2$				size $= 50$, $d_{\mathrm{acc}} = 0.6$, $d_{\mathrm{tr}} = 3$			
	dynamic	static	pebble	Rabit	dynamic	static	pebble	Rabit
time	59.12s	52.78s	18.22s	1655.84s	0.55s	0.52s	2.09s	127.53s
success %	80%	82%	86%	58%	100%	100%	100%	100%
average k	3.66	4.33	1.93	-	1.01	1.01	1.01	-

Experiments

mutual exclusion protocols							
	multi-letter				multi-pebble		Rabit
	k	dynamic	k	static	k	pebble	
Mcs	1	22.94s	1	23.48s	1	25.41s	39.00s
FischerV2	1	0.07s	1	0.07s	1	0.07s	0.09s
Peterson	1	0.01s	1	0.01s	1	0.01s	0.03s
Bakery	1	7.22s	1	7.16s	1	7.23s	4.43s
Phils	1	0.02s	1	0.02s	1	0.02s	0.11s
Fischer	1	40.80s	1	47.30s	1	47.37s	3.41s
FischerV3	-	$>$1h	-	$>$1h	-	$>$1h	7.63s
FischerV4	-	$>$1h	-	$>$1h	-	$>$1h	2136.70s
BakeryV2	2	36.01s	2	7.06s	-	$>$1h	$>$1h

random NBA								
	size $= 30$, $d_{\text{acc}} = 0.1$, $d_{\text{tr}} = 2$				size $= 50$, $d_{\text{acc}} = 0.6$, $d_{\text{tr}} = 3$			
	dynamic	static	pebble	Rabit	dynamic	static	pebble	Rabit
time	59.12s	52.78s	18.22s	1655.84s	0.55s	0.52s	2.09s	127.53s
success %	80%	82%	86%	58%	100%	100%	100%	100%
average k	3.66	4.33	1.93	-	1.01	1.01	1.01	-

- incremental testing
  - reasonable approach for inclusion problem
  - when succeed, comparable to the complete method

Literatures

- Etessami, K. A Hierarchy of Polynomial-Time Computable Simulations for Automata. *CONCUR* 2002.
- Henzinger et al. Fair Simulation. *Inf. Comput.* 2002.
- Abdulla et al. Advanced Ramsey-Based Büchi Automata Inclusion Testing. *CONCUR* 2011
- Jurdzinski, M. Small Progress Measures for Solving Parity Games. *Inf. Comput.* 2000.
- Friedmann, O. Lange, M. Solving Parity Games in Practice. *ATVA* 2009.
- Tabakov, D. Vardi, M. Experimental Evaluation of Classical Automata Constructions. *LPAR* 2005.