

Strategy shapes for population games

+ Some open problems

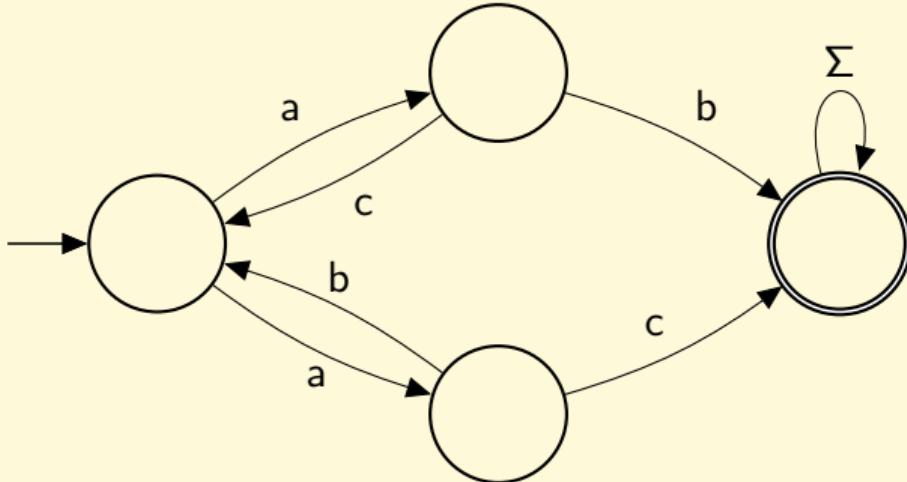
Ongoing work with Hugo Gimbert and Patrick Totzke



Dagstuhl

Population games

\mathcal{A} an NFA

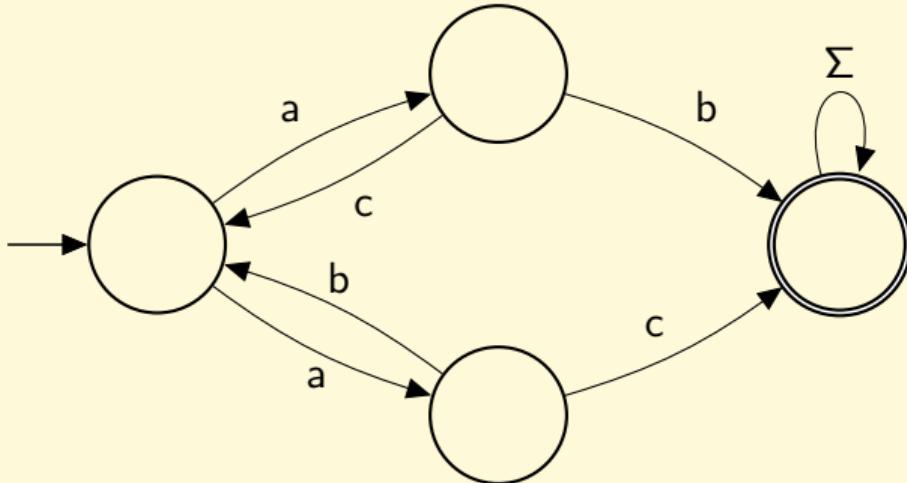


Population games

\mathcal{A} an NFA



Laetitia



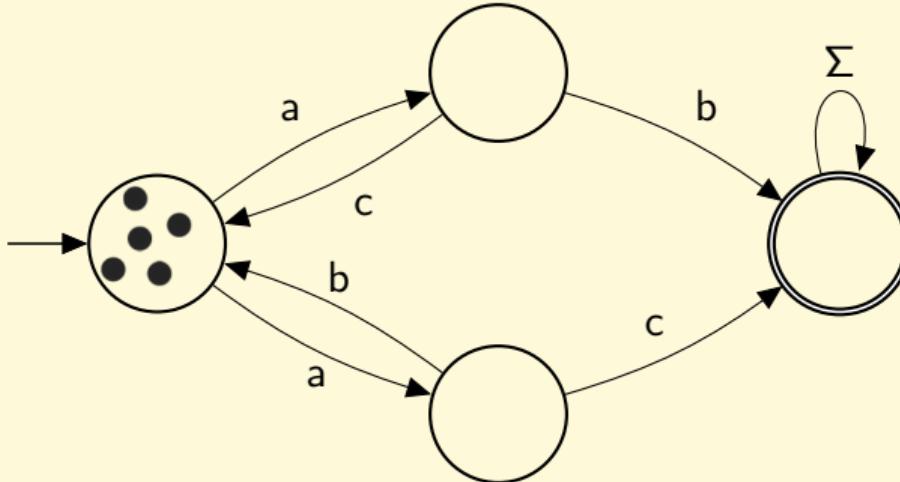
Terrence

Population games

\mathcal{A} an NFA



Laetitia



Terrence

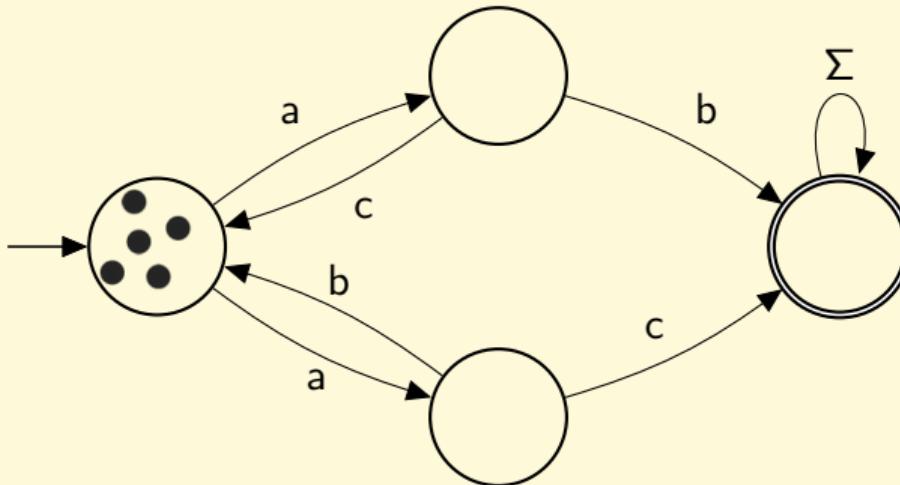
- ▶ We put N tokens in the initial state.

Population games

\mathcal{A} an NFA



Laetitia

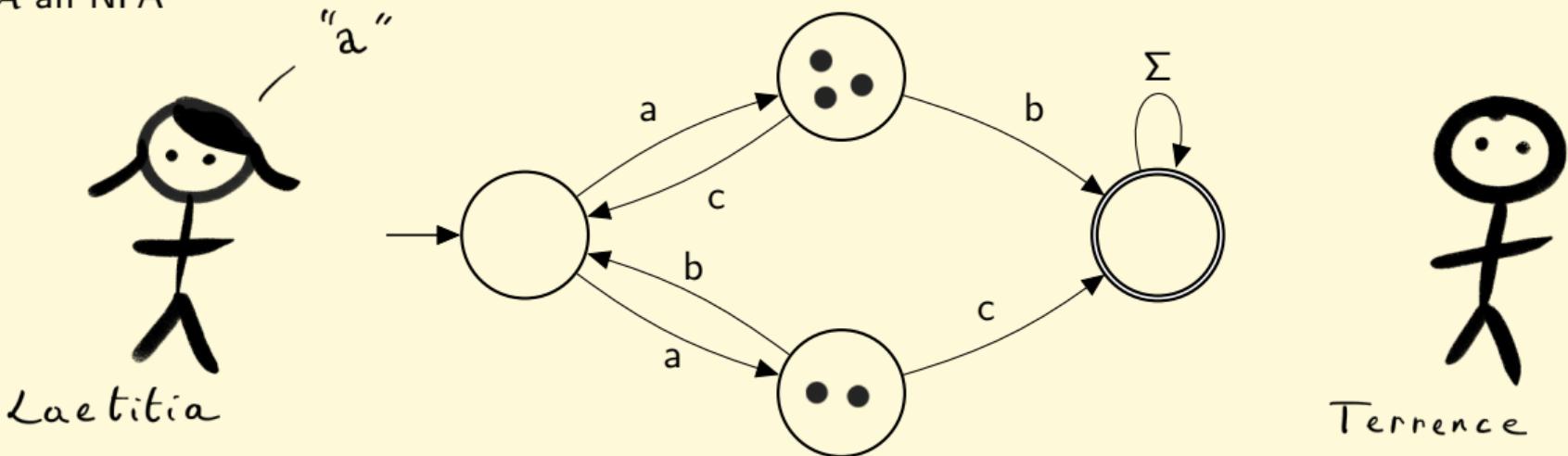


Terrence

- ▶ We put N tokens in the initial state.
- ▶ At each turn:
 - Laetitia picks a letter $x \in \Sigma$
 - Terrence moves each token along an x -transition
- ▶ Laetitia wins when all tokens are in the final state.

Population games

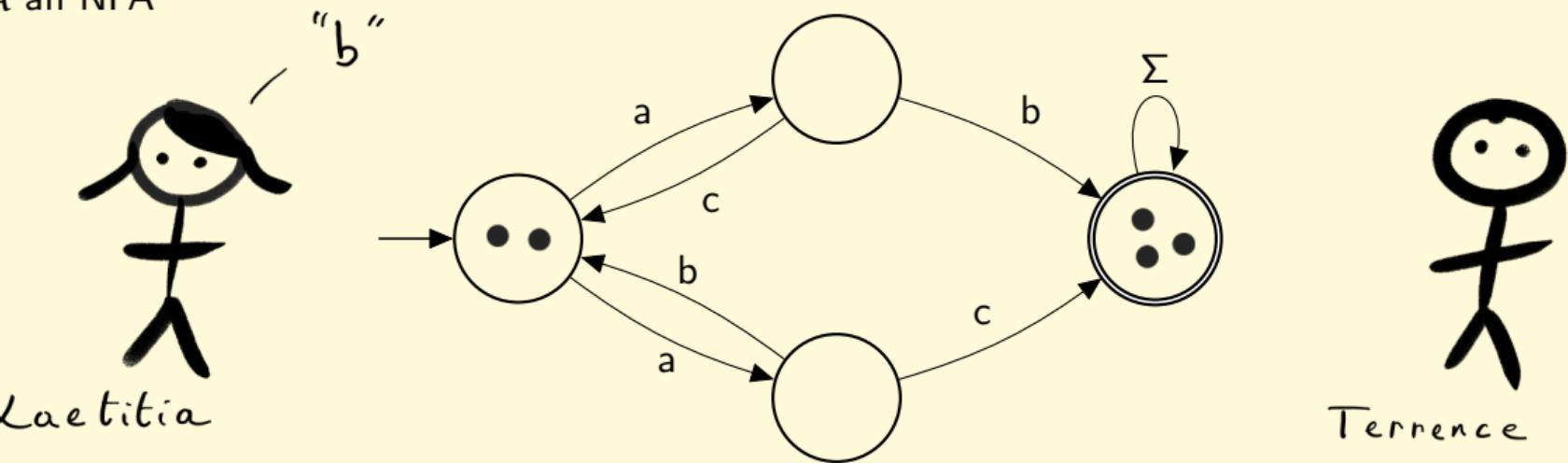
\mathcal{A} an NFA



- ▶ We put N tokens in the initial state.
- ▶ At each turn:
 - Laetitia picks a letter $x \in \Sigma$
 - Terrence moves each token along an x -transition
- ▶ Laetitia wins when all tokens are in the final state.

Population games

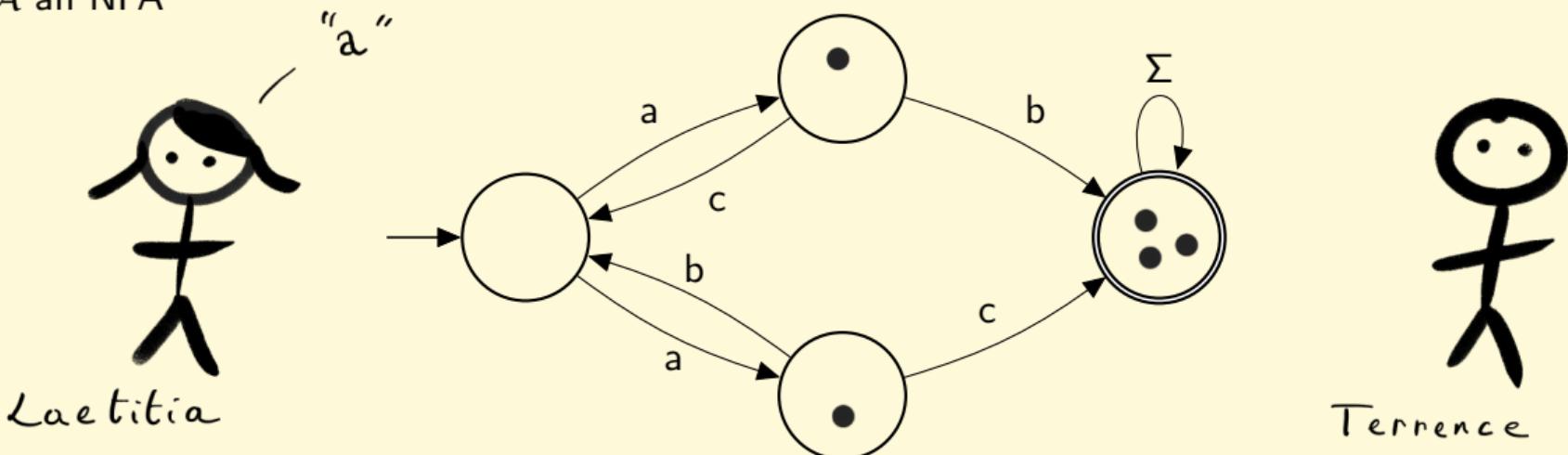
\mathcal{A} an NFA



- ▶ We put N tokens in the initial state.
- ▶ At each turn:
 - Laetitia picks a letter $x \in \Sigma$
 - Terrence moves each token along an x -transition
- ▶ Laetitia wins when all tokens are in the final state.

Population games

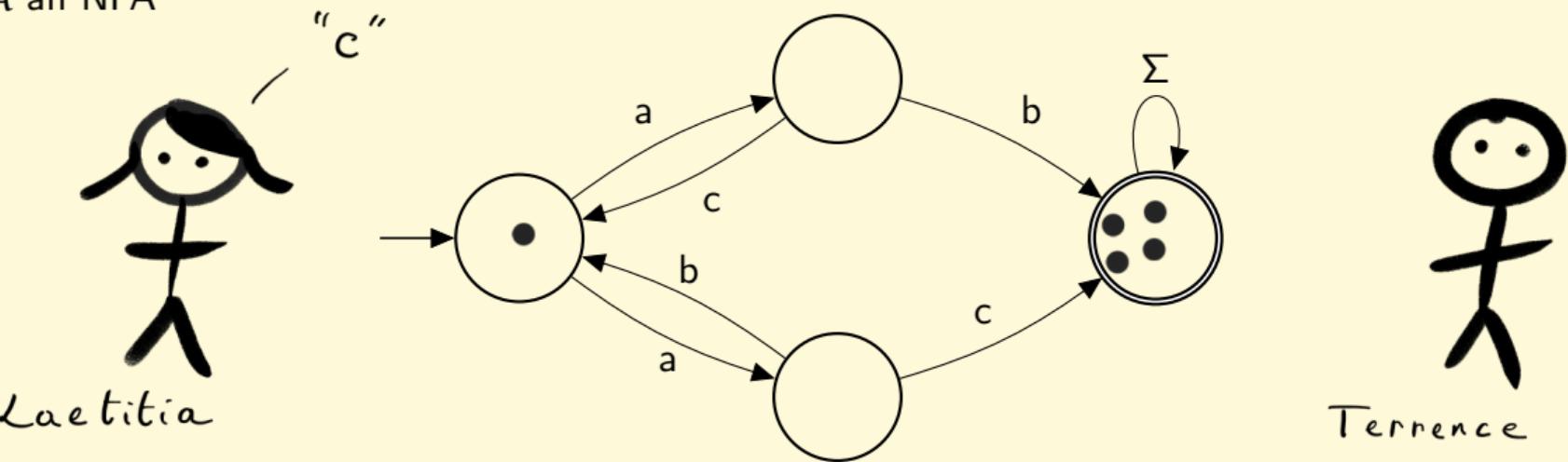
\mathcal{A} an NFA



- ▶ We put N tokens in the initial state.
- ▶ At each turn:
 - Laetitia picks a letter $x \in \Sigma$
 - Terrence moves each token along an x -transition
- ▶ Laetitia wins when all tokens are in the final state.

Population games

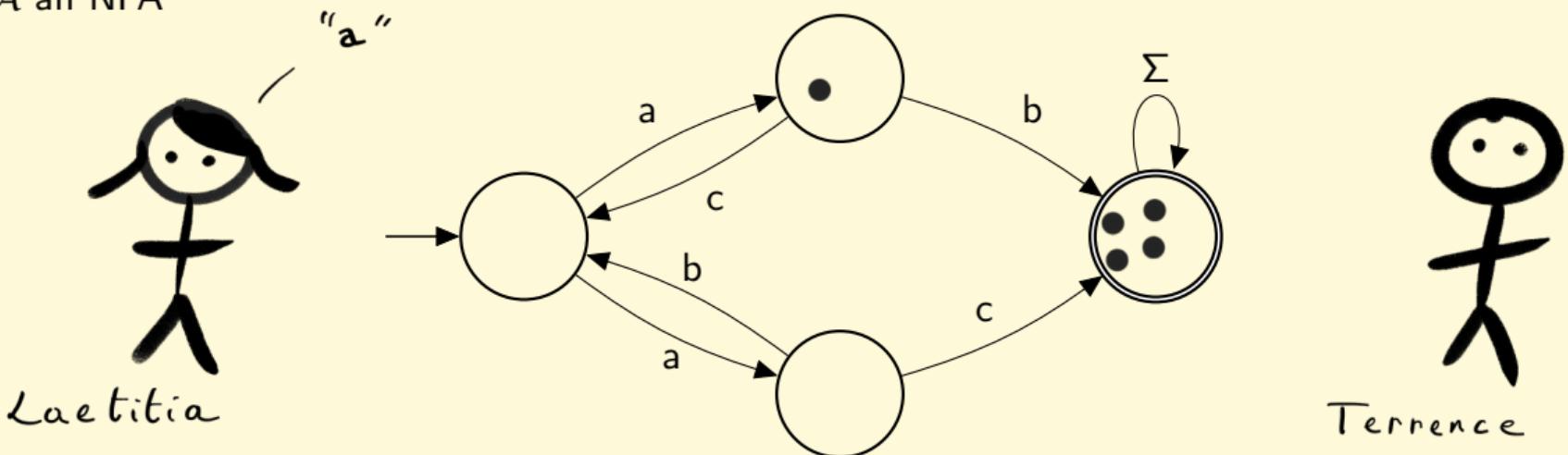
\mathcal{A} an NFA



- ▶ We put N tokens in the initial state.
- ▶ At each turn:
 - Laetitia picks a letter $x \in \Sigma$
 - Terrence moves each token along an x -transition
- ▶ Laetitia wins when all tokens are in the final state.

Population games

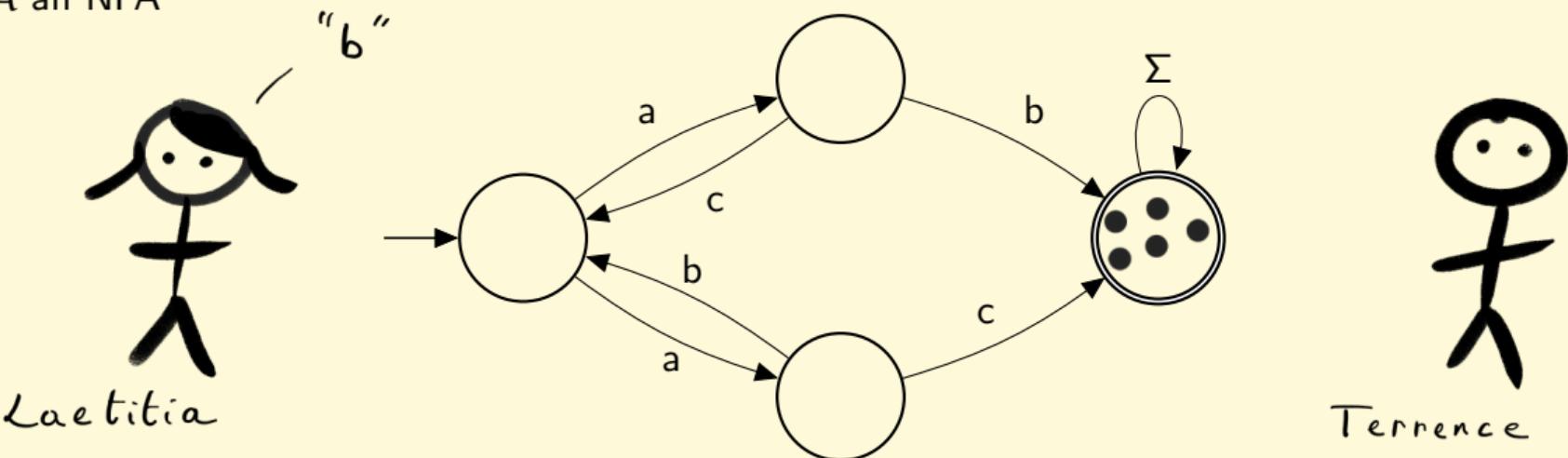
\mathcal{A} an NFA



- ▶ We put N tokens in the initial state.
- ▶ At each turn:
 - Laetitia picks a letter $x \in \Sigma$
 - Terrence moves each token along an x -transition
- ▶ Laetitia wins when all tokens are in the final state.

Population games

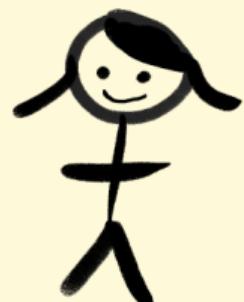
\mathcal{A} an NFA



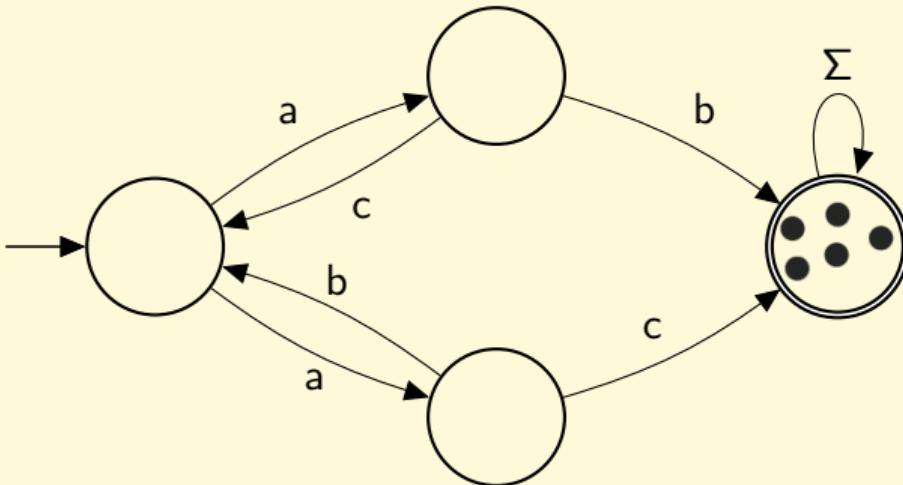
- ▶ We put N tokens in the initial state.
- ▶ At each turn:
 - Laetitia picks a letter $x \in \Sigma$
 - Terrence moves each token along an x -transition
- ▶ Laetitia wins when all tokens are in the final state.

Population games

\mathcal{A} an NFA



Laetitia



Terrence

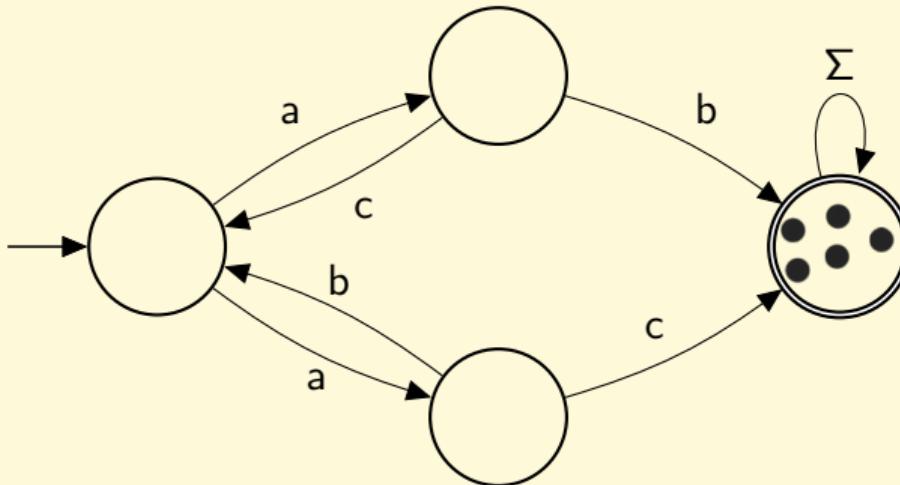
- ▶ We put N tokens in the initial state.
- ▶ At each turn:
 - Laetitia picks a letter $x \in \Sigma$
 - Terrence moves each token along an x -transition
- ▶ Laetitia wins when all tokens are in the final state.

Population games

\mathcal{A} an NFA



Laetitia



Terrence

- ▶ We put N tokens in the initial state.
- ▶ At each turn:
 - Laetitia picks a letter $x \in \Sigma$
 - Terrence moves each token along an x -transition
- ▶ Laetitia wins when all tokens are in the final state.

Population control problem

I: an NFA \mathcal{A}

O: Can Laetitia win over \mathcal{A} for all N ?

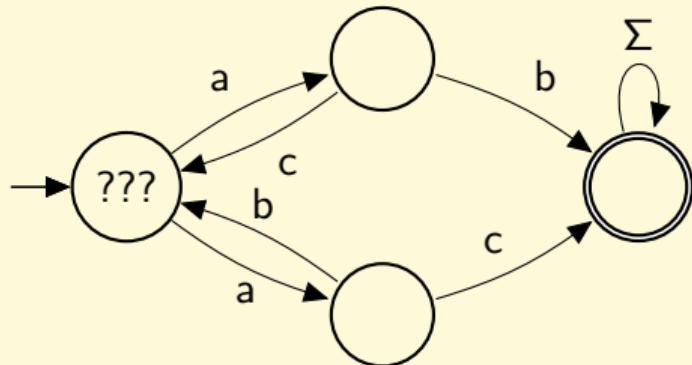
[Bertrand, Dewaskar, Genest, Gimbert 2017]

The Population Control Problem is EXPTIME-complete.

The Population Control Problem is EXPTIME-complete.

Proof:

Hide the number of tokens!

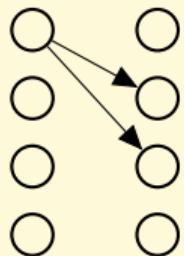
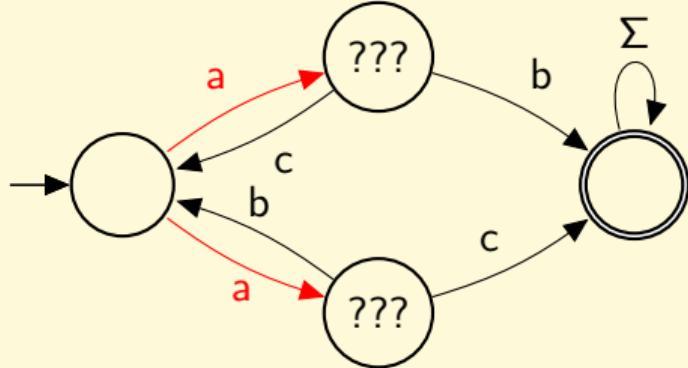


[Bertrand, Dewaskar, Genest, Gimbert 2017]

The Population Control Problem is EXPTIME-complete.

Proof:

Hide the number of tokens!

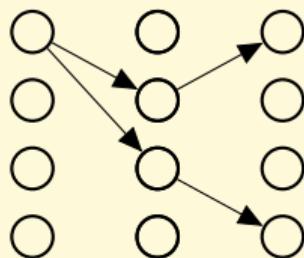
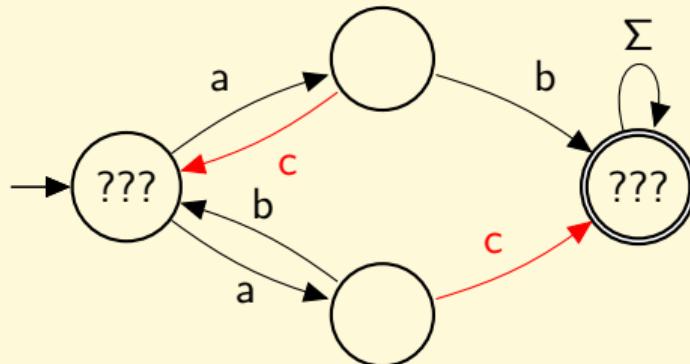


[Bertrand, Dewaskar, Genest, Gimbert 2017]

The Population Control Problem is EXPTIME-complete.

Proof:

Hide the number of tokens!

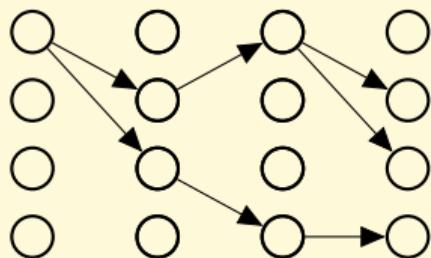
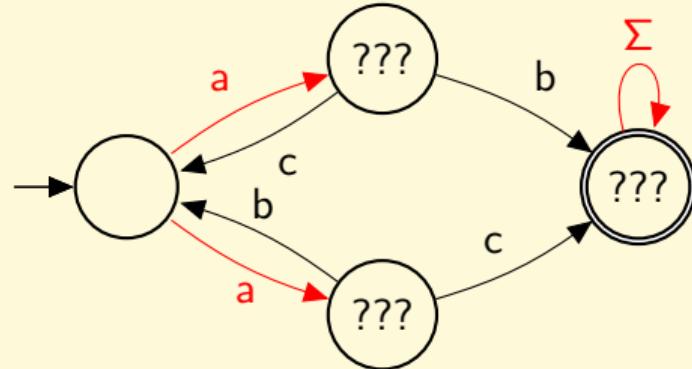


[Bertrand, Dewaskar, Genest, Gimbert 2017]

The Population Control Problem is EXPTIME-complete.

Proof:

Hide the number of tokens!

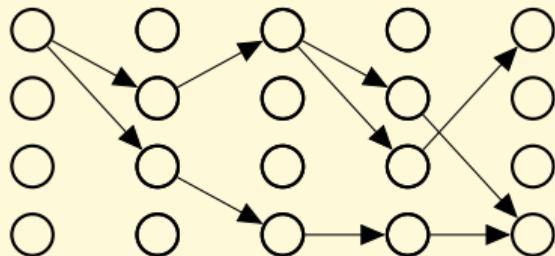
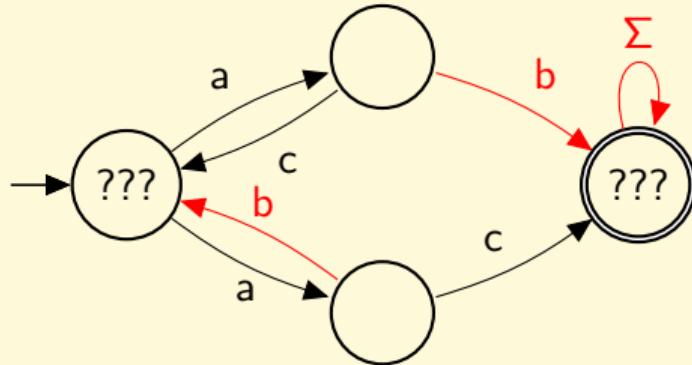


[Bertrand, Dewaskar, Genest, Gimbert 2017]

The Population Control Problem is EXPTIME-complete.

Proof:

Hide the number of tokens!

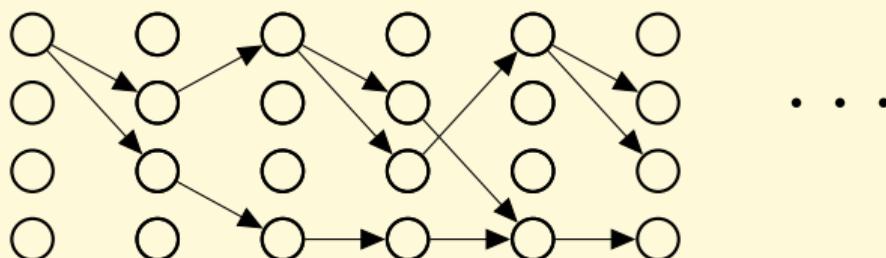
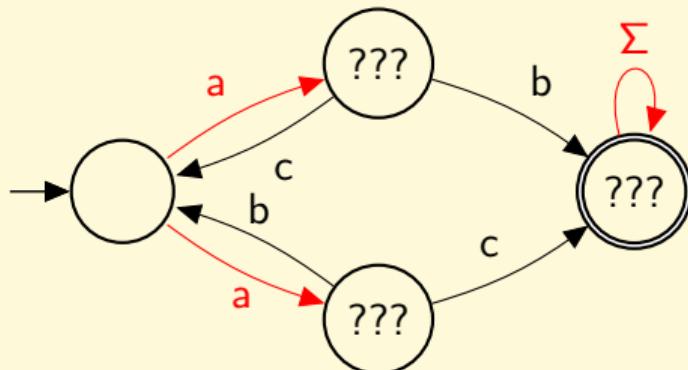


[Bertrand, Dewaskar, Genest, Gimbert 2017]

The Population Control Problem is EXPTIME-complete.

Proof:

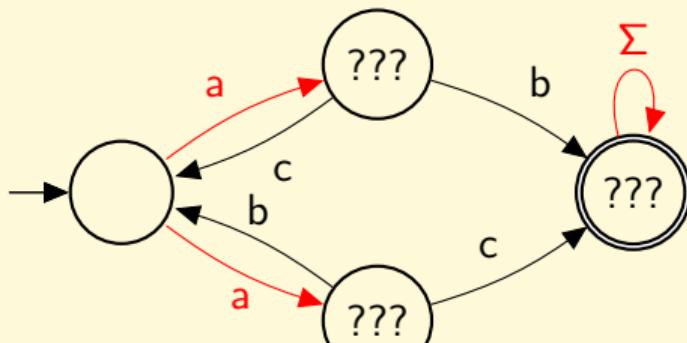
Play a different game!



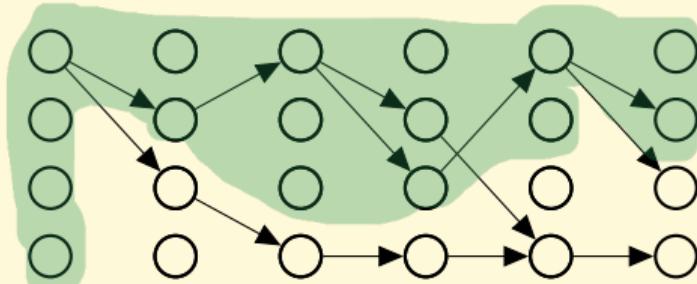
The Population Control Problem is EXPTIME-complete.

Proof:

Play a different game!



Ternence cheating



...

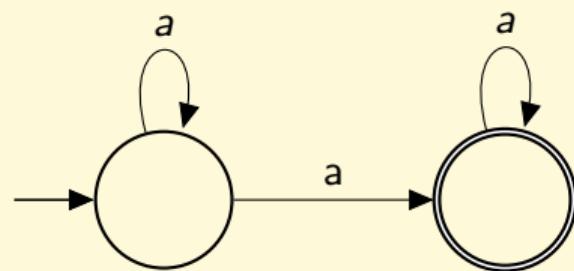


Randomised Population Control Problem

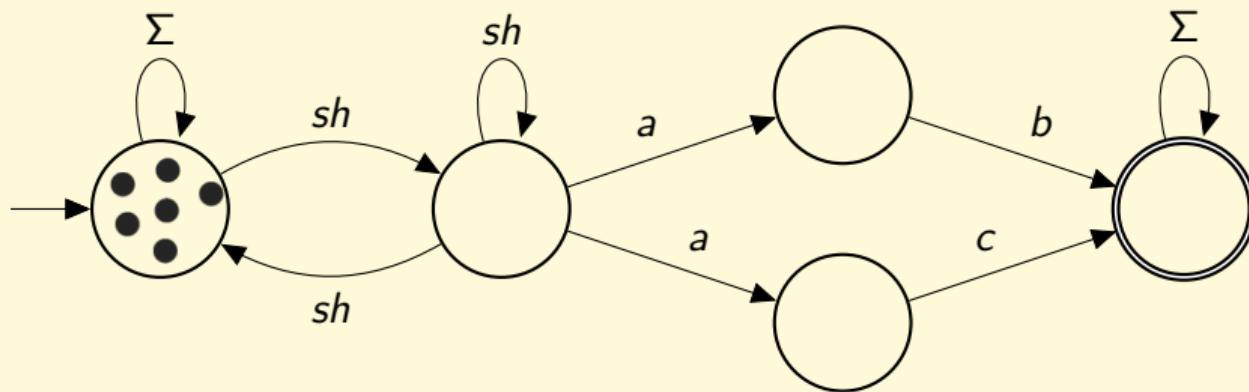
Terrence chooses the next transition of each token uniformly at random.

Randomised Population Control Problem

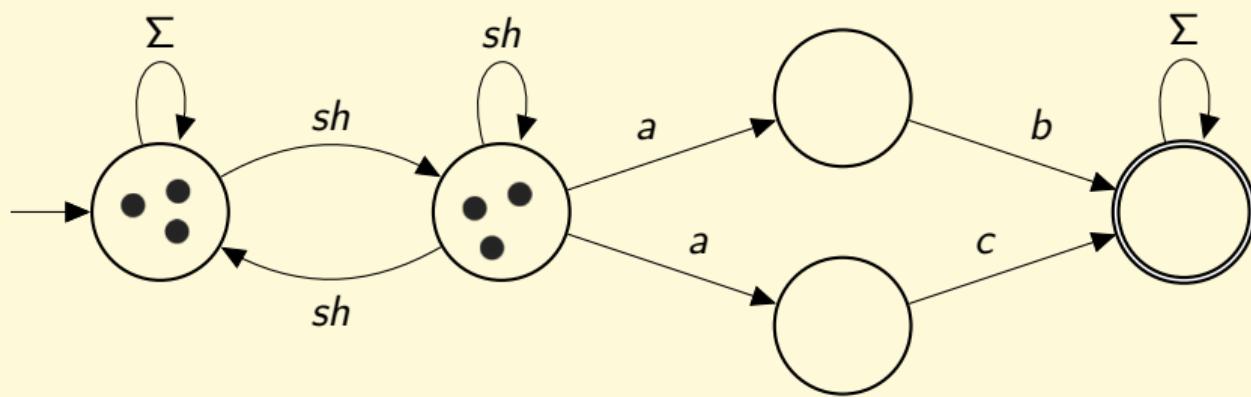
Terrence chooses the next transition of each token uniformly at random.



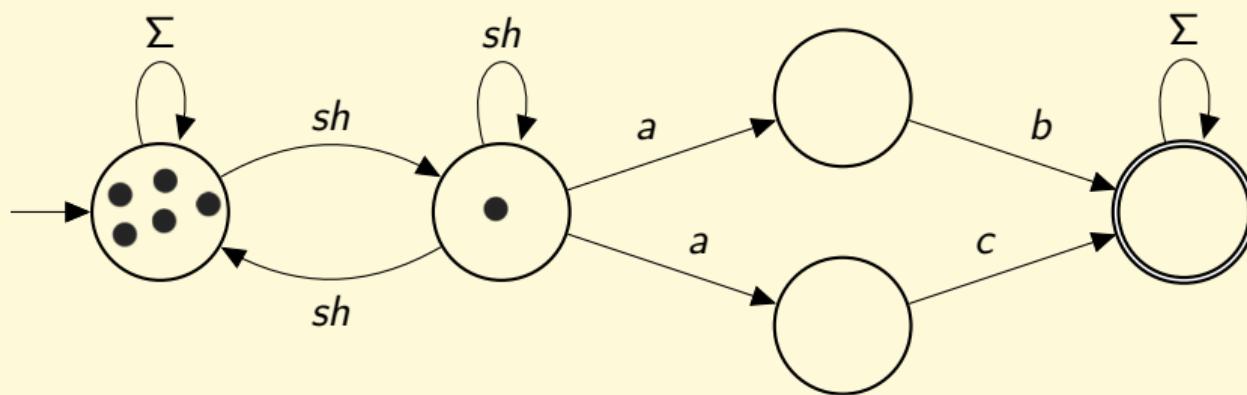
Randomised Population Control Problem



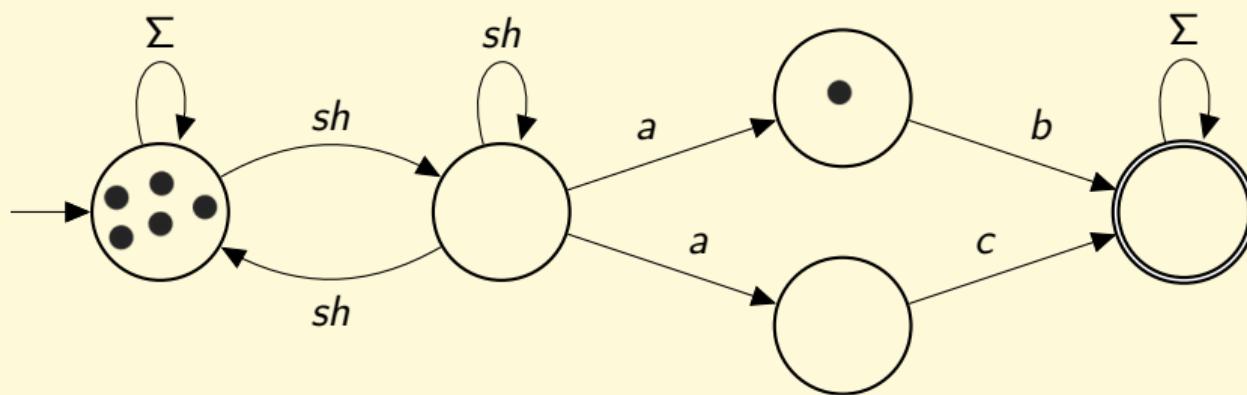
Randomised Population Control Problem



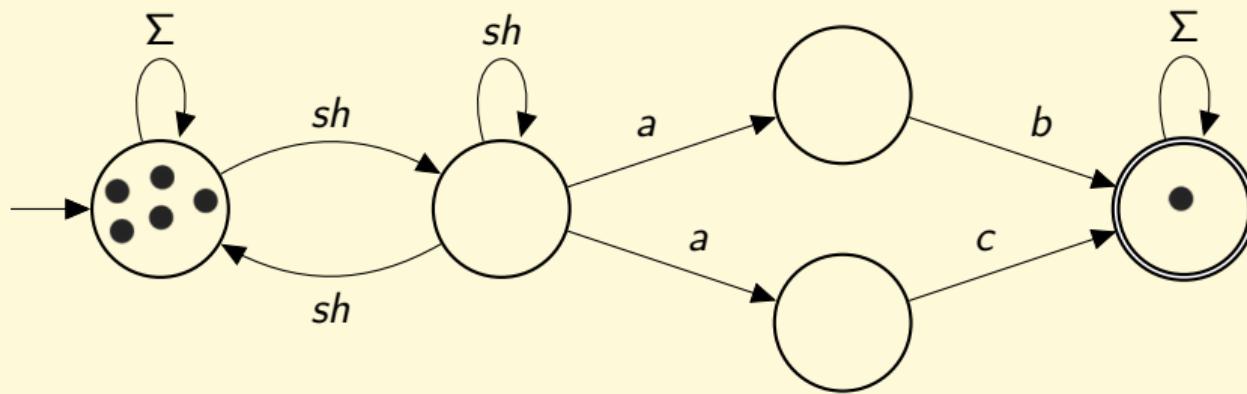
Randomised Population Control Problem



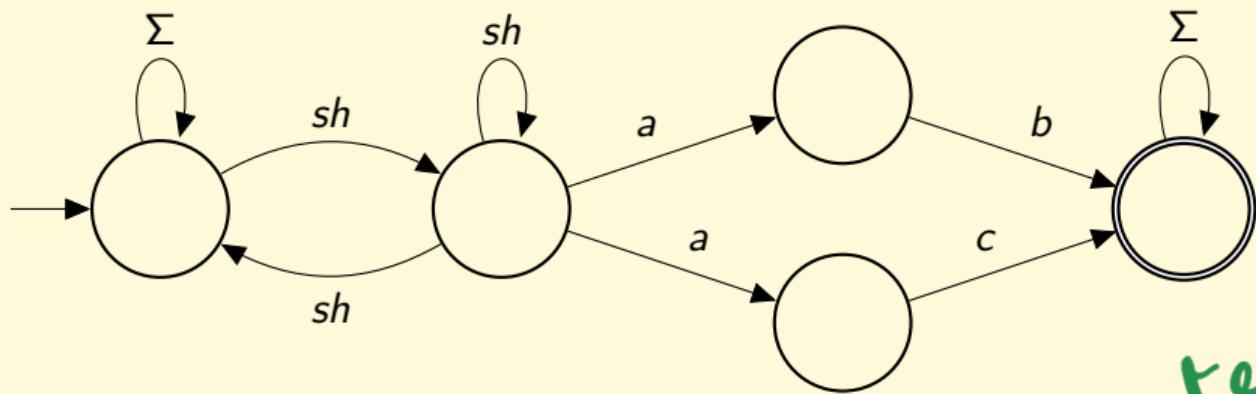
Randomised Population Control Problem



Randomised Population Control Problem



Randomised Population Control Problem



Strategies
need to count!

Randomised Population Control Problem

[Colcombet, Fijalkow, Ohlmann 2020]

The Randomised Population Control Problem is decidable.

Proof: transform the game.

- ▶ Compute the winning region as a fixpoint.
- ▶ Iteration uses cost functions
- ▶ Terminates thanks to well quasi-orders

[M., Shirmohammadi, Totzke 2019]

The Randomised Population Control Problem is EXPTIME-hard.

Randomised Population Control Problem

[Colcombet, Fijalkow, Ohlmann 2020]

The Randomised Population Control Problem is decidable.

Proof: transform the game.

- ▶ Compute the winning region as a fixpoint.
- ▶ Iteration uses cost functions
- ▶ Terminates thanks to well quasi-orders

→ Ackermannian complexity

[M., Shirmohammadi, Totzke 2019]

The Randomised Population Control Problem is EXPTIME-hard.

Strategy for RPCP

[Gimbert, M., Totzke]

The Randomised Population Control Problem is EXPTIME-complete.



Strategy for RPCP

[Gimbert, M., Totzke]

The Randomised Population Control Problem is EXPTIME-complete.



- ▶ Try to take a $(\omega, 0, 1)$ -path to the end.

Strategy for RPCP

[Gimbert, M., Totzke]

The Randomised Population Control Problem is EXPTIME-complete.



- ▶ Try to take a $(\omega, 0, 1)$ -path to the end.

Strategy for RPCP

[Gimbert, M., Totzke]

The Randomised Population Control Problem is EXPTIME-complete.



- ▶ Try to take a $(\omega, 0, 1)$ -path to the end.

Strategy for RPCP

[Gimbert, M., Totzke]

The Randomised Population Control Problem is EXPTIME-complete.



- ▶ Try to take a $(\omega, 0, 1)$ -path to the end.
- ▶ If something goes wrong, bring back isolated tokens to ω .

Strategy for RPCP

[Gimbert, M., Totzke]

The Randomised Population Control Problem is EXPTIME-complete.



- ▶ Try to take a $(\omega, 0, 1)$ -path to the end.
- ▶ If something goes wrong, bring back isolated tokens to ω .

Strategy for RPCP

[Gimbert, M., Totzke]

The Randomised Population Control Problem is EXPTIME-complete.



- ▶ Try to take a $(\omega, 0, 1)$ -path to the end.
- ▶ If something goes wrong, bring back isolated tokens to ω .

Strategy for RPCP

[Gimbert, M., Totzke]

The Randomised Population Control Problem is EXPTIME-complete.



- ▶ Try to take a $(\omega, 0, 1)$ -path to the end.
- ▶ If something goes wrong, bring back isolated tokens to ω .

Strategy for RPCP

[Gimbert, M., Totzke]

The Randomised Population Control Problem is EXPTIME-complete.



- ▶ Try to take a $(\omega, 0, 1)$ -path to the end.
- ▶ If something goes wrong, bring back isolated tokens to ω .
- ▶ We can ensure that tokens of different colours never meet
⇒ At most $|Q|$ isolated tokens per state

↳ Bounded constants

Open problems: Speed of victory

Fact

If Laetitia wins the randomised game then she can win in exponential expected time in the number of tokens.

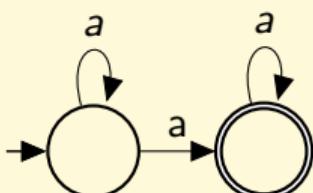


Open problems: Speed of victory

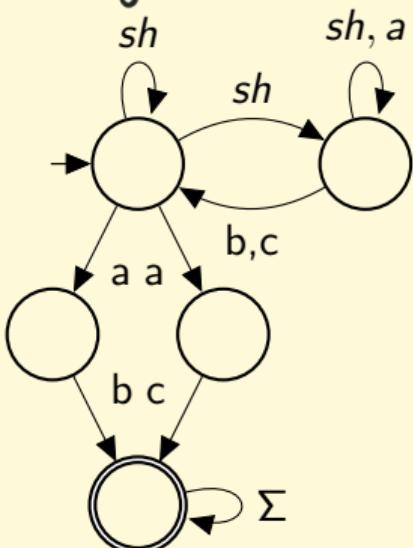
Fact

If Laetitia wins the randomised game then she can win in exponential expected time in the number of tokens.

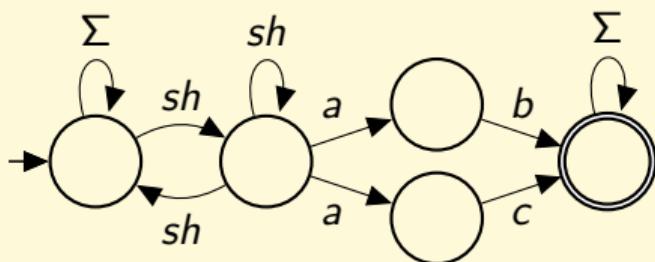
Polylog



Polynomial



Exponential

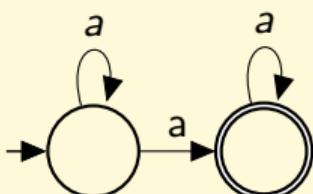


Open problems: Speed of victory

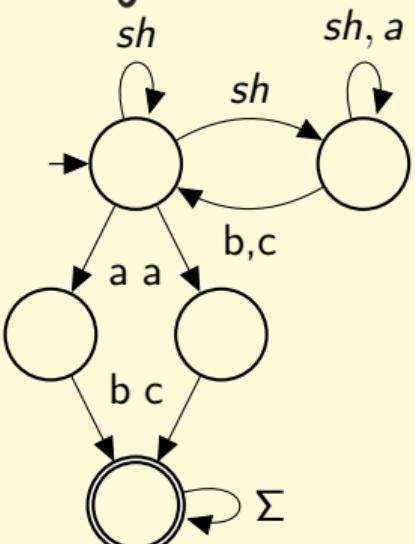
Fact

If Laetitia wins the randomised game then she can win in exponential expected time in the number of tokens.

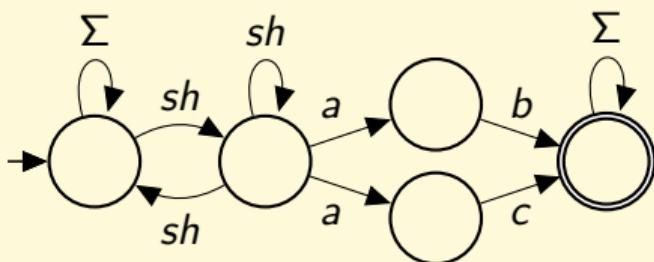
Polylog



Polynomial



Exponential



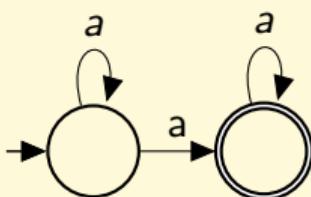
Problem: characterize
these classes

Open problems: Speed of victory

Fact

If Laetitia wins the randomised game then she can win in exponential expected time in the number of tokens.

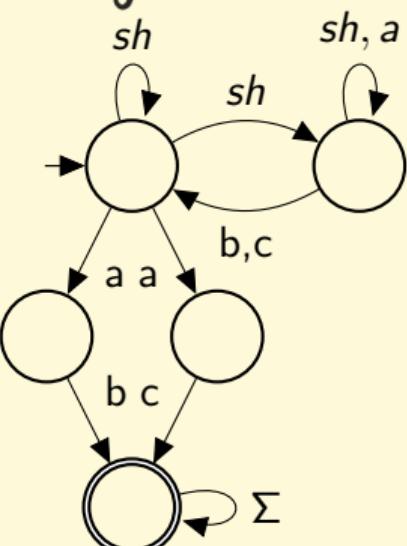
Polylog



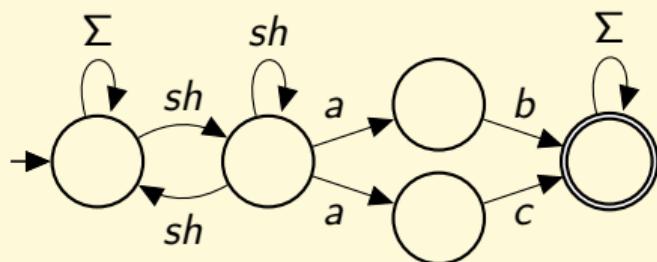
Conjecture:

Characterised by the
Markov Monoid
generated by $(\Sigma, \cdot, \#)$

Polynomial



Exponential



Problem: characterize
these classes

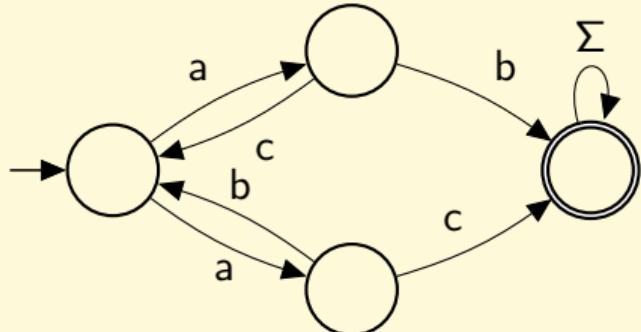
Open problems: Speed of victory

[Bertrand, Dewaskar, Genest, Gimbert, Godbole 2019]

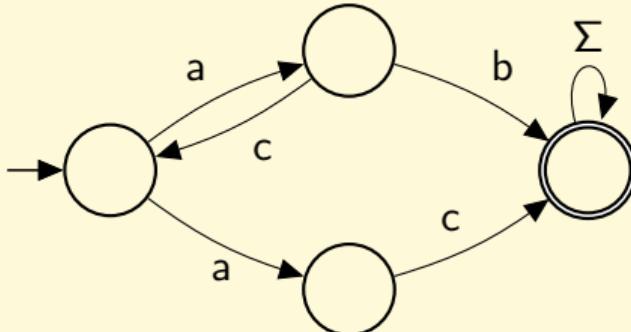
If Laetitia wins the adversarial game then she can win in polynomial time in the number of tokens.

What about polylogarithmic time?

Poly log



Polynomial



Open problems: Explorable automata [Hazard, Kuperberg 2023]

\mathcal{A} a non-deterministic parity automaton

- ▶ Put N tokens on the initial state
- ▶ Laetitia picks letters, Terrence moves tokens

Open problems: Explorable automata [Hazard, Kuperberg 2023]

\mathcal{A} a non-deterministic parity automaton

- ▶ Put N tokens on the initial state
- ▶ Laetitia picks letters, Terrence moves tokens
- ▶ Terrence wins if either
 - the sequence of letters picked by Laetitia is not in $L(\mathcal{A})$

Open problems: Explorable automata [Hazard, Kuperberg 2023]

\mathcal{A} a non-deterministic parity automaton

- ▶ Put N tokens on the initial state
- ▶ Laetitia picks letters, Terrence moves tokens
- ▶ Terrence wins if either
 - the sequence of letters picked by Laetitia is not in $L(\mathcal{A})$
 - or at least one token follows an accepting run.

Open problems: Explorable automata [Hazard, Kuperberg 2023]

\mathcal{A} a non-deterministic parity automaton

- ▶ Put N tokens on the initial state
- ▶ Laetitia picks letters, Terrence moves tokens
- ▶ Terrence wins if either
 - the sequence of letters picked by Laetitia is not in $L(\mathcal{A})$
 - or at least one token follows an accepting run.

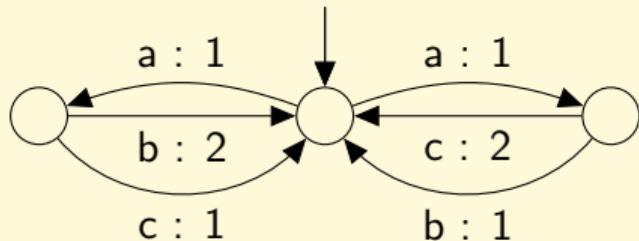
\mathcal{A} is *explorable* if Terrence wins for some N .

Open problems: Explorable automata [Hazard, Kuperberg 2023]

\mathcal{A} a non-deterministic parity automaton

- ▶ Put N tokens on the initial state
- ▶ Laetitia picks letters, Terrence moves tokens
- ▶ Terrence wins if either
 - the sequence of letters picked by Laetitia is not in $L(\mathcal{A})$
 - or at least one token follows an accepting run.

\mathcal{A} is *explorable* if Terrence wins for some N .

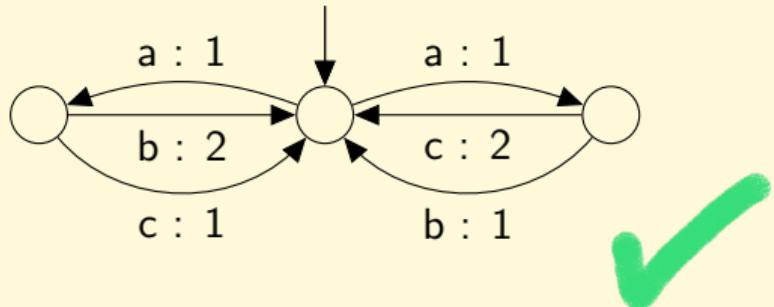


Open problems: Explorable automata [Hazard, Kuperberg 2023]

\mathcal{A} a non-deterministic parity automaton

- ▶ Put N tokens on the initial state
- ▶ Laetitia picks letters, Terrence moves tokens
- ▶ Terrence wins if either
 - the sequence of letters picked by Laetitia is not in $L(\mathcal{A})$
 - or at least one token follows an accepting run.

\mathcal{A} is *explorable* if Terrence wins for some N .

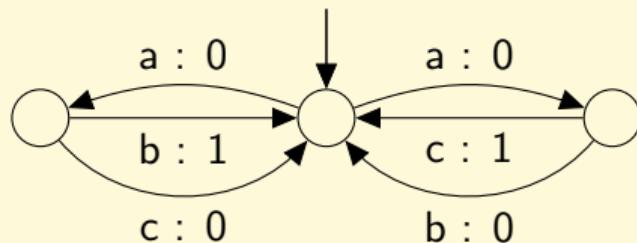
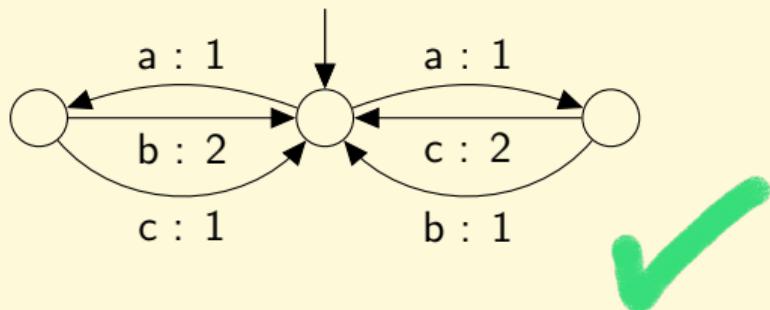


Open problems: Explorable automata [Hazard, Kuperberg 2023]

\mathcal{A} a non-deterministic parity automaton

- ▶ Put N tokens on the initial state
- ▶ Laetitia picks letters, Terrence moves tokens
- ▶ Terrence wins if either
 - the sequence of letters picked by Laetitia is not in $L(\mathcal{A})$
 - or at least one token follows an accepting run.

\mathcal{A} is *explorable* if Terrence wins for some N .

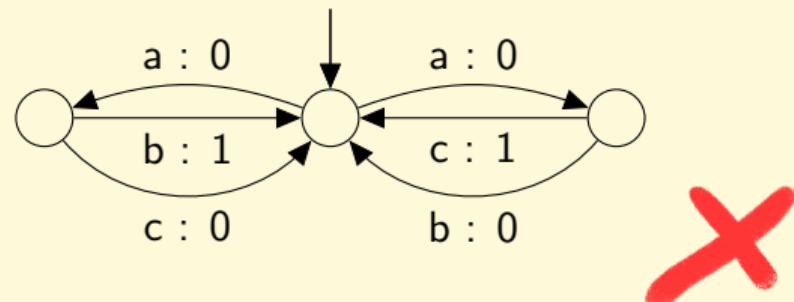
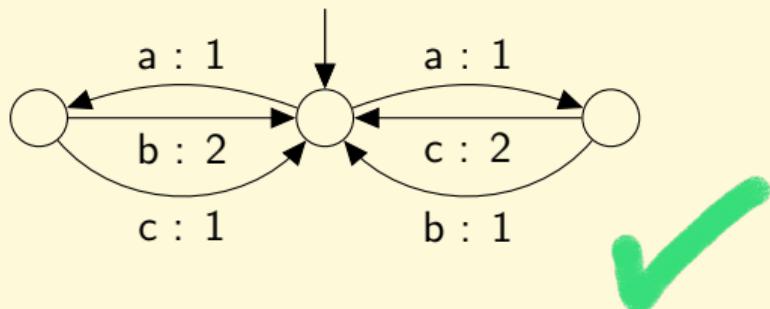


Open problems: Explorable automata [Hazard, Kuperberg 2023]

\mathcal{A} a non-deterministic parity automaton

- ▶ Put N tokens on the initial state
- ▶ Laetitia picks letters, Terrence moves tokens
- ▶ Terrence wins if either
 - the sequence of letters picked by Laetitia is not in $L(\mathcal{A})$
 - or at least one token follows an accepting run.

\mathcal{A} is *explorable* if Terrence wins for some N .



Open problems: Explorable automata [Hazard, Kuperberg 2023]

\mathcal{A} a non-deterministic parity automaton

- ▶ Put N tokens on the initial state
- ▶ Laetitia picks letters, Terrence moves tokens
- ▶ Terrence wins if either
 - the sequence of letters picked by Laetitia is not in $L(\mathcal{A})$
 - or at least one token follows an accepting run.

\mathcal{A} is *explorable* if Terrence wins for some N .

[Hazard, Idir, Kuperberg]

The explorability problem is decidable on parity automata iff it is decidable on $[1, 3]$ -automata

Future work

- ▶ Speed of victory
- ▶ Deciding (ω -)explorability for parity automata
- ▶ Threshold problems: Can Laetitia:
 - gather some proportion r of the tokens with probability 1?
 - gather all tokens with probability $\geq p$?

Future work

- ▶ Speed of victory
- ▶ Deciding (ω -)explorability for parity automata
- ▶ Threshold problems: Can Laetitia:
 - gather some proportion r of the tokens with probability 1?
 - gather all tokens with probability $\geq p$?

Thanks!