Controlling a population of identical NFA

Nathalie Bertrand

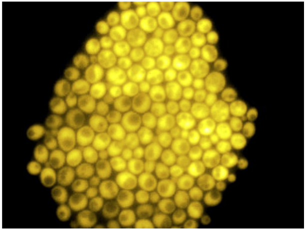
Inria Rennes

joint work with Miheer Dewaskar (ex CMI student), Blaise Genest (IRISA) and Hugo Gimbert (LaBRI)

SynCoP & PV workshops @ ETAPS 2018

Motivation

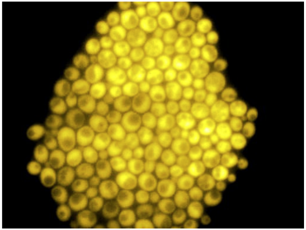
Control of gene expression for a population of cells



credits: G. Batt

Motivation

Control of gene expression for a population of cells

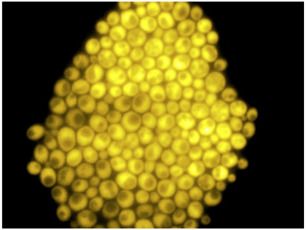


credits: G. Batt

- cell population
- gene expression monitored through fluorescence level
- drug injections affect all cells
- response varies from cell to cell
- obtain a large proportion of cells with desired gene expression level

Motivation

Control of gene expression for a population of cells



credits: G. Batt

- cell population
- gene expression monitored through fluorescence level
- drug injections affect all cells
- response varies from cell to cell
- obtain a large proportion of cells with desired gene expression level

- arbitrary nb of components
- full observation

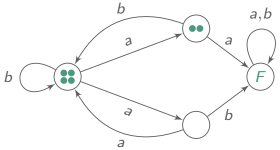
- uniform control
- non-det. model for single cell
- global reachability objective

Problem formalisation

- population of N identical NFA
- uniform control policy under full observation
- resolution of non-determinism by an adversary

Problem formalisation

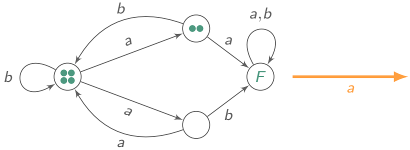
- population of N identical NFA
- uniform control policy under full observation
- resolution of non-determinism by an adversary



config: # copies in each state

Problem formalisation

- population of N identical NFA
- uniform control policy under full observation
- resolution of non-determinism by an adversary

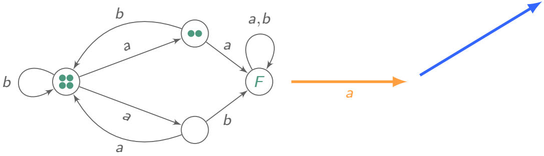


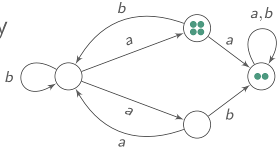
config: # copies in each state

- controller chooses the action (*e.g.* a)

Problem formalisation

- population of N identical NFA
- uniform control policy under full observation
- resolution of non-determinism by an adversary



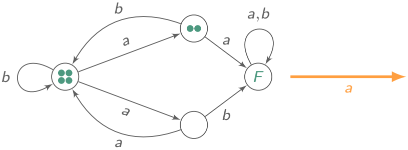


config: # copies in each state

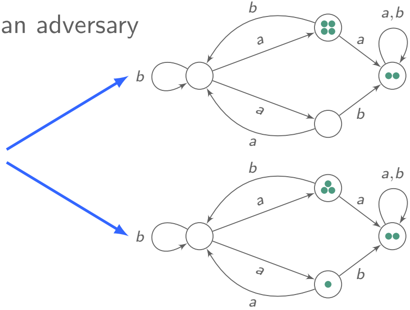
- controller chooses the action (*e.g.* a)
- adversary chooses how to move each individual copy (a-transition)

Problem formalisation

- population of N identical NFA
- uniform control policy under full observation
- resolution of non-determinism by an adversary



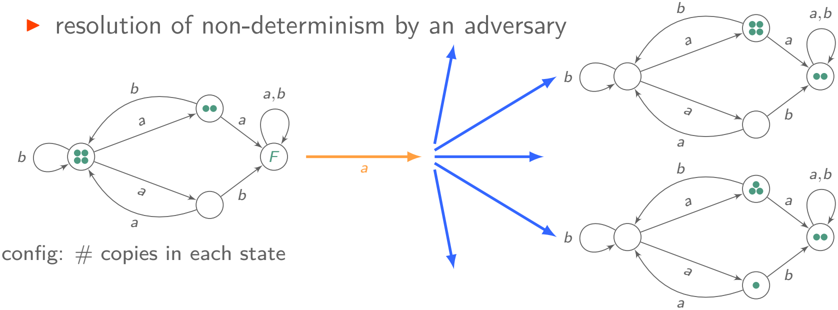
config: # copies in each state



- controller chooses the action (*e.g.* a)
- adversary chooses how to move each individual copy (a-transition)

Problem formalisation

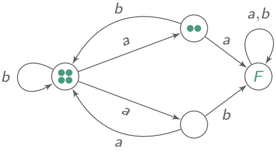
- population of N identical NFA
- uniform control policy under full observation
- resolution of non-determinism by an adversary



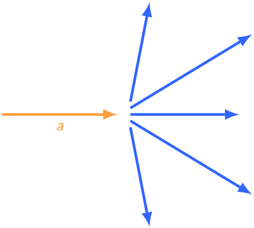
- controller chooses the action (*e.g.* a)
- adversary chooses how to move each individual copy (a-transition)

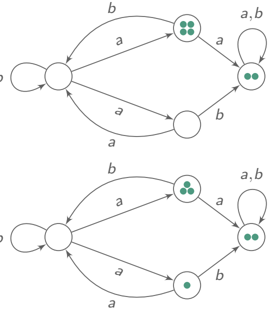
Problem formalisation

- population of N identical NFA
- uniform control policy under full observation
- resolution of non-determinism by an adversary



config: # copies in each state





- controller chooses the action (*e.g.* a)
- adversary chooses how to move each individual copy (a-transition)

Question can one control the population to ensure that for all non-deterministic choices all NFAs simultaneously reach a target set?

Population control

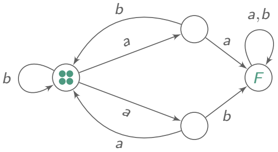
Fixed N: build finite 2-player game, identify global target states, decide if controller has a winning strategy for a reachability objective

Population control

Fixed N: build finite 2-player game, identify global target states, decide if controller has a winning strategy for a reachability objective

Challenge: **Parameterized control**

$$\forall N \ \exists \sigma \ \forall \tau \ (\mathcal{A}^N, \sigma, \tau) \models \Diamond F^N?$$

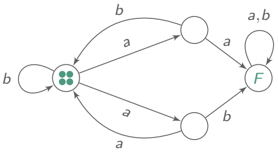


Population control

Fixed N: build finite 2-player game, identify global target states, decide if controller has a winning strategy for a reachability objective

Challenge: **Parameterized control**

$$\forall N \; \exists \sigma \; \forall \tau \; (\mathcal{A}^N, \sigma, \tau) \models \Diamond F^N?$$



This talk

- decidability and complexity
- memory requirements for controller σ
- admissible values for N

Monotonicity property and cutoff

Monotonicity property: the larger N, the harder for controller

$$\exists\sigma\ \forall\tau(\mathcal{A}^N,\sigma,\tau) \models \Diamond F^N \quad\Longrightarrow\quad \forall M \leq N\ \exists\sigma\ \forall\tau(\mathcal{A}^M,\sigma,\tau) \models \Diamond F^M$$

Monotonicity property and cutoff

Monotonicity property: the larger N, the harder for controller

$$\exists\sigma\ \forall\tau(\mathcal{A}^N,\sigma,\tau) \models \Diamond F^N \quad \Longrightarrow \quad \forall M \leq N\ \exists\sigma\ \forall\tau(\mathcal{A}^M,\sigma,\tau) \models \Diamond F^M$$

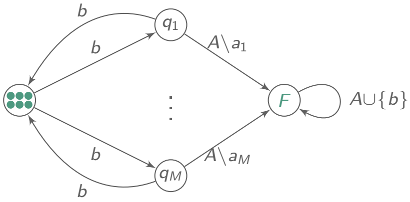
Cutoff: smallest N for which controller has no winning strategy

Monotonicity property and cutoff

Monotonicity property: the larger N, the harder for controller

$$\exists\sigma\ \forall\tau(\mathcal{A}^N,\sigma,\tau) \models \Diamond F^N \quad\Longrightarrow\quad \forall M \leq N\ \exists\sigma\ \forall\tau(\mathcal{A}^M,\sigma,\tau) \models \Diamond F^M$$

Cutoff: smallest N for which controller has no winning strategy



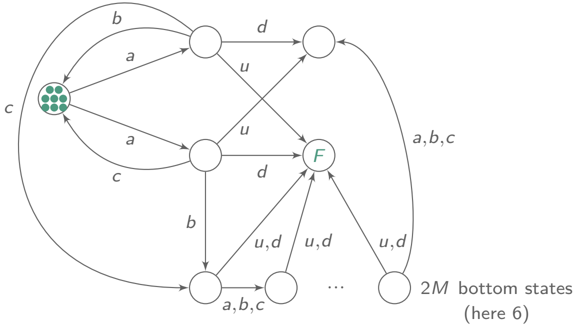
$A = \{a_1, \cdots, a_M\}$
unspecified edges lead to a sink state

winning σ if $N < M$
play b then a_i s.t. q_i is empty

winning τ for $N = M$
always fill all q_i's

cutoff is M

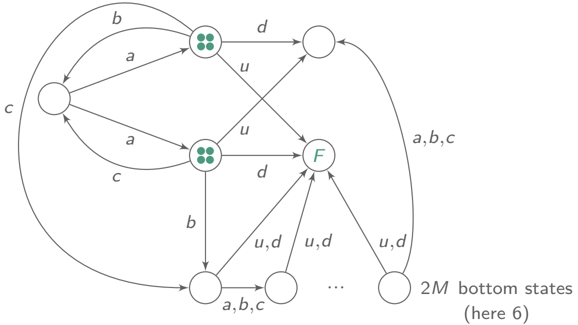
Lower bound on the cutoff



- $\forall N \leq 2^M, \ \exists \sigma, \ \mathcal{A}^N \models \forall_\sigma \Diamond F^N$
  accumulate copies in bottom states, then u/d to converge
- for $N > 2^M$ controller cannot avoid reaching the sink state

Cutoff $\mathcal{O}(2^{|\mathcal{A}|})$

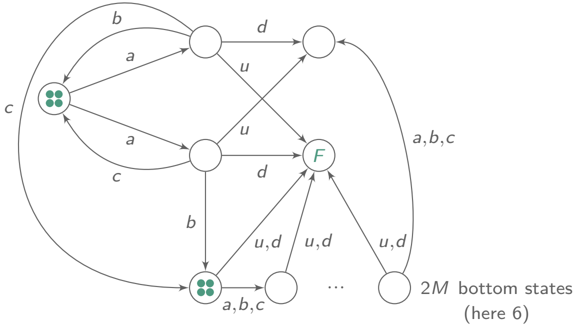
Lower bound on the cutoff



- $\forall N \leq 2^M,\ \exists \sigma,\ \mathcal{A}^N \models \forall_\sigma \Diamond F^N$
  accumulate copies in bottom states, then u/d to converge
- for $N > 2^M$ controller cannot avoid reaching the sink state

Cutoff $\mathcal{O}(2^{|\mathcal{A}|})$

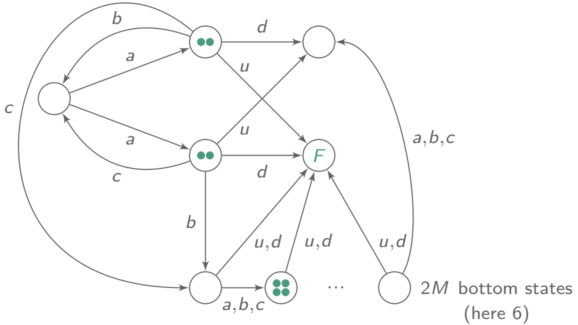
Lower bound on the cutoff



- $\forall N \leq 2^M,\ \exists \sigma,\ \mathcal{A}^N \models \forall_\sigma \Diamond F^N$
  accumulate copies in bottom states, then u/d to converge
- for $N > 2^M$ controller cannot avoid reaching the sink state

Cutoff $\mathcal{O}(2^{|\mathcal{A}|})$

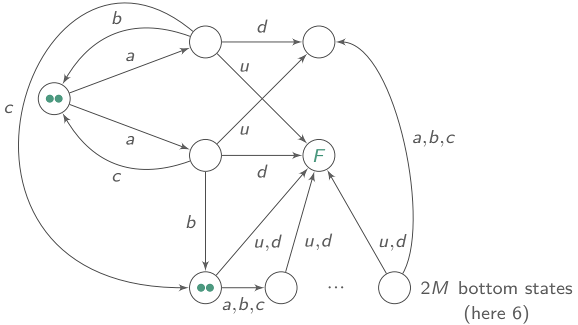
Lower bound on the cutoff



- $\forall N \leq 2^M,\ \exists \sigma,\ \mathcal{A}^N \models \forall_\sigma \Diamond F^N$
  accumulate copies in bottom states, then u/d to converge
- for $N > 2^M$ controller cannot avoid reaching the sink state

Cutoff $\mathcal{O}(2^{|\mathcal{A}|})$

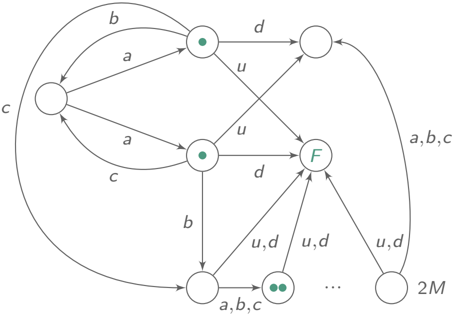
Lower bound on the cutoff



- $\forall N \leq 2^M,\ \exists \sigma,\ \mathcal{A}^N \models \forall_\sigma \Diamond F^N$
  accumulate copies in bottom states, then u/d to converge
- for $N > 2^M$ controller cannot avoid reaching the sink state

Cutoff $\mathcal{O}(2^{|\mathcal{A}|})$

Lower bound on the cutoff

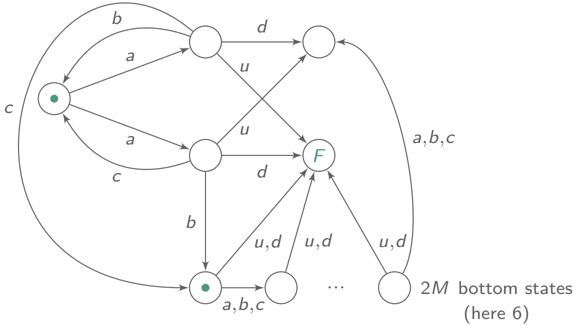


2M bottom states (here 6)

- $\forall N \leq 2^M, \ \exists \sigma, \ \mathcal{A}^N \models \forall_\sigma \Diamond F^N$
  accumulate copies in bottom states, then u/d to converge
- for $N > 2^M$ controller cannot avoid reaching the sink state

Cutoff $\mathcal{O}(2^{|\mathcal{A}|})$

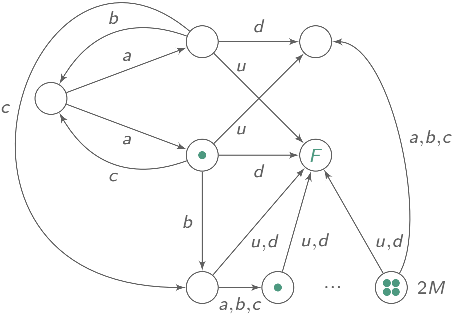
Lower bound on the cutoff



- $\forall N \leq 2^M,\ \exists \sigma,\ \mathcal{A}^N \models \forall_\sigma \Diamond F^N$
  accumulate copies in bottom states, then u/d to converge
- for $N > 2^M$ controller cannot avoid reaching the sink state

Cutoff $\mathcal{O}(2^{|\mathcal{A}|})$

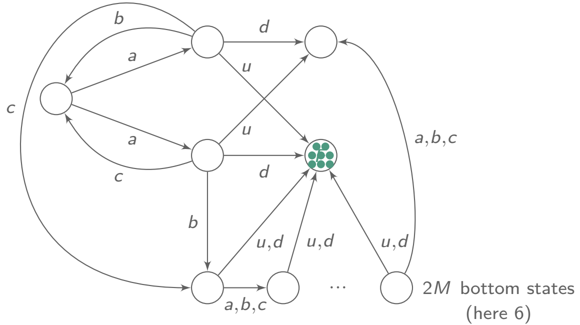
Lower bound on the cutoff



- $\forall N \leq 2^M,\ \exists \sigma,\ \mathcal{A}^N \models \forall_\sigma \Diamond F^N$
  accumulate copies in bottom states, then u/d to converge
- for $N > 2^M$ controller cannot avoid reaching the sink state

Cutoff $\mathcal{O}(2^{|\mathcal{A}|})$

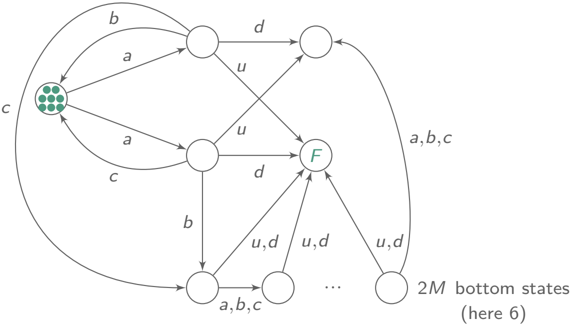
Lower bound on the cutoff



- $\forall N \leq 2^M,\ \exists \sigma,\ \mathcal{A}^N \models \forall_\sigma \Diamond F^N$
  accumulate copies in bottom states, then u/d to converge
- for $N > 2^M$ controller cannot avoid reaching the sink state

Cutoff $\mathcal{O}(2^{|\mathcal{A}|})$

Lower bound on the cutoff

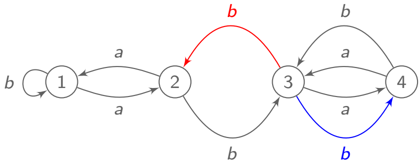


- $\forall N \leq 2^M,\ \exists\sigma,\ \mathcal{A}^N \models \forall_\sigma \Diamond F^N$
  accumulate copies in bottom states, then u/d to converge
- for $N > 2^M$ controller cannot avoid reaching the sink state

Cutoff $\mathcal{O}(2^{|\mathcal{A}|})$

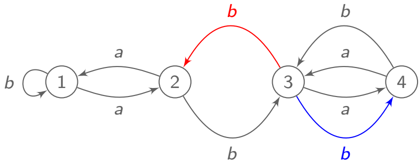
Combined with a counter, cutoff is even doubly exponential!

A natural attempt: the support game



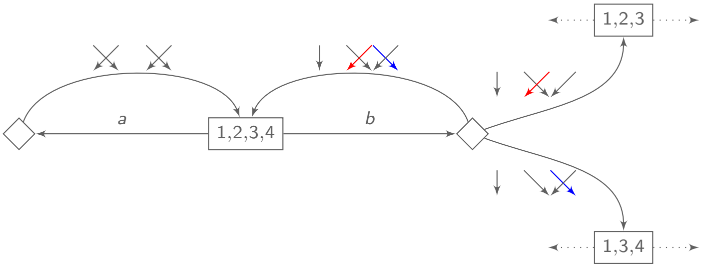
Assumption: if state 2 or 4 is empty, controller wins

A natural attempt: the support game

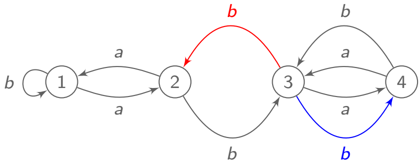


Assumption: if state 2 or 4 is empty, controller wins

Support game: □ Eve chooses action
◇ Adam chooses transfer graph (footprint of copies' moves)



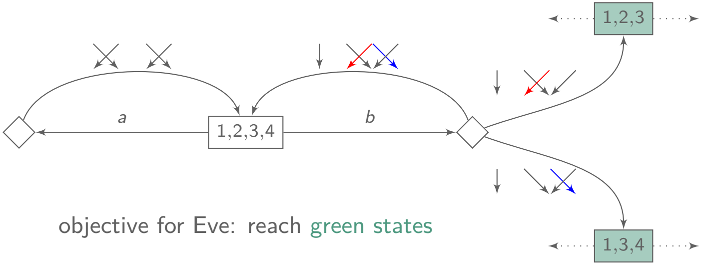
A natural attempt: the support game



Assumption: if state 2 or 4 is empty, controller wins

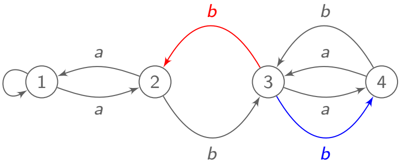
Support game: □ Eve chooses action

◇ Adam chooses transfer graph (footprint of copies' moves)



objective for Eve: reach green states

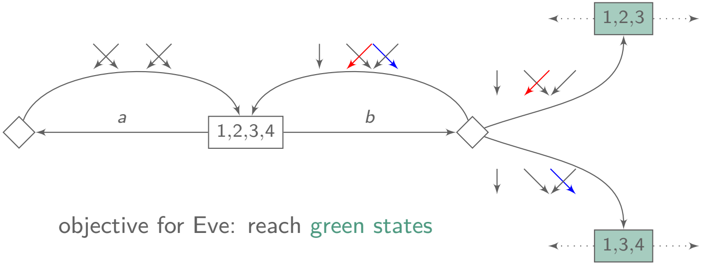
A natural attempt: the support game



Assumption: if state 2 or 4 is empty, controller wins

Support game: □ Eve chooses action

◇ Adam chooses transfer graph (footprint of copies' moves)

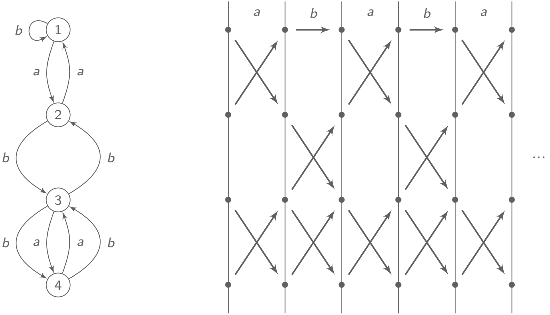


objective for Eve: reach green states

If Eve wins support game then controller has a winning strategy for all N

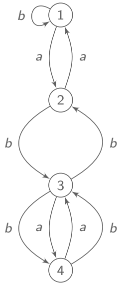
Support game is not equivalent to population game

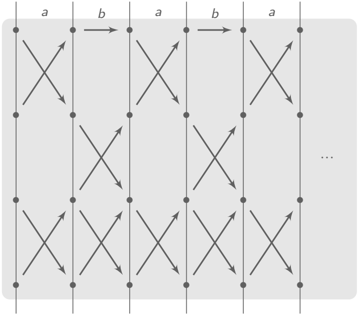
- controller alternates *a* and *b* ;
- adversary must always fill 2 and 4 in the *b*-step



Support game is not equivalent to population game

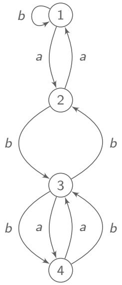
- controller alternates *a* and *b* ;
- adversary must always fill 2 and 4 in the *b*-step

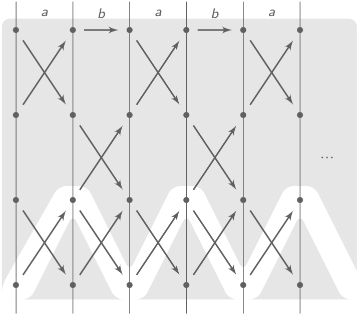




Support game is not equivalent to population game

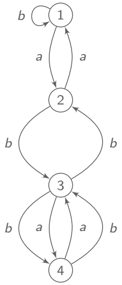
- controller alternates *a* and *b* ;
- adversary must always fill 2 and 4 in the *b*-step

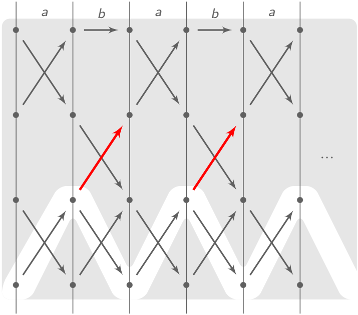




Support game is not equivalent to population game

- controller alternates *a* and *b* ;
- adversary must always fill 2 and 4 in the *b*-step

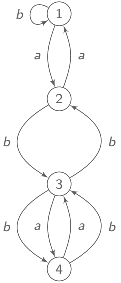


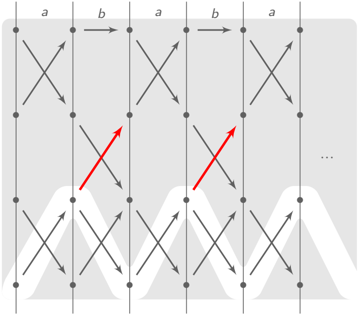


infinitely many leaks from the white flow

Support game is not equivalent to population game

- controller alternates *a* and *b* ;
- adversary must always fill 2 and 4 in the *b*-step



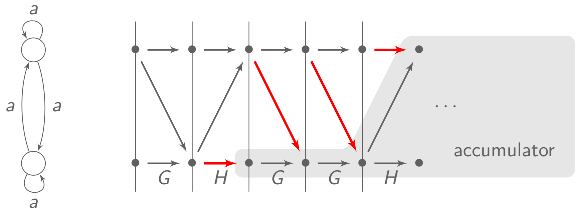


infinitely many leaks from the white flow

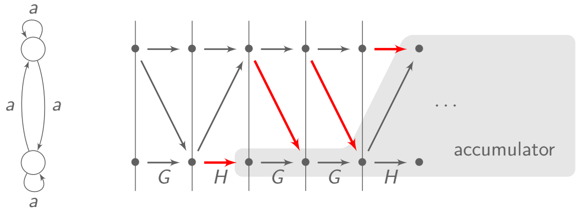
Above play from support game is not realisable in population control

- Controller wins with $(ab)^\omega$!
- Eve loses the support game

Capacity game: refining winning condition of support game

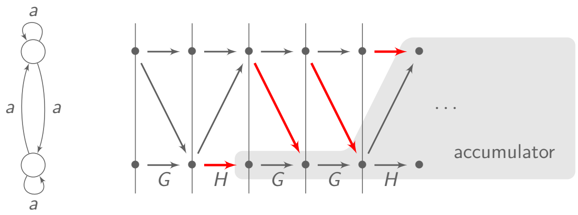


Capacity game: refining winning condition of support game



Finite capacity play: all accumulators have finitely many entries
Bounded capacity play: finite bound on # entries for accumulators

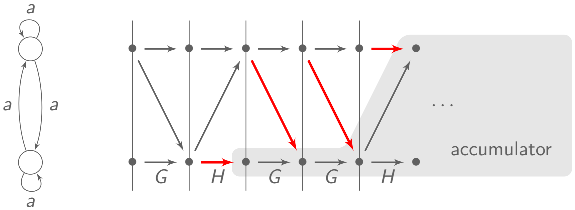
Capacity game: refining winning condition of support game



Finite capacity play: all accumulators have finitely many entries
Bounded capacity play: finite bound on # entries for accumulators

Bounded capacity
- corresponds to realizable plays
- does not seem to be regular

Capacity game: refining winning condition of support game

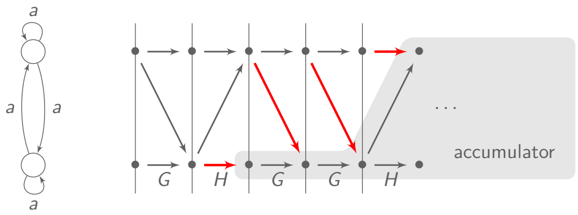


Finite capacity play: all accumulators have finitely many entries
Bounded capacity play: finite bound on # entries for accumulators

Bounded capacity
- corresponds to realizable plays
- does not seem to be regular

Capacity game: Eve wins a play if either it reaches a subset of F, or it does not have finite capacity.

Capacity game: refining winning condition of support game



Finite capacity play: all accumulators have finitely many entries
Bounded capacity play: finite bound on # entries for accumulators

Bounded capacity
- corresponds to realizable plays
- does not seem to be regular

Capacity game: Eve wins a play if either it reaches a subset of F, or it does not have finite capacity.

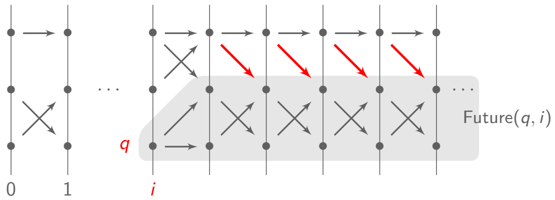
Eve wins capacity game iff Controller has a winning strategy for all N

Solving the capacity game in 2EXPTIME

The set of plays with infinite capacity is ω-regular

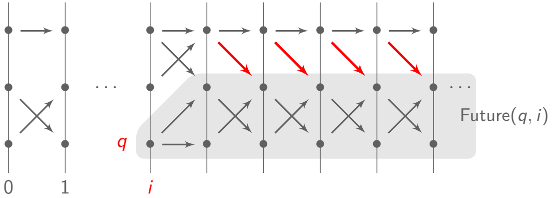
Solving the capacity game in 2EXPTIME

The set of plays with infinite capacity is ω-regular



Solving the capacity game in 2EXPTIME

The set of plays with infinite capacity is ω-regular

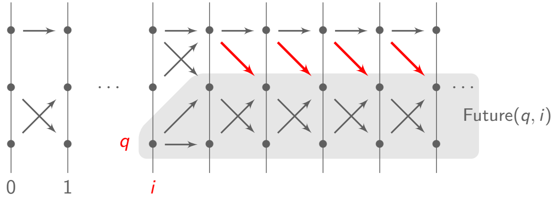


Non-deterministic Büchi automaton

1. guesses a step i, and state q
2. checks that the accumulator Future(q, i) has infinitely many entries

Solving the capacity game in 2EXPTIME

The set of plays with infinite capacity is ω-regular



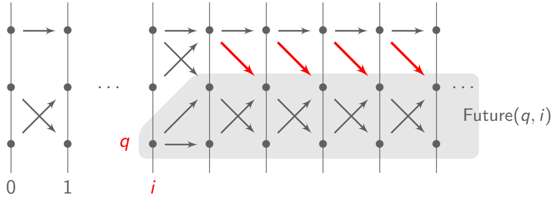
Non-deterministic Büchi automaton

1. guesses a step i, and state q
2. checks that the accumulator Future(q, i) has infinitely many entries

- Non-det. Büchi determinized into det. parity automaton
- Resolution of doubly exp. parity game

Solving the capacity game in 2EXPTIME

The set of plays with infinite capacity is ω-regular



Non-deterministic Büchi automaton

1. guesses a step i, and state q
2. checks that the accumulator Future(q, i) has infinitely many entries

- Non-det. Büchi determinized into det. parity automaton
- Resolution of doubly exp. parity game

2EXPTIME decision procedure in the size of NFA $\mathcal{A}$

Solving the capacity game in EXPTIME

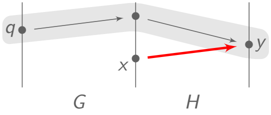
Ad-hoc deterministic parity automaton with

$\#$ states $=$ simply exponential in $|\mathcal{A}|$ $\quad$ $\#$ priorities $=$ polynomial in $|\mathcal{A}|$

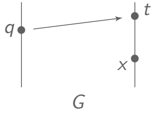
Solving the capacity game in EXPTIME

Ad-hoc deterministic parity automaton with

$\#$ states $=$ simply exponential in $|\mathcal{A}|$ $\#$ priorities $=$ polynomial in $|\mathcal{A}|$



$x \to y$ enters accumulator Future(q)



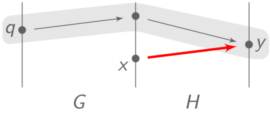
G separates pair (t, x)

- entries arise from separated pairs
- tracking transfer graphs separating new pairs is sufficient

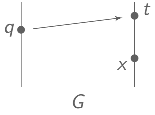
Solving the capacity game in EXPTIME

Ad-hoc deterministic parity automaton with

$\#$ states $=$ simply exponential in $|\mathcal{A}|$ $\#$ priorities $=$ polynomial in $|\mathcal{A}|$



$x \to y$ enters accumulator Future(q)



G separates pair (t, x)

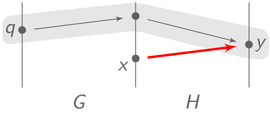
- entries arise from separated pairs
- tracking transfer graphs separating new pairs is sufficient

Parity game:
capacity game enriched with tracking lists in states
priorities reflect how the tracking list evolves (removals, shifts, etc.)

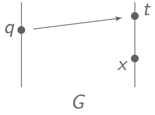
Solving the capacity game in EXPTIME

Ad-hoc deterministic parity automaton with

$\#$ states = simply exponential in $|\mathcal{A}|$ $\#$ priorities = polynomial in $|\mathcal{A}|$



$x \to y$ enters accumulator Future(q)



G separates pair (t, x)

- entries arise from separated pairs
- tracking transfer graphs separating new pairs is sufficient

Parity game:
capacity game enriched with tracking lists in states
priorities reflect how the tracking list evolves (removals, shifts, etc.)

Parity game is equivalent to capacity game.

Complexity of the population control problem

Theorem:
The population control problem is EXPTIME-complete.

Upper bound :

- population control problem $\equiv$ capacity game
- capacity game $\equiv$ ad hoc parity game
- solving parity game of size exp. and poly. priorities

Lower bound : encoding of poly space alternating Turing machine

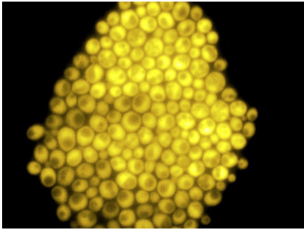
Summary of results

Uniform control of a population of identical NFA

- parameterized control problem: gather all copies in F
- (surprisingly) quite involved!
- tight results for complexity, cutoff, and memory
  - complexity: EXPTIME-complete decision problem
  - bound on cutoff: doubly exponential
  - memory requirement: exponential memory (orthogonal to supports) is needed and sufficient for controller

Back to motivations

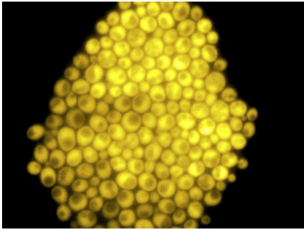
Control of gene expression for a population of cells



credits: G. Batt

Back to motivations

Control of gene expression for a population of cells



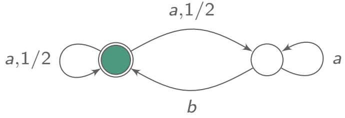
credits: G. Batt

- need for truely probabilistic model
  $\rightarrow$ MDP instead of NFA
- need for truely quantitative questions
  $\rightarrow$ proportions and probabilities instead of convergence and (almost)-sure

$$\forall N \max_{\sigma} \mathbb{P}_\sigma(\mathcal{A}^N \models \Diamond \text{ at least 80\% of MDPs in } F\ ) \geq .7?$$

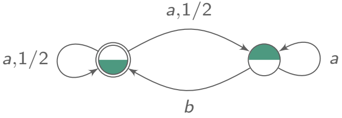
Probabilistic population

Discrete approximation of probabilistic automata



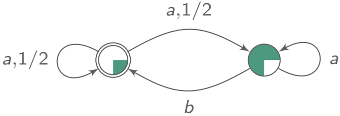
Probabilistic population

Discrete approximation of probabilistic automata



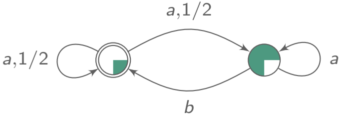
Probabilistic population

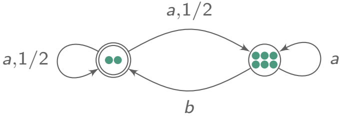
Discrete approximation of probabilistic automata



Probabilistic population

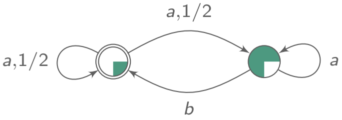
Discrete approximation of probabilistic automata

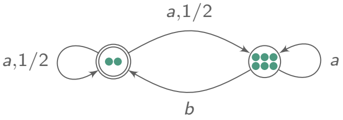




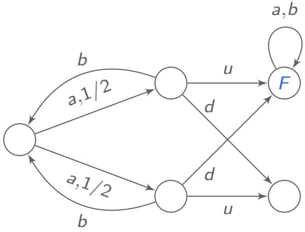
Probabilistic population

Discrete approximation of probabilistic automata



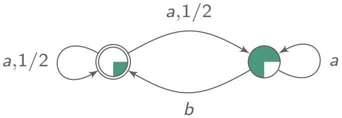


Gap: optimal reachability probability not continuous when $N \to \infty$

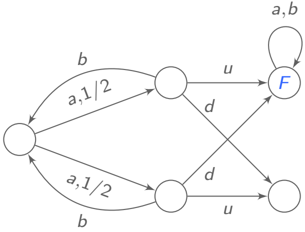


Probabilistic population

Discrete approximation of probabilistic automata



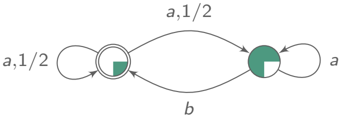
Gap: optimal reachability probability not continuous when $N \to \infty$

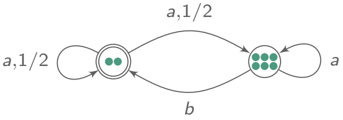


- $\forall N, \exists \sigma,\ \mathbb{P}_\sigma(\Diamond F^N) = 1.$
- In the PA, the maximum probability to reach F is .5.

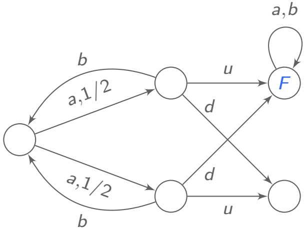
Probabilistic population

Discrete approximation of probabilistic automata





Gap: optimal reachability probability not continuous when $N \to \infty$



- $\forall N, \exists \sigma,\ \mathbb{P}_\sigma(\Diamond F^N) = 1$.
- In the PA, the maximum probability to reach F is .5.

Good news? hope for alternative **more tractable** semantics for PA

$\epsilon\nu\chi\alpha\rho\iota\sigma\tau\acute{\omega}$!