Computational complexity of majority automata under different updating schemes

Pedro Montealegre [1] Eric Goles[2]

[1]Laboratoire d'Informatique Fondamentale d'Orlans
Université d'Orléans, Orléans, France

[2]Facultad de Ciencias y Tecnología
Universidad Adolfo Ibáñez, Santiago, Chile

September 19, 2013

Automata network

An *Automata Network* is a triple $\mathcal{A} = (G, Q, f_i : i \in V)$, where

- $G = (V, E)$ is a simple undirected graph and $V = \{1, \ldots, n\}$.
- Q the set of states ($Q = \{0, 1\}$)
- $f_i : \{0, 1\}^n \to \{0, 1\}$ is the transition function associated to the vertex i.

We say that vertices in state 1 are *active* while vertices in state 0 are *passive*.

Updating Schemes

An *updating scheme* (US) of the automaton $\mathcal{A}$ is a function

$$\phi : V \to \{1 \ldots |V|\}$$

st. if u and v are vertices and $\phi(u) < \phi(v)$ then the state of u is updated before v, and if $\phi(u) = \phi(v)$ then nodes u and v are update at the same time.

Types of US

Types of US

- **Synchronous:** $\phi = 1$.
  (All vertices are updated at the same time.)

Types of US

- **Synchronous:** $\phi = 1$.
  (All vertices are updated at the same time.)
- **Sequential:** $\phi = \sigma$, where σ is a permutation of V.
  (One vertex at a time)

Types of US

- **Synchronous:** $\phi = 1$.
  (All vertices are updated at the same time.)
- **Sequential:** $\phi = \sigma$, where σ is a permutation of V.
  (One vertex at a time)
- **Block sequential:**

$$V = \cup_{i=1}^{k} V_i, \qquad \cap_{i=1}^{k} V_i = \emptyset, \qquad \phi|_{V_i} = i$$

  The vertex set is partitioned into several subsets, st. into the same set every vertex is updated at the same time, and different subsets are updated sequentially in some order.

Trajectory of a configuration

Let $x \in \{0,1\}^n$ be a configuration of an automaton. The trajectory $T^{\phi}(x)$ of x with the updating scheme ϕ is the set

$$T^{\phi}(x) = \{x(t) : t \geq 0\}$$

where $x(0) = x$ and $x(t+1)$ is obtained from $x(t)$ after every vertex is updated according to ϕ.

The trajectory of x *enters in a limit cycle of period* p if $|T(x(t))| = p$ for some $t \geq 0$.
(A cycle of period 1 is a *fixed point*.)

The trajectory of x *enters in a limit cycle of period p* if $|T(x(t))| = p$ for some $t \geq 0$.
(A cycle of period 1 is a *fixed point*.)

There are at most 2^n different configurations (finite graph), then the trajectory of any configuration eventually enters to a limit cycle for any US. (Steady state)

The trajectory of x *enters in a limit cycle of period p* if $|T(x(t))| = p$ for some $t \geq 0$.
(A cycle of period 1 is a *fixed point*.)

There are at most 2^n different configurations (finite graph), then the trajectory of any configuration eventually enters to a limit cycle for any US. (Steady state)

$\tau_\phi(x)$: steps to reach the steady state starting from x with a US ϕ.

The trajectory of x *enters in a limit cycle of period* p if $|T(x(t))| = p$ for some $t \geq 0$.
(A cycle of period 1 is a *fixed point*.)

There are at most 2^n different configurations (finite graph), then the trajectory of any configuration eventually enters to a limit cycle for any US. (Steady state)

$\tau_\phi(x)$: steps to reach the steady state starting from x with a US ϕ.

$\tau_\phi(\mathcal{A}) = \max\{\tau_\phi(x) : x \in \{0,1\}^n\}$ is the transient length of $\mathcal{A}$.

Decision Problem

One Cell Prediction: OCP

Decision Problem

One Cell Prediction: OCP

Given:

Decision Problem

One Cell Prediction: OCP

Given:

- An automaton $\mathcal{A} = (G, \{0, 1\}, (f_i : i \in V))$,

Decision Problem

One Cell Prediction: OCP

Given:

- An automaton $\mathcal{A} = (G, \{0, 1\}, (f_i : i \in V))$,
- $x \in \{0, 1\}^n$ a configuration of $\mathcal{A}$,

Decision Problem

One Cell Prediction: OCP

Given:

- An automaton $\mathcal{A} = (G, \{0, 1\}, (f_i : i \in V))$,
- $x \in \{0, 1\}^n$ a configuration of $\mathcal{A}$,
- ϕ an updating scheme of $\mathcal{A}$,

Decision Problem

One Cell Prediction: OCP

Given:

- An automaton $\mathcal{A} = (G, \{0,1\}, (f_i : i \in V))$,
- $x \in \{0,1\}^n$ a configuration of $\mathcal{A}$,
- ϕ an updating scheme of $\mathcal{A}$,
- and $v \in V$ a vertex initially passive ($x_v = 0$),

Decision Problem

One Cell Prediction: OCP

Given:

- An automaton $\mathcal{A} = (G, \{0,1\}, (f_i : i \in V))$,
- $x \in \{0,1\}^n$ a configuration of $\mathcal{A}$,
- ϕ an updating scheme of $\mathcal{A}$,
- and $v \in V$ a vertex initially passive ($x_v = 0$),

Does there exists $y \in T^{\phi}(x)$ such that $y_v = 1$?

Majority automata

Here we will consider only *majority functions*, i.e.:

$$f_i(x) = \begin{cases} 1 & \text{if} \quad \sum_{j \in N(i)} x_i > \frac{|N(i)|}{2} \\ 0 & \text{if} \quad \sum_{j \in N(i)} x_i \leq \frac{|N(i)|}{2} \end{cases}$$

where $N(i)$ is the set of neighbors of vertex i.

An automata network with this rule is called a *majority automata*.

Parallel and sequential US.

Theorem

For parallel and sequential updating schemes, **OCP** *is in* **P**

Parallel and sequential US.

Theorem

For parallel and sequential updating schemes, **OCP** *is in* **P**

Idea: Simulate $\mathcal{A}$ until v changes.

Parallel and sequential US.

Theorem

For parallel and sequential updating schemes, **OCP** *is in* **P**

Idea: Simulate $\mathcal{A}$ until v changes.

For a configuration $x(t)$

Parallel and sequential US.

Theorem

For parallel and sequential updating schemes, **OCP** *is in* **P**

Idea: Simulate $\mathcal{A}$ until v changes.

For a configuration $x(t)$

- For any $i \in V$, $x_i(t+1)$ can be computed in $\mathcal{O}(n)$ time.

Parallel and sequential US.

Theorem

For parallel and sequential updating schemes, **OCP** *is in* **P**

Idea: Simulate $\mathcal{A}$ until v changes.

For a configuration $x(t)$

- For any $i \in V$, $x_i(t+1)$ can be computed in $\mathcal{O}(n)$ time.
- $x(t+1)$ can be computed in $\mathcal{O}(n^2)$ time.

Parallel and sequential US.

Theorem

For parallel and sequential updating schemes, **OCP** *is in* **P**

Idea: Simulate $\mathcal{A}$ until v changes.

For a configuration $x(t)$

- For any $i \in V$, $x_i(t+1)$ can be computed in $\mathcal{O}(n)$ time.
- $x(t+1)$ can be computed in $\mathcal{O}(n^2)$ time.

$|T(x)| = |\{x(t) \ : \ t > 0\}|$ is $poly(n)$?

[E. Goles, F. Fogelman, D. Pellegrin]

If $\{f_1, \dots, f_n\}$ are threshold functions with weights matrix A and threshold vector b.

$$E_{syn}[x(t)] = -\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} x_i(t-1) x_i(t) + \sum_{i=1}^{n} b_i (x_i(t) + x_i(t-1))$$

$$E_{seq}[x(t)] = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} x_i(t) x_j(t) + \sum_{i \in V} b_i x_i(t)$$

[E. Goles, F. Fogelman, D. Pellegrin]

[E. Goles, F. Fogelman, D. Pellegrin]

- $|E(x)|$ is $\mathcal{O}(n^2)$

[E. Goles, F. Fogelman, D. Pellegrin]

- $|E(x)|$ is $\mathcal{O}(n^2)$
- $\Delta_t E = E[x(t+1)] - E[x(t)] \leq 0$

[E. Goles, F. Fogelman, D. Pellegrin]

- $|E(x)|$ is $\mathcal{O}(n^2)$
- $\Delta_t E = E[x(t+1)] - E[x(t)] \leq 0$
  (E constant in cycles)

[E. Goles, F. Fogelman, D. Pellegrin]

- $|E(x)|$ is $\mathcal{O}(n^2)$
- $\Delta_t E = E[x(t+1)] - E[x(t)] \leq 0$
  (E constant in cycles)
- Synchronous US $\rightarrow$ reach at most cycles of length 2.
  Sequential US $\rightarrow$ reach only fixed points.

[E. Goles, F. Fogelman, D. Pellegrin]

- $|E(x)|$ is $\mathcal{O}(n^2)$
- $\Delta_t E = E[x(t+1)] - E[x(t)] \leq 0$
  (E constant in cycles)
- Synchronous US $\rightarrow$ reach at most cycles of length 2.
  Sequential US $\rightarrow$ reach only fixed points.
- $\tau(\mathcal{A})$ is $\mathcal{O}(n^3)$

Block sequential US

Block sequential US

Theorem

There is a block sequential update scheme in a majority automata, such that each block has cardinality 2 and the limit cycle has a super-polynomial length.

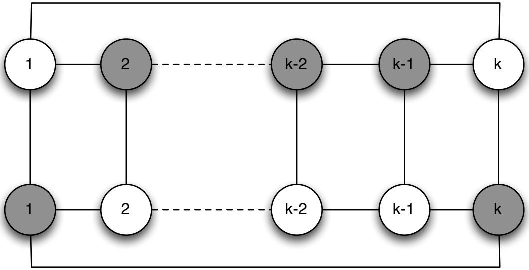
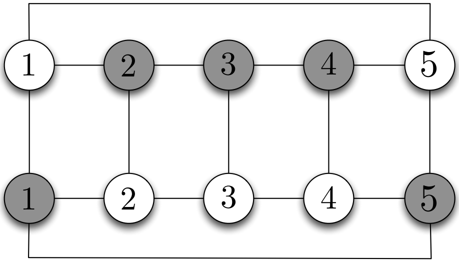
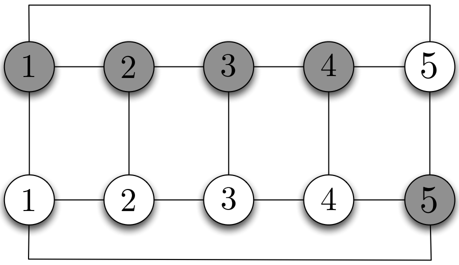


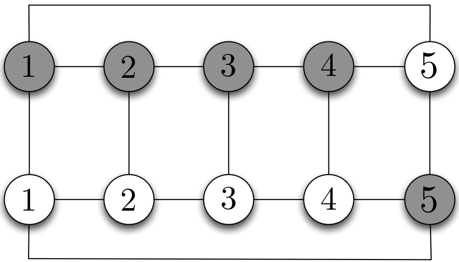
Figure: Ladder



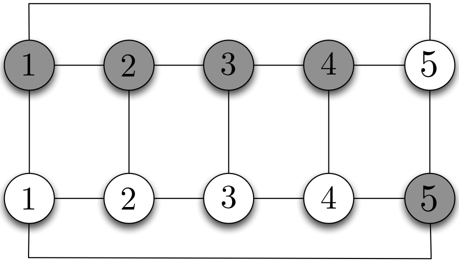
$t = 0$



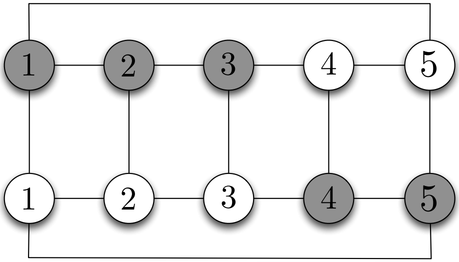
$t = 0 - 1$



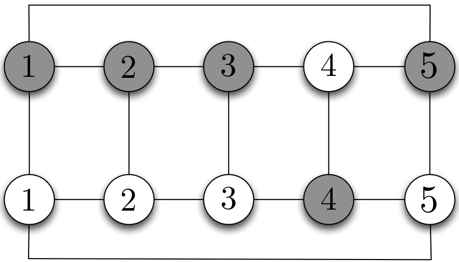
$t = 0 - 2$



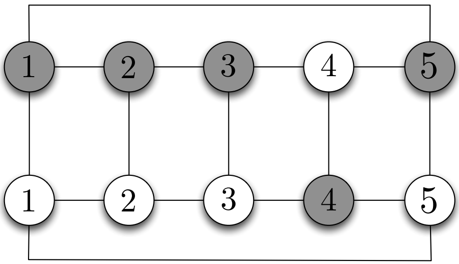
$$t = 0 - 3$$



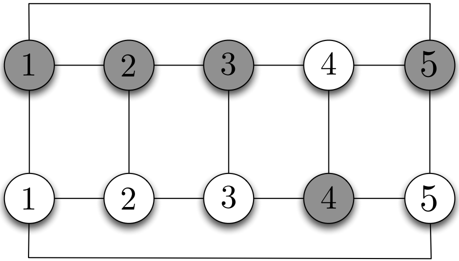
$t = 0 - 4$



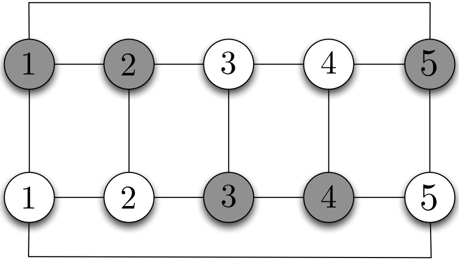
$$t = 0 - 5 = 1$$



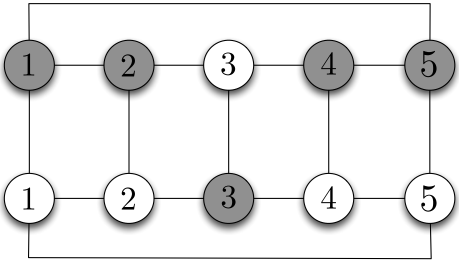
$$t = 1 - 1$$



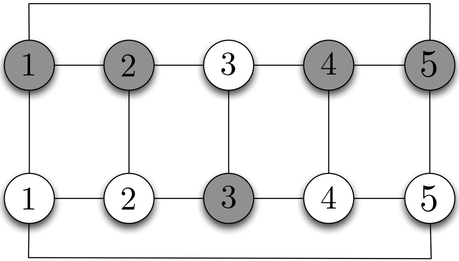
$t = 1 - 2$



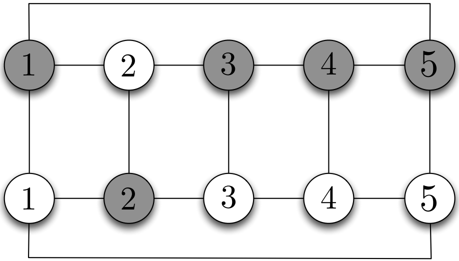
$t = 1 - 3$



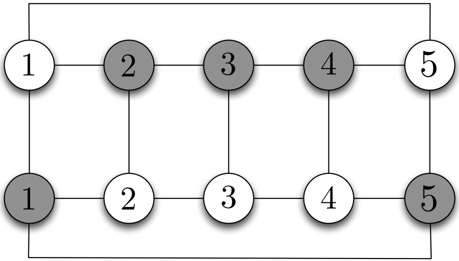
$t = 1 - 4$



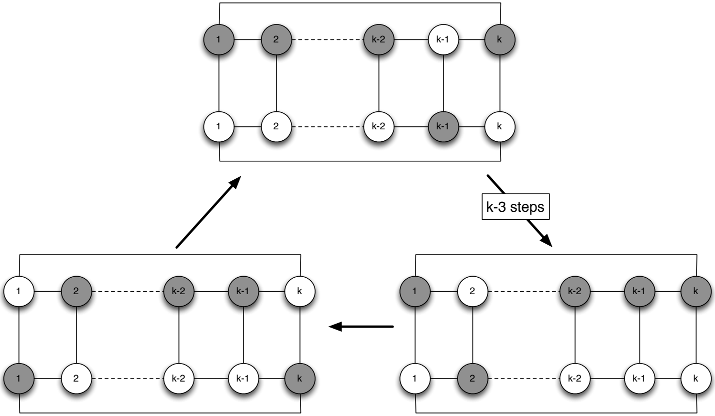
$$t = 1 - 5 = 2$$

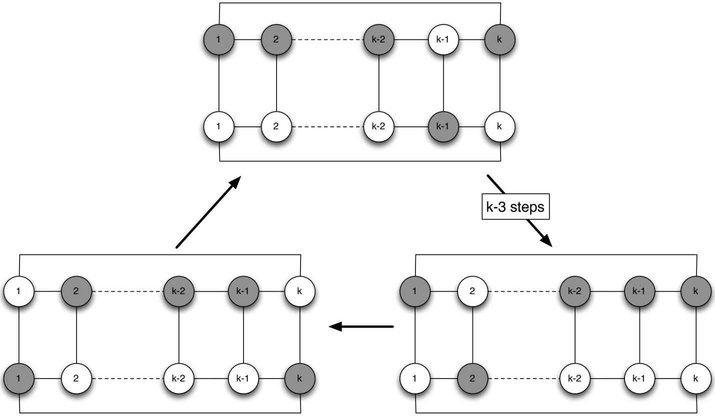


$t = 3$

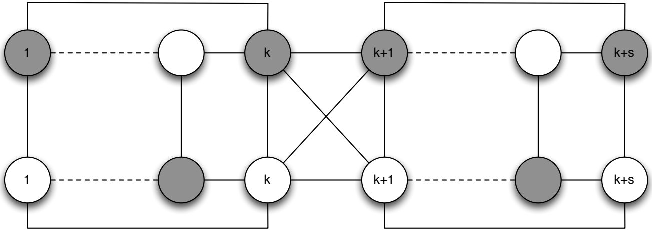


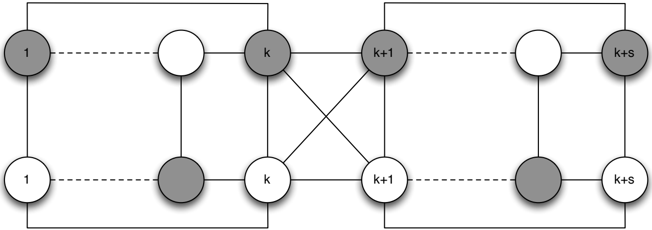
$t = 4$


1
2
k-2
k-1
k
k-3 steps



Limit cycle of length $k - 1$


1
k
k+1
k+s
1
k
k+1
k+s



Limit cycle of length $lcm(k-1, s-1)$

Let m be a positive integer, and let $\pi(m)$ the number of primes not exceeding m.

Let m be a positive integer, and let $\pi(m)$ the number of primes not exceeding m.

Let G the graph obtained from $\pi(m)$ ladders of sizes $(p_1 + 1), (p_2 + 1), \ldots, (p_{\pi(m)} + 1)$, where $\{p_1, p_2 \ldots, p_{\pi(m)}\}$ the first $\pi(m)$ primes.

Let m be a positive integer, and let $\pi(m)$ the number of primes not exceeding m.

Let G the graph obtained from $\pi(m)$ ladders of sizes $(p_1 + 1), (p_2 + 1), \ldots, (p_{\pi(m)} + 1)$, where $\{p_1, p_2 \ldots, p_{\pi(m)}\}$ the first $\pi(m)$ primes.

Then

Let m be a positive integer, and let $\pi(m)$ the number of primes not exceeding m.

Let G the graph obtained from $\pi(m)$ ladders of sizes $(p_1 + 1), (p_2 + 1), \ldots, (p_{\pi(m)} + 1)$, where $\{p_1, p_2 \ldots, p_{\pi(m)}\}$ the first $\pi(m)$ primes.

Then

$$V(G) \leq \sum_{i=1}^{\pi(m)} 2(p_i + 1) \leq 2\pi(m)(m+1)$$

$$\text{limit cycle of } G = lcm(p_1, \ldots, p_{\pi(m)}) = \prod_{i=1}^{\pi(m)} p_i = e^{\theta(m)}$$

where $\theta(m) = \sum_{i=1}^{\pi(m)} \log(pi)$.

Let m be a positive integer, and let $\pi(m)$ the number of primes not exceeding m.

Let G the graph obtained from $\pi(m)$ ladders of sizes $(p_1 + 1), (p_2 + 1), \ldots, (p_{\pi(m)} + 1)$, where $\{p_1, p_2 \ldots, p_{\pi(m)}\}$ the first $\pi(m)$ primes.

Then

$$V(G) \leq \sum_{i=1}^{\pi(m)} 2(p_i + 1) \leq 2\pi(m)(m+1)$$

$$\text{limit cycle of } G = lcm(p_1, \ldots, p_{\pi(m)}) = \prod_{i=1}^{\pi(m)} p_i = e^{\theta(m)}$$

where $\theta(m) = \sum_{i=1}^{\pi(m)} \log(pi)$.

From the Prime Number Theorem:

$$lcm(p_1, \ldots, p_{\pi(m)}) \geq e^{\Omega(\sqrt{|V(G)| \log(|V(G)|}))}$$

For block sequential update schemes...
OCP is in **P**?

For block sequential update schemes...
OCP is in **P**? we don't think so...

For block sequential update schemes...
OCP is in **P**? we don't think so...

Clearly **OCP** is in **P**-SPACE.

For block sequential update schemes...
OCP is in **P**? we don't think so...

Clearly **OCP** is in **P**-SPACE.

Theorem

The problem **OCP** *is NP-Hard for block sequential updating schemes.*

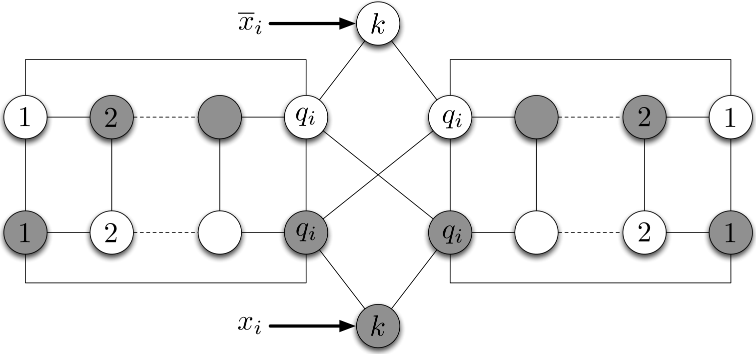
For block sequential update schemes...
OCP is in **P**? we don't think so...

Clearly **OCP** is in **P**-SPACE.

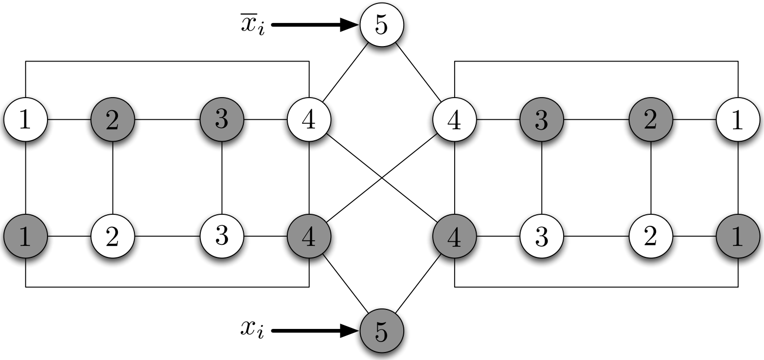
Theorem

The problem **OCP** *is NP-Hard for block sequential updating schemes.*

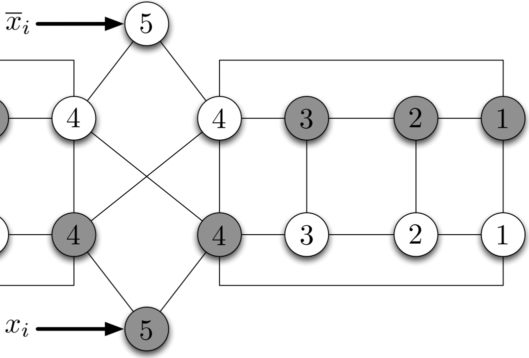
Proof: Reduce $3 - SAT$.



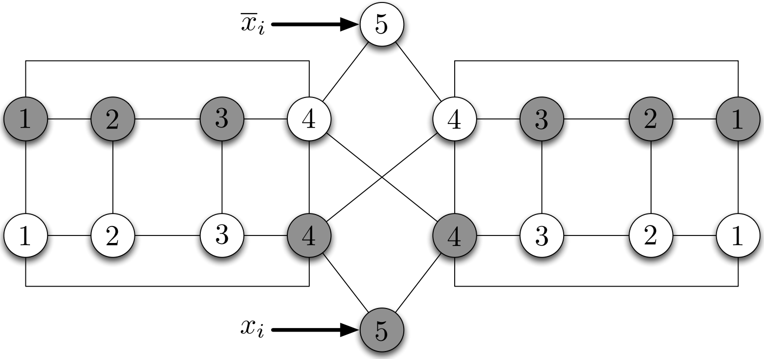
Gadget for variable x_i (positive and negative literals)



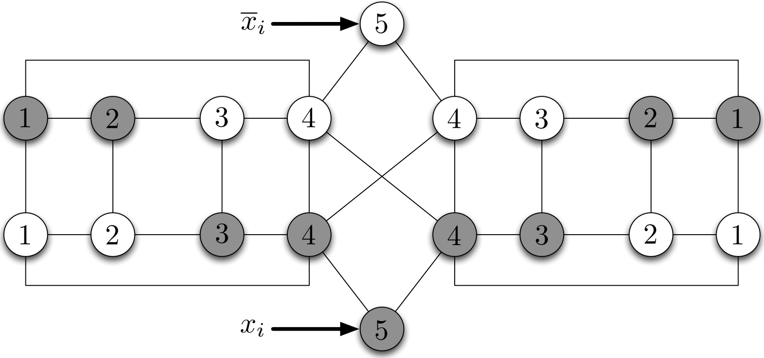
$t = 0$



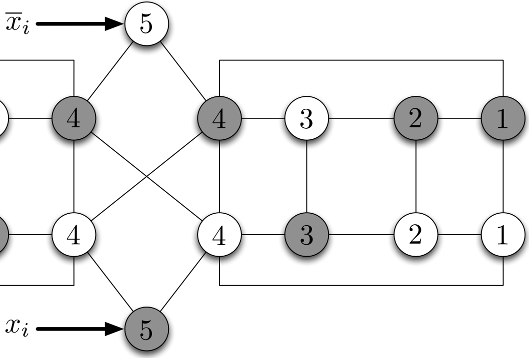
$t = 0 - 1$



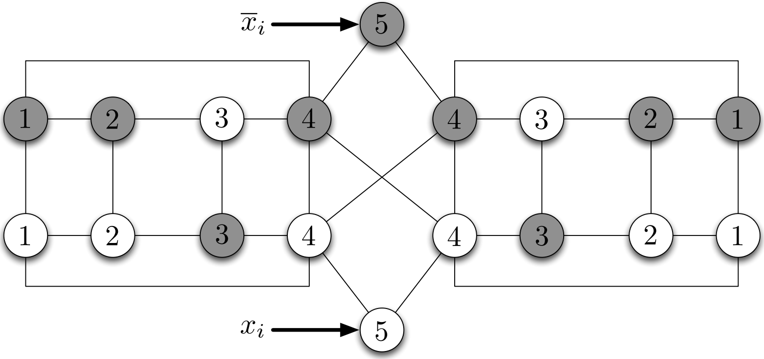
$t = 0 - 2$



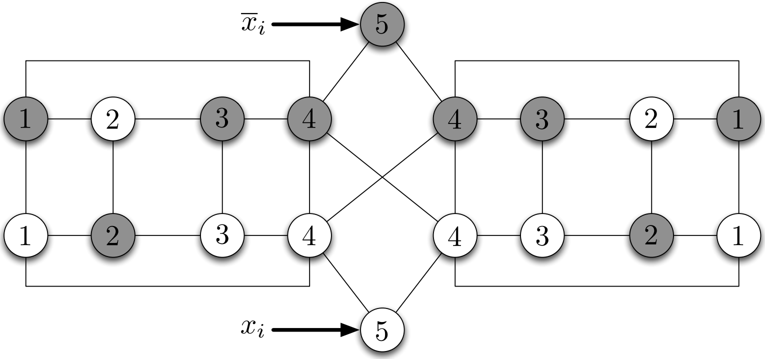
$t = 0 - 3$



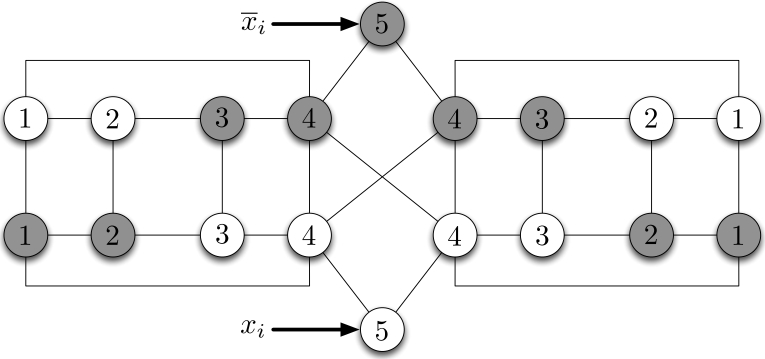
$t = 0 - 4$



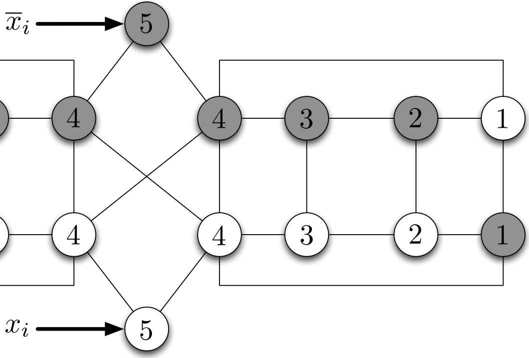
$$t = 0 - 5 = 1$$



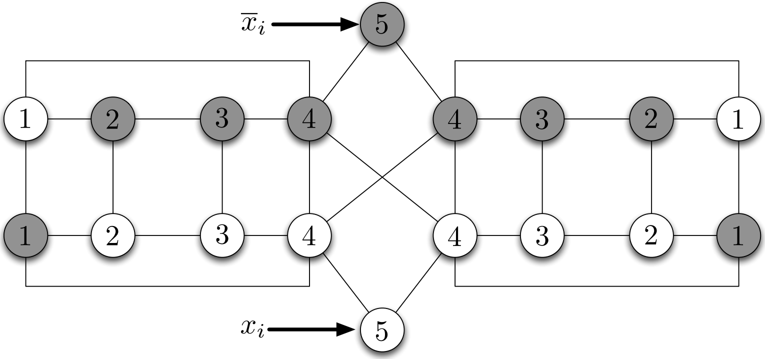
$t = 2$



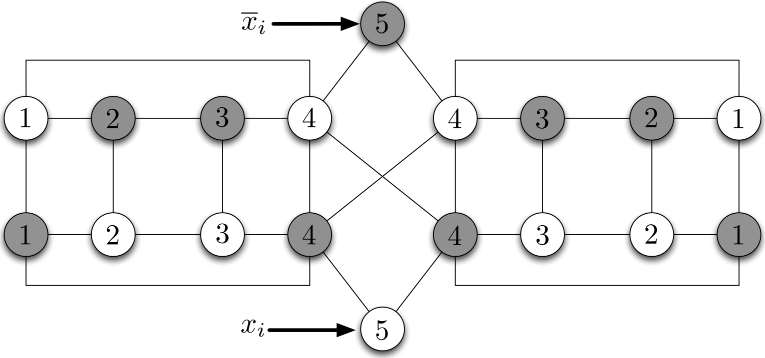
$t = 2 - 1$



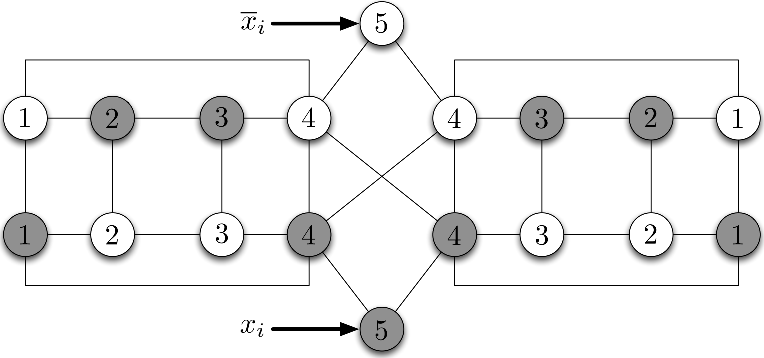
$t = 2 - 2$



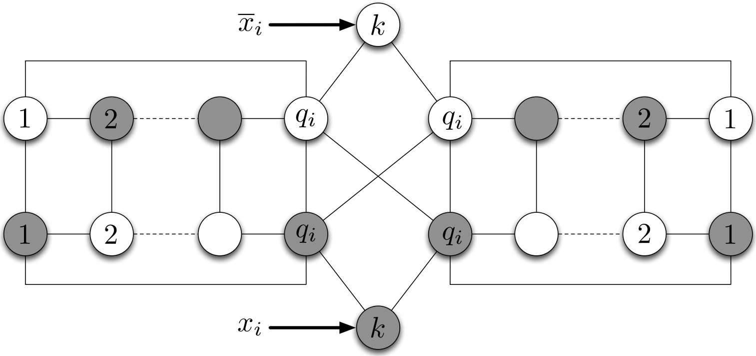
$t = 2 - 3$



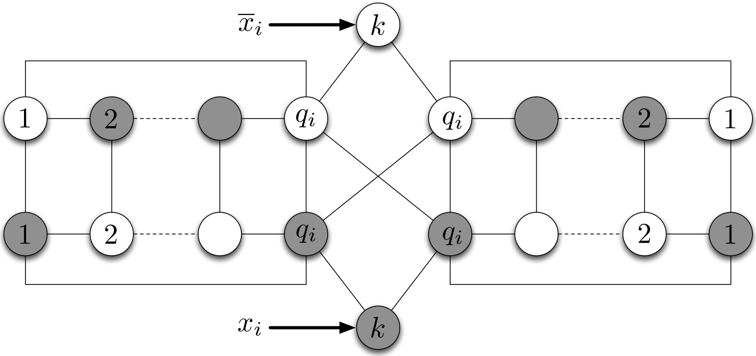
$t = 2 - 4$



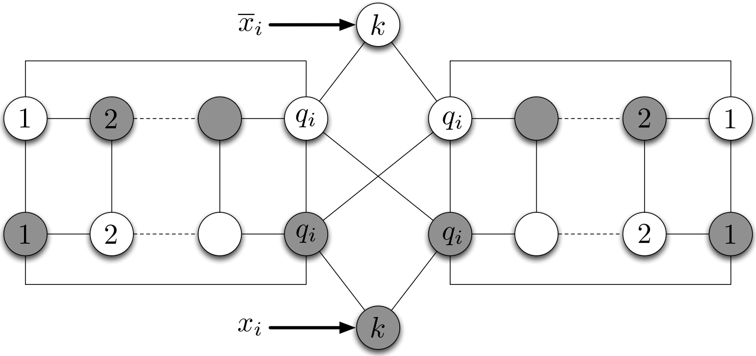
$t = 2 - 5 = 3$



$\overline{x}_i$
k
1
2
q_i
q_i
2
1
1
2
q_i
q_i
2
1
x_i
k

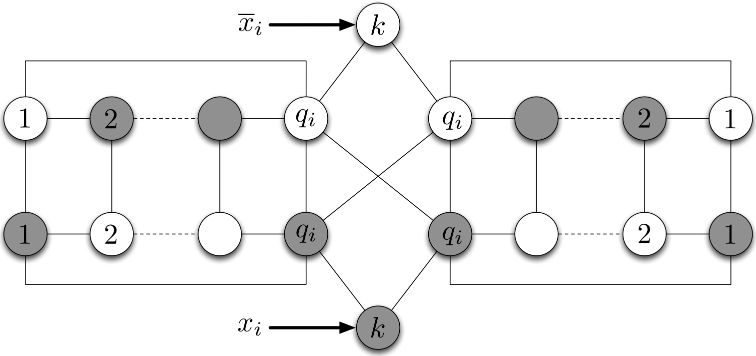


$q_i = p_i + 1$ where p_i i-th prime



$q_i = p_i + 1$ where p_i i-th prime

$$k = p_n + 2$$



$q_i = p_i + 1$ where p_i i-th prime

$k = p_n + 2$

$x_i = 1$ in steps multiple of p_i, and $x_i = 0$ otherwise

Then combination of values of variables $(x_1, x_2, x_3, \ldots, x_n)$, $x_i \in \{0, 1\}$ happens in step

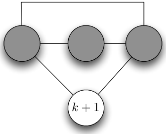
$$p_1^{x_1} \times p_2^{x_2} \times \cdots \times p_n^{x_n}$$

Then combination of values of variables $(x_1, x_2, x_3, \ldots, x_n)$, $x_i \in \{0, 1\}$ happens in step

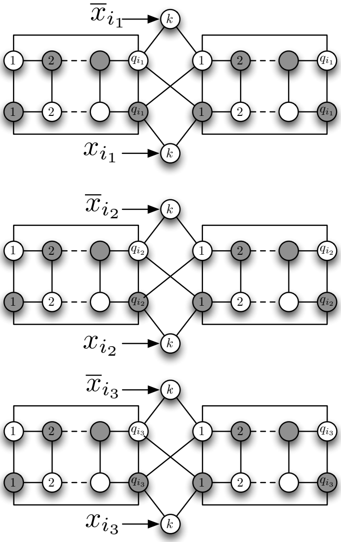
$$p_1^{x_1} \times p_2^{x_2} \times \cdots \times p_n^{x_n}$$

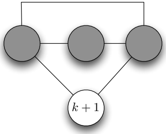
Any possible combination of input values can be simulated.

$$C_i = (x_{i_1} \vee \overline{x}_{i_2} \vee \overline{x}_{i_3})$$

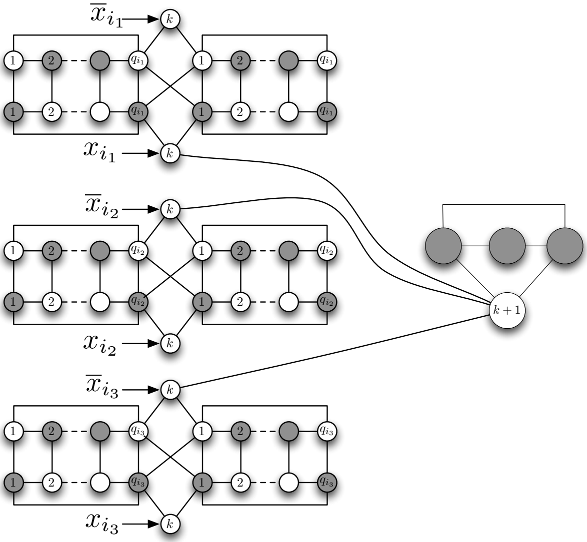


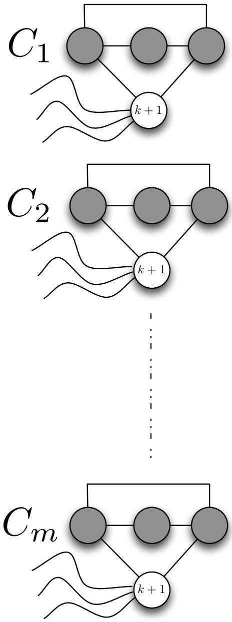
$C_i = (x_{i_1} \vee \overline{x}_{i_2} \vee \overline{x}_{i_3})$



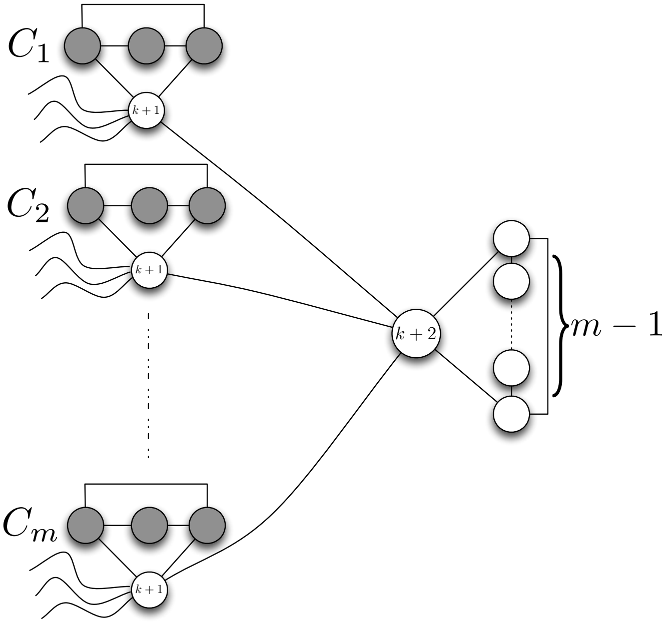


$$C_i = (x_{i_1} \vee \overline{x}_{i_2} \vee \overline{x}_{i_3})$$





C_1
$k+1$
C_2
$k+1$
C_m
$k+1$



C_1
$k+1$
C_2
$k+1$
C_m
$k+1$
$k+2$
$m-1$

Conclusions

Conclusions

For the majority automata:

Conclusions

For the majority automata:

- For synchronous and sequential US, **OCP** is in P.

Conclusions

For the majority automata:

- For synchronous and sequential US, **OCP** is in P.
  (is P-Complete)

Conclusions

For the majority automata:

- For synchronous and sequential US, **OCP** is in P.
  (is P-Complete)
- For the block sequential updating schemes the problem is NP-Hard.

Conclusions

For the majority automata:

- For synchronous and sequential US, **OCP** is in P.
  (is P-Complete)
- For the block sequential updating schemes the problem is NP-Hard.
  (We conjecture that **OCP** is $PSPACE$-Complete.)

Conclusions

For the majority automata:

- For synchronous and sequential US, **OCP** is in P.
  (is P-Complete)
- For the block sequential updating schemes the problem is NP-Hard.
  (We conjecture that **OCP** is $PSPACE$-Complete.)

An automata have a "**portable**" complexity if the complexity of the One Cell Prediction problem does not depend on the updating scheme.
(ex. majority with "frozen" active nodes).

Conclusions

For the majority automata:

- For synchronous and sequential US, **OCP** is in *P*.
  (is *P*-Complete)
- For the block sequential updating schemes the problem is *NP*-Hard.
  (We conjecture that **OCP** is *PSPACE*-Complete.)

An automata have a "**portable**" complexity if the complexity of the One Cell Prediction problem does not depend on the updating scheme.
(ex. majority with "frozen" active nodes).

Danke!