Turning block-sequential automata networks into smaller parallel networks with isomorphic limit dynamics

Pacôme Perrotin, Sylvain Sené

LIS

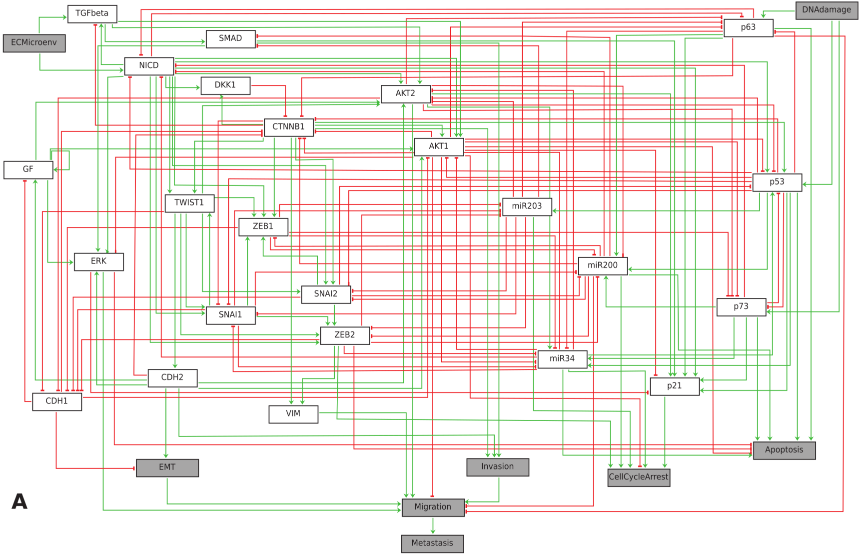
July 27, 2023

Biological networks

Context

The modelisation of neurological networks starts in the 40s (McCulloch 1943)
The modelisation of gene regulatory networks starts in the 60s (Kauffman 1969, Thomas 1973)
Both are applications of automata networks.

Context



extracted from D. PA Cohen et al. "Mathematical modelling of molecular pathways enabling tumour cell invasion and migration". In: *PLoS computational biology* 11.11 (2015), e1004571.

Automata networks

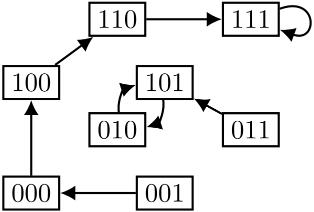
Definitions

Let Σ be an alphabet. An Automata Network (AN) is a function $F : \Sigma^n \to \Sigma^n$, for some $n \in \mathbb{N}$.

Definitions

Let Σ be an alphabet. An Automata Network (AN) is a function $F : \Sigma^n \to \Sigma^n$, for some $n \in \mathbb{N}$.

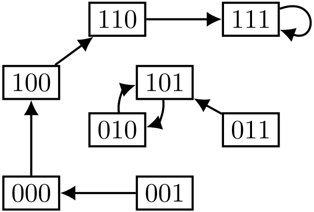
Example : $\Sigma = \{0, 1\}$, $n = 3$.



Definitions

Let Σ be an alphabet. An Automata Network (AN) is a function $F : \Sigma^n \to \Sigma^n$, for some $n \in \mathbb{N}$.

Example : $\Sigma = \{0, 1\}, \ n = 3.$



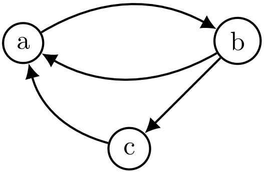
All functions applied in parallel

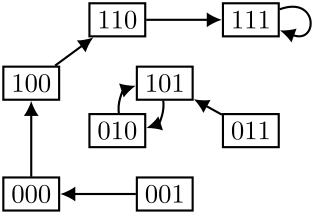
$$f_a(x) = x_b \vee \neg x_c$$
$$f_b(x) = x_a$$
$$f_c(x) = x_b$$

Definitions

Let Σ be an alphabet. An Automata Network (AN) is a function $F : \Sigma^n \to \Sigma^n$, for some $n \in \mathbb{N}$.

Example : $\Sigma = \{0, 1\},\ n = 3$.





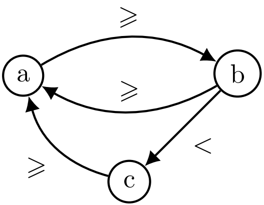
All functions applied in parallel

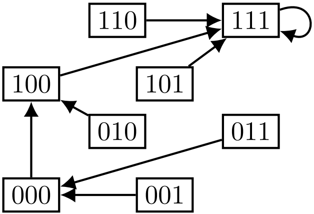
$$f_a(x) = x_b \vee \neg x_c$$
$$f_b(x) = x_a$$
$$f_c(x) = x_b$$

Definitions

Let Σ be an alphabet. An Automata Network (AN) is a function $F : \Sigma^n \to \Sigma^n$, for some $n \in \mathbb{N}$.

Example : $\Sigma = \{0, 1\},\ n = 3.$

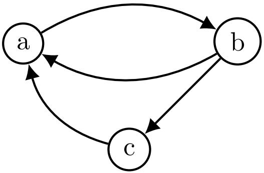


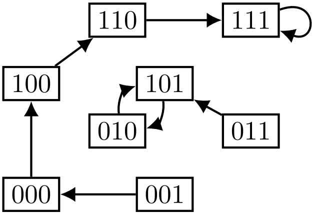


Update schedule : $(\{a, b\}, \{c\})$

$$f_a(x) = x_b \vee \neg x_c$$
$$f_b(x) = x_a$$
$$f_c(x) = x_b$$

Definitions

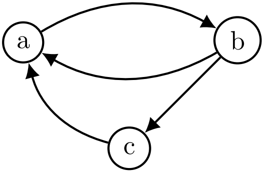




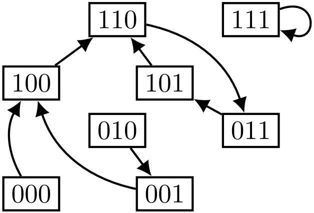
$$f_a(x) = x_b \lor \neg x_c$$
$$f_b(x) = x_a$$
$$f_c(x) = x_b$$

The *limit dynamics* of F is the subgraph of the dynamics that contains only the configurations x such that $F^k(x) = x$ for some $k \in \mathbb{N}^*$.

Examples

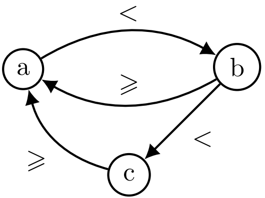


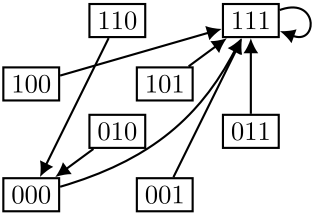
Updated in parallel.



$$f_a(x) = \neg x_b \vee x_c$$
$$f_b(x) = x_a$$
$$f_c(x) = x_b$$

Examples

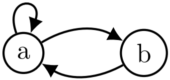




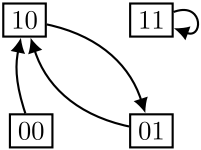
Update schedule ({a}, {b}, {c}).

$$f_a(x) = \neg x_b \lor x_c$$
$$f_b(x) = x_a$$
$$f_c(x) = x_b$$

Examples

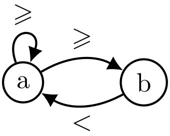


Updated in parallel

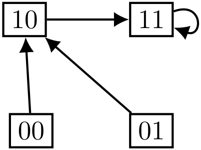


$$f_a(x) = \neg x_a \vee x_b$$
$$f_b(x) = x_a$$

Examples



Update schedule ({b}, {a})

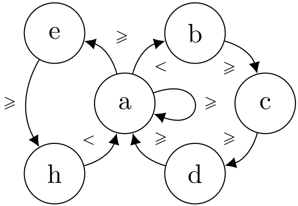


$$f_a(x) = \neg x_a \vee x_b$$
$$f_b(x) = x_a$$

- Automata networks are dynamical systems
- Deciding if a network has a given attractor is NP-complete
- Various update schedules increase the combinatorial complexity of the objects
- We would like to reduce this complexity in some cases

Parallelisation

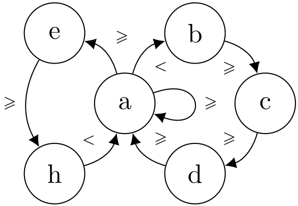
Parallelization algorithm



$(\{h\}, \{a, c, d, e\}, \{b\})$

$$f_a(x) = x_a \vee x_d \vee \neg x_h$$
$$f_b(x) = \neg x_a$$
$$f_c(x) = x_b$$
$$f_d(x) = x_c$$
$$f_e(x) = x_a$$
$$f_h(x) = x_e$$

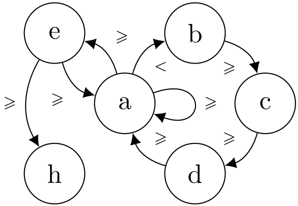
Parallelization algorithm



$(\{h\}, \{a, c, d, e\}, \{b\})$

$$f_a(x) = x_a \vee x_d \vee \neg\theta_h$$
$$f_b(x) = \neg\theta_a$$
$$f_c(x) = x_b$$
$$f_d(x) = x_c$$
$$f_e(x) = x_a$$
$$f_h(x) = x_e$$

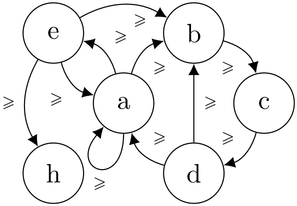
Parallelization algorithm



$(\{a, c, d, e, h\}, \{b\})$

$$f_a(x) = x_a \vee x_d \vee \neg x_e$$
$$f_b(x) = \neg\theta_a$$
$$f_c(x) = x_b$$
$$f_d(x) = x_c$$
$$f_e(x) = x_a$$
$$f_h(x) = x_e$$

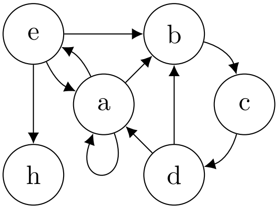
Parallelization algorithm



$(\{a, b, c, d, e, h\})$

$$f_a(x) = x_a \vee x_d \vee \neg x_e$$
$$f_b(x) = \neg(x_a \vee x_d \vee \neg x_e)$$
$$f_c(x) = x_b$$
$$f_d(x) = x_c$$
$$f_e(x) = x_a$$
$$f_h(x) = x_e$$

Parallelization algorithm

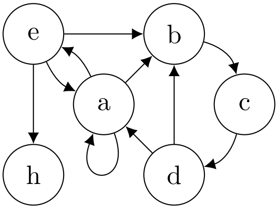


$(\{a, b, c, d, e, h\})$

$$f'_a(x) = x_a \vee x_d \vee \neg x_e$$
$$f'_b(x) = \neg(x_a \vee x_d \vee \neg x_e)$$
$$f'_c(x) = x_b$$
$$f'_d(x) = x_c$$
$$f'_e(x) = x_a$$
$$f'_h(x) = x_e$$

- *Our contribution* is an algorithm that simplifies the result of the parallelization.
- It operates following simple rules:
  - if two automata have the same local function (up to an operation) then we remove one of them,
  - if an automaton has no influence over the network we remove it.
- In the worst case, running this algorithm requires solving a polynomial amount of CoNP-complete problems.

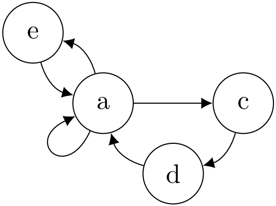
Size reduction algorithm



$(\{a, b, c, d, e, h\})$

$$f'_a(x) = x_a \vee x_d \vee \neg x_e$$
$$f'_b(x) = \neg(x_a \vee x_d \vee \neg x_e)$$
$$f'_c(x) = x_b$$
$$f'_d(x) = x_c$$
$$f'_e(x) = x_a$$
$$f'_h(x) = x_e$$

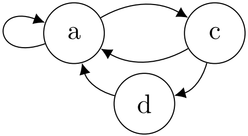
Size reduction algorithm



$(\{a, c, d, e\})$

$$f'_a(x) = x_a \lor x_d \lor \neg x_e$$
$$f'_c(x) = \neg x_a$$
$$f'_d(x) = x_c$$
$$f'_e(x) = x_a$$

Size reduction algorithm



$(\{a, c, d\})$

$$f'_a(x) = x_a \vee x_d \vee \neg x_c$$
$$f'_c(x) = x_a$$
$$f'_d(x) = \neg x_c$$

Tangential cycles

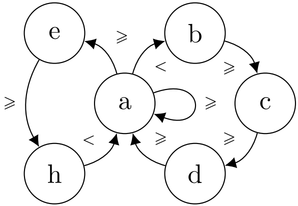
Tangencial cycles: definitions

Definition

Tangencial cycles are ANs composed of cycles which intersect on a segment called the tangent.

Despite their simple definition, the behavior of tangencial cycles is mostly not understood.

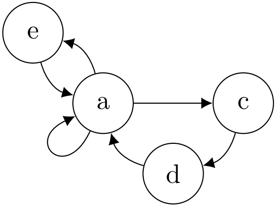
Tangencial cycles: examples



$(\{h\}, \{a, c, d, e\}, \{b\})$

$$f_a(x) = x_a \lor x_d \lor \neg x_h$$
$$f_b(x) = \neg x_a$$
$$f_c(x) = x_b$$
$$f_d(x) = x_c$$
$$f_e(x) = x_a$$
$$f_h(x) = x_e$$

Tangencial cycles: examples



$(\{a, c, d, e\})$

$$f'_a(x) = x_a \lor x_d \lor \neg x_e$$
$$f'_c(x) = \neg x_a$$
$$f'_d(x) = x_c$$
$$f'_e(x) = x_a$$

Tangencial cycles: results

- Bloc sequential tangencial cycles taken through our algorithm always result in smaller parallel tangencial cycles.
- This implies that a complete characterisation of the parallel case is also a complete characterisation of the bloc-sequential cases.

For example, the attractors of double disjunctive cycles are characterised in parallel (M. Noual. "Updating Automata Networks". PhD thesis. École Normale Supérieure de Lyon, 2012). Our contribution extends this result to the bloc-sequential case.

Conclusion

In conclusion,

- ANs are dynamical systems, we want to understand their limit behaviour
- We are looking for cases where we can count attractors in polynomial time
- Counting has nontrivial solutions even in the simplest families we know

In the future,

- We want to find more families that can be counted in polynomial time
- Candidates include intersections of more cycles, and chains of cycles

In conclusion,

- ANs are dynamical systems, we want to understand their limit behaviour
- We are looking for cases where we can count attractors in polynomial time
- Counting has nontrivial solutions even in the simplest families we know

In the future,

- We want to find more families that can be counted in polynomial time
- Candidates include intersections of more cycles, and chains of cycles

Thank you!