Faulty cellular automata: reliable computations in finite and infinite time.

Andrei Romashchenko
LIRMM, Université Montpellier II

11 Januay 2013

Deterministic CA:

- grid of cells $\mathbb{Z}^d$ (infinite space) of $\mathbb{Z}_n^d$ (finite space)

Deterministic CA:

- grid of cells $\mathbb{Z}^d$ (infinite space) of $\mathbb{Z}_n^d$ (finite space)
- a finite automaton at each cell

Deterministic CA:

- grid of cells $\mathbb{Z}^d$ (infinite space) of $\mathbb{Z}_n^d$ (finite space)
- a finite automaton at each cell
- local transformation rule (next state of a cell depends only on its a finite neighborhood)

Deterministic CA:

- grid of cells $\mathbb{Z}^d$ (infinite space) of $\mathbb{Z}_n^d$ (finite space)
- a finite automaton at each cell
- local transformation rule (next state of a cell depends only on its a finite neighborhood)

Synchronous Probabilistic CA:

- the local transformation rule is *probabilistic*

Deterministic CA:

- grid of cells $\mathbb{Z}^d$ (infinite space) of $\mathbb{Z}_n^d$ (finite space)
- a finite automaton at each cell
- local transformation rule (next state of a cell depends only on its a finite neighborhood)

Synchronous Probabilistic CA:

- the local transformation rule is *probabilistic*
- we assume sources of randomness for different cells are independent

Deterministic CA:

- grid of cells $\mathbb{Z}^d$ (infinite space) of $\mathbb{Z}_n^d$ (finite space)
- a finite automaton at each cell
- local transformation rule (next state of a cell depends only on its a finite neighborhood)

Synchronous Probabilistic CA:

- the local transformation rule is *probabilistic*
- we assume sources of randomness for different cells are independent

Faulty CA:
Let A be a deterministic CA. A PCA $\mathcal{A}$ is called an ϵ-perturbation of A if

$$\mathcal{A}(\cdot) = \begin{cases} A(\cdot), & \text{with prob } \geq 1-\epsilon, \\ \text{whatever}, & \text{with prob } \leq \epsilon. \end{cases}$$

Why faulty CA? What is it all about?

Why faulty CA? What is it all about?

Motivations: a model for random noise, perturbations, errors, etc.

Why faulty CA? What is it all about?

Motivations: a model for random noise, perturbations, errors, etc.

The crucial point: We do *not* want to specify a faulty CA *precisely*.

Why faulty CA? What is it all about?

Motivations: a model for random noise, perturbations, errors, etc.

The crucial point: We do *not* want to specify a faulty CA *precisely*.
We specify a *deterministic* rule + the *threshold* ϵ for the random noise.

Why faulty CA? What is it all about?

Motivations: a model for random noise, perturbations, errors, etc.

The crucial point: We do *not* want to specify a faulty CA *precisely*.
We specify a *deterministic* rule + the *threshold* ϵ for the random noise.

cf. Nazim Fatès (yesterday afternoon):

- his *fault* is a failure of an update
- our *fault* is an unpredictable update.

Why faulty CA? What is it all about?

Motivations: a model for random noise, perturbations, errors, etc.

The crucial point: We do *not* want to specify a faulty CA *precisely*.
We specify a *deterministic* rule + the *threshold* ϵ for the random noise.

cf. Nazim Fatès (yesterday afternoon):

- his *fault* is a failure of an update
- our *fault* is an unpredictable update.

Typical questions:

- Can we construct a faulty CA with a stable behavior?

Why faulty CA? What is it all about?

Motivations: a model for random noise, perturbations, errors, etc.

The crucial point: We do *not* want to specify a faulty CA *precisely*.
We specify a *deterministic* rule + the *threshold* ϵ for the random noise.

cf. Nazim Fatès (yesterday afternoon):

- his *fault* is a failure of an update
- our *fault* is an unpredictable update.

Typical questions:

- Can we construct a faulty CA with a stable behavior?
- Can we organize reliable computations?

Why faulty CA? What is it all about?

Motivations: a model for random noise, perturbations, errors, etc.

The crucial point: We do *not* want to specify a faulty CA *precisely*.
We specify a *deterministic* rule + the *threshold* ϵ for the random noise.

cf. Nazim Fatès (yesterday afternoon):

- his *fault* is a failure of an update
- our *fault* is an unpredictable update.

Typical questions:

- Can we construct a faulty CA with a stable behavior?
- Can we organize reliable computations?

The aim of this talk: survey the classic techniques and remind some open problems.

Why faulty CA? What is it all about?

Motivations: a model for random noise, perturbations, errors, etc.

The crucial point: We do *not* want to specify a faulty CA *precisely*.
We specify a *deterministic* rule + the *threshold* ϵ for the random noise.

cf. Nazim Fatès (yesterday afternoon):

- his *fault* is a failure of an update
- our *fault* is an unpredictable update.

Typical questions:

- Can we construct a faulty CA with a stable behavior?
- Can we organize reliable computations?

The aim of this talk: survey the classic techniques and remind some open problems.

- Do you have problems that can be solved with these techniques?
- Do you have new techniques to solve these open problems?

Question 1: How to keep some information in a faulty medium?

Question 1: How to keep some information in a faulty medium?

Infinite version of question 1: How to keep some information in a faulty medium on $\mathbb{Z}^d$ *forever* ?

Question 1: How to keep some information in a faulty medium?

Infinite version of question 1: How to keep some information in a faulty medium on $\mathbb{Z}^d$ *forever* ?
Finite version of question 1: How to keep some information in a faulty medium on $\mathbb{Z}_n^d$ *for exponentially long time* ?

Question 1: How to keep some information in a faulty medium?

Infinite version of question 1: How to keep some information in a faulty medium on $\mathbb{Z}^d$ *forever* ?
Finite version of question 1: How to keep some information in a faulty medium on $\mathbb{Z}_n^d$ *for exponentially long time* ?

What does it mean that a faulty automaton $\mathcal{A}_\epsilon$ keeps some information? At least one bit of information?

Question 1: How to keep some information in a faulty medium?

Infinite version of question 1: How to keep some information in a faulty medium on $\mathbb{Z}^d$ *forever* ?
Finite version of question 1: How to keep some information in a faulty medium on $\mathbb{Z}_n^d$ *for exponentially long time* ?

What does it mean that a faulty automaton $\mathcal{A}_\epsilon$ keeps some information? At least one bit of information?

A weak property: There exist two initial configuration $\mathcal{I}_0$ and $\mathcal{I}_1$ and $\delta > 0$ such that for all steps t (a finite version: for $t < exp\{n\}$)

$$\mathrm{dist}(\mathcal{A}_\epsilon^t(\mathcal{I}_0), \mathcal{A}_\epsilon^t(\mathcal{I}_1)) > \delta$$

Question 1: How to keep some information in a faulty medium?

Infinite version of question 1: How to keep some information in a faulty medium on $\mathbb{Z}^d$ *forever* ?
Finite version of question 1: How to keep some information in a faulty medium on $\mathbb{Z}_n^d$ *for exponentially long time* ?

What does it mean that a faulty automaton $\mathcal{A}_\epsilon$ keeps some information? At least one bit of information?

A weak property: There exist two initial configuration $\mathcal{I}_0$ and $\mathcal{I}_1$ and $\delta > 0$ such that for all steps t (a finite version: for $t < exp\{n\}$)

$$\mathrm{dist}(\mathcal{A}_\epsilon^t(\mathcal{I}_0), \mathcal{A}_\epsilon^t(\mathcal{I}_1)) > \delta$$

A somewhat stronger property: There exist two different $\mathcal{A}_\epsilon$-invariant measures μ_0 and μ_1.

Toom theorem: There exists a deterministic CA on $\{0,1\}^{\mathbb{Z}^2}$ such that small enough ϵ, for all ϵ-perturbations $\mathcal{A}$ of this automaton *a natural property* holds: if we start with the initial configuration $\mathcal{I}_0$ (all zeros) or $\mathcal{I}_1$ (all ones), then for each step t, for each cell (i,j)

$$\mathrm{Prob}[(\mathcal{A}^t(\mathcal{I}_0))_{ij} = 0] > 1 - O(\varepsilon),$$

$$\mathrm{Prob}[(\mathcal{A}^t(\mathcal{I}_1))_{ij} = 1] > 1 - O(\varepsilon).$$

Toom theorem: There exists a deterministic CA on $\{0,1\}^{\mathbb{Z}^2}$ such that small enough ϵ, for all ϵ-perturbations $\mathcal{A}$ of this automaton *a natural property* holds: if we start with the initial configuration $\mathcal{I}_0$ (all zeros) or $\mathcal{I}_1$ (all ones), then for each step t, for each cell (i,j)

$$\mathrm{Prob}[(\mathcal{A}^t(\mathcal{I}_0))_{ij} = 0] > 1 - O(\varepsilon),$$

$$\mathrm{Prob}[(\mathcal{A}^t(\mathcal{I}_1))_{ij} = 1] > 1 - O(\varepsilon).$$

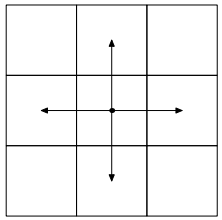
A weaker corollary: There exist two initial configuration $\mathcal{I}_0$ (all zeros) and $\mathcal{I}_1$ (all ones) and $\delta > 0$ such that for all steps t

$$\mathrm{dist}(\mathcal{A}_\epsilon^t(\mathcal{I}_0), \mathcal{A}_\epsilon^t(\mathcal{I}_1)) > \delta$$

A somewhat stronger corollary: There exist two different $\mathcal{A}_\epsilon$-invariant measures μ_0 (most cells are 0) and μ_1 (most cells are 1).

Isn't **the Toom theorem** trivial? Let's take the majority vote

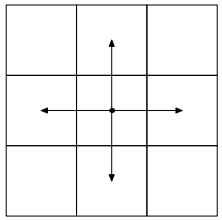
$$A(i,j,t+1) = \mathrm{maj}\{A(i,j,t), A(i+1,j,t), A(i-1,j,t), A(i,j+1,t), A(i,j-1,t)\}$$



Should it work?

Isn't **the Toom theorem** trivial? Let's take the majority vote

$$A(i,j,t+1) = \mathrm{maj}\{A(i,j,t), A(i+1,j,t), A(i-1,j,t), A(i,j+1,t), A(i,j-1,t)\}$$

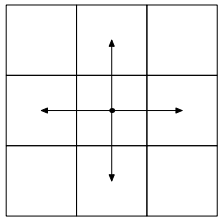


Should it work?

Well, *may be* it works...

Isn't **the Toom theorem** trivial? Let's take the majority vote

$$A(i,j,t+1) = \mathrm{maj}\{A(i,j,t), A(i+1,j,t), A(i-1,j,t), A(i,j+1,t), A(i,j-1,t)\}$$

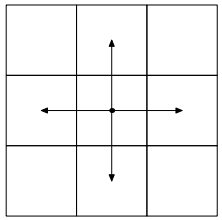


Should it work?

Well, *may be* it works... But this is still an open problem!

Isn't **the Toom theorem** trivial? Let's take the majority vote

$$A(i,j,t+1) = \mathrm{maj}\{A(i,j,t), A(i+1,j,t), A(i-1,j,t), A(i,j+1,t), A(i,j-1,t)\}$$



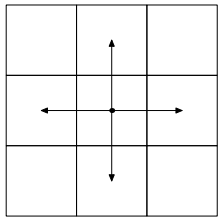
Should it work?

Well, *may be* it works... But this is still an open problem!

What is wrong with the symmetric *majority* rule?

Isn't **the Toom theorem** trivial? Let's take the majority vote

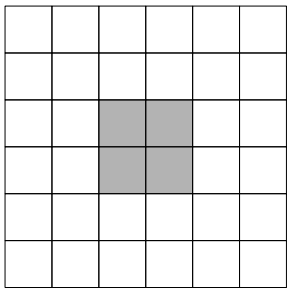
$$A(i,j,t+1) = \mathrm{maj}\{A(i,j,t), A(i+1,j,t), A(i-1,j,t), A(i,j+1,t), A(i,j-1,t)\}$$



Should it work?

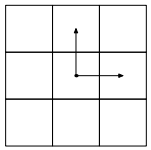
Well, *may be* it works... But this is still an open problem!

What is wrong with the symmetric *majority* rule? It does not erode an island of 1-s in the ocean of 0-s.



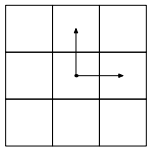
The idea of Andrei Toom: take **NEC** majority:

$$\mathrm{NEC}(i,j,t+1) = \mathrm{majority}\{A(i,j,t), A(i+1,j,t), A(i,j+1,t)\}$$



The idea of Andrei Toom: take **NEC** majority:

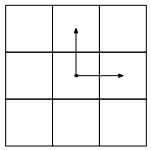
$$\mathrm{NEC}(i,j,t+1) = \mathrm{majority}\{A(i,j,t), A(i+1,j,t), A(i,j+1,t)\}$$



What so good about **NEC** majority?

The idea of Andrei Toom: take **NEC** majority:

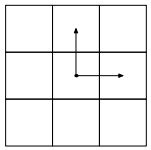
$$\mathrm{NEC}(i,j,t+1) = \mathrm{majority}\{A(i,j,t), A(i+1,j,t), A(i,j+1,t)\}$$



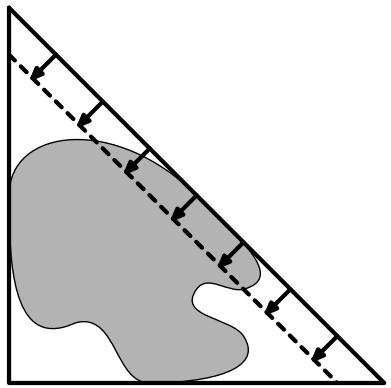
What so good about **NEC** majority?
Every island of 1-s in the ocean of 0-s shrinks.

The idea of Andrei Toom: take **NEC** majority:

$$\mathrm{NEC}(i,j,t+1) = \mathrm{majority}\{A(i,j,t), A(i+1,j,t), A(i,j+1,t)\}$$



What so good about **NEC** majority?
Every island of 1-s in the ocean of 0-s shrinks. Moreover, it shrinks in *fast*. (Remember the slides of Lise Ponselet!)



Sketch of the proof of Toom's result:

Sketch of the proof of Toom's result:

Take the initial configuration of all 0.

Sketch of the proof of Toom's result:

Take the initial configuration of all 0. Apply the **NEC** perturbed by ε-bounded random noise...

Sketch of the proof of Toom's result:

Take the initial configuration of all 0. Apply the **NEC** perturbed by ε-bounded random noise...

Step 1 (probabilistic): With probability 1 all errors on the space-time diagram can be split into isolated and sparse *islands* of different ranks:

Sketch of the proof of Toom's result:

Take the initial configuration of all 0. Apply the **NEC** perturbed by ε-bounded random noise...

Step 1 (probabilistic): With probability 1 all errors on the space-time diagram can be split into isolated and sparse *islands* of different ranks:

- small islands with a small clean "security zones" around,
- larger islands with a larger "security zones",
- huge islands with a huge "security zones",
- ... etc.

Sketch of the proof of Toom's result:

Take the initial configuration of all 0. Apply the **NEC** perturbed by ε-bounded random noise...

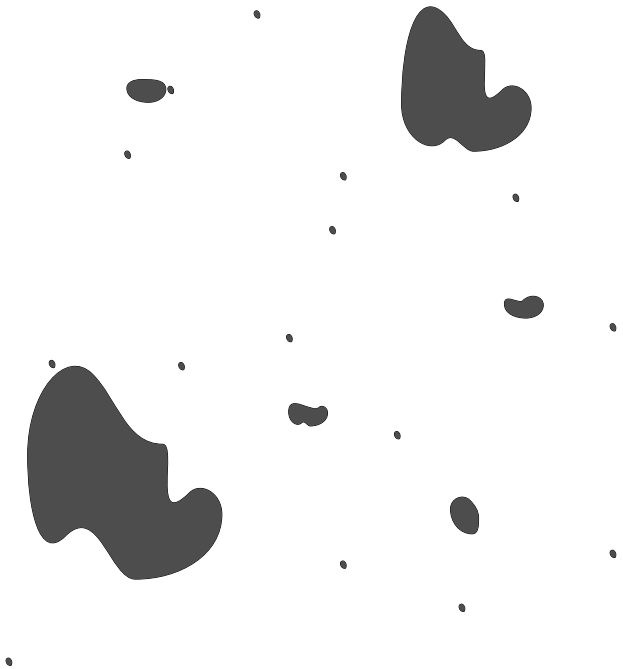
Step 1 (probabilistic): With probability 1 all errors on the space-time diagram can be split into isolated and sparse *islands* of different ranks:

- small islands with a small clean "security zones" around,
- larger islands with a larger "security zones",
- huge islands with a huge "security zones",
- ... etc.

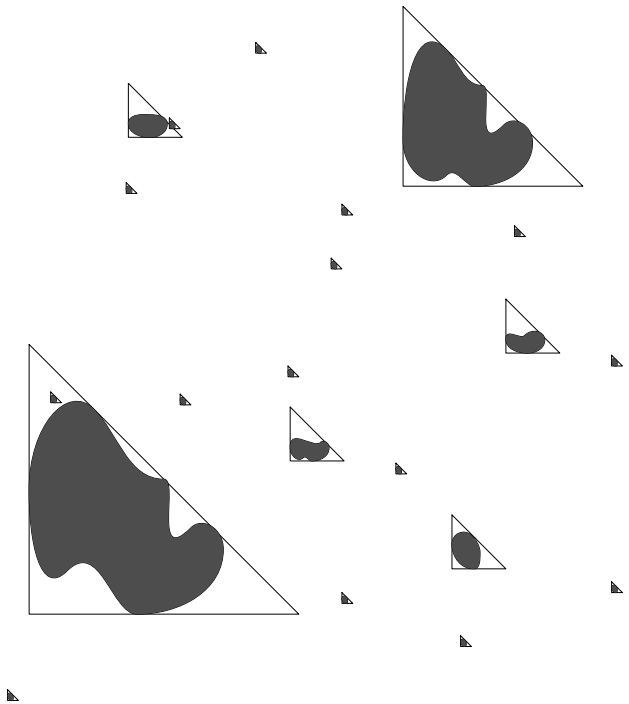
Step 2 (combinatorial):

- small islands shrink without interacting with other islands,
- larger islands shrink without interacting with other islands,
- huge islands shrink without interacting with other islands,
- ... etc.

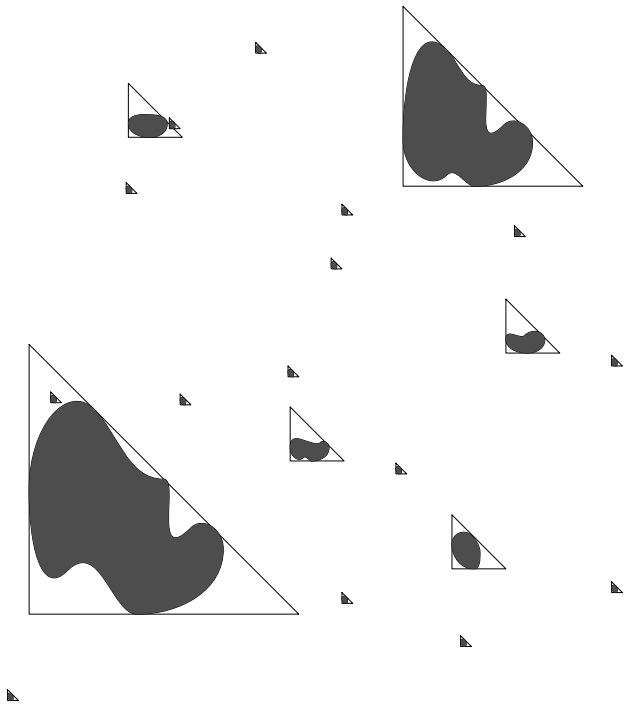
Metaphorical picture of **Step 1**: split faults into isolated islands



Metaphorical picture of **Step 2**: the Toom rule shrinks islands



Metaphorical picture of **Step 2**: the Toom rule shrinks islands



Conclusion: Only small neighborhoods of islands are affected by the random faults. Information about the initial state is kept forever.

Extensions of this technique:

- Finite case (torus $\mathbb{Z}_n^2$): The initial information is kept until a too large islands of errors comes.
- More accurate bounds for the finite case.
- A large class of other *eroders* that work similar to the NEC majority.

Extensions of this technique:

- Finite case (torus $\mathbb{Z}_n^2$): The initial information is kept until a too large islands of errors comes.
- More accurate bounds for the finite case.
- A large class of other *eroders* that work similar to the NEC majority.

What is still open:

- Is an ϵ-perturbation of the *symmetric* majority ergodic?

Extensions of this technique:

- Finite case (torus $\mathbb{Z}_n^2$): The initial information is kept until a too large islands of errors comes.
- More accurate bounds for the finite case.
- A large class of other *eroders* that work similar to the NEC majority.

What is still open:

- Is an ϵ-perturbation of the *symmetric* majority ergodic?
- Other invariant measures for an ϵ-perturbation of the NEC majority?

Question 2: How to implement stable computations in a faulty medium?

Question 2: How to implement stable computations in a faulty medium?

Toom's technique is enough to organize stable (Turing universal) computation in $\mathbb{Z}^3$.

Question 2: How to implement stable computations in a faulty medium?

Toom's technique is enough to organize stable (Turing universal) computation in $\mathbb{Z}^3$.

Construction suggested by Gács and Reif :

- We simulate a one-tape Turing machine

Question 2: How to implement stable computations in a faulty medium?

Toom's technique is enough to organize stable (Turing universal) computation in $\mathbb{Z}^3$.

Construction suggested by Gács and Reif :

- We simulate a one-tape Turing machine or a 1D deterministic CA.

Question 2: How to implement stable computations in a faulty medium?

Toom's technique is enough to organize stable (Turing universal) computation in $\mathbb{Z}^3$.

Construction suggested by Gács and Reif :

- We simulate a one-tape Turing machine or a 1D deterministic CA.
- each vertical column of our 3D CA simulates a tape of the TM

Question 2: How to implement stable computations in a faulty medium?

Toom's technique is enough to organize stable (Turing universal) computation in $\mathbb{Z}^3$.

Construction suggested by Gács and Reif :

- We simulate a one-tape Turing machine or a 1D deterministic CA.
- each vertical column of our 3D CA simulates a tape of the TM
- in each horizontal plane we do self-correcting á la Toom.

Question 2: How to implement stable computations in a faulty medium?

Toom's technique is enough to organize stable (Turing universal) computation in $\mathbb{Z}^3$.

Construction suggested by Gács and Reif :

- We simulate a one-tape Turing machine or a 1D deterministic CA.
- each vertical column of our 3D CA simulates a tape of the TM
- in each horizontal plane we do self-correcting á la Toom.

Proof of correctness (sketch):

- split all faults in the 4D space-time diagrams into isolated islands
- prove that consequences of each island are bounded in space-time

Question 2: How to implement stable computations in a faulty medium?

Toom's technique is enough to organize stable (Turing universal) computation in $\mathbb{Z}^3$.

Construction suggested by Gács and Reif :

- We simulate a one-tape Turing machine or a 1D deterministic CA.
- each vertical column of our 3D CA simulates a tape of the TM
- in each horizontal plane we do self-correcting á la Toom.

Proof of correctness (sketch):

- split all faults in the 4D space-time diagrams into isolated islands
- prove that consequences of each island are bounded in space-time

Remark 1: The fine version of the construction also works fine.
Remark 2: Essentially we *repeat* the argument from the Toom theorem. We cannot (?) use Toom's theorem as a black box.

What is good about Gács-Reif construction?

What is good about Gács-Reif construction?

- easy to explain the construction
- rather easy to prove the correctness.

What is good about Gács-Reif construction?

- easy to explain the construction
- rather easy to prove the correctness.

What is bad about this construction?

What is good about Gács-Reif construction?

- easy to explain the construction
- rather easy to prove the correctness.

What is bad about this construction?

- we simulate 1D computation in a 3D medium.

Question 3: Do there exists an intrinsically universal faulty CA ?

Question 3: Do there exists an intrinsically universal faulty CA ?

Gács construction: a 2D faulty CA performs a robust simulation for a 2D deterministic CA

Question 3: Do there exists an intrinsically universal faulty CA ?

Gács construction: a 2D faulty CA performs a robust simulation for a 2D deterministic CA

Main idea: Robust self-simulation.

Question 3: Do there exists an intrinsically universal faulty CA ?

Gács construction: a 2D faulty CA performs a robust simulation for a 2D deterministic CA

Main idea: Robust self-simulation. A $Q \times Q$ block of automata $\mathcal{A}$ simulates one cell of $\mathcal{A}$. (cf. von Neumann's self-reproducing automata).

Question 3: Do there exists an intrinsically universal faulty CA ?

Gács construction: a 2D faulty CA performs a robust simulation for a 2D deterministic CA

Main idea: Robust self-simulation. A $Q \times Q$ block of automata $\mathcal{A}$ simulates one cell of $\mathcal{A}$. (cf. von Neumann's self-reproducing automata).

The feature: simulation *reduces* the probability of an error.

Question 3: Do there exists an intrinsically universal faulty CA ?

Gács construction: a 2D faulty CA performs a robust simulation for a 2D deterministic CA

Main idea: Robust self-simulation. A $Q \times Q$ block of automata $\mathcal{A}$ simulates one cell of $\mathcal{A}$. (cf. von Neumann's self-reproducing automata).

The feature: simulation *reduces* the probability of an error.

Rough scheme: Apply robust self-simulation several times.

Question 3: Do there exists an intrinsically universal faulty CA ?

Gács construction: a 2D faulty CA performs a robust simulation for a 2D deterministic CA

Main idea: Robust self-simulation. A $Q \times Q$ block of automata $\mathcal{A}$ simulates one cell of $\mathcal{A}$. (cf. von Neumann's self-reproducing automata).

The feature: simulation *reduces* the probability of an error.

Rough scheme: Apply robust self-simulation several times. On the top level faults become improbable, and we can simulate a deterministic CA.

Question 3: Do there exists an intrinsically universal faulty CA ?

Gács construction: a 2D faulty CA performs a robust simulation for a 2D deterministic CA

Main idea: Robust self-simulation. A $Q \times Q$ block of automata $\mathcal{A}$ simulates one cell of $\mathcal{A}$. (cf. von Neumann's self-reproducing automata).

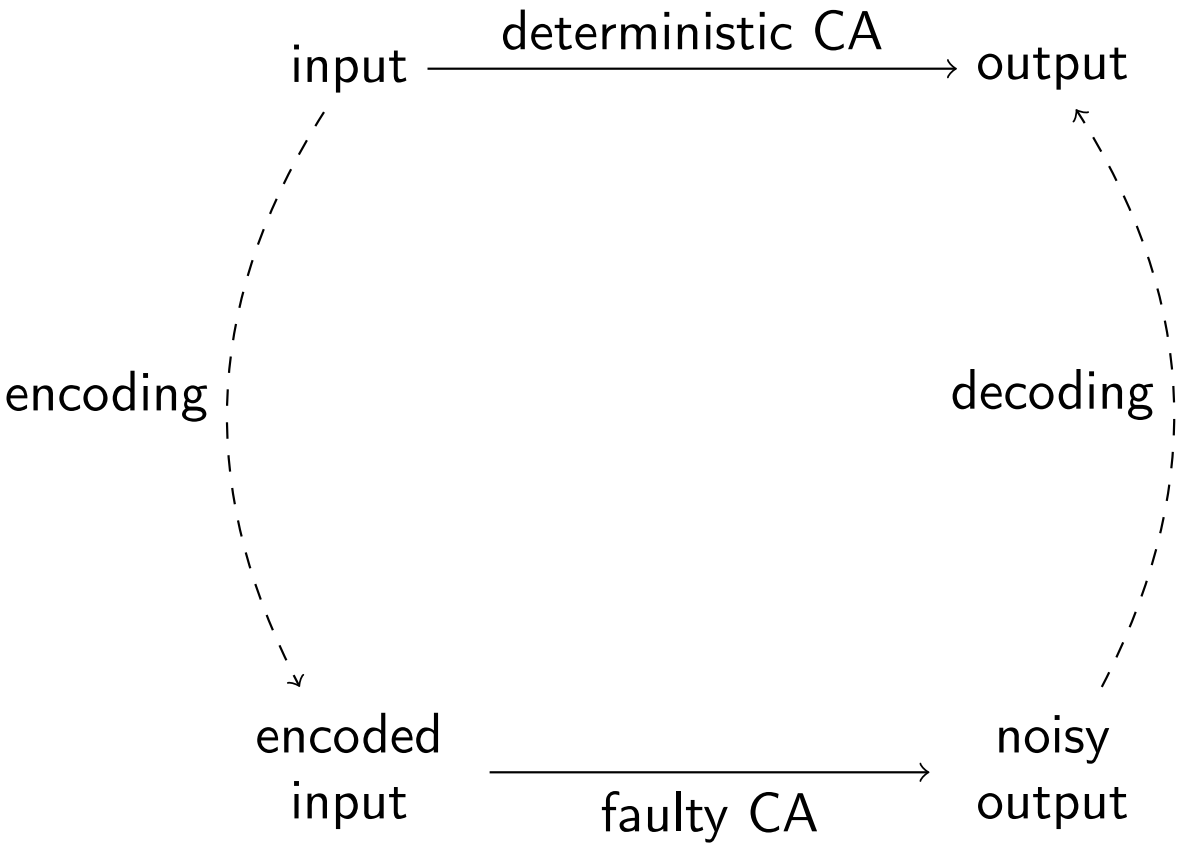
The feature: simulation *reduces* the probability of an error.

Rough scheme: Apply robust self-simulation several times. On the top level faults become improbable, and we can simulate a deterministic CA.

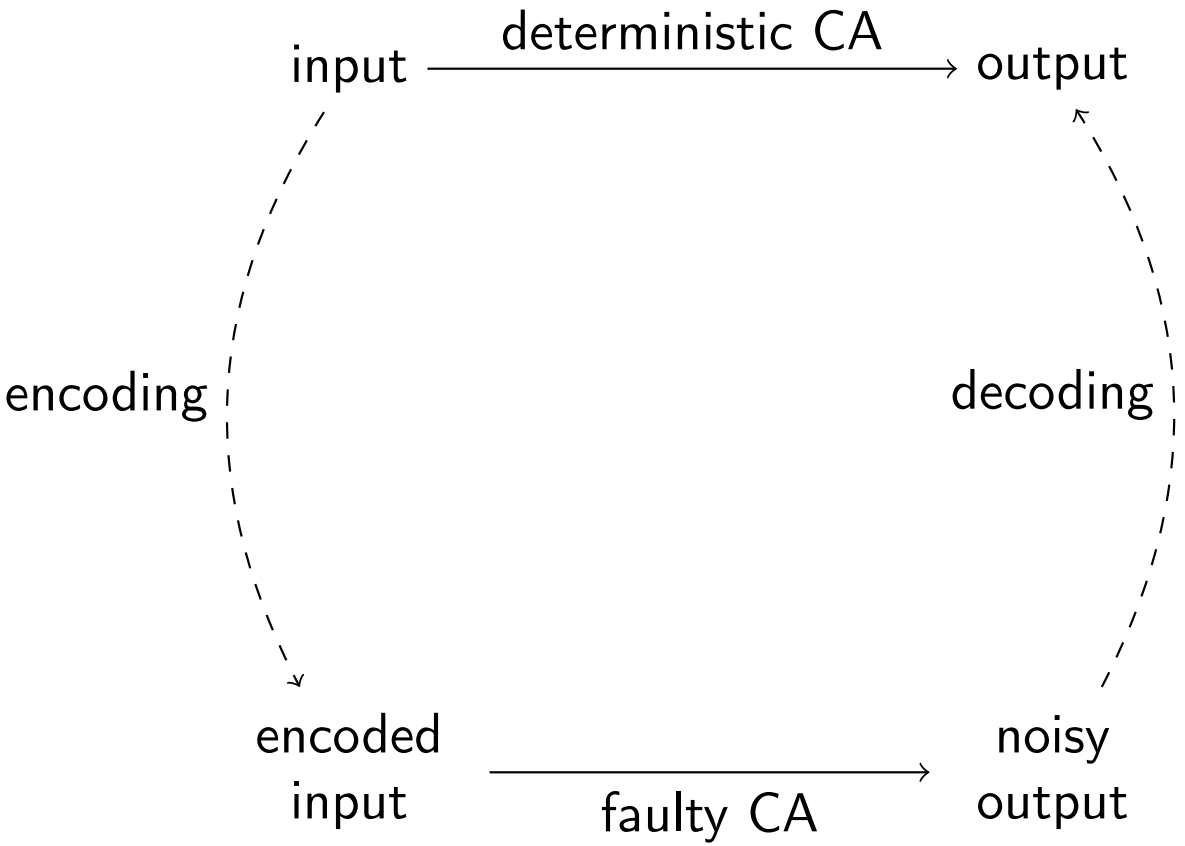
Under the hood of Gács' construction:

- Each cell "knows" its coordinates in the block and "counts" its steps in the loop. Self-correction *á la* Toom maintains the correct information about coordinates and the counter.
- Self-referential constriction *á la* Kleene provides self-simulation.
- Simple duplication helps to suppress small islands of errors.

Pitfalls of "robust simulation"

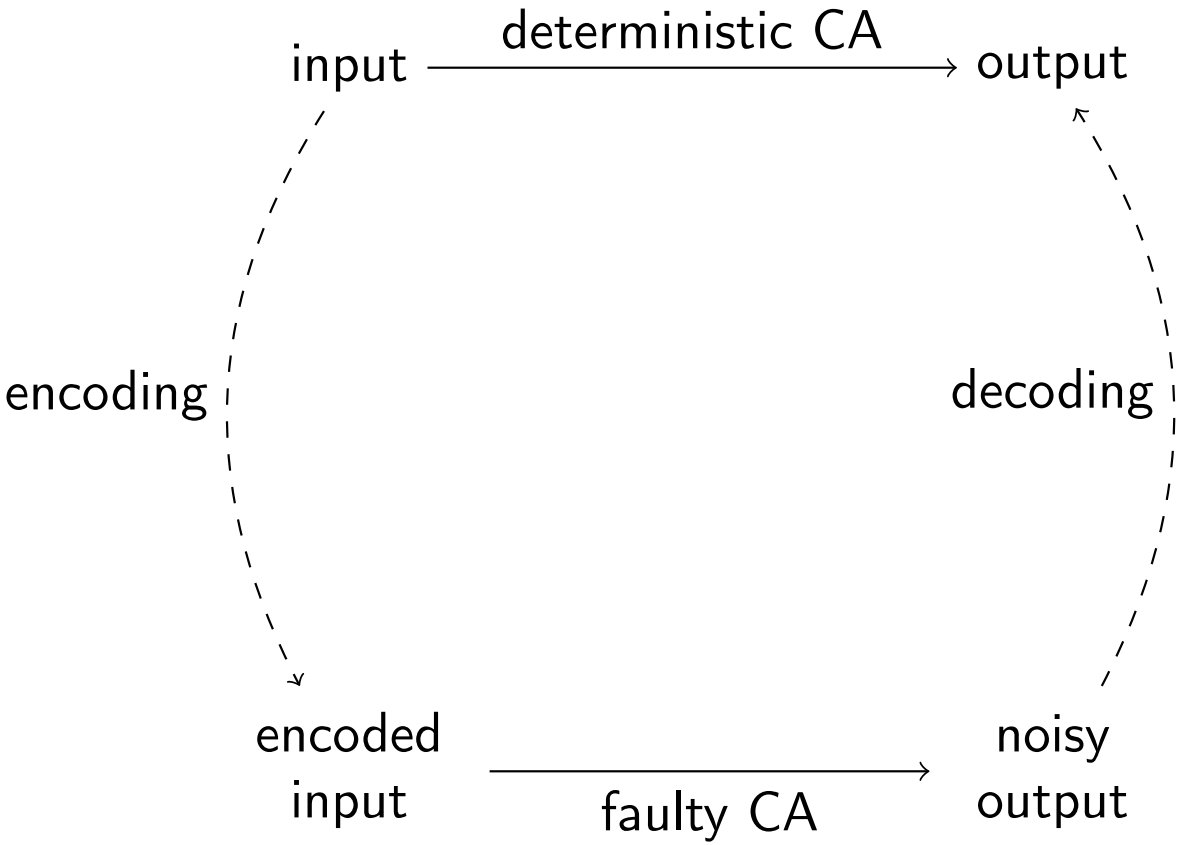


Pitfalls of "robust simulation"



Encoding/decoding should not do all the job!

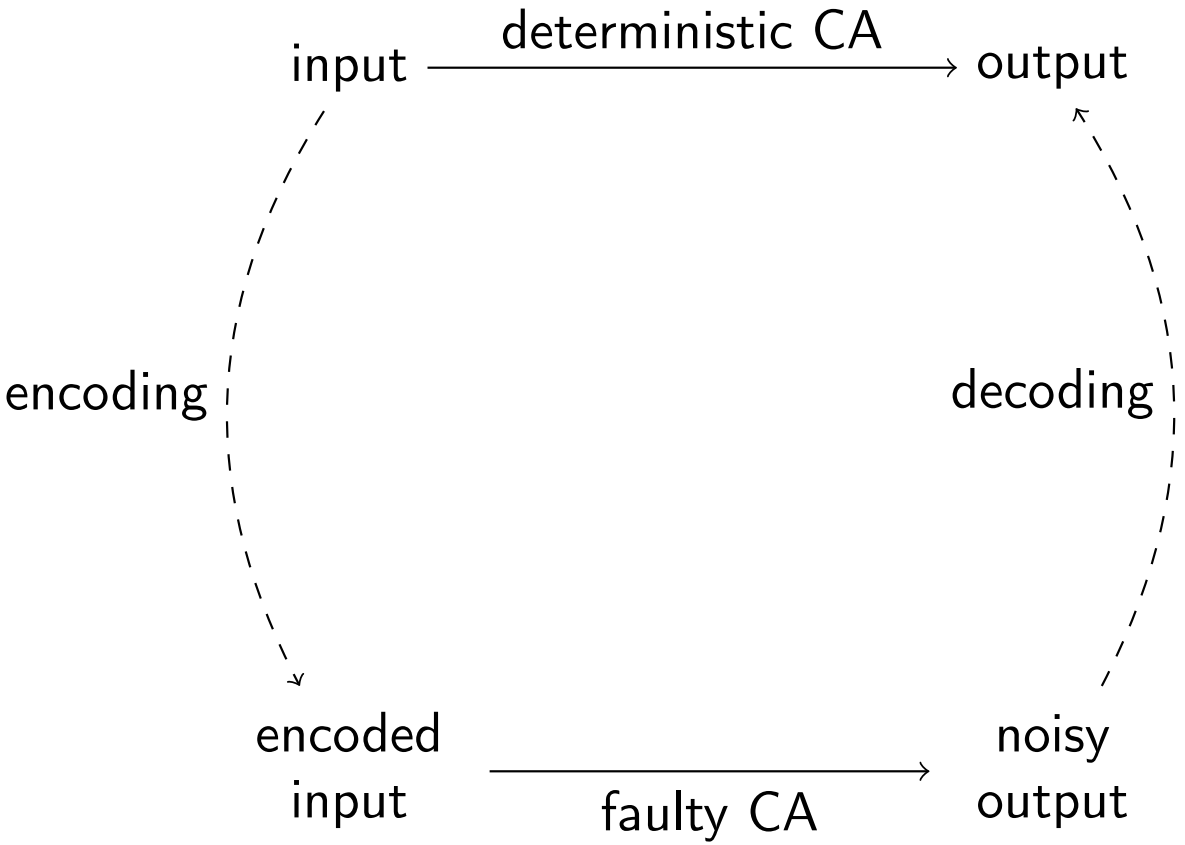
Pitfalls of "robust simulation"



Encoding/decoding should not do all the job!

Open problem: What is the *right* definition of the robust simulation?

Pitfalls of "robust simulation"



Encoding/decoding should not do all the job!

Open problem: What is the *right* definition of the robust simulation?

Open problem: How to define robust simulation for an infinite medium?

1D version of the Gács construction.

1D version of the Gács construction.

In the infinite case: a counter-example to the *positive rate conjecture* (a faulty 1D automata keeps the initial information forever).

1D version of the Gács construction.

In the infinite case: a counter-example to the *positive rate conjecture* (a faulty 1D automata keeps the initial information forever).

Under the hood of the Gács 1D construction:

- Each cell "knows" its coordinates in the block and "counts" its steps in the loop. ~~Self-correction *á la* Toom~~
- Self-referential constriction *á la* Kleene provides self-simulation.
- Simple duplication helps to suppress small islands of errors.

1D version of the Gács construction.

In the infinite case: a counter-example to the *positive rate conjecture* (a faulty 1D automata keeps the initial information forever).

Under the hood of the Gács 1D construction:

- Each cell "knows" its coordinates in the block and "counts" its steps in the loop. ~~Self-correction *á la* Toom~~
- Self-referential constriction *á la* Kleene provides self-simulation.
- Simple duplication helps to suppress small islands of errors.

... the same idea as in 2D, but without ~~the cat~~ Toom's rule!

1D version of the Gács construction.

In the infinite case: a counter-example to the *positive rate conjecture* (a faulty 1D automata keeps the initial information forever).
Under the hood of the Gács 1D construction:

- Each cell "knows" its coordinates in the block and "counts" its steps in the loop. ~~Self-correction *á la* Toom~~
- Self-referential constriction *á la* Kleene provides self-simulation.
- Simple duplication helps to suppress small islands of errors.

... the same idea as in 2D, but without ~~the cat~~ Toom's rule!
Hint: An self-coherent island "suicides" when it "realizes" it cannot run self-simulation on some level.

1D version of the Gács construction.

In the infinite case: a counter-example to the *positive rate conjecture* (a faulty 1D automata keeps the initial information forever).
Under the hood of the Gács 1D construction:

- Each cell "knows" its coordinates in the block and "counts" its steps in the loop. ~~Self-correction *á la* Toom~~
- Self-referential constriction *á la* Kleene provides self-simulation.
- Simple duplication helps to suppress small islands of errors.

... the same idea as in 2D, but without ~~the cat~~ Toom's rule!
Hint: An self-coherent island "suicides" when it "realizes" it cannot run self-simulation on some level.
Beyond this talk: an automata *á la* Gács can keep information with positive density ($O(1)$ bits/cell).

A survey of the survey:

A survey of the survey:

1 idea: Hierarchical islands of errors

A survey of the survey:

1 idea: Hierarchical islands of errors

2 idea: Toom's technique of eroders

- **advantages:** very simple automata, rather simple argument
- **disadvantage:** $\dim \geq 2$.

A survey of the survey:

1 idea: Hierarchical islands of errors

2 idea: Toom's technique of eroders

- **advantages:** very simple automata, rather simple argument
- **disadvantage:** $\dim \geq 2$.

3 idea: Gács' technique of robust self-simulation

- **advantages:** works for 1D, very flexible
- **disadvantages:** huge alphabet, huge zoom factor

A survey of the survey:

1 idea: Hierarchical islands of errors

2 idea: Toom's technique of eroders

- **advantages:** very simple automata, rather simple argument
- **disadvantage:** $\dim \geq 2$.

3 idea: Gács' technique of robust self-simulation

- **advantages:** works for 1D, very flexible
- **disadvantages:** huge alphabet, huge zoom factor

Several questions remain:

- A simpler construction of a nonergodic faulty CA in 1D.
- Reliable computations in infinite medium: find the "right" definition.
- "Smooth" universality: can a faulty CA be intrinsically universal for PCA's in any reasonable sense? [*cf.* Arrighi-Schabanel-Theyssier'12]

Merci de votre attention !