école
normale
supérieure
paris—saclay

Space-time deterministic graph rewriting

P. Arrighi, M. Costes, G. Dowek and L. Maignan

CNRS,LMF
Université Paris-Saclay

ÉNS Paris-Saclay

January 30, 2025

Introduction

- **Dynamical systems** usually assume a global clock . . .

arXiv:2404.05838v1 [cs.DM] 8 Apr 2024

Space-time deterministic graph rewriting

Pablo Arrighi[1], Marin Costes[1], Gilles Dowek[1], and Luidnel Maignan[2]

[1] Université Paris-Saclay, Inria, CNRS, LMF, 91190 Gif-sur-Yvette, France
[2] Univ Paris Est Creteil, LACL, 94000, Creteil, France

Abstract. We study non-terminating graph rewriting models, whose local rules are applied non-deterministically—and yet enjoy a strong form of determinism, namely space-time determinism. Of course in the case of terminating computation it is well-known that the mess introduced by asynchronous rule applications may not matter to the end result, as confluence conspires to produce a unique normal form. In the context of non-terminating computation however, confluence is a very weak property, and (almost) synchronous rule applications is always preferred e.g. when it comes to simulating dynamical systems. Here we provide sufficient conditions so that asynchronous local rule applications conspire to produce well-determined events in the space-time unfolding of the graph, regardless of their application orders. Our first example is an asynchronous simulation of a dynamical system. Our second example features time dilation, in the spirit of general relativity.

Keywords: Causal graph dynamics · Cellular automata · Time covariance · Commutation · Strong confluence · Distributed computation · Task dependencies · DAG · Poset · Space-like cut · Foliation

1 Introduction

In short Distributed models of physical, biological, social or technological objects are often composed of interacting elements (particles, cells, agents, processes etc.) that evolve according to local rules. From this local evolution, the global evolution may be defined in various ways. Dynamical systems like cellular automata assume a global clock, each element synchronously undergoing one local rule step at each tick. Rewrite systems on the other hand assume that each element asynchronously performs a local rule step, whilst the other may remain unchanged. Synchronism is often criticised for being costly and physically unrealistic, but asynchronism leads to an inherent lack of determinism and inconsistencies therein. The paper shows that, even for asynchronous graph rewriting, a strong form of determinism, namely space-time determinism, is still possible. For this purpose it introduces a formalism for graph rewriting based on a DAG of dependencies and some local rule. It proves one proposition and one theorem, whereby local conditions on the local rule entail that its asynchronous applications produce well-determined events.
Dynamical systems refer to the global evolution of an entire configuration at time t into another at time $t+1$, $t+2$ etc., iteratively. Whether the dynamical system is a grid based model (e.g. representing particles [1,2], fluids [3,4], traffic jams [5], demographics and

Introduction

- **Dynamical systems** usually assume a global clock . . .
- . . . whereas in **rewriting systems** each element can be updated asynchronously.

arXiv:2404.05838v1 [cs.DM] 8 Apr 2024

Space-time deterministic graph rewriting

Pablo Arrighi[1], Marin Costes[1], Gilles Dowek[1], and Luidnel Maignan[2]

[1] Université Paris-Saclay, Inria, CNRS, LMF, 91190 Gif-sur-Yvette, France
[2] Univ Paris Est Creteil, LACL, 94000, Creteil, France

Abstract. We study non-terminating graph rewriting models, whose local rules are applied non-deterministically—and yet enjoy a strong form of determinism, namely space-time determinism. Of course in the case of terminating computation it is well-known that the mess introduced by asynchronous rule applications may not matter to the end result, as confluence conspires to produce a unique normal form. In the context of non-terminating computation however, confluence is a very weak property, and (almost) synchronous rule applications is always preferred e.g. when it comes to simulating dynamical systems. Here we provide sufficient conditions so that asynchronous local rule applications conspire to produce well-determined events in the space-time unfolding of the graph, regardless of their application orders. Our first example is an asynchronous simulation of a dynamical system. Our second example features time dilation, in the spirit of general relativity.

Keywords: Causal graph dynamics · Cellular automata · Time covariance · Commutation · Strong confluence · Distributed computation · Task dependencies · DAG · Poset · Space-like cut · Foliation

1 Introduction

In short Distributed models of physical, biological, social or technological objects are often composed of interacting elements (particles, cells, agents, processes etc.) that evolve according to local rules. From this local evolution, the global evolution may be defined in various ways. Dynamical systems like cellular automata assume a global clock, each element synchronously undergoing one local rule step at each tick. Rewrite systems on the other hand assume that each element asynchronously performs a local rule step, whilst the other may remain unchanged. Synchronism is often criticised for being costly and physically unrealistic, but asynchronism leads to an inherent lack of determinism and inconsistencies therein. The paper shows that, even for asynchronous graph rewriting, a strong form of determinism, namely space-time determinism, is still possible. For this purpose it introduces a formalism for graph rewriting based on a DAG of dependencies and some local rule. It proves one proposition and one theorem, whereby local conditions on the local rule entail that its asynchronous applications produce well-determined events.

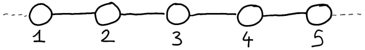
Dynamical systems refer to the global evolution of an entire configuration at time t into another at time $t+1$, $t+2$ etc., iteratively. Whether the dynamical system is a grid based model (e.g. representing particles [1,2], fluids [3,4], traffic jams [5], demographics and

1 - Motivations

Cellular automata

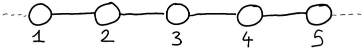
- **Space :** grid of dimension d–i.e. $\mathbb{Z}^d$.

  Example : $d = 1$.

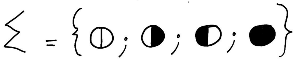


Cellular automata

- **Space :** grid of dimension d–i.e. $\mathbb{Z}^d$.
  Example : $d = 1$.



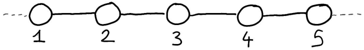
- **Internal states:** finite alphabet Σ.
  Example : $\Sigma = \{0, 1\}^2$.



Cellular automata

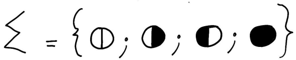
- **Space :** grid of dimension d–i.e. $\mathbb{Z}^d$.

  Example : $d = 1$.



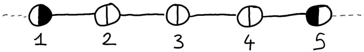
- **Internal states:** finite alphabet Σ.

  Example : $\Sigma = \{0, 1\}^2$.



- **Configuration :** function $c : \mathbb{Z}^d \to \Sigma$.

  Example : $1 \mapsto (0, 1), 7 \mapsto (1, 0)$.

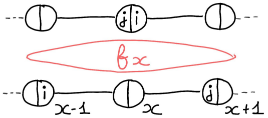


Cellular automata

- **Local rule :** function $f : \Sigma^{\mathcal{N}} \to \Sigma$.

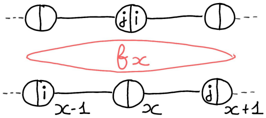
Cellular automata

- **Local rule :** function $f : \Sigma^{\mathcal{N}} \to \Sigma$.



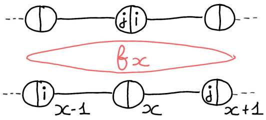
Cellular automata

- **Local rule :** function $f : \Sigma^{\mathcal{N}} \to \Sigma$.
- **Cellular automata :** function $A : \Sigma^{\mathbb{Z}^d} \to \Sigma^{\mathbb{Z}^d}$ which applies the local rule simultaneously everywhere.



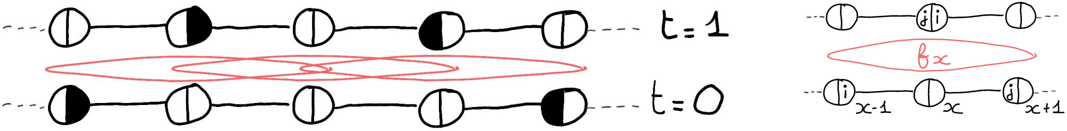
Cellular automata

- **Local rule :** function $f : \Sigma^{\mathcal{N}} \to \Sigma$.
- **Cellular automata :** function $A : \Sigma^{\mathbb{Z}^d} \to \Sigma^{\mathbb{Z}^d}$ which applies the local rule simultaneously everywhere.



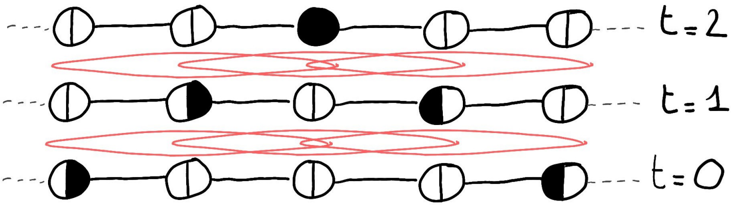
Cellular automata

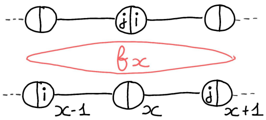
- **Local rule :** function $f : \Sigma^{\mathcal{N}} \to \Sigma$.
- **Cellular automata :** function $A : \Sigma^{\mathbb{Z}^d} \to \Sigma^{\mathbb{Z}^d}$ which applies the local rule simultaneously everywhere.



Cellular automata

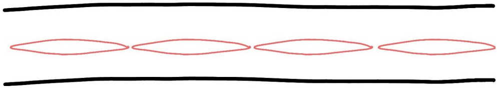
- **Local rule :** function $f : \Sigma^{\mathcal{N}} \to \Sigma$.
- **Cellular automata :** function $A : \Sigma^{\mathbb{Z}^d} \to \Sigma^{\mathbb{Z}^d}$ which applies the local rule simultaneously everywhere.



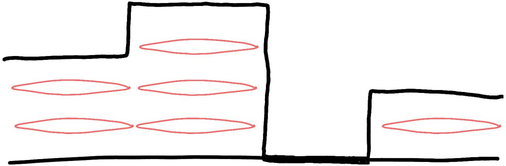


Global clock versus rewriting

Dynamical systems assume a global clocks. . .



. . . whereas rewriting systems can apply the local rule more freely.

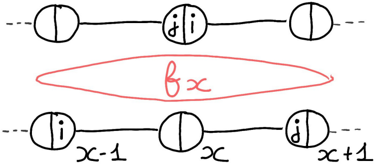


Rewriting and consistency

However we might end up with
inconsistencies.

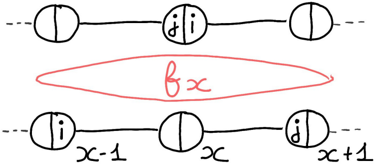
Rewriting and consistency

However we might end up with inconsistencies.



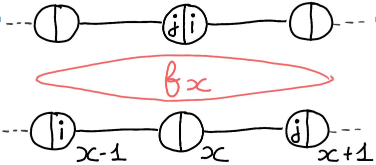
Rewriting and consistency

However we might end up with inconsistencies.



G

Rewriting and consistency



However we might end up with inconsistencies.

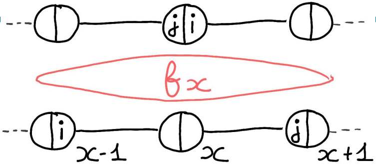
G

G

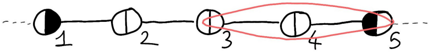
G

Rewriting and consistency

However we might end up with inconsistencies.



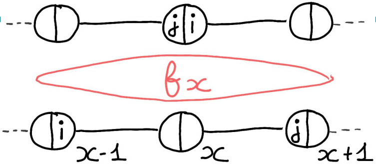
G



G

G

Rewriting and consistency



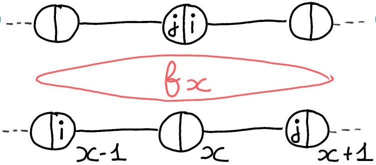
However we might end up with inconsistencies.

G

A_4G

G

Rewriting and consistency



However we might end up with inconsistencies.

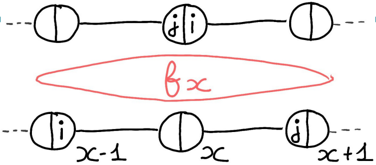
G

A_3A_4G

G

Rewriting and consistency

However we might end up with inconsistencies.

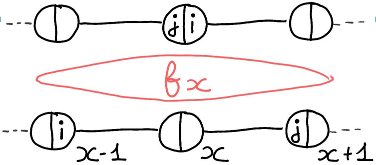


G

$A_2A_3A_4G$

G

Rewriting and consistency



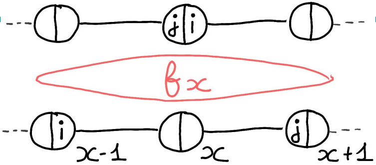
However we might end up with inconsistencies.

G

$A_2A_3A_4G$

A_2G

Rewriting and consistency



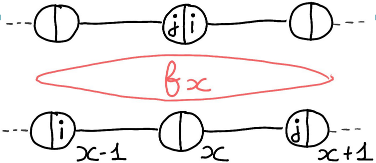
However we might end up with inconsistencies.

G

$A_2A_3A_4G$

A_3A_2G

Rewriting and consistency



However we might end up with inconsistencies.

G

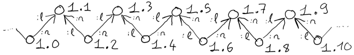
$A_2A_3A_4G$

$A_4A_3A_2G$

2 - Examples

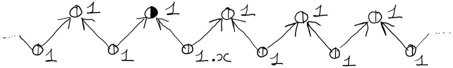
Definition

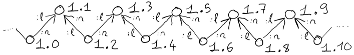
- **Space :** graph of "dimension" 1 (two ports).



Definition

- **Space :** graph of "dimension" 1 (two ports).
- **Internal states:** finite alphabet $\Sigma = \{0, 1\}^2$.

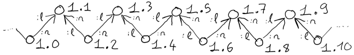


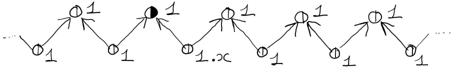


G

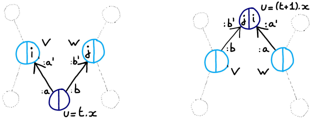
Definition

- **Space :** graph of "dimension" 1 (two ports).
- **Internal states:** finite alphabet $\Sigma = \{0,1\}^2$.





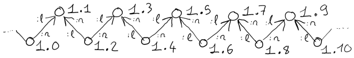
G

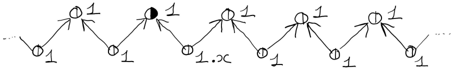


Local rule

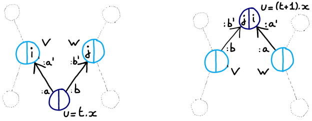
Definition

- **Space :** graph of "dimension" 1 (two ports).
- **Internal states:** finite alphabet $\Sigma = \{0, 1\}^2$.





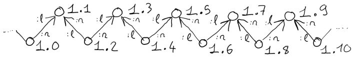
G

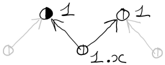


Local rule

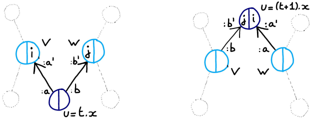
Definition

- **Space :** graph of "dimension" 1 (two ports).
- **Internal states:** finite alphabet $\Sigma = \{0, 1\}^2$.





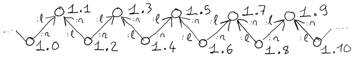
$G_{\mathcal{N}_x}$

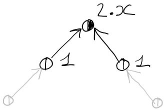


Local rule

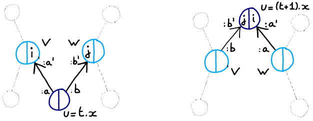
Definition

- **Space :** graph of "dimension" 1 (two ports).
- **Internal states:** finite alphabet $\Sigma = \{0,1\}^2$.





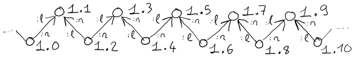
$A_x G_{\mathcal{N}_x}$

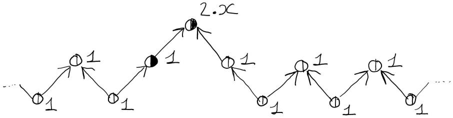


Local rule

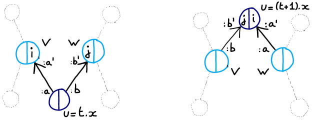
Definition

- **Space :** graph of "dimension" 1 (two ports).
- **Internal states:** finite alphabet $\Sigma = \{0,1\}^2$.



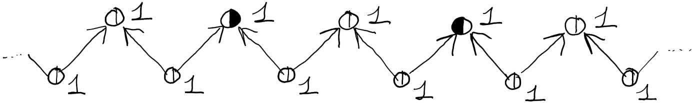


$A_x G$

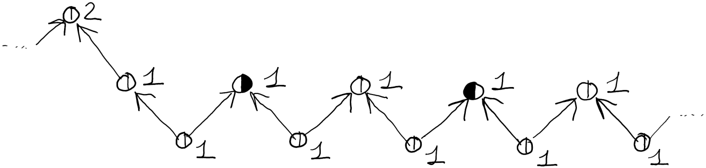


Local rule

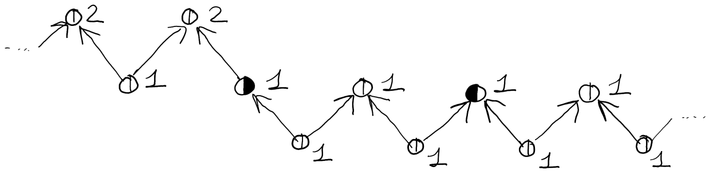
Particle system example



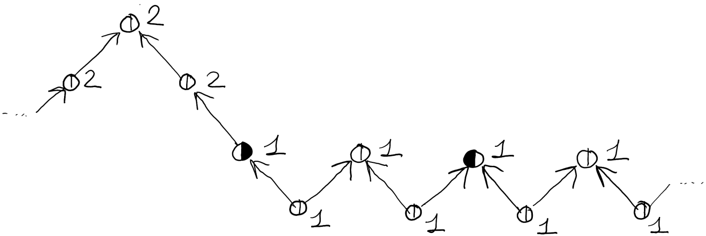
Particle system example



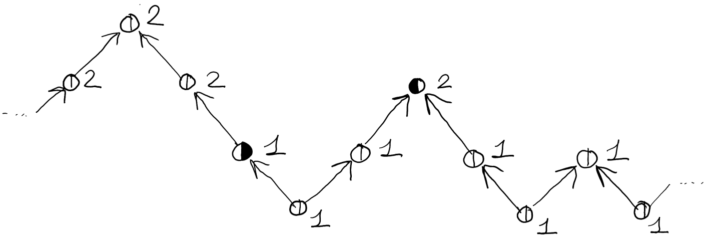
Particle system example



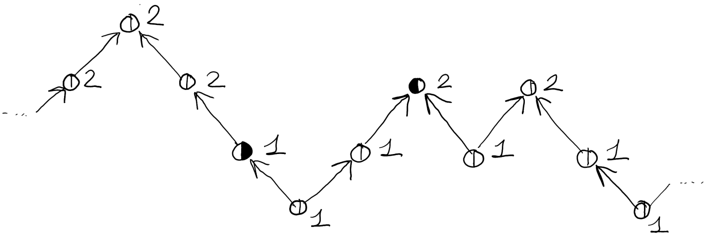
Particle system example



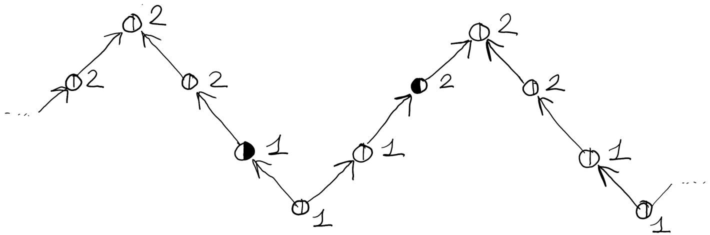
Particle system example



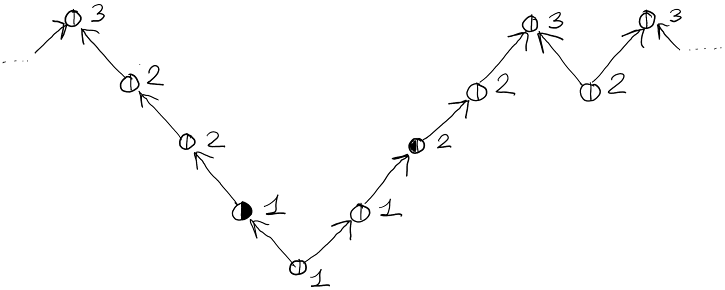
Particle system example



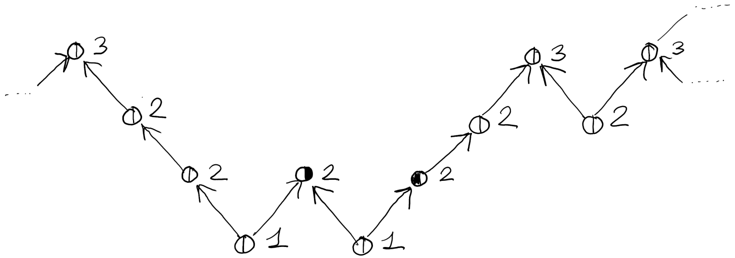
Particle system example



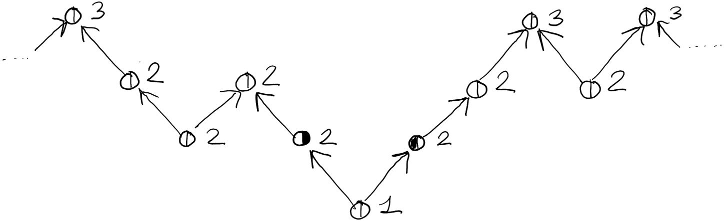
Particle system example



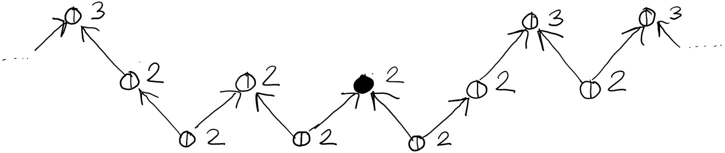
Particle system example



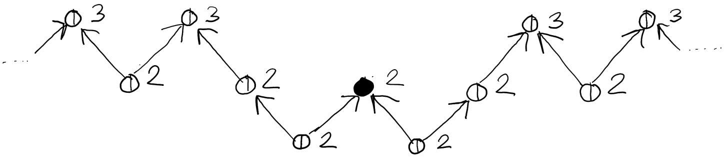
Particle system example



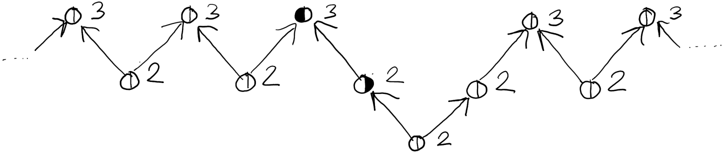
Particle system example



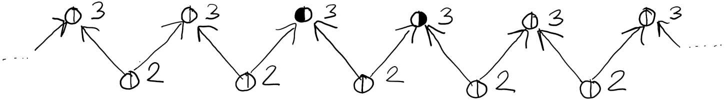
Particle system example



Particle system example

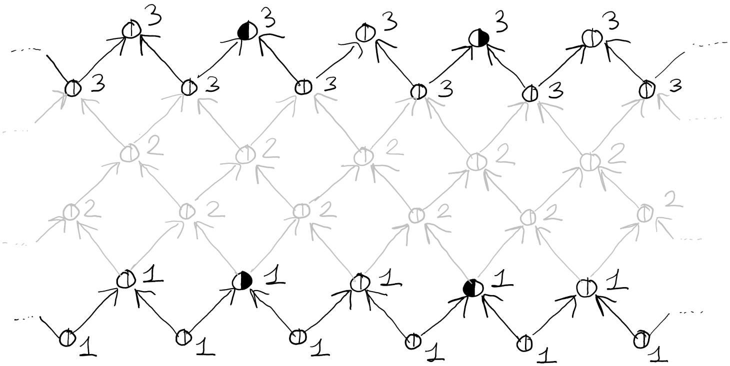


Particle system example



Particle system example

Particle system example



2 - Examples

Motivations

Examples

- Particle system example
- Time dilation example
- Simulating any CA

Consistency

Conclusion

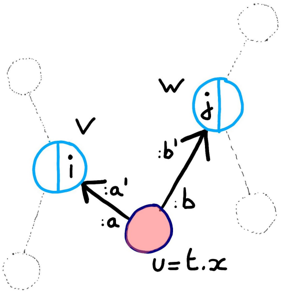
Definition

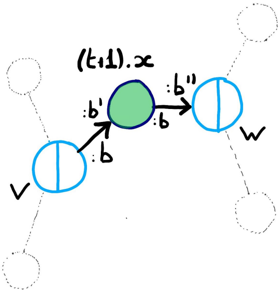
- **Internal states:** finite alphabet
  $\Sigma = \{0,1\}^2 \cup \{r, g\}$.

Definition

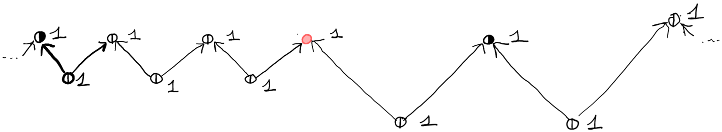
- **Internal states:** finite alphabet
  $\Sigma = \{0,1\}^2 \cup \{r, g\}.$

- **Local rule :** the evolution of a red vertex creates an anomaly.

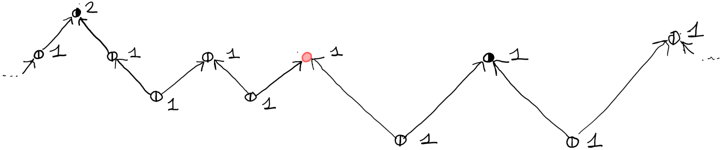




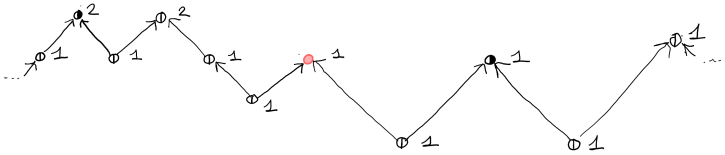
Time dilation example



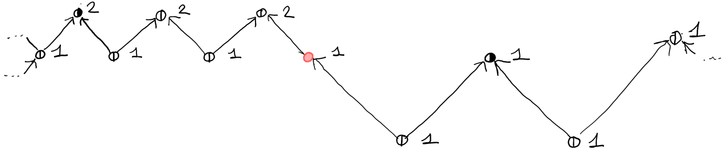
Time dilation example



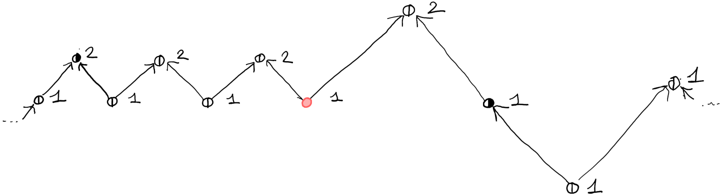
Time dilation example



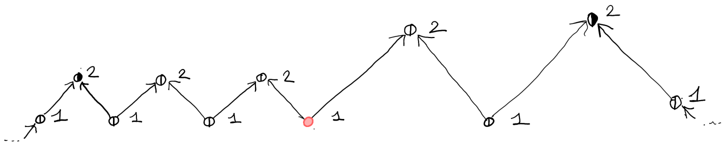
Time dilation example



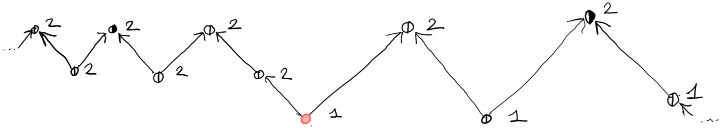
Time dilation example



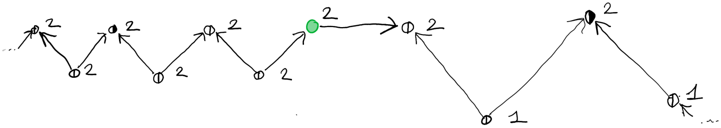
Time dilation example



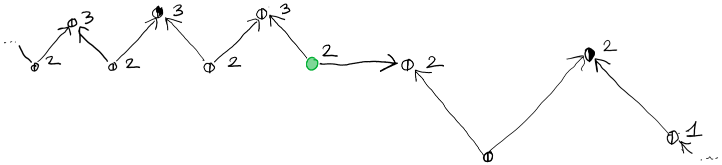
Time dilation example



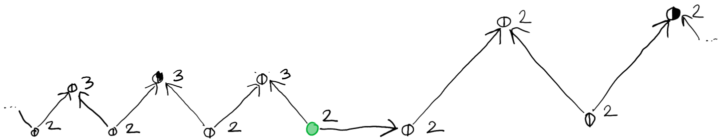
Time dilation example



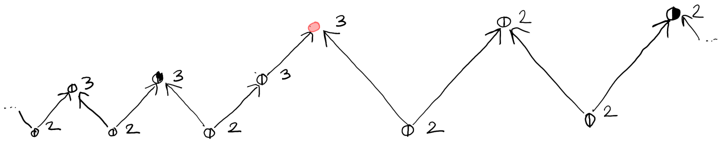
Time dilation example



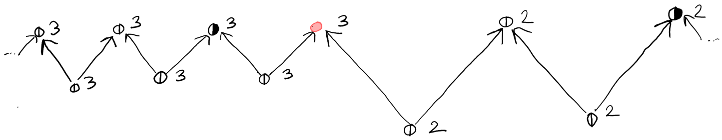
Time dilation example



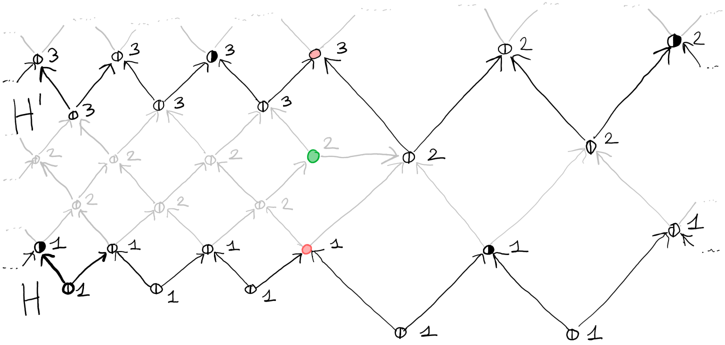
Time dilation example



Time dilation example



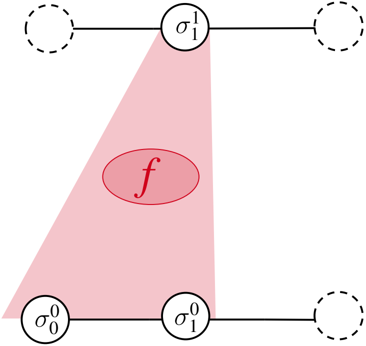
Time dilation example



2 - Examples

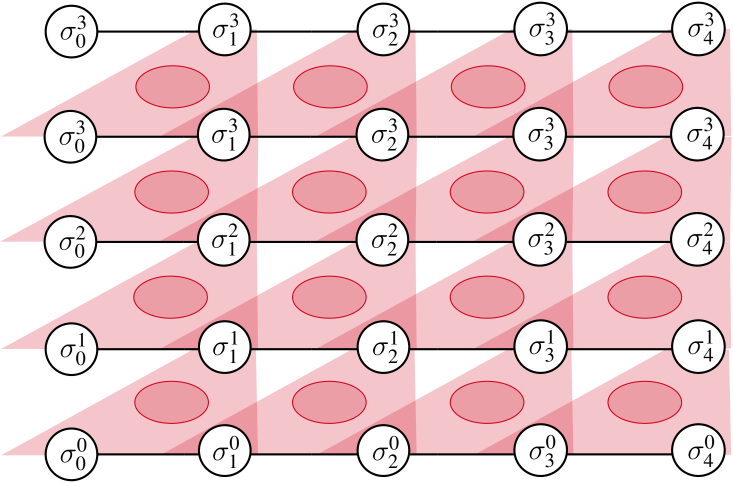
Radius one half CA

Radius one half cellular automata are universal (see [IBAR87]).

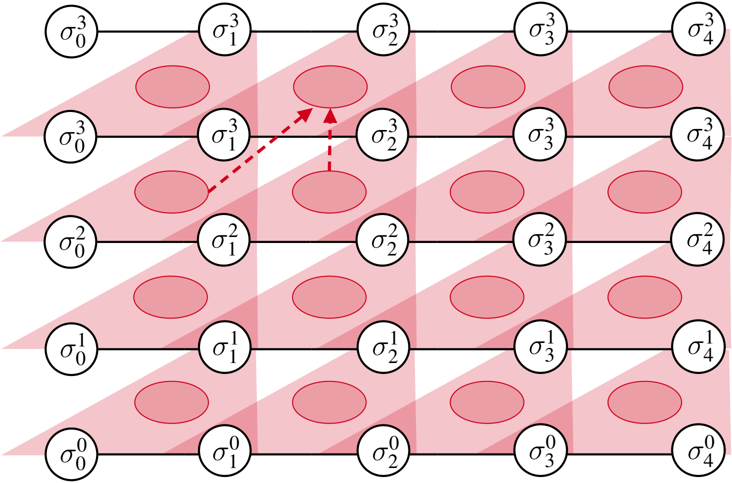


The local rule f of a radius one half CA

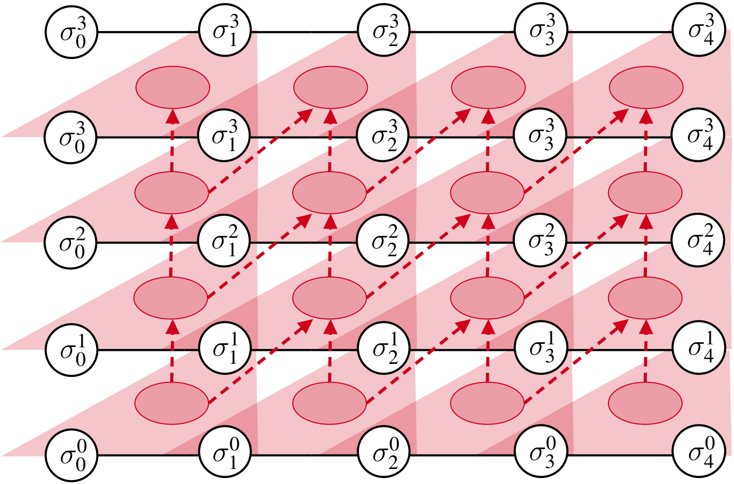
Causal structure



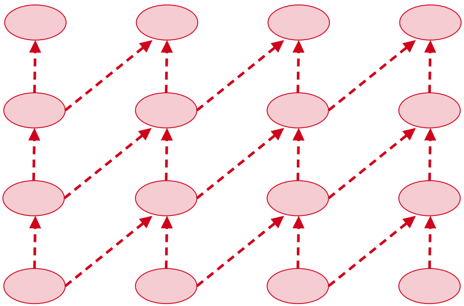
Causal structure



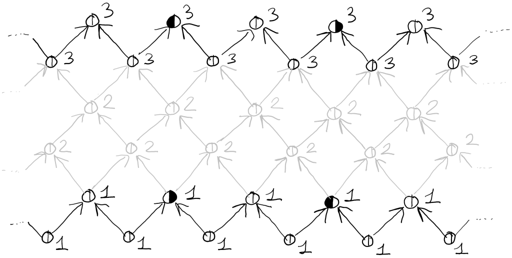
Causal structure



Simulating $1D$-CA



The dynamic of this synchronous CA . . .

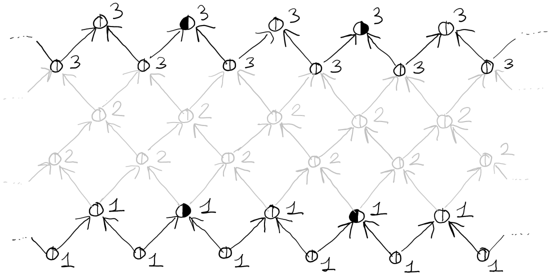


. . . can be simulated by this rewriting system.

3 - Consistency

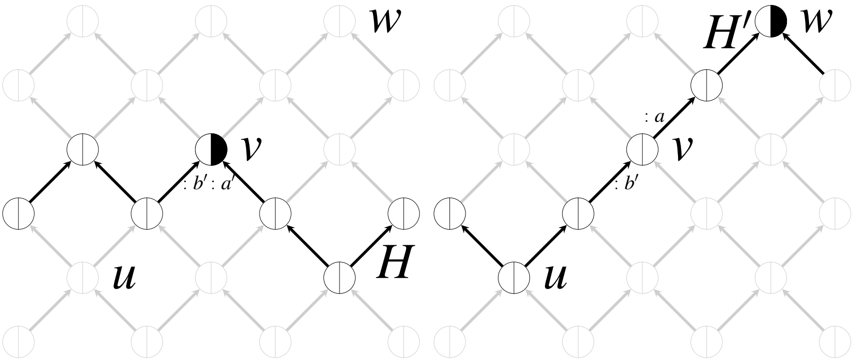
A first definition . . .

$$G_v = H_v.$$



…dismissed by the example

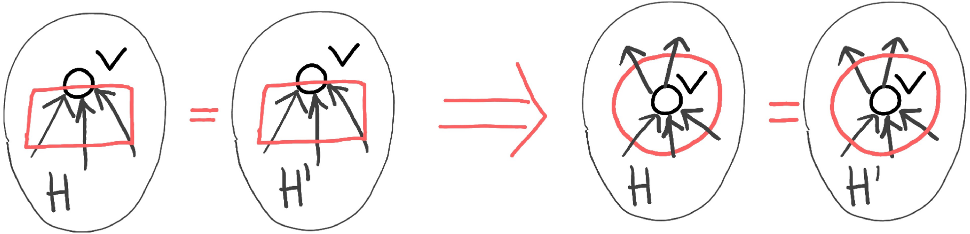
$$G_v = H_v.$$



Here $\sigma_H(v) = (0,1)\ldots$ … there $\sigma_{H'}(v) = (0,0)$.

Consistency

$$\pi.\mathrm{E}_G(v) = \pi.\mathrm{E}_H(v) \implies G_v = H_v.$$



Main result

How can we ensure consistency ?

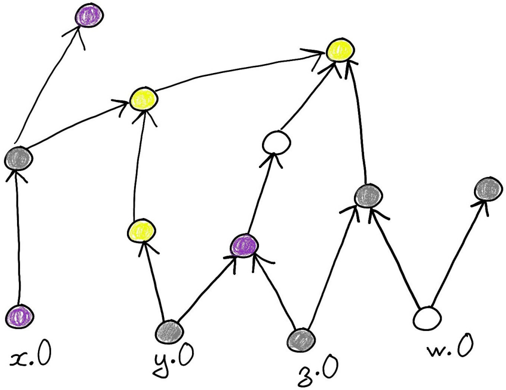
Main result

How can we ensure consistency ?

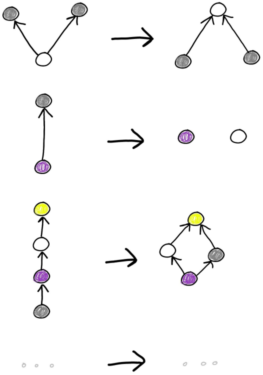
Theorem 1

Any **commutative**, **private** and **port-decreasing** local rule generates a **consistent** space-time diagram.

Framework

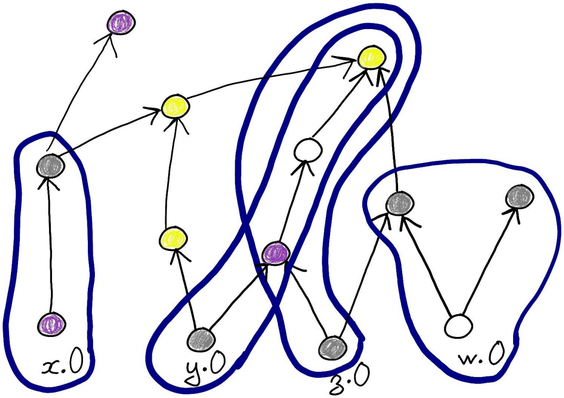


A configuration G.

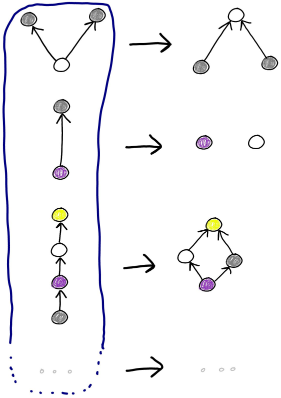


A local rule $A_{(-)}$.

Framework

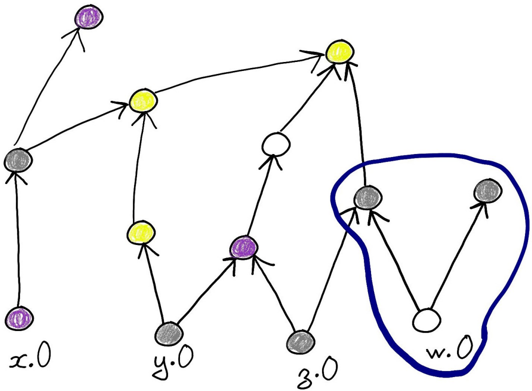


A configuration G.

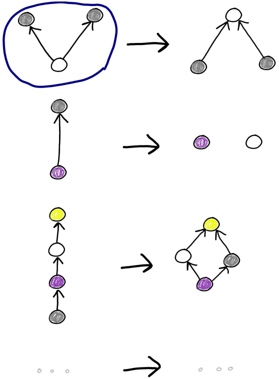


A local rule $A_{(-)}$.

Framework

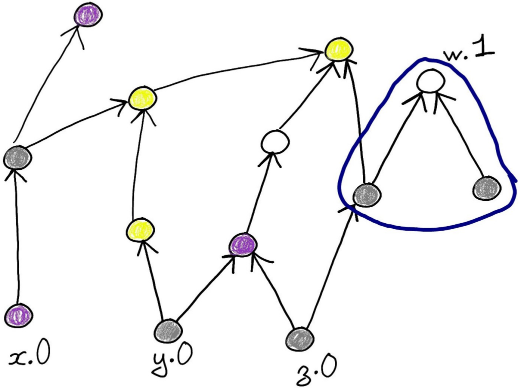


A configuration G.

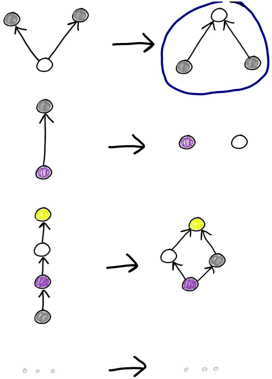


A local rule $A_{(-)}$.

Framework



A configuration $A_w G$.

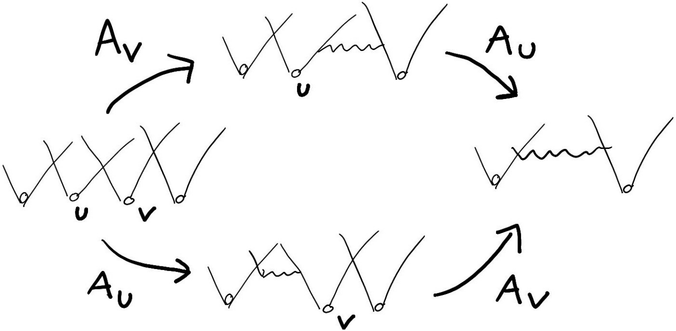


A local rule $A_{(-)}$.

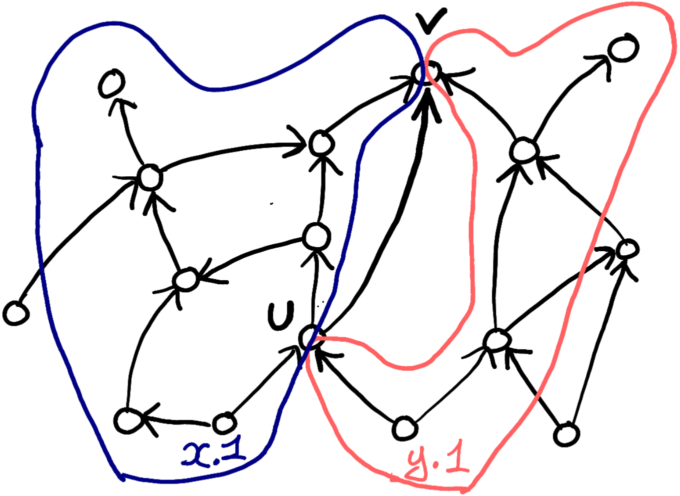
école normale supérieure paris-saclay

H1 : Commutation

$$A_u A_v(G) = A_u A_v(G)$$

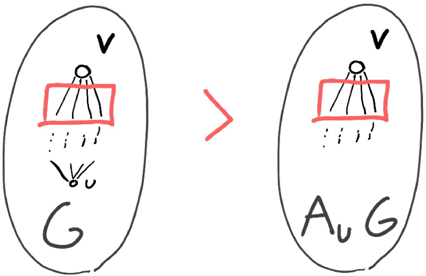


H2 : Privacy



H3 : Port-decreasing

$$\pi.\mathrm{E}_G(v) > \pi.\mathrm{E}_{A_u G}(v)$$



Summary

Our model is :

- Based on **graph rewriting**, . . .

Summary

Our model is :

- Based on **graph rewriting**, . . .
- . . . able to simulate any synchronous cellular automata . . .

Summary

Our model is :

- Based on **graph rewriting**, . . .
- . . . able to simulate any synchronous cellular automata . . .
- . . . and to encode some intrinsically non synchronous example.

Summary

Our model is :

- Based on **graph rewriting**, . . .
- . . . able to simulate any synchronous cellular automata . . .
- . . . and to encode some intrinsically non synchronous example.

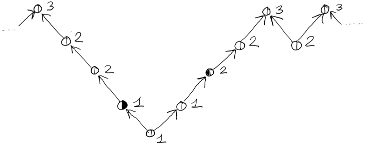
Theorem 1

Any **commutative**, **private** and **port-decreasing** local rule generates a **consistent** space-time diagram.

Future work

Reversibility

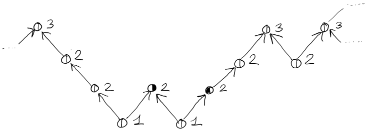
$$\exists A_{(-)}^{-1},\ A_x^{-1} A_x G = A_x A_x^{-1} G = G$$



Future work

Reversibility

$$\exists A_{(-)}^{-1},\ A_x^{-1} A_x G = A_x A_x^{-1} G = G$$



Future work

Reversibility

$$\exists A_{(-)}^{-1},\ A_x^{-1} A_x G = A_x A_x^{-1} G = G$$