Points, Distances, and Cellular Automata: Geometric and Spatial Algorithmics

Luidnel Maignan
ALCHEMY: INRIA SACLAY, LRI, UNIV. PARIS XI
luidnel.maignan@inria.fr

New World of Computation 2011
Orléan - 23-24 May 2011

Introduction

Spatial Computing and Cellular Automata

- Massively Distributed Systems $\Rightarrow$ Spatial Features

Introduction

Spatial Computing and Cellular Automata

- Massively Distributed Systems $\Rightarrow$ Spatial Features
- Why? Physics and Locality

Introduction

Spatial Computing and Cellular Automata

- Massively Distributed Systems $\Rightarrow$ Spatial Features
- Why? Physics and Locality
- Exemple? Computer Architecture, Communication

Introduction

Spatial Computing and Cellular Automata

- Spatial Computing: focus on space
- Cellular Automata: simple framework, precise results

Global Statement

In the same manner that geometry is deeply based on distances, basing spatial algorithmics on the intrinsic metric of the spatial computers leads to more precise and generic formulation.

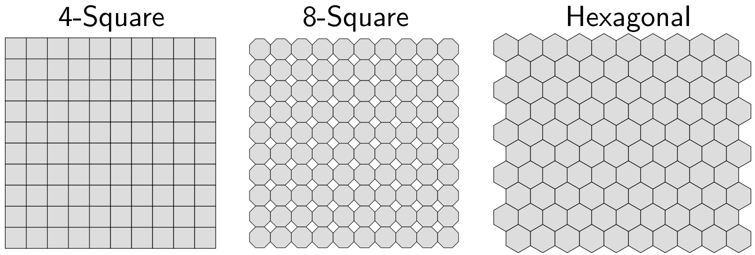
Outline

Space, Time, and Cellular Automata

Cellular Automata

Cellular Automata

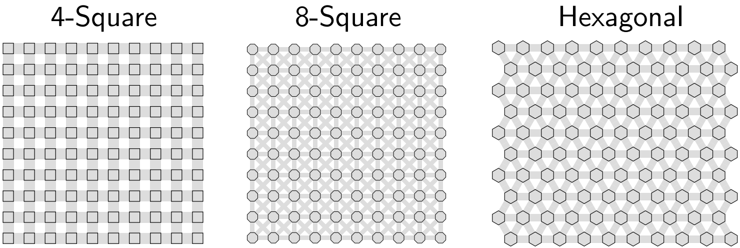
- Regular lattice of cells, also called **sites**, (or **points**)
- All sites **states** are **updated synchronously**
- State updates depends only on **neighborhood** sites states



Cellular Automata

Cellular Automata

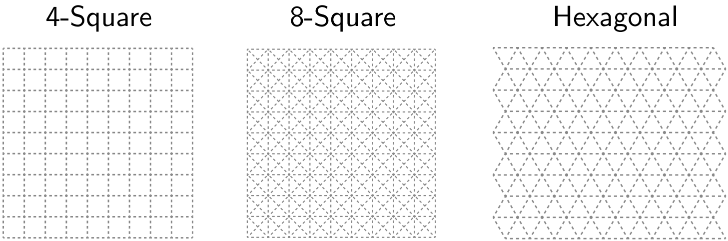
- Regular lattice of cells, also called **sites**, (or **points**)
- All sites **states** are **updated synchronously**
- State updates depends only on **neighborhood** sites states



Cellular Automata

Cellular Automata

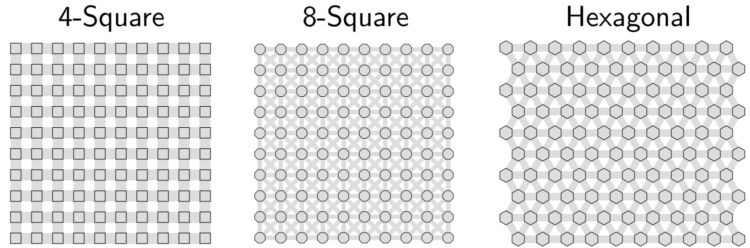
- Regular lattice of cells, also called **sites**, (or **points**)
- All sites **states** are **updated synchronously**
- State updates depends only on **neighborhood** sites states



Cellular Automata and Distances

Directions and Distances

- Traditionnaly, neighbors are named North, South, East, West
- In this work, no direction, only the graph and its metric
- Distances only ⇔ Rotational invariance



Distance Fields and Gradients

Classical Definition and Computation

Definition (Distance Map)

The distance map D_P of a given set of particles P associates to each point x its distance $d(x, y)$ to the closest particle $y \in P$.

$$D_P(x) = d(P, x) = \min\{\, d(x, y) \mid y \in P \,\}.$$

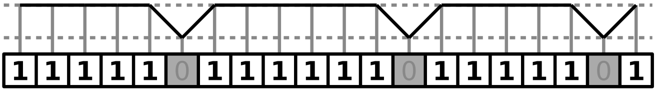
Classical Distance Field

$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$

Distance Field Evolution

Classical Distance Field

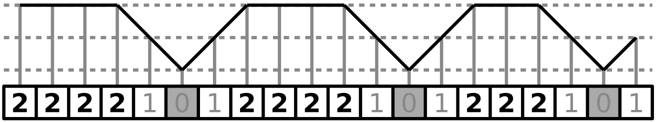
$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



Distance Field Evolution

Classical Distance Field

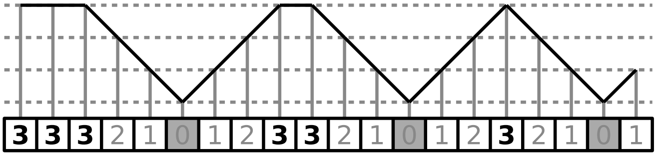
$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



Distance Field Evolution

Classical Distance Field

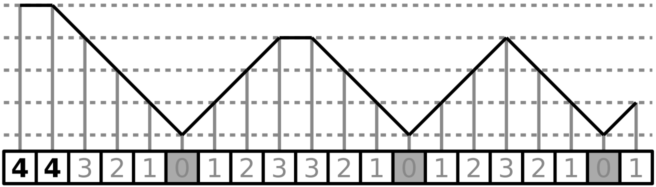
$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



Distance Field Evolution

Classical Distance Field

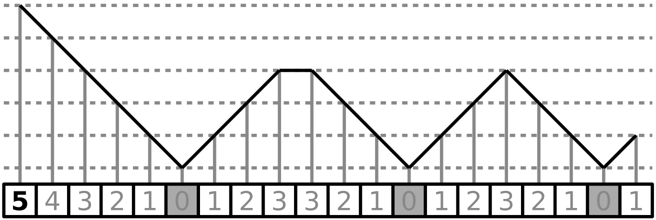
$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



Distance Field Evolution

Classical Distance Field

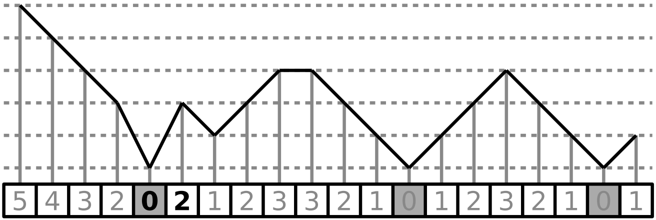
$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



Distance Field Evolution

Classical Distance Field

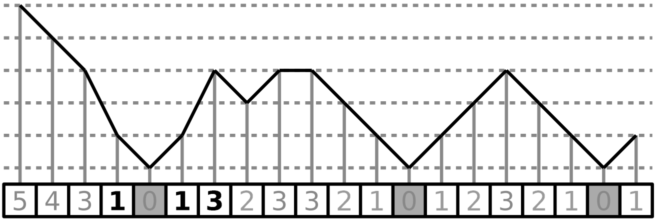
$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



Distance Field Evolution

Classical Distance Field

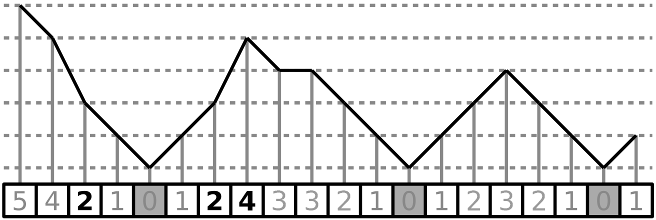
$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



Distance Field Evolution

Classical Distance Field

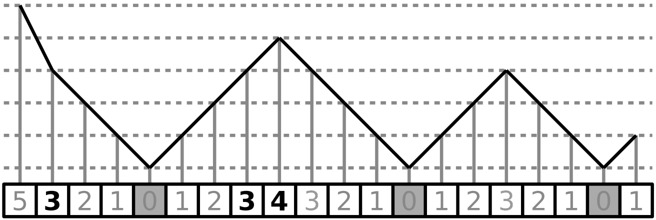
$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



Distance Field Evolution

Classical Distance Field

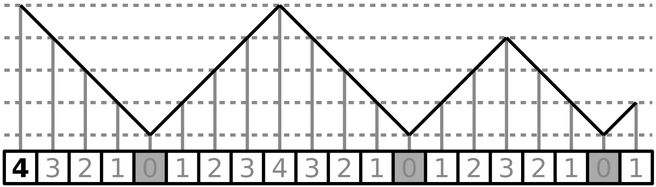
$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



Distance Field Evolution

Classical Distance Field

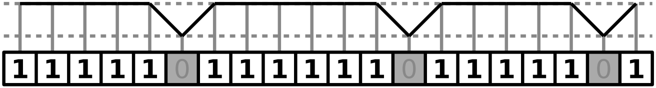
$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



Corrected Distance Field Evolution

Corrected Distance Field

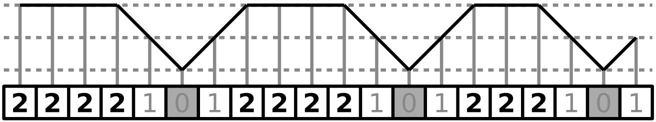
$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_{t+1} \text{ else:} \\ 0.5 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



Corrected Distance Field Evolution

Corrected Distance Field

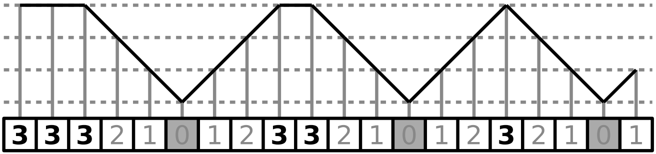
$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_{t+1} \text{ else:} \\ 0.5 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



Corrected Distance Field Evolution

Corrected Distance Field

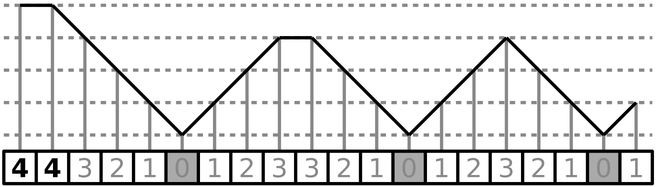
$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_{t+1} \text{ else:} \\ 0.5 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



Corrected Distance Field Evolution

Corrected Distance Field

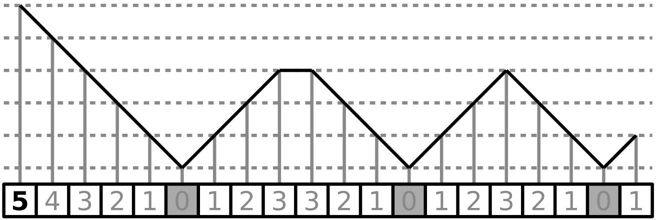
$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_{t+1} \text{ else:} \\ 0.5 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



Corrected Distance Field Evolution

Corrected Distance Field

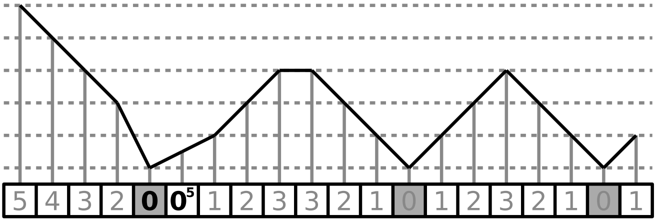
$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_{t+1} \text{ else:} \\ 0.5 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



Corrected Distance Field Evolution

Corrected Distance Field

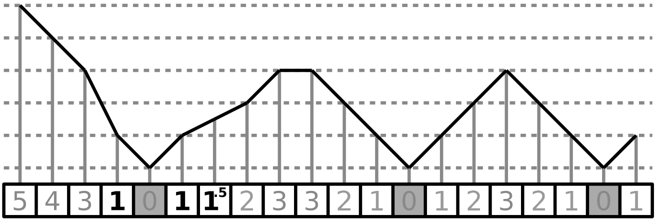
$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_{t+1} \text{ else:} \\ 0.5 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



Corrected Distance Field Evolution

Corrected Distance Field

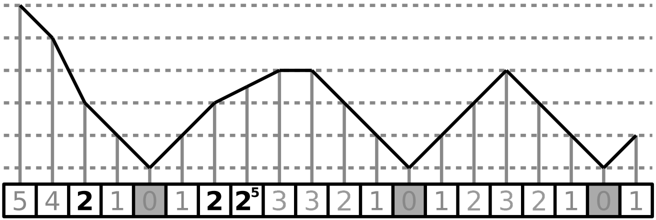
$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_{t+1} \text{ else:} \\ 0.5 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



Corrected Distance Field Evolution

Corrected Distance Field

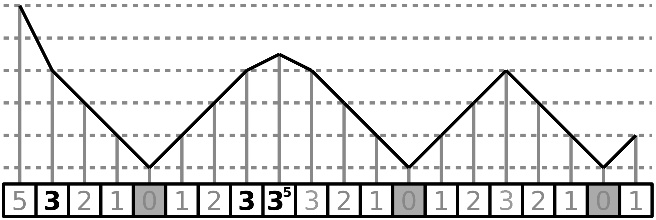
$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_{t+1} \text{ else:} \\ 0.5 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



Corrected Distance Field Evolution

Corrected Distance Field

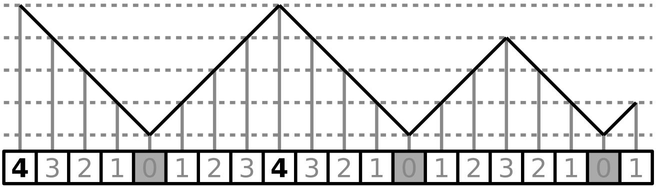
$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_{t+1} \text{ else:} \\ 0.5 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



Corrected Distance Field Evolution

Corrected Distance Field

$$D[P]_{t+1}(x) = \begin{cases} 0 \text{ if } x \in P_{t+1} \text{ else:} \\ 0.5 \text{ if } x \in P_t \text{ else:} \\ \min\{\, 1 + D[P]_t(y) \mid y \in N(x) \,\}. \end{cases}$$



From Infinite To Finite Field

Checkpoint

- We have: distances locally, globally, and dynamically
- We don't have: finite number of states

From Infinite To Finite Field

Checkpoint

- We have: distances locally, globally, and dynamically
- We don't have: finite number of states

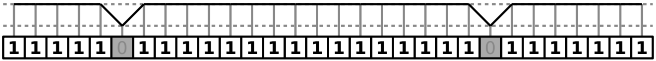
Bounded information

- No bound on distances
- Bounded gradient (differences between neighboring sites)
- What about modulo ?

From Infinite To Finite Field (Cont.)

Modulo in action

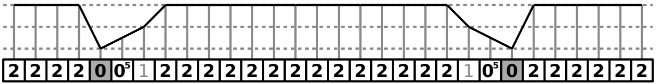
- Particles **maximal speed** determines **maximal gradient**



From Infinite To Finite Field (Cont.)

Modulo in action

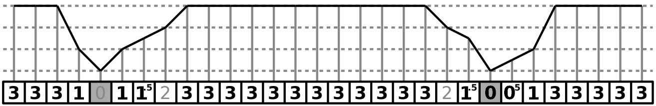
- Particles **maximal speed** determines **maximal gradient**



From Infinite To Finite Field (Cont.)

Modulo in action

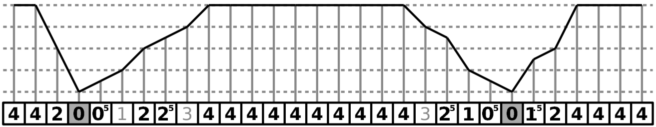
- Particles **maximal speed** determines **maximal gradient**



From Infinite To Finite Field (Cont.)

Modulo in action

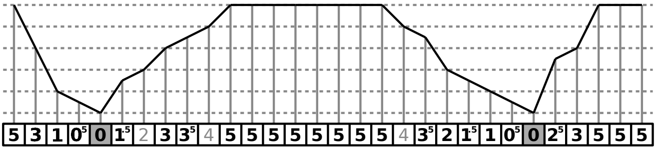
- Particles **maximal speed** determines **maximal gradient**



From Infinite To Finite Field (Cont.)

Modulo in action

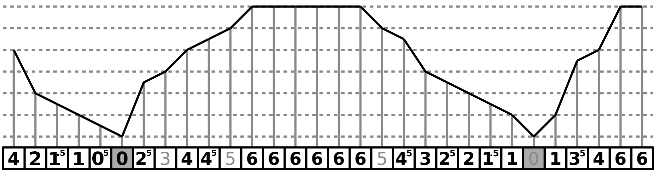
- Particles **maximal speed** determines **maximal gradient**



From Infinite To Finite Field (Cont.)

Modulo in action

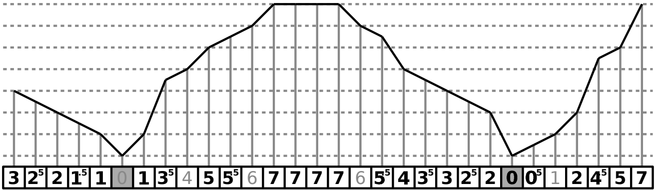
- Particles **maximal speed** determines **maximal gradient**



From Infinite To Finite Field (Cont.)

Modulo in action

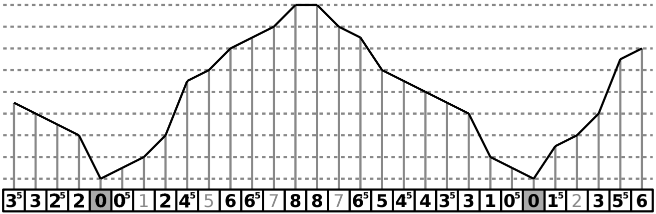
- Particles **maximal speed** determines **maximal gradient**



From Infinite To Finite Field (Cont.)

Modulo in action

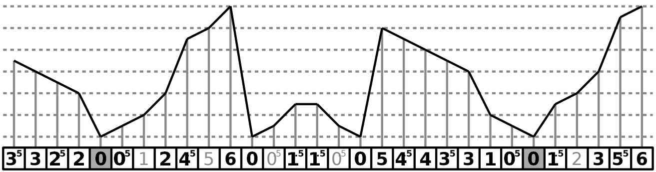
- Particles **maximal speed** determines **maximal gradient**
- In this case: 2 consecutive moves $\Rightarrow$ gradient bound of 3



From Infinite To Finite Field (Cont.)

Modulo in action

- Particles **maximal speed** determines **maximal gradient**
- In this case: gradient bound of 3 $\Rightarrow$ modulo 7



Building on top of distances

Distance fields as building blocks

- Moving according to the distance field
- Detecting patterns of distances and particles

Case Study

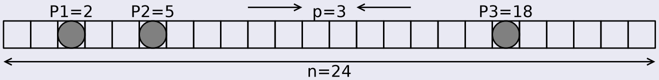
- Density Uniformisation (unidimensional)
- Convex Hull (multidimensional)
- Gabriel Graph (multidimensional)

Density Uniformisation

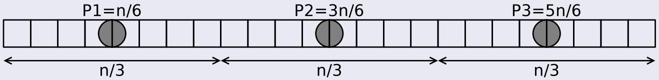
Problem Statement

Problem Definition

- Move the particles to a uniform distribution
- Input:



- Output:



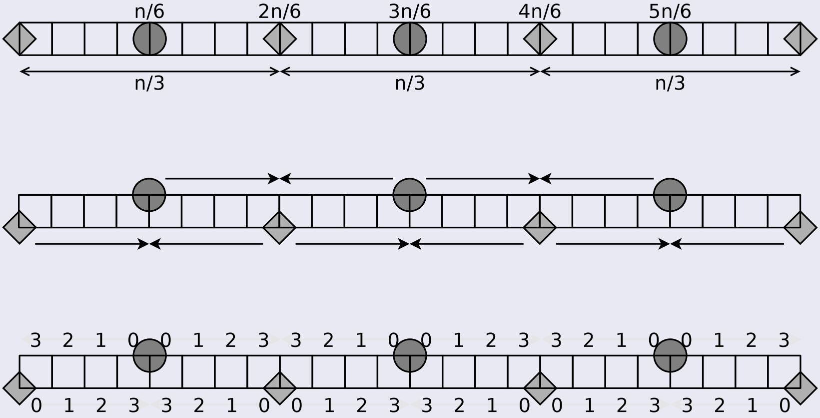
Problem Analysis

Intuition

- Each particle needs to occupy its space
- Boundary between individual spaces $\Leftrightarrow$ middles
- Occupy its space $\Leftrightarrow$ be at the middles

Application: 1D Uniformisation

Solution



The resulting system

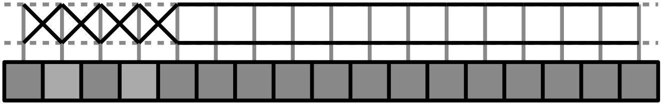
Initial system state

$$\begin{cases} \mathrm{p}_0(x) & = & x \in P \\ \mathrm{w}_0(x) & = & x \notin P \end{cases}$$

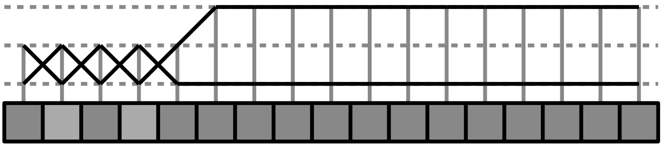
System fields composition

$$\begin{cases} \mathrm{dp} & = & D[\mathrm{p}] \\ \mathrm{dw} & = & D[\mathrm{w}] \\ \mathrm{p} & = & M[\mathrm{p}_0, B[\mathrm{dp}] \wedge \mathrm{Dir}[\mathrm{dw}, \leq]] \\ \mathrm{w} & = & M[\mathrm{w}_0, B[\mathrm{dw}] \wedge \mathrm{Dir}[\mathrm{dp}, \leq]] \end{cases}$$

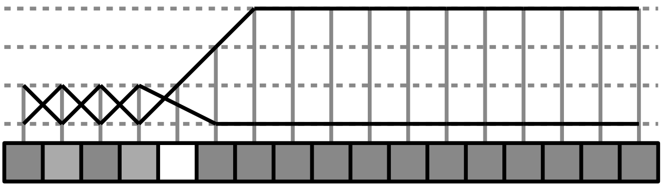
The resulting evolution



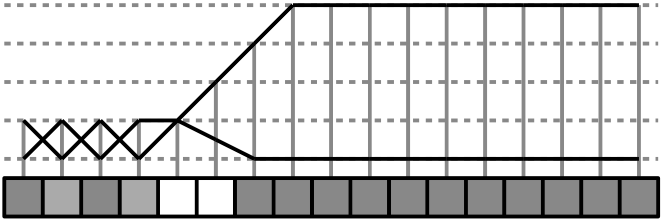
The resulting evolution



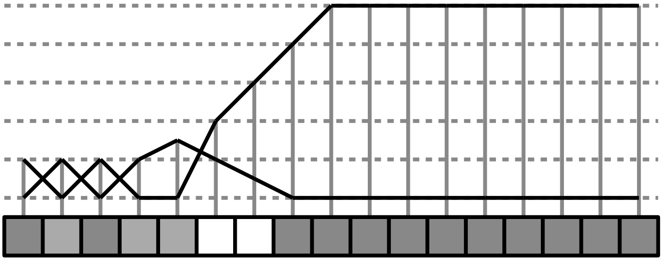
The resulting evolution



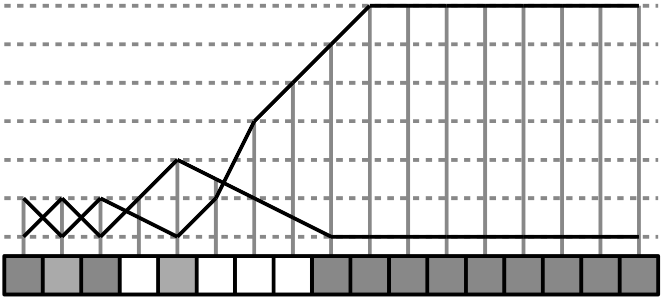
The resulting evolution



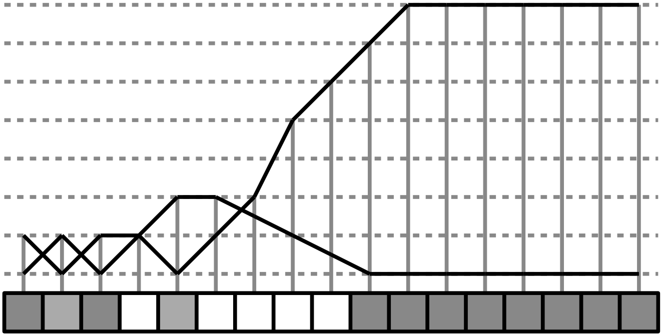
The resulting evolution



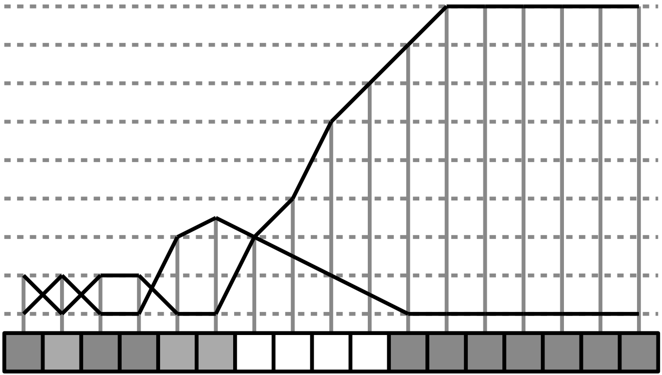
The resulting evolution



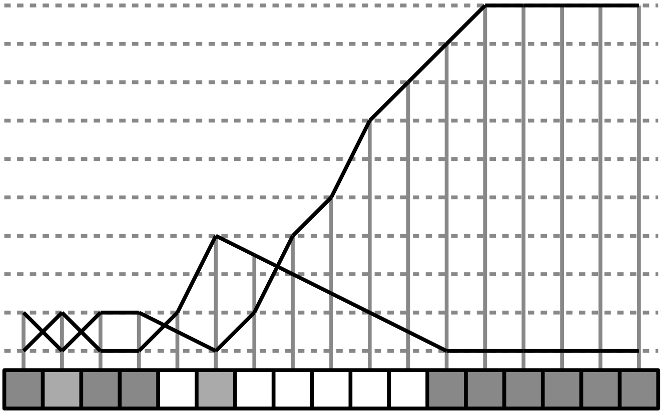
The resulting evolution



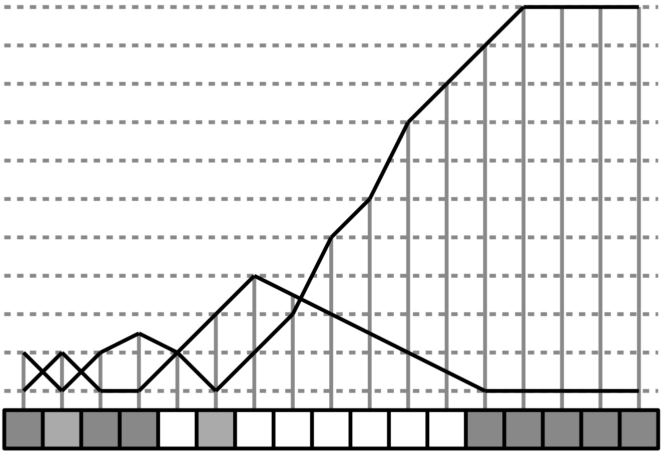
The resulting evolution



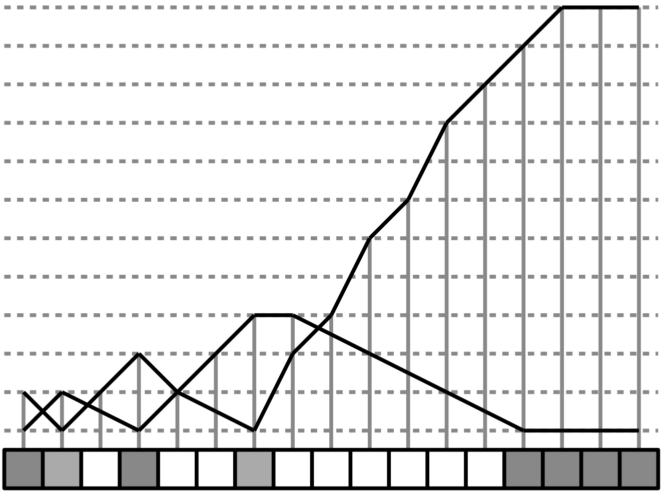
The resulting evolution



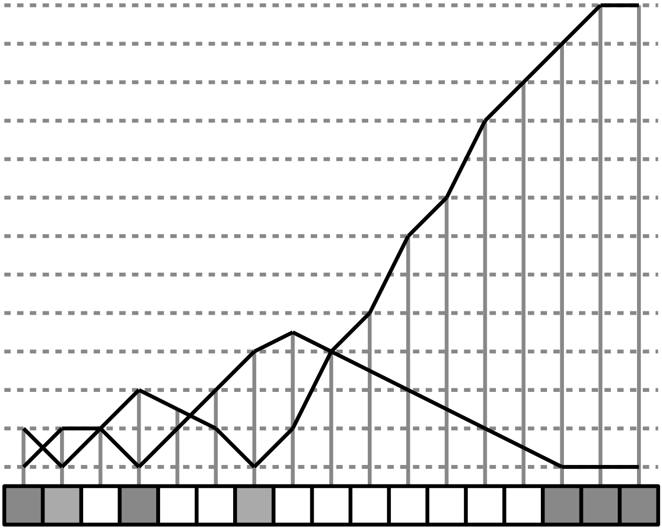
The resulting evolution



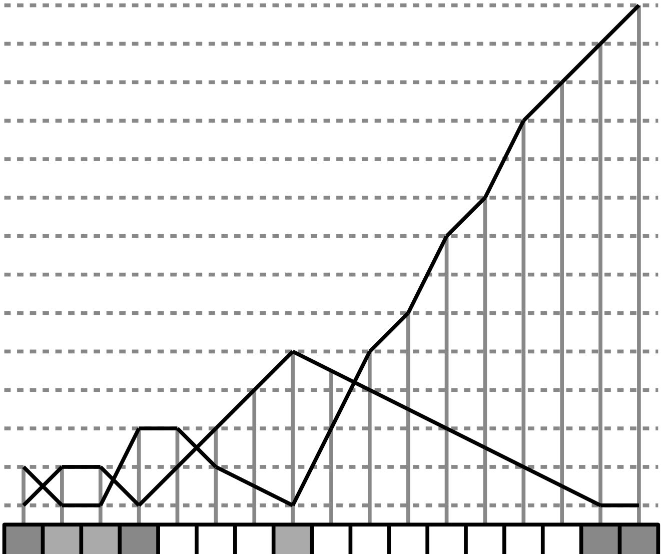
The resulting evolution



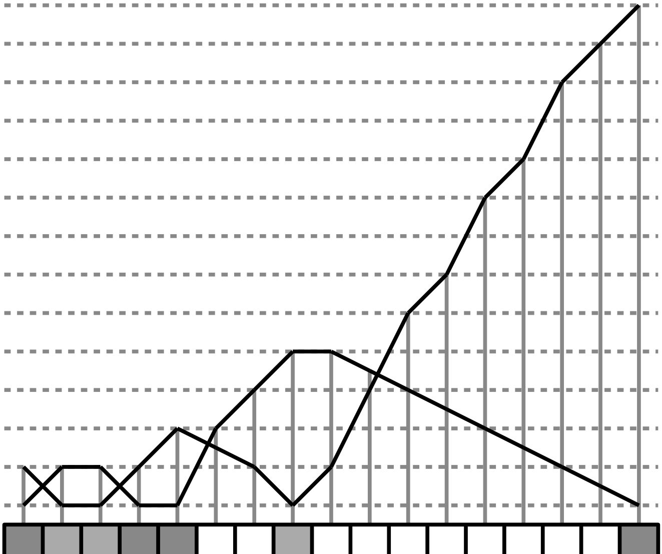
The resulting evolution



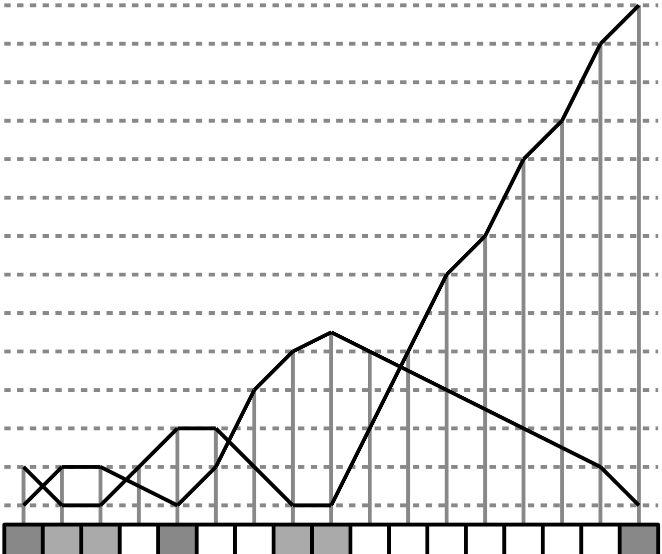
The resulting evolution



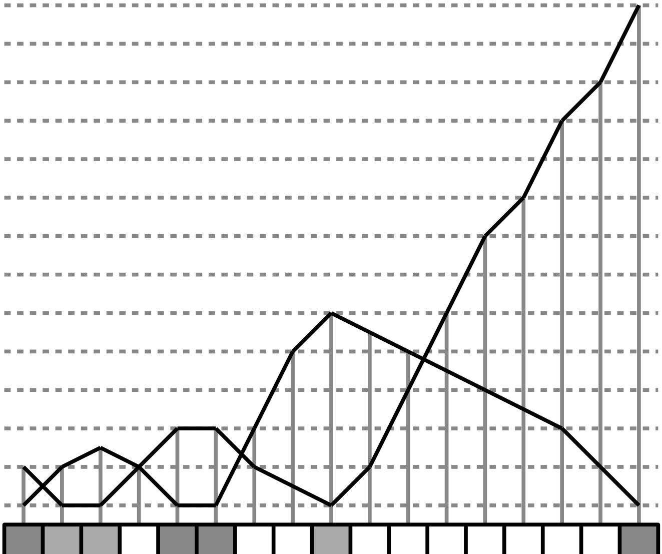
The resulting evolution



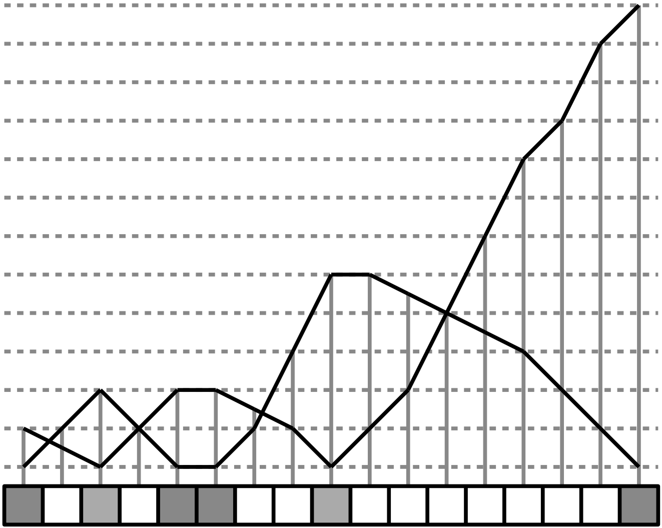
The resulting evolution



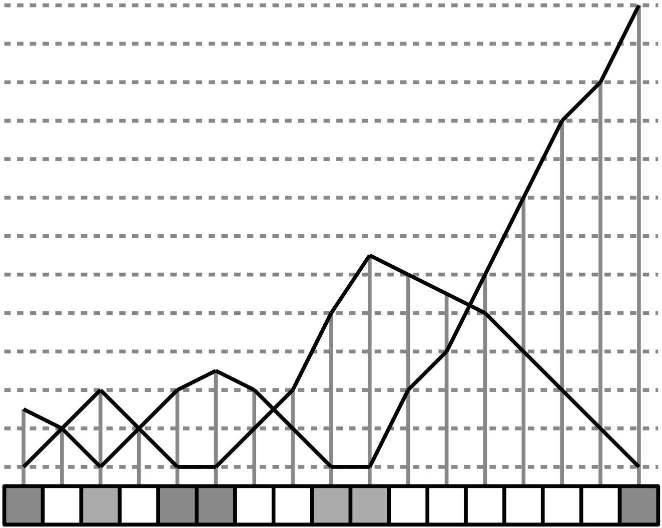
The resulting evolution



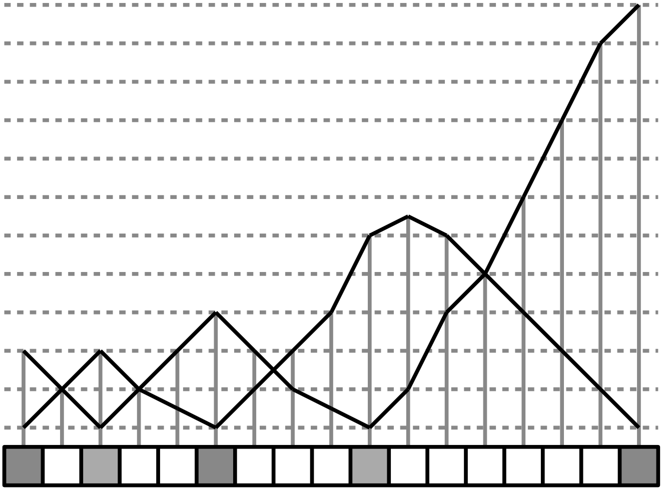
The resulting evolution



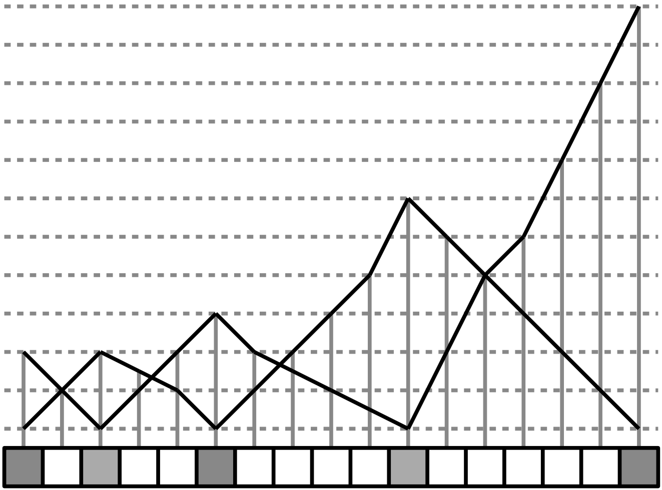
The resulting evolution



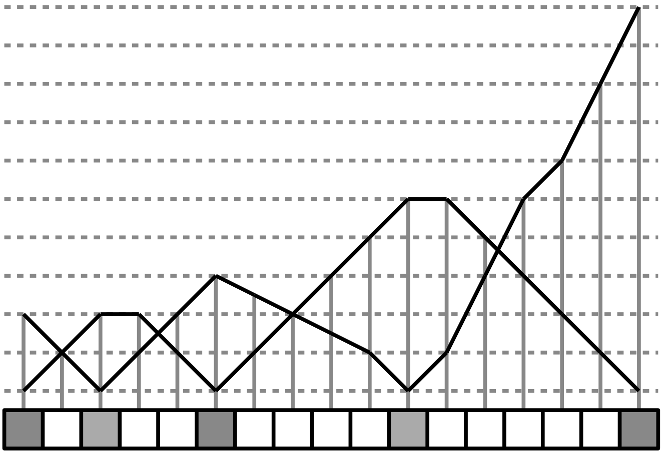
The resulting evolution



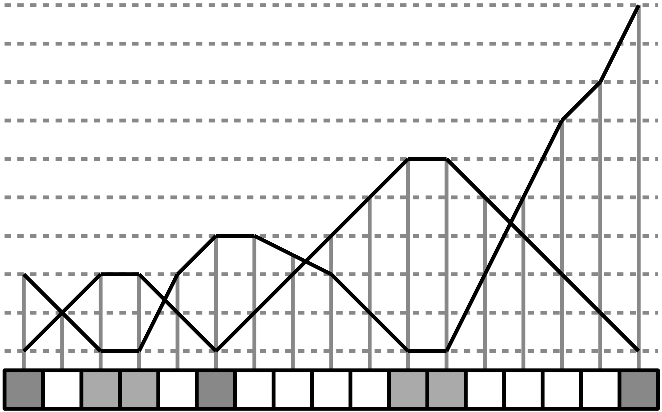
The resulting evolution



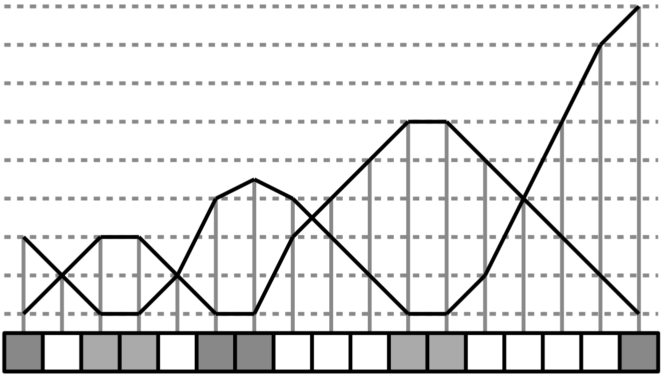
The resulting evolution



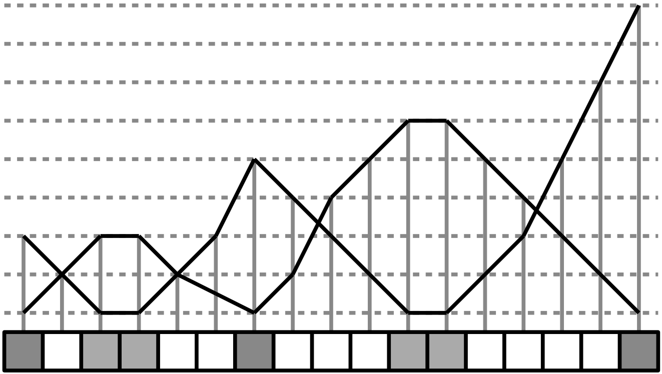
The resulting evolution



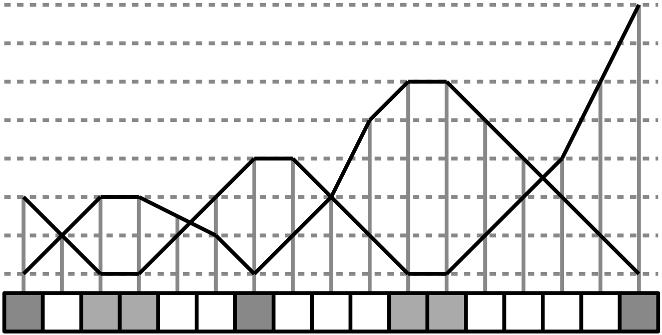
The resulting evolution



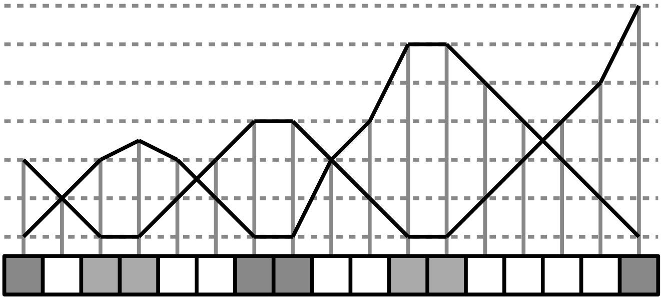
The resulting evolution



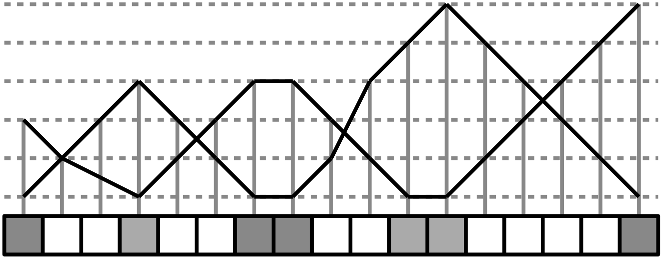
The resulting evolution



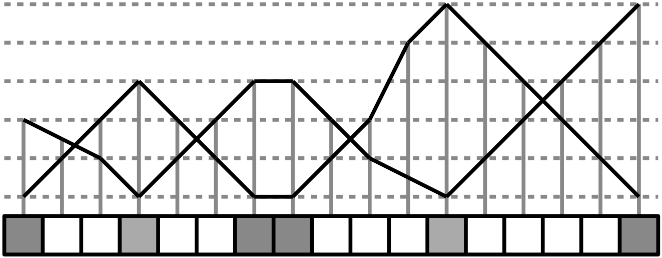
The resulting evolution



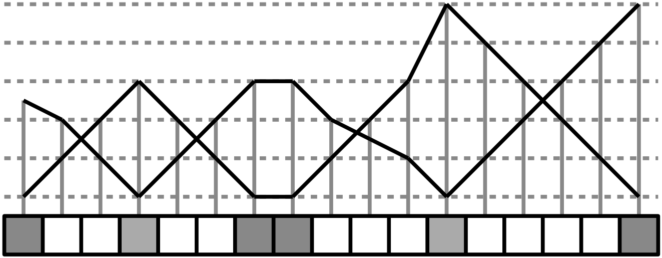
The resulting evolution



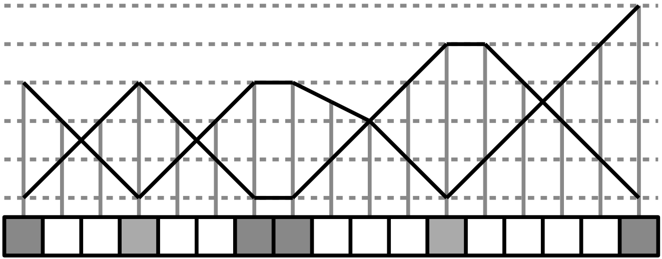
The resulting evolution



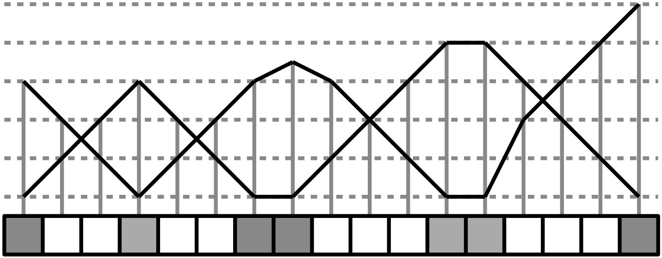
The resulting evolution



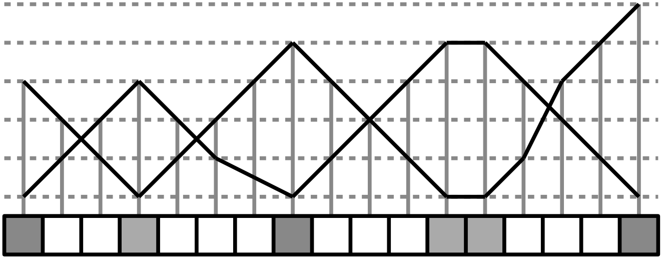
The resulting evolution



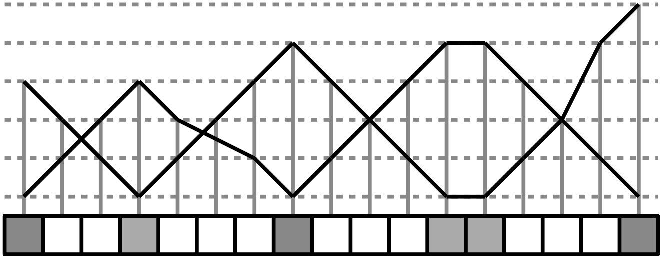
The resulting evolution



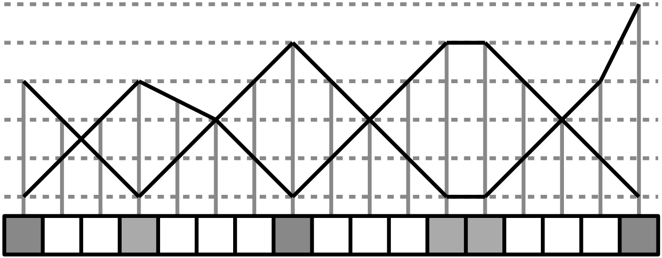
The resulting evolution



The resulting evolution



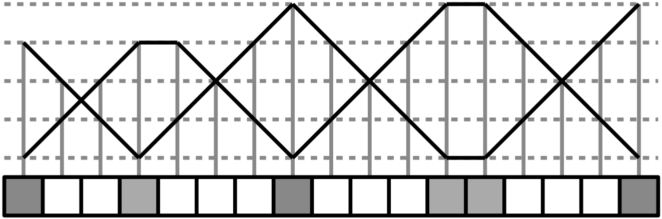
The resulting evolution



The resulting evolution

Signal and Dynamics

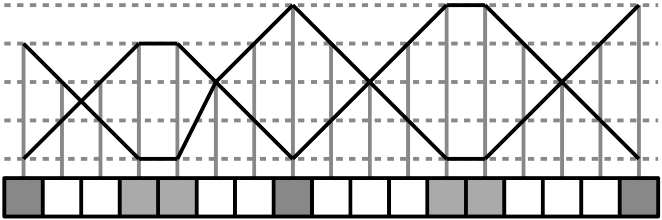
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

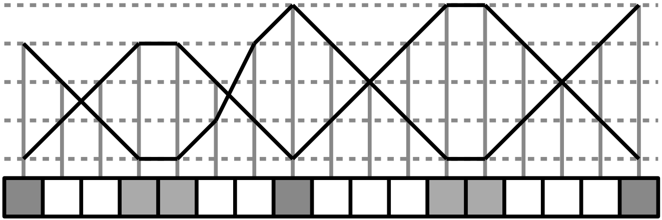
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

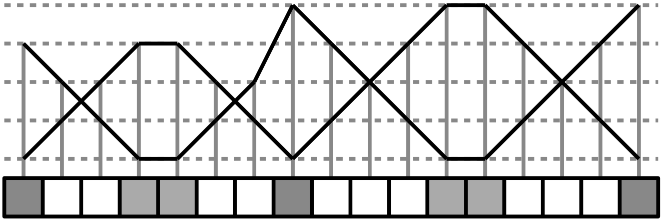
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

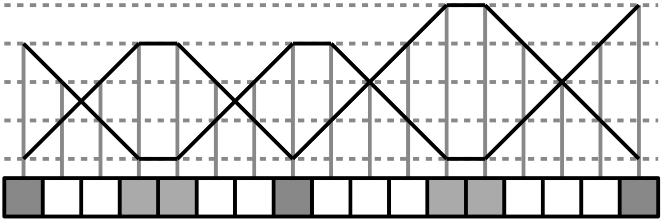
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

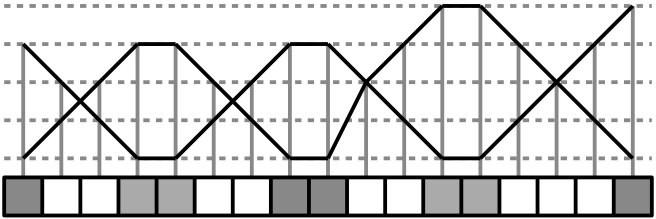
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

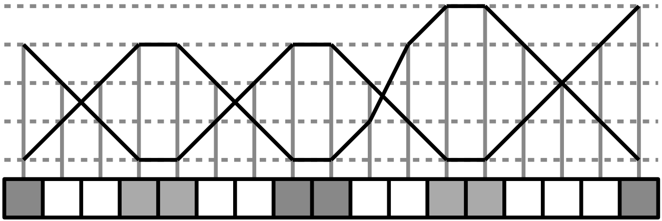
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

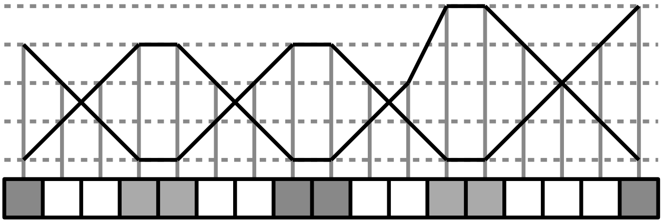
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

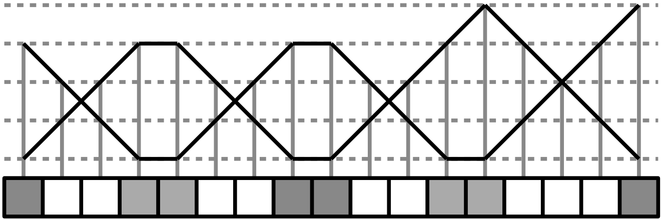
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

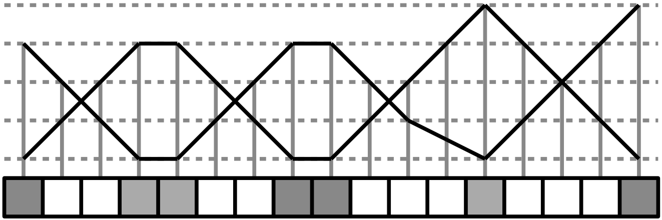
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

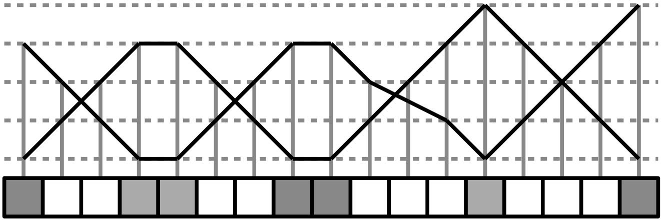
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

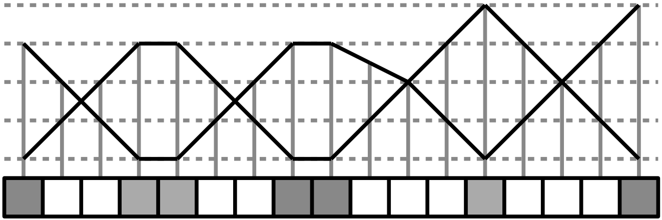
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

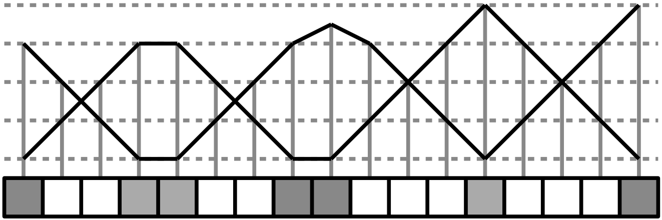
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

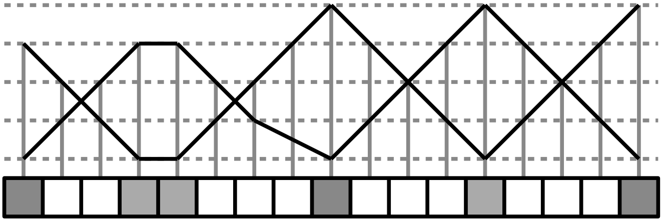
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

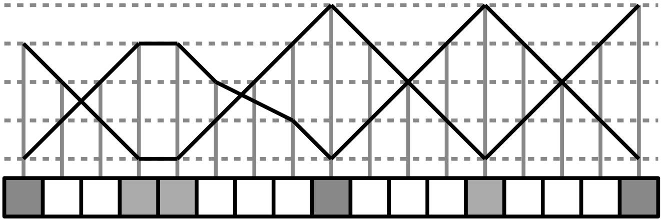
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

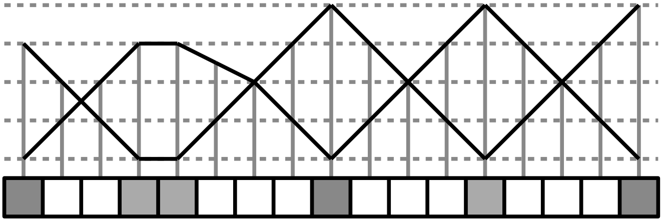
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

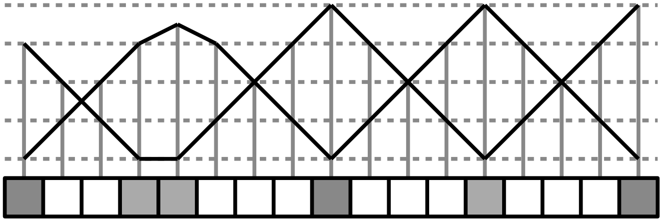
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

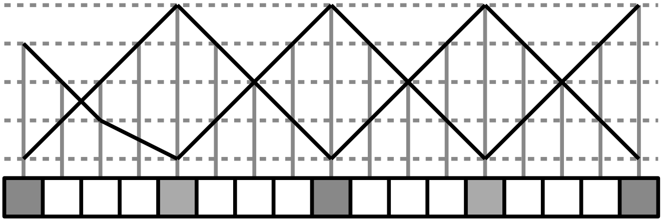
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

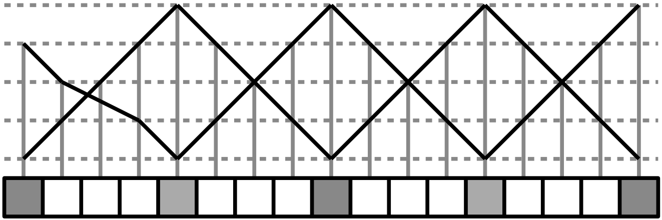
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

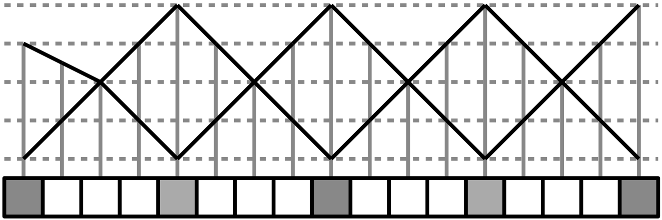
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

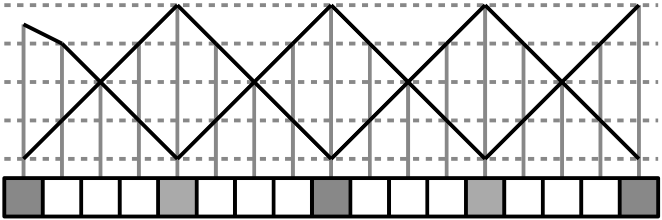
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



The resulting evolution

Signal and Dynamics

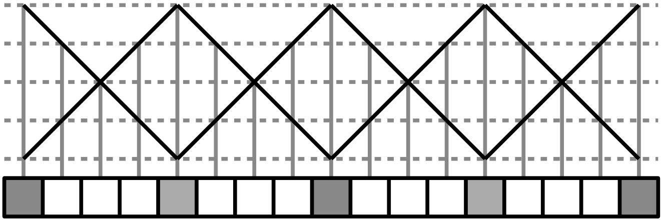
- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



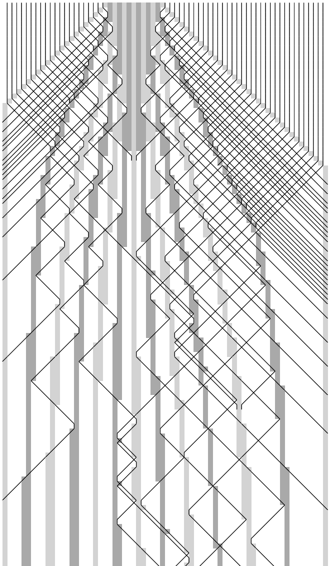
The resulting evolution

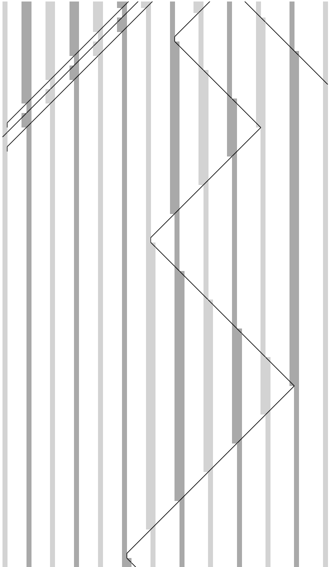
Signal and Dynamics

- We can see that signals travels through the space
- We can assign energy and momentum to these signals
- Defined by fields; composed for global system



Space-time diagram of the uniformisation



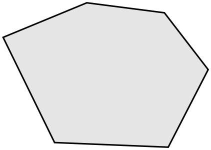


Convex Hulls

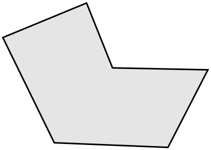
Convexity in Euclidean Space

Definition (Euclidean convex region)

A convex region contains all segments joining two of its points



Convex Polygon

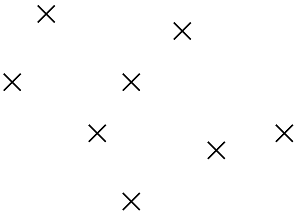


Concave Polygon

Convexity in Euclidean Space (Cont.)

Definition (Convex Hull)

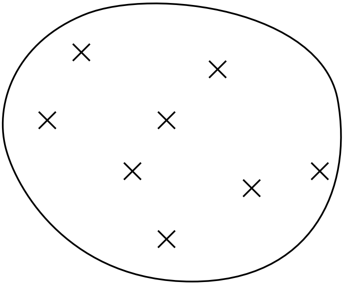
The convex hull is the smallest convex region containing a set



Convexity in Euclidean Space (Cont.)

Definition (Convex Hull)

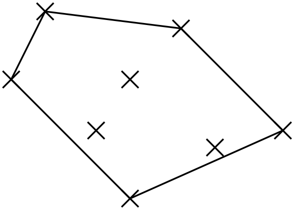
The convex hull is the smallest convex region containing a set



Convexity in Euclidean Space (Cont.)

Definition (Convex Hull)

The convex hull is the smallest convex region containing a set

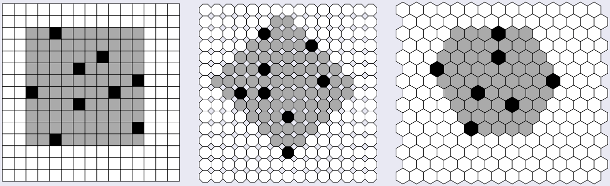


Convexity in Cellular Space

Definition (Metric convex region)

A convex region contain all **shortest paths** joining two of its points

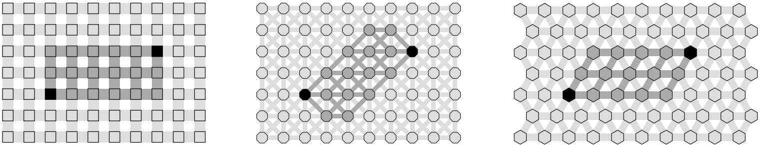
Convexity for Metric Cellular Space



Convexity in Cellular Space (Cont.)

Shortest paths between two points

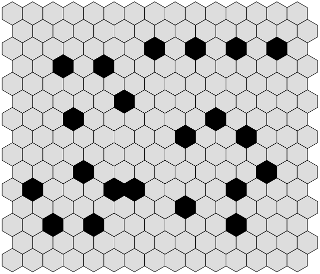
- Many shortest path between two points: Interval
- $[x, y] = \{\, z \in S \mid d(x, z) + d(z, y) = d(x, y) \,\}$



First Step: Local convexity

Definition

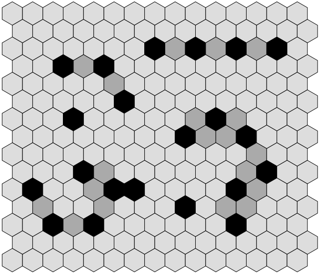
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

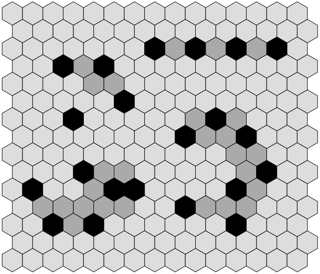
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

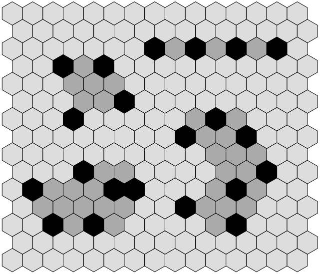
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

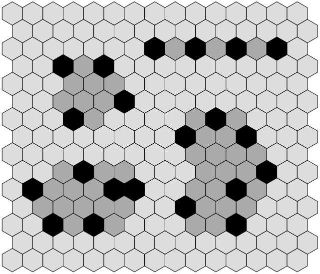
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

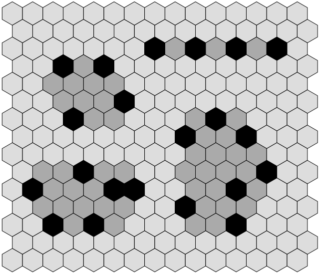
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

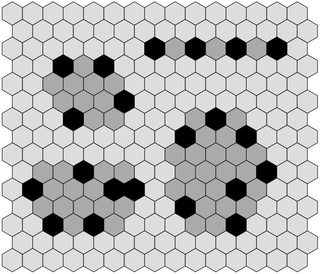
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

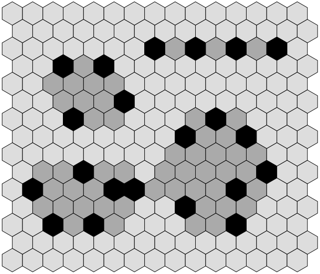
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

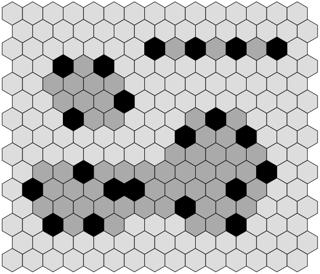
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

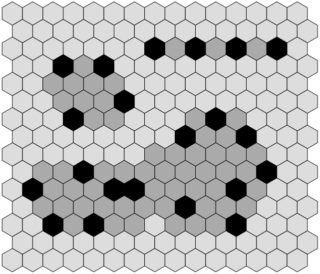
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

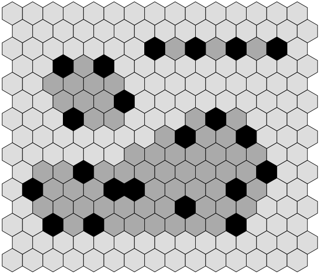
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

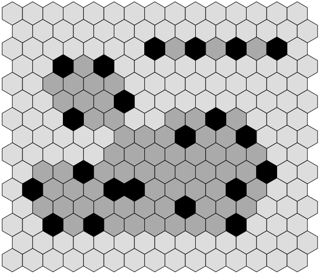
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

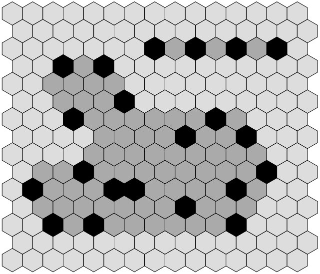
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

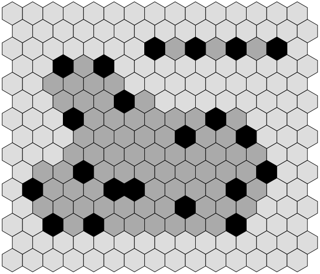
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

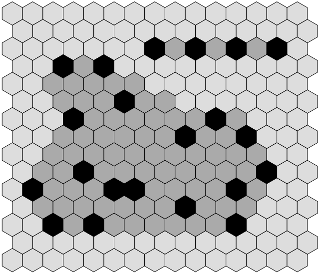
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

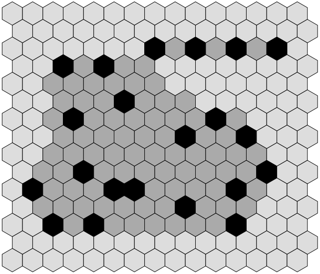
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

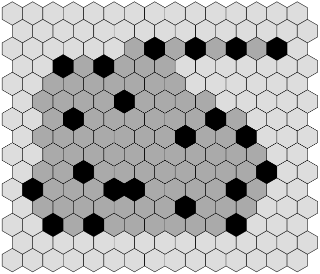
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

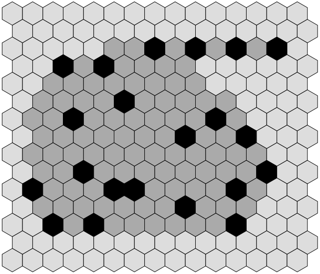
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

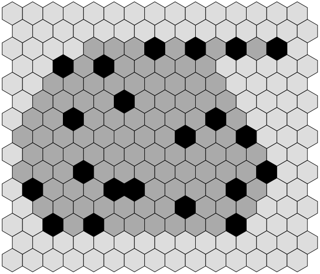
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

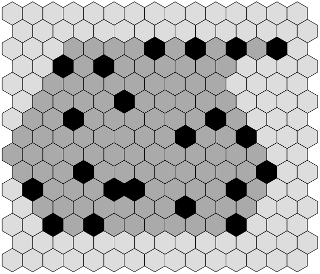
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

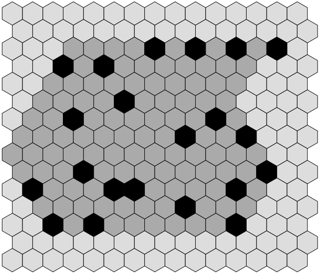
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

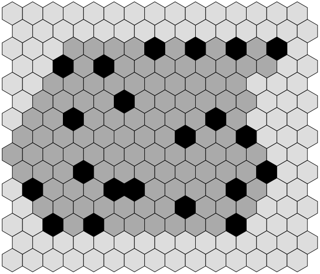
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

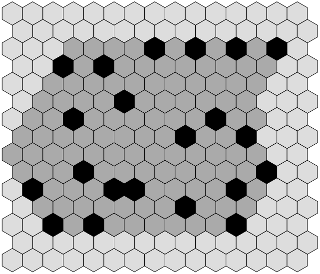
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

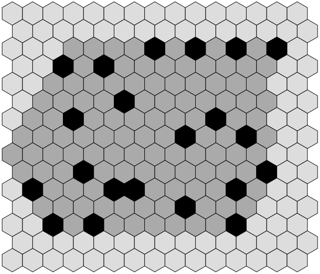
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

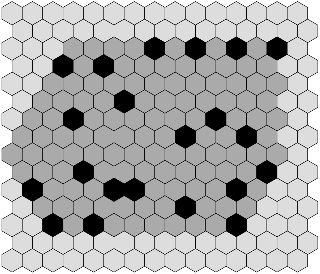
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

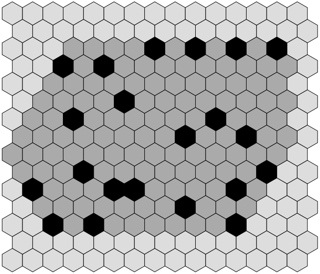
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

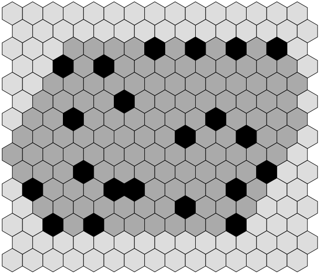
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

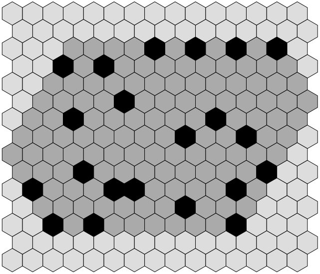
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



First Step: Local convexity

Definition

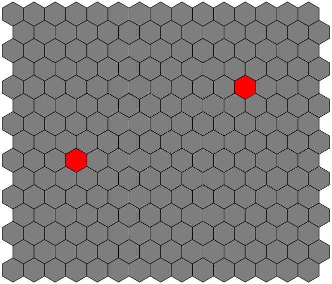
$$conv_t(x) = \exists y_0, y_1 \in \{y \in N(x) \mid y \in P_t \vee conv_{t-1}(y)\}; x \in [y_0, y_1]$$



Second Step: Global Convexity for Two Particles

Required Fields:

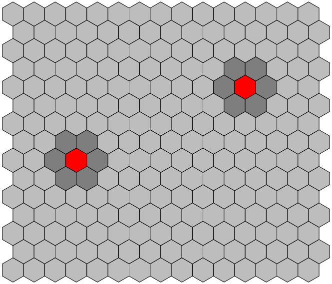
- Grow a distance field modulo 3 (static particles)



Second Step: Global Convexity for Two Particles

Required Fields:

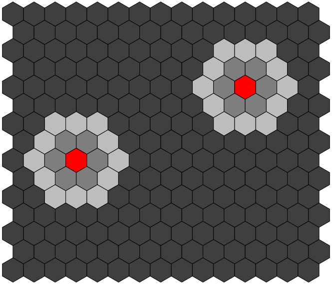
- Grow a distance field modulo 3 (static particles)



Second Step: Global Convexity for Two Particles

Required Fields:

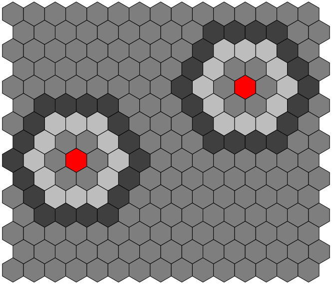
- Grow a distance field modulo 3 (static particles)



Second Step: Global Convexity for Two Particles

Required Fields:

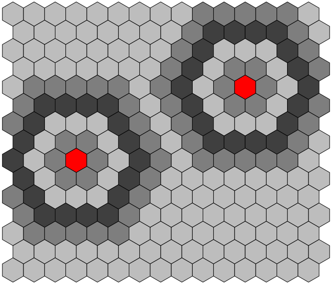
- Grow a distance field modulo 3 (static particles)



Second Step: Global Convexity for Two Particles

Required Fields:

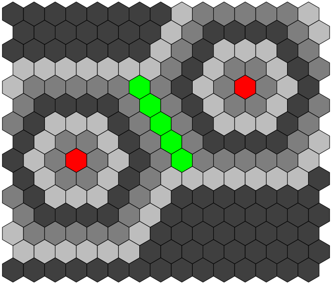
- Grow a distance field modulo 3 (static particles)



Second Step: Global Convexity for Two Particles

Required Fields:

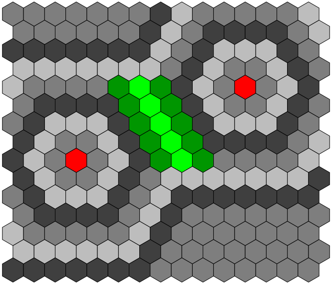
- Grow a distance field modulo 3 (static particles)
- Detect the middles of the shortest paths



Second Step: Global Convexity for Two Particles

Required Fields:

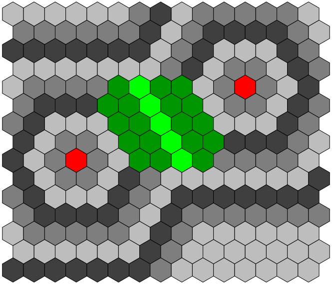
- Grow a distance field modulo 3 (static particles)
- Detect the middles of the shortest paths
- Go back from the middles to the particles



Second Step: Global Convexity for Two Particles

Required Fields:

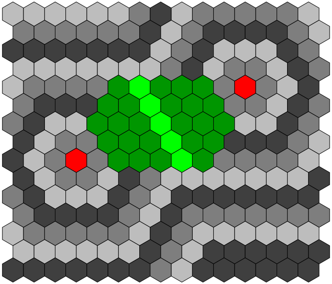
- Grow a distance field modulo 3 (static particles)
- Detect the middles of the shortest paths
- Go back from the middles to the particles



Second Step: Global Convexity for Two Particles

Required Fields:

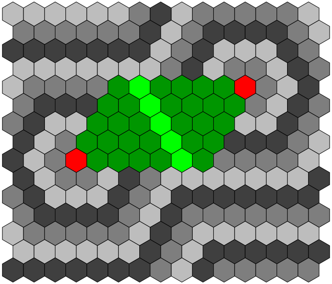
- Grow a distance field modulo 3 (static particles)
- Detect the middles of the shortest paths
- Go back from the middles to the particles



Second Step: Global Convexity for Two Particles

Required Fields:

- Grow a distance field modulo 3 (static particles)
- Detect the middles of the shortest paths
- Go back from the middles to the particles



Last Step: Global Convexity for Many Particles

Convex Hull of Two Points

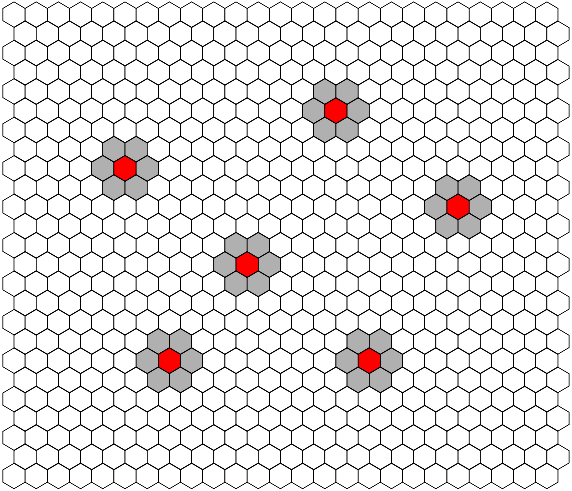
- Grow a distance field modulo 3
- Detect the middles of the shortest paths
- Go back from the middles to the points

Last Step: Global Convexity for Many Particles

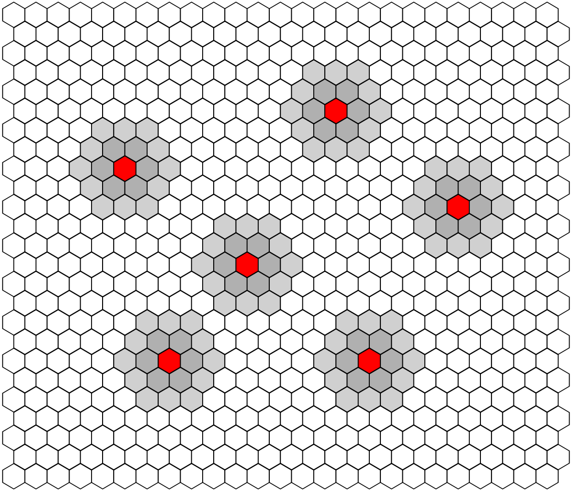
Convex Hull of Many Points

- Grow a distance field modulo 3
- Detect the middles of the shortest paths
- Go back from the middles to the points

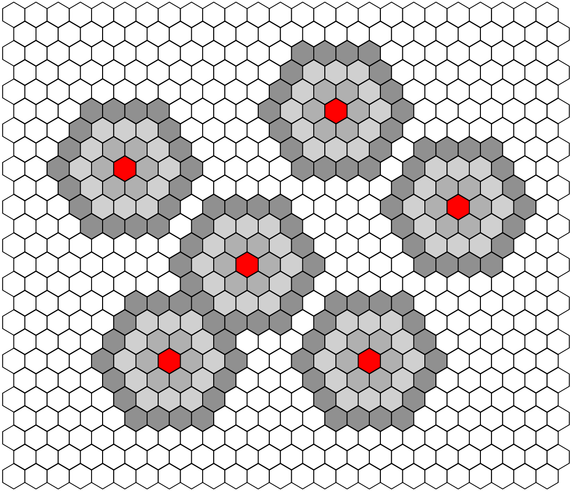
Real Challenge: Global Convexity



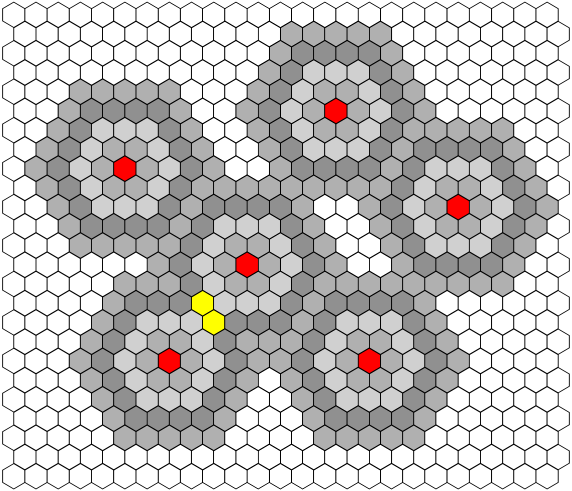
Real Challenge: Global Convexity



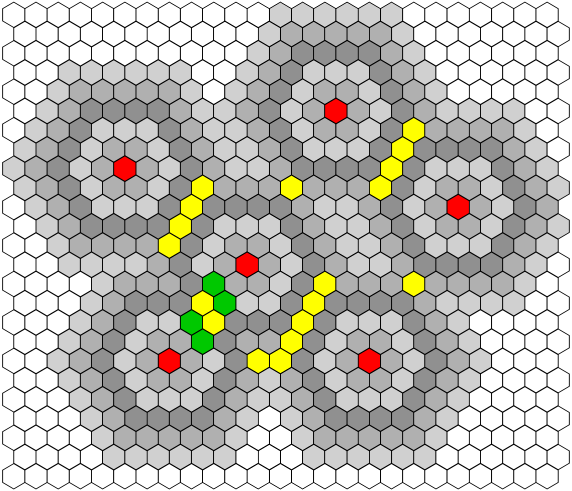
Real Challenge: Global Convexity



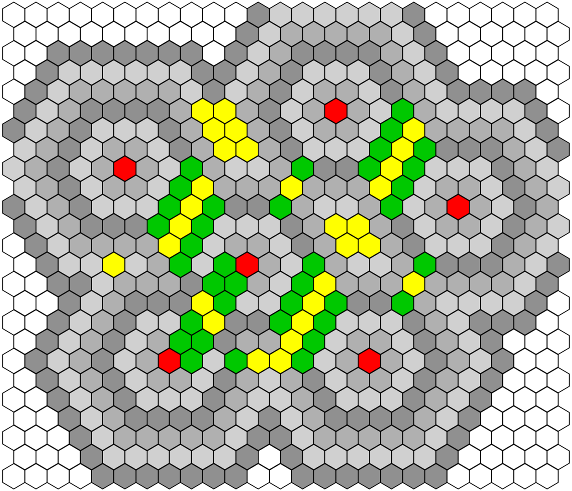
Real Challenge: Global Convexity



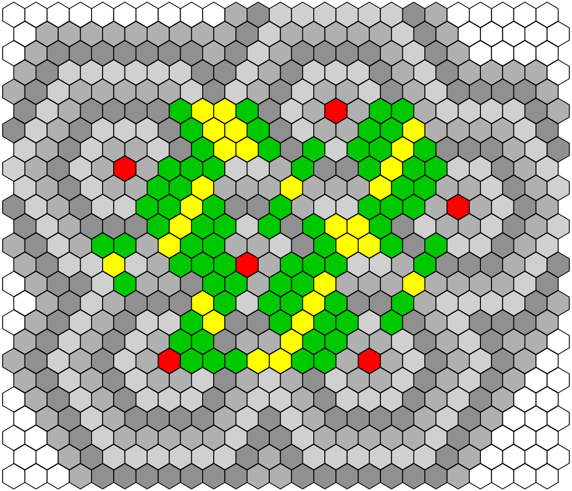
Real Challenge: Global Convexity



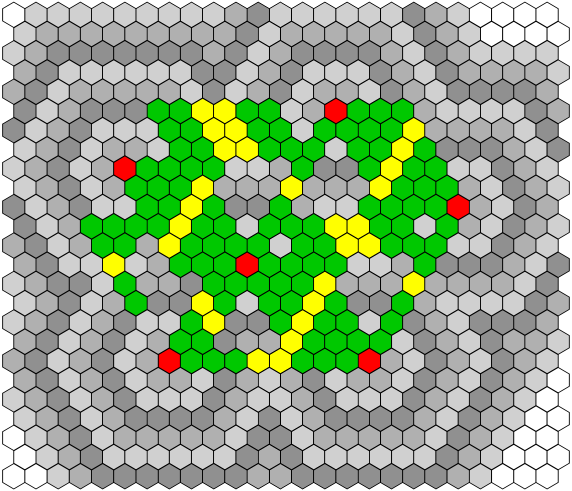
Real Challenge: Global Convexity



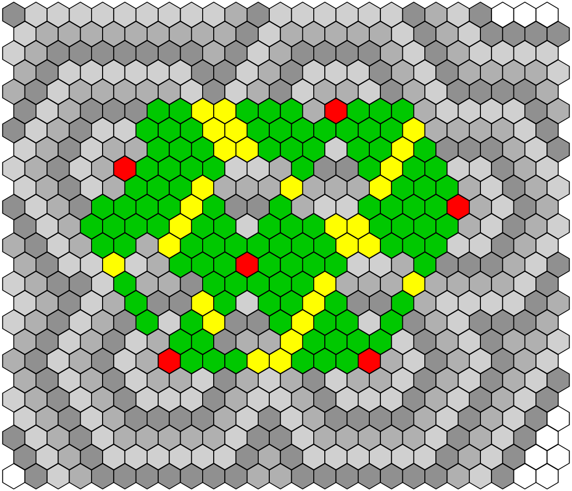
Real Challenge: Global Convexity



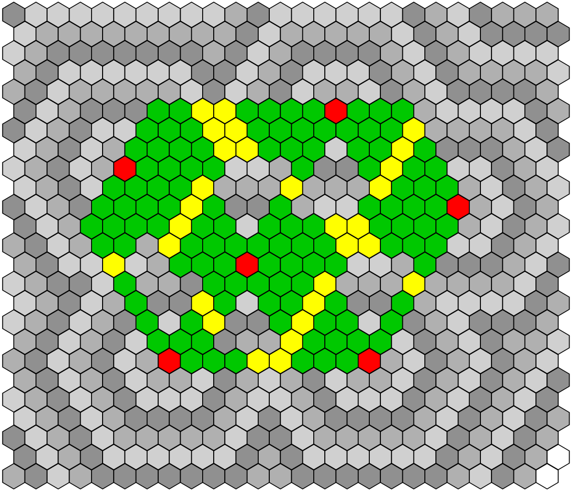
Real Challenge: Global Convexity



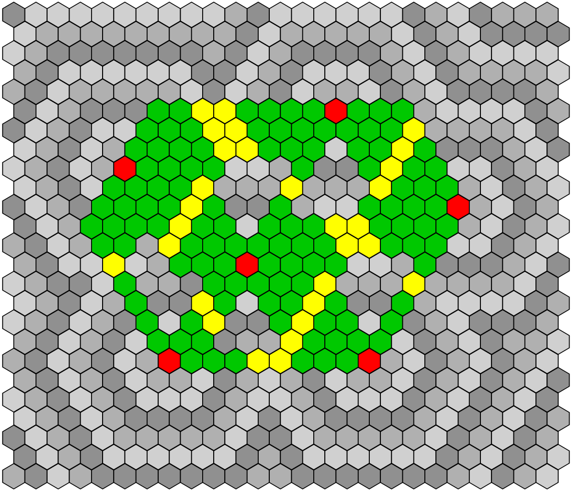
Real Challenge: Global Convexity



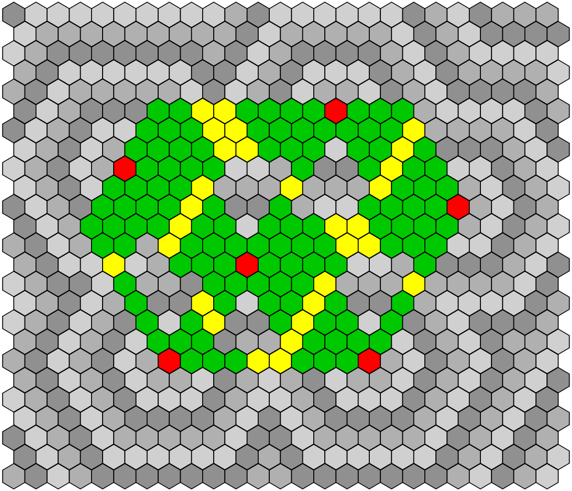
Real Challenge: Global Convexity



Real Challenge: Global Convexity

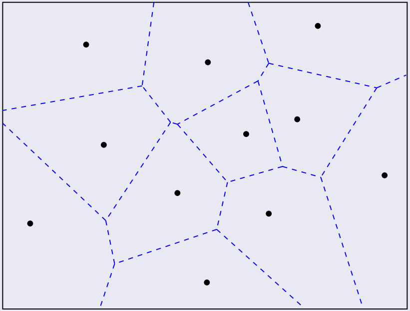


Real Challenge: Global Convexity



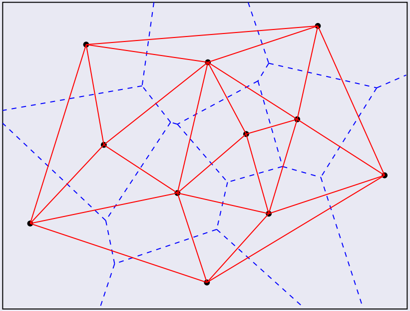
Proximity Graph Characterisation

Distance Field and Voronoi Diagram



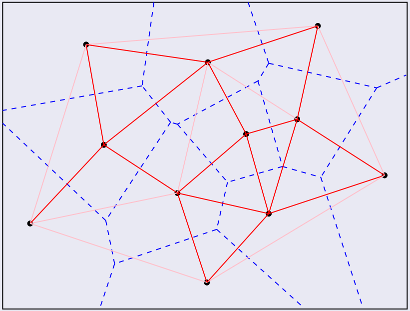
Proximity Graph Characterisation

Delaunay graph ?



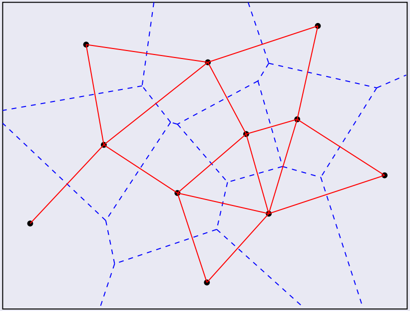
Proximity Graph Characterisation

Delaunay graph ? No, only a subset



Proximity Graph Characterisation

Gabriel Graph !



Gabriel graphs

Original Gabriel graphs

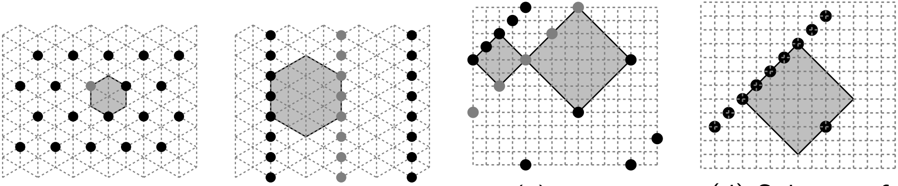
Definition (Gabriel Graph)

- Euclidean spaces
- Connects two particles x and y if and only if the ball using the segment $[xy]$ as diameter does not contain any other particle.

Gabriel graphs on Cellular Spaces

- Connected for Euclidean... and for cellular spaces ?

Gabriel graphs on Cellular Spaces



(a) Each point (b) Each line (c) Each diagonal (d) Subcase of (c)

Generalisation

- **Connectedness is not ensured** in general
- The cause is the non-uniqness of diameters and minimal balls
- We **need to generalize** the definition

Principles of Gabriel Graphs

Connectedness, Minimality, and Locality

- Minimality and connectedness:
  - **minimum spanning trees**
- Locality and connectedness :
  - arbitrarily choice implies global coherence
  - **union of all** minimum spanning trees
- Locality and minimality :
  - Edge decision should be local
  - union of all **local** minimum spanning trees

From Principles to Definition

Definition (Metric Gabriel Graph)

- Any metric space
- Connects two particles x and y if and only if there is a ball $B(c, r)$ such that $d(x, y) = 2r$ and $\{x, y\}$ is an edge of a minimum spanning tree of $P \cap B(c, r)$.

From Principles to Definition (Cont.)

Definition (Metric Gabriel Graph)

- Any metric space
- Connects two particles x and y of P if and only if there is a ball $B(c, r)$ such that there is a cut $\{P_0, P_1\}$ of $P \cap B(c, r)$ with $(x, y) \in P_0 \times P_1$ and $d(P_0, P_1) = 2r$.

From Principles to Definition (Cont.)

Definition (Metric Gabriel Graph)

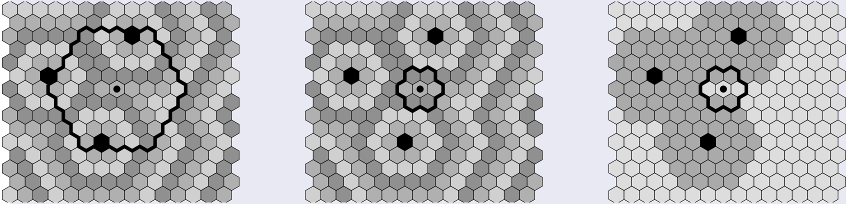
- Any metric space
- Connects two particles x and y of P if and only if there is a ball $B(c, r)$ such that there is a cut $\{P_0, P_1\}$ of $P \cap B(c, r)$ with $(x, y) \in P_0 \times P_1$ and $d(P_0, P_1) = 2r$.

Preservation of the Properties

- Metric Gabriel graphs are always **connected**
- On **Euclidean spaces**, they are **Gabriel graphs**

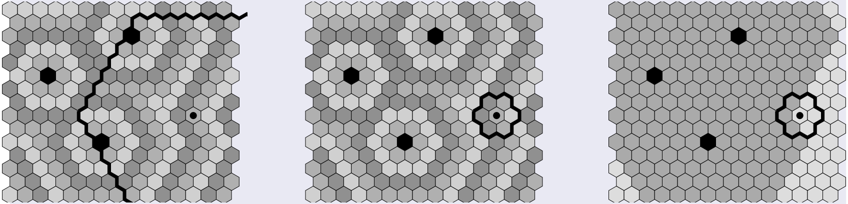
Distance fields and dilations

Example of a metric Gabriel ball center

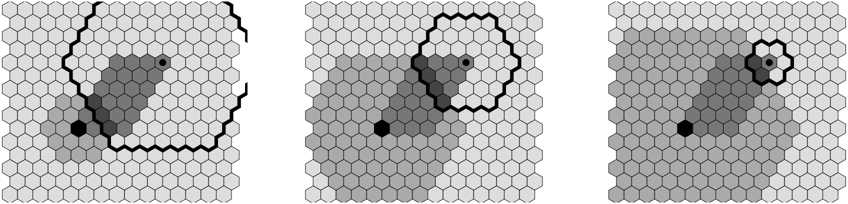


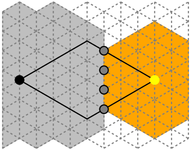
Distance fields and dilations (Cont.)

Example of a non-metric Gabriel ball center

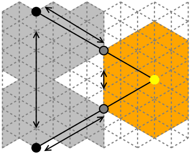


Dilations and interval slices

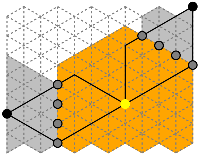




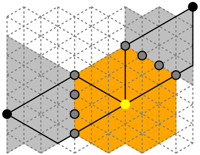
One particle



Two particles



$r = 4$



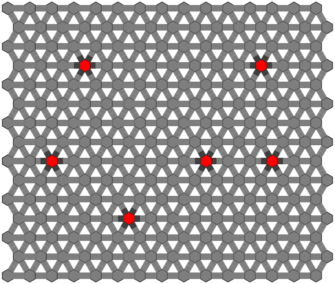
$r = 3$

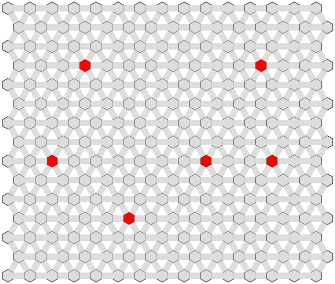
The metric Gabriel ball centers field

$$\mathrm{cent}_t(x) = \begin{cases} \bot \text{ if } t = 0 \\ \top \text{ if } \overline{\mathrm{cent}}_t(x,x) \\ \top \text{ if } \exists y \in N(x), \overline{\mathrm{cent}}_t(x,y) \\ \bot \text{ otherwise;} \end{cases}$$

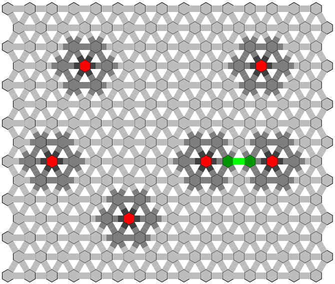
$$\begin{aligned} \overline{\mathrm{cent}}_t(x,y) &= |\overline{Q}_t(x,y)/C^+_{2r_{xy}}| \geq 2 \\ \overline{Q}_t(x,y) &= \{\, z \in B(xy, r_{xy}) \mid D[P]_{t-1}(z) + r_{xy} = \overline{D}[P]_{t-1}(x,y) \,\} \end{aligned}$$

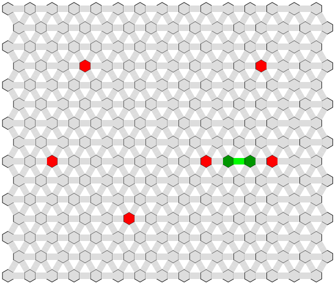
The resulting cellular automaton



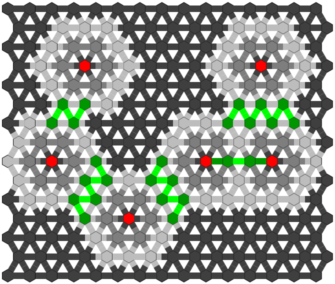


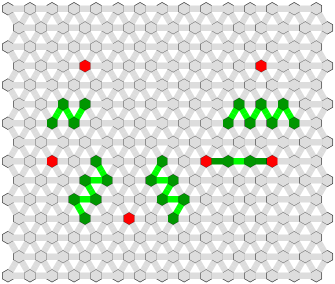
The resulting cellular automaton



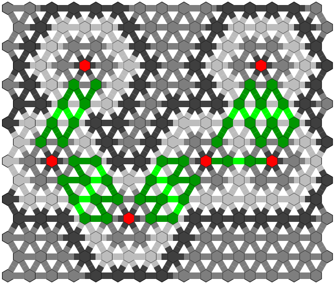


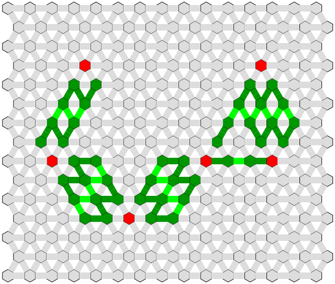
The resulting cellular automaton



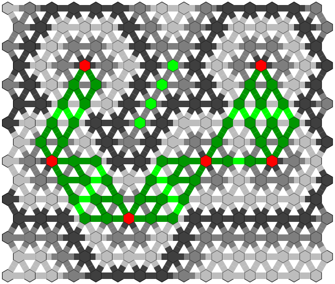


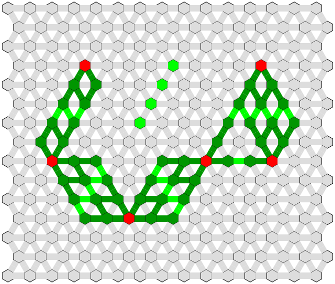
The resulting cellular automaton



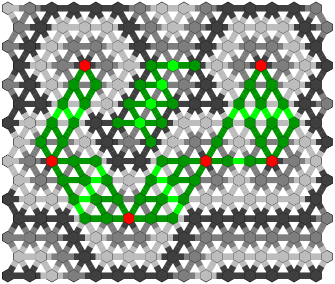


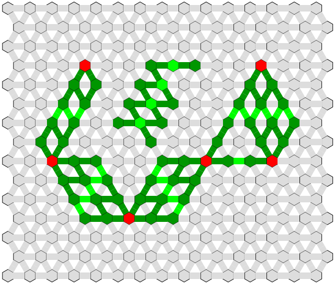
The resulting cellular automaton



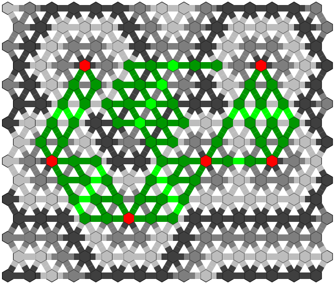


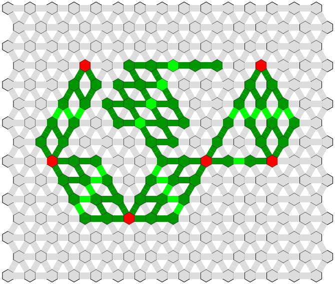
The resulting cellular automaton



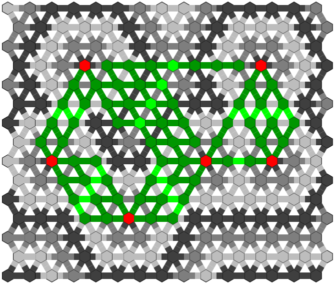


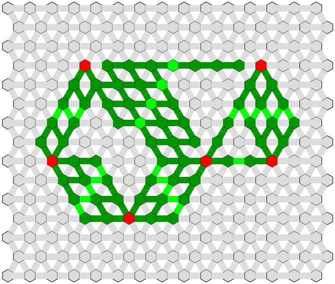
The resulting cellular automaton



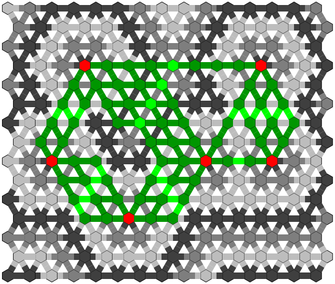


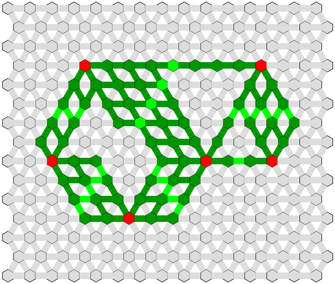
The resulting cellular automaton





The resulting cellular automaton





Conclusion and Perpectives

Perspectives

In the same framework

- Voronoi Diagram Field
- Firing Squad Synchronisation Problem

Extending the framework

- Cayley Graphs
- Asynchronicity
- Amorphous Computers