Dynamics, Emergent Computation, and Evolution in Cellular Automata

Wim Hordijk

Santa Fe Institute

Emergent Computation in Nature

- Global patterns out of local interactions
- Patterns give rise to global information processing
- Examples:
  1. Spiral waves in aggregating amoeba
  2. Foraging and nest building in social insects
  3. Synchronized oscillations in the brain

➔Emergent computation in decentralized spatially extended systems

Two Important Questions about Emergent Computation

1. How do the dynamics (i.e., the pattern forming behavior) of these decentralized spatially extended systems give rise to emergent computation?
2. How is this capability for emergent computation produced by evolution?

Research Approach

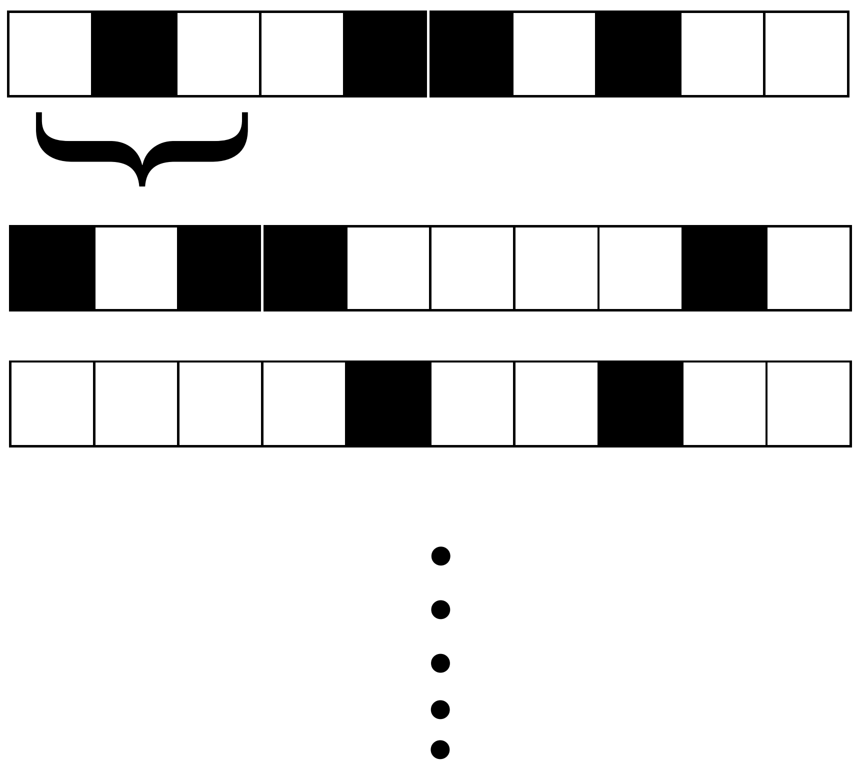
1. Use cellular automata (CAs) as a simple class of decentralized spatially extended systems.
2. Use a genetic algorithm (GA) as a simple model of evolution.
3. Use the GA to evolve CAs to perform computational tasks requiring global information processing.
4. Study the emergent computation that evolves in these CAs.

➔ A study of the relation among dynamics, emergent computation, and evolution in cellular automata.

Cellular Automata (CAs)

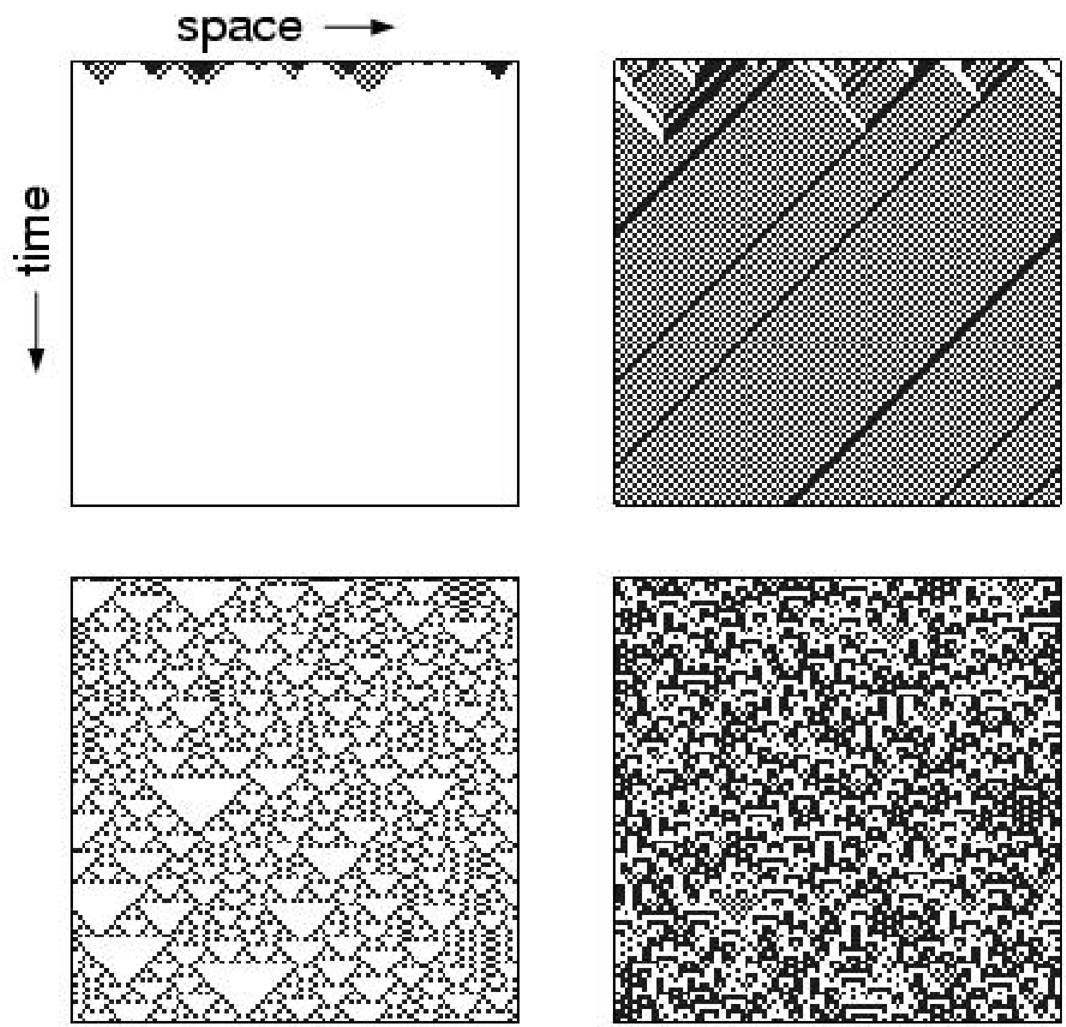
- Regular lattice of simple finite automata
- Local interactions between cells
- No central control
- Synchronous updates→discrete dynamical system
- Large variety in CA behavior, from fixed point or periodic to completely random or even chaotic; also “emergent” behavior

CA Example

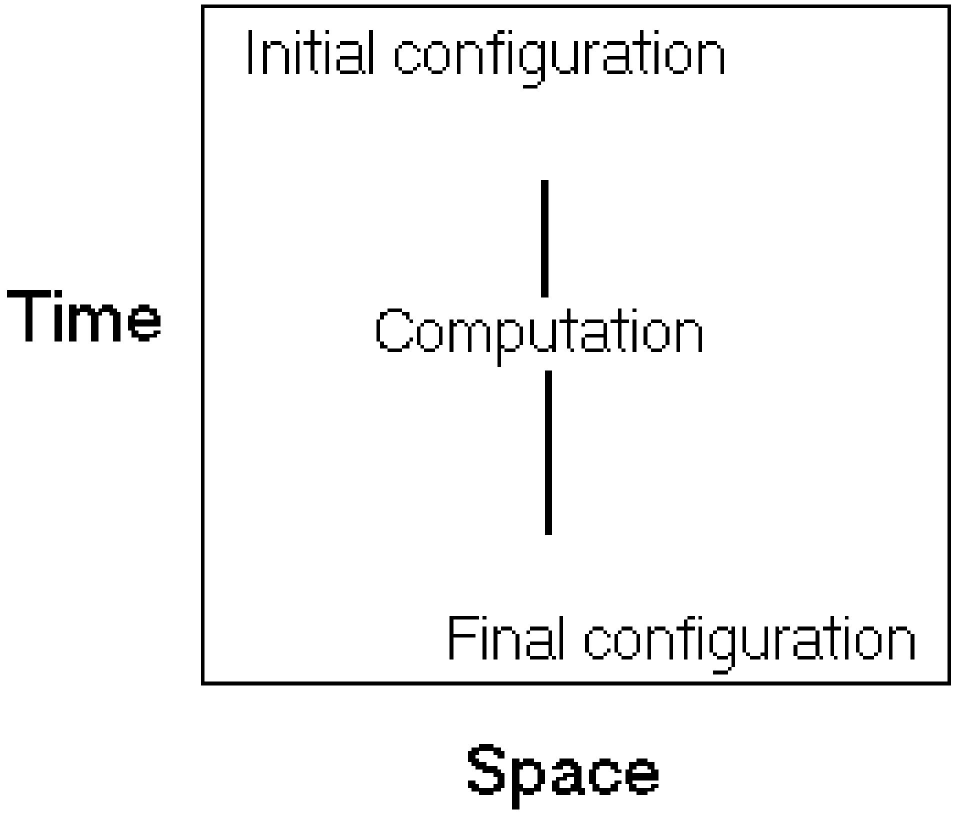


000 → 0
001 → 1
010 → 0
011 → 0
100 → 1
101 → 0
110 → 0
111 → 0

CA Dynamics



Computation in CAs



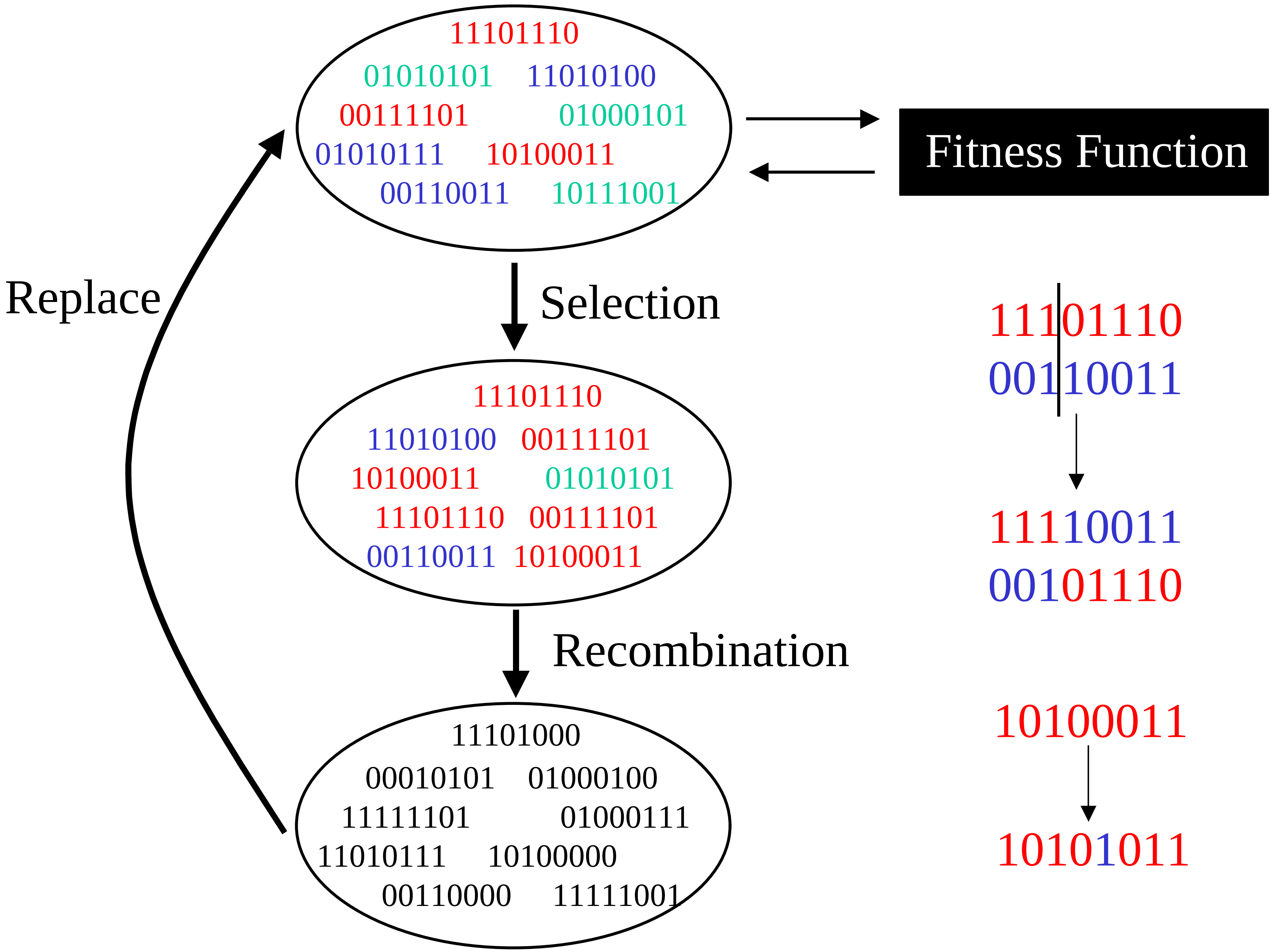
Density Classification

- Decide whether the initial configuration contains more black cells or more white cells.
- If more white cells: settle down to an all-white configuration.
- If more black cells: settle down to an all-black configuration.

➔Does there exist a CA that can perform this task??

Genetic Algorithms (GAs)

- Stochastic search algorithm based on natural evolution
- Translate the problem in a fitness function, encode possible solutions as “genotypes”
- Maintain population of candidate solutions
- Create offspring populations by means of selection, recombination, and mutation
- Let a solution “evolve”


11101110
01010101 11010100
00111101 01000101
01010111 10100011
00110011 10111001
Fitness Function
Selection
Replace
11101110
11010100 00111101
10100011 01010101
11101110 00111101
00110011 10100011
Recombination
11101000
00010101 01000100
11111101 01000111
11010111 10100000
00110000 11111001
11101110
00110011
11110011
00101110
10100011
10101011

Evolving CAs with GAs

11101000

00010101 01000100

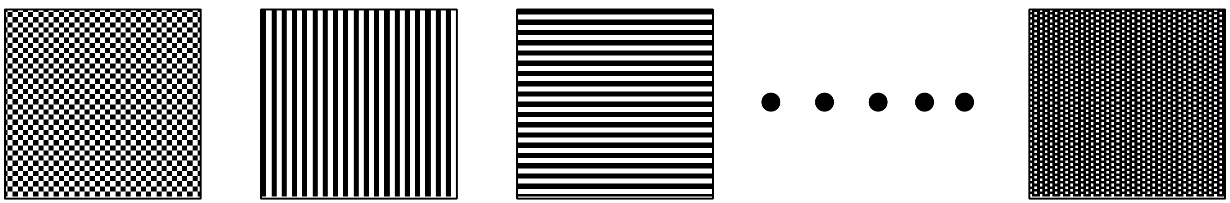
11111101 01000111

11010111 10100000

00110000 11111001

11101000 →

000	**1**
001	**1**
010	**1**
011	**0**
100	**1**
101	**0**
110	**0**
111	**0**



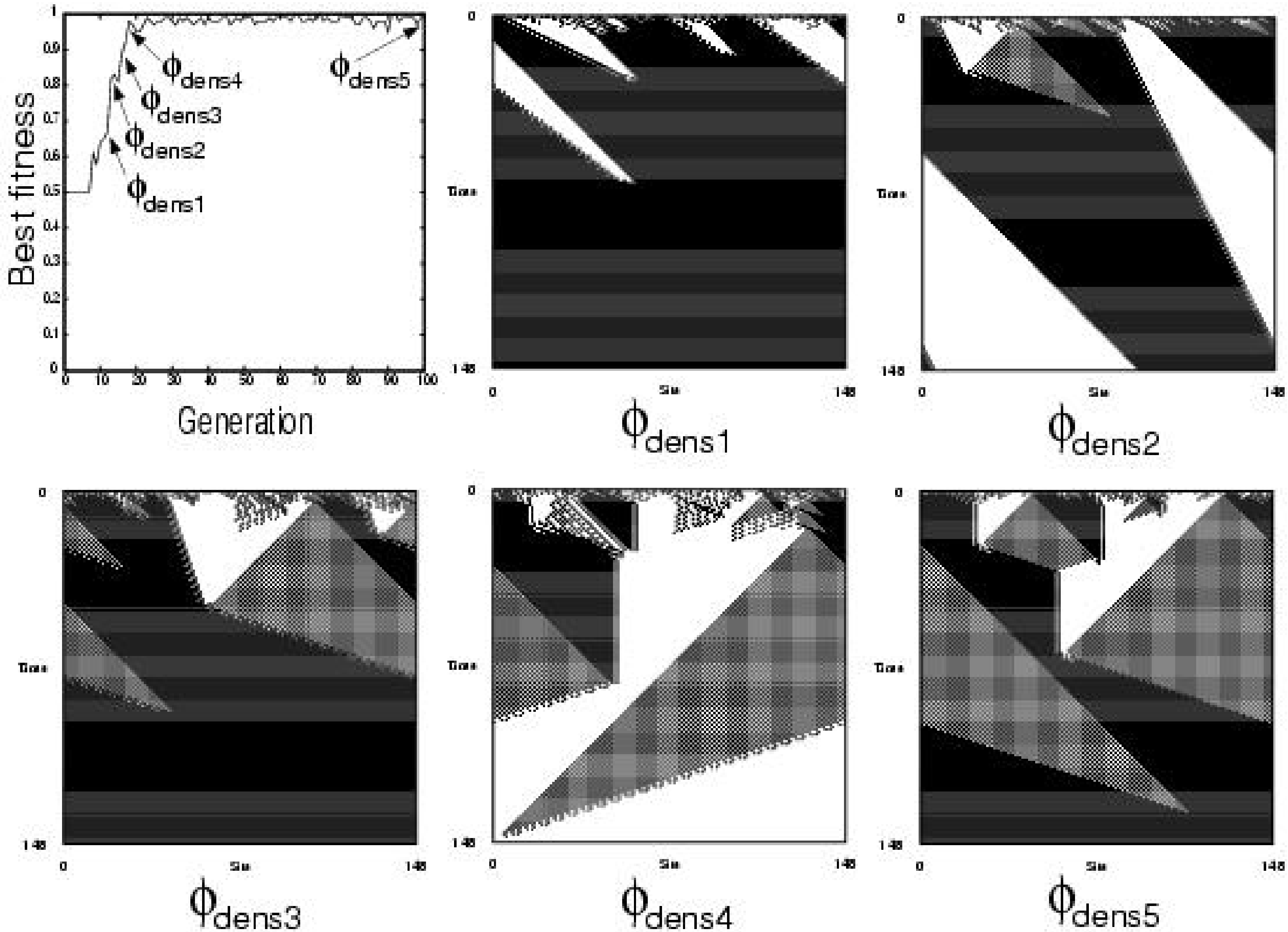
1 0 1 1 → Fitness = Ó

Some Numbers…

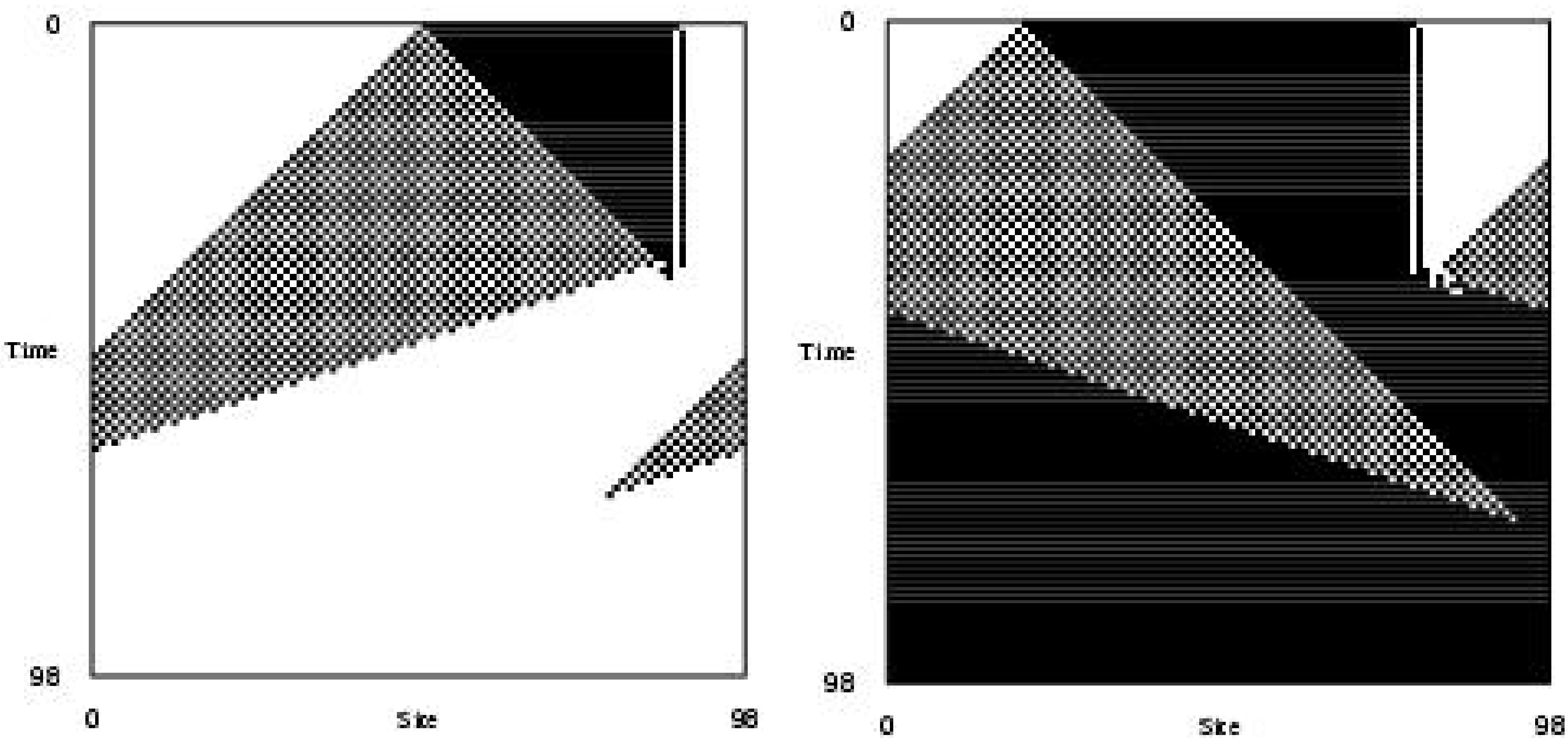
- CA radius: 3
- Local neighborhood size: $2\times 3+1=7$
- CA lookup table size: $2^7 = 128$
- Number of possible CA rules: $2^{128} \approx 3.4\times 10^{38}$

- GA population size: 100
- Number of test cases for fitness: 100
- Number of generations: 100

Results for Density Classification



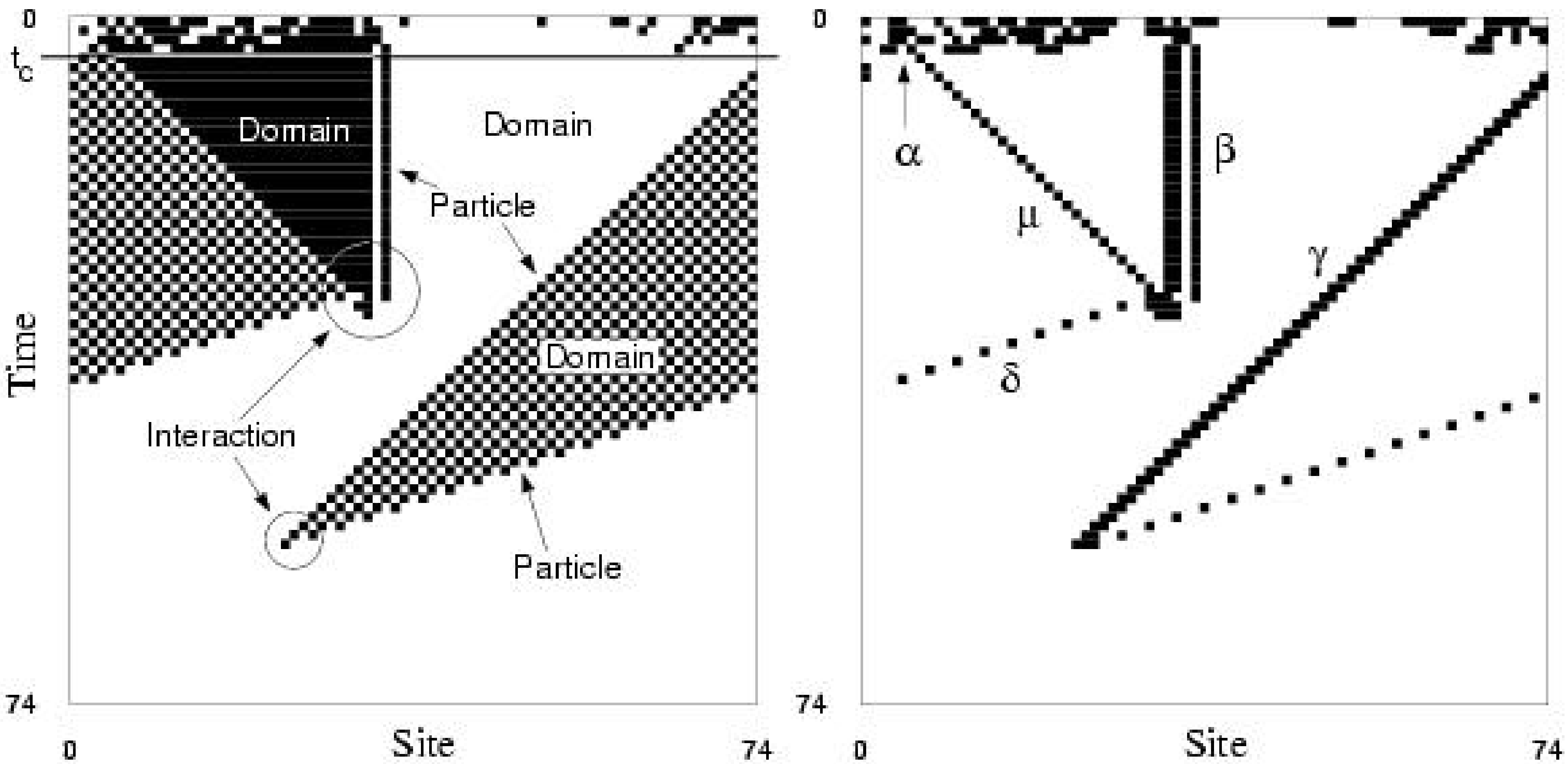
Emergent Computation



Computational Mechanics

- Identify and classify space-time patterns using formal languages
  - Regular domains
  - Particles
  - Particle interactions
- Use domain specifications to filter out regularities from space-time behavior
- Collect info about particles and their interactions in a "particle catalog"

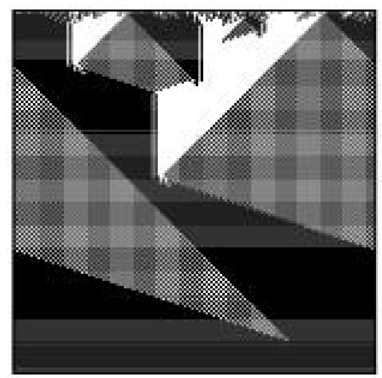
CM—An Example



The Particle Catalog

Domains		
White (**W**)		0*
Black (**B**)		1*
Checkerboard (**#**)		(01)*
Particles		
a	**WB**	**0**
b	**BW**	**0**
c	**W#**	**-1**
d	**#W**	**-3**
e	**#B**	**1**
f	**B#**	**3**

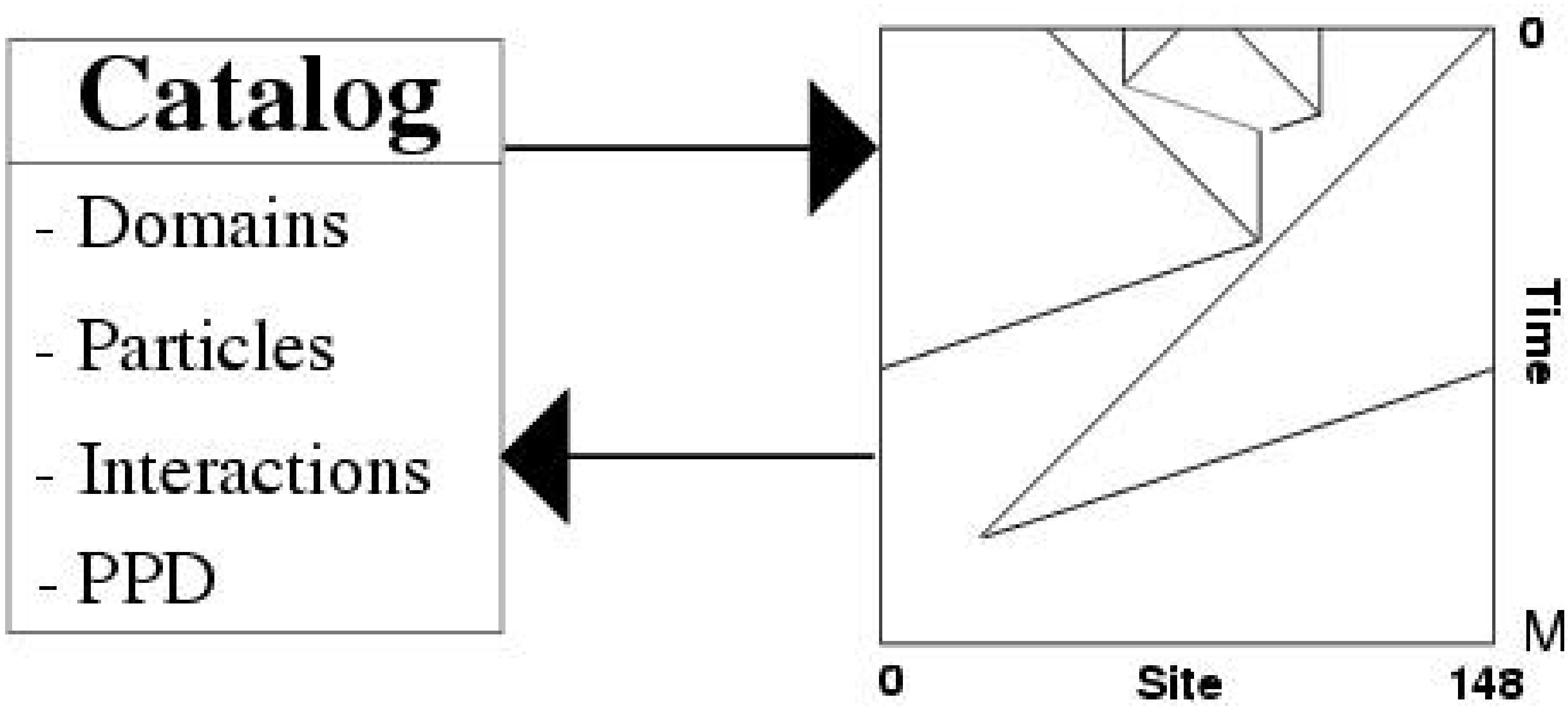
Interactions
a → c + e
b + c → f
e + b → d
c + d → Ø
f + d → b
f + e → Ø



The Particle Model

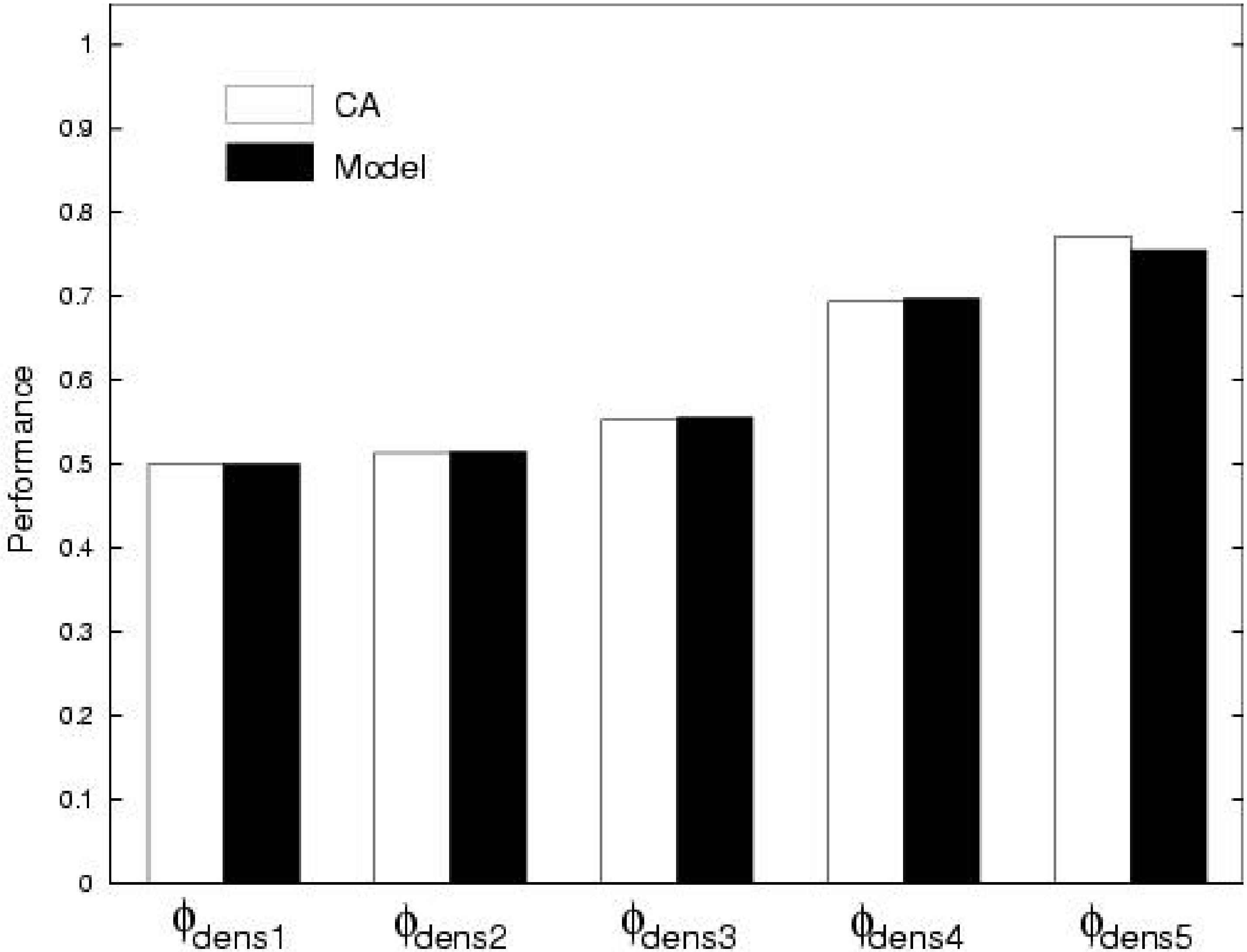
- Start with random particle configuration i.o. random black/white cells
- Calculate next interaction time ***t***
- "Fast forward" to ***t*** and replace interacting particles with their interaction result
- Repeat until no more particles or maximum number of time steps reached
- See if the correct answer state is reached

The Particle Model



Claim: This particle model captures the main mechanisms of the emergent computation in the evolved CAs

Model Performance



Some Conclusions

- The emergent dynamics of evolved CAs can be accurately captured by the particle model
- This model leads to accurate predictions of a CA's behavior and its performance
- Changes in a CA's emergent dynamics can be directly related to changes in its performance
- These changes can be directly related to the evolutionary history of the evolved CAs

Useful Links

- www.santafe.edu/~wim
- www.santafe.edu/projects/evca
- www.santafe.edu/projects/CompMech