PART II

TECHNISCHE UNIVERSITÄT DARMSTADT

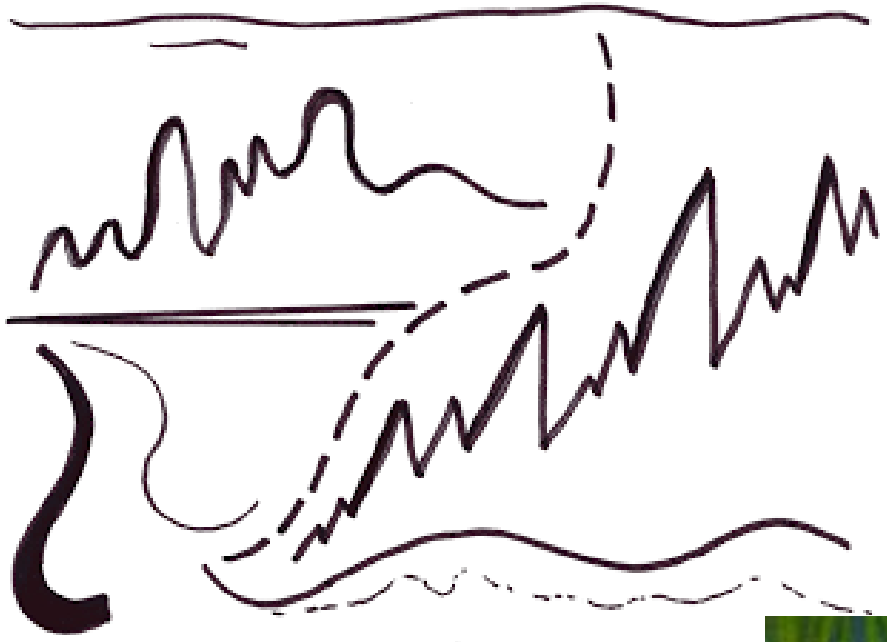
Checkerboard Pattern Formed by Cellular Automata Agents

Rolf Hoffmann

presented at Summer Solstice Int. Conference on Discrete Models of Complex Systems, Catania Italy, June 21-23, 2017

Outline

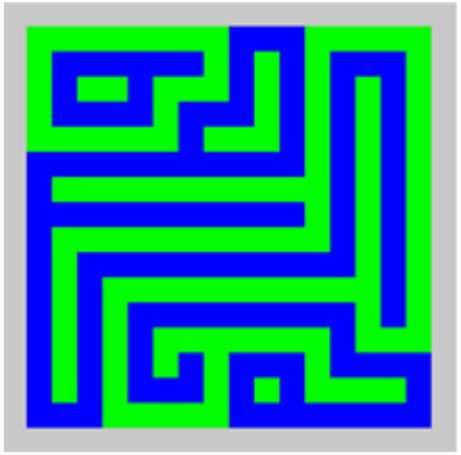
1. Introduction
2. Checkerboard Pattern, Properties
3. Modeling the Multi-Agent-System
4. Designed Algorithms
5. Evolved Algorithms (by a Genetic Algorithm)
6. Conclusion & Future Work



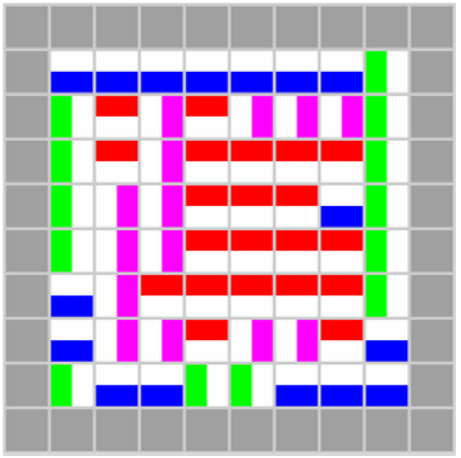
1. Introduction

Patterns formed by agents (former work)

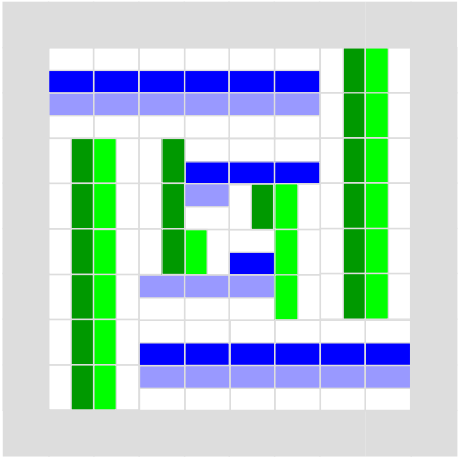
Path Pattern



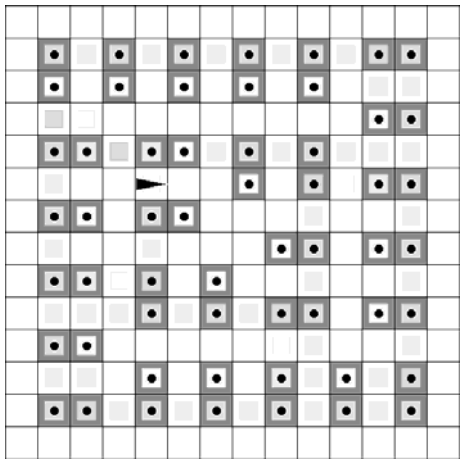
String Pattern



Line Pattern



Domino Pattern

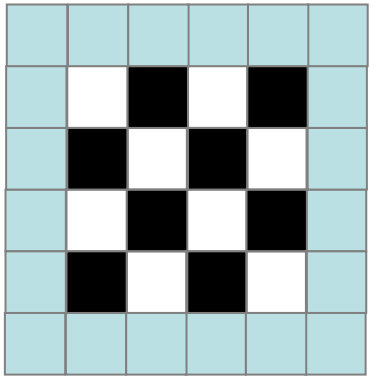


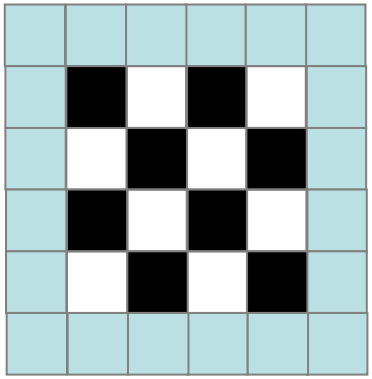
1. Introduction
Why pattern formation by agents?

- Applications
  - forming structures (physical, biological, chemical, mechanical, computational) by robots, nano-robots, laser-beams
- Checkerboard (CB) pattern
  - simple enough to study the problem not only by simulation but also theoretically

2. Checkerboard Pattern, Properties

4 x 4

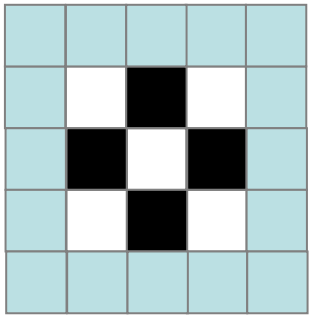




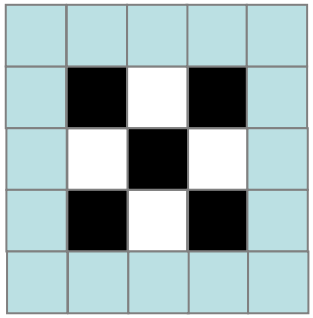
n = 4 (even)

#black = #white = $n^2/2$

3 x 3



4 black, 5 white



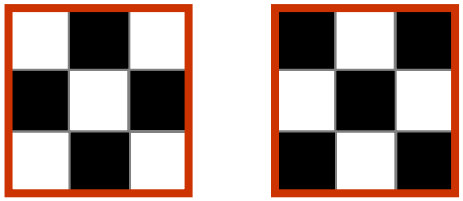
5 black, 4 white

n = 3 (odd)

2. Checkerboard Pattern, Properties

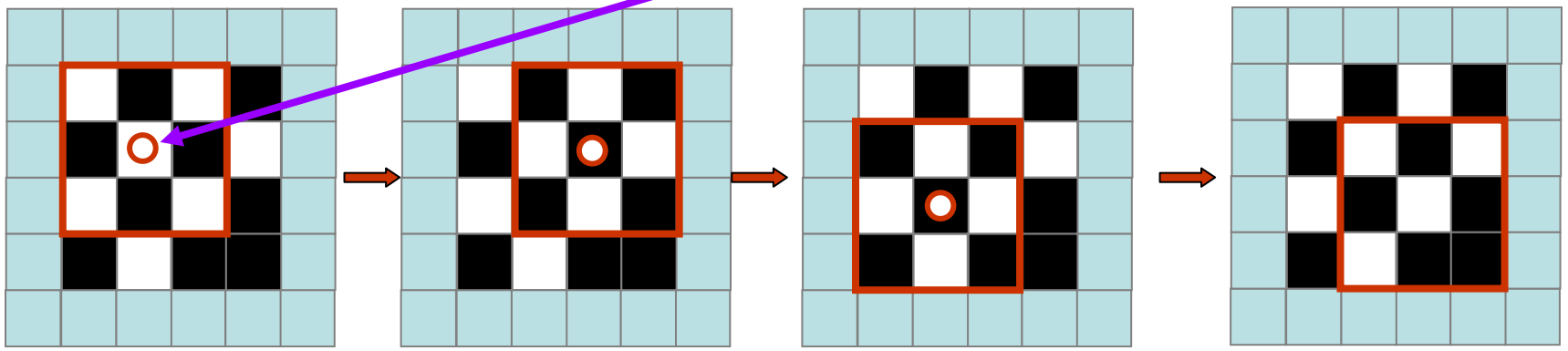
Pattern class defined by *templates* (3 x 3 matching patterns)

Templates:



How to evaluate a given pattern?
Does it belong to the pattern class?

→ Templates are tested at each site (x, y), does any of them matches? If yes, mark a HIT

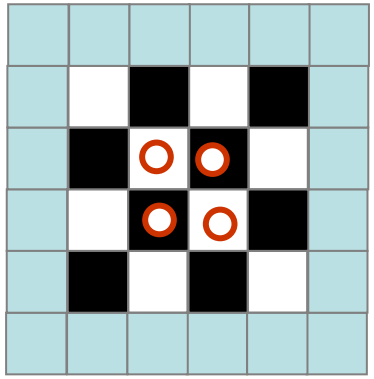


HIT ***HIT*** ***HIT*** ***NO HIT***

2. Checkerboard Pattern, Properties

Degree of order

h = degree of order = sum of all hits

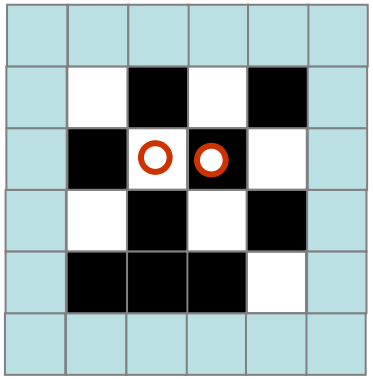


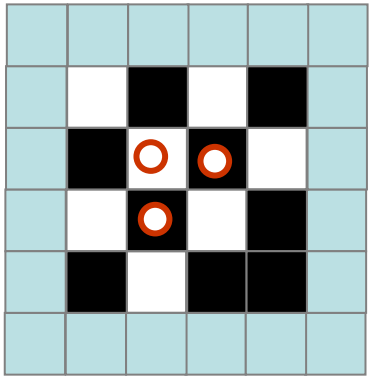
perfect pattern

$h_{max} = (n - 2)^2$

h = 4

non-perfect patterns





h = 2 ***h = 3***

3. Modeling

The problem: forming CB pattern by agents

GIVEN

n x n cell field with borders

color: L ∈ {0, 1, -1} ≡ {white, black, border color}

k moving agents, can change the colors

Initial configuration:

all cells are colored **white**

initial placement of the agents: **dedicated or random**

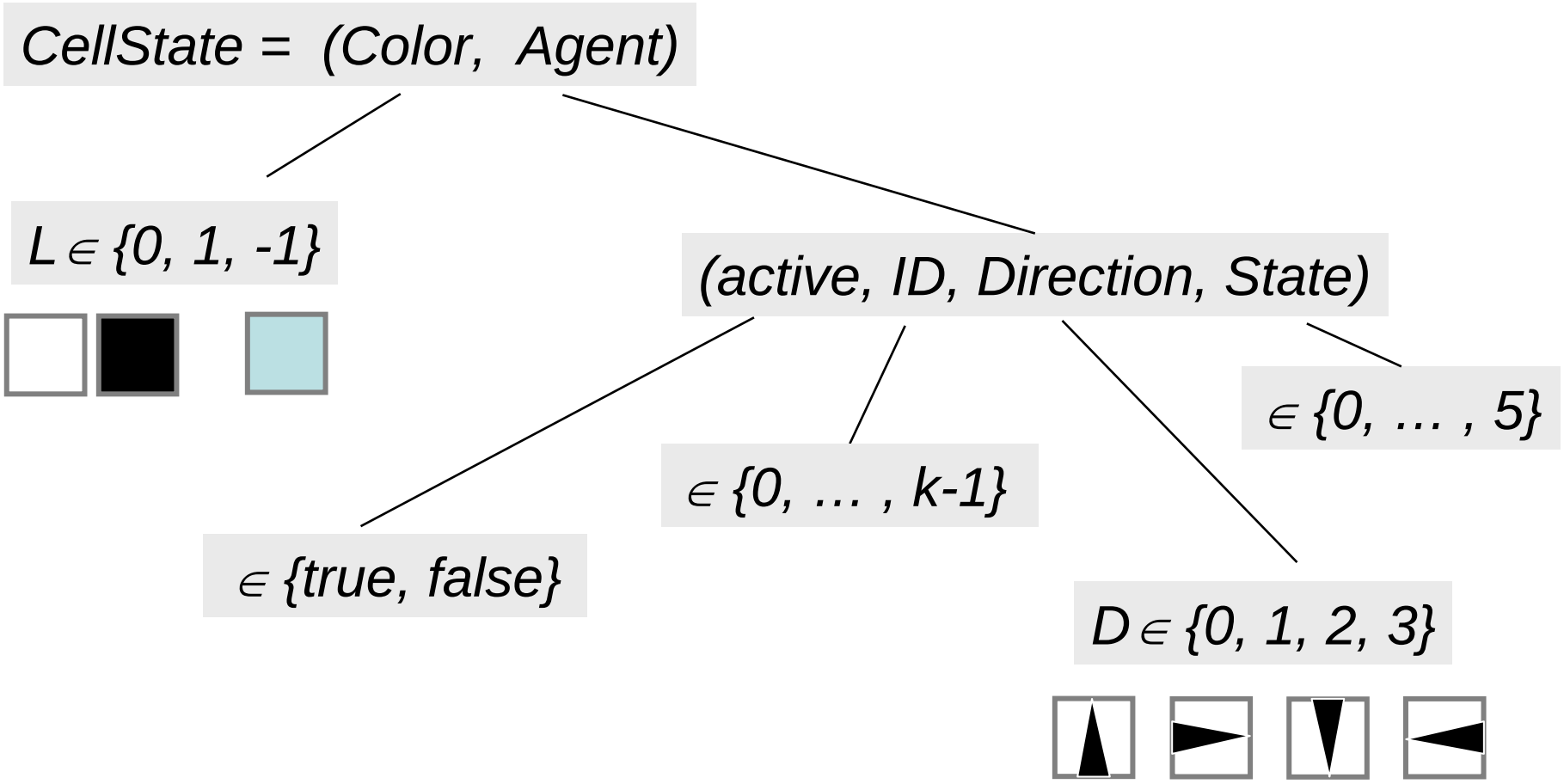
FIND AN AGENT ALGORITHM

such that the multi-agent system forms successfully a CB pattern

degree of order $\mathbf{h = h_{max}}$

Minimize the number of time steps t → min

3. Modeling
Cell state



3. Modeling
Agent‘s actions

- **all actions are executed simultaneously**

- **next control state**

  $state \leftarrow nextstate \in \{0, 1, \ldots, 5\}$

- **move**

  $move \in \{0, 1\}$, *wait or go*

- **turn**

  $new\ direction\ D(t+1) = (D(t) + turn)\ mod\ 4,\ \ turn \in \{0, 1, 2, 3\}$

- **flip color L**

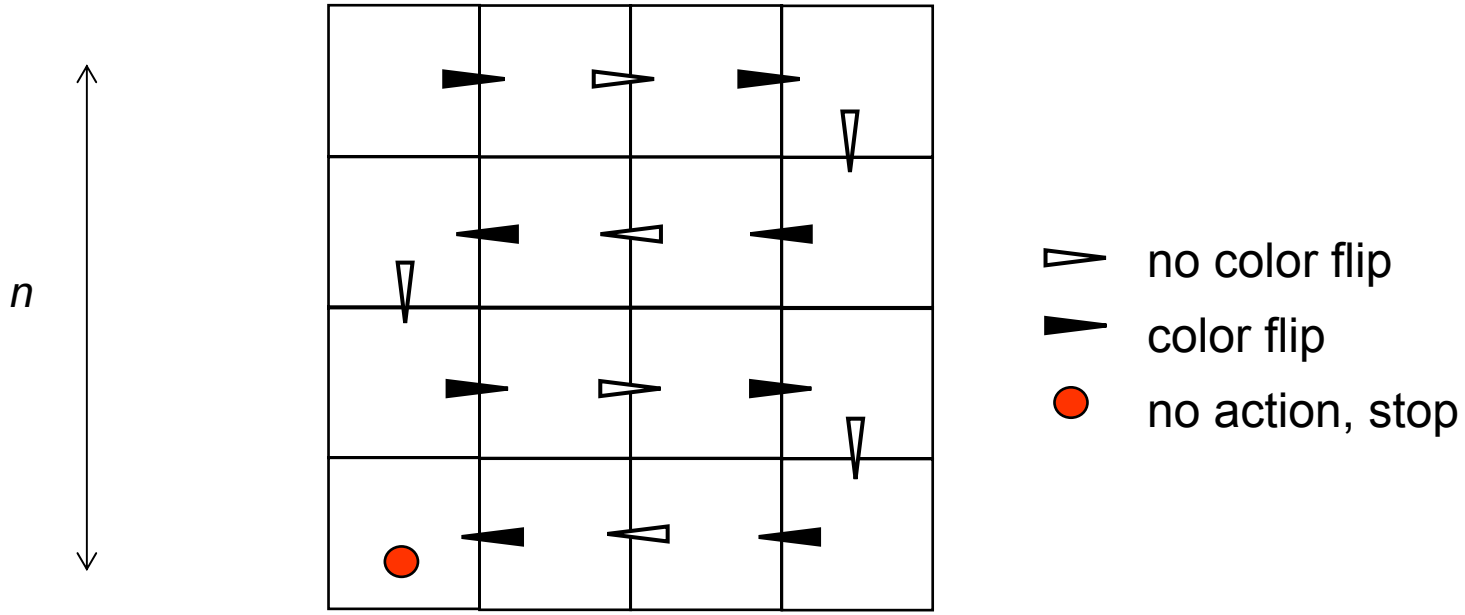
  $L(t+1) = (L(t) + flipcolor)\ mod\ 2,\ \ flipcolor \in \{0, 1\}$

3. Modeling
Agent‘s inputs

- own **control state**
- own **direction**
- **color C**
- **color in front C_F (including border detection)**
- **blocked by agent**
  - another agent in front,
  - by conflict (another prior agent can move to the front cell)

4. Designed Algorithm
with 6 states

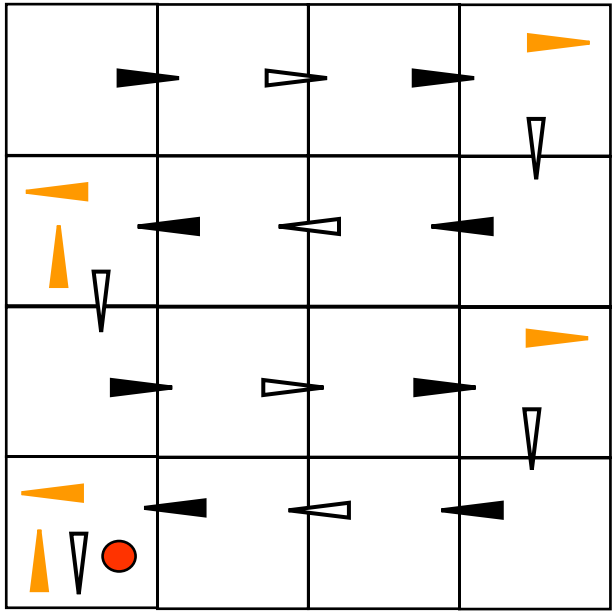
Special case: Agent starts in a corner

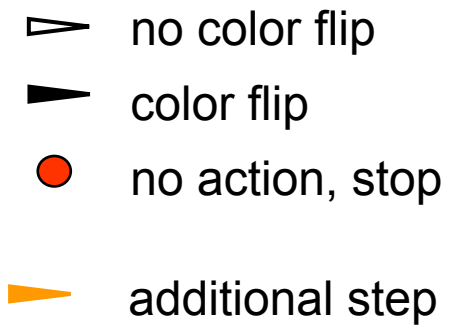


Algorithm with 6 states can be found

$$t = n^2 = N \quad \textit{(time steps)}$$

4. Designed Algorithm
with 4 states

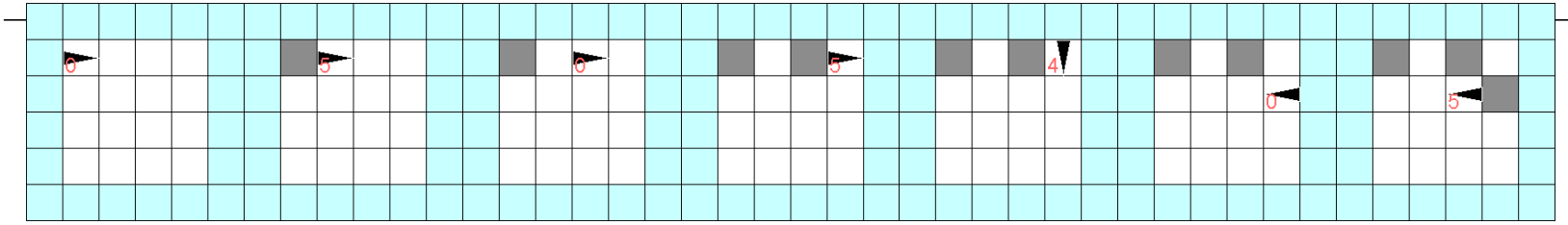




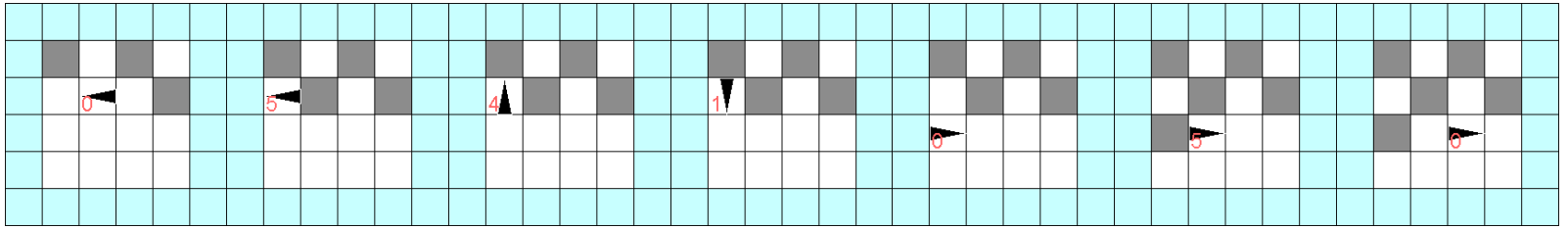
$t = n^2 + n/2 + 2n/2 = O(n^2)$

$t = O(N)$

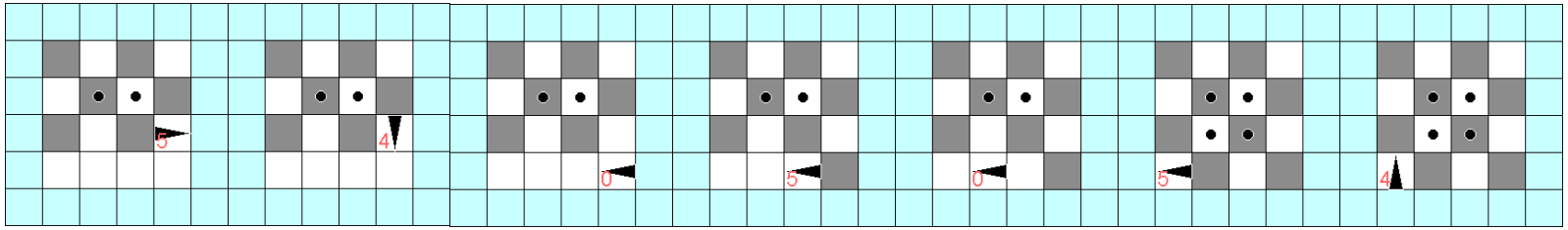
4. Designed Algorithm
with 4 states, simulation



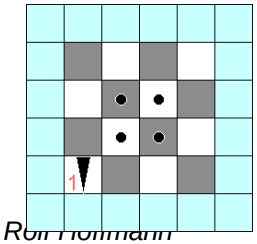
t,state=0,0 1,5 2,0 3,5 4,4 5,0 6,5



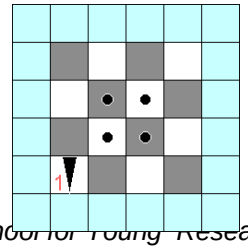
t,state=7,0 8,5 9,4 10,1 11,0 12,5 13,0



t,state=14,5 15,4 16,0 17,5 18,0 19,5 20,4



21,1

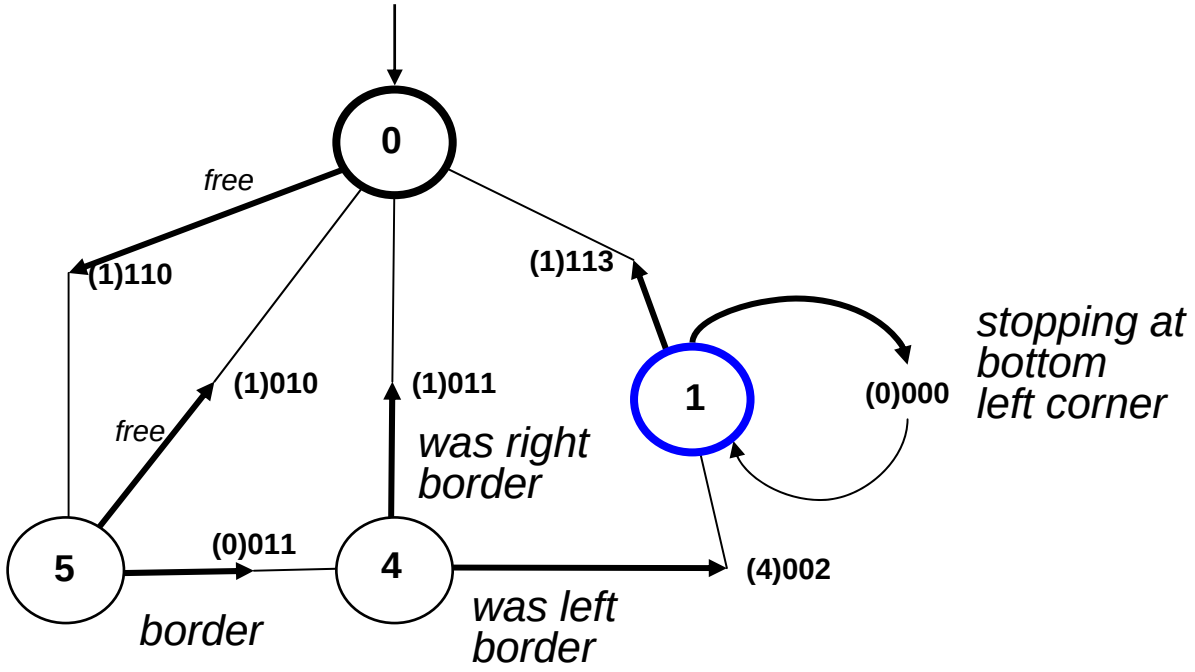


22,1

system stops totally
„hard-termination“

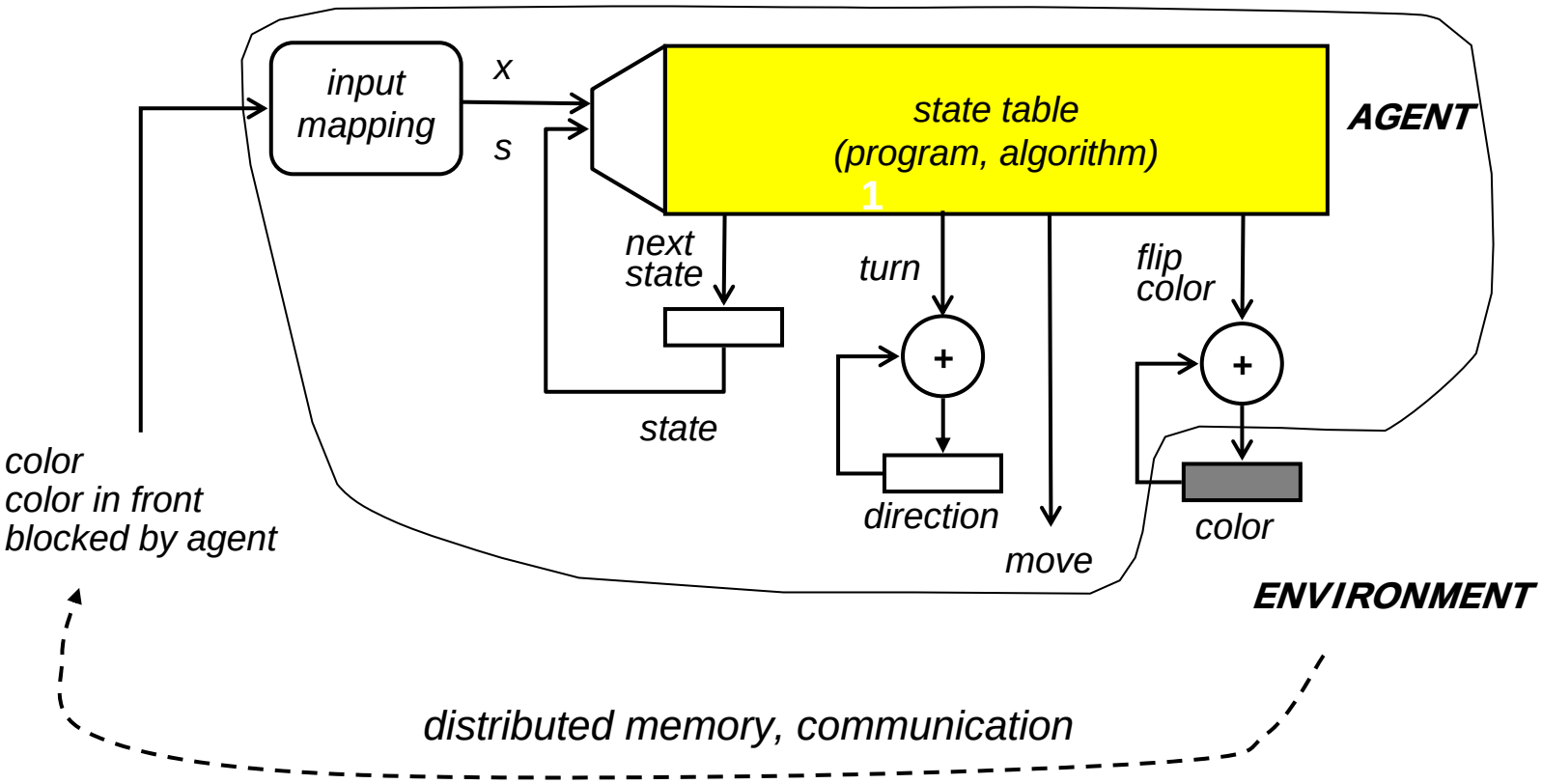
4. Designed Algorithm
with 4 states, Automaton

conditions:
(1): colors CC_F =00 and free
(4): colors CC_F =01 and free
(0): blocked by border
actions:
110: flipcolor, go,--
010: --, go, --
011: --, go, turn right
002: --, --, turn back
113: flipcolor, go, turn left
000: --, wait, --



5. Evolved Algorithms (by a Genetic Algorithm)
Agent‘s behavior

defined by a finite state machine (FSM)



5. Evolved Algorithms
Input mapping

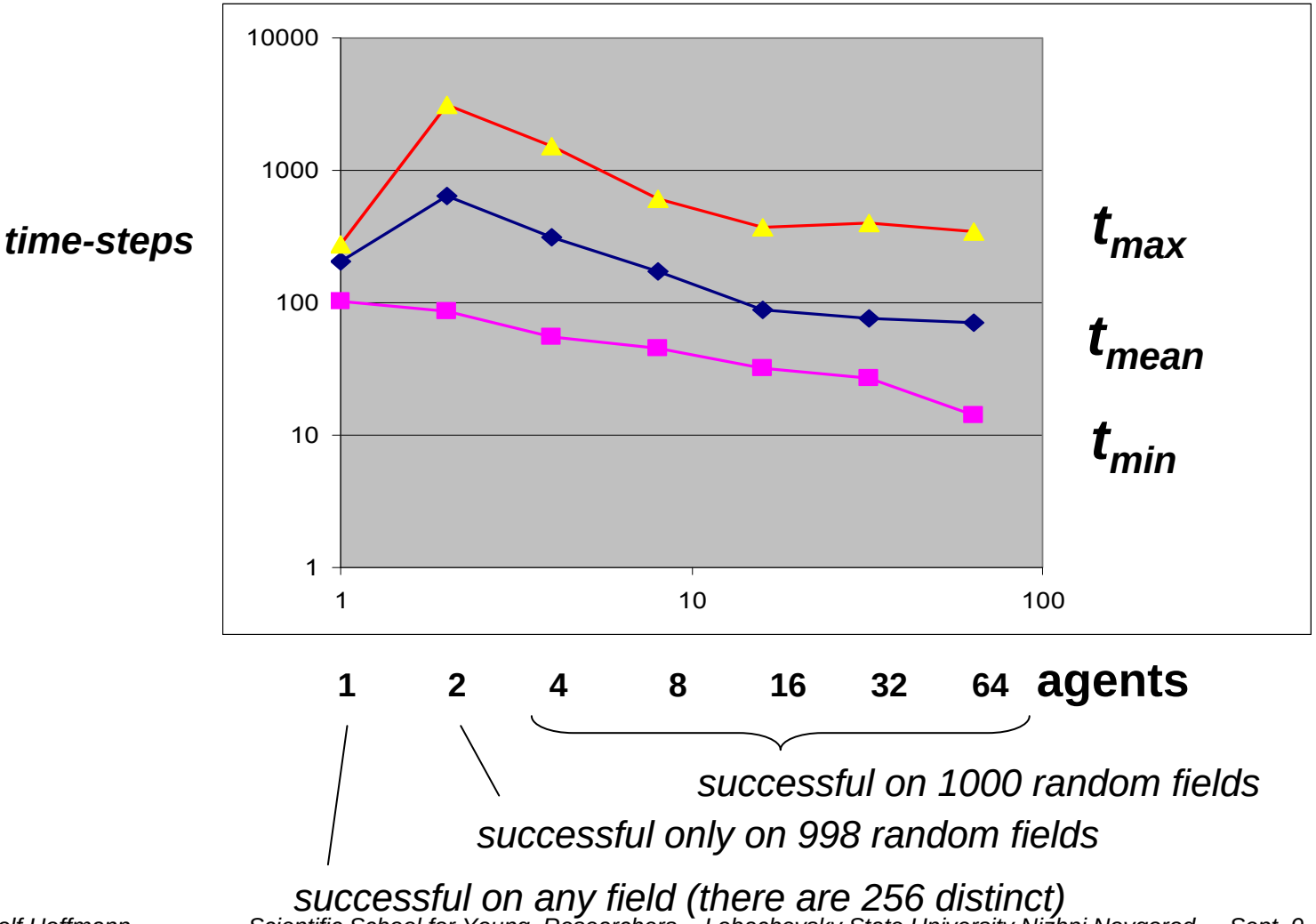
x	color	front color	
0	not used	-1	blocked by border
1	0	0	
2	1	1	not blocked (by agent or conflict)
3	1	0	
4	0	1	
5	0	0	blocked (by agent or conflict)
8	1	1	
7	1	0	
6	0	1	

single-agent system

multi-agent system

fully packed system

5. Evolved Algorithms (by a Genetic Algorithm)
Performance of FSMs: Mean Time



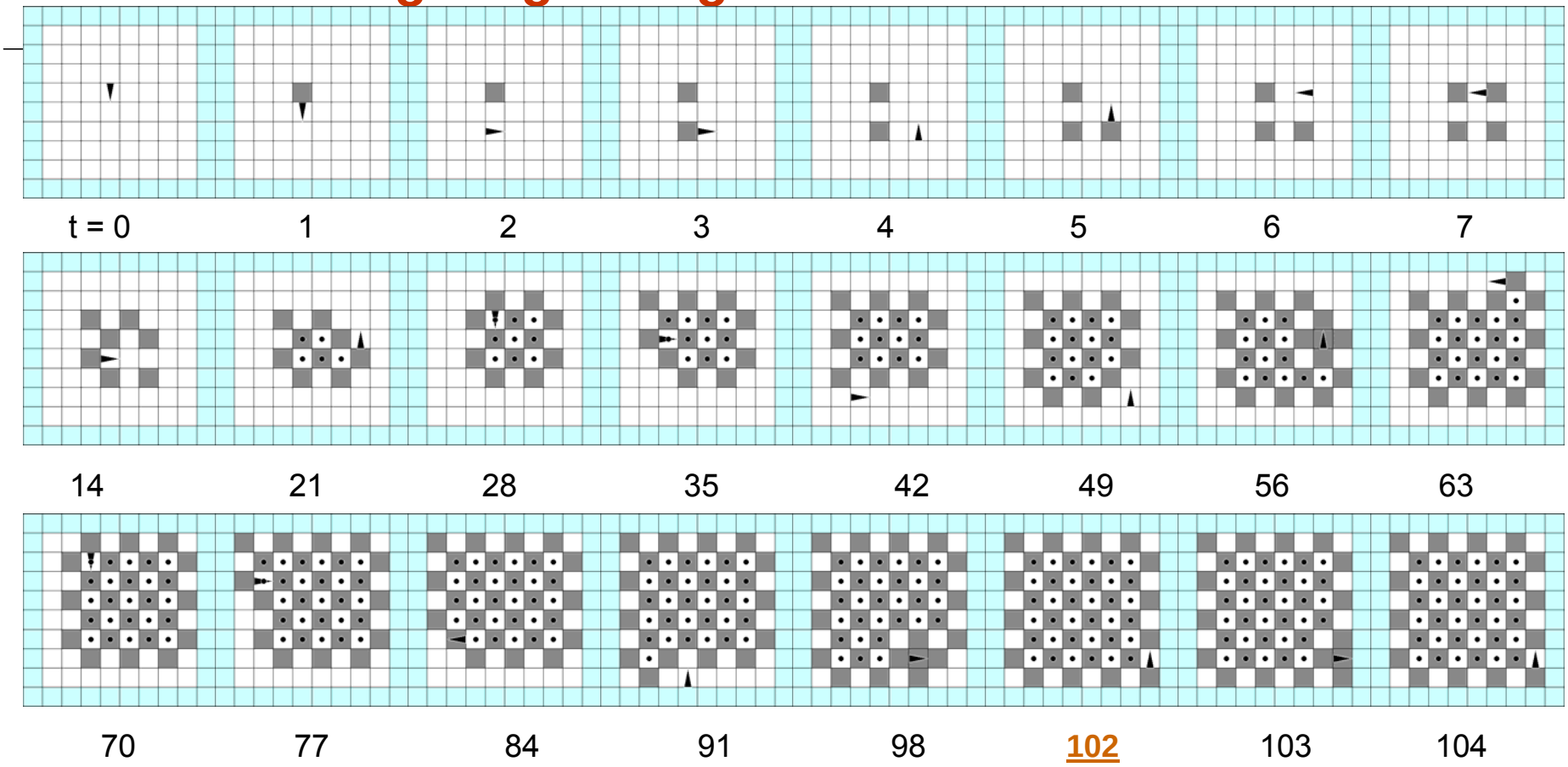
5. Evolved Algorithms (by a Genetic Algorithm)
Performance of FSMs: Speedup, Cost

agents k	*1*	*2*	*4*	*8*	*16*	*32*	*64*
speedup	**1**	0.32	0.65	1.17	**2.31**	2.64	**2.86**
cost per cell t_{mean} *k / 64 (time-units per cell)*	**3.25**	19.8	19.6	21.8	**22**	38.5	**71**
comment on cost	**most economic**				**acceptable**		**too expensive**

density = 1/4

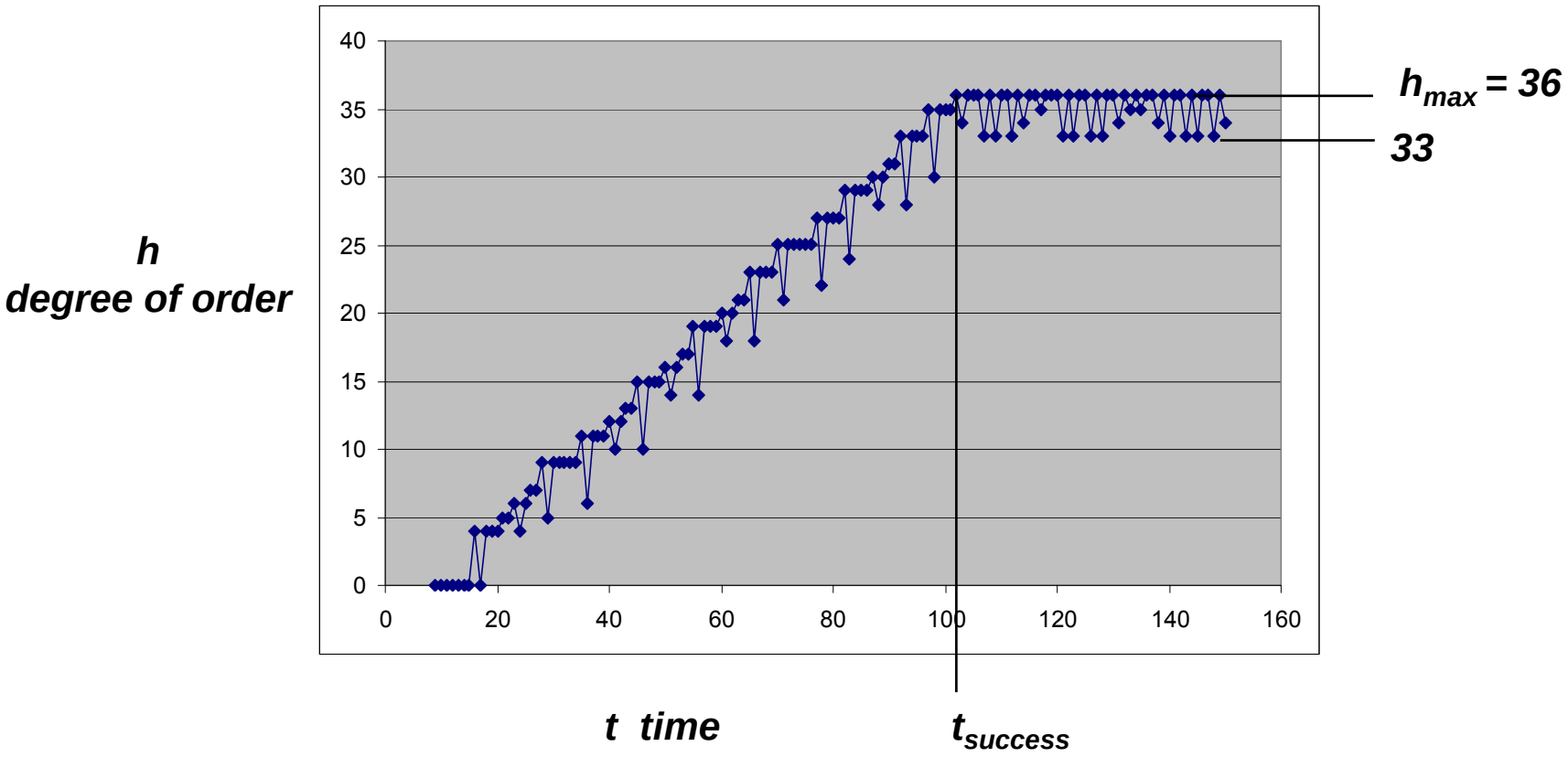
5. Evolved Algorithms (by a Genetic Algorithm)

Simulation single-agent algorithm



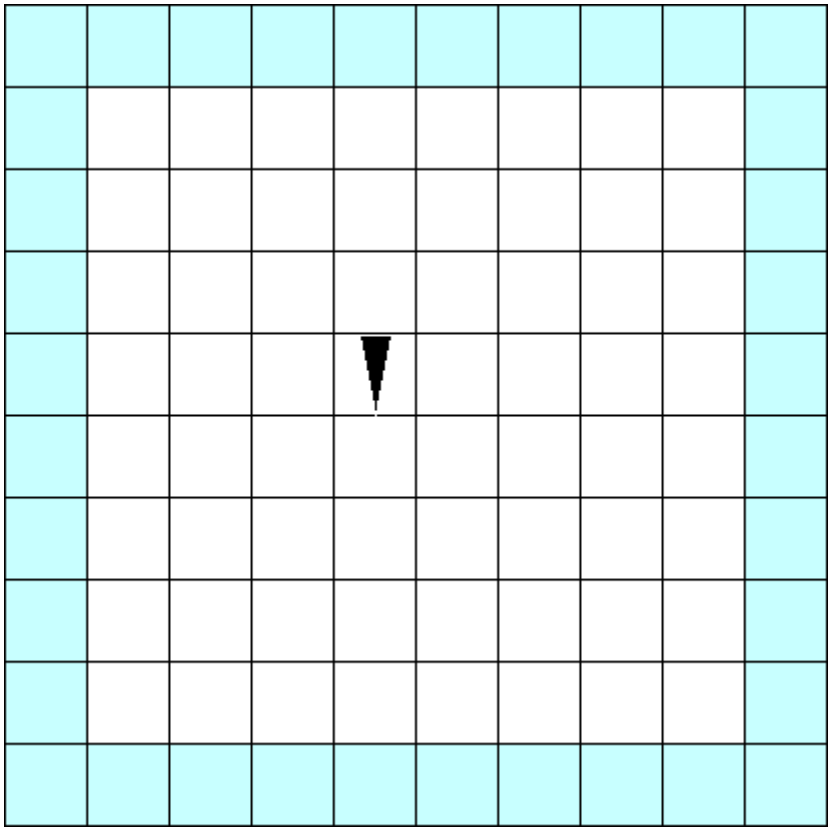
Simulation of a 8x8 single-agent system with the best found FSM.
The pattern is sucessfully formed at $t_{success}$ =102 (minimum).
After success, the agent walks along the border, thereby slighty changing the pattern.

5. Evolved Algorithms (by a Genetic Algorithm)
Single-agent algorithm: degree of order h(t)



After success, the agent walks along the border, thereby changing the degree of order (36 ←→ 33).

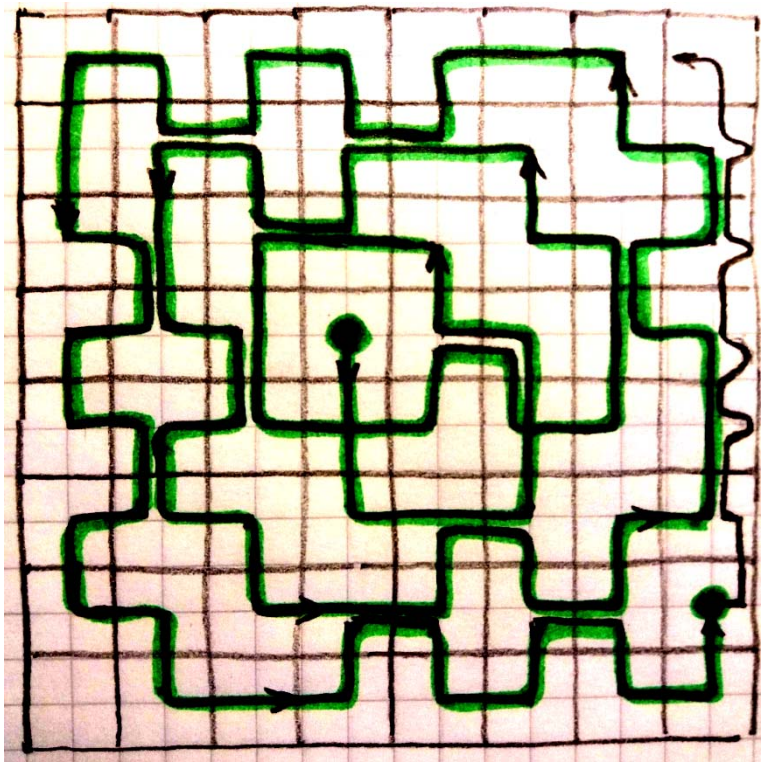
5. Evolved Algorithms (by a Genetic Algorithm)
Simulation single-agent algorithm

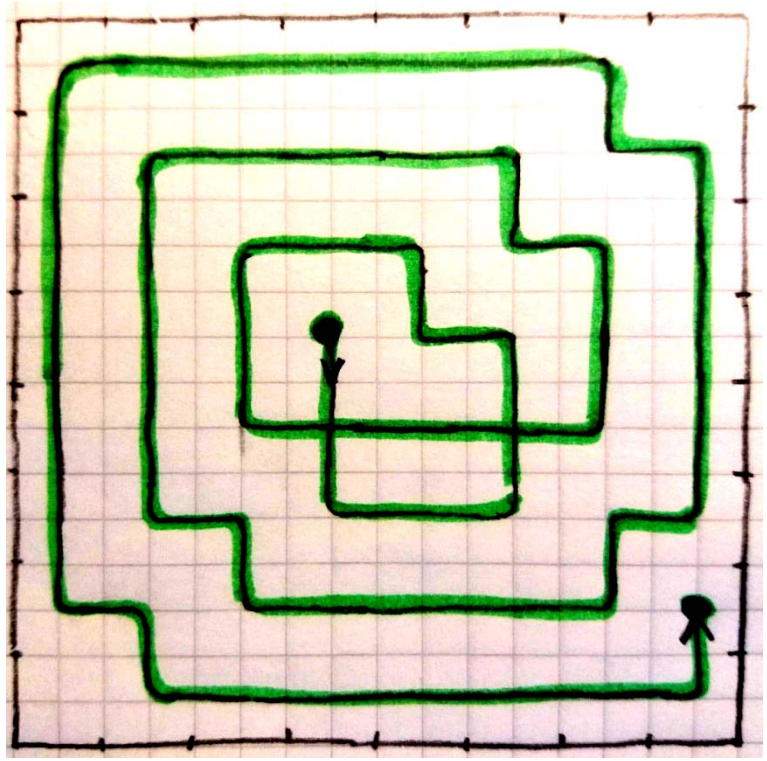


animation ppt only

5. Evolved Algorithms

Path of the single-agent algorithm



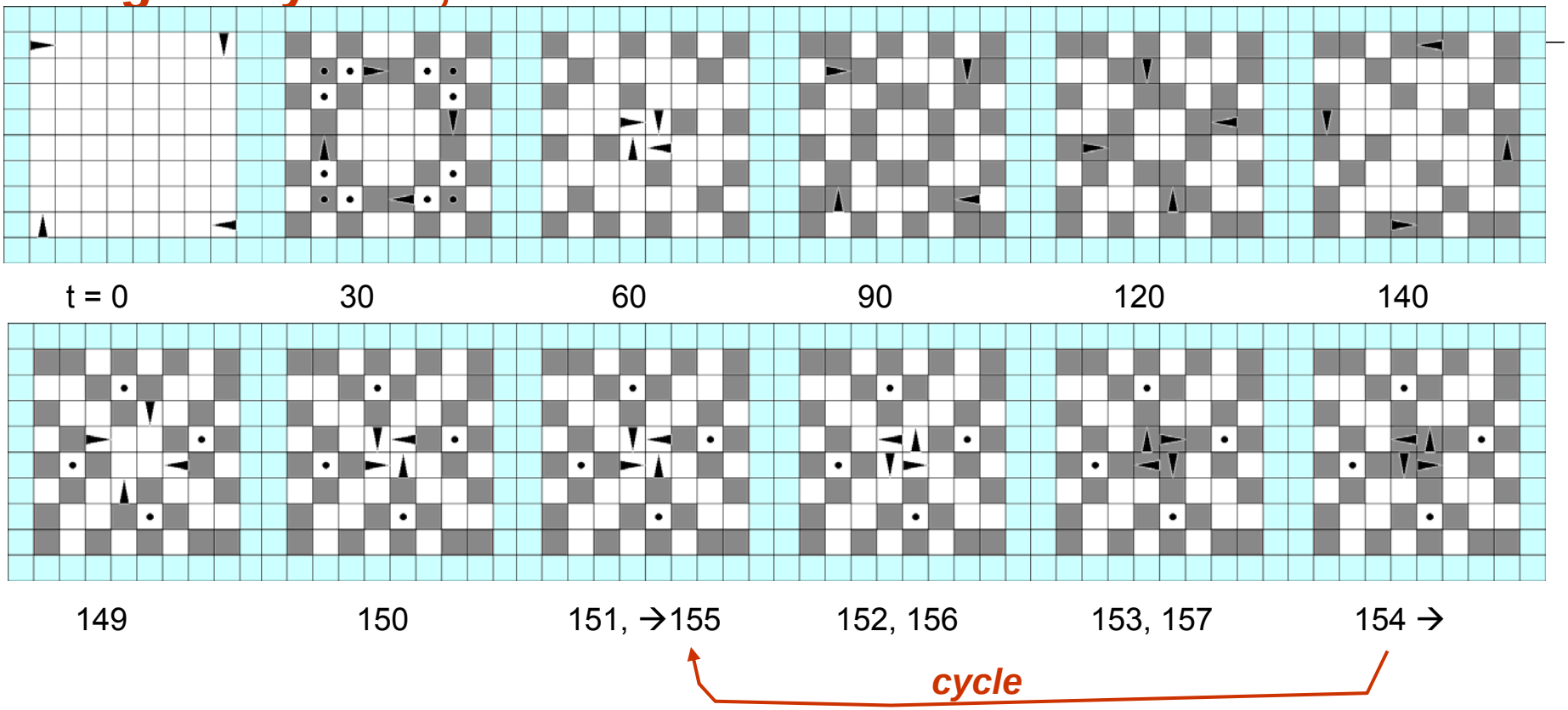


smoothed

$$t_{mean} = n^2(n+11)/6 = O(N^{3/2})$$

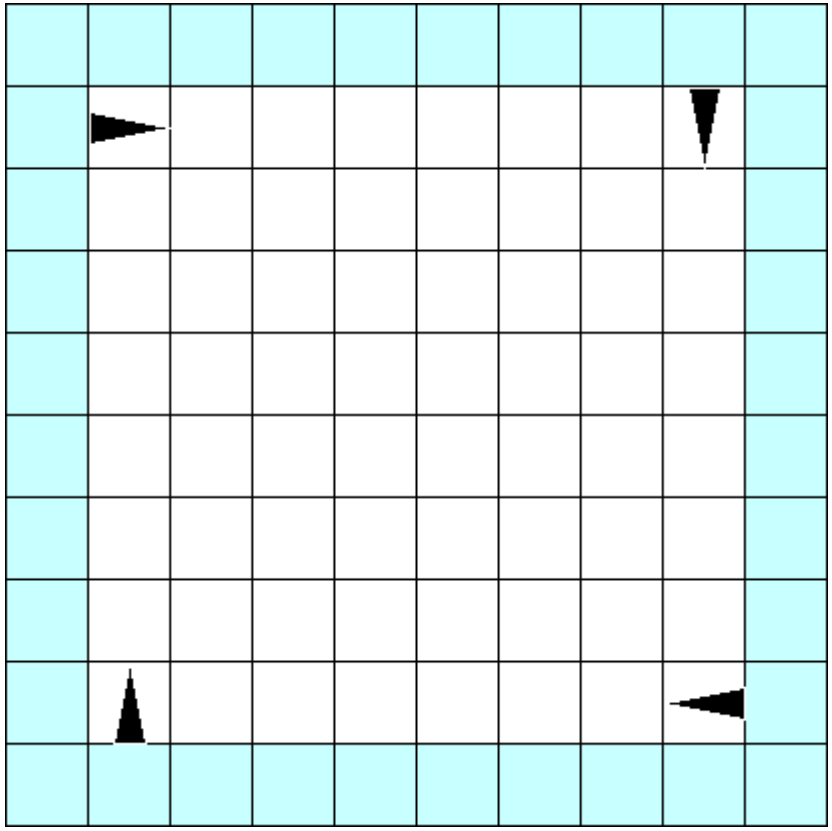
5. Evolved Algorithms

4-agent system, Live-lock



Evolved FSM was successful on 1000 random configurations.
But, starting from a **symmetric configuration**, the desired CB pattern could not be achieved! **The system is running into a live-lock**.

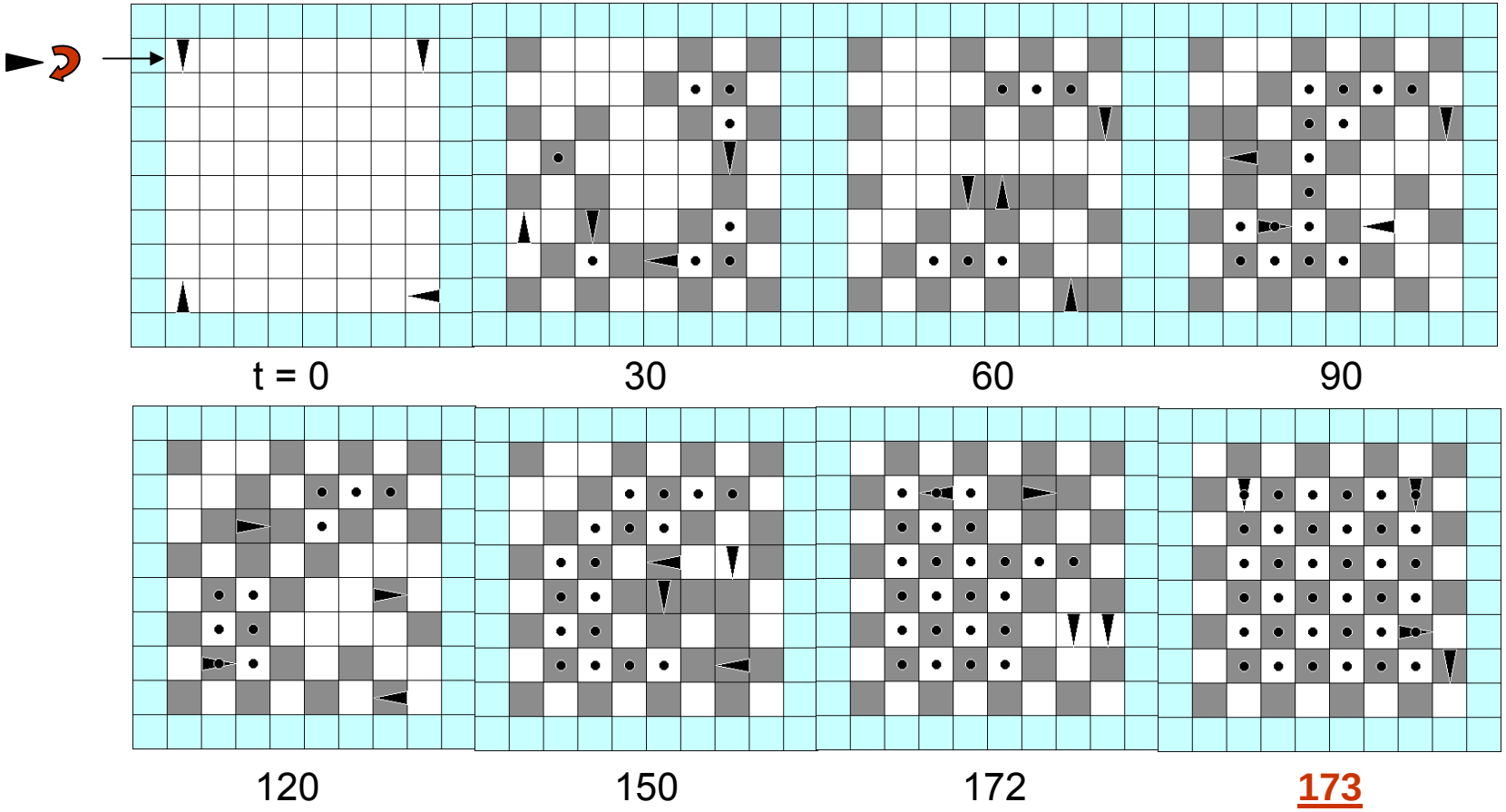
5. Evolved Algorithms
SIMULATION: 4-agent system, Live-lock



animation ppt only

5. Evolved Algorithms

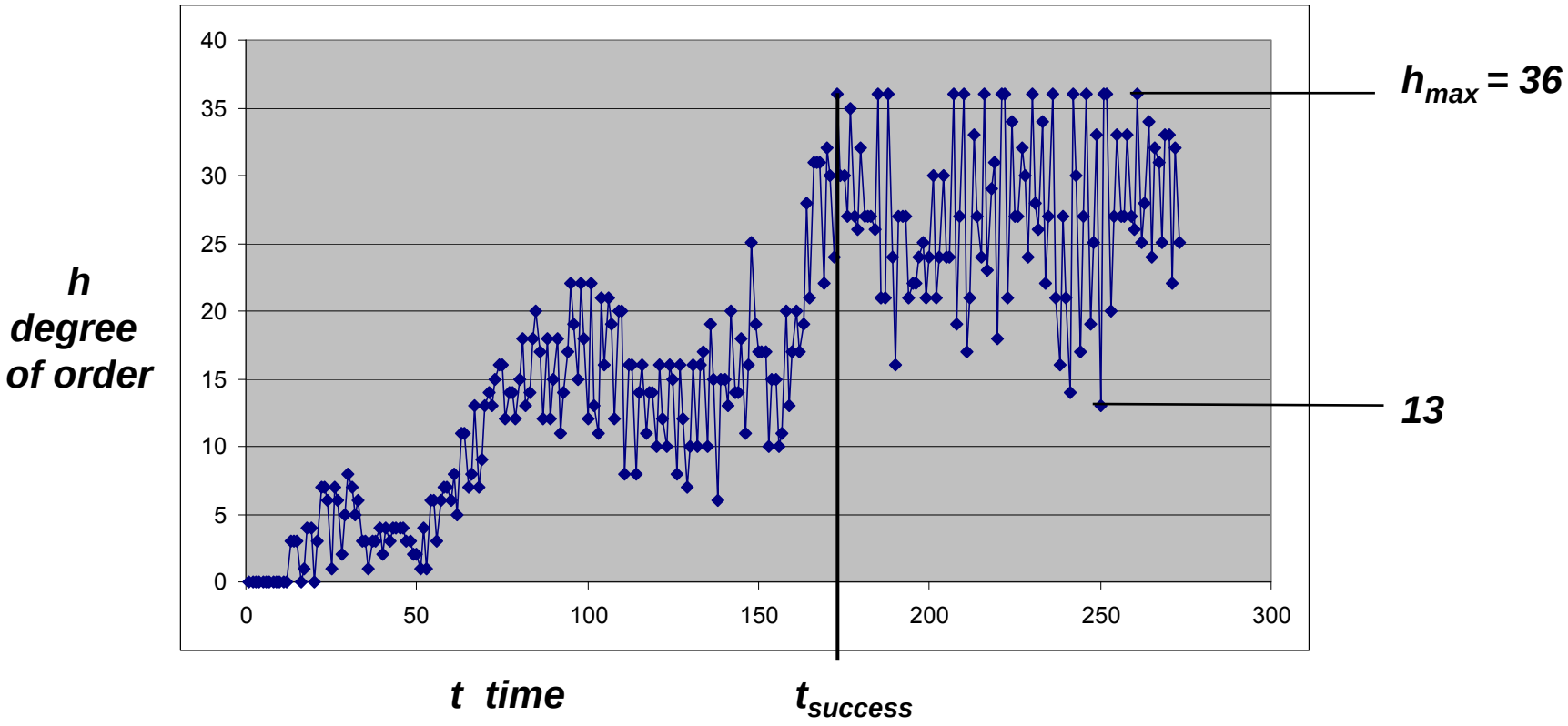
4-agent system: a successful initial configuration



Starting from a slightly different, asymmetric configuration, the CB pattern appears at t = 173.

5. Evolved Algorithms

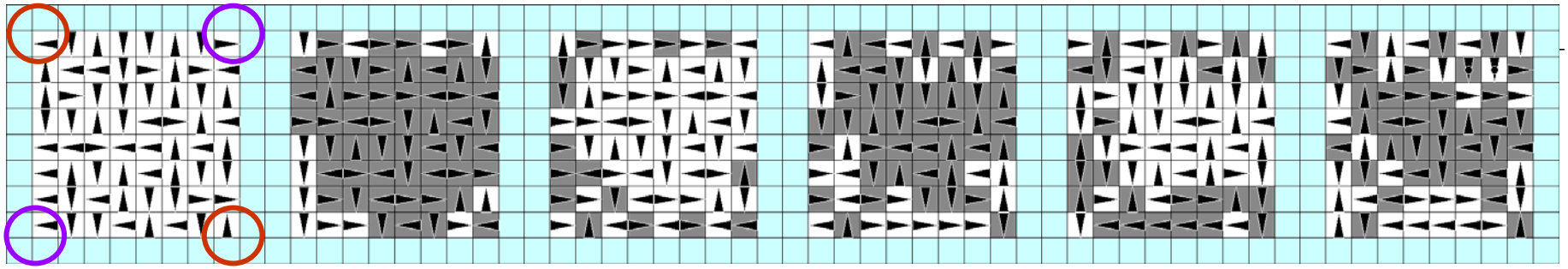
4-agent system: a successful initial configuration, h(t)



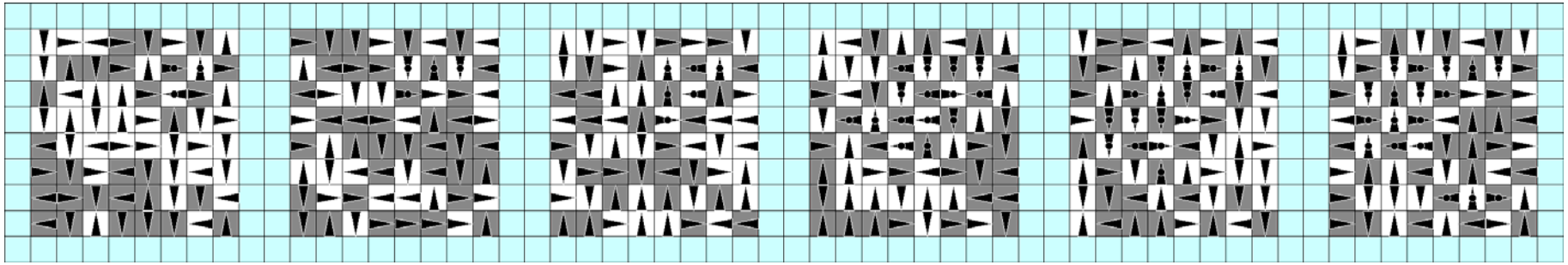
not stable

5. Evolved Algorithms

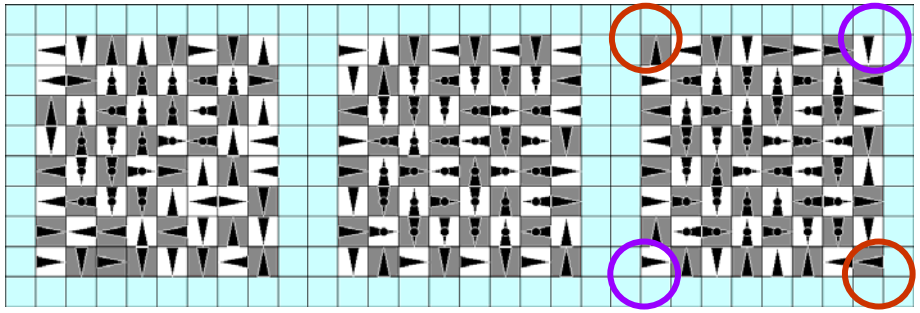
Fully-packed system



t, h = 0, 0 | 1,0 | 2, 0 | 3, 0 | 4, 0 | 5, 2



t, h = 6, 3 | 7, 4 | 8, 6 | 9, 15 | 10, 16 | 11, 18

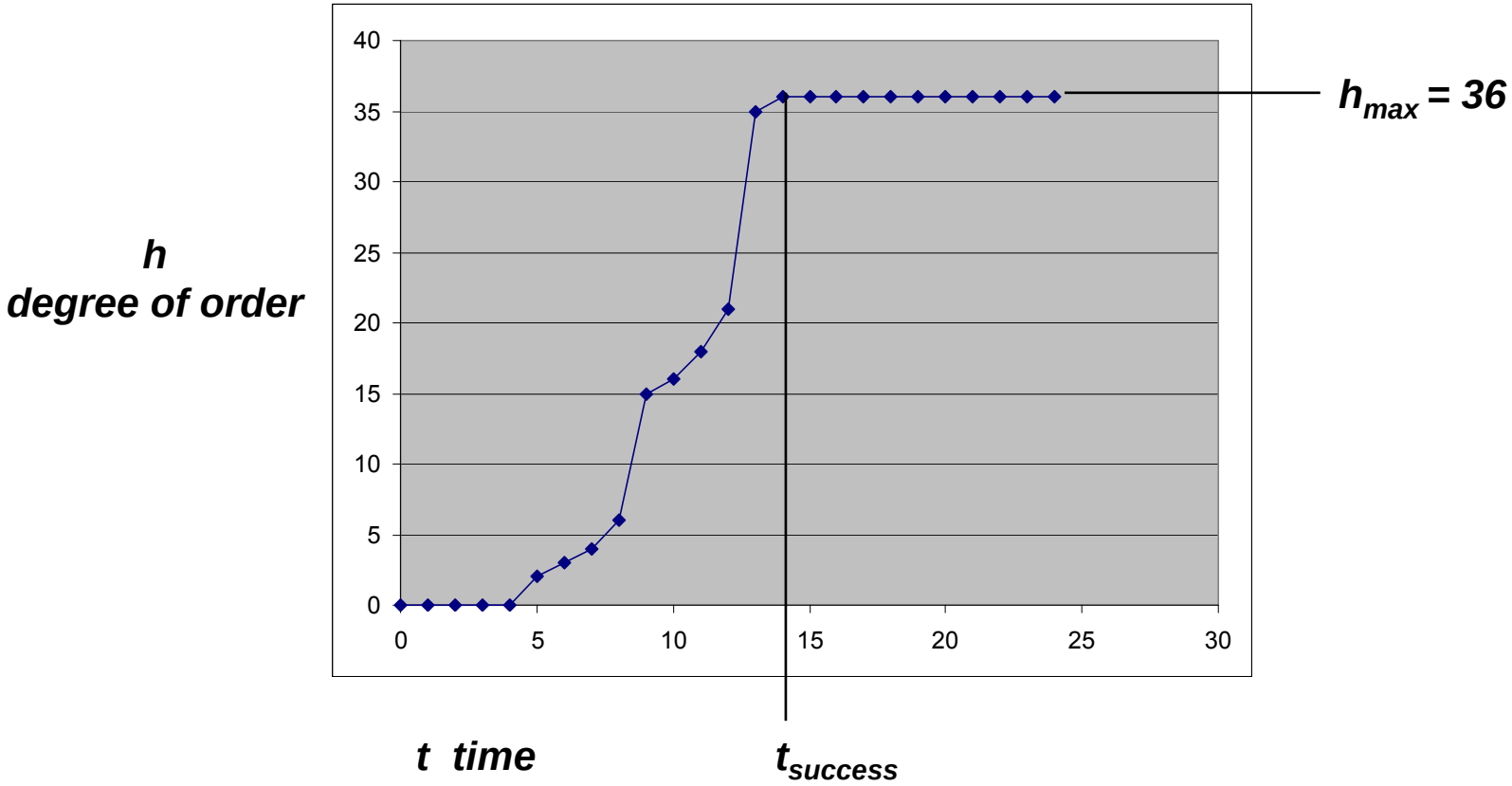


12, 21 | 13, 35 | 14+, 36

Simulation of a 64-agent system.
An initial configuration that yields the minimum time t_{min} = 14.
The pattern remains **stable**, only the directions of the agents are changing.
(„Soft"-Termination)

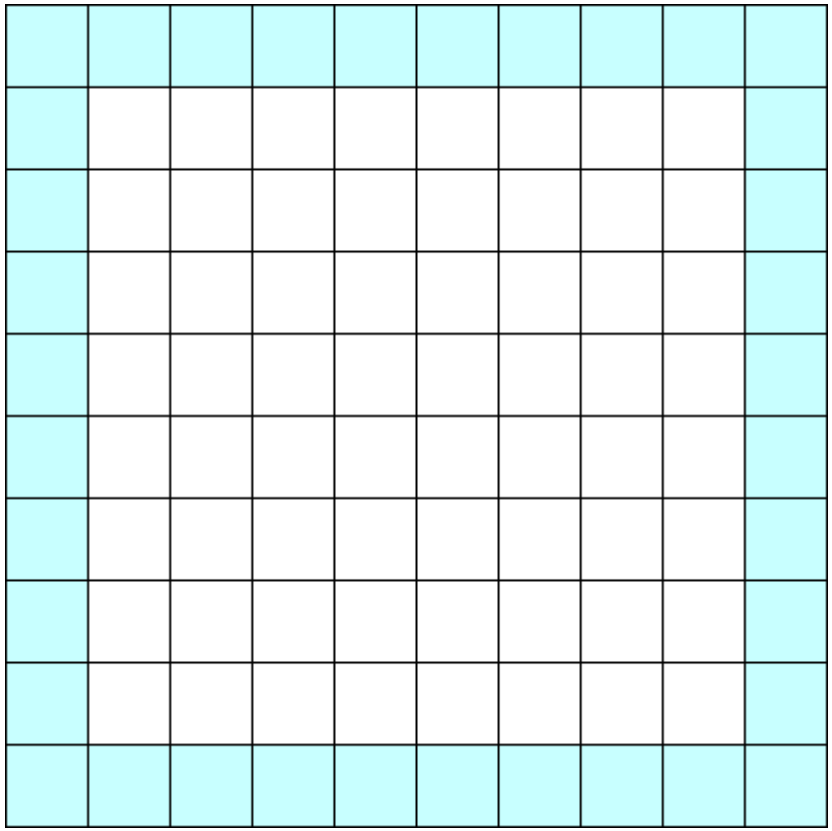
5. Evolved Algorithms

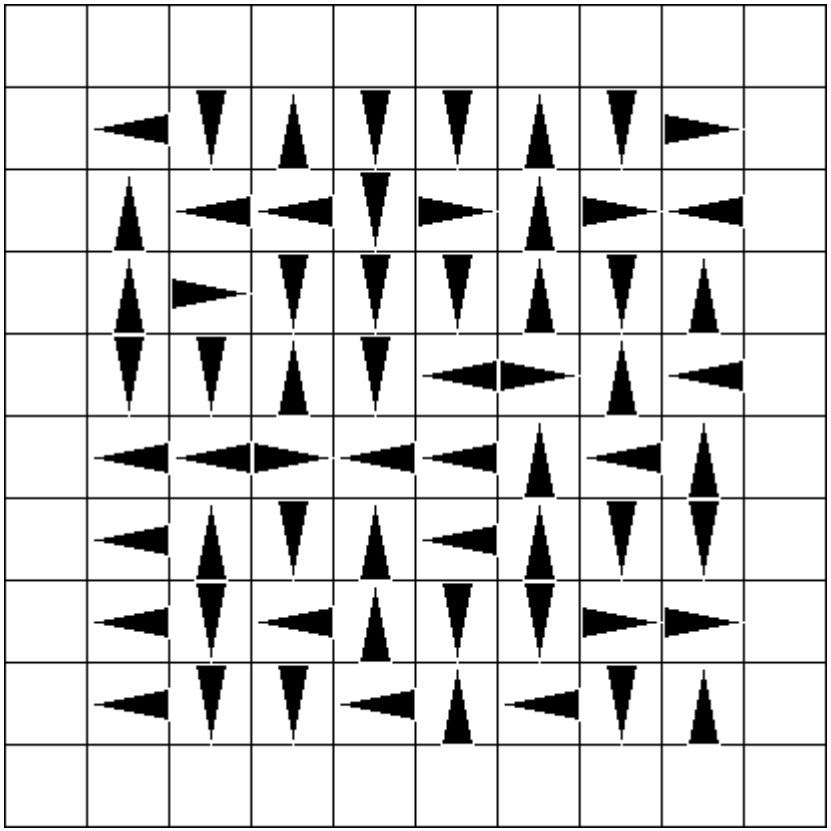
Fully-packed system: Degree of Order h(t)



stable

5. Evolved Algorithms
SIMULATION: Fully-packed system





animation ppt only

Conclusion & Future Work

- **Designed single-agent algorithm**, where agent start in a corner: O(N) time-steps
- **Evolved single-agent algorithm**, where agent starts anywhere: $O(N^{3/2})$ time-steps
- **Speedup** is 2.64 for 8x8 system with 16 agents
- **Live-locks:** Multiple-agent systems are not always successful, especially for symmetric init. configurations
- **FUTURE WORK:**
  - break the symmetry
  - study the termination problem in more detail
  - find more general and more effective algorithms

Thank you very much for you attention !