Evolving Hybrid Time-Shuffled Behavior of Agents

Computer Architecture Group

NIDISC 2010, April 19, Atlanta

<u>Patrick Ediger</u> and Rolf Hoffmann

Image Source: www.kulturschnitte.de (Claudia Lohmann)

Motivation

TECHNISCHE UNIVERSITÄT DARMSTADT

Develop *efficient GA-based methods* that allow to find the optimal *local behavior of moving agents*.

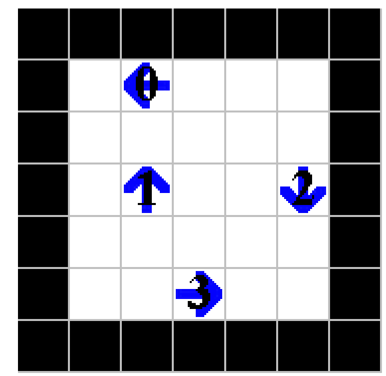
- **Hybrid behavior**: mixture of different behaviors (strategies)
  - Is mixing effective?
  - In which way can we mix it?

- Applications with agents
  - Simulation of “real” worlds
  - Artificial worlds
  - Distributed algorithms
  - Routing ...

Problem Statement: All-to-All Communication

TECHNISCHE UNIVERSITÄT DARMSTADT

- Given is a 2D-Cellular Automaton (CA) with moving agents.
- Initially, the *information* is distributed mutually exclusive.
- All agents shall exchange all their information.
- Information is exchanged and propagated when agents meet with a cell in between them.

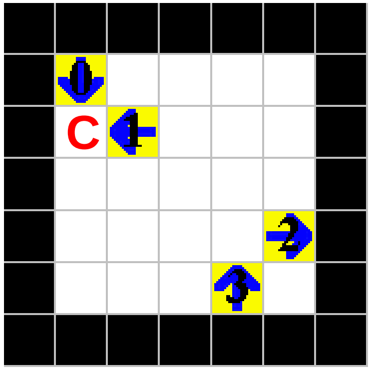


initial information: **1000** **0100** **0010** **0001**

Problem Statement: All-to-All Communication

TECHNISCHE UNIVERSITÄT DARMSTADT

- Given is a 2D-Cellular Automaton (CA) with moving agents.
- Initially, the *information* is distributed mutually exclusive.
- All agents shall exchange all their information.
- Information is exchanged and propagated when agents meet with a cell in between them.

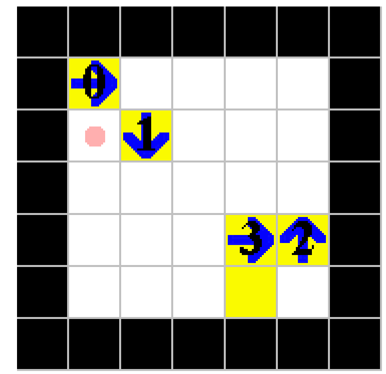


	0	1	2	3
initial information:	1000	0100	0010	0001
	1100	1100	0010	0001

Problem Statement: All-to-All Communication

TECHNISCHE UNIVERSITÄT DARMSTADT

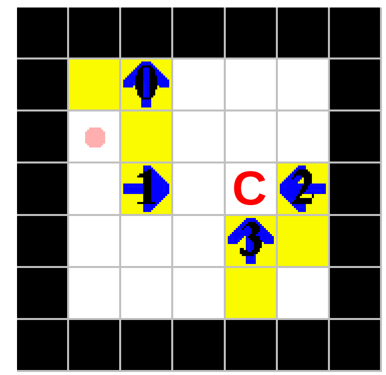
- Given is a 2D-Cellular Automaton (CA) with moving agents.
- Initially, the *information* is distributed mutually exclusive.
- All agents shall exchange all their information.
- Information is exchanged and propagated when agents meet with a cell in between them.

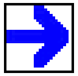
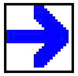
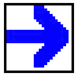
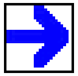


	0	1	2	3
initial information:	1000	0100	0010	0001
	1100	1100	0010	0001

Problem Statement: All-to-All Communication

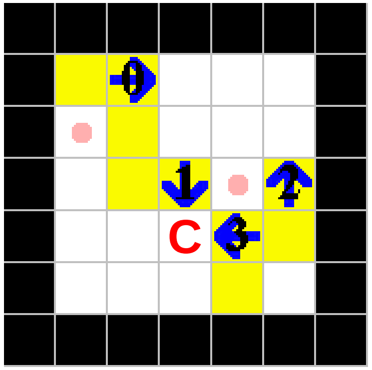
- Given is a 2D-Cellular Automaton (CA) with moving agents.
- Initially, the *information* is distributed mutually exclusive.
- All agents shall exchange all their information.
- Information is exchanged and propagated when agents meet with a cell in between them.



	 0	 1	 2	 3
initial information:	1000	0100	0010	0001
	1100	1100	0010	0001
	1100	1100	0011	0011

Problem Statement: All-to-All Communication

- Given is a 2D-Cellular Automaton (CA) with moving agents.
- Initially, the *information* is distributed mutually exclusive.
- All agents shall exchange all their information.
- Information is exchanged and propagated when agents meet with a cell in between them.

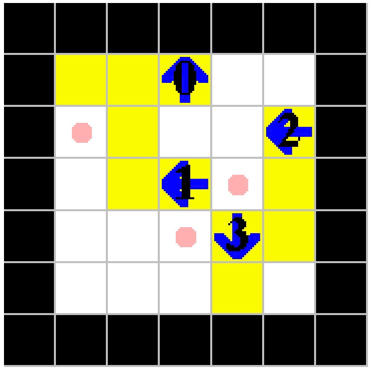


	0	1	2	3
initial information:	1000	0100	0010	0001
	1100	1100	0010	0001
	1100	1100	0011	0011
	1100	1111	0011	1111

Problem Statement: All-to-All Communication

TECHNISCHE UNIVERSITÄT DARMSTADT

- Given is a 2D-Cellular Automaton (CA) with moving agents.
- Initially, the *information* is distributed mutually exclusive.
- All agents shall exchange all their information.
- Information is exchanged and propagated when agents meet with a cell in between them.

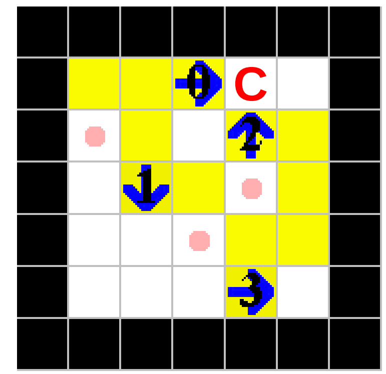


	0	1	2	3
initial information:	1000	0100	0010	0001
	1100	1100	0010	0001
	1100	1100	0011	0011
	1100	1111	0011	1111

Problem Statement: All-to-All Communication

TECHNISCHE UNIVERSITÄT DARMSTADT

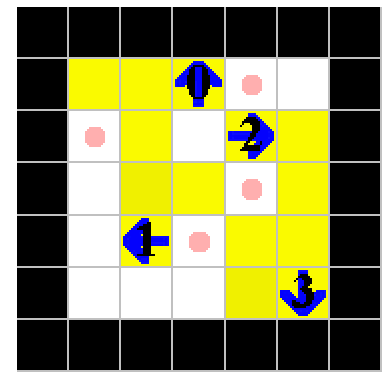
- Given is a 2D-Cellular Automaton (CA) with moving agents.
- Initially, the *information* is distributed mutually exclusive.
- All agents shall exchange all their information.
- Information is exchanged and propagated when agents meet with a cell in between them.



	0	1	2	3
initial information:	1000	0100	0010	0001
	1100	1100	0010	0001
	1100	1100	0011	0011
	1100	1111	0011	1111
	1111	1111	1111	1111

Problem Statement: All-to-All Communication

- Given is a 2D-Cellular Automaton (CA) with moving agents.
- Initially, the *information* is distributed mutually exclusive.
- All agents shall exchange all their information.
- Information is exchanged and propagated when agents meet with a cell in between them.

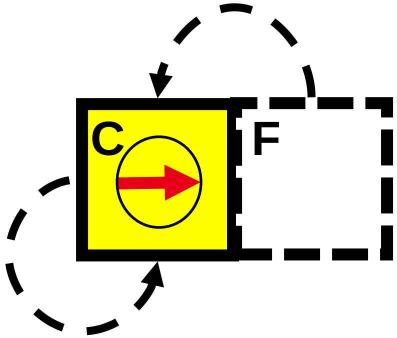


	0	1	2	3
initial information:	1000	0100	0010	0001
	1100	1100	0010	0001
	1100	1100	0011	0011
	1100	1111	0011	1111
	1111	1111	1111	1111

Cellular Automata Model:
Modeling Moving Agents

- Agents are directed: N, E, S, W

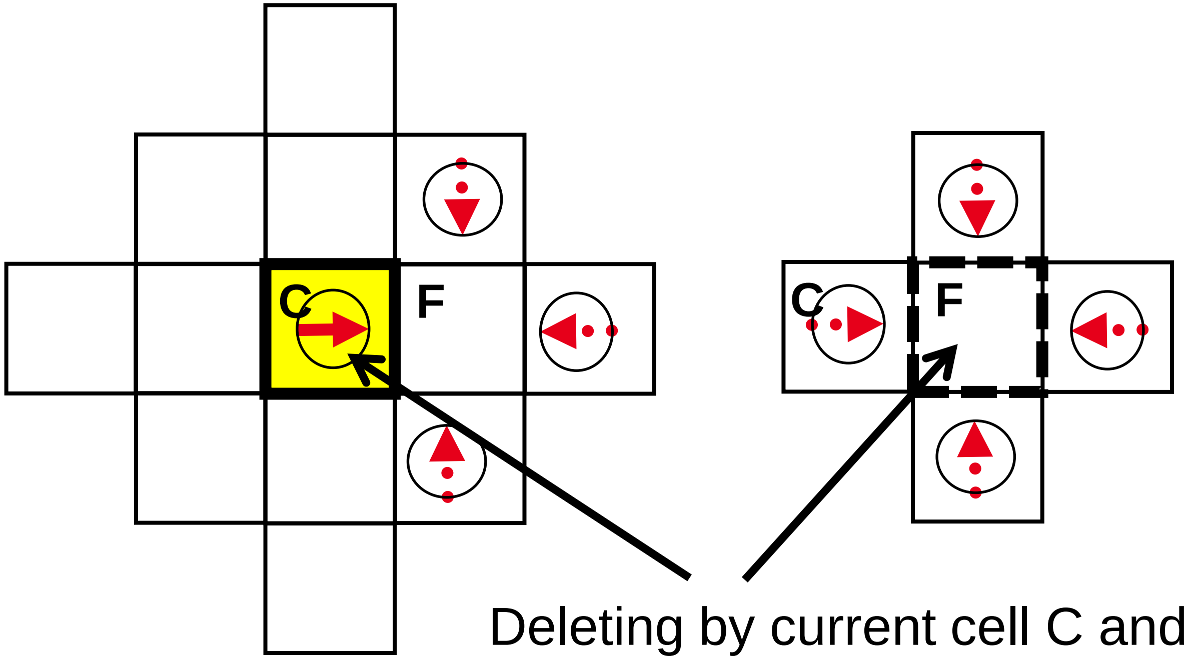
front cell F reads and copies the agent



current cell C deletes the agent from itself

Cellular Automata Model:
Extended Neighborhood

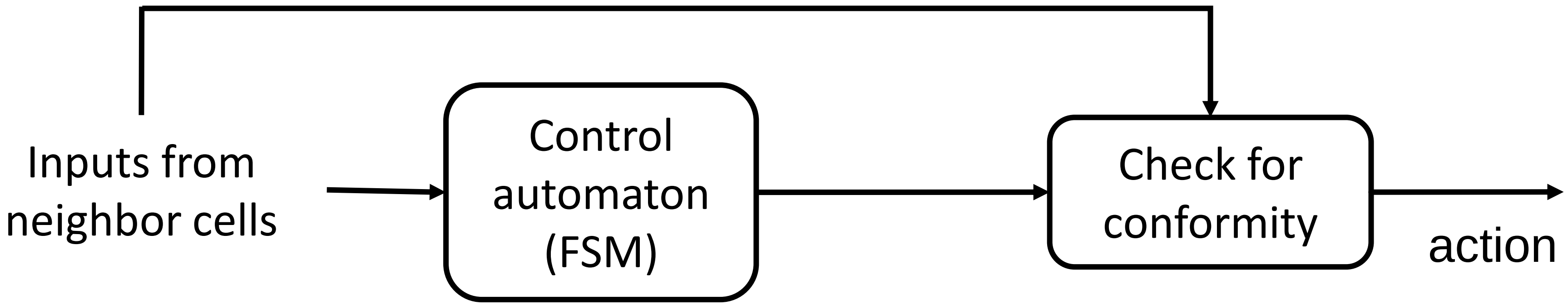
- Conflict resolution requires an extended neighborhood (Manhattan Distance 2)



Deleting by current cell C and copying by the front cell F must be consistent and thus based on the same information.

Modeling Agent Behavior (I)

- Agents react on inputs from the neighbor cells.
- Agents are controlled by finite state machines (FSM) with limited complexity.
- The output of the FSM activates an action, that is checked for conformity.
  - Turn **R**ight/**L**eft (+ **m**ove ahead if possible): *R*, *L*, *Rm*, *Lm*



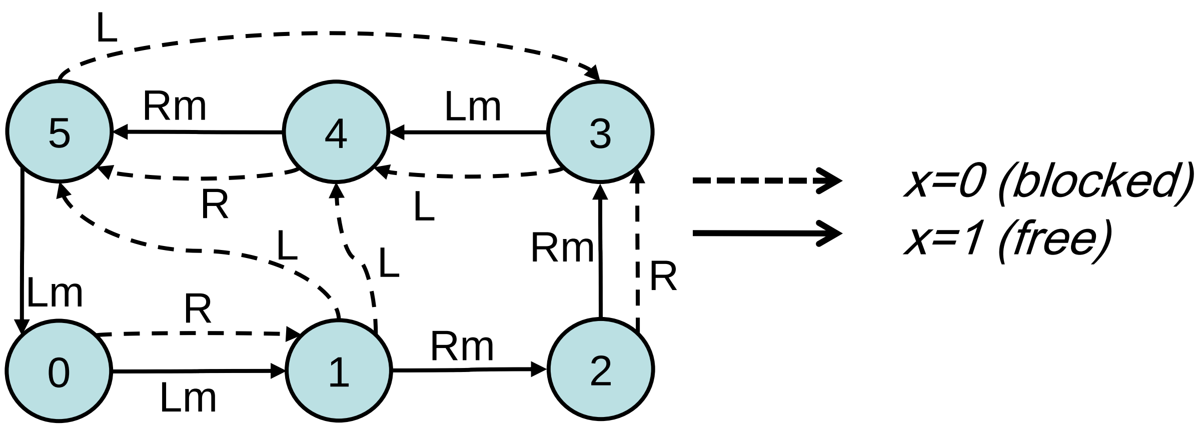
Cellular Automata Model:

Modeling Agent Behavior (II)

TECHNISCHE UNIVERSITÄT DARMSTADT

- Decision between the actions *Lm*, *Rm*, *L* and *R* is defined by a finite state machine (e.g., 6-states).

state graph



state table, defining the behavior (algorithm) of an agent, used as genome

x	0						1						*input*
s	0	1	2	3	4	5	0	1	2	3	4	5	*state*
s',y	1,1	5,0	3,0	4,1	5,1	3,0	1,0	2,1	3,1	4,0	5,1	0,0	*nextstate, output*
action	R	L	R	L	R	L	Lm	Rm	Rm	Lm	Rm	Lm	*action*
i	0	1	2	3	4	5	6	7	8	9	10	11	*index used in GA*

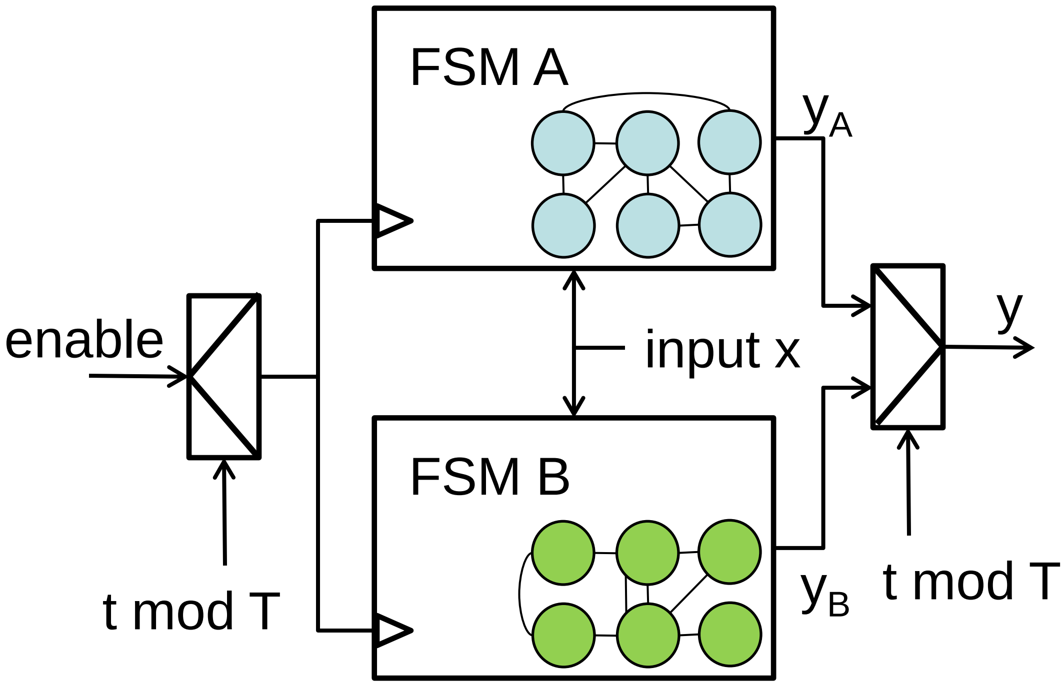
Goal of this particular investigation

Develop *efficient GA-based methods* that allow to find the optimal *local behavior of moving agents*.

- NIDISC 2009
  - ***non-hybrid behavior*** vs. ***hybrid behavior***
  - hybrid behavior by separately evolving FSMs for subtasks and joining FSMs by ***time-shuffling***
- NIDISC 2010
  - Can hybrid behavior be ***evolved directly*** (not separately)?
  - Is directly evolving more efficient than separately evolving?

The Time-Shuffling Technique

- Time-shuffling exploits the individual abilities of two different algorithms (strategies) by **alternating** them **in time**.



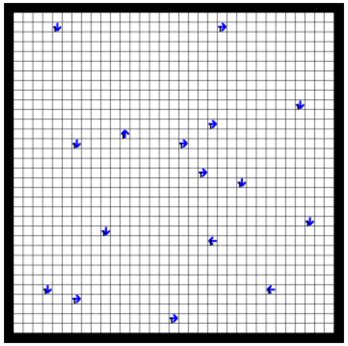
- FSMs A and B are used alternately, changing every *T* CA-generations.
- Note that AB ≠ BA
- T can be different for A and B (T_A and T_B)
- here: FSM with 6 states, T varied from 1-600

The Problem Set of Initial Configurations

- A given set of initial configurations of the environments.
  - 20 environments with 33x33 cells
  - k = 16 agents placed randomly in the grid with a random direction

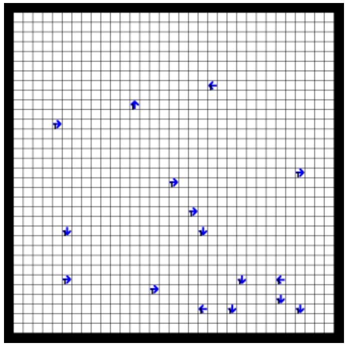
  - **Subset A**: 10 environments with border

A



0

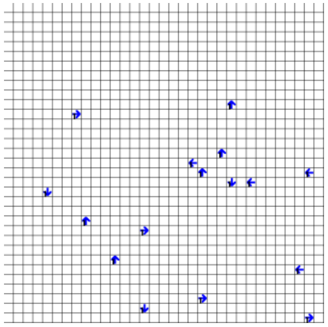
...



9

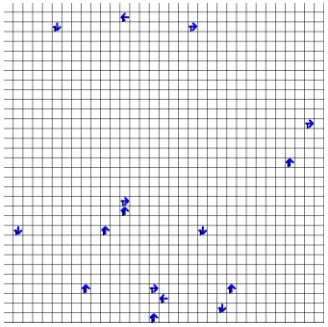
  - **Subset B**: 10 environments with wrap-around

B



10

...



19

Types of Evolved Algorithms

- From NIDISC 2009:
  - **Z**: non-hybrid (one FSM), evolved on entire set (A and B)
  - **XY_T**: hybrid (two FSMs, one shuffle period), evolved separately (X on subset A, Y on subset B)

- New:
  - **UV_T**: hybrid (**two FSMs**, **one shuffle period**), evolved directly on entire set (A and B)
  - **U_TV_T**: hybrid (**two FSMs**, **two shuffle periods**), evolved directly on entire set (A and B)

Fitness Function

- Each FSM is assigned to a certain fitness value F

$$\boxed{\mathbf{F = 10^5(16 - a_i) + 10^4(1 - c) + g}}$$

- a_i: no. of completely informed agents (with bit vector 11…1)
- c = 1, if any information was exchanged, else c = 0
- g: the number of CA-generations needed to fulfill the task completely (all agents are informed)
- Lower values for F indicate a better fitness.
- **F = s**, if the task was solved for the simulated environment.

Island Model GA

- Hybrid Genome***:***
  - *state table(FSM-A) + state table (FSM B) + T*
    - *(search space: $600 \cdot 12^{24}$)*
  - *state table(FSM-A) + state table (FSM B) + T_A + T_B*
    - *(search space: $600^2 \cdot 12^{24}$)*

- ***P*** populations of ***N*** individuals are updated in each generation. In each generation ***M*** children are produced in each population.
- The union of the current *N* individuals and the *M* children
  - sorted according to their fitness
  - *N* best are selected forming the next population.

Crossover Techniques

FSM A													FSM B													T_A	T_B	
s',y	1,1	5,0	3,0	4,1	5,1	3,0	1,0	2,1	3,1	4,0	5,1	0,0	s',y	4,0	2,0	4,1	1,0	1,1	5,0	0,0	3,1	1,0	4,0	3,0	0,1	134	27	Parent 1
s',y	2,1	4,1	2,0	4,0	0,1	1,1	3,0	2,1	5,0	2,0	4,1	0,1	s',y	1,0	2,1	3,1	3,0	5,0	4,1	0,1	0,1	2,1	5,0	5,1	2,1	421	12	Parent 2
s',y	2,1	5,0	2,0	4,0	5,1	3,0	1,0	2,1	5,0	4,0	4,1	0,0	s',y	4,0	2,0	4,1	3,0	5,0	4,1	0,1	3,1	1,0	5,0	5,1	0,1	421	12	Child (a)
s',y	2,1	4,1	3,0	4,1	0,1	1,1	1,0	2,1	3,1	2,0	4,1	0,1	s',y	1,0	2,0	4,1	3,0	1,1	5,0	0,1	0,1	1,0	4,0	3,0	2,1	277	19	Child (b)

- Each component either taken from parent A or parent B

- Technique *a*: value T of one of the parents chosen
- Technique *b*: childs value T is average of parents‘ values

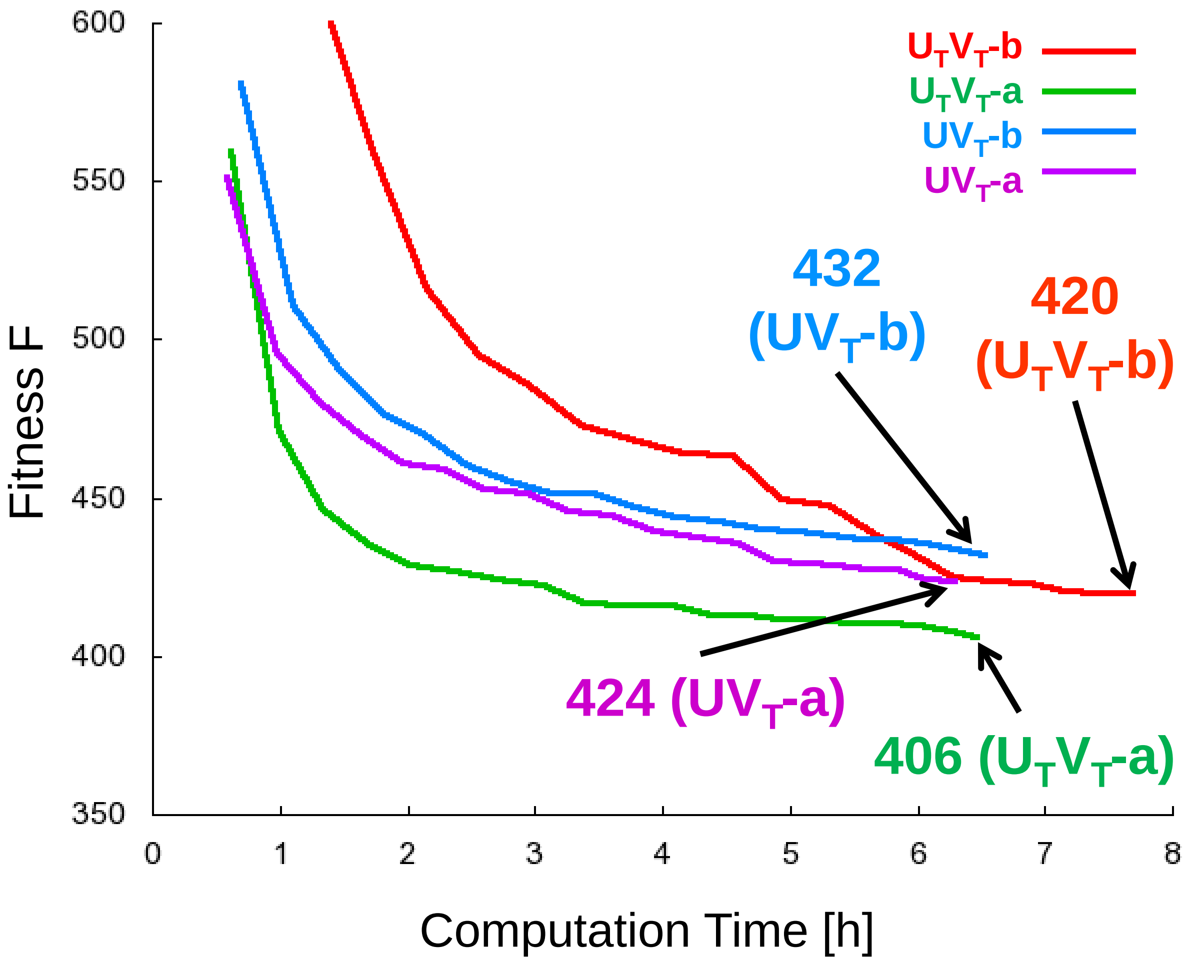
- → UV_T-a, UV_T-b, U_TV_T-a, U_TV_T-b

Best Fitness Values (I)

Averaged over 6 independent runs of the GA

Crossover technique *a* is more efficient than technique *b*.

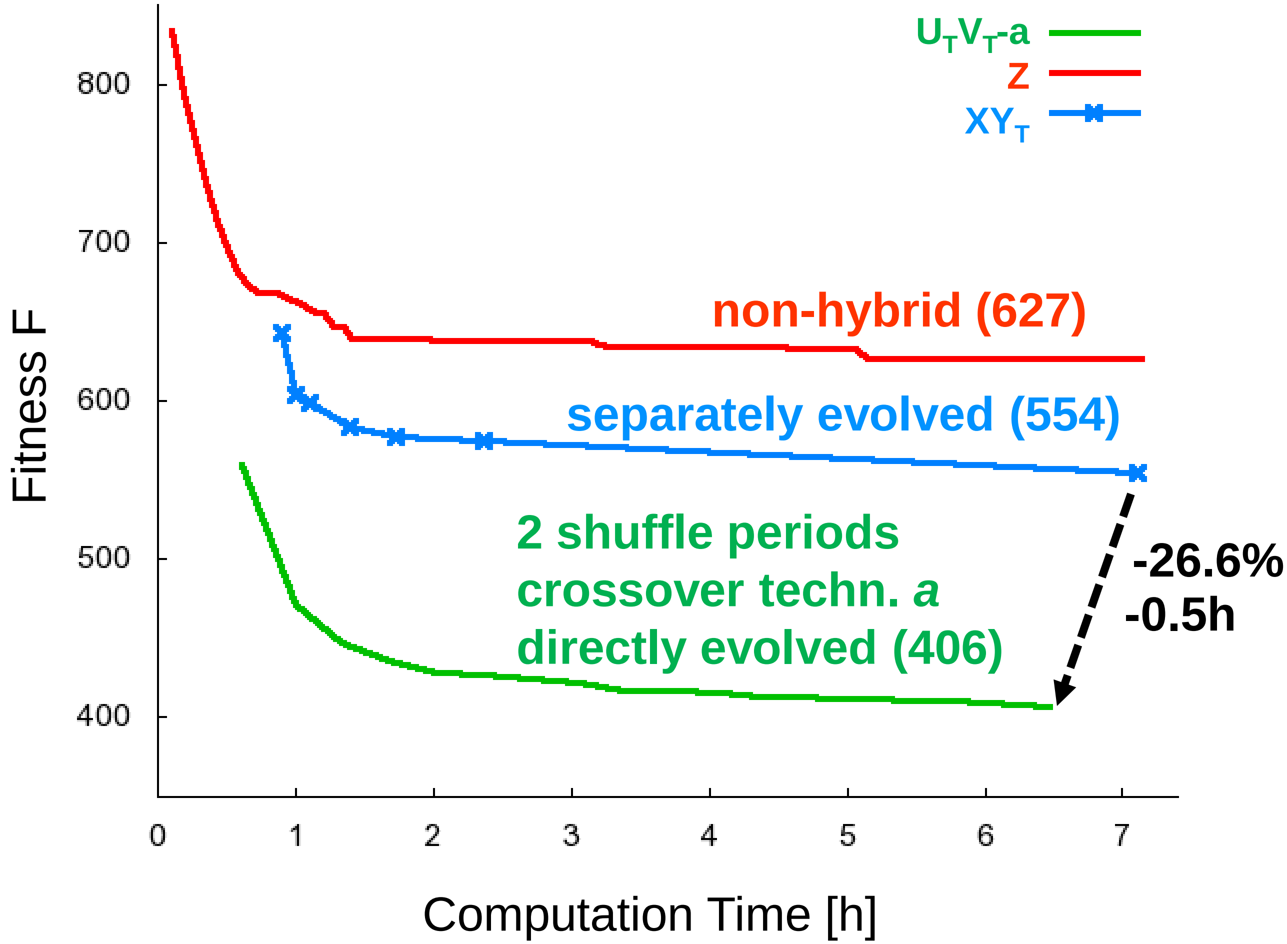
GA with 2 shuffle periods finds better algorithms, but is less reliable.



Best Fitness Values (II)

TECHNISCHE UNIVERSITÄT DARMSTADT

Directly evolving is more efficient.

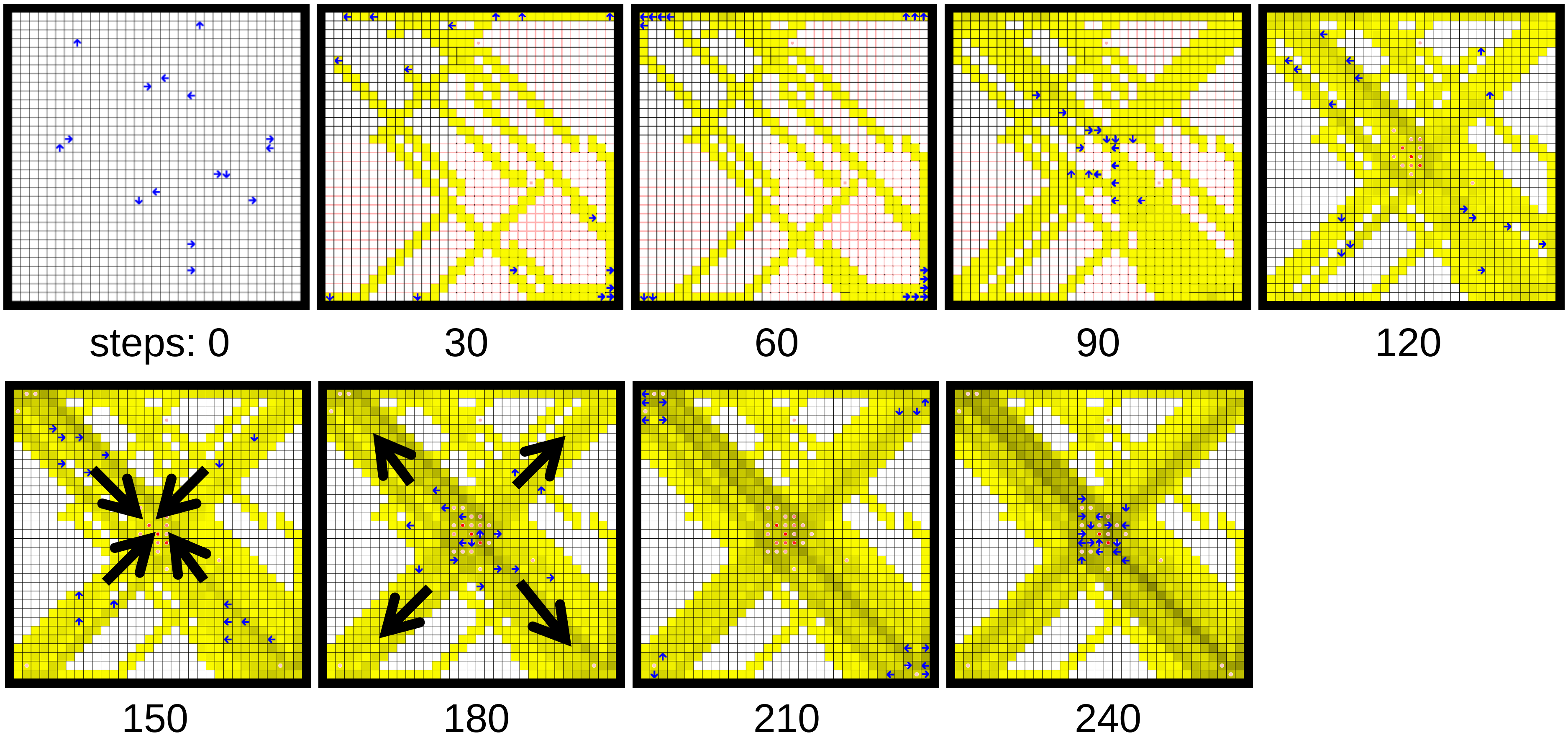


Specialists vs. Allrounders

- How do U and V behave as non-hybrid algorithms?
  - *cannot solve any of the environments*
- *X: "specialist" for subset A*
- *Y: "specialist" for subset B*
- *Z: "allrounder" for entire set*

- *two "specialists" time-shuffled XY_T are better than an "allrounder" Z*
- *two "allrounders" time-shuffled ZZ_T are better than an "allrounder" Z, but worse than XY_T*
- *Best combination comprises U and V that are **only** good in combination.*

Strategy of the Agents

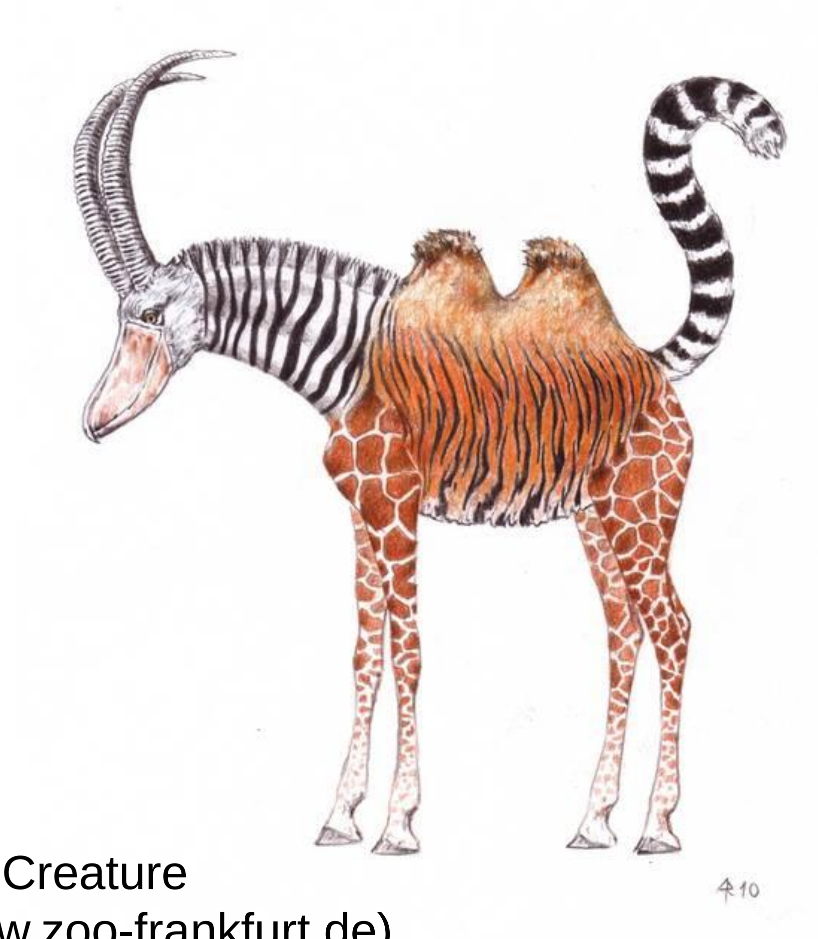


Conclusion and Future Work

- Conclusion
  - Hybrid algorithms were evolved for the All-to-all communication task with different methods.
  - Crossover technique *a* is better than *b*.
  - Directly evolving is more effective than separately evolving.
  - The computing time for evolution can be reduced by including the time-shuffling period in the genome.
- Future work
  - Optimize separately evolving (saves time by parallelizing) to produce good hybrid algorithms
  - Varying the complexity of the FSMs
  - Time-Shuffle more than 2 FSMs
  - Comparing with other Heuristics
  - Using Hardware-Support (FPGAs)

Thank you for your attention!

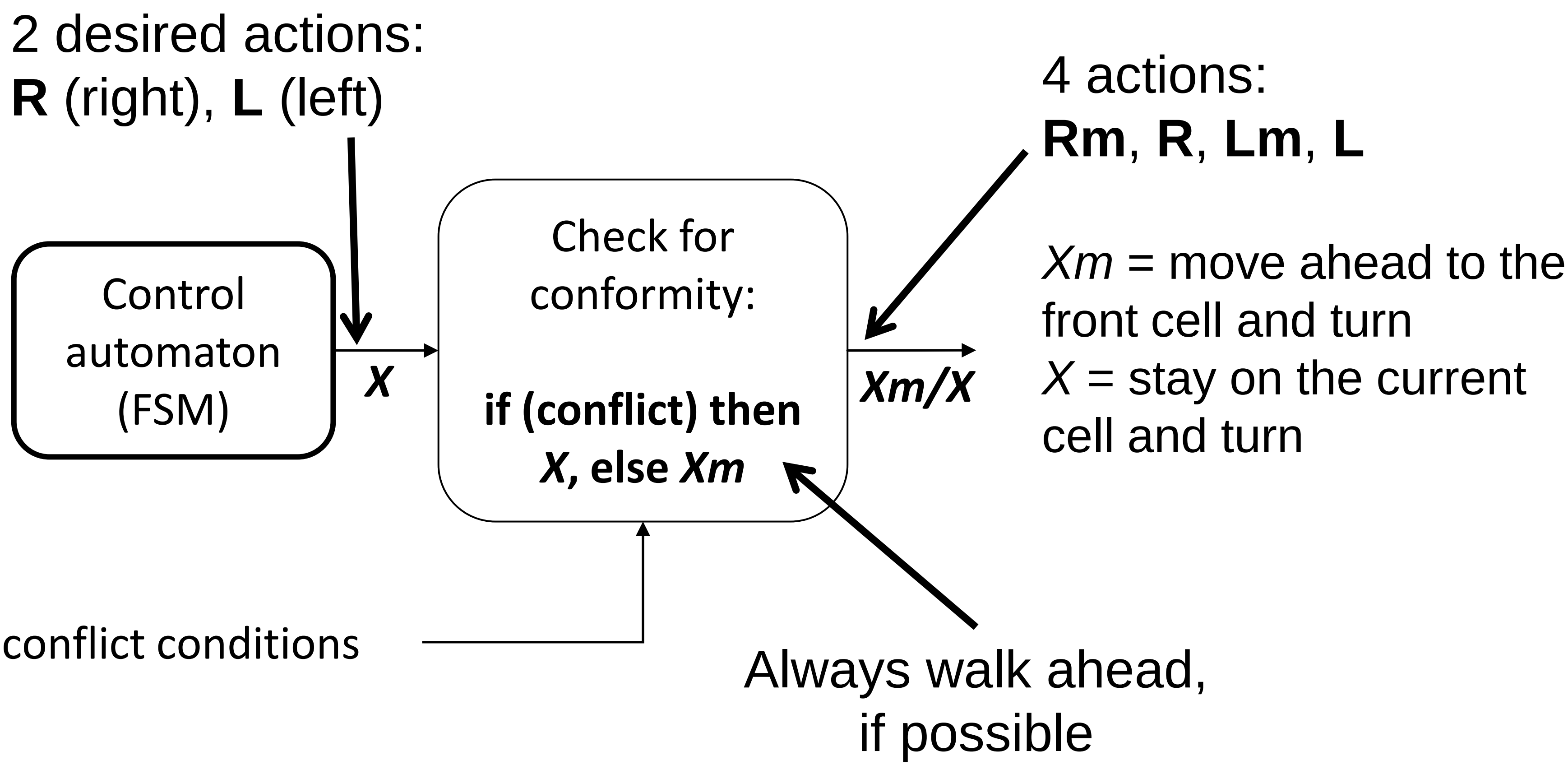
TECHNISCHE UNIVERSITÄT DARMSTADT



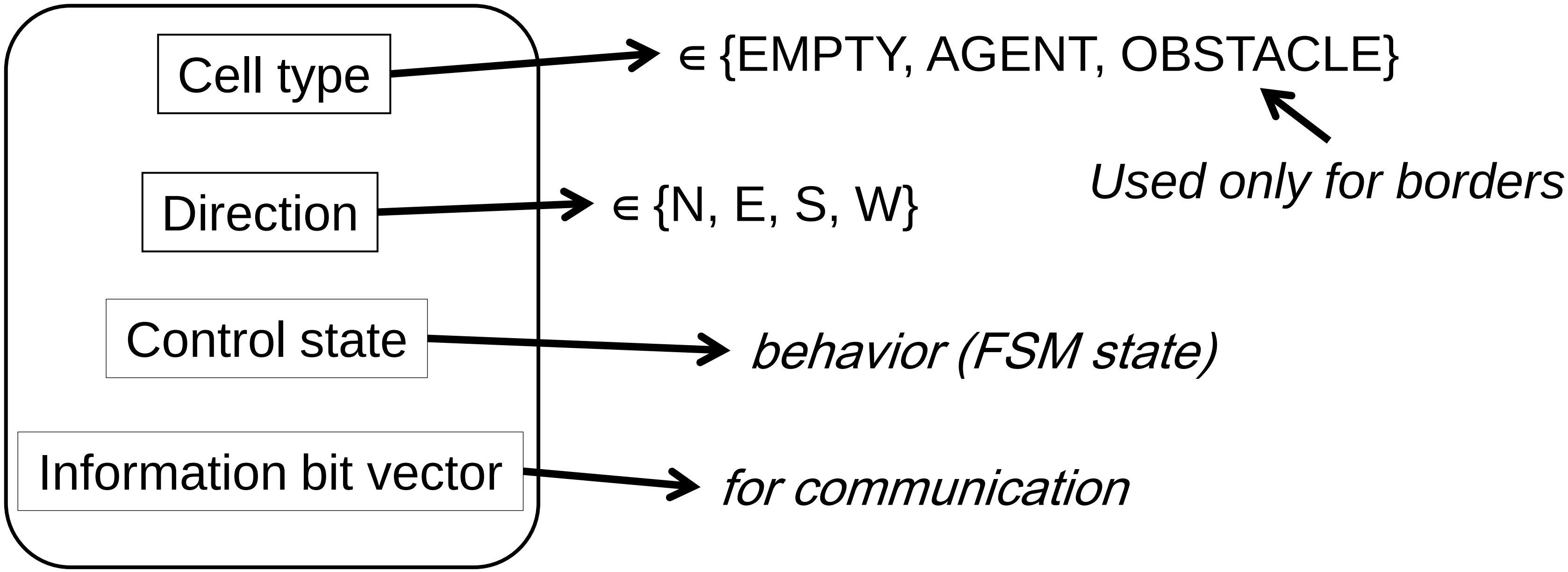
The Frankfurt Fabulous Creature
Image Source: Frankfurt Zoo (www.zoo-frankfurt.de)

APPENDIX

Modeling Agent Behavior



Cell State


Cell type
∈ {EMPTY, AGENT, OBSTACLE}
Used only for borders
Direction
∈ {N, E, S, W}
Control state
behavior (FSM state)
Information bit vector
for communication

Cellular Automata Model:
Cell Rule

If (cell type == EMPTY):
- Find neighboring AGENT with direction to “me”.
- If there is exactly one agent, copy agents’ *control state* and *direction* and perform FSM transition.
- Update *direction*, change cell type to AGENT.

If (cell type == AGENT):
- Detect possibility of movement (obstacle, conflicts).
- If possible, change cell type to EMPTY
- If not, perform FSM transition and update *direction*, *control state* and *information bit vector*.

If (cell type == OBSTACLE):
- Do nothing

Island Model GA:
Parameter Settings

P **=** ***7*** populations with **N = 100** individuals each

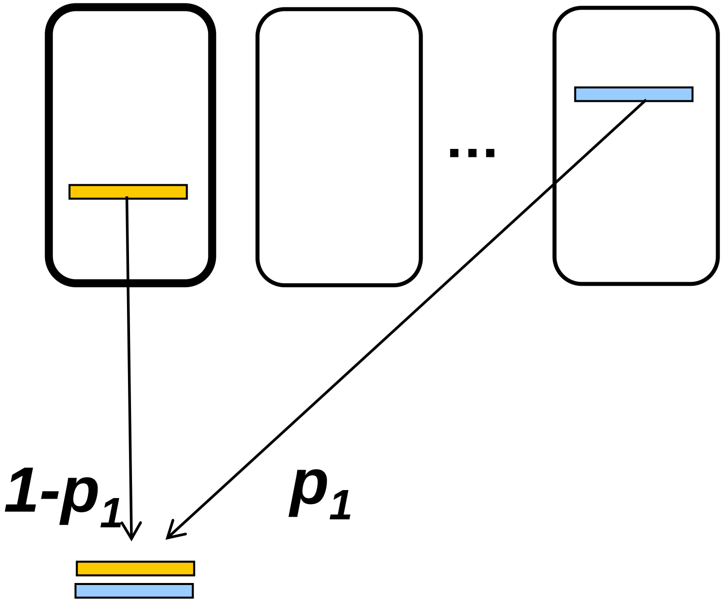
M **=** ***10*** offsprings

$\boldsymbol{p_1}$ **=** ***2%*** immigration rate (chosen from other population)
$\boldsymbol{1-p_1}$ **=** ***98%*** complement of immigration rate (chosen from own population)

$\boldsymbol{p_2}$ **=** ***9%*** mutation rate

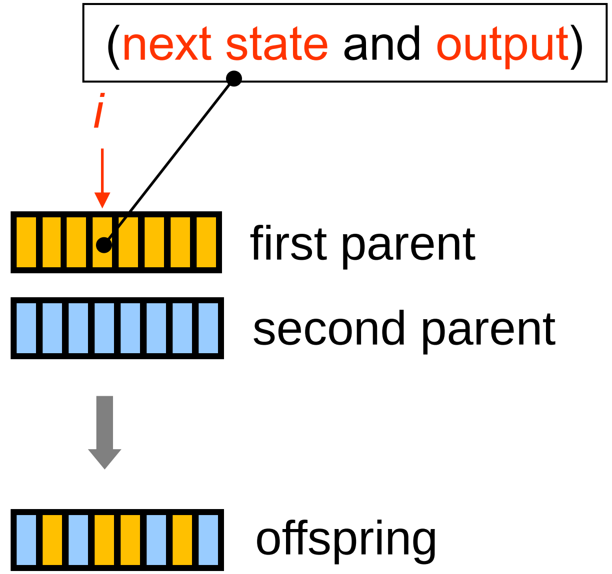
Parent Selection

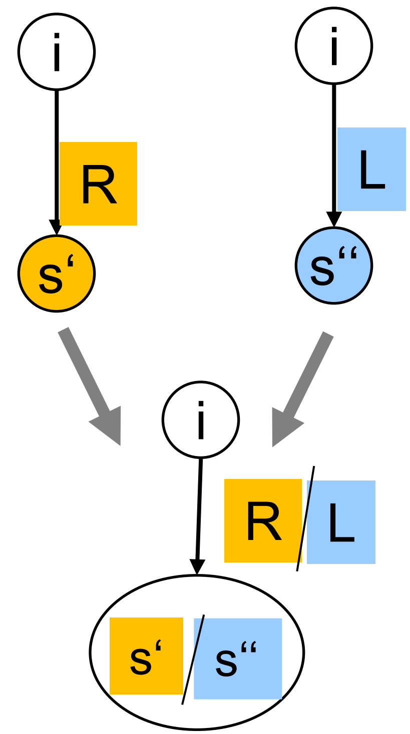
- Two parents are chosen for each population.
- First parent is chosen from the own population with a probability of ***(1-p$_1$)*** and second parent from an arbitrary other population with the probability of ***p$_1$*** ***(immigration rate)*** .



Island Model GA:
Uniform Crossover

- Each **new component *i*** of the genome string is taken from either the first parent or the second parent with a probability of 50%.
- Thereby **the next state and the output is taken from either parent at position *i*.**

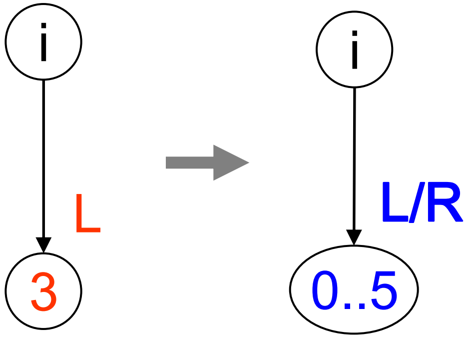




x	0						1					
s	0	1	2	3	4	5	0	1	2	3	4	5
s',y	1,1	5,0	3,0	4,1	5,1	3,0	1,0	2,1	3,1	4,0	5,1	0,0
action	R	L	L	R	R	L	Lm	Rm	Rm	Lm	Rm	Lm
i	0	1	2	3	4	5	6	7	8	9	10	11

Island Model GA:
Mutation

- The string being modified by the crossover is afterwards mutated with a probabiltity of $\boldsymbol{p_2}$.
- If a mutation shall be performed, an **arbitrary position *i*** is chosen and a new value (randomly choosen from the set of valid values) is replacing the existing one.
- Thereby **the next state and the output is randomly changed at position *i*.**



x	0						1					
s	0	1	2	3	4	5	0	1	2	3	4	5
s',y	1,1	5,0	3,0	4,1	5,1	**3,0**	1,0	2,1	3,1	4,0	5,1	0,0
action	R	L	L	R	R	**L**	Lm	Rm	Rm	Lm	Rm	Lm
i	0	1	2	3	4	5	6	7	8	9	10	11

Simulation Details

- 6 independend runs of the GA for each type of algorithm
- UV (per run):
  - 10,000 generations → 700,000 tested algorithms
  - 700,000·20 environments = 14,000,000 simulations
- XY (per run):
  - 10,000 generations → 700,000 tested algorithms
  - 700,000·10 environments = 7,000,000 simulations
  - 600·10·10 (2·Top10) = 60,000 Time-Shuffled algorithms
  - 60,000·20 environments = 1,200,000 simulations
- Z (per run):
  - 10,200 generations → 714,000 tested algorithms
  - 714,000·20 environments = 14,280,000 simulations

Best Fitness Values (I)

Type	t	F(avrg)	TOP1 Algorithm	F(TOP1)
Z	7.13h	626.5	3L4R3L4R0R2R-3R4L1L5L0R2R	605.6
XY_T	7.11h	554.2	X = 3R2R4L2R5L4L-3R0L1L5R0R3L Y = 2L3L1R4L1R3L-2L0R5R4L1L3R T = 377	497.3
UV_T-a	6.29h	424.1	U = 1L2L1R5L3L2R-4L3L5L5L3R3R V = 1L0L0L5R3R5R-1L4L1R2L1R1L T = 48	405.2
UV_T-b	6.52h	432.0	U = 2L5L0L4R3R1L-4L2L5R4R0R4L V = 3R4L5L1L2L1L-3R4L4L0L0R3R T = 40	407.6
U_TV_T-a	6.46h	**406.4**	U = 2R5L3R5L4R1R-1R5L0R2L2L1R V = 4R5L3L5R2R3R-2R5L1R5R3L1R $T_U = 60; T_V = 12$	369.4
U_TV_T-b	7.68h	420.0	U = 2R5L0L0L2L0R-5L5L1L4L3L1R V = 5L3R1L1R0R2R-4L2R1L1R1L3L $T_U = 60; T_V = 36$	**356.9**

6 independent runs of the GA for each type

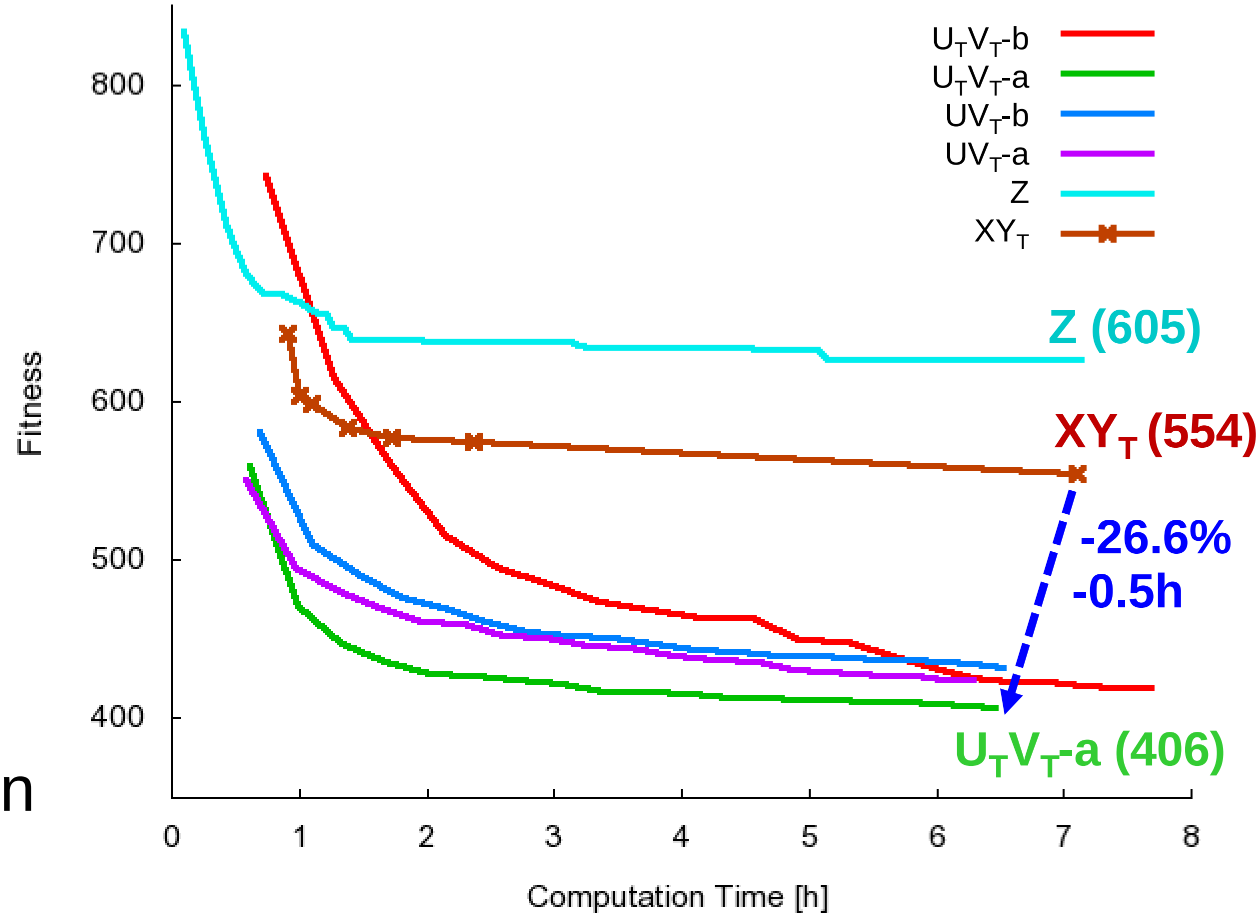
Directly evolving is more effective.

Best Fitness Values (II)

Averaged over the 6 runs

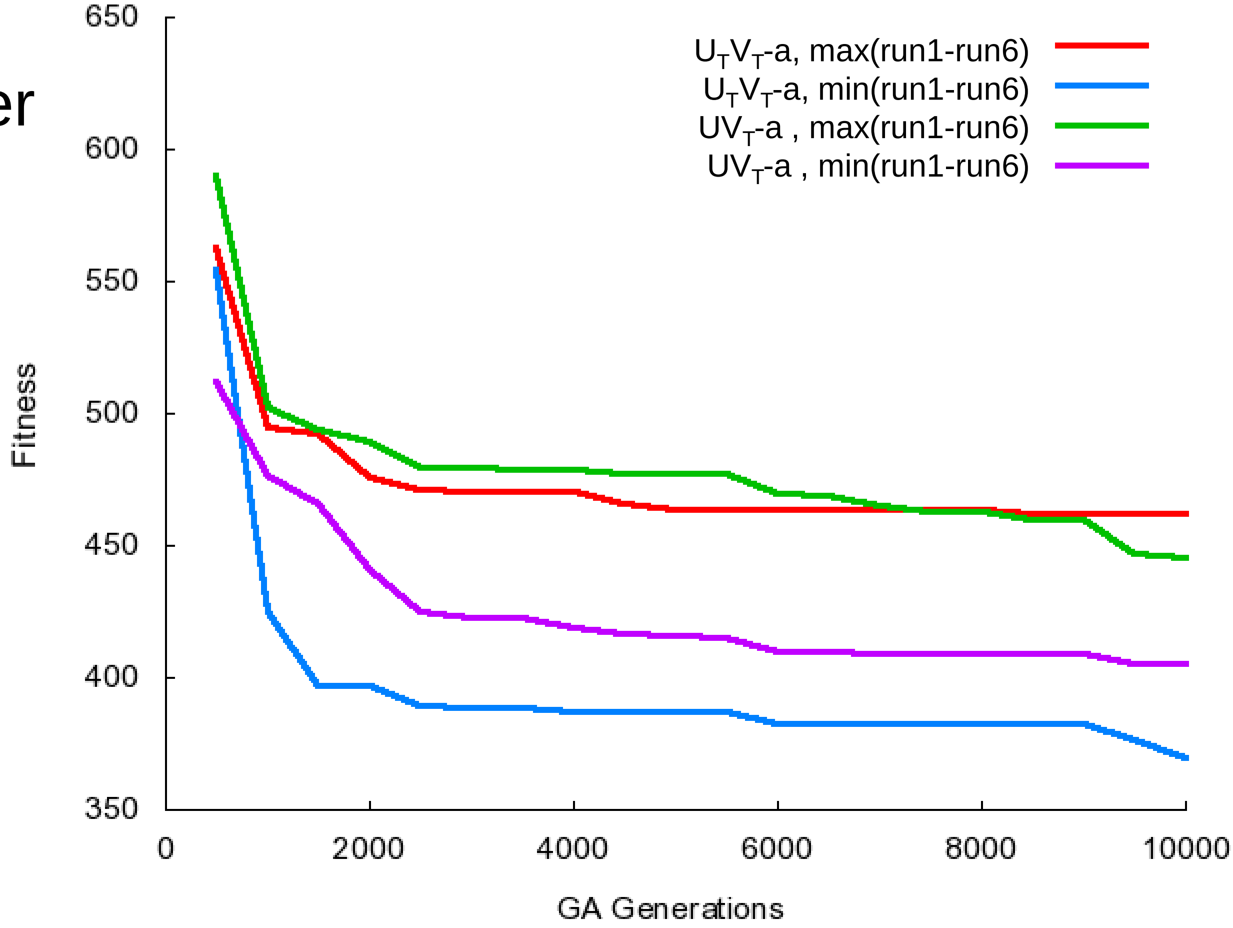
Directly evolving is also more efficient.

Crossover technique *a* is more efficient than technique b.



Variance of Best Fitness Values

GA with two periods finds better algorithms, but is also less reliable than GA with one period.



Robustness Test

(1) 80 environments with varying number of agents

- completely successful with same strategy
- needs longer for less agents

(2) 24 manually designed environments with obstacles

- successful for 19/24 environments
- difficulties when middle is blocked (only border) or narrow gaps exist

not solved

solved