How efficient are creatures with time-shuffled behaviors?

Patrick Ediger Rolf Hoffmann
Mathias Halbach

TU Darmstadt, FB Informatik, FG Rechnerarchitektur
Hochschulstraße 10, D-64289 Darmstadt
{ediger, hoffmann, halbach}@ra.informatik.tu-darmstadt.de

Abstract: The task of the creatures in the "creatures' exploration problem" is to visit all empty cells in an environment with a minimum number of steps. We have analyzed this multi agent problem with time-shuffled algorithms (behaviors) in the cellular automata model. Ten different "uniform" (non-time-shuffled) algorithms with good performance from former investigations were used alternating in time. We designed three time-shuffling types differing in the way how the algorithms are interweaved. New metrics were defined for such a multi agent system, like the absolute and relative efficiency. The efficiency relates the work of an agent system to the work of a reference system. A reference system is such a system that can solve the problem with the lowest number of creatures with uniform or time-shuffled algorithms. Some time-shuffled systems reached high efficiency rates, but the most efficient system was a uniform one with 32 creatures. Among the most efficient successful systems the uniform ones are dominant. Shuffling algorithms resulted in better success rates for one creature. But this is not always the case for more than one creature.

1 Introduction

The general goal of our project is to optimize the cooperative behavior of moving creatures in order to fulfill a certain global task in an artificial environment. A creature (another term: agent) behaves according to an algorithm which is stored in the creature.

We distinguish *uniform* and *time-shuffled* systems of creatures. A uniform system comprises creatures with one uniform behavior (algorithm) only whilst a time-shuffled system comprises creatures with generation-wise alternating behaviors. The goal of this investigation was to find out for the creatures' exploration problem (explained below), which algorithms "harmonize" best, meaning which combinations of algorithms with how many creatures are the most efficient. Different measures for efficiency were defined and used to compare the different systems. When we are speaking about efficiency you may think of cost (e. g., Euros) which you have to pay in total for the involved creatures to fulfill the task.

We are modeling the behavior by a finite state machine (Sec. 2). In the past we have tried to find out the best algorithm for one creature by enumeration. The number of state machines which can be coded using a state table is $M = (\#s\#y)^{(\#s\#x)}$ where $n = \#s$

is the number of states, $\#x$ is the number of different input states and $\#y$ is the number of different output actions. Note that M increases dramatically, especially with $\#s$, which makes it very difficult or even impossible to check the quality of all algorithms by enumeration in reasonable time. By hardware support (FPGA technology) we were able to simulate and evaluate all 12^{12} 6-state algorithms (including algorithms with less than 6 states and including redundant ones) for a test set of 5 initial configurations [HHB06]. The 10 best algorithms (with respect to percentage of visited cells) were used in further investigations to evaluate the robustness (using additional 21 environments) and the efficiency of $k > 1$ creatures. It turned out that more than one creature may solve the problem with less cost than a single one [HH07]. In our investigation we have concentrated on time-shuffled systems using the previously found algorithms. This time we are using 16 new environments compared to the environments used before. Now we use a field of fixed size 35 $\times$ 35 with a fixed number of obstacles (129). Thereby we are able to place the creatures at the beginning in regular formations and the number of obstacles becomes a constant which simplifies the analysis.

We have already started experiments using metaheuristics to optimize the algorithms. We are optimistic to find agent algorithms for more complex agent problems and we will use a cluster of FPGAs for that purpose to exploit the inherent parallelism in the metaheuristics. Our current results also give a partial answer to the question: If algorithms are combined, what are the expectation rates for good or bad combinations of them?

Modeling the behavior with a state machine with a restricted number of states and evaluation by enumerations was also undertaken in SOS [MSPPU02]. Additional work was done by these authors using genetic algorithms. The creatures' exploration problem based on our model was further investigated in [DL06]. Randomness was added which led to a higher degree of success. Our research in general is related to works like: Evolving optimal rules for cellular automata (CA) [Sip97, ST99], finding out the center of gravity by marching pixels [FKSL07, FS05, KMF07], evolving hardware [US06], using evolutionary algorithms and metaheuristics [Alb05].

The remainder of this paper is organized as follows. Sec. 2 describes how the problem is modeled in the CA model. Sec. 3 describes how well only uniform creatures under varying the environment solve the problem. Sec. 4 describes how efficient creatures with time-shuffled behavior can solve the problem.

2 CA model for the creatures' exploration problem

The problem is the following: p creatures are moving around in an environment that consists of empty cells and obstacle cells in order to visit all reachable empty cells in the shortest time. Creatures cannot move on obstacle cells, and only one creature can be located on an empty cell at the same time. Creatures can look forward one cell ahead which is in its moving direction. The creatures may perform four different actions: R (turn right only), L (turn left only), Rm (move forward and simultaneously turn right), Lm (move forward and simultaneously turn left).

If the "front cell" (the cell ahead) is not free, because it is an obstacle cell, another creature stands on it, or a collision conflict is anticipated, the action R or L is performed. In all other cases the action Lm or Rm is performed (Fig. 1). The detection of anticipated conflicts is realized by an arbitration signal from the destination cell. Each creature sends a request to its front cell, which sends back a grant signal if only one creature has sent a request [HHB06].

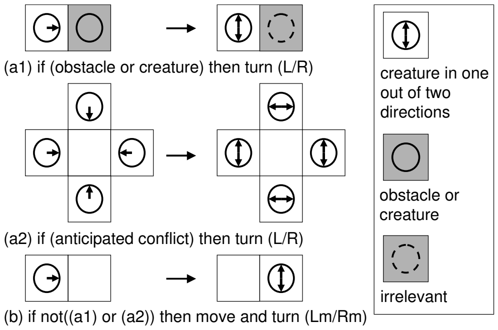
(a1) if (obstacle or creature) then turn (L/R)
(a2) if (anticipated conflict) then turn (L/R)
(b) if not((a1) or (a2)) then move and turn (Lm/Rm)
creature in one out of two directions
obstacle or creature
irrelevant

Figure 1: The conditions for the moving creatures' rules

The modeling of the behavior was done by implementing the rules with a state machine considered as a Mealy automaton with inputs (m, s), next state s' and output d (Fig. 2). An algorithm is defined by the contents of a state table assigned to the state machine. We are coding an algorithm into a string representation or a simplified string representation by concatenating the contents line by line to a string or a corresponding number, e. g.,

	1L2L0L4R5R3R-3Lm1Rm5Lm0Rm4Lm2Rm	*string representation*
=	1L2L0L4R5R3R-3L1R5L0R4L2R	*simplified string representation*

The state table can be represented more clearly as a state graph (Fig. 2). If the state machine uses n states, we call such an algorithm n-state algorithm. If the automaton is considered as a Moore automaton instead of a Mealy automaton, the number of states will be the product $n \times \#r$, where $\#r$ is the number of possible directions (4 in our case).

3 Uniform systems with one creature

In preceding investigations [HHB06] we could discover and evaluate the best 6-state algorithms for one creature by the aid of special hardware. The behavior of all relevant algorithms was simulated and evaluated for 26 initial test configurations (they are different from the ones we are using here). The following 10 best algorithms were ranked using a dominance relation with the criteria (1.) success, (2.) coverage and (3.) speed:

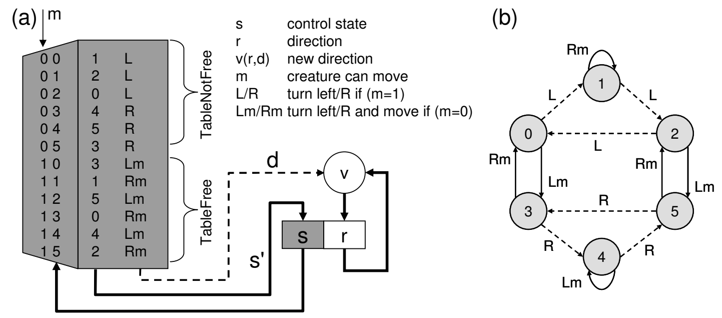

Figure 2: A state machine (a) models a creature's behavior. Corresponding 6-state algorithm (b)

1. G: 1L2L0L4R5R3R-3L1R5L0R4L2R
2. B: 1R2R0R4L5L3L-3R1L5R0L4R2L
3. C: 1R2R0R4L5L3L-3R4R2L0L1L5R
4. A: 0R2R3R4L5L1L-1R5R4R0L2L3L
5. D: 1R2R3R1L5L1L-1R0L2L4R3L1L
6. E: 1R2L0R4L5L3L-3R4R5R0L1L2R
7. F: 1R2L0L4R5R3R-3L4L5L0R1L2R
8. H: 1L2L3R4L2R0L-2L4L0R3L5L4R
9. I: 1L2L3L4L2R0L-2L4L0R3R5L4R
10. J: 1R2R3R0R4L5L-4R5R3L2L0L1L

The following definitions and metrics are used:

- k := number of creatures
- R := number of empty cells
- g := generation (time steps)
- $r(g)$: = number of visited cells in generation g
- $r_{\max}$:= the maximum number of cells which can be visited for $g \to \infty$
- $g_{\max}$:= the first generation in which $r_{\max}$ is achieved
- $e := r_{\max}/R[\%]$, the coverage or exploration rate, i. e. $\frac{\text{visited cells}}{\text{all empty cells}}$
- *speed* := $R/g_{\max}$ (only defined for successful algorithms)
- *step rate* := $1/$*speed* (the number of cells visited in one generation)

In order to find time-shuffled algorithms that are robust against changes of the environments, we have created a set of $I = 16$ environments which all contain 129 obstacle cells (Fig. 3). Then the algorithms A to J were simulated on them with one creature. The results show that none of the algorithms is capable of solving every environment successfully (Tab. 1). The best algorithms with respect to these environments are the algorithms B, G and J. Algorithm B has a *mean speed* (for $k = 1$ creature, see definition in Sec. 4) of 16.12$\%$ for the successful environments. The highest reachable mean speed would be 100$\%$ (visiting one new empty cell in each step). The *mean step rate*, which is the reciprocal of the mean speed, is 5.31 (every 5.31 steps an empty cell is visited). We can interpret the mean step rate as work or cost which has to be paid for a creature to visit a cell (e. g., 5.31€ per empty cell to visit, or to "clean" one "square meter", if you think of cleaning a room). Fig. 4 shows an example how algorithms may behave in principle. Algorithm B is successful for the environments 6, 10 and 16 and the speed is comparable (around 960 / 5400). Algorithm J is not successful for the environments 6 and 10.

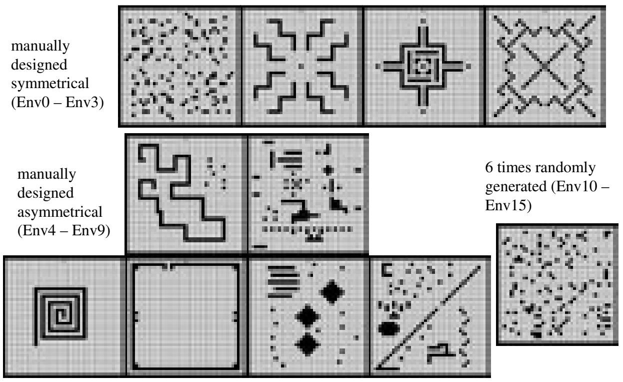
manually designed symmetrical (Env0 – Env3)
manually designed asymmetrical (Env4 – Env9)
6 times randomly generated (Env10 – Env15)

Figure 3: The 16 environments with 35×35 cells, manually designed or randomly generated. Each environment comprises $R = 960$ empty cells and 129 obstacles.

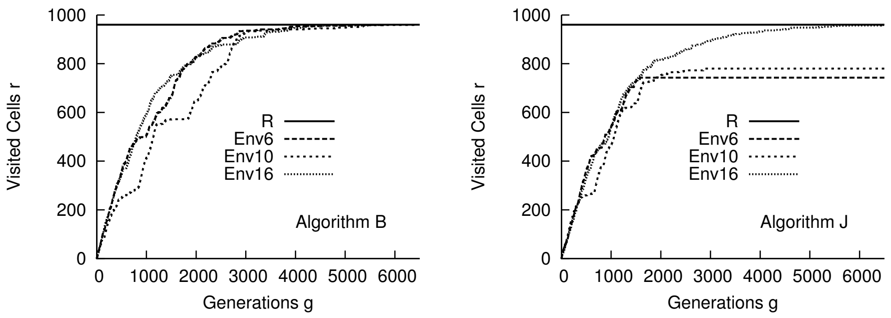

Figure 4: Curves showing the number of visited cells $r(g)$ (generations g on the x-axis, visited cells r on the y-axis) for the algorithms B and J for the environments Env6, Env10 and Env16. $R = 960$.

4 Multi creature and time-shuffled systems

In [HH07, HH06] we have evaluated that increasing the number of creatures can lead to synergy effects, i. e., the uniform creatures can work more efficiently together than by their own. 1 to 64 creatures were arranged on the cellular field symmetrically side by side at the borders (Fig. 5). The same distribution was used for the following investigation.

In order to investigate the performance of *time-shuffled systems* (multi agent system with time-shuffled algorithms), we have simulated all pairs (X, Y) of the former best single algorithms (A to J) on all 16 environments (Fig. 3) whereas the uniform pairs (X, X) are included for comparison. We used three different modes of time-shuffling the different algorithms (Fig. 6):

- *common* (c), each creature works with both algorithms alternating them generation-

Algorithm	Success Env0	Success Env1	Success Env2	Success Env3	Success Env4	Success Env5	Success Env6	Success Env7	Success Env8	Success Env9	Success Env10	Success Env11	Success Env12	Success Env13	Success Env14	Success Env15	T = No. Successful	total visiting percentage	mean g_max (successful)	mean r_max	mean speed (successful)	mean step rate
B	O	O	O	O	X	O	X	X	X	O	O	X	O	X	X	O	7	88,74%	5956	852	16,12%	5,31
G	O	O	O	O	X	O	X	X	X	O	O	X	O	X	X	O	7	87,32%	5998	838	16,01%	5,10
J	O	O	O	O	O	X	O	X	O	X	X	O	X	X	X	O	7	85,89%	6813	825	14,09%	5,78
C	X	O	O	O	X	O	X	X	O	O	O	O	O	X	X	O	6	86,92%	5626	834	17,06%	4,86
E	O	O	O	O	O	X	O	X	O	X	O	O	O	X	O	O	4	83,42%	6664	801	14,41%	5,20
A	O	O	X	X	O	O	O	O	O	O	O	O	O	O	O	X	3	82,38%	14297	791	6,71%	8,32
F	O	O	O	O	O	O	O	O	O	X	O	O	O	X	O	O	2	73,08%	6405	702	14,99%	4,60
D	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	0	47,55%	-	456	-	2,96
H	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	0	36,20%	-	348	-	4,69
I	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	0	36,20%	-	348	-	4,75

Table 1: Results of the simulation of one creature. The values are averaged over all environments, respectively over the T successfully visited environments in the case of "mean $g_{\max}$" and "mean *speed*". An X in the columns "Success Env$n$" indicates that the environment was successfully solved, an *O* that it was not.

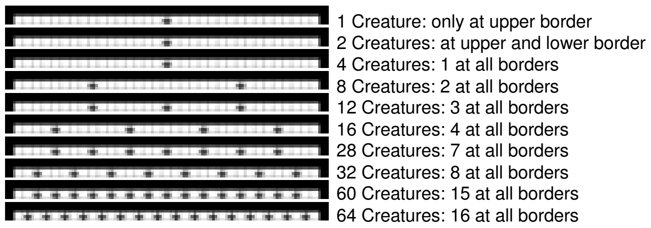
1 Creature: only at upper border
2 Creatures: at upper and lower border
4 Creatures: 1 at all borders
8 Creatures: 2 at all borders
12 Creatures: 3 at all borders
16 Creatures: 4 at all borders
28 Creatures: 7 at all borders
32 Creatures: 8 at all borders
60 Creatures: 15 at all borders
64 Creatures: 16 at all borders

Figure 5: The arrangement of creatures in a multi creature system.

wise (odd/even). But it has only one common state s which is shared by the algorithms.

- *simultaneous* (s), each creature works with both algorithms simultaneously. Each of the algorithm has its own state (s_x, s_y). The output is taken alternating from X/Y for $t = 0/1$. Both states are updated in each generation.

- *alternate* (a), an individual state is used for each algorithm. For $t = 0$ the output is taken from X and only the state s_x is updated. For $t = 1$ the output is taken from Y and only the state s_y is updated.

The shuffling modes c and s are identical for all "pairs" (X, X). In that case they are in fact uniform systems.

One creature time-shuffled. The results (Tab. 2) show that one creature with time-shuffling can solve more environments successfully than the uniform systems. With the alternate shuffling up to 12 environments can be solved (B, J), 11 with the simultaneous variant (A, G) and the best pair with the common time-shuffling is (G, F).

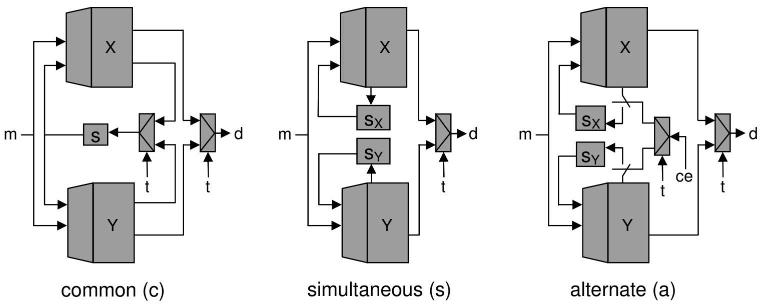

Figure 6: Types of time-shuffling of creature state machines. X is the first algorithm, Y the second. $t = 0/1$ is a function of the generation counter and controls the outputs and in the cases c and a also the state transition. ce (clock enable) is a signal that enables the state transition.

Algorithm X	Algorithm Y	Shuffling	S1	S2	S3	S4	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	T = No. Successful	total visiting percentage	mean speed (successful)
G	F	c	X	O	O	O	X	O	O	O	X	O	X	O	X	X	X	X	8	85,44%	9,51%
E	C	c	X	O	O	O	X	O	O	O	X	O	X	X	X	X	O	X	8	82,50%	9,61%
C	B	c	O	O	O	O	X	O	X	O	X	X	X	O	X	O	X	X	8	71,50%	11,54%
A	**G**	**s**	**O**	**X**	**O**	**X**	**X**	**X**	**X**	**O**	**X**	**O**	**X**	**O**	**X**	**X**	**X**	**X**	**11**	**96,33%**	**3,82%**
J	B	s	O	X	O	O	X	O	X	O	O	X	X	O	X	X	X	X	9	92,42%	5,07%
B	J	s	O	X	O	O	X	O	X	O	X	X	X	O	X	X	O	X	9	85,61%	6,01%
B	**J**	**a**	**X**	**X**	**X**	**O**	**X**	**O**	**X**	**O**	**X**	**O**	**X**	**X**	**X**	**X**	**X**	**X**	**12**	**85,84%**	**5,79%**
B	**A**	**a**	**X**	**O**	**O**	**X**	**X**	**O**	**X**	**O**	**X**	**X**	**X**	**X**	**X**	**O**	**X**	**X**	**11**	**99,38%**	**4,48%**
C	A	a	X	X	O	X	X	O	X	X	X	O	X	X	X	O	X	O	11	95,24%	4,90%

Table 2: The three pairs of algorithms for each kind of shuffling that have the most successes (1.) and the highest total visiting percentage (2.) with one creature. The overall most successful algorithm pairs are shown in bold letters.

In the following we will denote a system in the way *XYp-k*, where *XY* is the pair of algorithms, k the number of agents and p the mode of time-shuffling (optional). E. g., ABa-8 is a system with 8 creatures using an alternate time-shuffling of the algorithms (A,B).

Successful systems, depending on the number of creatures. By increasing the number of creatures the success rate also increases (Tab. 3). At least 8 creatures are necessary to visit successfully all environments. The systems J-8 and nine other time-shuffled systems with 8 creatures are successful. With 64 creatures 9 out of 10 uniform systems (B-64 to J-64) and 58 out of 280 time-shuffled systems are successful. In total 41/100 uniform and 185/2800 time-shuffled systems are successful for all 16 environments.

Metrics for time-shuffled systems with more than one creature. To be able to evaluate and compare time-shuffled systems (including uniform systems) with a different number of creatures we have defined additional metrics:

NO. OF CREATURES	SUCCESSFUL (UNIFORM ONLY)	NO. OF SUCCESSFUL SYSTEMS / NO. OF COMBINATIONS (TIME-SHUFFLED)			
		c+s+a	common (c)	simultaneous (s)	alternate (a)
1	-	0/280	0/90	0/90	0/100
2	-	0/280	0/90	0/90	0/100
4	-	0/280	0/90	0/90	0/100
8	J	9/280	1/90	3/90	5/100
12	B,C,J	9/280	1/90	5/90	3/100
16	B,C,E,G,J	16/280	2/90	7/90	7/100
28	B,C,D,E,I,J	25/280	6/90	8/90	11/100
32	C,D,E,F,G,H,I,J	23/280	7/90	6/90	10/100
60	B,C,D,E,F,G,H,I,J	45/280	12/90	14/90	19/100
64	B,C,D,E,F,G,H,I,J	58/280	13/90	20/90	24/100
total	41/100	185/2800	43/900	63/900	79/1000

Table 3: Percentage of algorithm pairs that solve successfully all 16 environments

- *mean speed per creature* $= ms(k) = \frac{\sum r_{\max,i}}{k \cdot \sum g_{\max,i}}$. The speed $ms(k)$ is an average over all environments i and is related to one creature. This measure expresses how fast a creature can visit a cell on average (maximum is 100%). This measure should not be used if any environment can not be successfully visited because then the mean speed might be believed higher than reasonable.

- *mean normalized work* $= mw(k) = \frac{k \cdot \sum g_{\max,i}}{\sum r_{\max,i}} = \frac{1}{ms(k)}$. This value represents the work which is necessary, or the costs one has to pay for one creature to visit a cell.

- *relative efficiency* $= \frac{ms(XY\text{-}k)}{ms(XY\text{-}k_{min})}$. First a reference system *XY*-$k_{\min}$ has to be found which can solve the problem with the lowest number $k_{\min}$ of creatures. The relative efficiency relates the mean speed of the system *XY*-k to the mean speed of the reference system *XY*-$k_{\min}$. This measure compares the costs of the reference system *XY*-$k_{\min}$ with the cost of the system *XY*-k. If the relative efficiency is higher than one, the work can be done cheaper with the system *XY*-k. Two similar measures can be defined if the uniform reference system *X*-$k_{\min}$ or the uniform reference system *Y*-$k_{\min}$ is chosen instead of *XY*-$k_{\min}$.

- *absolute efficiency* $= \frac{ms(XY\text{-}k)}{ms(UV\text{-}k_{min})}$. In distinction to the relative efficiency another reference algorithm pair is used. The reference is the fastest of any algorithm pair *UV* (including the uniform “pairs” *UU*) which can solve the problem with a minimum number of creatures. A similar measure can be defined if the fastest of any uniform reference systems *U*-$k_{\min}$ is chosen instead of *UV*-$k_{\min}$.

The efficiency measures are only defined if a $k_{\min}$ exists and if the system *XY*-k solves all 16 environments successfully. We have assumed for the reference algorithm the common time-shuffling mode because it is the least complex (only one state register).

The fastest systems. In the following evaluations we will only take into account those systems that were successful on all environments. The system BCc-64 is the fastest, needing only 218 generations on average to solve the problem (Tab. 4). 8 of the 10 fastest pairs are uniform systems.

No. of Creatures (k)	Algorithm X	Algorithm Y	Shuffling	mean g_max	mean speed per creature	mean normalized work	absolute efficiency compared to J8
64	**B**	**C**	**c**	**218**	**6,89%**	**14,51 €**	**1,038**
60	J	J	u	224	7,13%	14,03 €	1,073
60	**B**	**C**	**c**	**233**	**6,86%**	**14,58 €**	**1,033**
60	G	G	u	245	6,52%	15,34 €	0,982
64	J	J	u	257	5,83%	17,15 €	0,878
60	C	C	u	257	6,22%	16,09 €	0,936
64	B	B	u	261	5,76%	17,38 €	0,867
60	B	B	u	276	5,80%	17,25 €	0,873
64	C	C	u	285	5,27%	18,99 €	0,793
64	G	G	u	314	4,78%	20,92 €	0,720

Table 4: The 10 absolute fastest systems (sorted ascending by "mean $g_{\max}$") which solve all 16 environments. "u" in shuffling means that these are actually uniform systems.

The most efficient systems. When looking at Tab. 5 you will notice that the most efficient (absolute efficiency) system J-32 is uniform with the efficiency of 1.184. Furthermore 7 of the top ten (absolute efficiency) are uniform. The reference system is J-8 for all the systems. The three time-shuffled systems with efficiencies higher than one are BCc-28/64/60. We can conclude that only common time-shuffling led to efficiencies higher than one but not in an amount that was expected when we had started this investigation. Time-shuffled systems with one creature behaved much better than uniform systems with one creature, but it cannot be deduced that time-shuffled systems always behave better than uniform systems.

Considering only successful systems, 7 out of 41 uniform systems and 3 out of 185 time-shuffled systems reach an absolute efficiency higher than one. These results give a hint on what will happen when good agents are randomly combined through time-shuffling, e. g., during genetic methods: only a low percentage will be better than their "parents".

Tab. 5 includes also values for the relative efficiencies which can get higher than the absolute efficiencies, e. g., the relative efficiency of BCc-28 compared to B-12 is 1.881.

We have also counted the different types of conflicts (Tab. 5): (a) static obstacle is on the front cell, (b) another creature is on the front cell (dynamic obstacle), (c) anticipated conflicts (2, 3 or 4 creatures want to visit the same cell). At the moment it is not clear, whether there is a significant relation between the efficiency and the types and frequency of the conflicts.

The overproportionate gain of efficiency for parallel systems with more than one creature can be interpreted as "synergy". We presume this effect is due to many reasons, like: Creatures react individually to their local environment and can be in different states, additional communication implicitely exists by the conflict anticipation mechanism and creatures can start at different positions which can be an advantage.

No. of Creatures (k)	Algorithm X	Algorithm Y	Shuffling	mean g_max	mean speed per creature	mean normalized work	mean conflicts (c)	mean conflicts (b)	mean conflicts (a)	mean conflicts per creature	conflicts per visit	relative efficiency (uniform X)	relative efficiency (uniform Y)	absolute efficiency compared to J-8	realtive efficiency (common XY)	k_min X	k_min Y	k_min XY
32	J	J	u	382	7,86%	12,72 €	8	7	60	74	0,08	1,184	1,184	1,184	3,113	8	8	8
28	**B**	**C**	**c**	**437**	**7,85%**	**12,74 €**	**11**	**6**	**70**	**88**	**0,09**	**1,881**	**1,527**	**1,182**	**4,697**	**12**	**12**	**28**
16	J	J	u	768	7,81%	12,80 €	8	6	126	140	0,15	1,176	1,176	1,176	3,092	8	8	8
28	B	B	u	456	7,52%	13,30 €	9	8	68	84	0,09	1,803	1,803	1,132	4,044	12	12	12
12	J	J	u	1069	7,48%	13,36 €	8	7	155	170	0,18	1,127	1,127	1,127	2,963	8	8	8
28	J	J	u	463	7,41%	13,49 €	8	8	69	86	0,09	1,116	1,116	1,116	2,934	8	8	8
28	C	C	u	475	7,22%	13,85 €	9	8	72	90	0,09	1,404	1,404	1,087	1,664	12	12	12
60	J	J	u	224	7,13%	14,03 €	9	10	35	54	0,06	1,073	1,073	1,073	2,822	8	8	8
64	**B**	**C**	**c**	**218**	**6,89%**	**14,51 €**	**10**	**9**	**35**	**54**	**0,06**	**1,652**	**1,341**	**1,038**	**4,126**	**12**	**12**	**28**
60	**B**	**C**	**c**	**233**	**6,86%**	**14,58 €**	**9**	**10**	**36**	**55**	**0,06**	**1,644**	**1,334**	**1,033**	**4,105**	**12**	**12**	**28**
8	*J*	*J*	*u*	*1807*	*6,64%*	*15,06 €*	*10*	*6*	*262*	*278*	*0,29*	*1,000*	*1,000*	*1,000*	*2,629*	*8*	*8*	*8*

Table 5: The top 10 most absolute efficient (lowest total costs) non-uniform systems. Constraint: all algorithm pairs are successful in all 16 environments (visiting percentage = 100%). "u" in shuffling means that these are actually uniform systems.

5 Conclusion

The creatures' exploration problem was investigated in the CA model for multiple creatures using time-shuffled combinations of 10 algorithms (behaviors). These algorithms had shown a good performance in former investigations. The analysis was performed for 16 new environments of size 35×35 and 129 obstacles each. New metrics have been defined for such multi creature systems, especially the mean speed, the relative efficiency (comparing the work of a system with an algorithmic similar system using the minimum number of creatures which can solve the problem), and the absolute efficiency (comparing the work of a system with an algorithmic potentially different system using the minimum number of creatures which can solve the problem). A single creature is not successful for all environments. One time-shuffled creature was more successful but still could not visit all environments successfully. The problem could only be solved using at least 8 creatures for the uniform system J-8 and nine other time-shuffled systems. 185 time-shuffled systems out of all 2800 time-shuffled combinations were successful. The overall fastest system is the time-shuffled system BJc-64, but it is not the most efficient. The most efficient system is uniform: J-32. It turned out that the system BCc-28 (28 creatures, algorithms B and C, time-shuffled with a common state) is 18% more efficient than the uniform reference system J-8. Under the top ten most efficient systems are 3 time-shuffled and 7 uniform ones.

Our future work is directed to investigate other methods of time-shuffling or of combining different algorithms. It is also promising to compare creatures which are different in space (non-uniform), study the relation between conflicts and efficiency, and optimizing the behavior through heuristics.

References

[Alb05] Enrique Alba. *Parallel Metaheuristics: A New Class of Algorithms*. John Wiley & Sons, NJ, USA, August 2005.

[DL06] Bruno N. Di Stefano and Anna T. Lawniczak. Autonomous Roving Object's Coverage of its Universe. In *CCECE*, pages 1591–1594. IEEE, 2006.

[FKSL07] Dietmar Fey, Marcus Komann, Frank Schurz, and Andreas Loos. An Organic Computing architecture for visual microprocessors based on Marching Pixels. In *ISCAS*, pages 2686–2689. IEEE, 2007.

[FS05] Dietmar Fey and Daniel Schmidt. Marching-pixels: a new organic computing paradigm for smart sensor processor arrays. In Nader Bagherzadeh, Mateo Valero, and Alex Ramírez, editors, *Conf. Computing Frontiers*, pages 1–9. ACM, 2005.

[HH06] Rolf Hoffmann and Mathias Halbach. Are Several Creatures More Efficient Than a Single One? In Samira El Yacoubi, Bastien Chopard, and Stefania Bandini, editors, *ACRI*, volume 4173 of *Lecture Notes in Computer Science*, pages 707–711. Springer, 2006.

[HH07] Mathias Halbach and Rolf Hoffmann. Solving the Exploration's Problem with Several Creatures More Efficiently. In Roberto Moreno-Díaz, Franz Pichler, and Alexis Quesada-Arencibia, editors, *EUROCAST*, volume 4739 of *Lecture Notes in Computer Science*, pages 596–603. Springer, 2007.

[HHB06] Mathias Halbach, Rolf Hoffmann, and Lars Both. Optimal 6-State Algorithms for the Behavior of Several Moving Creatures. In Samira El Yacoubi, Bastien Chopard, and Stefania Bandini, editors, *ACRI*, volume 4173 of *Lecture Notes in Computer Science*, pages 571–581. Springer, 2006.

[KMF07] Marcus Komann, Andreas Mainka, and Dietmar Fey. Comparison of Evolving Uniform, Non-uniform Cellular Automaton, and Genetic Programming for Centroid Detection with Hardware Agents. In Victor E. Malyshkin, editor, *PaCT*, volume 4671 of *Lecture Notes in Computer Science*, pages 432–441. Springer, 2007.

[MSPPU02] Bertrand Mesot, Eduardo Sanchez, Carlos-Andres Peña, and Andres Perez-Uribe. SOS++: Finding Smart Behaviors Using Learning and Evolution. In R. Standish, M. Bedau, and H. Abbass, editors, *Artificial Life VIII: The 8th International Conference on Artificial Life*, pages 264–273, Cambridge, Massachusetts, 2002. MIT Press.

[Sip97] Moshe Sipper. *Evolution of Parallel Cellular Machines, The Cellular Programming Approach*, volume 1194 of *Lecture Notes in Computer Science*. Springer, 1997.

[ST99] Moshe Sipper and Marco Tomassini. Computation in artificially evolved, non-uniform cellular automata. *Theor. Comput. Sci.*, 217(1):81–98, 1999.

[US06] Andres Upegui and Eduardo Sanchez. On-chip and on-line self-reconfigurable adaptable platform: the non-uniform cellular automata case. In *IPDPS*. IEEE, 2006.