Is a non-uniform system of creatures more efficient than a uniform one?

NIDISC, April 14, 2008

Patrick Ediger, Rolf Hoffmann and Mathias Halbach

- The Creatures‘ Exploration Problem
- Action Set of Creatures (Rules)
- CA Model for Creature Systems
- Modeling of the Creatures‘ Behavior

- Optimization Strategies
- Success Rates
- Non-uniform Multi Creature Systems

- Conclusion
The Creatures‘ Exploration Problem

- **The Problem**
  - Given is a 2D-CA with obstacles and moving creatures.
  - **Goal:** Find an optimal local rule for the creatures to visit a maximum number of empty cells with a minimum number of time steps for a given set of initial configurations.

- **Applications**
  - Mowing a lawn in shortest time
  - Vacuum cleaning a room
  - Exploring an unknown environment
  - Distribute „information“
Action Set of Creatures (Rules)

- Creature has 4 directions (N,E,S,W)
- Creature can perform 4 actions
  - Turn Right, Turn Left (R,L)
  - Turn Right or Left and simultaneously move forward (Rm, Lm)
- System has 3 rules
  - a) several creatures point to the same cell → L oder R
  - b) creature points to an obstacle or another creature → L oder R
  - c) else Lm oder Rm
CA Model for Creature Systems

- Defining the environment
- Different states of cells

```
  cell
  /   \
empty obstacle creature ...

  additional states
  (e.g. direction, information)
```
Modeling the Creatures' Behavior

- Rule a) needs arbitration logic (within a CA-Generation creatures send a Request at the target cell, whose asynchronous feedback logic sends back a Grant if only one creature requests.
- Decision between Lm, Rm, L and R (obeying the rules) with state machines

**e.g.: 6-state algorithm**

<table>
<thead>
<tr>
<th>m (grant signal)</th>
<th>0 0</th>
<th>0 1</th>
<th>0 2</th>
<th>0 3</th>
<th>0 4</th>
<th>0 5</th>
<th>1 0</th>
<th>1 1</th>
<th>1 2</th>
<th>1 3</th>
<th>1 4</th>
<th>1 5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>Lm</td>
<td>Rm</td>
<td>Lm</td>
<td>Rm</td>
<td>Lm</td>
<td>Rm</td>
</tr>
</tbody>
</table>

- action d
  - blocked, m=0
  - free (move), m=1
Optimization Strategies

- How can we develop good local algorithms for creatures with a very restricted intelligence, that have to fulfill a global task?

- Enumerate all possible state machines?
  
  \# Algorithms = (#s \#y) (#s \#x) → only for few states

  [\#s = no. of states, \#x = no. of input values, \#y = no. of actions]

- Evolve state machines with heuristic methods
  
  - What is a good heuristic method?
  
  - Can we derive a good 12-state algorithm from two good 6-state algorithms?
  
  - Do creatures with distinct behaviors harmonize in the same environment?
Environment Test Set

- 16 environments
  - 33x33 cells with 129 obstacles and $R = 960$ empty cells
    - **Manually** designed symmetrical:
      - ![Symmetrical Environments](image)
    - **Manually** designed asymmetrical:
      - ![Asymmetrical Environments](image)
    - **Randomly** generated:
      - ![Randomly Generated Environments](image)
## Success Rates of the best 6-state-Algorithms for one Creature

- **Success rate**: number of successful (100%) visited environments
- used: 10 best algorithms A to J from former investigation

### Table: Success Rates

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Success Env0</th>
<th>Success Env1</th>
<th>Success Env2</th>
<th>Success Env3</th>
<th>Success Env4</th>
<th>Success Env5</th>
<th>Success Env6</th>
<th>Success Env7</th>
<th>Success Env8</th>
<th>Success Env9</th>
<th>Success Env10</th>
<th>Success Env11</th>
<th>Success Env12</th>
<th>Success Env13</th>
<th>Success Env14</th>
<th>Success Env15</th>
<th>No. Successful</th>
<th>total visiting percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>O O O O O X O X X X O O X O X X O</td>
<td>7</td>
<td>88.74%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>O O O O O X O X X X O O X O X X O</td>
<td>7</td>
<td>87.32%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>J</td>
<td>O O O O O X O X O X X O X X X O</td>
<td>7</td>
<td>85.89%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>X O O O O X X X O O O O X X O</td>
<td>6</td>
<td>86.92%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>O O O O O X X X O X O O X O</td>
<td>4</td>
<td>83.42%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>O O X X O O O O O O O O O O X X</td>
<td>3</td>
<td>82.38%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>O O O O O O O X O O X O O X O</td>
<td>2</td>
<td>73.08%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>O O O O O O O O O O O O O O O O</td>
<td>0</td>
<td>47.55%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H</td>
<td>O O O O O O O O O O O O O O O</td>
<td>0</td>
<td>36.20%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I</td>
<td>O O O O O O O O O O O O O O O</td>
<td>0</td>
<td>36.20%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Mean over all environments.
Non-uniform Multi Creature Systems

- Creatures: \( k = 1 .. 64 \), equally distributed along the borders (powers of 2, plus 12, 28 and 60)
- All pairs of algorithms A-J used with all placing modes
- Notation of systems:
  - **Uniform**: \( X-k \)
    - \( X \): algorithm
    - \( k \): number of creatures
  - **Non-uniform**: \( XYp-k \)
    - \( X/Y \): algorithms
    - \( p \): placing mode (c, s, a)
    - \( k \): number of creatures
Non-uniform Multi Creature Systems

- **No. of Completely successful systems** (successful on all 16 environments)

<table>
<thead>
<tr>
<th>k</th>
<th>uniform</th>
<th>corner</th>
<th>side</th>
<th>alternate</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>-</td>
<td>0/10</td>
<td>3/100</td>
<td>0/10</td>
</tr>
<tr>
<td>4</td>
<td>-</td>
<td>0/10</td>
<td>19/100</td>
<td>27/100</td>
</tr>
<tr>
<td>8</td>
<td>J</td>
<td>1/10</td>
<td>22/100</td>
<td>71/100</td>
</tr>
<tr>
<td>12</td>
<td>B,C,J</td>
<td>3/10</td>
<td>76/100</td>
<td>85/100</td>
</tr>
<tr>
<td>16</td>
<td>B,C,E,G,J</td>
<td>5/10</td>
<td>65/100</td>
<td>91/100</td>
</tr>
<tr>
<td>28</td>
<td>B,C,D,E,I,J</td>
<td>6/10</td>
<td>93/100</td>
<td>94/100</td>
</tr>
<tr>
<td>32</td>
<td>C,D,E,F,G,H,I,J</td>
<td>8/10</td>
<td>91/100</td>
<td>95/100</td>
</tr>
<tr>
<td>60</td>
<td>B,C,D,E,F,G,H,I,J</td>
<td>9/10</td>
<td>97/100</td>
<td>97/100</td>
</tr>
<tr>
<td>64</td>
<td>B,C,D,E,F,G,H,I,J</td>
<td>9/10</td>
<td>97/100</td>
<td>97/100</td>
</tr>
<tr>
<td>Sum</td>
<td></td>
<td>41/90</td>
<td>660/900</td>
<td>615/810</td>
</tr>
</tbody>
</table>
Metrics for Multi Creature Systems

- Only **completely successful** systems will be compared.
- **step rate** := the number of generations $g_{\text{max}}$ needed to visit one new empty cell with $k$ creatures (averaged over all environments).
  
  \[
  \text{step rate} := \frac{g_{\text{max}}}{R} \quad \text{empty cells}
  \]

- **normalized work** := average steps needed by all $k$ creatures to visit one new empty cell (cost for each visited cell).
  
  \[
  \text{normalized work} = \text{nw} := k \times \text{step rate}
  \]

- **absolute efficiency** := the cost compared to a reference system $XYP_{k_{\text{min}}}$.
  
  \[
  \text{absolute efficiency} := \frac{\text{nw}_{XYP_{k_{\text{min}}}}}{\text{nw}}
  \]

  1. find completely succ. system with $k_{\text{min}}$
  2. use most efficient uniform system
The most efficient Multi Creature Systems

<table>
<thead>
<tr>
<th>No. of Creatures (k)</th>
<th>Algorithm X</th>
<th>Algorithm Y</th>
<th>Placement</th>
<th>mean g_max</th>
<th>mean normalized work</th>
<th>absolute efficiency compared to J-8</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>C</td>
<td>G</td>
<td>a</td>
<td>339</td>
<td>11.31 €</td>
<td>1.332</td>
</tr>
<tr>
<td>28</td>
<td>J</td>
<td>G</td>
<td>s</td>
<td>415</td>
<td>12.10 €</td>
<td>1.244</td>
</tr>
<tr>
<td>16</td>
<td>J</td>
<td>B</td>
<td>s</td>
<td>738</td>
<td>12.30 €</td>
<td>1.224</td>
</tr>
<tr>
<td>32</td>
<td>J</td>
<td>G</td>
<td>a</td>
<td>371</td>
<td>12.37 €</td>
<td>1.218</td>
</tr>
<tr>
<td>8</td>
<td>J</td>
<td>C</td>
<td>s</td>
<td>1486</td>
<td>12.38 €</td>
<td>1.216</td>
</tr>
<tr>
<td>28</td>
<td>J</td>
<td>B</td>
<td>a</td>
<td>433</td>
<td>12.64 €</td>
<td>1.191</td>
</tr>
<tr>
<td>28</td>
<td>C</td>
<td>J</td>
<td>s</td>
<td>434</td>
<td>12.66 €</td>
<td>1.190</td>
</tr>
<tr>
<td>60</td>
<td>C</td>
<td>J</td>
<td>c</td>
<td>203</td>
<td>12.68 €</td>
<td>1.188</td>
</tr>
<tr>
<td>16</td>
<td>B</td>
<td>J</td>
<td>c</td>
<td>761</td>
<td>12.68 €</td>
<td>1.187</td>
</tr>
<tr>
<td>32</td>
<td>J</td>
<td>J</td>
<td>*</td>
<td>382</td>
<td>12.72 €</td>
<td>1.184</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>J</td>
<td>J</td>
<td>*</td>
<td>1807</td>
<td>15.06 €</td>
<td>1.000</td>
</tr>
</tbody>
</table>

- Increase of efficiency by using non-uniform systems? → Yes
  Most of the efficient systems are combinations of J, B, C and G

- Increase of efficiency by increasing k? → Yes
  The top 10 most efficient systems use at least 8 creatures

- Efficiency higher than 1 means synergy through cooperation
Mixing algorithms temporally

- 3 different time shuffling modes (c, s and a)
  - Toggling algorithms X and Y generation-wise (t = 0/1)

**common (c)**
- Only one common state
- State transition and output are toggled

**simultaneous (s)**
- Two states always updated
- Output is toggled

**alternate (a)**
- Two states
- State transition and output are toggled
Conclusion

- The **Creature's Exploration Problem** was modeled by CA
- The behavior of a creature is modeled by a **Mealy-Control-Machine**
- Systems comprising of creatures with different behavior were investigated
- **New metrics** for multi-creature systems were defined
- More creatures lead to **more success**, to **faster exploring** and to **more efficiency**
- Combinations of formerly found good algorithms were used
- Better single algorithms are also better in combination
- Further investigations planned with
  - time-shuffling modes
  - different actions, (e.g. move forward, move backward)
  - communicating creatures
  - heuristic optimization methods
Random Walk

- Comparison of the non-uniform systems against random walk
  - 300 runs on each environment with 32 creatures
  - 1833 generations in average to visit all cells
  - Most efficient system CG-32a needs 339.2 generations in average
  - Any system XY-32p is faster (1779 is the worst value)
  - 9 of the 280 non-uniform systems cannot solve all environments successfully

<table>
<thead>
<tr>
<th>Environment</th>
<th>Random walk</th>
<th>CG-32a</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mean of 300 runs</td>
<td>min</td>
</tr>
<tr>
<td>Env1</td>
<td>869</td>
<td>418</td>
</tr>
<tr>
<td>Env6</td>
<td>2750</td>
<td>1432</td>
</tr>
<tr>
<td>Env16</td>
<td>728</td>
<td>401</td>
</tr>
<tr>
<td>Average</td>
<td>1833</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CG-32a</th>
</tr>
</thead>
<tbody>
<tr>
<td>245</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>623</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>285</td>
</tr>
</tbody>
</table>

Average 339.2
State Graph Structure

- Can we determine how a state graph of a good algorithm has to look like in principle?

- Good behavior seems to derive from cycles in state graphs

\[ E: 1R2L0R4L5L3L-3R4R5R0L1L2R \]