OTTO VON GUERICKE
UNIVERSITÄT
MAGDEBURG

INF

FAKULTÄT FÜR
INFORMATIK

Computational Intelligence in Games

Emergence

Otto-von-Guericke-University Magdeburg

January 20, 2015

Agenda

- Agents
  - Stay Alive Agent
  - Heuristic Agent (HR)
  - MCTS Agent (RL)
  - EA Agent (NI)
- Experiment Result
- Development Process
- Main Problems & Difficulties
- Conclusion
- Future Work

Stay Alive Agent

Stay Alive by using

- the advance() method multiple times
- the grid observation
- a combination of that approaches

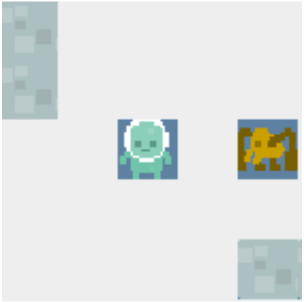


Figure : Advancing safe actions

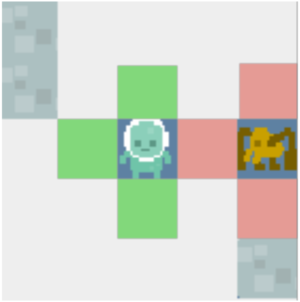


Figure : Grid search for safe actions

Heuristic Agent

- Heuristic for selecting the next best step (including the Stay Alive Strategy)
- Target is found by using an Explorer that is searching for the point of interests
- An *Environment* class builds up the knowledge base and safes blocking, loosing, scoring and winning objects
- A* algorithm is used to reached the good classified objects

Heuristic Agent II

$$dist(u, v) = |x_1 - x_2| + |y_1 - y_2| \tag{1}$$

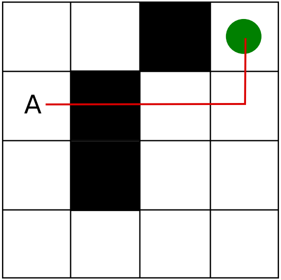


Figure : Manhatten distance for two dimensions

MCTS

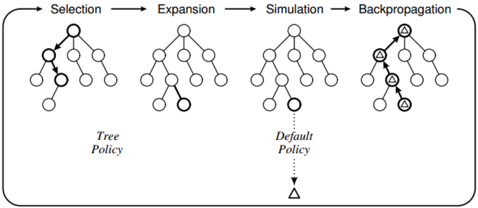


Figure : Monte Carlo Tree Search [2]

- often used for *Decision Making* processes
- balancing between exloration and exploitation

MCTS II - Implementation

- tree policy
  - modified *Upper Confidence Bounds for Trees*:

$$UCT = \frac{Q_c}{V_c + \epsilon \cdot r} + \sqrt{\frac{\ln\left(V_n + 1\right)}{V_c}}$$

  - random node is expanded
- default policy
  - random path, other ideas were discarded
    - four-room-policy
    - self-avoiding-policy
    - edge-weighted-policy

MCTS III - Implementation

- open loop approach
  - algorithm works on a sequence of actions (no state observation is saved!)
  - simulation: path from root to expanded node has to be simulated
- rolling horizon
  - goal: save computing power
  - reuse MCTS tree of previous gamestep
  - new root-node = child of previous root-node (with corresponding action)

EA

Algorithm 1 pseudocode of an evolutionary algorithm [3]

```
Initialize Population with random candidate solutions;
Evaluate each candidate;
while Termination condition not satisfied do
    Select parents
    Recombine pairs of parents
    Mutate the resulting offspring
    Evaluate new candidates
    Select individuals for the next generation
end while
```

EA Agent

DeltaScoreEvaluation function

$$s = \sum_{t=0}^{n} (H(s_t) - H(s_{t-1}))$$

is calculated by using the function

$$H(s_i, s_{i-1}) = \begin{cases} 10, & \text{if isWinner} \\ -10, & \text{if isLooser} \\ score(s_i) - score(s_{i-1}), & \text{otherwise.} \end{cases}$$

EA Agent II

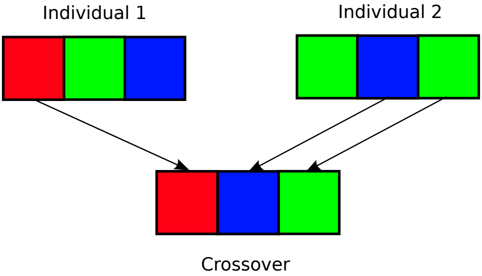


Figure : Crossover of an individual

EA Agent III

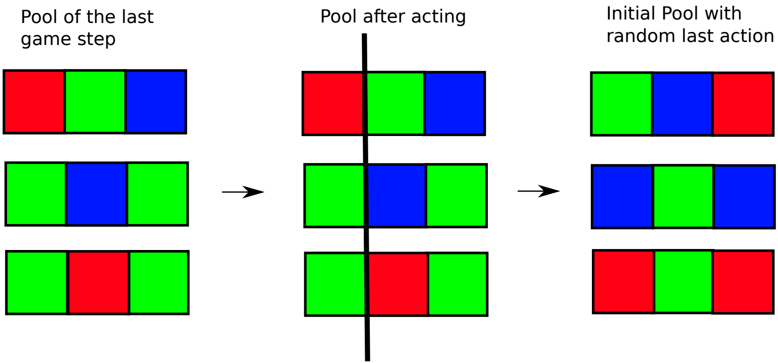


Figure : Sliding Window

Experiment Result

- Comparison among each approach to be fair (1000 games, one game 50 times, 10 times each level)
- Evaluation of the best of each algorithm (3000 games, one game 150 times, 30 times each level)

CPU	Intel i5-4210U @ 1.70Ghz
Memory	8 GB DDR3 L
Operating System	Ubuntu 14.04.1 LTS
Java Version	1.7.0_65

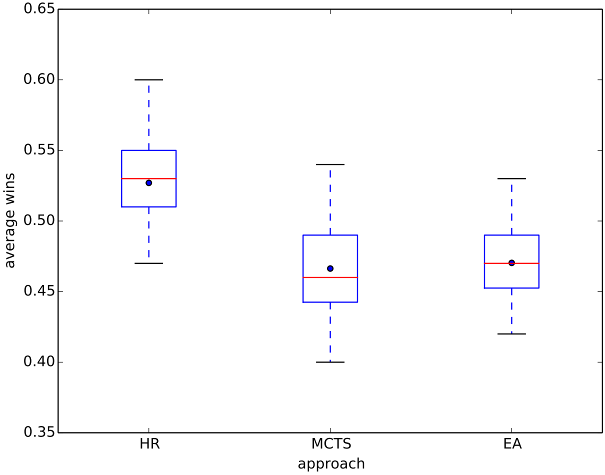
Table : experiment setup

Experiment Result II

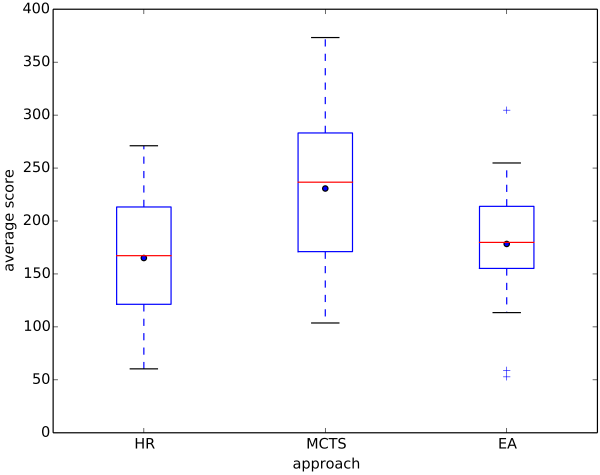
Approach	Avg Wins	Std Wins	Avg Score	Std Score	Avg time steps	Std time steps
HR	**0.527**	0.029	165.05	59.51	**695.86**	36.17
MCTS	0.467	0.034	**230.69**	74.64	942.06	**34.00**
EA	0.470	**0.026**	178.33	**51.85**	818.72	38.47

Table : results of all algorithms

Experiment Result III - boxplot wins



Experiment Result IV - boxplot score



Development Process

- upload (submitting) / framework
- some approaches (e.g. modification of default policy) doesn't work
    - random approach was often better
    - maybe too much forbidden things
- we often change the approach of an agent
- often agents were rewritten completely

Main Problems & Difficulties

- time limitation (controllers, report, ...)
- computing power
  - differences between machines $\Rightarrow$ large variation in the results
  - not all parameters could have been tested
- surprised of the winner (heuristic approach)

Conclusion

- only results (win/loss/score) were considered for evaluation (no ingame-behavior)
- Heuristic controller performed best (number of wins)
- MCTS controller performed best (score)
- Reasons
  - implementations of MCTS and EA not effective enough
  - games with very few winning-sprites
- can not make general statements about the accuracy of the three approaches (HR, MCTS, NI)

Future Work

- test algorithms with more computing power
  - more iterations
  - loop over possible parameters to find the best
- test algorithms on unknown (test set) games
- implement controller with other approaches (Neuronal Nets, Pheromone)
- combine approaches and win the GVGAI Competition ;)

Thank you for your attention!