Evaluation of Techniques for a Learning-Driven Modeling Methodology in Multiagent Simulation

Robert Junges (robert.junges@oru.se)
Franziska Kluegl (franziska.klugl@oru.se)
Modeling and Simulation Research Center
Örebro University - Sweden

MATES 2010

Agenda

- Motivation
- Learning-Driven Modeling Methodology
- Learning Techniques
- Test Case: Pedestrian Evacuation Scenario
- Conclusions and Next Steps

Motivation

- Multiagent simulation model design:
  - Often trial and error process
  - Often unclear level of detail
  - How to create the proper agent behavior?
  - Agent model outcome X Meso Level X Overall simulation outcome

Motivation

- Multiagent simulation model design:
  - Often trial and error process
  - Often unclear level of detail
  - How to create the proper agent behavior?
  - Agent model outcome X Meso Level X Overall simulation outcome

- Suggestion: Agent Learning for Behavior Modeling

Design Strategy

1. Develop a model of the environment
2. Define the perceptions and actions of the agents
3. Describe the intended outcome: reward function on the performance of the agent
4. Apply an agent learning technique
5. Analyze and test learned behavior for validity - if not go back to step 1

Model Design Support

- Human modeler is in charge of the model
  - Takes responsibility for model quality
  - Our aim: support for modeler providing inspiration about local behavior
- Learning must produce optimal and readable output
  - **Feasibility**
  - **Interpretability**
  - **Plausibility**

Question…

- What would be a good agent learning technique to cope with the requirements of this methodology?

Question…

- What would be a good agent learning technique to cope with the requirements of this methodology?

- Candidates…
  - XCS - Learning Classifier Systems
  - Q-Learning - Reinforcement Learning
  - FFNN - Neural Networks

XCS - Learning Classifier System

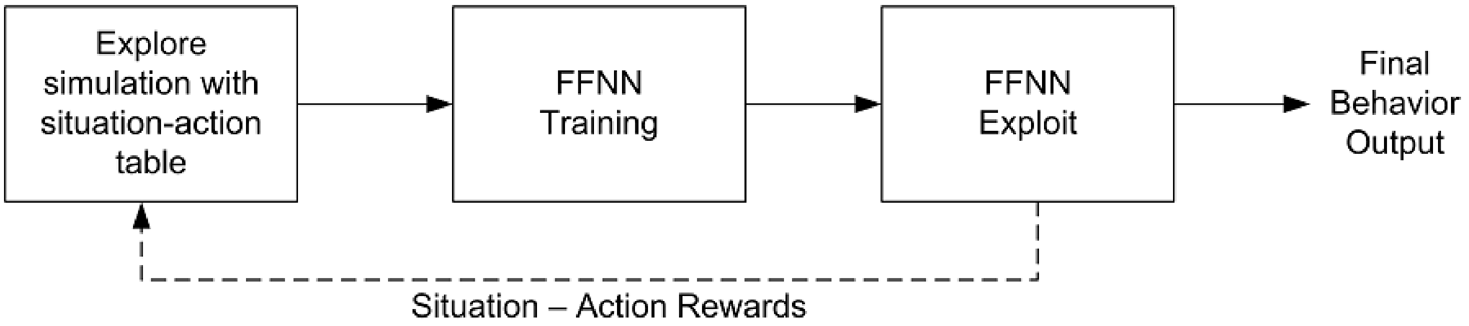
- Iterative online learning system
- Knowledge represented by a fixed-size population of **condition-action-prediction classifiers**
  - Classifiers predict the reward and actions given the conditions
  - Reinforcement Learning-based technique
- GA component → discovery of new classifiers
  - Fitness selection: accuracy

Q-Learning - Reinforcement Learning

- Reinforcement learning technique
- Develops an action-value function
    - Expected utility for action A in a specific state S
    - Q-Table: situation-action pairs + Q-Value

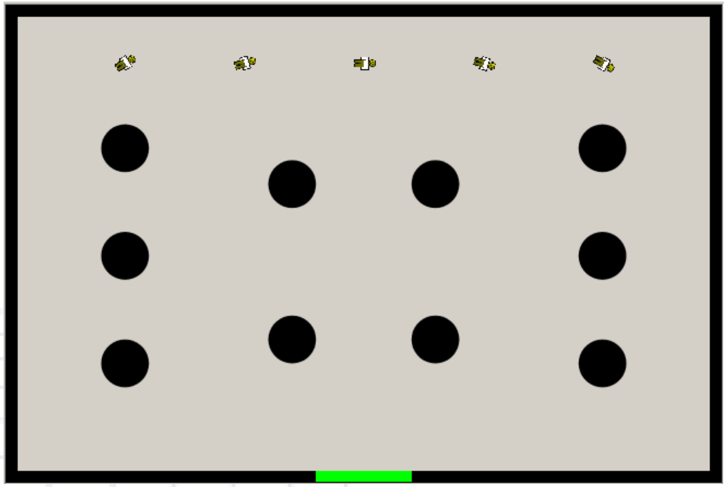
FFNN – Neural Networks

- Artificial neural network
  - Information moves forward
- Our methodology is designed for online reward-based learning…
  - FFNN → supervised training
  - We modified the overall learning process:



Test Case Scenario - Environment

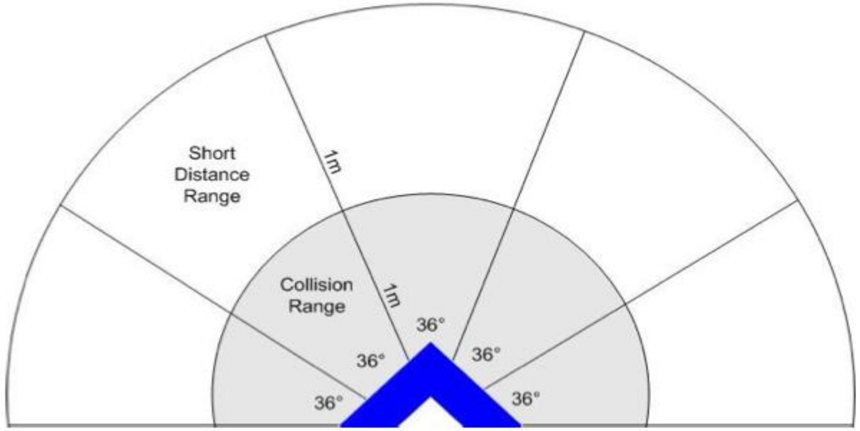
- Pedestrian evacuation scenario
    - 20x30 room with column-type obstacles and 1 exit
    - 1, 2 and 5 agents positioned in the upper-half of the room
    - 1, 5 and 10 obstacles
    - 100 explore-exploit trials each simulation



Scenario Perception - Action

- Perceptions: Obstacle and Exit
- Actions

  $Move_{Straight}$, $Move_{Left}$, $Move_{Right}$, $Move_{SlightlyLeft}$, $Move_{SlightlyRight}$, StepBack, Noop
- MDP on collision avoidance

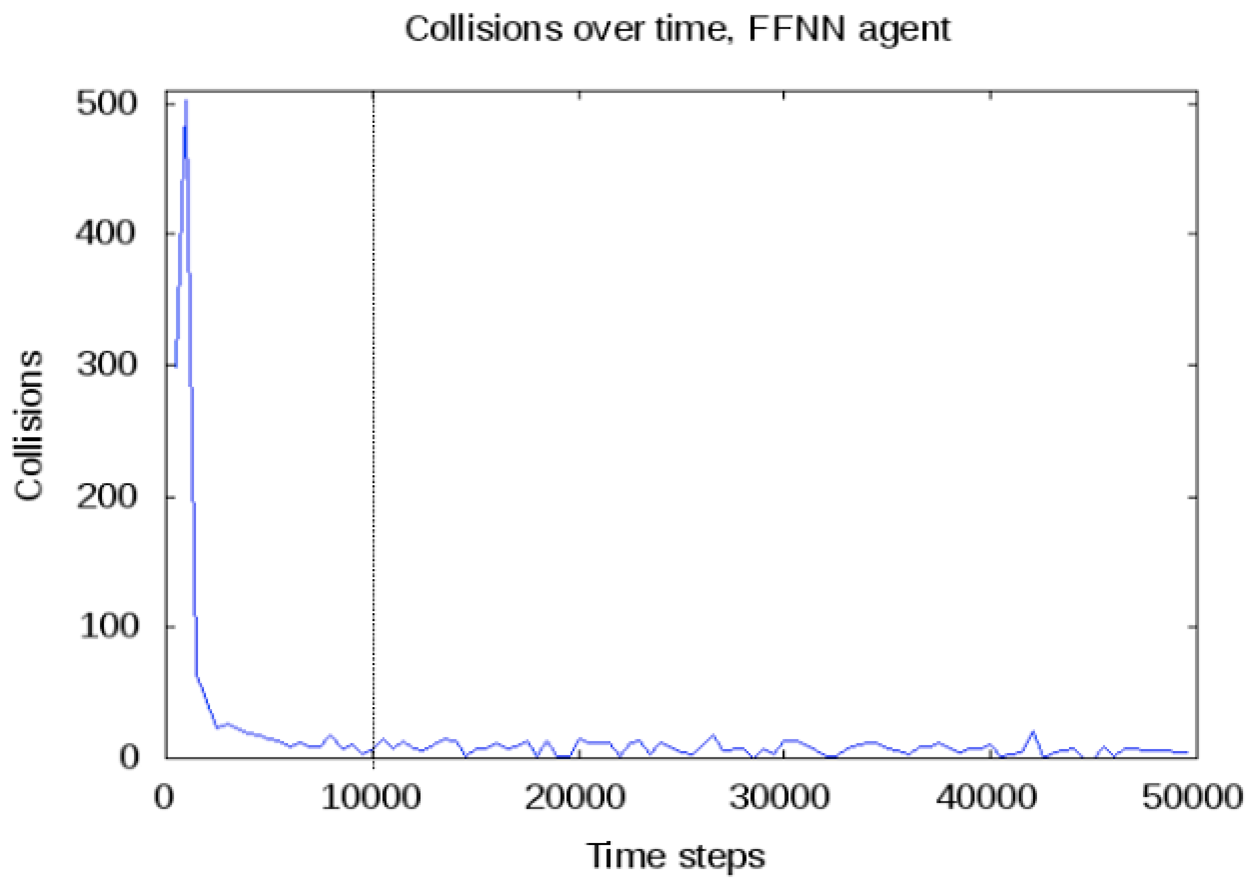


Scenario Rewards

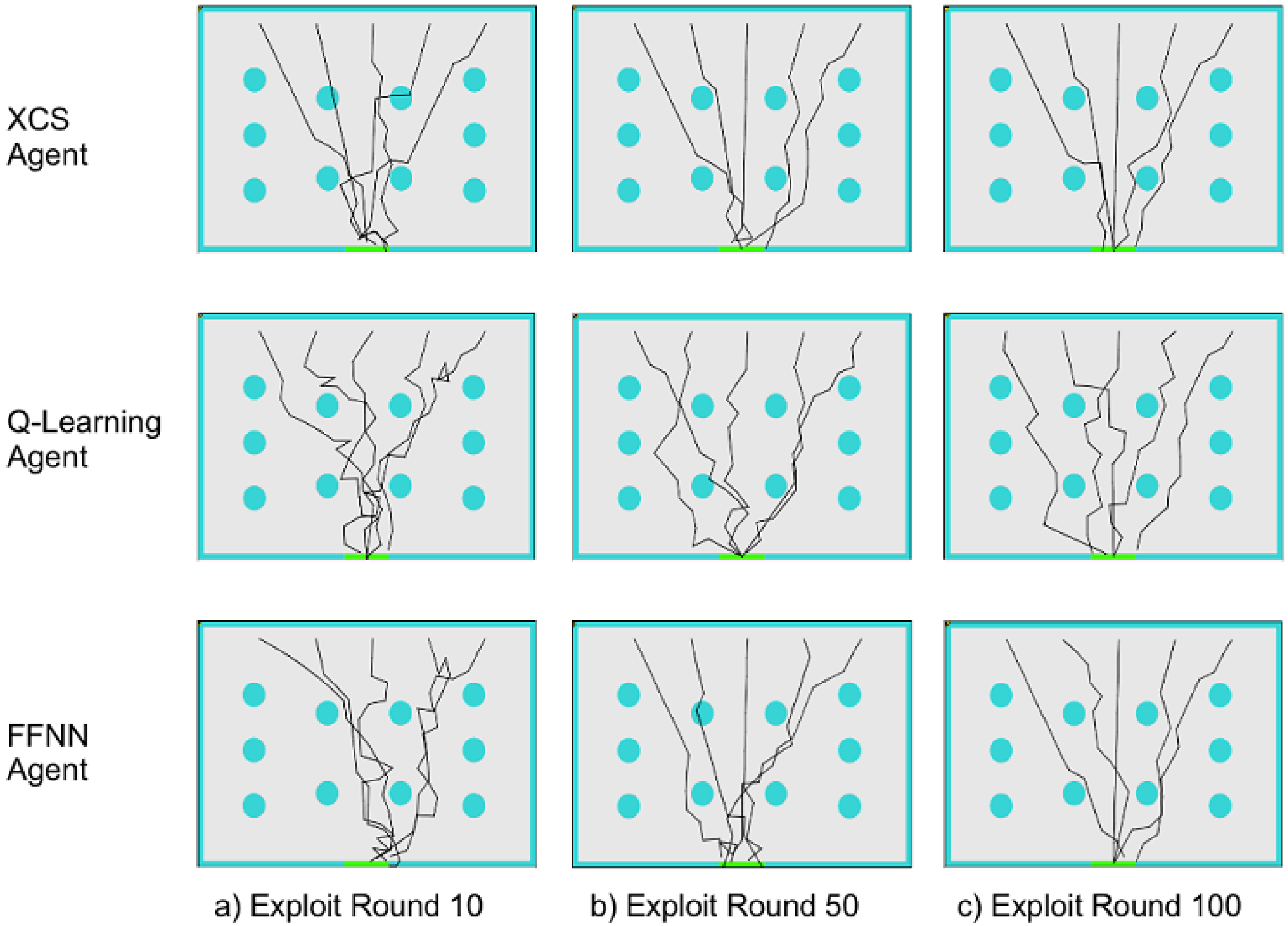
- $Reward = Reward_{Exit} + Reward_{Distance} + Collision$
- Where:
  - $Reward_{Exit}$ = 200 for exit or 0 otherwise
  - $Reward_{Distance} = \beta \times [\, d_t(exit) - d_{t-1}(exit) \,]$, with $\beta = 5$
  - Collision = 100 for a collision free movement, 0 for no movement, and -100 if a collision occurred

Learning Examples

5 Agents and 10 Obstacles, FFNN



Learning Examples

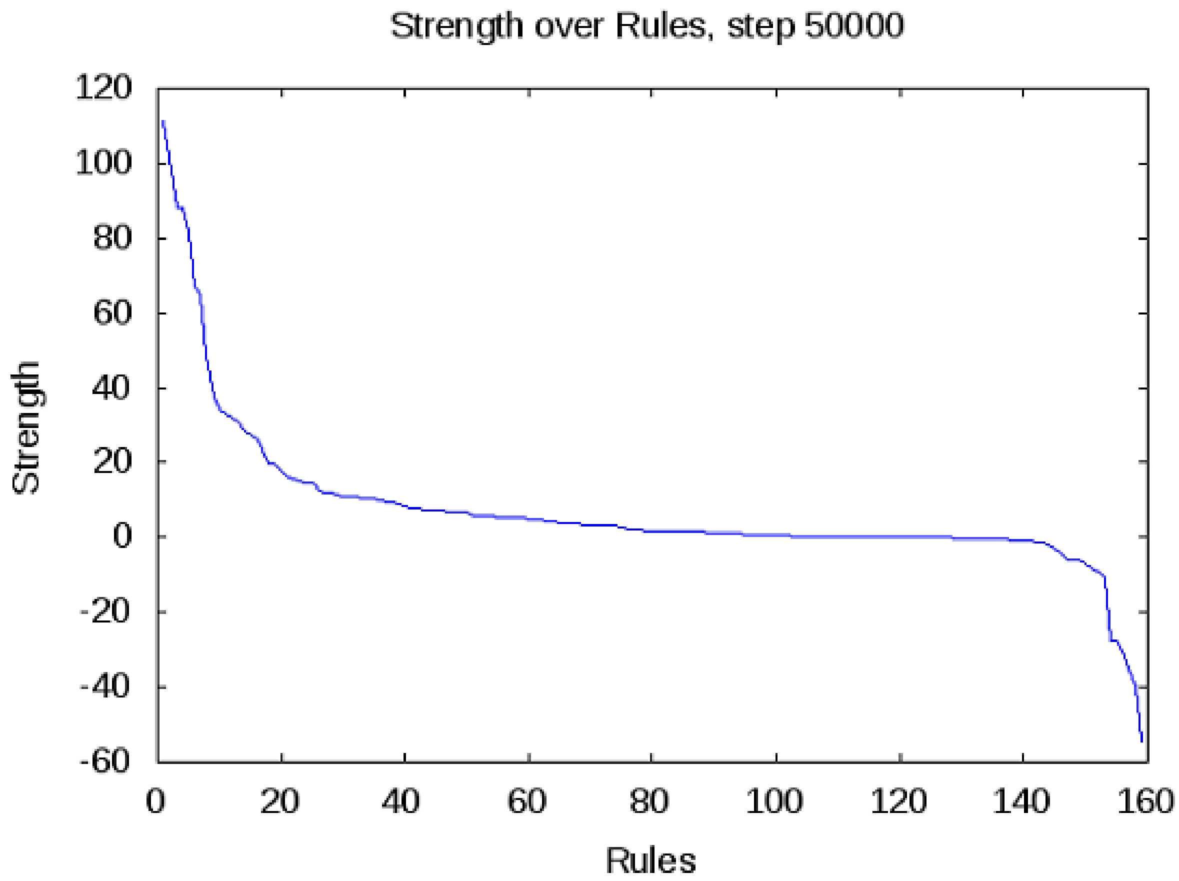


Learning Evaluation – 5 agents

- XCS
  - 160 rules
  - 71.25% positive reward
  - Generalization through don't care bits
- Q-Learning
  - 1961 rules
  - 44% not experienced; Q-Value = 0
  - No generalization
- FFNN
  - Last exploit round used 45 rules
    - Selection of best actions for each situation
  - No learning from negative experience

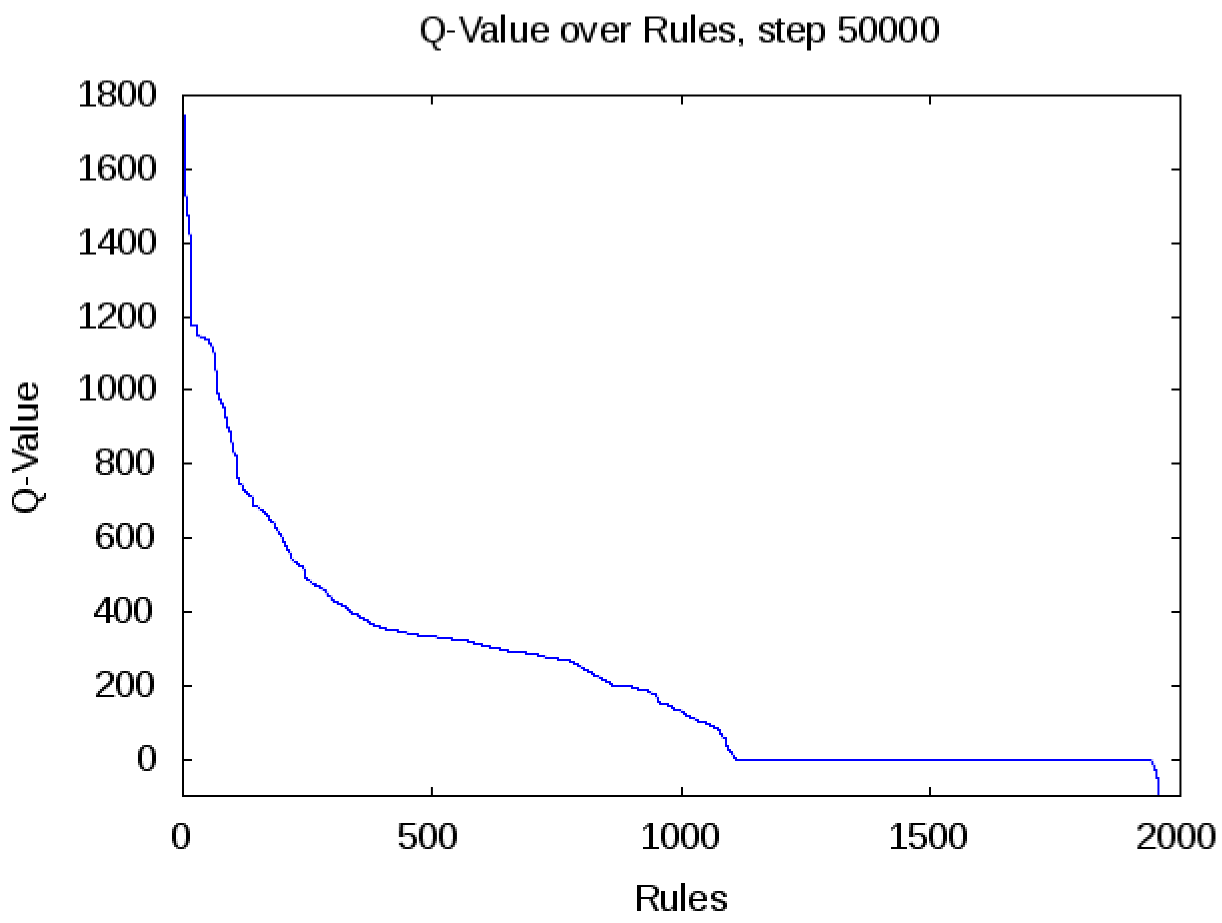
Learning Evaluation – 5 agents

XCS Rules Strength



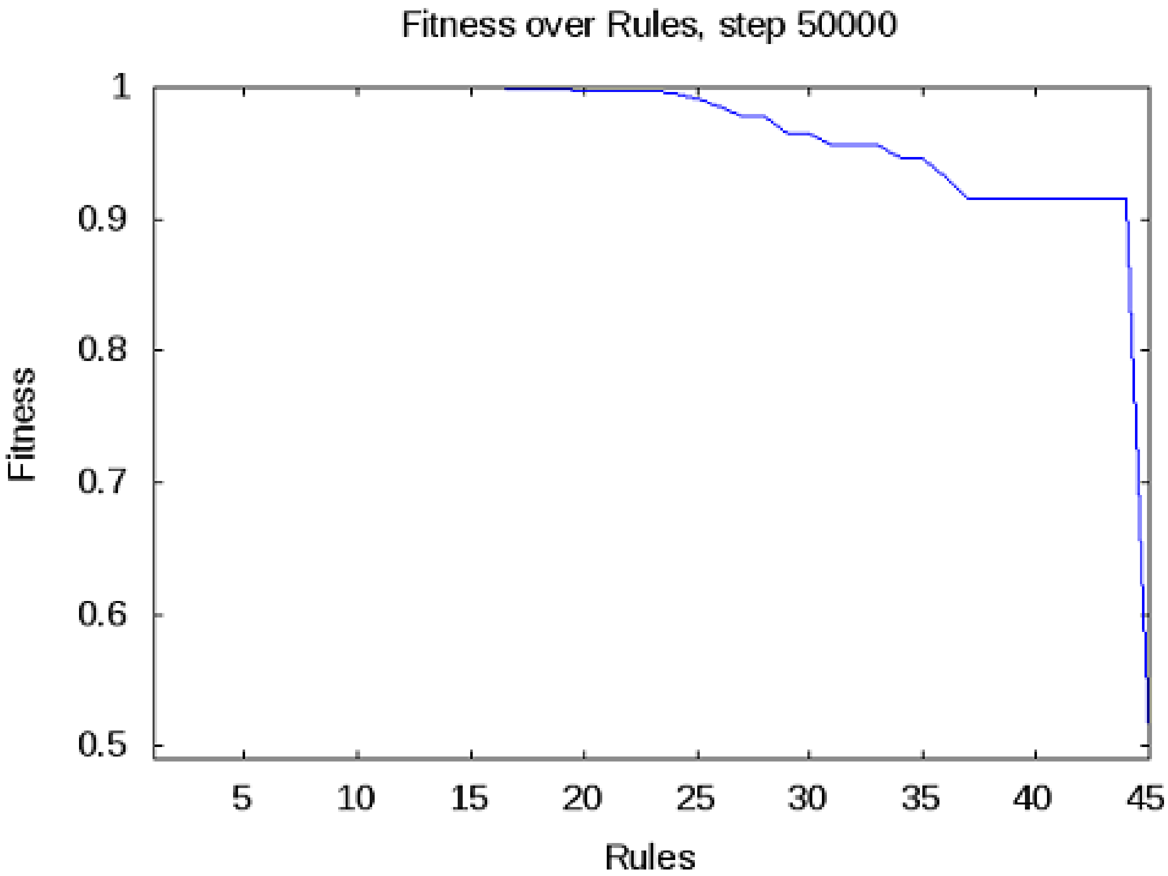
Learning Evaluation – 5 agents

Q-Learning Reward Distribution



Learning Evaluation – 5 agents

FFNN Rules Fitness Distribution



Final Rules Example

- XCS rules: 5 best rules

Condition (bit string)	Condition Interpretation	Action	Strength	Fitness	Experience
******0******0******	No obstacle immediately right No obstacle near left	$Move_{SlightlyRight}$	111.12	0.71	61
****0*0******0******	No obstacle immediately ahead or right No obstacle near left	$Move_{Straight}$	101.32	0.58	61
00*0*0***0*0*0**	No obstacle immediately left No exit near left, right or ahead	$Move_{SlightlyLeft}$	88.30	0.56	81
0**0*******00**010**	No obstacle or exit left No exit ahead or right Obstacle near right	$Move_{Left}$	81.62	0.69	24
01**00**1*00*0	No obstacle right No exit left Obstacle ahead	$Move_{SlightlyRight}$	65.1	0.41	21

Conclusions - XCS

- Better interpretation of rules
  - Reward prediction
  - Fitness
  - Experience
- Generalization: don’t care bits
- Evolutionary rule discovery

Conclusions – Q-Learning

- Less computation time
- State-action pairs table offers a good base for model design
- However, large set of rules…
  - How to generalize and interpret these rules? Post processing?
  - Rules measured only by reward prediction

Conclusions - FFNN

- Less performance in this implementation
- Training does not consider utility value
  - Does not consider similarity of actions in terms of Q-Values
  - Does not consider actions to avoid: negative Q-Value
- Black box system

Conclusions

- Investigation towards a learning-driven methodology by evaluating different learning techniques
- In a small evacuation scenario, the employed learning produced plausible behavior in an agent-based simulation
    - No clear technique showed the best performance
    - XCS technique outclasses the two other when it comes to the accessibility and usability of the learned behavior model

Next Steps

- **How to improve generalization and interpretation of the rules learned (reinforcement learning case)?**
  - **Learning convergence**
  - **Post processing step**
- More complex scenarios
- Other techniques: evolutionary programming, other reinforcement learning
- Catalogue of properties to show the appropriateness of the learning techniques
- Integration of the behavior into explicit models

Thank you!