Organic Computing

Übung 1

Stefan Rudolph

Lehrstuhl für Organic Computing
Institut für Informatik
Universität Augsburg

8. Mai 2015

Universität Augsburg
Fakultät für Angewandte Informatik

ORGANIC COMPUTING

Überblick

- Heute
  - Organisatorisches
  - Einführung in BW
  - Semesteraufgabe
  - Wiederholung: Extended Classifier Systems
  - Zusammenfinden der Gruppen
  - Bearbeitung des ersten Übungsblatts
- Kontakt
  - Sprechstunde nach Vereinbarung per Email
  - stefan.rudolph@informatik.uni-augsburg.de
  - johannesjungbluth@googlemail.com

Organisatorisches
Übungsblätter

- ~2-3 Wochen Bearbeitungszeit
- Abgabe per Email
  - johannesjungbluth@googlemail.com
- Entweder  oder 
  - Falls  muss nachgearbeitet werden.
- Die Lösungen sind pünktlich abzugeben

Organisatorisches
Überblick über die Übungsblätter

1. XCS
2. Genetic Algorithms
3. Swarming
   - Eigener BW Spieler

- Ablauf der Übung
  - Wiederholung des Vorlesungsstoffes
  - Weitergehende Hinweise zur Übung
  - Betreutes Programmieren
- „Präsenzübung“

- Bonus
  - Ein Notenschritt (= 1/3 Note) für mündliche Prüfung.
- Voraussetzungen
  - Erfolgreiche Bearbeitung der Übungsblätter
  - Teilnahme am Abschlussturnier

Starcraft: Brood War
Was ist das?

- Meistverkauftes Echtzeit-Strategiespiel.
  - fast 10 Millionen Exemplare.
- Drei Challenges
  - Ressourcen
  - Macro
  - Micro
- Drei Parteien
  - Terraner
  - Protoss
  - Zerg

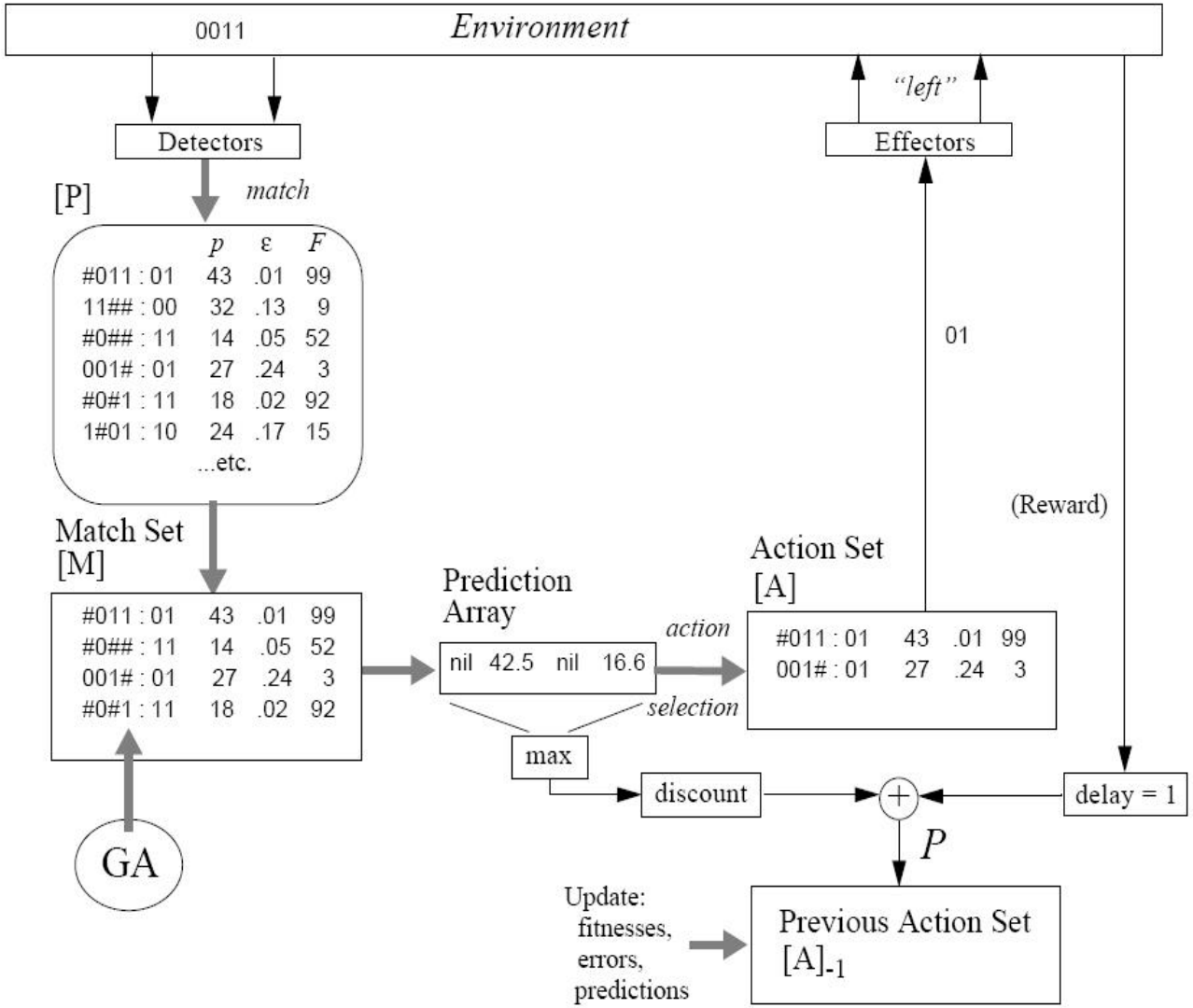
- BW ist ein komplexes System
  - Viel Subsysteme mit nicht-trivialen Zusammenhängen
  - Passt genau ins Forschungsfeld Organic Computing
- Wird für aktuelle Forschung herangezogen
  - Konferenzen
  - AI Competitions
- E-Sport

- Einfaches Setting
  - Einheiten vorgegeben
  - Kein Bauen von Gebäuden
  - Kein Bauen von Einheiten
- Organic Computing Starcraft Micro AI Championship
  - OC-SMAC 2015
  - Turnier am Ende des Semesters
- Bei weitergehendem Interesse
  - Projektmodul
  - Abschlussarbeiten
  - ...

Sonstiges

- Folien im Digicampus
- stefan.rudolph@informatik.uni-augsburg.de
- Sprechstunde nach Vereinbarung per Email

XCS - Overview



- classifiers are extended 5-tuples
  (Condition, Action, Prediction (p), Prediction error (ε), Fitness (F))
- Condition, Action: same as in ZCS
- Prediction: measuring the pay-off received for an *action*
- Prediction error: measuring the error of a classifier's *prediction*
- Fitness: a function of the inverse *prediction error*
- other parameters can be kept depending on the implementation, e.g., number of times classifier has been used (experience)
- prediction and prediction error used in action selection
- fitness used by the genetic algorithm and for action selection

- match set calculated as before
- different policies for selecting an action from the prediction array, e.g.
  - deterministic selection: highest prediction wins
  - probabilistically based on fitness-weighted average of predictions
- selected action is sent to the effectors
- reward may (or may not) be received in return
- classifiers are updated (differently for single- and multi-step problem)

- reinforcement consists of updating prediction, prediction error, and fitness
- update prediction for each classifier C_j in [A]:
  - $p_j \leftarrow p_j + \beta(P - p_j)$
  - P is the current reward
  - β is called the learning rate
- update prediction error accordingly
  - $\varepsilon_j \leftarrow \varepsilon_j + \beta(|P - p_j| - \varepsilon_j)$
- update fitness accordingly
  - $F_j \leftarrow F_j + \beta(\kappa_j' - F_j)$
  - κ_j' is the relative accuracy across [A], $\kappa_j \cong 1/\varepsilon_j$

- for multi-step problems updating is done based on $[A]_{-1}$
- delayed update allows to retrieve "information from the future"
- inspired by Q-Learning (reinforcement learning technique)
- in XCS:
  $P = r + \gamma * \max P(a)$

Reward at t-1

Prediction array at t