The Multi-Agent Programming Contest

Miriam Kölle

Research lab – Summer term 2014

University Koblenz-Landau

The Multi-Agent Programming Contest

- International online competition since 2005
- Incite research by
  1. Identifying key problems
  2. Collecting suitable benchmarks
  3. Gathering test cases

The Multi-Agent Programming Contest

- Changing scenarios
  - 2005: Food Gatherer
  - 2006-07: Goldminers
  - 2008-10: Cows and Cowboys
  - 2011-13: Agents on Mars
  - 2014: unsettled
- Focus: Agent cooperation and coordination
- Implementation technology left to participants
- Communication technology left to participants

Agents on Mars

»In the year 2033 mankind finally populates Mars.«

- Search for water wells and occupy the best zones

- Challenge:
  - Find water and occupy zones
  - Attack rivals and defend
  - Earn money (*milestones*, as 5 successful attacks)

- **Goal: maximize the score** (zones and money)

Environment

- Weighted graph
  - Weighted edge: costs of traversing this edge
  - Labeled node: value of this water well
- Unknown in the beginning
    - Agents must explore it

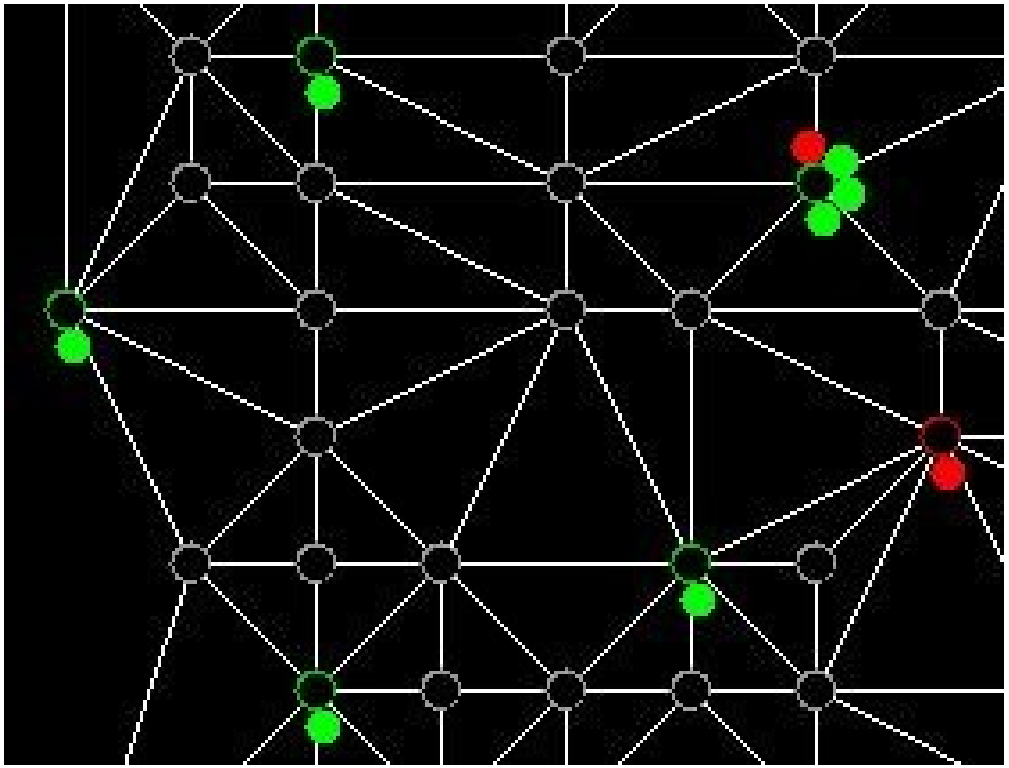
Occupying Zones

- Depends on all agents' current position

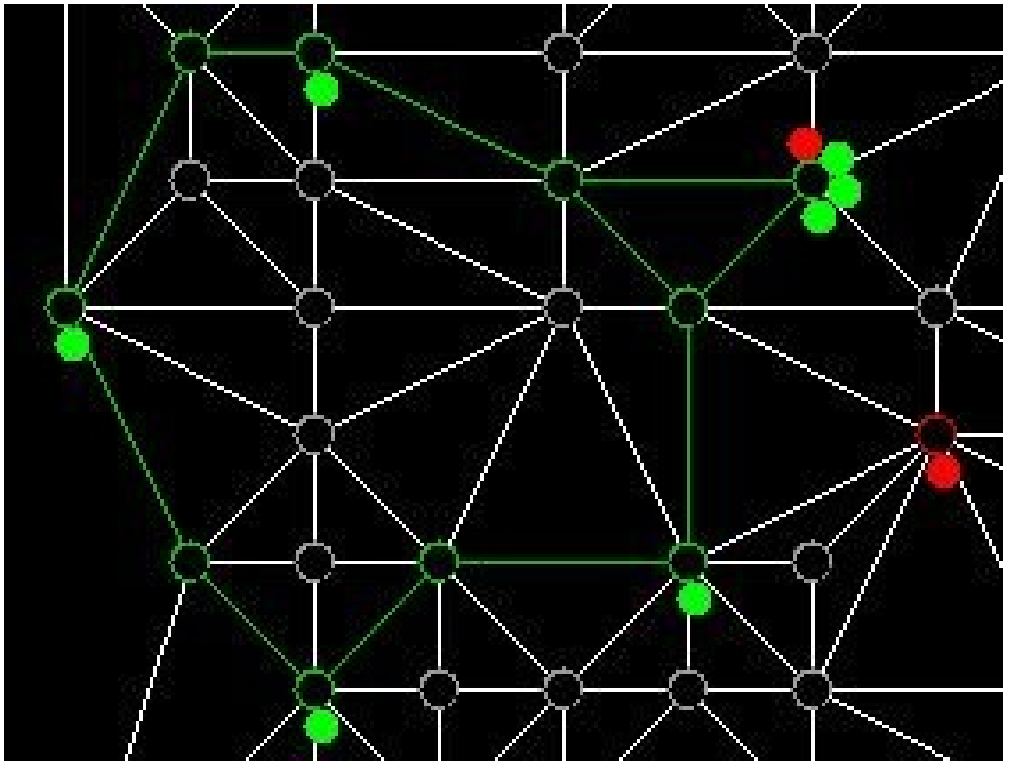
1. A node belongs to that team with the majority of agents standing on this node
2. Neighbors dominated by at least two neighbors belong to that team
3. All nodes in an isolated zone belong to that team

- **One agent allein cannot establish a zone**

Occupying Zones

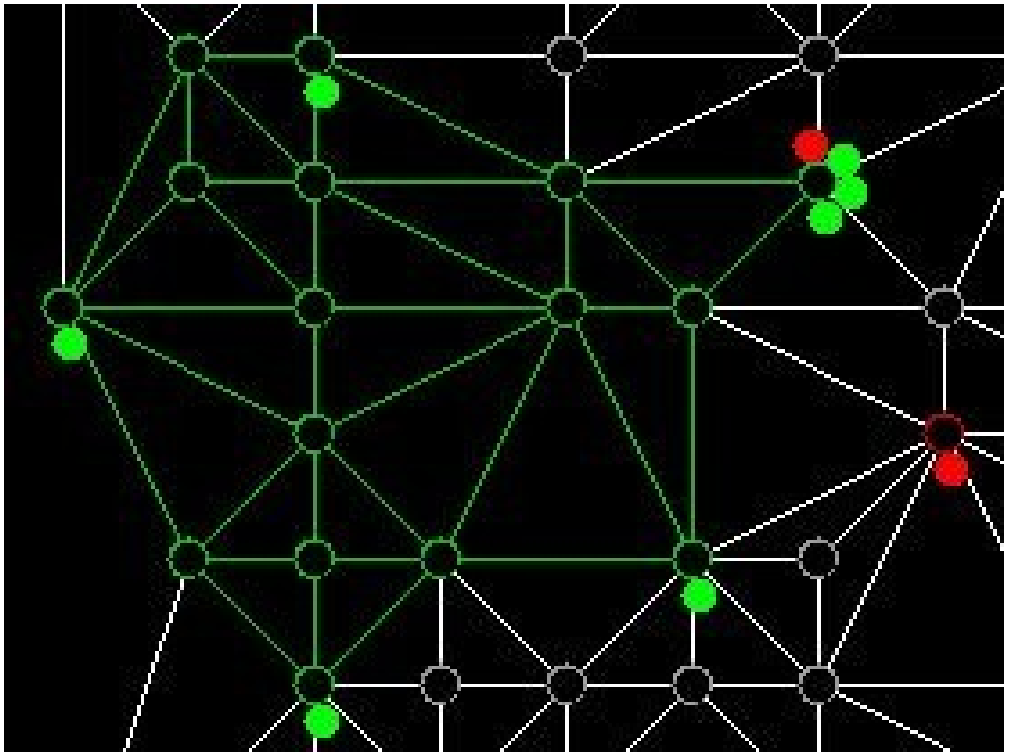


(1)

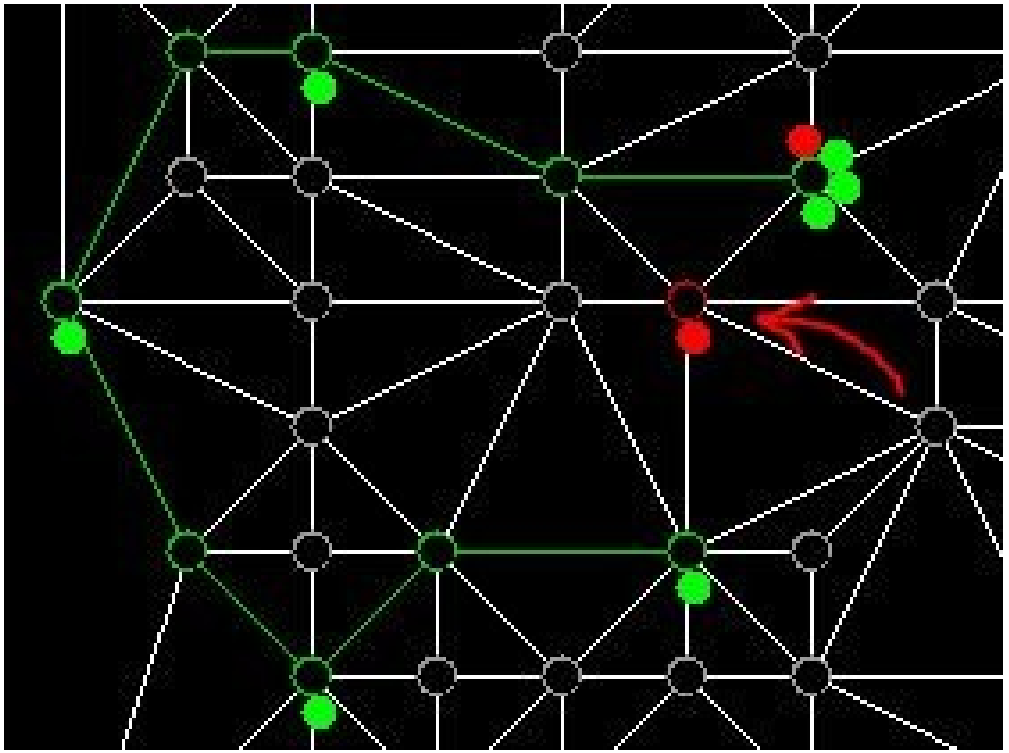


(2)

Occupying Zones



(3)



(4)

Agents

- Attributes
  - *Energy, health, strength, visibility range*
- Actions
  - Most are self-explanatory (next slide)
  - *Probe*: find out the value of the current node
  - *Survey*: find out adjacent edges' weights
  - *Inspect*: find out current attributes of other agents
  - *Buy*: improve attributes
- **Five different roles → "experts"**

Agents

Role	Actions	Energy	Health	Strength	Visibility Range
Explorer	Skip, goto, **probe**, survey, buy, recharge	12	4	0	2
Repairer	Skip, goto, parry, survey, buy, **repair**, recharge	8	6	0	1
Saboteur	Skip, goto, parry, survey, buy, **attack**, recharge	7	3	4	1
Sentinel	Skip, goto, parry, survey, buy, recharge	10	1	0	3
Inspector	Skip, goto, **inspect**, survey, buy, recharge	8	6	0	1

- Attributes can vary during simulation:
  - Actions cost energy, Agents can get disabled

MASSim Server

- Agent teams run on participant's infrastructure
- Simulated environment runs on MASSim server
  - Agents communicate with the server by exchanging XML messages
- 3 phases:
  - Initial
  - Simulation
  - Final

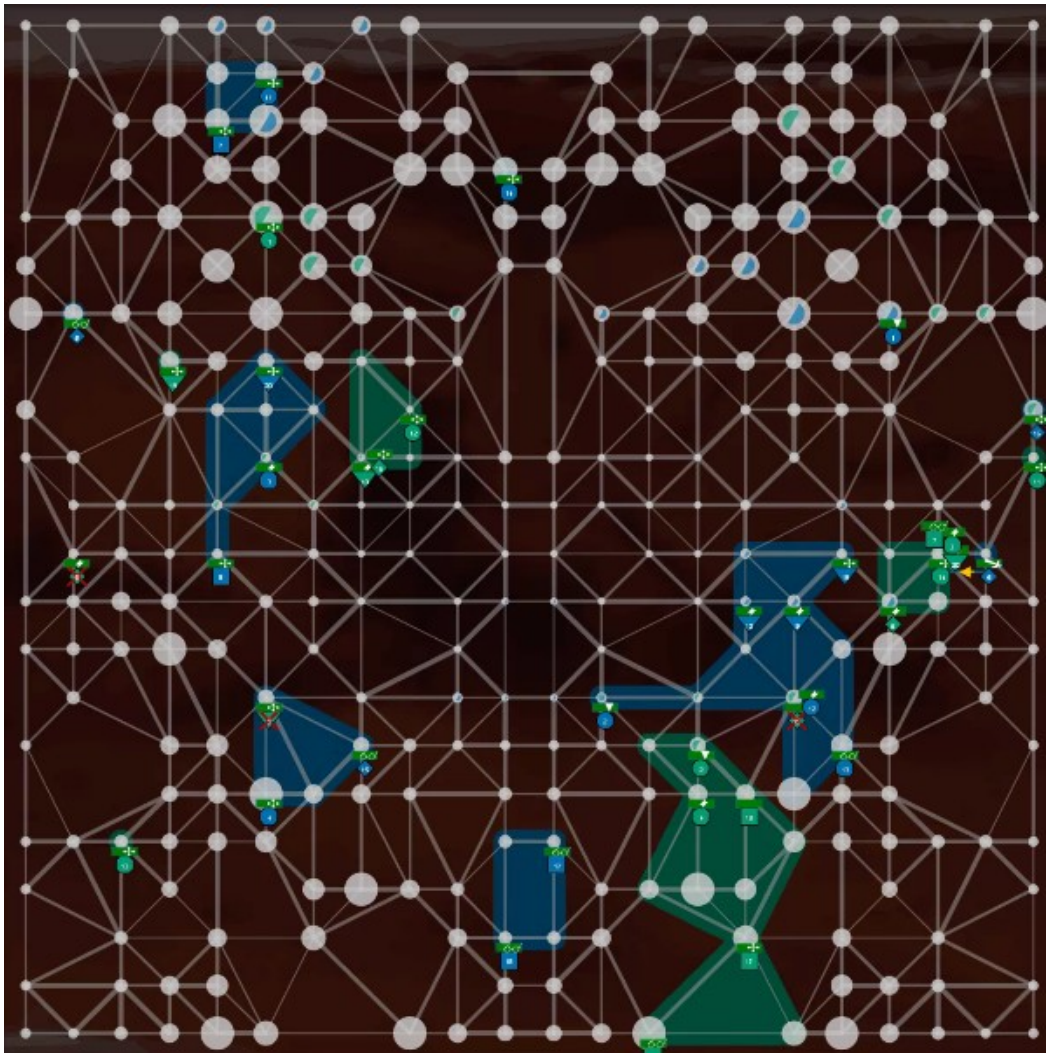
Simulation Step

- collect all actions from the agents,
- let each action fail with a specific probability,
- execute all remaining attack and parry actions,
- determine disabled agents,
- execute all remaining actions,
- prepare percepts,
- deliver the percepts.

Simulation Step

- Perceptions:
  - State of the simulation
  - State of the team
  - State of the agent
  - Visible vertices (+ dominating team)
  - Visible edges
  - Visible agents (+ node, team)
  - Returns of *probe, survey,* and *inspect*

Visualization



Environment Design

- Agents are situated in an environment
- Agents (inter)act upon the environment

  → based on perceptions
- Here: environment is only simulated
  - Needs to be modelled
  - Interesting approach: ELMS
    - *Environment Description Language for MAS*
    - Part of the MAS-SOC framework
    - XML Syntax

ELMS Modelling

- Objects
  - Set of properties
  - Set of re-actions (to agent's actions)
- Example:

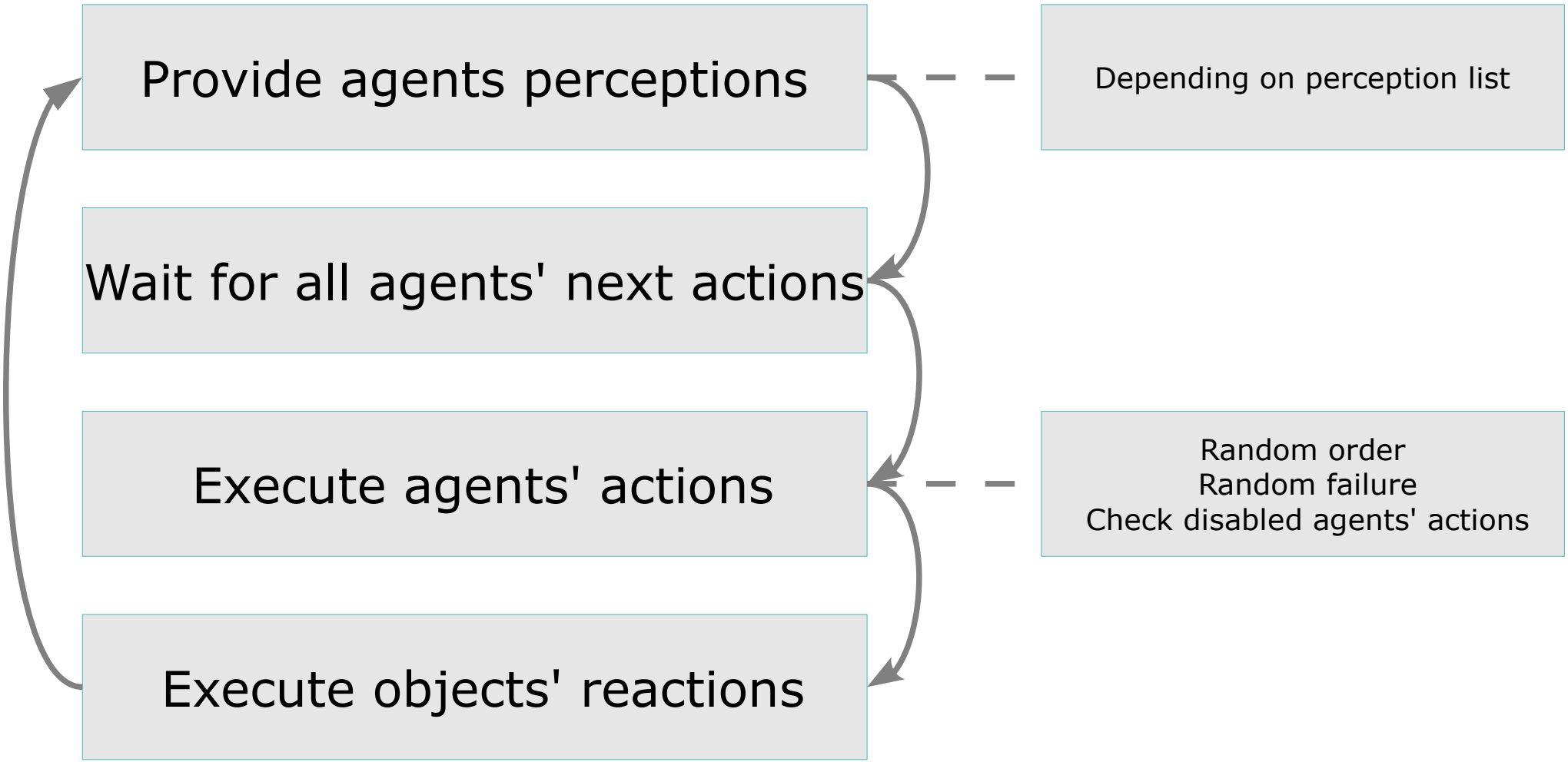
```
<RESOURCE NAME = “water”>
   <STRING NAME = “state” VALUE “liquid”/>
   <INTEGER NAME = “temperature”>23</INTEGER>
   <REACTIONS>
      <ITEM NAME = “solidify”/>
      <ITEM NAME = “melt”/>
   </REACTIONS>
</RESOURCE>
```

ELMS Modelling

- Agents
  - Set of attributes
  - Set of actions
  - Set of perceptions
- Example:

```
<AGENT_BODY NAME = "agent_x">
    <INTEGER NAME = "id"> "SELF" </INTEGER>
    <ACTIONS>
        <ITEM NAME = "walk_right"/>
        <ITEM NAME = "walk_left"/>
    </ACTIONS>
    <PERCEPTIONS>
        <ITEM NAME = "vision"/>
    </PERCEPTIONS>
</AGENT_BODY>
```

Summary


Provide agents perceptions
Depending on perception list
Wait for all agents' next actions
Execute agents' actions
Random order
Random failure
Check disabled agents' actions
Execute objects' reactions

Conclusion

- Each Team: 28 agents
  - Competition: all against all others, 3 sim
- Agents need to
  - Choose zones in order to maximize the score
    - Only probed nodes “count their value”
- Find the best strategy
  - One agent gets all percepts and decides ?
  - “auction-based” agreement ?
  - Shared knowledge of the graph ?

Conclusion

- Other participants
  - Some take part quite regularly
  - UFSC (Winner of last two years)
    - Jason / JaCaMo (Jason, CArtAgO, Moise)
  - DTU
    - GOAL, Python
  - Others: Java, C++, JIAC

Thank you for your attention!

References