Multi-Agent Programming Contest

Artur Daudrich, Sergey Dedukh, Manuel Mittler,

Michael Ruster, Michael Sewell, Yuan Sun.

Research lab – Summer term 2014

University Koblenz-Landau

Outline

- The contest
- Scenario
- Our Strategies
  - Zoning strategy
  - Agent specific strategies
- Implementation details
- Competition outcome
- Lessons learned

The Contest – Aims and Scope

- International online competition since 2005
- Attempt to stimulate research in the area of multi-agent system development and programming
- Focus: Agent cooperation and coordination. Implementation technology left to participants

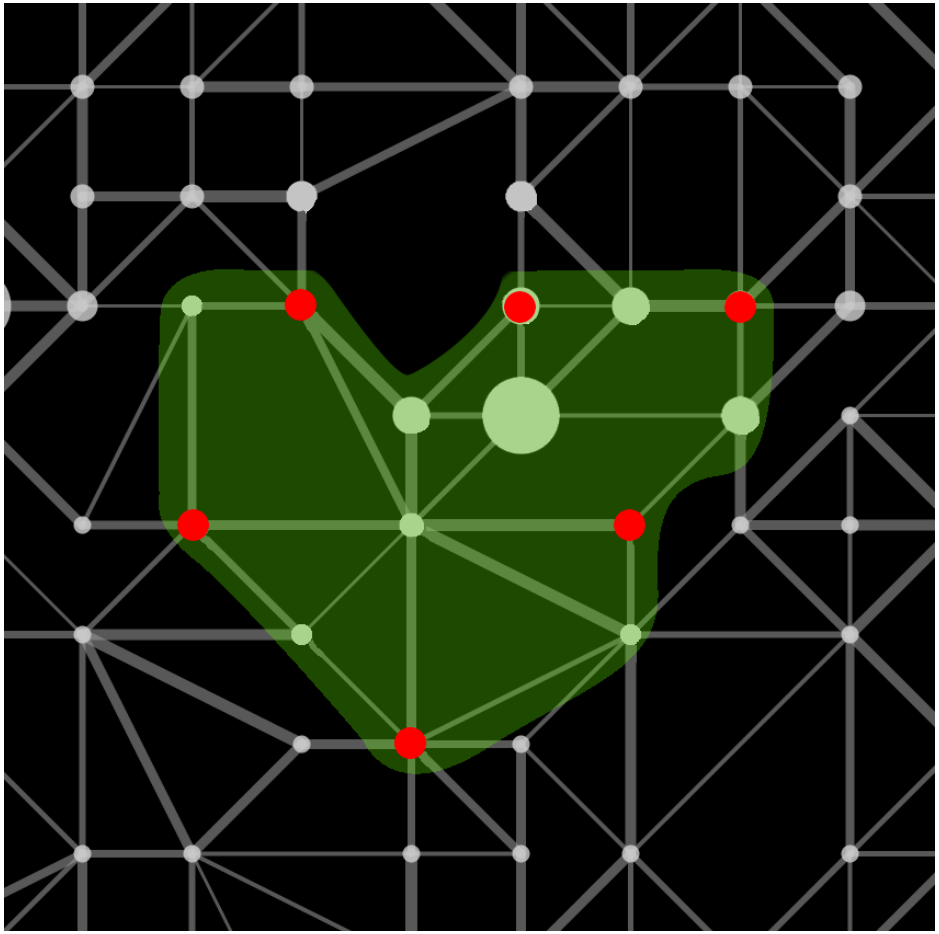
Scenario

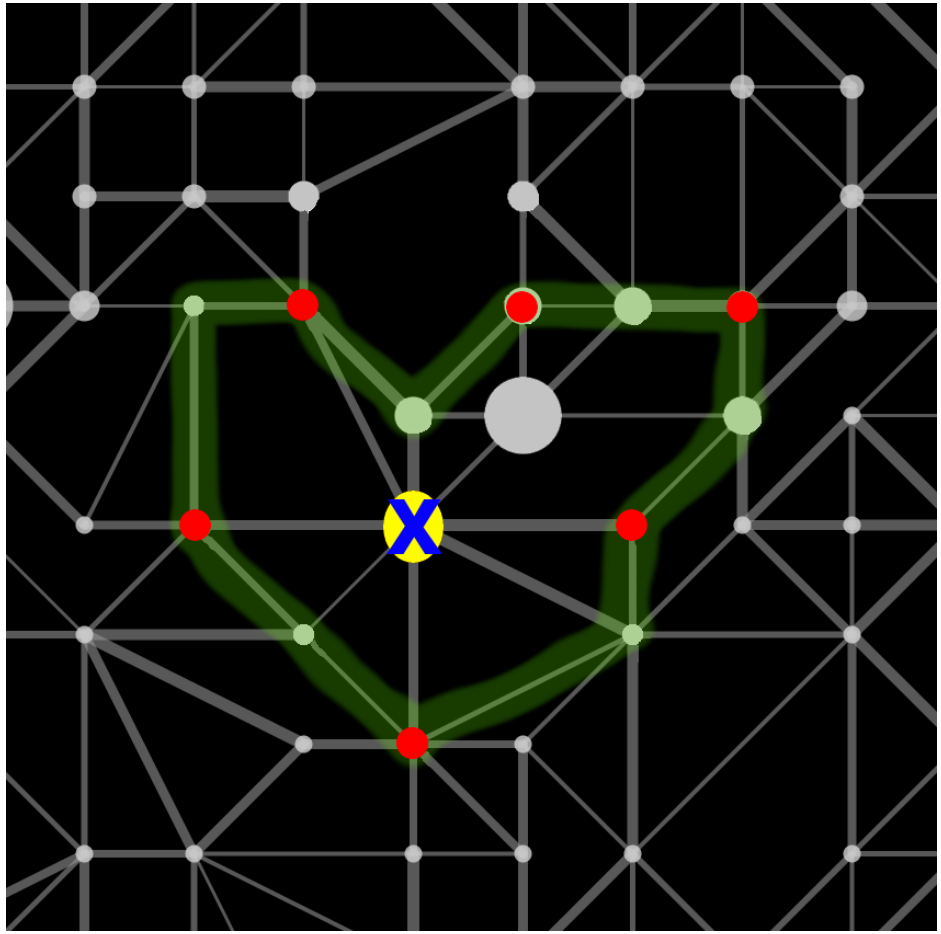
- Two teams send 28 agents each onto Mars
- There are water wells with varying amounts of water
- The teams try to find the most valuable spots and defend them
- Teams may attack each other
- **Goal**: Maximise points given by achievements and occupied wells.

Scenario - Environment

- Weighted graph
  - Weighted edge: costs of traversing this edge
  - Labeled node: value of this water well
- Graph is unknown at the beginning
  - Neither the costs of traversing an edge nor the value of a node are known
  - Agents must explore the graph

Scenario – Occupying Zones/ Graph colouring algorithm





Scenario - Agents

- Attributes
  - Energy, health, strength, visibility range
- Five different roles:
  - Explorer, saboteur, sentinel, repairer and inspector
  - All can skip, goto, survey, buy and recharge
  - May perform various role-specific actions
  - Have different attributes

Scenario - Agents

Role	Special actions	Energy	Health	Strength	Visibility range
Explorer	Probe	35	4	0	2
Repairer	Repair, parry	25	6	0	1
Saboteur	Attack, parry	20	3	3	1
Sentinel	Parry	30	1	0	3
Inspector	Inspect	25	6	0	1

Scenario - Agents

- Agents run on participant's infrastructure
- Simulated environment runs on a remote server
- Agents communicate with the server by exchanging XML messages
- After each simulation step the server delivers new percepts on which the agents then can reason

Our Strategies

- Two overall phases: exploration and zone mode
- We start with exploring the graph
- After that all agents but saboteurs switch to zone mode
- Saboteurs focus on attacking.

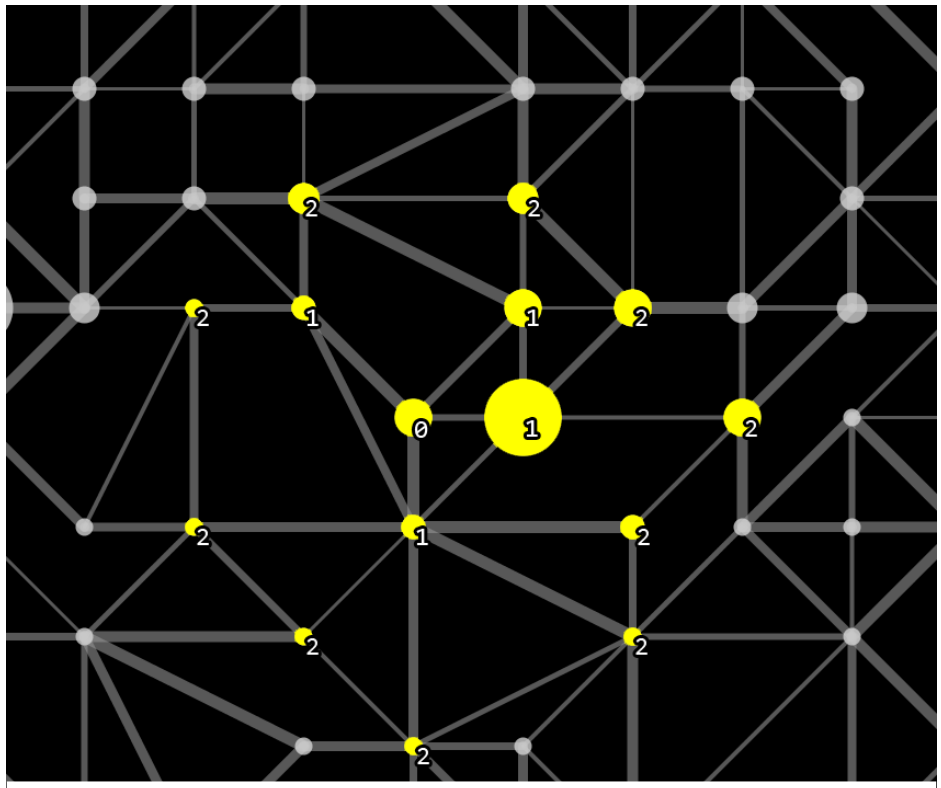
Our Strategies - Exploration

- Before we start with building zones, the graph needs to be explored (DFS)
- Inspector, repairer and sentinel agents perform survey action to retrieve the edge weights of the nodes
- Explorers perform probe action before surveying
- Processed nodes are stored in order to not survey/probe a node twice

Our Strategies – Zone Mode

- Once there are no unsurveyed nodes anymore, agents switch to zone mode
- Agents register for zoning (called "zoners")
- Each zoner calculates his best zone in a given range
- Best means highest zone value with minimal number of agents

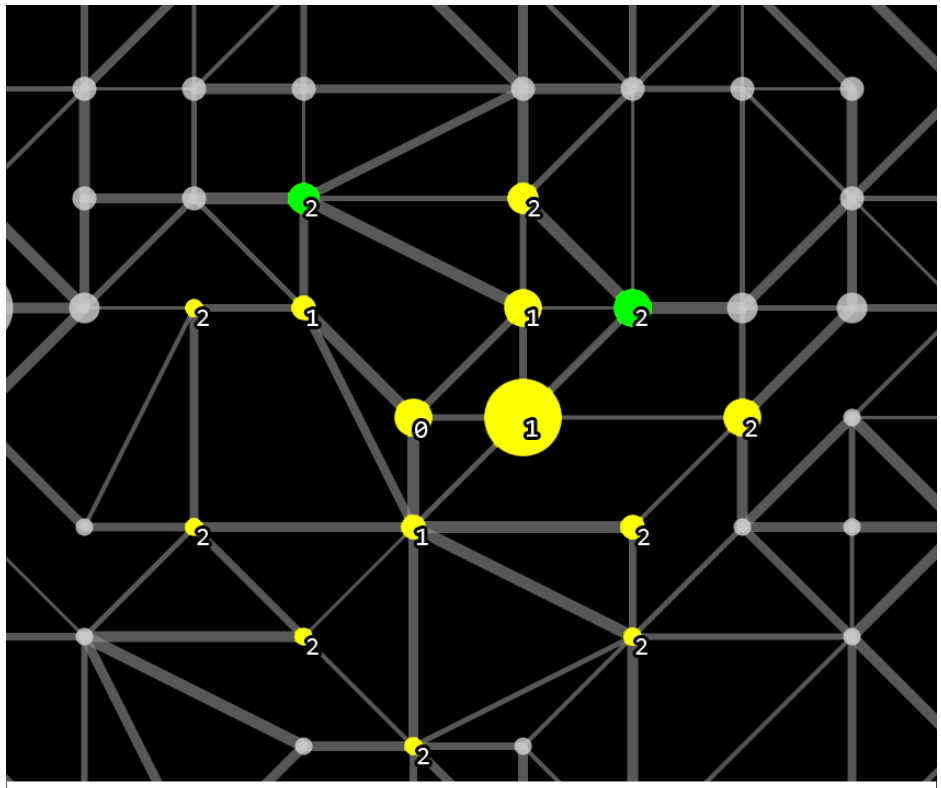
Our Strategies – Zone Calculation



Yellow: vertices to consider
Green: an agent must be placed there
Red: no agent should be placed there

- All vertices in maximum 2-hop-distance marked yellow
- Numbers indicate distance to centre

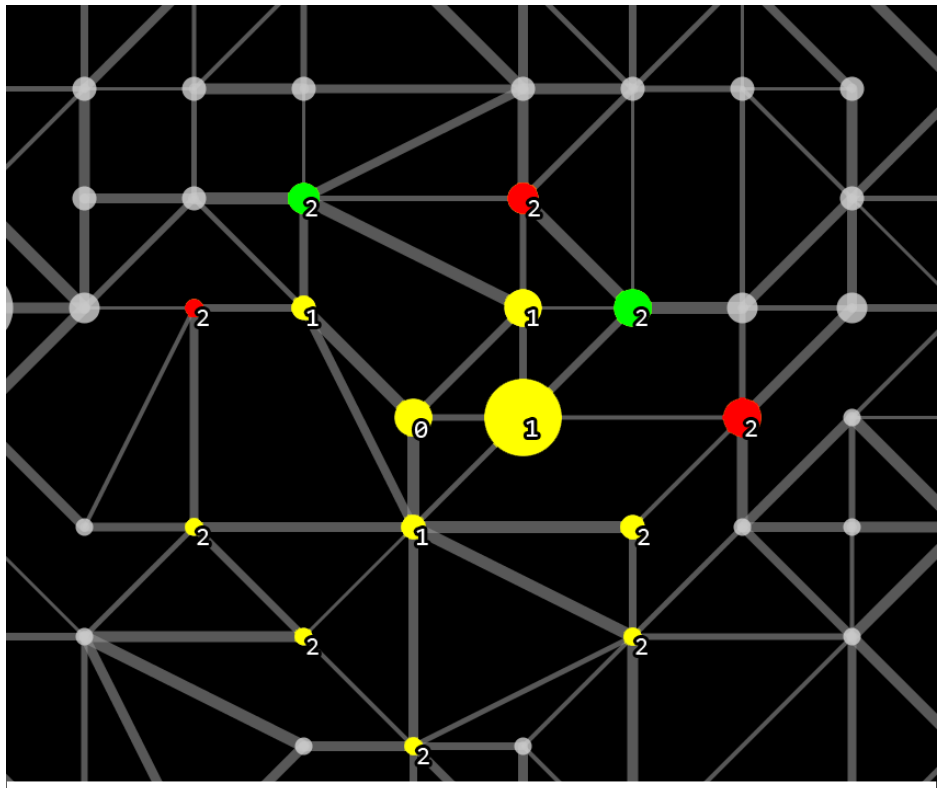
Our Strategies – Zone Calculation



Yellow: vertices to consider
Green: an agent must be placed there
Red: no agent should be placed there

- All 2-hop-vertices connected to 2 or more 1-hop-vertices are marked green

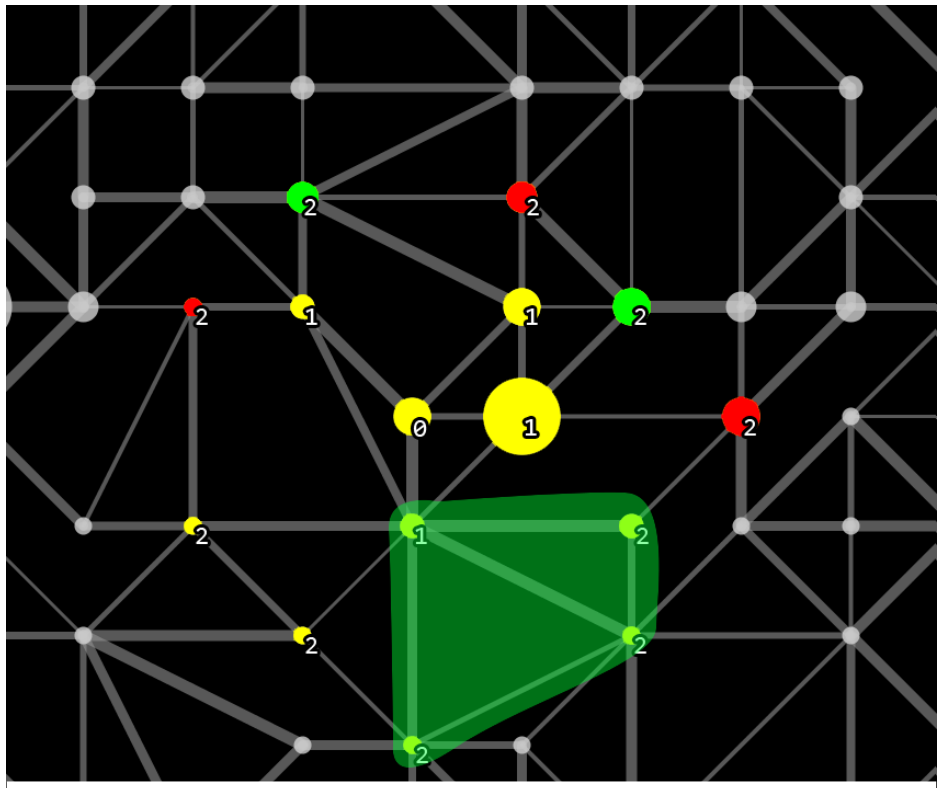
Our Strategies – Zone Calculation



Yellow: vertices to consider
Green: an agent must be placed there
Red: no agent should be placed there

- All 2-hop-vertices connected to a green 2-hop-vertex by maximum 1 hop are red

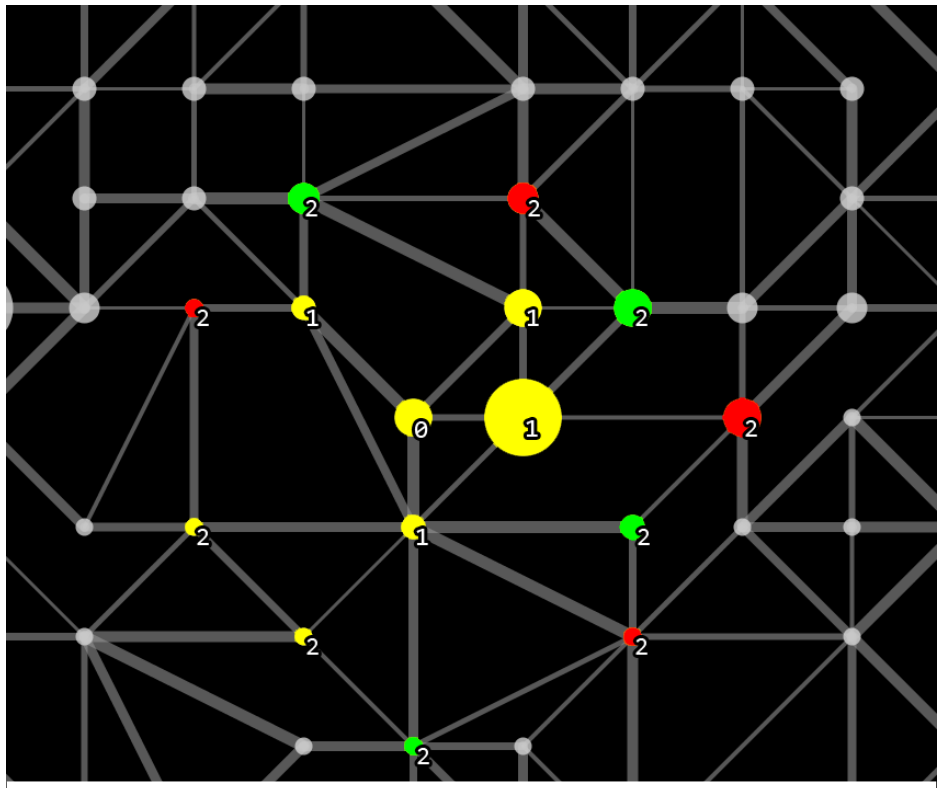
Our Strategies – Zone Calculation



Yellow: vertices to consider
Green: an agent must be placed there
Red: no agent should be placed there

- "bridges":
  - three 2-hop-vertices
  - all connected to a 1-hop-vertex and
  - maximum two hops away from each other

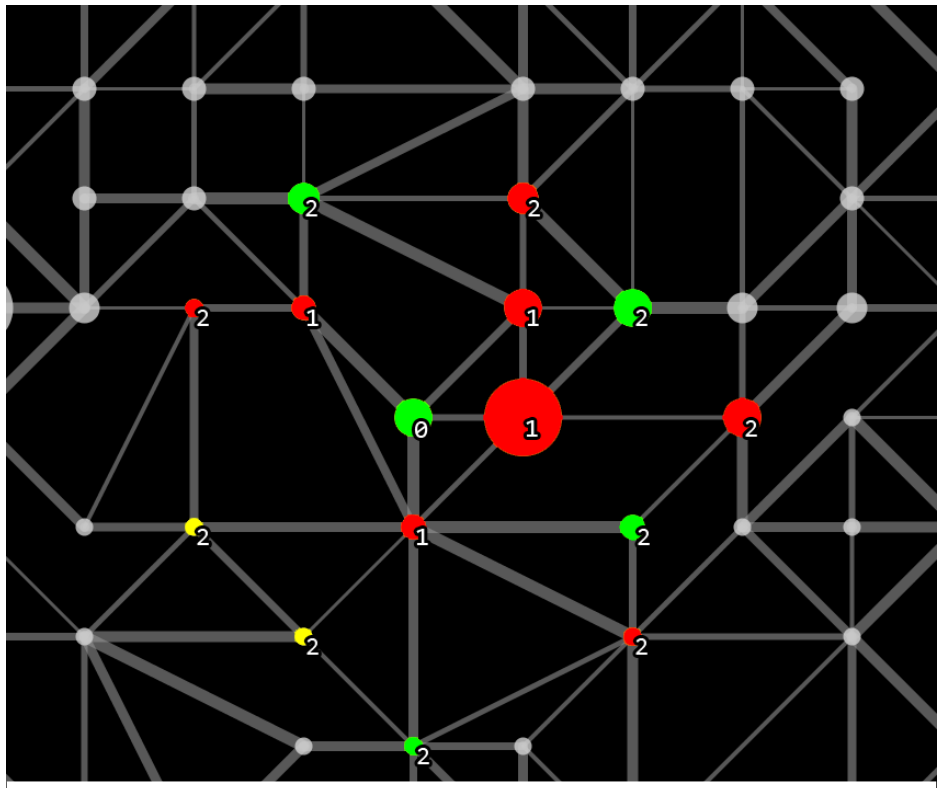
Our Strategies – Zone Calculation



Yellow: vertices to consider
Green: an agent must be placed there
Red: no agent should be placed there

- Found “bridges” have their ends marked green and their connecting vertex red.

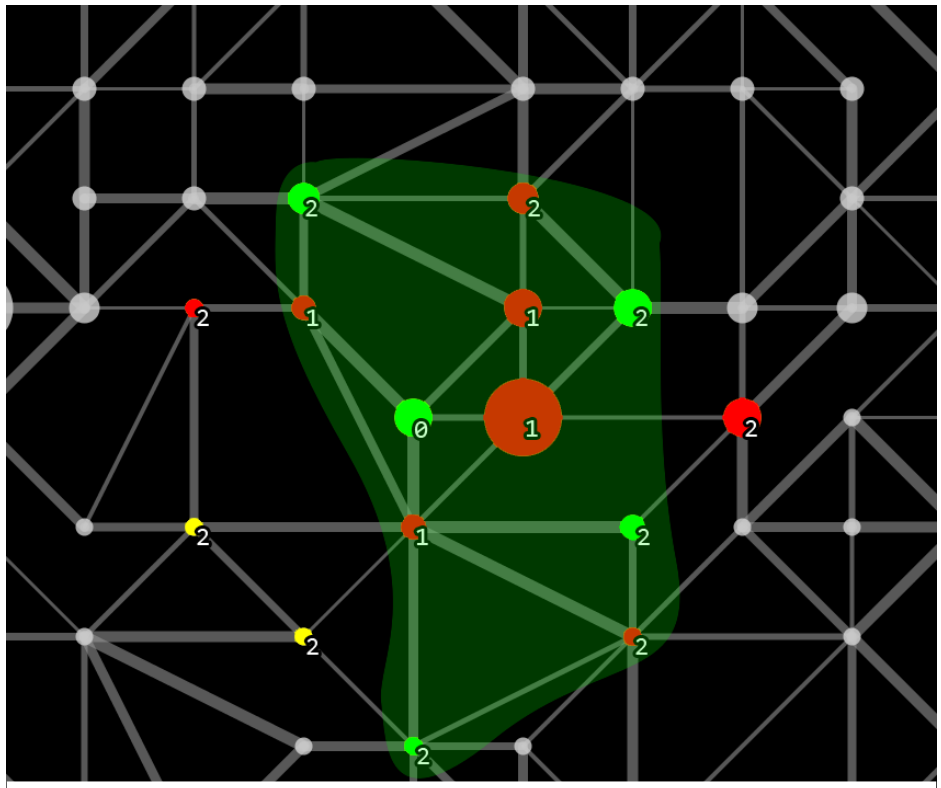
Our Strategies – Zone Calculation



Yellow: vertices to consider
Green: an agent must be placed there
Red: no agent should be placed there

- The centre is marked green as well.
- All 1-hop-vertices are marked red.

Our Strategies – Zone Calculation



Yellow: vertices to consider
Green: an agent must be placed there
Red: no agent should be placed there

- Five agents cover a zone of eleven vertices.

Our Strategies – Zone Mode

- If there are enough agents (zoners) available for zoning do the following:
  1. communicate every zone to every zoner
  2. best zone is picked
  3. use closest zoners to build best zone
  4. all participants of the zone are removed from available zoners
- If not: agent goes to highest value node in certain range

Our Strategies – Explorer Strategy

- Explorers stay longer in zone mode as they keep probing
- Probe in clusters/circles
  - Closest vertex with most already probed neighbours
  - Should help find highly valuable zones.

Our Strategies – Saboteur Strategy

- Defend established zones
- Attack closest enemy agent
- We had one “artillery agent” which upgraded his visibility range every time it did not see an enemy agent

Our Strategies – Repairer Strategy

- If an agent gets disabled he calls the closest repairer
- Respond to calls for help and repair disabled agents
- Disabled agents move to the repairer

  Hence, active zones are not broken up

Implementation details – AgentSpeak(L) example

```
role(saboteur).

+!attack(Enemy):
  energy(X) & X >= 2
  ← attack(Enemy).
+!attack(_)
  ← recharge.
```

Implementation details

- Programming language: Jason, a combination of Java and AgentSpeak(L)
- Plans (reasoning and execution of actions) were implemented in AgentSpeak(L)
- Computationally intense tasks and storing of graph related percepts was done in Java objects
- Interaction between Java and AgentSpeak(L) via Jason's internal actions

Competition outcome

- Team MaKo scored 2nd
- The team from the USFC won three times in a row

Position	Team	Score difference	Matches won
1.	SMADAS-UFSC	526038	11
2.	MAKo	-159782	6
3.	TUB	32475	5
4.	TheWonderbolts	-303668	5
5.	GOAL-DTU	-95063	3

Lessons Learned

- Communication between agents in Jason was an extreme bottleneck
- Recursion is very expensive in AgentSpeak(L)
- When a dash appears in a Literal it is interpreted as an arithmetic expression
- Aggressive strategy was quite effective

End Of Presentation

Thanks for your attention!