The Multi-Agent Programming Contest

A challenge for everyone

Tobias Ahlbrecht Jürgen Dix
Niklas Fiekas Tabajara Krausburg
Department of Informatics, Clausthal University of Technology

WESAAC Lecture, 11 August 2021

Outline

1. What is the MAPC?

Origins

Toni/Torroni (2004): Need for a competition event for

1. **modelling** (problems using logic-based agents)
2. **specifying** (logic-based multi-agent systems, given a problem)
3. **programming/implementing** (logic-based multi-agent solutions)

Dastani/Dix/Novak were lured into it. Jomi Hübner joined later. Current SC: Dix, Hübner, Novak. 15 Agent Contests by now.

Simulation Server: Suitable for problems like simulated mobile/cognitive robotics. **Provide a dynamic environment for those simulated players.**

Aim

- Stimulate research in the area of **agent-oriented programming**
- Identify **key problems** for MAS
- Collect **suitable benchmarks** that can serve as milestones for evaluating new tools, models, and techniques
- Gathering test cases which require and enforce **coordinated actions**

First Contest in 2005

- No **server platform** available.
- Participants were provided with a precisely defined scenario.
- They had to implement the scenario and solve the problem.
- Complete system sent to us for inspection.

Running these systems . . . was a nightmare.

2. The MASSim platform

The MASSim Platform

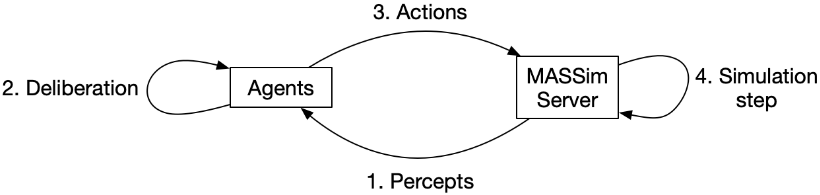
Server: Unified scenario implementation

Agents: Connect remotely (sockets)

- Receive percepts (state of the visible environment)
- Deliberate and send actions back

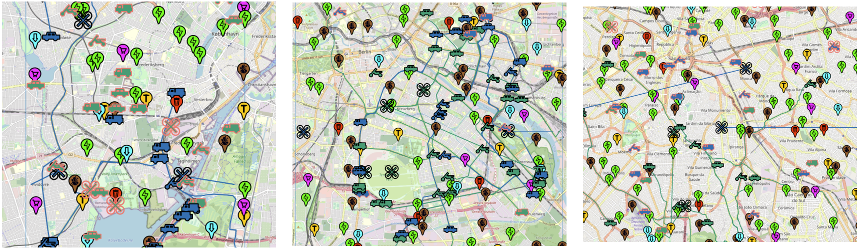
Simulation: Discrete steps

- actions of all agents processed after each step



Tournament

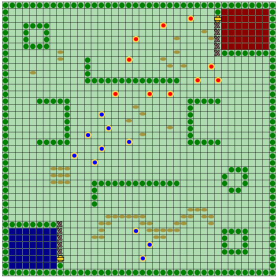
- Games between **2 teams** of agents
- Each team plays against each other
- Deliberation: **4 seconds** per step for network roundtrip and action computation
- **3 simulations** (different parameters per match)
  - environment characteristics
  - agent characteristics (features, number, . . .)

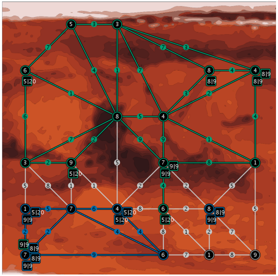


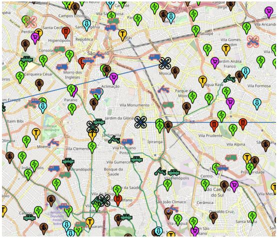
3. Games of the MAPC

Scenarios

- Six **distinct** scenarios so far
- Scenarios **evolve** over multiple years
- Specific **design goals**
  - **Advantages** for agent-based solutions
  - Enforce **cooperation**
  - No single best **winning strategy**
  - **Simple** to start, but **hard** to master

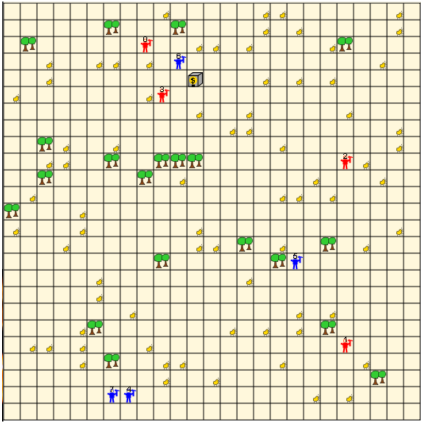


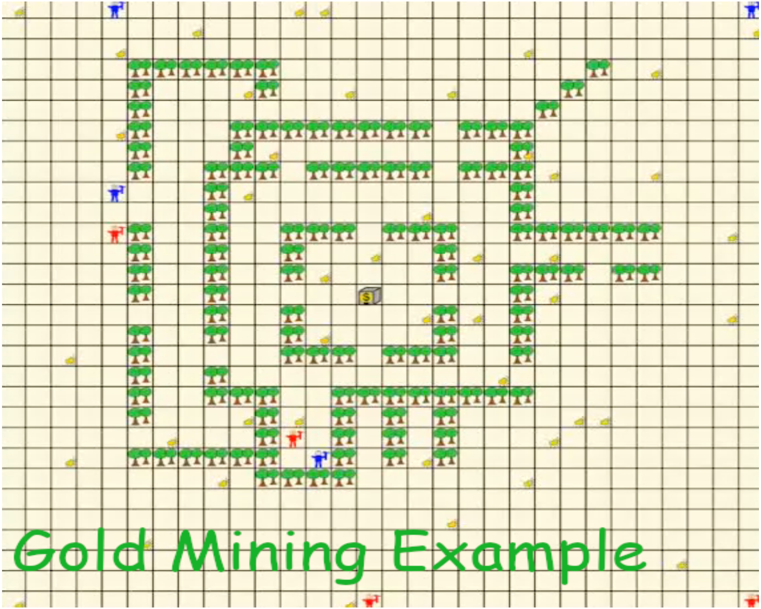




The early days: Food gathering & Gold mining (2005-2007)

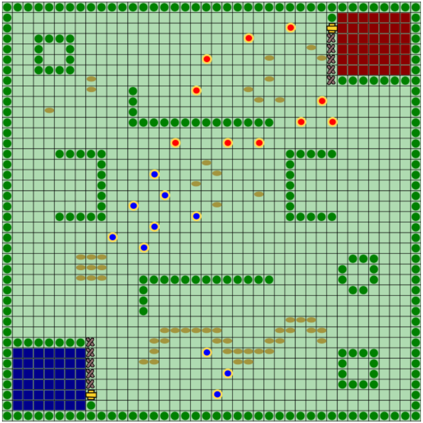
- Simple **grid** environment
- Food/gold appears **dynamically**
- Teams of four agents collect and deliver
- Agents perceive food/gold at their current position only
- Solutions based on **computational logic**

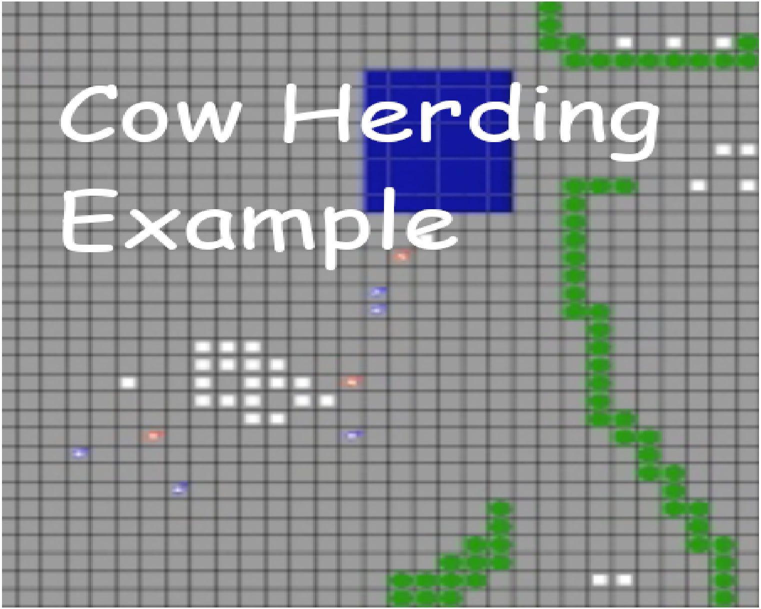



Gold Mining Example

More cooperation: Cow Herding (2008-2010)

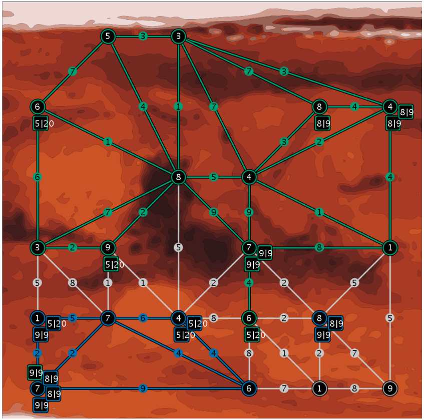
- **Grid** environment again
- Navigation: Obstacles
- Dynamic entities: **Cows**
  - Flocking behaviour
- Cooperation: **Fences**
- Goal: Corrals





Cow Herding Example

Leaving earth behind: Agents on Mars (2011-2014)



- **Graph** environment
- Goal: Occupy zones
- Agents have **roles**
  - e.g. explorer, saboteur, . . .
- Interaction (**attack** and **repair** actions)
- Achievements and upgrades

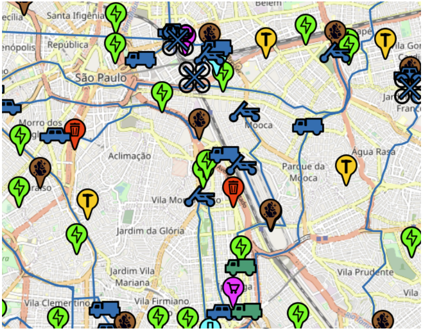
Looking back (but not in anger)

Are MAS really better than traditional programming languages? **Are there killer applications?**

- How to design a scenario?
  - Not: A scenario that can be solved with a smart idea.
  - A scenario that evaluates various aspects of a MAS.
  - **Collaboration** should pay of.
  - Teams should evolve dynamically, not planned ahead.
  - Are some scenarios better suited for MAS?
  - Destructive strategies . . .
- Determinism vs indeterminism: learning agents.
- A game of **two (or more?)** MAS?

Agents in the city (2016-2018)

- **OpenStreetMap**-based environment
- Agents: Different vehicles
- Goal: complete jobs
  - Buy or gather **resources**
  - Assemble items (cooperation)
  - Deliver items
  - Don’t forget to recharge



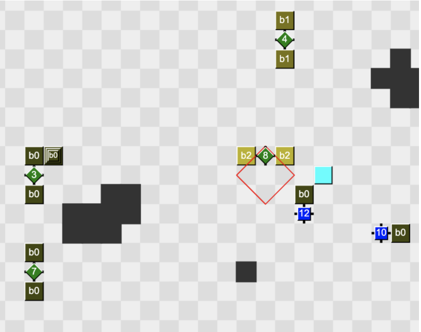
Today and beyond: Agents assemble (2019-?)

Idea: Combine some of the “best” features from previous scenarios.

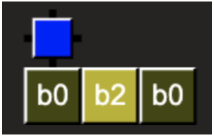
- **Environment structure** (Cow Scenario)
  - Good visibility (what’s going on?)
  - Agents interact and interfere “physically”
- **Goals** (City Scenario)
  - High degree of planning and cooperation required
- Explicit action to **interfere** with opposing agents (Mars Scenario) (`clear`)

Agents assemble

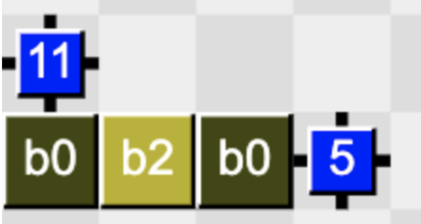
- **Grid** environment
- Agents can carry **blocks**
- **Obstacles** complicate navigation
  - … but can be destroyed!
- Goal: Complete **tasks** to earn points
- **Actions**: move, attach/detach blocks, connect blocks, clear
- Dynamic **events**



Completing tasks

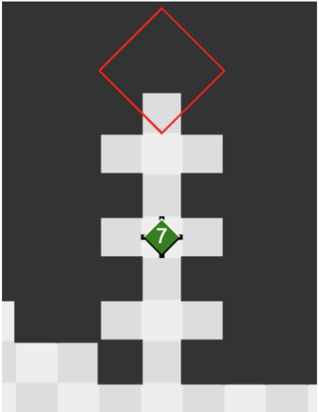


Task description

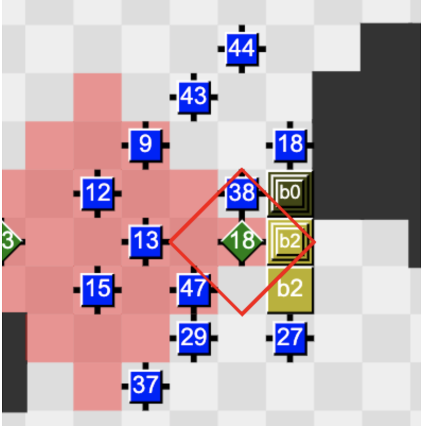


Agents assemble together

Clear action

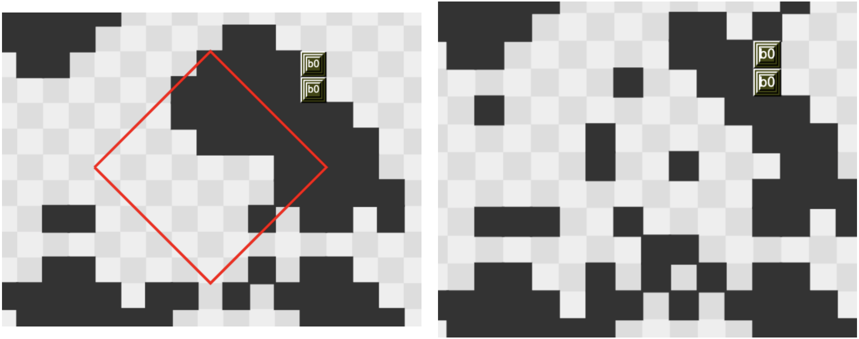


Clears a path



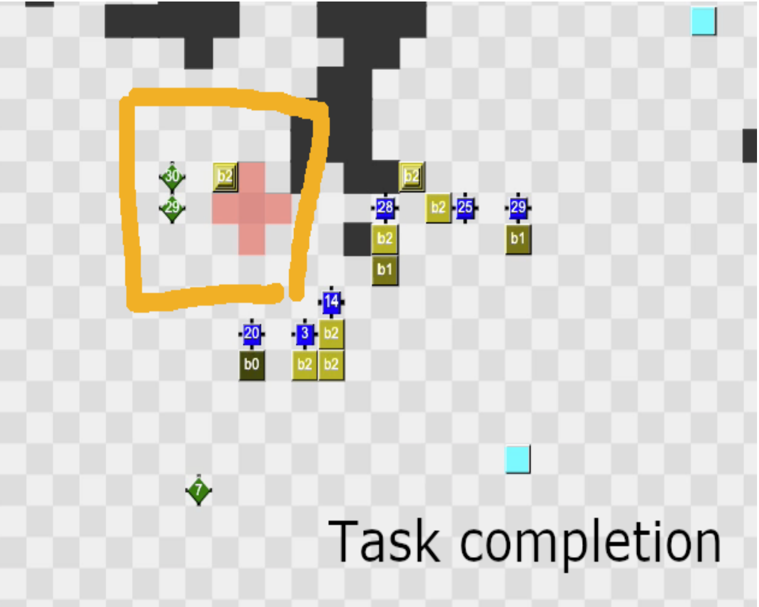
Disables (or scares) agents

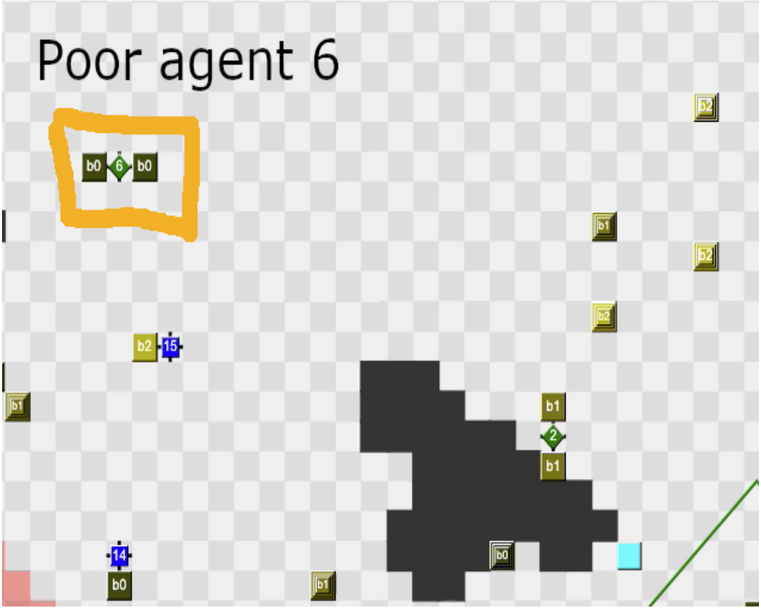
Clear events



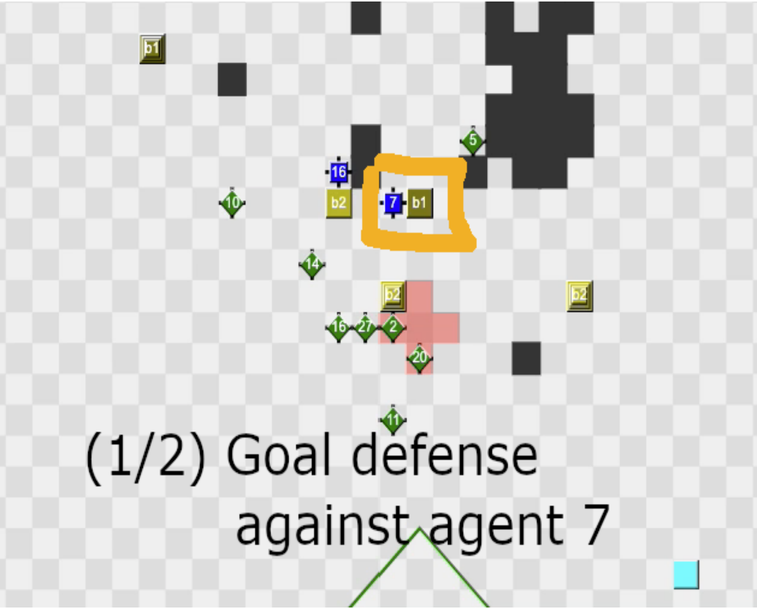
Remove and scatter new obstacles

- Make navigation more difficult
- May force static agents to leave their place

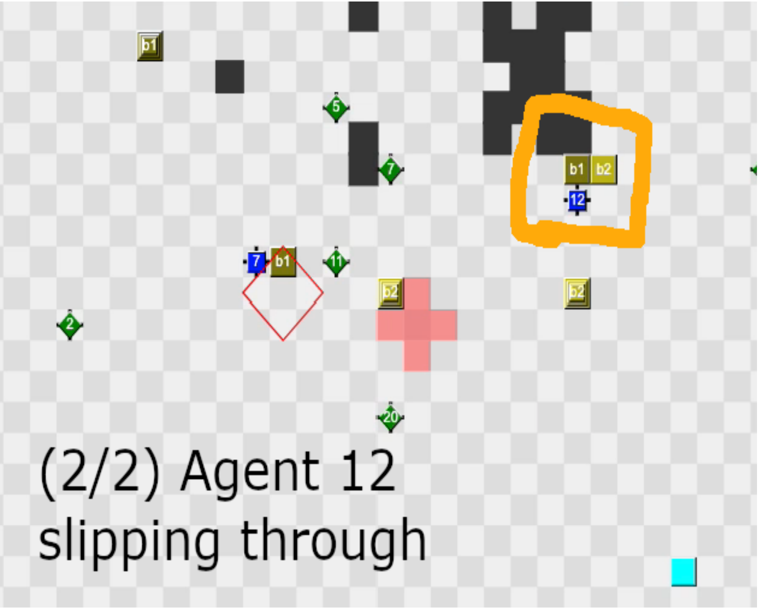

30
29
b2
b2
28
b2
25
29
b2
b1
b1
14
20
3
b2
b0
b2
b2
7
Task completion



Poor agent 6



(1/2) Goal defense against agent 7



(2/2) Agent 12
slipping through

4. Lessons learned

Lessons learned (so far)

- Cooperation needs to be **enforced by all means**
- Agents (and people) tend to take the **path of least resistance**
- **Debugging** MAS is very challenging
- Difficult to discourage **centralized** coordinators
  - Still: mostly agent platforms winning
- **Statistics and visualisations** important for analysis

Why do people participate?

Results from questionnaire:

- Learn about agent programming
- Practice using an agent programming language
- Evaluate (their own) agent platforms, techniques, …

5. Participating

How to participate in the MAPC?

- Contest announcement: September/October
- Contest: Spring 2022
- Join the mailing list
- Join the short course **tomorrow**!

Links

- `https://multiagentcontest.org`
- `https://twitter.com/AgentContest`
- `https://groups.google.com/forum/#!forum/agentcontest`

6. Future of the MAPC

Future of the MAPC

- Try to run agents on the contest server again
  - New technologies like containers etc.
- Test agent **autonomy and robustness** with more games, longer games, . . .
- Improve (semi-)automatic **analysis**
- Experiment with **continuous** (time and space) environments?

Thank you