Final Report
Multi-Agent Programming Contest

Artur Daudrich, Sergey Dedukh, Maunuel Mittler, Michael Ruster, Michael Sewell, and Yuan Sun

University Koblenz-Landau

1 Motivation

2 Scientific Background and Fundamentals

2.1 Agent Programming Concepts

2.1.1 BDI.

2.1.2 Formal Methods.

2.1.3 Negotiation and Argumentation.

2.1.4 Agent Societies.

2.2 Agent Programming Languages

2.2.1 GOLOG and FLUX.

2.2.2 Jadex.

2.2.3 AgentSpeak (L) and Jason.

Why are AS(L) and Jason so awesome that we chose them over the other options? Why didn't we invent our own language or at least our own system/architecture/infrastructure?

2.3 MAPC: Contest and Scenario

What is the general scenario? Agents on mars trying to find water in a competitive manner against another team. Explain the concept of zones, gaining points, achievements and also how agents differ from each other. Introduce the notion of *zoning* as the process of finding, forming, defending and destroying a zone.

3 Team Organisation

3.1 Structure and Meetings

Dynamic working groups that were built weekly to tackle the newly crafted tasks per week. In the beginning, we also tried some hacking sessions. But we quickly found out that working from home works best for us. Plans and discussions were held together in the weekly meetings (on the whiteboard).

3.2 Git, Hangouts and Skype

Revisionsing system, Wiki for minutes and issues for problems. VoIP-solutions for collaborative programming. Not all problems were transformed into issues. Some were just mentioned and discussed in the Hangouts group chat and then quickly solved after.

4 Architectural (?) Structure

4.1 Agents

Talk a bit about generalisation e.g. a saboteur is a specialisation of an agent. I.e. both share exploring but the saboteur also knows how and when to attack. Explain what tasks our agents have and where our priorities are.

4.2 Simulation Phases

Explain the general split up into an exploration and a zoning phase. Also mention the parallel aggressive strategy of the saboteurs. Talk about repairers, explorers and inspectors and their special tasks but without going into details abouttheir complete strategy (this is part of the next chapter).

5 Algorithms and Strategies

5.1 General Strategy Overview

This could also be an introductionary text which motivates the following subsections.

5.2 DSDV

What is it? How is it used in our context? What are advantages we gain from it? What is problematic (speed loss)?

5.3 Exploration

How do agents move around during the exploration phase?

5.4 Zone Forming

Zoning is the most important part in the MAPC Mars scenario [1].It describes the process of agents occupying nodes in a way that they enclose a subgraph. In subsection 5.4.3 we introduce two additional agent roles which are assigned during zoning. Said roles define the tasks and duties of an agent in this phase.

For our approach, zoning should take place after the map exploration phase. This should ensure that enough information about the map has been gathered to calculate high valuable zones close to the agents' current positions. The algorithm for finding these zones and determining which agents have to occupy which nodes is presented in subsection 5.4.1.

The process of forming a zone and the associated agent communication is presented in the last subsection 5.4.2. It features the assignment of zone roles to agents. Furthermore, it is illustrated what zone is to be built and what the agents have to do to achieve this.

5.4.1 Colouring Algorithm for Zone Finding

In some way, we try to find local maxima to build small zones with as few agents as possible. How do we find zones? How do we find out how many agents we need? Explain that extending zones describes how the score resulting from an active zone can be increased by using idle agents. How do we determine what additional spots for zone extensions exist? Present our colouring algorithm and the concept of a centre node.

5.4.2 Zone Negotiation

Zoning happens asynchronously. Who registers when for zoning? How do they unregister? How do agents decide what zone to form? Which agents become coaches and which become minions? How do agents know which node they have to occupy?

5.4.3 Zoning Roles

This subsection describes the two roles exclusive to zoning and their associated tasks and duties throughout the lifecycle of a zone. These roles are those of a coach and a minion. They are assigned when a concrete zone is about to be built. Zoning agents keep either of these roles until the zone is broken up or they have to leave it. The roles regulate the agents' behaviour throughout the time they spent in a zone. Each zone is built by one coach and a varying amount of minions. Minions are agents which are dedicated to build a zone by obeying their coach's orders. Every agent may only be part of one zone at a time.

Before looking at border cases, an ideal case of a zone lifecycle is presented. There, the zone negotiation described in subsection 5.4.2 ends with all agents knowing about the same best zone. This zone was found by one agent which will then become the zone's coach. Next, the coach informs the agents which will be part of the zone where to go to. On receipt of this message, the agents become minions and move to their designated node. The coach will also have to move to his node, which happens to be the centre node of the zone.

We assume due to our colour algorithm for zone finding that a node within a zone will be occupied by at most one agent. Then, any enemy agent endangers a zone. This is because a zone may not spread across an enemy inside of it [1]. Furthermore, enemy saboteurs can disable zoning agents, which similarly destroys the zone in its original form [1]. Hence, coaches check once per step whether an enemy agent is close to the zone. If this is the case, the coach broadcast a message to all saboteurs to come and defend the zone. The saboteurs bid for this with the closest saboteur to the zone's centre winning. He will then move towards the enemy to disable him. If the coach detects in a next step that the enemy moves away from the zone, he will cancel the zone defence.

5.5 Agent Specific Strategies

How are the agents specialised? Explorers keep probing for long. Inspectors probe enemies so that we can avoid saboteurs. Saboteurs are attacking and can be called for zone defence. Disabled agents communicate with repairers and approach them.

6 Implementation Details

6.1 BDI in AS(L) and Jason

Or in general: how did we implement what we had learnt from our scientific background?

6.2 Information Flow

Who gets what information how and when? How do we communicate with the server?

6.3 Lifecycle of one Step

Maybe illustrate what happens within one step and how we prevent multiple actions to be executed in one step.

6.4 Competition results

What place did we rank? How did the others do? Analyse our matches shortly and point out problems we faced, how we tackled them and point out what had gone well.

6.5 Lessons Learned

Here we could explain what was working well and what was troublesome. Java: fast. AS(L): slow and hard for us to program. Communication: extreme bottleneck. Also we could note that all this would need much more time and preparation (or a team that is more familiar with agent programming). We can illustrate this with our approaches of a dedicated cartographer agent and the node agents. We might also illustrate further failed approaches.

7 Conclusion

References

1. Ahlbrecht, T., Dix, J., Köster, M., Schlesinger, F.: Multi-agent programming contest scenario description. Tech. rep. (2014)