Exercício Prático 2: MAPC Agents on Mars

Alan Rafael Fachini[1], Lucas Nascimento[1], and Márcio F. Stabile Jr.[2]

[1] PCS-EPUSP
[2] IME-USP

Resumo Este relatório descreve a implementação de um time usando a abordagem de sistema multi-agente para a competição no senário "Agents on Mars" desenvolvida pelo "Multi-Agent Programming Contest" (MAPC). Neste senários os agentes devem encontrar as melhores zonas com os maiores pesos nos nós do grafo. A implementação apresentada usou como base o sistema "LTI-USP Team" desenvolvida por Franco e Sichman [1] para a competição de 2013, e testes com estratégias diferentes para a organização dos times foram realizados a fim de tentar identificar estratégias mais eficientes.

1 Introdução

O "Multi-Agent Programming Contest"(MAPC) é uma competição realizada todos os anos, onde são propostos problemas de domínios variados para a resolução utilizando sistemas Multi-agentes. Seu principal objetivo é o fomento a pesquisa nessa área de estudo. No MAPC, dois times de agentes competem diretamente no mesmo ambiente, oferecendo assim uma oportunidade para a comparação de estratégias e arquiteturas. Desde 2011 o cenário proposto, Agents on Mars, incentiva soluções baseadas em cooperação e coordenação. O objetivo geral do cenário é o controle de zonas em um mapa, representado por um grafo, por se posicionar os agentes em posições, vértices apropriados. A história de fundo do cenário gira em torno da exploração de marte por áreas com concentração de água.

O presente artigo apresenta a estratégia para o time ALM, desenvolvido com base no modelo do time participante da competição de 2013 do Laboratório de Técnicas Inteligentes (LTI) da USP. O time ALM foi desenvolvido como parte da avaliação da disciplina de Sistemas Multi-Agentes pelos alunos Alan Fachini, Lucas Nascimento dos Santos da Silva e Márcio Fernando Stabile Júnior.

2 System Analysis and Design

O time ALM foi desenvolvido com base no time LTI-USP participante das competições de 2012 e 2013. O time ALM, assim como o time em que é baseado, utiliza a framework de desenvolvimento para sistemas multi-agentes JaCaMo. JaCaMo é uma plataforma para a programação de sistemas multi-agentes que suporta vários níveis de abstração (agentes, ambiente, organização). JaCaMo

combina três tecnologias para isso: Jason , para programar agentes autônomos; CArtAgO, para programar artefatos do ambiente e Moise, para programar organizações de multi-agentes.

Nenhuma metodologia especifica para o desenvolvimento de sistemas multi-agentes foi utilizada, visto nenhum dos membros possuírem experiência para a mesma e dado o tempo necessário para a conclusão do trabalho.

Baseado no time LTI-USP da versão da competição de 2013, nosso time apresenta uma proposta não centralizada. Apesar de apresentar um agente coordenador, cada agente possui autonomia para decidir quais vértices irá ocupar para criar ou expandir uma zona. Cada agente possui seu modelo interno do ambiente e se comunica com outros agentes para informar a estrutura do mapa, posições dos oponentes ou solicitar reparos. De acordo com as regras da competição de 2013 o time é composto de 28 agentes.

A arquitetura dos agentes é baseada no modelo BDI, com cada agente possuindo sua base de crenças, desejos e intenções. Cada agente tem autonomia para decidir sua próxima ação em cooperação com os demais.

Os agentes se comunicam em forma de broadcast enviando, uns para os outros, mensagens somente em relação às novas percepções recebidas, assim os agentes só enviam mensagens nos casos em que a percepção altera o modelo interno do ambiente do agente. A sobrecarga causada por esse tipo de comunicação no começo da simulação, quando as percepções recebidas pelos agentes são novas, diminui com o progresso da simulação conforme os agentes completam um modelo do ambiente.

O ALM é um sistema multi-agente real no sentido que seus agentes são autônomos, proativos e reativos. São autônomos no sentido de decidir por si próprios qual a próxima ação a ser realizada, ainda que em coordenação com outros agentes e com o agente coordenador. Eles apresentam comportamento proativo quando selecionam o melhor vértice no mapa para se movimentarem. E apresentam comportamento reativo quando, por exemplo, o agente realiza uma recarga ao ficar com baixa energia.

3 Software Architecture

A arquitetura do time ALM é a mesma utilizada pelo time LTI-USP na competição de 2013, exibida na Figura 1. Nessa arquitetura foi utilizada a linguagem de programação Jason que provê conceitos de BDI, disponibilizando conceitos abstratos tais como planos, crenças e objetivos. Jason é baseado na plataforma Java e implementa uma extensão da linguagem de programação para agentes AgentSpeak. Os planos que compõem os agentes são definidos em AgentSpeak, e o agente decide qual plano utilizar de acordo com usas crenças e representação interna do mundo.

O modelo interno do mundo utilizado pelos agentes foi criado em Java usando estruturas de dados e classes, que representam todos os aspectos recebidos do simulador. A cada interação da simulação os agentes atualizam seu modelo interno do mundo de acordo com as percepções recebidas do servidor.

TODO Imagem

A framework CArtAgO foi utilizada para usar os artefatos organizacionais disponíveis em Moise. Em CArtAgO o ambiente pode ser desenhado como um conjunto dinâmico de entidade computacionais, chamadas de artefatos.

Moise é um modelo organizacional para sistemas multi-agentes baseado em três dimensões: estrutural funcional e normativa.

Os agentes se comunicam com o servidor MASSim por meio da interface EISMASSim, baseada em EIS, e distribuída para a competição.

4 Strategies, Details and Statistics

1. What is the main strategy of your team?
2. How does the overall team work together? (coordination, information sharing, ...)
3. How do your agents analyze the topology of the map? And how do they exploit their findings?
4. How do your agents communicate with the server?
5. How do you implement the roles of the agents? Which strategies do the different roles implement?
6. How do you find good zones? How do you estimate the value of zones?
7. How do you conquer zones? How do you defend zones if attacked? Do you attack zones?
8. Can your agents change their behavior during runtime? If so, what triggers the changes?
9. What algorithm(s) do you use for agent path planning?
10. How do you make use of the buying-mechanism?
11. How important are achievements for your overall strategy?
12. Do your agents have an explicit mental state?
13. How do your agents communicate? And what do they communicate?
14. How do you organize your agents? Do you use e.g. hierarchies? Is your organization implicit or explicit?
15. Is most of you agents' behavior emergent on and individual and team level?
16. If you agents perform some planning, how many steps do they plan ahead.
17. If you have a perceive-think-act cycle, how is it synchronized with the server?

5 Conclusion

1. What have you learned from the participation in the contest?
2. Which are the strong and weak points of the team?
3. How suitable was the chosen programming language, methodology, tools, and algorithms?
4. What can be improved in the context for next year?
5. Why did your team perform as it did? Why did the other teams perform better/worse than you did.

6. Which other research fields might be interested in the Multi-Agent Programming Contest?
7. How can the current scenario be optimized? How would those optimization pay off?

Short Answers

Please provide short answers to all the questions in a separate section. This does not count for the 10 pages limit. Please use the following style for this section:

```
\newpage
\section*{Short Answers}
\appendix
\section{Introduction}
\begin{enumerate}
\item What was the motivation to participate in the contest?
\item[A:] Our motiviation was ...
\item What is the (brief) history of the team?
(MAS course project, thesis evaluation, $\ldots$)
\item[A:] In 2006...
\end{enumerate}
```

Please note: The `A:` stands for "'Answer"'.

Referências

1. Mariana Ramos Franco and Jaime Simão Sichman. Improving the lti-usp team: A new jacamo based mas for the mapc 2013. In *Engineering Multi-Agent Systems*, pages 339–348. Springer, 2013.