Melhorias nos papéis dos agentes a partir da participação do LTI-USP no MAPC 2013

Guilherme Castro, Felipe Pierin, Andreiwid Corrêa, Nilberto Machado

Escola Politécnica (EP)
Universidade de São Paulo (USP)
{castroguilherme19, felipe.pierin, andreiwid, nilbertomachado}@gmail.com

Abstract. Este artigo tem por objetivo propor melhorias no sistema multi-agentes desenvolvido pelo LTI-USP para participação na edição de 2013 do *Multi-Agent Programming Contest* (MAPC). As melhorias introduzidas serão testadas a partir de competições com grupos de alunos da disciplina PCS 5703 - Sistemas Multi-agentes, de responsabilidade do Prof. Dr. Jaime Simão Sichman. O cenário de competição é interessante para avaliar e comparar diferentes sistemas, assim possibilitando identificar pontos fortes e fracos de cada sistema, promovendo e avaliando o desenvolvimento de todos os participantes. Propõe-se, neste trabalho, conceder uma maior agressividade ao time de agentes, com o intuito de interferir no sucesso da estratégia rival, e aumentar a exploração inicial do mapa, para que as melhores zonas sejam mais rapidamente identificadas e ocupadas, melhorando a pontuação total. Simulações demonstraram que o comportamento almejado foi alcançado por meio de um aumento de 2 vezes na quantidade de vértices explorados e um aumento de 16 vezes na quantidade de ataques realizados com sucesso.

1. Introdução

Anualmente é realizada a competição conhecida como *Multi-Agent Programming Contest* (MAPC) e que tem por objetivo estimular os estudos na área de sistemas multi-agente (SMA). Nesse concurso, diferentes equipes planejam e implementam estratégias particulares para competir por recursos ou outros objetivos específicos de um cenário previamente elaborado pelos mantenedores da competição.

O desafio preparado pelo MAPC para o ano de 2013 foi o de maximização de pontos em um determinado grafo aleatório que representa recursos do planeta Marte. Nele os vértices representam poços de água de valores diferentes e possíveis locais para os agentes. Os pesos das arestas denotam o custo para realizar um travessia e o objetivo final é conquistar o sub-grafo, ou "zona", que garanta a maior pontuação em relação ao adversário com o qual um determinado sistema estará competindo. Cada time é composto por 28 agentes de diferentes funções, sendo, obrigatoriamente, 6 agentes do tipo explorador, 6 do tipo reparador, 6 do tipo sentinela, 6 do tipo inspetor e 4 do tipo sabotador. Os diferentes tipos de agente possuem diferentes atributos e habilidades. Além disso, cada equipe joga contra todas as outros e um jogo é constituído por várias rodadas. O torneio é ganho pela equipe que conseguir o maior número de vitórias [1].

Situado nesse contexto, e incentivados pela disciplina de SMA da EP, alguns grupos foram formados para analisar e desenvolver mecanismos para a competição do MAPC de 2013 com o intuito de aplicar os conceitos aprendidos em sala de aula. Assim surgiu o time formado pelos autores deste artigo e que é composto por alunos de pós-graduação da Escola Politécnica da Universidade de São Paulo, provenientes dos programas de mestrado e doutorado *stricto-sensu*.

Este artigo foi elaborado com a finalidade de melhorar aspectos organizacionais dos agentes do sistema proposto por Franco e Sichman [2] na competição do MAPC de 2013. Neste trabalho, assim como naquele apresentado por Franco, foram usados a plataforma de programação multi-agente JaCaMo [3] e as tecnologias Jason [4], que permite a programação de agentes autonômos, o CArtAgO [5] que possibilita a a construção de artefatos de ambiente e, por fim, o Moise+ [6] para a elaboração das organizações multi-agente.

2. Análise e projeto do sistema

O sistema desenvolvido neste trabalho foi baseado no trabalho proposto por Franco e Sichman [2] e utilizou o JaCaMo [3], um *framework* para a programação de sistemas multi-agente que combina três tecnologias independentes que cobrem todos os níveis de abstração que são necessários para o desenvolvimento de sistemas multi-agente sofisticados: o Jason [4], o CArtAgO [5] e o Moise+ [6]. O Jason é um interpretador para uma versão estendida da linguagem AgentSpeak e que implementa sua semântica operacional [4]. O CArtAgO é um *framework* que torna possível programar e executar ambientes virtuais. Ele é uma infraestrutura baseada no meta modelo de agentes e artefatos para modelar e projetar sistemas multi-agentes [5]. Já o Moise é um modelo organizacional para SMA baseado em noções como papéis, grupos e missões [6].

O SMA proposto segue as regras definidas pelo MAPC. Isso significa que é um modelo constituído por 28 agentes distribuídos entre 6 exploradores, 6 reparadores, 6 sentinelas, 6 inspetores e 4 sabotadores. Cada um com ações, energia, pontos de vida, força e alcance de visão predefinidos [1]. Além disso, tem como objetivo maximizar a pontuação ao ocupar as melhores zonas do grafo proposto.

Neste trabalho, seguindo a linha e pesquisa proposta por Franco e Sichman [2], cada agente decide por si próprio qual vértice irá ocupar a fim de criar uma zona e posteriormente expandi-la. Essa característica é estimulada pela arquitetura SMA baseada em BDI (Belief-Desire-Intention) que permite aos agentes possuir crenças, desejos e intenções próprias desenhando uma visão peculiar do mundo ao qual ele pertence. Dessa maneira, os agentes são autônomos e possuem a capacidade de decidir por eles mesmos sobre qual a próxima ação a ser executada.

Embora os agentes tenham certo grau de autonomia estabelecida, cada decisão tomada leva em consideração a necessidade de cooperação entre eles. É justamente por esse motivo que os agentes se comunicam através dos atos de fala promovidos pela plataforma Jason informando dados relativos a estrutura do mapa, posição de agentes inimigos ou solicitando ajuda. Afinal, é através dessas informações os agentes podem tomar decisões proativas como, por exemplo, calcular o reparador mais próximo e receber reparações quando estiver danificado de maneira mais eficiente.

A autonomia dos agentes, no entanto, é limitada a um coordenador centralizado. No sistema proposto neste trabalho, existe um agente especializado com o papel de coordenador e que possui a responsabilidade de determinar as melhores possibilidades de pontuação no mapa gerado para a disputa entre os times e auxiliar os agentes a alcança-las.

3. Arquitetura do sistema

A arquitetura de sistema proposta neste trabalho segue o modelo ilustrado pela Figura 1 e elaborado previamente por Franco e Sichman [2], durante o desenvolvimento do time LTI-USP. Em outras palavras, a construção de agentes BDI constituídos por planos, base de crenças e modelos individuais de mundo foi alcançada a partir do uso da plataforma Jason para desenvolvimento de SMA. Com esse *framework,* e a partir de planos especificados em *AgentSpeak*, os agentes conseguem decidir o que executar de acordo com crenças e visão de mundos próprios.

O SMA desenvolvido neste trabalho comunica-se com um servidor central chamado MASSim, que permite aos participantes rodar os sistemas localmente e interagir com o servidor do torneio através da internet [8]. É através dela que os projetos competidores recebem dados detalhados relacionados a competição tais como vértices explorados, posições do oponente e pode tomar ações em relação a elas. Neste projeto, o modelo de mundo de cada agente é atualizado a cada passo da competição. Seja por meio de percepções recebidas do servidor central ou por comunicação com outros agentes. Desse modo, algumas informações tais como energia, papel, posição ou recursos monetários do time também acabam sendo guardadas na base de crenças do agente.

A comunicação com o MASSim se dá por meio da interface EISMASSim, distribuída no conjunto do pacote do torneio MAPC. Essa interface é baseada em um padrão de interação agente-ambiente muito robusto chamado EIS [9], que é baseado em Java e define conexões bidirecionais entre plataformas de agentes e ambiente arbitrários. Dessa maneira, são estabelecidas e mantidas de forma automática conexões autenticadas com o servidor e a comunicação entre o servidor do torneio e os agentes são simplificadas a chamadas e *call-backs* da linguagem Java. Por esse motivo, há uma extensão da arquitetura padrão do JaCaMo para perceber e agir no ambiente EIS e não somente nos artefatos CArtAgO [2].

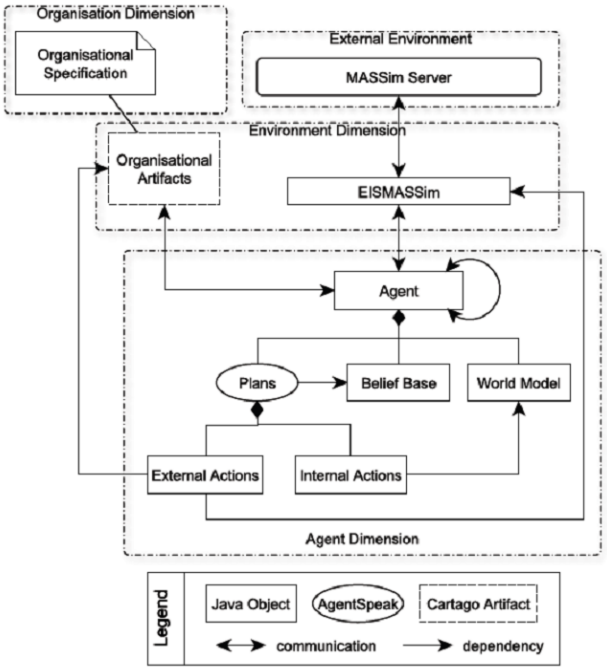

Figura 1. Arquitetura do tirme LTI-USP. Elaborado por Franco e Sichman [2].

O CArtAgO é um *framework* de propósito geral direcionado para SMA. Ele é baseado no meta-modelo de agentes e artefatos e faz uso de metáforas de alto nível abstraídas de ambientes de cooperação humana: agentes como entidades computacionais que realizam atividades orientadas a tarefas ou objetivos e artefatos como recursos e ferramentas. Neste projeto em especifico, seguindo a linha utilizada por Franco e Sichman [2], não houve a necessidade de adicionar nenhum novo artefato já que foram usados apenas os modelos organizacionais providos pelo Moise.

O Moise é um modelo organizacional para SMA que é baseado em noções como papéis. grupos e missões e que permite ao sistema possuir especificações explícitas da organização [6]. Neste trabalho, essa API é aplicada para definir os papéis, grupos e missões dos agentes bem como a auxiliar os agentes a raciocinar a partir da organização a qual eles pertencem.

O código produzido neste trabalho está disponível na internet através de um repositório de códigos fontes Git no BitBucket (https://bitbucket.org/ep2_usp_sma/ep2) e é constituído por aproximadamente 3900 linhas de código distribuídas entre código Java e *AgentSpeak*. O desenvolvimento do trabalho foi elaborado em uma máquina Linux equipada com o Eclipse IDE, o *framework* JaCaMo e o servidor MASSim.

4. Estratégias e resultados

4.1. Estratégia do time de agentes

A partir do sistema proposto por Franco e Sichman [2], e os relatos elencados após a competição oficial do MAPC 2013, o grupo introduziu suas modificações no sistema. Os algoritmos foram adaptados para uma maior exploração inicial do mapa. Percebeu-se que os agentes demoravam para achar as melhores zonas a serem ocupadas, conforme ilustrado na seção referente aos resultados das simulações. Este efeito mostrava-se menor ao final das simulações. Desse modo, por meio do comportamento adaptativo dos agentes do tipo *map_explorer_helper*, os agentes mudam de função após 350 steps. Além disso, configurou-se mais agressividade ao grupo de ataque com algumas redistribuições de papéis na organização dos agentes. As principais diferenças na estratégia do time de agentes está disposta a seguir:

1) Os agentes do tipo *map_explorer_helper* mudam de função após 350 *steps*

(anteriormente eram 250).

2) O agente que antes era do tipo *map_explorer* agora é do tipo *map_explorer_helper*.

3) 2 agentes *explorers* que eram do tipo *soldier* agora são do tipo *map_explorer_helper*.

4) 1 agente *sentinel* que era do tipo *soldier* agora é do tipo *sentinel* (no grupo de ataque).

5) 1 agente *saboteur* que era do tipo *soldier* agora é do tipo *saboteur*.

Da mesma maneira que em [2], foram utilizados três grupos principais: o de ataque, o da melhor zona e o da segunda melhor zona, além de agentes indenpendentes. Porém, diferenças organizacionais concederam um comportamento relativamente diferente ao time de agentes aqui proposto. O SMA proposto possui 5 agentes independentes no inicio da competição, 4 exploradores e 1 coordenador, contra os 3 do time proposto por Franco e Sichman, sendo 2 exploradores e 1 coordenador. No final da simulação, entretanto, o SMA aqui proposto consiste de apenas 1 agente independente, o coordenador, e o time de Franco e Sichman possui 2, 1 coordenador e 1 explorador. Essa característica reflete a escolha de uma exploração inicial mais rápida em busca das melhores zonas de ocupação e uma exploração final menor, dado que o mapa já deve ter sido praticamente todo explorado.

Por sua vez, o grupo de ataque do SMA proposto possui 5 agentes, sendo 1 reparador, 2 sabotadores e 2 sentinelas. Os agentes sabotadores têm a missão de atacar os agentes inimigos, os desabilitando, enquanto os agentes sentinelas têm a missão de se infiltrar nas zonas ocupadas pelos inimigos, as diminuindo, e o agente reparador repara os agentes desabilitados do grupo de ataque. O grupo de ataque proposto por Franco e Sichman [2] possui 3 agentes, 1 reparador, 1 sabotador e 1 sentinela. Dessa maneira, o grupo de ataque aqui proposto tem o dobro de agentes agressivos, com o objetivo de inviabilizar as estratégias inimigas por meio da desabilitação de agentes e da redução das zonas por eles ocupadas.

Em contrapartida, os grupos referentes à melhor zona e à segunda melhor zona propostos por Franco e Sichman [2] são maiores, com 14 e 10 agentes, respectivamente. Dentre eles, cada grupo possui 1 agente do tipo sabotador, com o intuito de atacar inimigos nas redondezas da zona ocupada, assim protegendo-a; 1 agente do tipo reparador, para reabilitar os agentes atacados; e, 1 agente do tipo explorador, com a finalidade de explorar os vértices ao redor da zona ocupada. O número remanescente de agentes possui o papel de soldado, independentemente do seu tipo, cujo objetivo é formar zonas de ocupação. A diferença do time de agentes aqui proposto em relação aos grupos de ocupação da melhor zona e da segunda melhor zona está na quantidade de agentes com papel de soldado, inicialmente 7 para a melhor zona (11 após 350 *steps*) - e 6 para a segunda melhor zona.

4.2. Resultados

Com os objetivos de demonstrar o comportamento e avaliar o desempenho do time proposto, foram realizadas simulações de partidas da competição, com 700 steps cada. O time aqui proposto, assim como o time proposto em [2], competiram contra o time Targaryen, disponível em [1]. Cada time jogou 10 partidas contra o time Targaryen e os resultados são apresentados na Tabela I.

Tabela I. Resultados das simulações contra o time Targaryen.

	Vitórias	**Derrotas**
LTI-USP	7	3

Grupo 2	8	2

Enquanto o time proposto em [2] teve 7 vitórias e 3 derrotas, o time aqui proposto ganhou 8 partidas e perdeu 2. A diferença é pequena, porém, considerável. As derrotas de ambos os times ocorreram quando o time Targaryen criava grandes zonas de ocupação utilizando os cantos dos mapas, destacando um ponto fraco comum em suas estratégias. Apesar disso, o time Grupo 2 obteve um desempenho melhor que o do LTI-USP ao constantemente anular a estratégia do grupo Targaryen com seus agentes de ataque, comprovado pela menor diferença de pontos totais na derrota, cuja média da diferença de pontos foi 32,35% menor para as derrotas do Grupo 2. Para sanar a fraqueza identificada, seria necessário montar um time ainda mais agressivo, o que poderia, entretanto, comprometer o objetivo principal da estratégia proposta, o de dominar as melhores zonas do mapa.

Com a finalidade de comparar de forma mais direta o desempenho dos times LTI-USP e Grupo 2, foram realizadas três partidas entre eles, simulando a final do campeonato proposto pela disciplina de SMA. O time Grupo 2 foi o vencedor de todas as partidas, validando os benefícios da estratégia elaborada. O comportamento de ambos os times está ilustrado nas Figuras 2, 3 e 4, correspondentes a uma das simulações, onde o time LTI-USP é o time A, verde, e o Grupo 2 é o time B, azul. A Figura 2 demonstra a pontuação por zonas ao longo da simulação, enquanto a Figura 3 demonstra a quantidade de vértices explorados. Dessa maneira, observa-se como a exploração de vértices está diretamente ligada à descoberta de zonas de melhores e à pontuação obtida pelos times e comprova-se os benefícios de uma maior exploração do mapa. Por sua vez, a Figura 4 indica a quantidade de ataques realizados com sucesso, validando o comportamento mais agressivo inicialmente almejado.

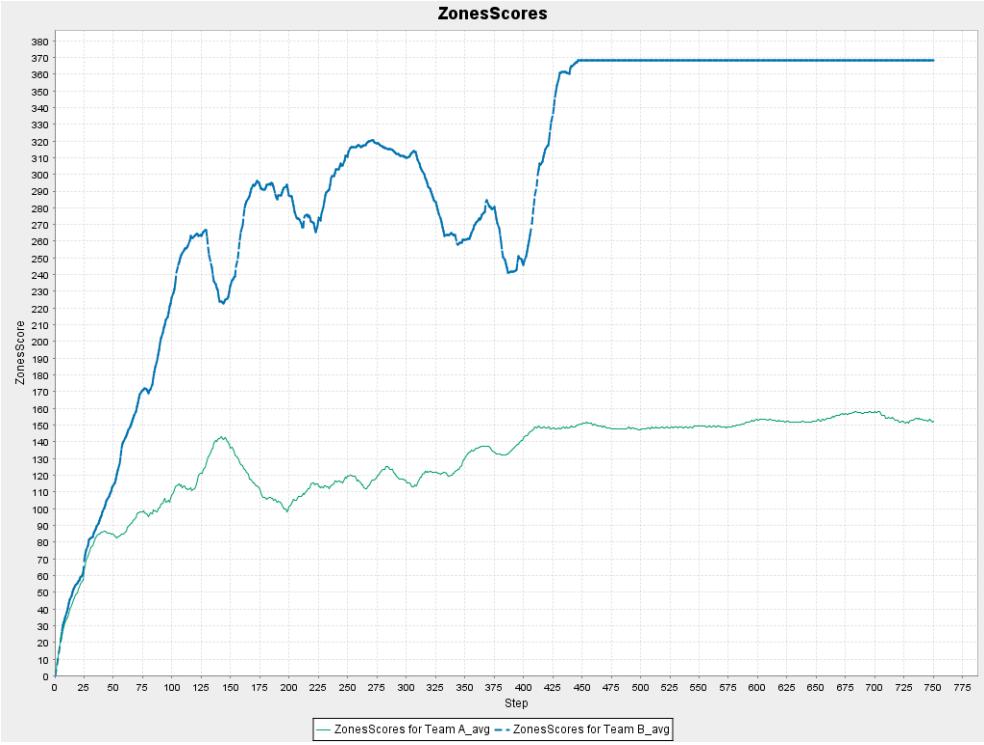

Figura 2. Pontuação por zonas dos times LTI-USP (time A, verde) e Grupo 2 (time B, azul).

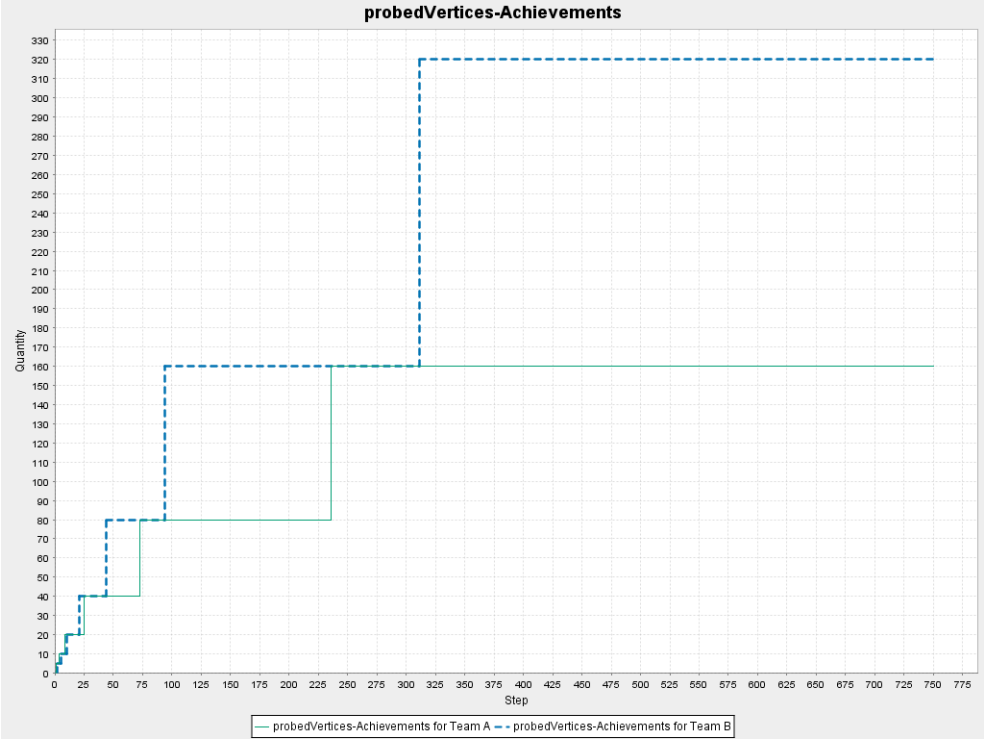

Figura 3. Vértices explorados pelos times LTI-USP (time A, verde) e Grupo 2 (time B, azul).

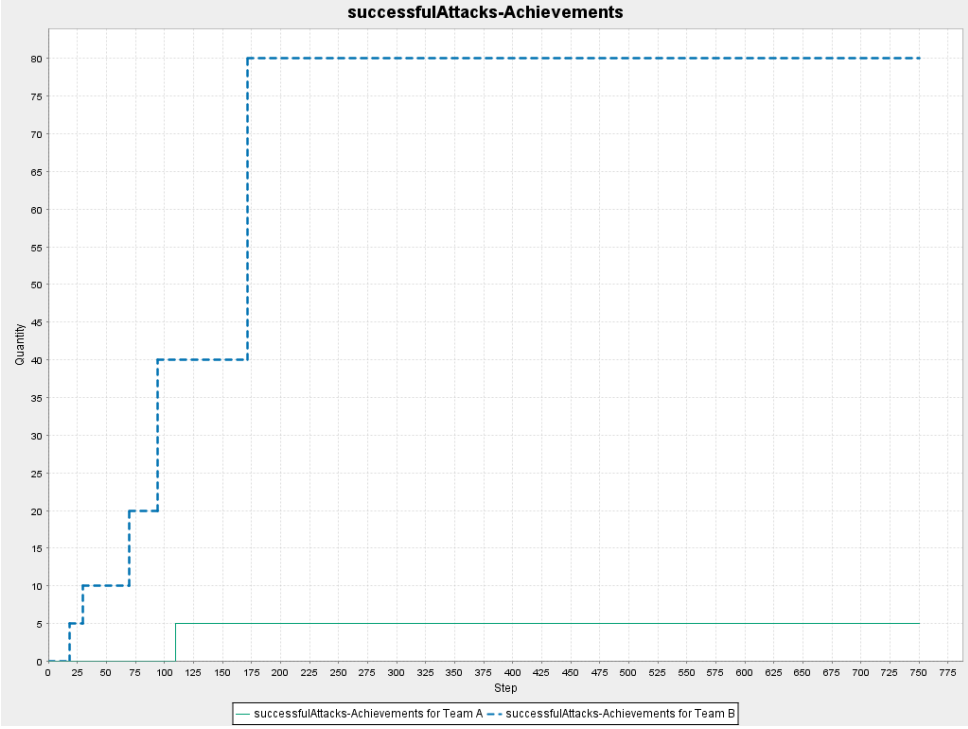

Figura 4. Ataques com sucesso dos times LTI-USP (time A, verde) e Grupo 2 (time B, azul).

5. Conclusão

Em conclusão, o time elaborado nesse trabalho cumpriu seus objetivos inicias de conceder mais agressividade ao grupo de ataque e desempenhar uma maior exploração inicial do ambiente em relação ao time proposto por Franco e Sichman [2]. A alteração de papéis dos agentes na organização do time e a utilização do comportamento adaptativo de alguns agentes, desempenhando papéis diferentes em momentos diferentes da partida, possibilitou a realização

das mudanças estratégicas. Validou-se o comportamento desejado por meio de simulações e seus benefícios foram demonstrados. O número de vértices explorados dobrou em comparação ao time proposto por Franco e Sichman, enquanto o número de ataques realizados com sucesso aumentou em um fator de 16 vezes. Apesar disso, as simulações também evidenciaram a permanência de uma fraqueza identificada no time de Franco e Sichman: a dificuldade de competição quando o time rival domina grandes zonas com o auxílio dos cantos do mapa. Em relação aos objetivos da disciplina e da competição, conclui-se que também foram cumpridos, devido à identificação de pontos fracos e fortes na formulação de estratégias cooperativas para o sistema multi-agente inserido em um ambiente competitivo, e ao aprendizado e avaliação do uso de diferentes ferramentas para sua implementação.

Como trabalho futuro, pode-se investigar os benefícios de ter agentes de ataque individuais, do tipo sentinela, que interfiram na formação de grandes zonas de ocupação - ponto fraco da estratégia proposta identificado por meio das simulações. Além disso, pode-se avaliar a utilização dos cantos do mapa para formar zonas de ocupação maiores em um dos grupos de ocupação, enquanto o outro ocuparia a melhor zona identificada.

References

[1] Multi-Agent Programming Contest. Disponível em: https://multiagentcontest.org/

[2] Mariana Ramos Franco e Jaime Simão Sichman. Improving the LTI-USP Team: A New JaCaMo Based MAS for the MAPC 2013. In Massimo Cossentino, Amal El Fallah Seghrouchni and Michael Winikoff (eds.). Engineering Multi-Agent Systems, vol. 8245 of LNAI series, Berlin, DE, 2013. Springer-Verlag. Pages pp 339-348. Disponível em http://link.springer.com/chapter/10.1007/978-3-642-45343-4_19.

[3] JaCaMo project. Disponível em http://jacamo.sourceforge.net.

[4] Jason - a Java based interpreter for an extended version of AgentSpeak. Disponível em: http://jason.sourceforge.net/wp/

[5] CArtAgO - COmmon ARTifact infrastructure for AGents Open envinronments. Disponível em: http://cartago.sourceforge.net/

[6] Hübner, J., Sichman, J., Boissier, O.: Developing organised multiagent systems using the MOISE+ model: programming issues at the system and agent levels. International Journal of Agent-Oriented Software Engineering (2007) 1–27.

[7] Rao, A.S.: AgentSpeak(L): BDI agents speak out in a logical computable language. In: Proceedings of the 7th European workshop on Modelling autonomous agents in a multi-agent world : agents breaking away: agents breaking away. MAAMAW ’96, Secaucus, NJ, USA, Springer-Verlag New York, Inc. (1996) 42–55

[8] Behrens T.; Dastani M.; Dix J.; Hübner J.; Köster M.; Novák P, Schlesinger F.; The Multi-Agent Programming Contest, 2012. Disponível em: http://aaaipress.org/ojs/index.php/aimagazine/article/viewFile/2439/2333

[9] Environment Interface Standard. Disponível em: http://sourceforge.net/projects/apleis/