Simulando o Jogo de Negociação Pit Game em um Sistema Multi-Agentes Implementado com o Framework JaCaMo

Abilio Castro e Silva[1], **Marcos de Oliveira**[1]

[1]Universidade Federal do Ceará - Campus Quixadá
Quixadá – CE – Brasil

abilio@alu.ufc.br, marcos.oliveira@ufc.br

Abstract. *This work presents the development of a closed multi-agent system implemented in the JaCaMo framework, for agent-oriented programming. The main goal is to perform a simulation as an attempt to evaluate and understand the dynamics of interactions. As well as, the general efficiency of the system in performing assigned tasks, and how agents can perceive and perform actions in a constantly changing environment that resembles a stock market. To simulate a situation in which agents can trade with each other, we used the dynamics of the trading game known as The Pit Game. Also, a graphical interface presents a visualization of the interactions between these agents.*

Resumo. *O presente trabalho apresenta o desenvolvimento de um sistema multi-agentes fechado implementado no framework JaCaMo, para programação orientada a agentes. O principal objetivo do sistema é realizar uma simulação usando o sistema desenvolvido para que se possa avaliar e entender a dinâmica das interações, a eficiência do sistema em geral na execução das tarefas designadas e como os agentes podem perceber e realizar ações em um ambiente que se encontra em constante mudança e se assemelha a um mercado de ações. Para simular um ambiente onde agentes realizam transações comerciais entre eles foi utilizado um ambiente baseado na dinâmica do jogo de negociação conhecido como The Pit Game. Uma interface gráfica foi desenvolvida para uma melhor visualização das interações entre esses agentes.*

1. Introdução

Os agentes inteligentes ou cognitivos são sistemas computacionais capazes de realizar tarefas de forma autônoma com a intenção de alcançar os objetivos para os quais eles foram criados. Ou seja, um agente deve descobrir por conta própria o que ele precisa realizar no sentido de satisfazer as necessidades de seu usuário ou dono. Um sistema multi-agentes(SMA) é composto por um conjunto de agentes que podem interagir entre si, de forma coordenada e cooperativa, na procura pela resolução de uma tarefa ou problema, quando esses agentes possuem os mesmos objetivos. Quando os agentes possuem interesses conflitantes ele podem negociar a melhor saída para atingir seus objetivos. [Wooldridge 2009]

Os agentes devem estar inseridos em um ambiente que fornece informações captadas por um agente específico através de sensores. As informações captadas se tornam percepções a serem utilizadas pelo o agente em sua tomada de decisão, que pode desencadear ou não uma ação a ser realizada pelos atuadores do agente de forma a alterar o ambiente ou até mesmo influenciar outros agentes.

Os SMA podem ser construídos para realizar simulações nas mais diversas áreas. Isto possibilita a recriação de sistemas complexos em que um agente pode representar qualquer entidade que possua algum comportamento autônomo e que interaja com o ambiente e com outros agentes. Em [Norling et al. 2000] procurou-se desenvolver uma simulação multi-agente de sociedades humanas com o objetivo de tornar o comportamento dos agentes mais próximos ao de seres humanos. Para tanto, buscou-se implementar estratégias de tomada de decisão presentes no comportamento humano. Utilizando a linguagem de programação funcional Haskell [Frank et al. 2001] foram construídas simulações multi-agentes com o objetivo de analisar o comportamento espacial cognitivo dos seres humanos e também as suas interações com o ambiente. Uma simulação de escolha de rotas diárias foi realizada em [Liu and Huang 2007] com a auxílio de um sistema multi-agente, permitindo-se assim a análise de duas regras diferentes de compartilhamento de informações entre os agentes. Com o uso do ambiente NetLogo em [Muscalagiu et al. 2013] é apresentado um modelo para implementação e simulação da predição de estruturas de proteínas, onde cada aminoácido de uma proteína de entrada era representado por um agente autônomo que interagia com os demais através da transmissão de mensagens.

Conforme apresentado, percebe-se que o tema da simulação em sistemas multi-agentes vem sendo explorado durante os anos e partindo dessas questões, já levantadas pela literatura, acredita-se que continuar essa exploração em outros domínios possibilitam análises de fenômenos que podem interferir no bom funcionamento de um sistema inerentemente complexo e distribuído. Tendo em vista que muitos desses sistemas encontram aplicações e auxiliam áreas como as ciências sociais, engenharia de transportes, ciências da natureza, entre outras não citadas no escopo deste trabalho.

O principal objetivo desse trabalho é implementar uma simulação na forma de um sistema multi-agentes fechado utilizando o framework JaCaMo [Boissier et al. 2013]. Pretende-se com isso avaliar e entender melhor a dinâmica das interações, a eficiência do sistema em geral na execução das tarefas designadas e como os agentes podem perceber e realizar ações em um ambiente que se encontra em constante mudança, e que se assemelha com um mercado de ações.

Este trabalho foi estruturado de forma que na seção 2 tem-se os trabalhos relacionados com o aqui apresentado, na seção 3 detalha-se como o sistema foi implementado, na seção 4 descreve-se a avaliação geral do sistema e na seção 5 as considerações finais sobre o trabalho.

2. Trabalhos Relacionados

As simulações multi-agentes permite abordar vários campos da ciência e o propósito deste trabalho é investigar em específico características presentes no domínio da ciência econômica, focando ainda no mercado de ações e aspectos de negociações entre agentes. Portanto, nesta seção serão apresentados alguns trabalhos que também exploram esta temática em específico.

O sistema desenvolvido em [Cuni et al. 2004] permite a negociação entre agentes e pessoas de maneira simultânea em vários mercados que comercializam peixes no atacado. Este sistema possibilita a criação, personalização e treinamento dos agentes que serão responsáveis por representar seus donos nas negociações. O MASFIT (*Multi-Agent*

System for Fish Trading) é considerado por seus autores como um sistema real, flexível e robusto capaz de operar e seguir a estrutura organizacional de casas de leilão de peixes tradicionais. Contudo, o MASFIT traz a vantagem de integrar casas de leilão virtuais e reais, permitindo que os negociantes possam operar de forma remota sendo representados ou não por seus respectivos agentes. O sistema foi avaliado através de um ambiente simulado onde os usuários poderiam avaliar se as ações dos agentes estavam em conformidade com suas expectativas, esta simulação inclusive foi importante para avaliar a performance dos agentes em relação à compradores humanos. A relação entre o MASFIT e o trabalho aqui apresentado reside na exploração de simulações multi-agente em um mercado de negociação, entretanto, existem diferenças claras como o fato do sistema resultante deste trabalho ser fechado, ou seja, outros agentes externos e mercados não podem se integrar ao ambiente, além disso, as negociações aqui realizadas são apenas entre os próprios agentes virtuais sem participação de agentes humanos.

O trabalho desenvolvido por [Hoffmann et al. 2007] procurou utilizar a abordagem de simulações sociais baseadas em agentes em conjunto com dados empíricos de economias de nível micro e macro, assim como, suas perspectivas, para desenvolver um mercado artificial de ações. Os resultados encontrados mostraram que esses mercados artificiais de ações viabilizaram formas de se analisar como diferentes processos de comportamentos de nível micro agregam ao fenômeno de nível macro, e também como esses resultados agregados podem influenciar o comportamento individual dos investidores. Neste sistema, os agentes investidores realizam sua tomada de decisão para investimentos através de regras de decisões empiricamente estimadas e interagem socialmente em diferentes estruturas de redes sociais. Dessas interações de mercado, surgem o preço de nível macro e a série de tempo de retorno que são então comparadas com dados empíricos de nível macro. As primeiras comparações mostraram semelhanças qualitativas e quantitativas entre os dados simulados e os dados reais. Este trabalho também explora o tópico da simulação multi-agente aplicado ao mercado de ações, mostrando como tais simulações podem ser de grande utilidade na procura por um melhor entendimento de como certos comportamentos de agentes podem influenciar nas demais partes que compõem um sistema complexo, permitindo ainda que se possa realizar uma comparação entre as simulações e as situações reais.

Utilizando-se dos conceitos presentes nas redes elétricas inteligentes ou *Smart Grids*, que preveem a possibilidade de uma rede de fornecimento onde pequenos geradores de energia podem negociar entre si enquanto lidam em tempo real com o dinamismo da demanda e oferta por eletricidade, [Vytelingum et al. 2010] propuseram um mercado de eletricidade que opera em uma configuração na qual os agentes negociam de maneira não cooperativa e, mesmo assim, o sistema evita a sobrecarga nas linhas de transmissão. Os agentes também são impossibilitados de realizarem negociações no mercado futuro visando lucro em transações entre dias. O alto nível de eficiência alcançado pelo sistema foi demonstrado por [Vytelingum et al. 2010] através de uma variedade de ambientes simulados que forneceram novas estratégias de negociação das quais se pode gerar uma eficiência de 99% do mercado e um limite inferior de 86% quando comportamentos simples são usados no sistema. Este novo mercado de eletricidade explorado no trabalho apresentado e que ainda está em desenvolvimento, assim como o conceito de *Smart Grid*, demonstra como a simulação multi-agentes se encaixa em mercados emergentes trazendo novos conhecimentos que auxiliam a forma como tais mercados podem ser modelados.

Procurando analisar o comportamento de diferentes tipos de negociantes ou *traders* no complexo mercado de ações [Bloembergen et al. 2015] analisou as estratégias adotadas por três tipos de *traders*: fundamentalistas, chartistas e os sem informação (*zero-information traders*). Os fundamentalistas estimam os preços das ações baseados no pagamento de dividendos ou lucro da empresa, os chartistas usam tendências para tentar prever futuras direções de preços e os sem informação baseiam seu comportamento de negociação apenas no preço atual do mercado. Essas estratégias foram encapsuladas em agentes e simulações de mercados de ações foram executadas com várias combinações de agentes representado os *traders*, as forças evolucionárias desses agentes foram comparadas utilizando tabelas de recompensas heurísticas (*heuristic payoff tables*) e a dinâmica do replicador da teoria evolucionária dos jogos. Os resultados obtidos por [Bloembergen et al. 2015] mostraram que não é fácil prever qual será a melhor estratégia de negociação, entretanto, percebeu-se que os fundamentalistas superavam os demais *traders* e os expulsavam para fora do mercado quando a informação era disponível gratuitamente, enquanto que os chartistas prosperavam quando as informações fundamentais eram caras. O desenvolvimento deste trabalho mostra a viabilidade de se analisar mercados de ações e principalmente o comportamento dos negociantes por meio de simulações multi-agente, possibilitando a observação de quais fatores que poderiam influenciar no sucesso ou falha das estratégias de negociação em um mercado complexo.

Recentemente, [Giulioni 2019] propôs uma ferramenta baseada em agentes para modelar mercados de commodities e realizar simulações envolvendo esses mercados. O software apresentado, desenvolvido na linguagem de programação Java utilizando a plataforma para modelagem de agentes Repast Simphony[1], entre outras funções permite que se monitore com o passar do tempo a formação de preços nos mercados, a variação de preço e quantidade negociada em um mercado específico, a quantidade de pagamentos realizados por um determinado negociante para os demais, a destinação de vendas de um negociante e o mapeamento das relações comerciais entre negociantes. As simulações executadas ilustram como este sistema pode ajudar na obtenção de conhecimento detalhado sobre a dinâmica do modelo implementado. [Giulioni 2019] afirma que com o uso de informações mais realistas a modelagem de mercados de commodities neste sistema seria de grande utilidade para a análise dos efeitos significantes de mudanças no contexto econômico, tais como choques de produção relevantes, proibições de importação e exportação, mudanças nas políticas de governo, entre outros fatores.

3. O Sistema Multi-Agente Implementado

Para ilustrar a interação entre os agentes utilizou-se a dinâmica do jogo de cartas conhecido como *The Pit Game*, assim como em [Purvis et al. 2003]. Considera-se que a forma com a qual os participantes desse jogo negociam entre si, simulando um mercado de ações, configura-se como uma opção válida para colocar em prática e avaliar os conceitos, técnicas e padrões de comunicação em sistemas multi-agente. As regras, papéis e demais detalhes sobre esse jogo serão apresentadas na subseção 3.1. O framework JaCaMo para programação orientada a multi-agente foi utilizado para a implementação do sistema. Esse framework é uma união de três tecnologias: Jason para a programação dos agentes autonômos, CArtAgO para a programação de artefatos que representam o ambiente compartilhado em que os agentes interagem e MOISE para a programação de

[1]Disponível em: https:// repast.github.io acessado em: 27 de julho de 2019

organizações de agentes [Boissier et al. 2013]. A subseção 3.2 descreverá melhor como os agentes foram arquitetados através do Jason e a subseção 3.4 apresentará como o artefato que representa o ambiente compartilhado foi definido no CArtAgO de forma a permitir a integração com a interface gráfica. A comunicação entre os participantes do jogo será detalhada na seção 3.3. Tendo em vista que esta simulação não pretende trabalhar com organizações de agentes as opções fornecidas pelo MOISE não foram utilizadas. O código do projeto está disponível em um repositório do GitHub[2]

3.1. The Pit Game

The Pit Game é um jogo de cartas que permite aos participantes realizarem ofertas (*bids*) de uma certa quantidade de *commodities* e também aceitarem outras ofertas. Tais *commodities* são representadas por cartas, cada tipo de carta possui uma pontuação específica e uma quantidade máxima de 9 cartas. No início do jogo um distribuidor (*dealer*) entrega para cada jogador, de forma aleatória, um conjunto de 9 cartas e se inicia então uma rodada de negociações. Quando um jogador realiza uma oferta ou *bid* e um outro jogador aceita essa oferta ocorre uma troca de cartas. O número máximo de cartas que pode se ofertado por vez é 4. Após algumas interações de maneira que os jogadores possam trocar várias cartas, o jogador que conseguir todas as 9 cartas do mesmo tipo e anunciar primeiro que venceu comunicando um *corner* será o vencedor da rodada e ganhará a pontuação do tipo utilizado para vencer. O jogo é finalizado quando algum jogador acumula uma pontuação maior ou igual a 500 pontos. Na implementação utilizada neste trabalho a quantidade de jogadores e consequentemente dos tipos de cartas foi limitada a três.

3.2. Arquitetura dos Agentes

Por ser uma extensão da linguagem de programação orientada a agentes *AgentSpeak*, baseada na arquitetura *Belief-Desire-Intention* (BDI), também é possível programar em Jason agentes que seguem essa mesma arquitetura [Bordini et al. 2007]. Nesse sentido, os agentes implementados no sistema possuem crenças iniciais que são estados mentais dos jogadores e podem inclusive representar características da ontologia do jogo.

As crenças mais simples são representadas por termos e as mais complexas por estruturas da linguagem *AgentSpeak*, tais termos e estruturas se assemelham com as formas de representação do conhecimento em linguagens de programação lógica como Prolog. Outra parte importante que compõe os agentes são os planos, construções que permitem aos agentes performar ações no ambiente ou alterar alguma crença visando alcançar um determinado objetivo quando um dado contexto (uma consequência lógica de um subconjunto da sua base de crenças) é verdade. No âmbito deste trabalho os planos foram implementados nos agentes para realizar estratégias de vitória e ações próprias da dinâmica de comunicação do jogo.

Todos os jogadores implementam as mesmas estratégias, primeiramente eles iniciam procurando trocar o tipo de carta que possuem em menor quantidade e de uma única vez, caso essa estratégia não funcione após um determinado tempo é chamada a estratégia de diminuir a quantidade de cartas que foram ofertadas da última vez, novamente se essa estratégia falhar uma estratégia de realizar uma oferta de um tipo aleatório é chamada e por fim, caso essa estratégia aleatória falhe a estratégia inicial será reiniciada. O distribuidor (*dealer*) também possui crenças iniciais e planos, assim como um jogador (*player*)

[2]Disponível em: https://github.com/abiliocastro/thepitgame

mas, como sua função é mediar as interações do jogo o seu conjunto de planos é bem divergente dos planos de um *player*.

Nas tabelas 1 e 2 estão classificadas respectivamente as crenças e os planos para o jogador (*player*) e nas tabelas 3 e 4 tem-se a classificação respectivamente das crenças e planos do distribuidor (*dealer*). É importante lembrar que os planos listados nessas tabelas estão representados apenas pelo seu evento disparador, cada plano pode possuir mais de um contexto em que é aplicável e com um corpo de ações diferente para um contexto em questão. Além disso, um plano também possui o seu plano de falha, uma alternativa para quando a sequência de ações do plano não pode ser concluída.

Tabela 1. Classificação das crenças iniciais para um agente *player*

Representação do Ambiente	total_cartas(0).
	cartas(milho,0,60).
	cartas(feijao,0,80).
	cartas(trigo,0,100).
	score(0).
Outros Estados Mentais	aceitando_bids.
	timeout_bid(0).
	ultimo_tipo(null).
	ultimo_aceitei(null).
	esse_nao(null).
	esperar_acoes(1500).

Tabela 2. Classificação dos planos para um agente *player*

Ações no Jogo	!receber_cartas(Tipo, Q)[source(dealer)]
	!add_cartas(Essa, Destanto)
	!ganhar
	!iniciar
	!make_bid(Tipo,Qtd,Valor)
	!receber_bid(Qtd,Player,Valor)[source(dealer)]
	!recompor_mao(TipoAdd,TipoRet,Qtd)[source(dealer)]
	!corner(Player, Tipo)[source(dealer)]
	!corner
	!zerar_crencas[source(dealer)]
	!reiniciar[source(dealer)]
Estratégias	!estrategia_inicial
	!estrategia_diminuir
	!estrategia_aleatoria
	!incrementar_timeout
	!resetar_timeout
	!atualizar_ultimo_tipo(Tipo)

3.3. A comunicação entre os agentes

A maneira como ocorria a comunicação entre os agentes neste trabalho era centralizada, pois o *dealer* era o responsável, por exemplo, por receber as ofertas e distribuí-las para

Tabela 3. Classificação das crenças iniciais para o agente *dealer*

Representação do Ambiente	cartas(milho,9,60).
	cartas(feijao,9,80).
	cartas(trigo,9,100).
	tipos([milho,feijao,trigo]).
	players([agente1,agente2,agente3]).
	score(agente1,0).
	score(agente2,0).
	score(agente3,0).
Outros Estados Mentais	espera(1500).
	distribuir([agente1,agente2,agente3]).

Tabela 4. Planos para o agente *dealer*

Ações no Jogo	+distribuir([Players])
	+!bid(Tipo,Qtd,Valor)[source(Player)]
	+!aceitar_bid(Player,Qtd)[source(PlayerAccepted)]
	+!corner(Tipo)[source(PlayerCorner)]
	+!nova_rodada

todos os outros *players*. Ou seja, os jogadores não negociavam diretamente entre si, mas sim com a intermediação do *dealer*.

Essa concentração da comunicação no *dealer* tinha por objetivo facilitar o gerenciamento do fluxo do jogo e sua sincronia, uma vez que todos os jogadores precisavam receber uma mensagem do *dealer* para iniciarem as ações de receber as cartas, iniciar as negociações, receber ofertas, recompor a mão, receber a notícia de que outro jogador ganhou e reiniciar as crenças para uma nova rodada. O *dealer* também foi programado para receber de um determinado *player* mensagens informando: uma oferta, o aceite de uma oferta e um anúncio de *corner*. Quando uma oferta (*bid*) ou anúncio de *corner* eram recebidos pelo *dealer* este deveria comunicar essas informações para todos os agentes que deveriam assim avaliar o seu interesse em relação às ofertas e se preparar para um nova rodada no caso do *corner*.

O Jason possibilita que tais interações entre agentes sejam realizadas através de ações internas como `.send(agente,performativa,conteudo)` para enviar o `conteudo` proposicional a um `agente` utilizando uma `performativa` e `.broadcast(performativa,conteudo)` para enviar o `conteudo` à todos os outros agentes utilizando também a `performativa` definida. Uma `performativa` irá informar ao agente recebedor qual a intenção do agente que enviou a mensagem.

Sabe-se que a semântica de uma `performativa` em Jason se assemelha com a semântica da linguagem de comunicação de agentes *Knowledge Query and Manipulation Language* (KQML), tal linguagem é baseada na teoria dos atos de fala [Bordini et al. 2007]. Nesse sentido, utilizou-se para a troca de mensagens entre os agentes a performativa `achieve` que tem o propósito de informar ao agente recebedor que o `conteudo` da mensagem trata-se de um objetivo a ser alcançado.

3.4. Integração com o Ambiente Gráfico

Procurando uma melhor visualização das interações entre os agentes desenvolveu-se uma interface gráfica em Java com o auxílio de uma biblioteca para aprendizagem de programação de jogos chamada JPlay[3].

O motivo para o desenvolvimento dessa interface foi a dificuldade em se acompanhar apenas por texto, no console do JaCaMo, o fluxo do jogo. Para tanto, se fez necessário a criação de um artefato CArtAgO com o intuito de permitir que as ações dos agentes fossem refletidas no ambiente gráfico. Nesse artefato CArtAgO foram definidas operações em métodos Java que solicitavam a uma classe estática nomeada *Jogo*, controladora das ações na interface gráfica, a chamada de métodos que realizavam ações como distribuir cartas, atualizar a pontuação e entre outros.

O diagrama UML da Figura 1 apresenta o artefato CArtAgO *GameEnv* se relacionando com a classe *Jogo* que agrega as demais classes que são entidades do domínio do jogo e necessárias para a composição do ambiente gráfico. A Figura 2 mostra a interface gráfica logo após uma primeira rodada do jogo onde o `agente2` já possui 100 pontos, pode-se observar também a janela do console do JaCaMo sobreposta sobre a interface para a visualização de mais informações sobre o sistema.

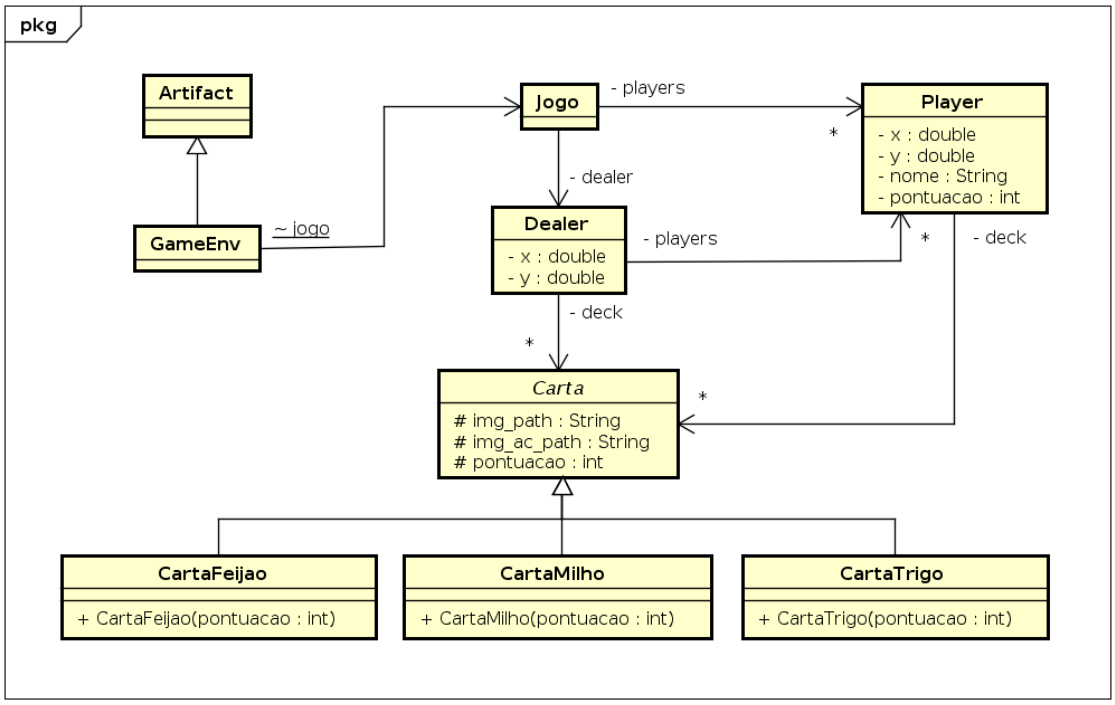

Figura 1. Diagrama UML das Classes da Interface Gráfica

4. Avaliando o Sistema

Durante o desenvolvimento deste sistema multi-agente muitos testes foram executados e conforme a complexidade das interações entre os agentes aumentava percebeu-se o surgimento de alguns problemas de comunicação. No início da implementação de tarefas

[3]Disponível em: http://www2.ic.uff.br/jplay/index.html acessado em: 28 de julho de 2019

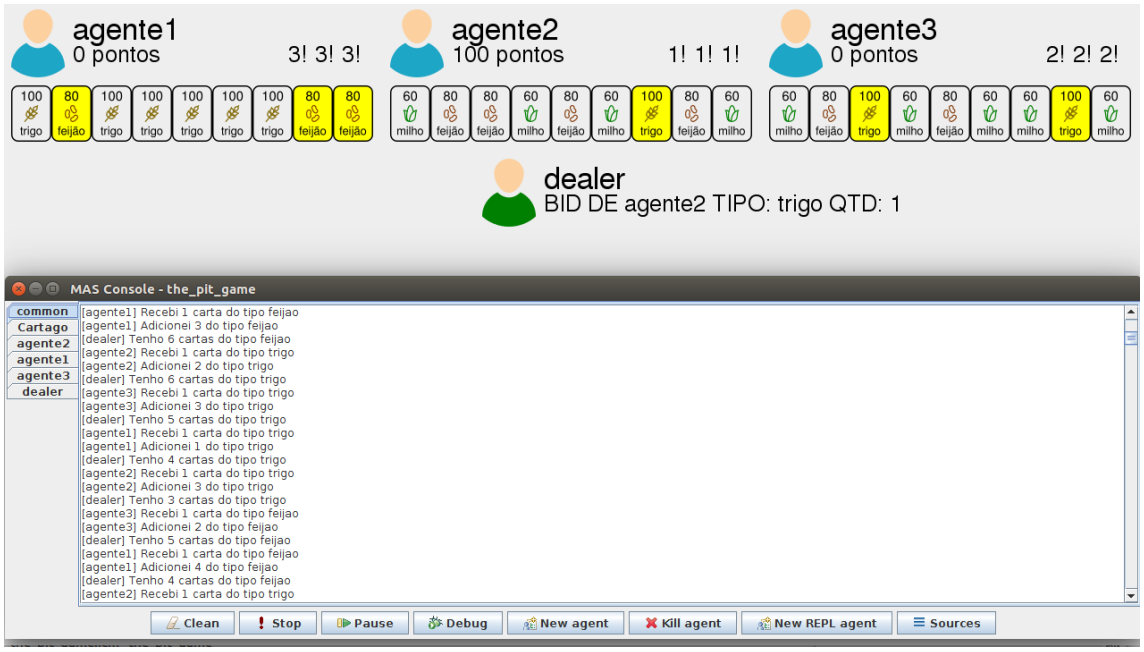

Figura 2. Interface Gráfica no Início da Primeira Rodada

simples como a distribuição das cartas em que apenas o *dealer* era o agente ativo não se verificou a presença de erros de comunicação significativos. Entretanto, em ações como trocar e receber cartas observou-se questões de concorrência onde um *player* acabava aceitando ofertas de dois outros *players*, essa questão fazia com que a base de crenças do *player* que aceitava duas ofertas registrasse uma quantidade inconsistente de cartas e isso fazia com que o jogo nunca fosse concluído, pois os *players* não conseguiriam chegar a situação de vitória ou *corner*.

Para contornar os problemas de concorrência, logo no início do corpo de execução de alguns planos adicionava-se à base de crenças do agente um crença que identificava que esse agente estaria realizando uma tarefa que não poderia ser interrompida, uma técnica que se assemelha com o uso de variáveis de trava ou semáforos para limitar o acesso à uma região crítica de memória em problemas clássicos de programação concorrente. Com isso, verificou-se que a maioria das interações ocorriam conforme o esperado, ou seja, um *player* anunciava *corner* e se iniciavam novas rodadas até que um dos *players* vencia o jogo ao alcançar um total de 500 ou mais pontos. Apesar dos avanços com o uso dessas crenças, que travavam o agente em uma determinada ação, ainda assim surgiam inconsistências na base de crenças dos agentes quando se aumentava a velocidade com que as interações eram realizadas.

5. Considerações Finais

O presente trabalho tinha como objetivo realizar uma simulação em um sistema multi-agente fechado implementado no framework JaCaMo. Foi utilizado para a simulação de interações entre os agentes a dinâmica de um jogo de negociação de cartas chamado *The Pit Game*.

Todos os jogadores possuíam crenças inciais para a representação do ambiente, cada agente deveria conhecer também a mesma ontologia que representava esse ambiente

para que as mensagens enviadas e recebidas pudessem fazer sentido dentro do jogo. A linguagem de programação Jason, uma das tecnologias que compõe o JaCaMo, permitia a utilização de ações internas para o envio de mensagens entre os agentes. Um artefato CArtAgO, outra tecnologia embutida no JaCaMo, foi criado para a integração entre agentes programados em Jason e um ambiente gráfico programado em Java.

Contudo, durante o desenvolvimento do sistema verificou-se problemas de concorrência na comunicação entre os agentes fazendo com que suas bases de crenças ficassem inconsistentes e o jogo nunca chegasse ao fim. Dado as experiências relatadas, evidencia-se que simulações em sistemas multi-agentes podem trazer desafios para a programação em linguagens orientadas a agentes como o Jason, pois, questões de sincronismo levam a problemas inesperados conforme prevê [Wooldridge 2009] no capítulo de comunicação de agentes. Considera-se então o trabalho desenvolvido como um passo inicial em futuras pesquisas envolvendo as técnicas e tecnologias de sistemas multi-agentes e as características de mercados financeiros.

Referências

Bloembergen, D., Hennes, D., Parsons, S., and Tuyls, K. (2015). Survival of the chartist: An evolutionary agent-based analysis of stock market trading. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 1699–1700. International Foundation for Autonomous Agents and Multiagent Systems.

Boissier, O., Bordini, R. H., Hübner, J. F., Ricci, A., and Santi, A. (2013). Multi-agent oriented programming with jacamo. *Science of Computer Programming*, 78(6):747–761.

Bordini, R. H., Hübner, J. F., and Wooldridge, M. (2007). *Programming multi-agent systems in AgentSpeak using Jason*, volume 8. John Wiley & Sons.

Cuni, G., Esteva, M., Garcia, P., Puertas, E., Sierra, C., and Solchaga, T. (2004). Masfit: Multi-agent system for fish trading. In *Proceedings of the 16th european conference on artificial intelligence*, pages 710–714. Citeseer.

Frank, A. U., Bittner, S., and Raubal, M. (2001). Spatial and cognitive simulation with multi-agent systems. In *International Conference on Spatial Information Theory*, pages 124–139. Springer.

Giulioni, G. (2019). An agent-based modeling and simulation approach to commodity markets. *Social Science Computer Review*, 37(3):355–370.

Hoffmann, A. O. I., Jager, W., and Von Eije, J. H. (2007). Social simulation of stock markets: Taking it to the next level. *Journal of Artificial Societies and Social Simulation*, 10(2):7.

Liu, T.-L. and Huang, H.-J. (2007). Multi-agent simulation on day-to-day route choice behavior. In *Third International Conference on Natural Computation (ICNC 2007)*, volume 5, pages 492–498. IEEE.

Muscalagiu, I., Popa, H. E., Panoiu, M., and Negru, V. (2013). Multi-agent systems applied in the modelling and simulation of the protein folding problem using distri-

buted constraints. In *German Conference on Multiagent System Technologies*, pages 346–360. Springer.

Norling, E., Sonenberg, L., and Rönnquist, R. (2000). Enhancing multi-agent based simulation with human-like decision making strategies. In *International Workshop on Multi-Agent Systems and Agent-Based Simulation*, pages 214–228. Springer.

Purvis, M., Nowostawski, M., Oliveira, M., and Cranefield, S. (2003). Multi-agent interaction protocols of e-business. In *IEEE/WIC International Conference on Intelligent Agent Technology, 2003. IAT 2003.*, pages 318–324. IEEE.

Vytelingum, P., Ramchurn, S. D., Voice, T. D., Rogers, A., and Jennings, N. R. (2010). Trading agents for the smart electricity grid. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*, pages 897–904. International Foundation for Autonomous Agents and Multiagent Systems.

Wooldridge, M. (2009). *An introduction to multiagent systems*. John Wiley & Sons.