GNU Mages – Um Ambiente para Simulação de Múltiplos Agentes Autônomos, Cooperativos e Competitivos

João Ricardo Bittencourt[1], Fernando Osório[1]
[1]Mestrado em Computação Aplicada – Centro de Ciências Exatas e Tecnológicas
Universidade do Vale do Rio dos Sinos (UNISINOS)
Caixa Postal 275 – 93.022-000 – São Leopoldo – RS – Brasil
jrbitt@netu.unisinos.br, osorio@exatas.unisinos.br

Abstract. *We describe the implementation of a cooperative and competitive autonomous multi agents simulation environment. The main goal of this application is to provide statistical comparisons between different Artificial Intelligence algorithms applied in games. GNU Mages is free software and we hope to help all community with creation of this new open tool very useful to multi agents systems field.*

Resumo. *Este trabalho descreve a implementação de um ambiente para simulação de múltiplos agentes autônomos, cooperativos e competitivos, destinado a efetuar comparações estatísticas entre diferentes algoritmos de Inteligência Artificial em aplicações do tipo jogos. Destacamos que este sistema é de livre distribuição para comunidade e esperamos contribuir com a criação de uma nova ferramenta aberta de grande utilidade na pesquisa de sistemas multi- agentes.*

1. Introdução

Atualmente existem inúmeras definições de agentes, com várias diferenças e particularidades entre elas. Franklin & Graesser [1] analisaram inúmeros conceitos e chegaram a uma definição mais abrangente: "agentes autônomos estão situados em um ambiente, no qual eles sentem e agem, sobre o tempo, mantendo sua própria agenda e sentem os efeitos de suas ações no futuro". Com o crescimento desta área as pesquisas na área de Inteligência Artificial relacionadas com o desenvolvimento de sistemas multi-agentes também são impulsionadas. Surge assim a necessidade de desenvolver novas ferramentas e sistemas de controle destes agentes.

As vantagens dos agentes autônomos são inúmeras, mas podemos destacar a sua capacidade de adaptação ao ambiente e de agir sem a interferência humana. Desta forma temos softwares inteligentes capazes de executar tarefas complexas que usualmente eram executadas exclusivamente pelo homem. Existem inúmeros exemplos de aplicações, tais como a robótica autônoma, aplicações Web, comércio eletrônico, na indústria manufatora em geral, em aplicações groupware, educação à distância, agentes virtuais, entretenimento, agentes coletores de lixo e detectores de explosivos.

Sabe-se que independente da área de aplicação de um sistema multi-agentes existe a necessidade de projetar simuladores de agentes autônomos, com a finalidade de testar diferentes arquiteturas e compará-las através de experimentos. No caso da simulação de multi-agentes autônomos a construção de um ambiente de simulação é uma opção interessante pelo fato da possibilidade de desenvolver ferramentas de análise estatística que permitam a comparação entre diferentes abordagens de agentes e de técnicas de Inteligência Artificial. Também é possível adotar uma padronização dos dados para que possam ser gerados conjuntos de exemplos para o treinamento dos agentes e fazer a geração de relatórios estatísticos de seu desempenho. Além disso, é interessante que se possua também um visualizador gráfico que permita acompanhar a execução do treinamento dos agentes e a visualização das interações entre os agentes e o ambiente.

O objetivo deste trabalho é apresentar o GNU Mages (GNU Multi AGents Environment Simulator), um simulador que permite estudar técnicas de Inteligência Artificial aplicadas aos sistemas de multi-agentes colaborativos e competitivos permitindo observar o comportamento e a evolução destes agentes dentro de um ambiente dinâmico.

2. Jogos como domínio

É importante destacar nesta seção que a concepção de jogos é bastante adequada ao escopo dos ambientes de simulação para multi-agentes. Os jogos possuem uma meta bem definida, que seria a tarefa a ser executada, a característica da competitividade, e regras bem definidas que permitem caracterizar uma forma de avaliar a performance dos participantes (agentes).

Os jogos ainda possuem uma grande relevância mercadológica e científica. Segundo os dados da Interactive Digital Software Association (IDSA), os jogos eletrônicos nos Estados Unidos geraram U$ 6,02 bilhões em vendas durante o ano de 2000. Quanto ao aspecto científico, vários autores tais como Battaiola [3], Funge [5], Russel e Norvig [4] destacam os jogos eletrônicos como uma forma de desenvolver e testar questões técnico-científicas em áreas, tais como redes de computadores, computação gráfica e inteligência artificial. Segundo Russel & Norvig, os jogos sempre foram bons alvos para as pesquisas de Inteligência Artificial. Laird e Van Lent [6], caracterizam os jogos para computador como ambientes ricos para pesquisa de Inteligência Artificial ao nível da inteligência humana.

Paralelamente ao desenvolvimento do GNU Mages, o jogo Capture The Flag [14] está sendo implementado como um domínio de aplicação que será usado para a validação do simulador.

3. Simuladores Existentes

Existe muita produção científica envolvendo agentes. No site do produto AgentBuilder [7] da empresa Reticular Systems, Inc. encontramos uma listagem com cinqüenta e três ferramentas para construção de agentes. Evidentemente que existem outras ferramentas sendo desenvolvidas na comunidade científica e pela industria internacional, mas a análise desta modesta amostra dentro de uma população desconhecida serve como um referencial no desenvolvimento de um novo ambiente de simulação, pois é de extrema importância conhecer as tendências, vantagens e desvantagens das ferramentas existentes.

Através desta amostragem foi possível constatar que a maioria dos sistemas multi-agentes são softwares proprietários e a maioria das aplicações não distribuem livremente o código-fonte, como é o caso do Retsina [15] e do Hierarquical Agent Control (HAC) [8]. A linguagem de programação mais utilizada no desenvolvimento destes sistemas é linguagem Java e as aplicações de processamento distribuído e paradigma de programação orientado a agentes são os domínios com mais aplicações desenvolvidas, como é o caso do MadKit [9]. Tutores virtuais, simuladores sociais e agentes de software são domínios que possuem menos ferramental. Deve-se destacar que o Swarm [10] é a opção mais reconhecida na comunidade, principalmente pelo fato da livre distribuição, mas em contrapartida não possui recursos voltados para o aprendizado de máquinas.

A partir deste cenário optou-se por criar um ambiente de simulação de multi-agentes, voltado às aplicações do tipo jogos, livremente distribuído sob a licença GNU GPL. Sendo que a linguagem escolhida para implementação foi a linguagem Java em função desta ser multiplataforma.

4. Ambiente de Simulação GNU Mages

O GNU Mages fundamenta-se em dez requisitos: (a) permitir flexibilidade na utilização de sensores. Deve-se oferecer opções sensoriais variadas para serem utilizadas nas controladoras inteligentes; (b) usar de uma arquitetura hierarquizada para os sensores e os módulos de controle; (c) ter flexibilidade para criar novas controladoras inteligentes; (d) personalizar a interface gráfica, ou seja, a possibilidade de usar interfaces bidimensionais, tridimensionais ou não utilizar interface, usando somente dados textuais e a linha de comandos neste caso; (e) padronizar as informações para facilitar o intercâmbio entre os usuários; (f) utilizar uma arquitetura cliente-servidor; (g) permitir a configuração de parâmetros e intercâmbio de agentes, ambientes e outros objetos de simulação entre a comunidade de usuários; (h) permitir efetuar testes estatísticos que auxiliarão as pesquisas de Inteligência Artificial; (i) ser portável; (j) ser livremente distribuído.

O GNU Mages divide-se em duas aplicações: **GNU Mages Server** e **GNU Mages Client**.

O GNU Mages Server é responsável por executar inúmeros processos, denominados *daemons*. Um daemon é responsável por "escutar" uma determinada porta de comunicação, receber os pedidos de conexões dos usuários e efetuar o gerenciamento das sessões. O processamento lógico do serviço, ou seja, o domínio do problema propriamente dito é gerenciado pelas sessões. Desta forma o servidor é o responsável pelo controle central do andamento da simulação e do estado atual do sistema.

O GNU Mages Client oferece uma interface gráfica que permite ao usuário iniciar uma nova simulação ou agregar-se a uma simulação que esteja no estado de espera. A aplicação poderá ser executada em modo interativo ou em modo batch e os agentes podem ser controlados por operadores humanos ou por controladoras inteligentes. Além disso, permite visualizar o conjunto sensorial, as informações estáticas e dinâmicas de cada agente e salvar um arquivo contendo um log das ações que será utilizado na geração de bases de exemplos para o treinamento dos agentes.

Na verdade a aplicação cliente terá uma visão parcial da simulação e dependerá das informações que lhe são repassadas pelo servidor, podendo criar ou não uma representação interna e local da situação atual da simulação. Este conhecimento do cliente representa a sua crença em relação ao estado atual da simulação, entretanto a descrição completa do estado real da simulação só é conhecida pelo servidor.

Para a implementação desta arquitetura cliente-servidor foi utilizado o **Clisp** (CLient Server Pattern) um pacote, também de livre distribuição, desenvolvido durante este projeto, tendo como base o artigo de Morrison [11]. Seu objetivo é oferecer uma estrutura genérica para aplicações cliente-servidor através de mensagens serializáveis. Este pacote possibilita a criação de protocolos de comunicação, projetar processos daemons, gerenciar sessões e projetar aplicações clientes.

4.1. Características do GNU Mages

4.1.1. Editor de Mapas 2D

Em problemas que envolvem multi-agentes, principalmente quando os domínios são jogos surge a necessidade da criação de mapas que representam o ambiente de interação. Cada mapa pode ser formado por uma ou múltiplas camadas, sendo que estas são formadas por objetos de simulação organizados em uma matriz bidimensional. Entende-se como objeto de simulação qualquer ente que pode ser posicionado em uma camada, por exemplo, bandeiras, paredes, árvores, armas, itens e diferentes tipos de terrenos. Estes objetos de simulação podem possuir diversos parâmetros e inúmeras representações gráficas (ícones).

O editor de mapas permite manipular os ícones graficamente (rotação) e os parâmetros dos objetos de simulação também podem ser editados. Os mapas são formatados no padrão XML, permitindo facilmente o intercâmbio de informações.

O XML (eXtensible Markup Language) é um padrão da W3 Consortium que consiste de uma linguagem para descrever documentos semelhante ao HTML, onde o usuário pode construir seus próprios elementos para descrever seus objetos. Na implementação do GNU Mages foi utilizado Xerces-Java [12] do Apache XML Project que permite criar e efetuar um parser de um arquivo XML usando um DTD.

Além do mapa, outros objetos, tais como descritores de domínio e os dados das simulações foram padronizados usando o XML.

4.1.2 Arquitetura Genérica para Modelagem de Agentes

A arquitetura desenvolvida neste trabalho baseia-se no pacote Clay do TeamBots [13] e na estrutura hierárquica do Hierarquical Agent Control (HAC). Essencialmente constitui-se de nodos distribuídos em forma hierárquica sendo que os nodos superiores consultam os nodos inferiores em busca de informações, inclusive as sensórias. O nodo que efetuou a solicitações para os filhos executa algum processamento e entrega a informação para o nodo-pai. Esta estrutura de nodos facilita extremamente a criação de arquiteturas de agentes, pois permite a utilização de diferentes

tipos de sensores, a combinação destes, processamentos e a utilização de uma estrutura comunicacional entre agentes.

Para o desenvolvimento destas controladoras é necessário utilizar inúmeros algoritmos de Inteligência Artificial, por exemplo, implementações de Redes Neurais Artificiais (RNA). Para isto o GNU Mages utiliza o ANNeF [2], um framework de livre distribuição que implementa alguns algoritmos de RNAs.

5. Considerações Finais

Conclui-se o presente trabalho destacando a importância do desenvolvimento de uma aplicação de livre distribuição dentro de um cenário com poucas ferramentas desta natureza e com intensa pesquisa no campo de sistemas multi-agentes. Atualmente o GNU Mages encontra-se na fase final de implementação e informações atualizadas sobre o sistema podem ser obtidas em http://www.inf.unisinos.br/~jrbitt/mages.

Pelo fato da livre distribuição, utiliza conceitos de multiplataforma e da padronização dos dados utilizando o XML, desta forma facilitando o intercâmbio de informações e contribuindo para o crescimento dos sistemas multi-agentes e o aperfeiçoamento de técnicas de Aprendizado de Máquinas para diferentes domínios de problemas.

Referências Bibliográficas

[1] FRANKLIN, S. GRAESSER, A. Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agent. In: **Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages**. Springer-Verlag, 1997, 21-35 p.

[2] BITTENCOURT, J. R. OSÓRIO, F. ANNeF – Artificial Neural Networks Framework: Uma Solução Software Livre para o Desenvolvimento, Ensino e Pesquisa de Aplicações de Inteligência Artificial Multiplataforma. In: **Anais do II Workshop sobre Software Livre**. Porto Alegre: SBC, p. 13-16, 2001.

[3] BATTAIOLA, A. L. Jogos por Computador – Histórico, Relevância Tecnológica e Mercadológica, Tendências e Técnicas de Implementação. In: **Anais da XIX Jornada de Atualização em Informática – JAI**. Julho/2000, volume 2, pág. 83 – 122.

[4] RUSSEL, S. NORVIG, P. **Artificial Intelligence: A Modern Approach**. New Jersey: Prentice-Hall, 1995, 932 p.

[5] FUNGE, John David. **AI for Games and Animation: a cognitive modeling approach**. Massachutsetts: A.K Peters. 1999, 212p.

[6] LAIRD, J.E.; VAN LENT, M. Human-level AI's Killer Application: Interactive Computer Games. 2000. In: **Proceedings of the Seventeenth National Conference on Artificial Intelligence AAAI 2000** (Inveted Talk). Austin: AAAI Press, 2000.

[7] **Agent Construction Tools.** http://www.agentbuilder.com/AgentTools/index.html Visitado em 28/11/2001.

[8] ATKIN, M. S. KING, G. W. WESTBROOK, D. L. et al. Hierarchical Agent Control: A Framework For Defining Agent Behavior. In: **AGENTS' 2001: Proceedings of the Fifth International Conference on Autonomous Agents**. ACM Press,New York,2001.425-432

[9] **MadKit.** http://www.madkit.org Visitado em 29/11/2001

[10] **Swarm.** http://www.swarm.org Visitado em 29/11/2001

[11] MORRISON, Michael. Java Network Game Programming. **Gamasutra**, ago. 1997. http://www.gamasutra.com/features/19970812/morrison_01.htm

[12] **Xerces Java Parser**. http://xml.apache.org/xerces-j/index.html Visitado em 29/11/2001

[13] **TeamBots**. http://www.teambots.com. Visitado em 28/11/2001

[14] ATKIN, M.S. ; WESTBROOK, D.L. COHEN, P.R. Capture The Flag: Military Simulation Meets Computer Games. In: **Working Notes of the AAAI Symposium on AI and Computer Games**, 1-6 p, 1999.

[15] SYCARA, K. ; PAOLUCCI, M. ; VAN VELSEN, M. et al. **The RETSINA MAS Infrastructure**. Submitted to JAAMS, 2001.