Descrição e Geração de Ambientes para Simulações com Sistemas Multiagente*

Fabio Y. Okuyama[1], **Rafael H. Bordini**[1,2]

[1]Programa de Pós-Graduação em Computação
Universidade Federal do Rio Grande do Sul
Caixa Postal 15064 – 90501-970 Porto Alegre, RS

[2]Department of Computer Science, University of Durham
Durham DH1 3LE, U.K.

okuyama@inf.ufrgs.br, R.Bordini@durham.ac.uk

Abstract. *This dissertation is situated in the area of Multi-Agent Systems and approaches, in particular, the problem of environment modelling, an important aspect in the creation of simulations based on societies of cognitive agents, but one that is seldom discussed in the multi-agent systems literature. The main contribution of this work is the conception of a language, called ELMS, that is suitable for the definition of multi-agent environments, and the implementation of a prototype interpreter for that language. The result of such interpretation is a process which simulates the environment, described at high level, in a way that is appropriate for the interaction with the cognitive agents that will share the environment.*

Resumo. *Esta dissertação situa-se na área de Sistemas Multiagente e aborda particularmente o problema da modelagem de ambientes, um aspecto importante na criação de simulações baseadas em sociedades de agentes cognitivos, no entanto pouco tratado na literatura da área. A principal contribuição deste trabalho é a concepção de uma linguagem, chamada ELMS, própria para a definição de ambientes multiagente, e a implementação de um protótipo de interpretador para esta linguagem. O resultado da interpretação é um processo que simula o ambiente, descrito em alto nível, de maneira adequada para a interação com os agentes cognitivos que irão compartilhar o ambiente.*

1. Introdução

A Inteligência Artificial Distribuída (IAD), segundo [10], é o estudo, construção e aplicação de sistemas multiagente, que são sistemas nos quais vários agentes inteligentes interagem para realizar um conjunto de objetivos ou tarefas. De acordo com [5], a IAD se divide basicamente em Sistemas Multiagente (SMA) e Resolução Distribuída de Problemas (RDP).

Em [2], os autores afirmam que o enfoque principal dos SMA é prover mecanismos para criação de sistemas computacionais a partir de entidades de software autônomas, denominadas agentes, que interagem através de um ambiente compartilhado por todos os

*Trabalho realizado com auxílio da CAPES.

agentes de uma sociedade e sobre o qual atuam, alterando seu estado. Como os agentes possuem um conjunto específico e limitado de capacidades, freqüentemente os agentes precisam interagir para atingir seus objetivos. Assim, é possível para os projetistas de sistemas computacionais a criação de sistemas computacionais complexos e dinâmicos de forma naturalmente distribuída e *bottom-up*.

A especificação de um SMA pode ser feita através da modelagem do ambiente, agentes, mecanismos para interação e organizações (ou grupos) de agentes envolvidas. Um aspecto pouco explorado é a modelagem do ambiente, que é parte importante principalmente para simulações sociais onde é necessário um ambiente virtual onde os agentes irão interagir. De forma mais geral, quando o SMA é um sistema computacional completo (não situado em um ambiente real, como uma linha de produção industrial), a modelagem do ambiente compartilhado entre os agentes é parte essencial do projeto e desenvolvimento de sistemas multi-agentes.

Esta dissertação visa propor uma linguagem de alto nível na qual o ambiente a ser compartilhado pelos agentes de um SMA possa ser descrito [7]. Esta linguagem deve abranger diversos aspectos envolvidos na modelagem de ambientes para SMA, tais como as propriedades dos objetos que são alteradas pelas ações dos agentes, as propriedades que são perceptíveis a cada agente, que ações são permitidas a cada agente e sob que condições estas podem ser executadas. Durante o desenvolvimento da linguagem foi realizada uma simulação de crescimento urbano, cujos resultados podem ser vistos em [3, 6].

Esta linguagem foi desenvolvida no contexto do projeto **M**ulti-**A**gent **S**imulations for the **SOC**ial Sciences[1], que tem como objetivo a criação de simulações sociais baseadas em agentes cognitivos. A abordagem deste projeto dá ênfase ao uso da arquitetura BDI para agentes cognitivos, à comunicação inter-agente de alto nível (ou seja, baseada em atos de fala) e à modelagem de ambientes com a linguagem ELMS (**E**nvironment Description **L**anguage for **M**ulti-Agent **S**imulations), que foi proposta nesta dissertação [7] e também mencionada em [1] e apresentada mais detalhadamente em [8].

2. Modelagem de Ambientes

Agentes são sistemas computacionais situados em um ambiente, e que são capazes de ações autônomas neste ambiente para realizar seus objetivos [11]. Considerando um agente situado em um ambiente, quaisquer outros agentes serão, do ponto de vista deste, parte importante do ambiente. Desta forma a modelagem dos ambientes é um aspecto importante nas simulações multiagente. Mesmo quando não simulam um mundo real, a construção de "mundos sintéticos" tem grande importância na pesquisa de sistemas multiagente pois torna possível analisar certos mecanismos de interação de maneira mais sistemática do que em uma aplicação real [4].

Nos SMA reativos, o ambiente tem papel importante pois como agentes reativos não possuem memória, é somente através da percepção do ambiente que estes decidem como agir. Já os agentes cognitivos, que na maioria das vezes possuem uma representação interna do ambiente, podem tomar decisões baseadas em *mudanças* percebidas no ambiente (por comparação com a memória dos estados anteriores). Ou seja, em ambas classes de sistemas multiagente a modelagem do ambiente é um aspecto fundamental.

3. A Modelagem de Ambientes Utilizando ELMS

A descrição de um ambiente utilizando ELMS pode ser feita através da especificação de propriedades tais como: os objetos (também referidos como *recursos*) presentes no ambiente; os agentes, ou seja, a representação física de um agente que é visível para outros agentes no ambiente; as ações que cada classe de agentes pode realizar no ambiente; tipos de percepções disponíveis para cada classe de agentes; e as propriedades "observáveis" do ambiente, ou seja, aquilo que o usuário deseja observar durante a simulação (ou como resultado dela).

Os recursos ou objetos que estão presentes em um ambiente podem ser modelados como um conjunto de propriedades e reações que estes podem realizar em resposta a estímulos externos ao objeto. Em outras palavras, os objetos podem reagir, apenas agentes são pró-ativos. Agentes podem ser considerados componentes do ambiente, visto que, do ponto de vista de um agente, qualquer outro agente é parte do ambiente. Assim, os "corpos" dos agentes são definidos como uma lista de propriedades (que define os aspectos perceptíveis dos agentes situados neste ambiente), uma lista de ações que estes são capazes de realizar (pró-ativamente), e uma lista de percepções a que eles têm acesso. Do ponto de vista do ambiente, as atividades de deliberação de um agente não são relevantes, pois são internas a este, e não observáveis para os outros agentes. No projeto MAS-SOC, as atitudes mentais dos agentes são definidos através da linguagem AgentSpeak(L).

Para a definição dos tipos de percepção a que cada tipo de agente tem acesso, é necessário definir quais propriedades do ambiente, agentes e objetos serão perceptíveis para os agentes que tiverem acesso aquele tipo de percepção. Além disto, condições associadas com cada propriedade perceptível também podem ser especificadas (as condições sob as quais uma propriedade é informada a um agente quando este realiza a percepção). Na definição de percepções é possível especificar atributos de células, conteúdo de células, atributos de recursos ou agentes, e variáveis de controle do ambiente. O ELMS possibilita que agentes tenham como percepção algumas variáveis de controle do ambiente, como por exemplo o passo corrente da simulação, que pode servir para dar aos agentes uma noção abstrata de "tempo".

Uma ação é definida como uma seqüência de atribuições às propriedades (do ambiente, recursos ou agentes) que esta causa, e as pré-condições que devem ser satisfeitas para que a ação possa ser realizada no ambiente. As propriedades do ambiente acessíveis aos observadores externos (os usuários) são definidas da mesma forma que aquelas perceptíveis pelos agentes. Uma grade opcional pode ser usada para representar o espaço físico do ambiente. As posições da grade podem ser acessadas por coordenadas absolutas ou relativas. Coordenadas relativas são precedidas por '$+$' ou '$-$', então $(+1, -1, +0)$, por exemplo, se refere a posição na diagonal superior direita da posição atual do agente (na mesma profundidade).

4. Expressividade da Linguagem ELMS

Os ambientes que podem ser descritos com o uso do ELMS são, segundo a classificação de ambientes de [9], não-acessíveis, não-determinísticos (do ponto de vista do agente), não-episódicos, dinâmicos e normalmente discretos. Portanto, ambientes ELMS podem ser relativamente complexos. A seguir, será brevemente explicado como cada uma destas características de ambientes podem ser modeladas em ELMS:

Inacessível: Acessível é um ambiente no qual os agentes têm acesso ao estado completo do ambiente. Nos ambientes inacessíveis os agentes têm acesso restrito ao estado do ambiente. Na modelagem com o ELMS, o agente tem acesso apenas às propriedades do ambiente que foram escolhidas pelo projetista do ambiente, na definição das percepções.

Não-determinístico: Como os ambientes ELMS podem ser inacessíveis, e há a possibilidade de vários agentes realizarem ações simultaneamente, do ponto de vista de um agente, os ambientes ELMS podem parecer não-determinísticos.

Não-episódico: como os agentes são cognitivos, as ações tomadas por um agente poderão (ainda que não necessariamente) interferir em suas ações futuras, já que é um processo interno ao agente que determina como ele reage em cada ciclo de execução.

Dinâmico: enquanto um agente delibera, ou enquanto espera que uma ação selecionada seja efetuada, as ações de outros agentes podem mudar o estado do ambiente, impedindo a execução da ação conforme esperada pelo agente, o que faz com que o ambiente pareça dinâmico.

Discreto: como o "espaço físico" do ambiente é normalmente representado através de uma grade, o ambiente é discretizado em coordenadas inteiras; porém, vale lembrar que o uso da grade é opcional.

5. Execução de Ambientes ELMS

Em uma simulação que faça uso da linguagem ELMS, o ambiente é controlado por um processo gerado pelo interpretador a partir da especificação do ambiente feita nesta linguagem. A estrutura de dados que representa o ambiente é gerada (em Java) pelo interpretador. A linguagem possui construções que permitem a geração de uma descrição não apenas do estado inicial do ambiente de uma simulação, mas também de uma descrição instantânea de uma simulação em andamento. O processo que controla a execução do ambiente pode gerar uma descrição instantânea do ambiente (na linguagem ELMS), permitindo ao usuário salvar o estado de simulação para uma execução posterior, ou fazer mudanças no ambiente durante sua execução. Isto pode ser útil também na elaboração de formas mais complexas de visualização de simulações multiagente, assim como o fato de que a linguagem ELMS é baseada em XML. A notação XML foi escolhida pela grande diversidade de ferramentas disponíveis para criação e manipulação deste formato. A especificação e alteração do ambiente é feita pelo usuário através de uma interface gráfica, evitando que o usuário tenha que usar diretamente uma notações XML, já que estas são, em geral, bastante ilegíveis.

É possível selecionar a execução do ambiente no modo síncrono e assíncrono. No modo de execução síncrono, o processo que controla o ambiente envia, para todos os agentes da simulação, a cada ciclo de execução, os dados das percepções às quais estes têm acesso (conforme definido em ELMS). Neste modo de execução o ambiente irá aguardar (sendo que um tempo limite de espera pode ser especificado) que todos os agentes escolham uma ação para execução em um dado passo da simulação, e somente então as ações são executadas, em uma ordem aleatória. Em contrapartida, no modo de execução assíncrono, tem prioridade o agente que responde mais rápido às percepções, seja por possuir melhores recursos computacionais ou pela eficiência de seu código. A ordem das ações é importante, visto que uma ação pode bloquear a execução das seguintes.

Enquanto no modo de execução síncrono a duração da simulação é definida em

"passos", no modo assíncrono é definido um tempo limite para toda a simulação. Tanto o tempo limite da simulação assíncrona como o número de passos da simulação síncrona são definidos pelo usuário através da interface gráfica. Esta interface permite não só a gerência da execução de simulação, mas a definição de simulações também. É este "gerente" de simulações que recebe o resultado das propriedades definidas como observáveis.

6. Conclusão

O projeto MAS-SOC que tem como objetivo possibilitar que pessoas sem conhecimentos aprofundados em programação, tal como cientistas sociais, possam criar simulações sociais com agentes cognitivos. Para tal, era necessário um meio que facilitasse a criação dos ambientes em que os agentes estão situados durante tais simulações. Esta dissertação contribui para o projeto MAS-SOC neste aspecto importante, propondo uma linguagem de alto nível para a definição de ambientes multiagente e apresentando o protótipo de um interpretador para tal linguagem.

Assim, acreditamos que esta dissertação, juntamente com o projeto MAS-SOC, apresenta uma contribuição para a área de SMA, permitindo a implementação de simulações baseadas em SMA de forma simples. Além disto, as simulações geradas com este enfoque podem ser bem mais sofisticadas do que as normalmente geradas com outras plataformas disponíveis, já que permite o uso de agentes BDI, a principal arquitetura para agentes cognitivos na área de sistemas multiagente.

Referências

[1] Bordini, R., Okuyama, F. Y., de Oliveira, D., Drehmer, G., and Krafta, R. C. (2004). The MAS-SOC approach to multi-agent based simulation. In Lindemann, Moldt, and Paolucci, editors, *1st Int. Workshop on Regulated Agent-Based Social Systems (RASTA 2002, held with AAMAS02), Revised Selected and Invited Papers*, LNAI 2934, Berlin. Springer-Verlag.

[2] Bordini, R. H., Vieira, R., and Moreira, Á. F. (2001). Fundamentos de sistemas multiagentes. In Ferreira, C. E., editor, *Anais do XXI Congresso da Sociedade Brasileira de Computação (SBC2001), Volume 2, Fortaleza-CE, Brasil*, chapter 1, pages 3–41. SBC.

[3] de Oliveira, D. (2002). Simulação de aspectos sociais do crescimento urbano com sistemas multiagentes cognitivos. Projeto de Diplomação. II-UFRGS, Porto Alegre.

[4] Ferber, J. (1999). *Multi-Agent Systems - An Introduction to Distributed Artificial Intelligence*. Addison-Wesley, London.

[5] Gasser, L. (1987). Distribution and coordination of tasks among intelligent agents. In *Proc. of the 1st Scandinavian Conference on Artificial Intelligence*, Tromsø, Norway.

[6] Krafta, R., Oliveira, D. d., and Bordini, R. H. (2002). The city as object of human agency. In *4th International Space Syntax Symposium*, London.

[7] Okuyama, F. Y. (2003). Descrição e geração de ambientes para simulações com sistemas multiagente. Dissertação de mestrado, PPGC/UFRGS, Porto Alegre, RS.

[8] Okuyama, F. Y., Bordini, R. H., and Rocha Costa, A. C. (2004). ELMS: an enviroment description language for multi-agent simulations. *1st Int. Workshop on Environments for Multi-Agent Systems (E4MAS)* to be held with AAMAS'2004.

[9] Russel, S. and Norvig, P. (1995). *Artificial Intelligence - A Modern Approach*. Prentice-Hall, Englewood Cliffs.

[10] Weiss, G., editor (1999). *Multiagent Systems—A Modern Approach to Distributed Artificial Intelligence*. MIT Press, Cambridge, MA.

[11] Wooldridge, M. (2002). *An Introduction to Multiagent Systems*. John Wiley and Sons, England.