TDP MECATEAM 2009

Cleber Pinelli Teixeira*, Orivaldo Vieira Santana Júnior†, Augusto Loureiro da Costa‡

*Av. Adhemar de Barros s/n, Ondina - 40170-110
Universidade Federal da Bahia (UFBA)
Departamento de Ciência da Computação - Instituto de Matemática
Salvador, Bahia, Brasil

†Av. Professor Luís Freire s/n, Cidade Universitária - 50740-540
Universidade Federal de Pernambuco (UFPE)
Centro de Informática (CIn)
Recife, Pernambuco, Brasil

‡Rua Prof. Aristides Novis, Federação - 40210-630
Universidade Federal da Bahia (UFBA)
Departamento de Engenharia Elétrica - Politécnica
Salvador, Bahia, Brasil

Emails: pinelli@dcc.ufba.br, orivajr@dcc.ufba.br, augusto.loureiro@ufba.br

Abstract— The MecaTeam is a robot soccer simulation team that has been developing gradually. In 2007 a proposal of change in the team's architecture was peformed using multithread programming, fragmenting the architecture in three layers. The present Work's Plan proposed a continuation of the project that peform the knowledge bases of agents who controls the robots of MecaTeam 2008 to autonomous realization of plans. These plans contains the knowledge necessary to identify the current state of match, select the robot's goal, and with bases on current state, goal and the abilities of the robots choosing the behavior tha will be used.

Keywords— Intelligent Multiagent systems, Planning, reasoning and smart making decision.

Resumo— O MecaTeam é um time de futebol de robôs simulado que vem evoluindo gradativamente. Em 2007 a proposta de uma mudança na arquitetura do time, foi realizada, utilizando programação multi-thread, dividindo sua arquitetura em três camadas. O presente Plano de Trabalho propôs uma continuidade ao projeto que consiste na capacitação das bases de conhecimento dos agente que controlam os robôs do MecaTeam 2008 a realização autônoma de planos. Tais planos contém o conhecimento necessário para identificar o estado corrente do jogo, selecionar o objetivo do robô, e com base no estado corrente, no objetivo e nas habilidades dos robôs, escolher o comportamento que será utilizado.

Keywords— Sistemas multi-agentes inteligentes, Planejamento, raciocínio e tomada de decisão inteligente.

1 Introdução

O MecaTeam apresenta uma evolução na arquitetura do agente UFSC-Team-98 (Costa and Bittencourt, 1999b) na RoboCup'98, chamado Agente Autônomo Concorrente (Costa and Bittencourt, 1999a). Este baseia-se em um modelo híbrido para agente cognitivo (Bittencourt and Costa, 2001) e possui uma arquitetura de três camadas, cada uma delas representa um nível decisório distinto que complementa os demais para a construção de uma agente cognitivo. A arquitetura atual do MecaTeam é implementada re-utilizando o código do UvA Trilearn 2003 e a Expert-Coop++ (Costa et al., 2003).

O UvA Trilearn 2003 é um time de futebol de robôs simulado que disponibilizou seu código fonte na Internet. Este time possui uma boa codificação e documentação, por isso ele foi escolhido para fazer parte das camadas Reativa e Instintiva do Agente MecaTeam.

A Expert-Coop++ é uma biblioteca orientada a objetos destinada a auxiliar o desenvolvimento de sistemas multiagentes sob restrição de tempo real do tipo melhor esforço(Costa et al., 2003), implementada na linguagem de programação C++.

O MecaTeam 2009 utiliza Sistemas Multiagentes para implementação do controle distribuído de um sistema multi-robôs, onde, cada robô é controlado por um agente cognitivo. O raciocínio automático do agente é implementado num Sistema Baseado em Regras de Produção, que é composto por um analisador léxico e sintático de regras para o Agente MecaTeam. Este analisador é responsável por ler o arquivo de regras em modo texto, extrair estas regras e colocá-las em estruturas de dados, que são manipuladas pelo Sistema Baseado em Conhecimento.

O MecaTeam 2009 é apresentado neste artigo, de acordo com a seguinte disposição: a seção 2 apresenta a arquitetura do UvA Trilearn usado como base para implementar o agente do MecaTeam; A Expert-Coop++ é mostrada na seção 3, junto com seu Sistema Baseado em Conhecimento; A configuração atual do MecaTeam é mostrada na seção 4; Os problemas relacionados ao nível Cognitivo da arquitetura do MecaTeam é apresentado na seção 5; Na seção 6 está presente a conclusão e os tra-

balhos futuros.

2 Arquitetura do UvA Trilearn

A arquitetura do UvA Trilearn utiliza a clássica abordagem hierárquica e a abordagem comportamental. A característica comum em arquiteturas hierárquicas é que o sistema é dividido em níveis progressivos de abstração, os quais são representados por diferentes camadas arquiteturais. Nesta abordagem, o fluxo de informação é usado como a principal diretriz para a decomposição do sistema.

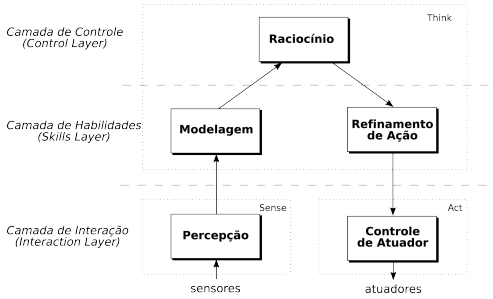

Figura 1: Arquitetura do agente UvA Trilearn 2001

A arquitetura mostrada na figura 1 é hierárquica, pois contém três camadas em diferentes níveis de abstração. A camada inferior é a *Camada de Interação*, a qual cuida da interação com o *Soccer Server* ambiente de simulação. A camada do meio é a *Camada de Habilidades*, que usa as funcionalidades oferecidas pela Camada de Interação para construir um abstrato modelo de mundo e implementar vários comportamentos do agente. A mais alta camada na arquitetura é a *Camada de Controle*, que, por sua vez, contém o componente de raciocínio do sistema. Nesta camada, a melhor ação possível é selecionada da Camada de Habilidades dependendo do atual estado do mundo e da estratégia atual do time. Assim, o agente UvA Trilearn é capaz de *perceber, raciocinar* e *agir* (Kok et al., 2003).

3 A Biblioteca Expert-Coop++

A **Expert-Coop++** (Costa et al., 2003) é uma biblioteca orientada a objetos destinada a auxiliar o desenvolvimento de sistemas multiagentes sob restrição de tempo real do tipo melhor esforço(Costa et al., 2003), implementada na linguagem de programação *C++*. Esta biblioteca oferece suporte a Sistemas Baseados em Conhecimento (SBC).

3.1 O Sistema Baseado em Conhecimento

O Sistema Baseado em Conhecimento pode ser classificado como um Sistema Baseado em Regras de Produção (SBRP) e apresenta uma arquitetura de três módulos: uma *base de regras*, uma *base de fatos ou memória de trabalho* e um *motor de inferência* (de Santana Jr et al., 2006) como ilustra a figura 2.

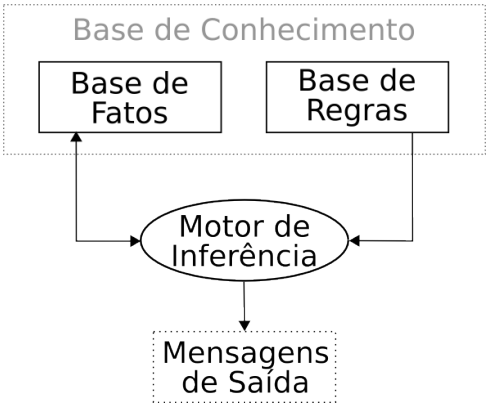

Figura 2: Arquitetura do Sistema Baseado em Conhecimento utilizado no MecaTeam

A base de regras e a base de fatos formam a base de conhecimento do SBRP, na qual é representado o conhecimento de um jogador de futebol. O motor de inferência é o mecanismo de controle do sistema que avalia e aplica as regras de acordo com as informações na base de fatos.

3.2 Base de Fatos

A base de fatos contém uma descrição dos estados do ambiente. Ela é construída, tendo como base árvores binárias, onde cada nó possui uma representação de conhecimento. O método de representação de conhecimento utilizado é a lógica de primeira ordem (Bittencourt, 2001), que é usada de forma implícita como é ilustrado abaixo.

```
( logic (<objeto> <atributo> <valor>))
```

A base de fatos é atualizada de duas maneiras: uma com informações do ambiente e outra com as regras. As informações mais relevantes sobre estado ambiente são fornecidas pelo modelo de mundo reutilizado do UvA Trilearn, visto na seção 2.

3.3 Base de Regras

A base de regras é responsável pela leitura do arquivo de regras, previamente definido pela extração destas regras e pelo seu armazenamento em uma lista. Este processo de extração é realizado por um analisador de regras.

A base de regras contém as reações aos possíveis estados do ambiente. A inteligência do agente é formalizada em regras de produção. Elas são criadas no intuito de prever os possíveis estados do ambiente e determinar qual o comportamento do agente em cada um destes estados.

3.4 Motor de Inferência

O motor de inferência é o principal componente do sistema. Seu funcionamento consiste na realização de inferências a partir da base de regras e da base de fatos. O modo de raciocínio usado no motor de inferência é o *forward chaining*(encadeamento progressivo) (Bittencourt, 2001), em que a parte esquerda da regra é comparada com a descrição da situação atual contida na base de fatos. Apenas as regras que satisfazem esta descrição são selecionadas. Um exemplo de regra é ilustrado na figura 3.

Ao final do processo de seleção das regras o motor de inferência dispõe de um conjunto de regras que satisfazem a situação atual do problema (denominado *conjunto de conflito*). Se este conjunto for vazio, o processo de inferência é finalizado; caso contrário, torna-se necessária a definição das regras que serão efetivamente executadas e sua ordem de execução(Bittencourt, 2001).

```
(rule_kickToGoal
   (if   (logic ( player ball_kickable true ))
         (logic ( player localization AREA_ATTACK )))
   (then (logic ( reactive_behavior active kick_to_goal ))))
```

Figura 3: Um exemplo de regra utilizada no agente MecaTeam

A formalização da gramática de descrição das regras foi feita tomando como base as regras preexistentes na versão anterior da Expert-Coop++, para manter a compatibilidade. Esta gramática descreve as estruturas da linguagem, sendo que as mais importantes são o lado direito e lado esquerdo das regras. O lado esquerdo das regras contém a representação de um estado do ambiente e é identificado através do token if, já o lado direito contém informações para modificar o estado e o then identifica este lado. Para representar o conhecimento sobre o estado da partida, é utilizado o formalismo da lógica (Bittencourt, 2001); logo a informação é formatada em objeto, atributo e valor como na figura 3.

3.5 A Análise Léxica e Sintática

No Agente MecaTeam, o analisador léxico faz a leitura do arquivo de regras, caractere a caractere, extraindo uma seqüência de símbolos léxicos, retornando o token correspondente a cada seqüência léxica identificada. Os símbolos léxicos são as palavras reservadas, identificadores e operadores. Estas palavras são *if, then, filter, logic, frame, message, to, from,* etc. Os identificadores válidos, relativos às informações do jogo são: *player, reactive_behavior, kick_to_goal, localization* etc; os operadores utilizados são os booleanos.

O método de análise baseia-se na estratégia top-down (descendente) e é do tipo recursivo preditivo, já que o token em análise determina exatamente qual produção deve ser aplicada na expansão de cada não-terminal. Isso é comprovado pois a gramática possui as seguintes características: não tem recursividade à esquerda; está fatorada à esquerda; e os não-terminais com mais de uma regra de produção possuem o primeiro terminal capaz de identificar univocamente a produção que deve ser aplicada em cada instante da análise (Price and Toscani, 2001).

4 MecaTeam 2009

Após alguns anos trabalhando-se na evolução do time, o MecaTeam hoje possui sua arquitetura de três camadas concluída, onde a camada reativa é tratada pelo Uva Trilearn, que estabelece a comunicação com o *soccer server*, a camada instintiva tratada pela Expert-Coop++ que é responsável pela escolha do comportamento, e a camada cognitiva que é responsável pelo nível estratégico do time, possibilitando a cooperação entre os agentes, que será mais detalhada na seção 5. A complexidade de comportamento do agente é incrementada a cada camada, como pode ser visto na Figura 4.

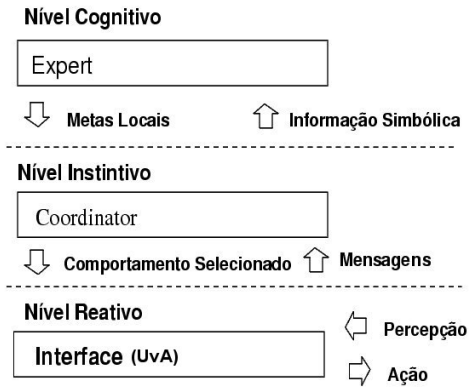

Figura 4: Arquitetura do Agente MecaTeam 2009

O presente Plano de Trabalho propôs uma continuidade ao projeto que consiste na capacitação das bases de conhecimento dos agente que controlam os robôs do MecaTeam 2009 a realização autônoma de planos. Tais planos contém o conhecimento necessário para identificar o estado corrente do jogo, selecionar o objetivo do robô, e como base no estado corrente e no objetivo e nas habilidades do robôs, escolher o comportamento que será utilizado.

5 O Nível Cognitivo

O nível superior da arquitetura do agente é denominado Cognitivo, seu papel principal é construir um modelo lógico do ambiente a partir das informações simbólicas provenientes do Instintivo, escolhendo planos e avaliando a validade do plano corrente. O nível Cognitivo não interfere diretamente sobre o nível Reativo, ele apenas determina o plano e o envia para o nível Instintivo,

que seleciona o comportamento reativo adequado (Costa, 2001).

O MecaTeam 2009 já possui o Nível cognitivo em funcionamento, embora muito simples, foi implementada utilizando *programação multithread*, e para garantir um bom funcionamento foi necessário a utilização de técnicas de sincronização (Teixeira et al., 2008), resolvendo o problema do produtor e consumidor (Tanenbaum, 2003), em que o Cognitivo pode precisar de novas informações do Instintivo que não foram enviadas ainda ou o Instintivo precisa mandar uma nova mensagem para o Cognitivo, que ainda não processou as informações anteriores.

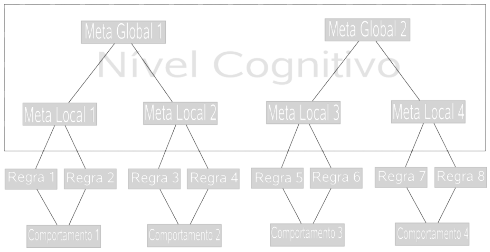

Figura 5: Arvore de decisão modelada com o cognitivo

A figura 5 ilustra o funcionamento da árvore de decisão, onde podemos observar sua estrutura hierárquica, onde são modeladas as metas globais, que são compartilhadas por mais de um agente, com o intuito de seguirem um plano estratégico e tendo um objetivo em comum, e as metas locais, que representam um estratégia individual, em que o agente executa um conjunto de tarefas a fim de satisfazer uma das tarefas da meta global, ou seja, a meta local apresenta-se como uma subtarefa da meta global.

6 Conclusões e Trabalhos Futuros

O MecaTeam tem apresentado-se como um laboratório de pesquisa em computação, pois como pode ser visto nas seções anteriores diversos problemas em distintas áreas de computações puderam ser resolvidos a fim de satisfazer a arquitetura que está estabelecida hoje.

O MecaTeam apresenta sua arquitetura concluída, e explorar o seu potencial é algo que tem sido feito. O principal objetivo atual é implementar estratégias de cooperação, onde os agentes passarão a adotar planos e deverão negociar entre si o melhor plano a ser seguido, descartando portanto o trabalho individual de cada jogador.

Referências

Bittencourt, G. (2001). *Inteligência Artificial Ferramentas e Teorias*, Editora da UFSC, ISBN 85-328-0138-2, 362 p., Florianópolis, SC, 2ª edição.

Bittencourt, G. and Costa, A. L. d. (2001). Hybrid cognitive model, *The Third International Conference on Cognitive Science ICCS'2001 : Workshop on Cognitive Angents and Agent Interaction*. Pequim, China.

Costa, A. L. d. (2001). *Conhecimento Social Dinamico: Uma Estratégia de cooperação para Sistemas Multiagentes Cognitívos*, PhD thesis, Universidade Federal de Santa Catarina / Programa de Pós-Graduação em Engenharia Elétrica, Florianópolis, Brasil.

Costa, A. L. d. and Bittencourt, G. (1999a). From a concurrent architecture to a concurrent autonomous agents architecture., *IJCAI'99, Third International Workshop in RoboCup* pp. 85–90. Springer, Lecture Notes in Artificial Inteligence.

Costa, A. L. d. and Bittencourt, G. (1999b). Ufsc-team: A cognitive multi-agent approach to the robocup'98 simulator league., *RoboCup98 Workshop - Team description* pp. 371, 377. Springer, Lecture Notes in Artificial Inteligence, vol.1694.

Costa, A. L. d., Bittencourt, G., Gonçalves, E. M. N. and Silva, L. R. (2003). Expert-coop++: Ambiente para desenvolvimento de sistemas multiagente., *IV ENIA Encontro Nacional de Inteligência Artificial* pp. 597–606. XXIII Congresso da Sociedade Brasileira de Computação.

de Santana Jr, O. V., Sousa, J. P. R. P., Linder, M. S. and da Costa, A. L. (2006). Mecateam: Um sistema multiagente para o futebol de robôs simulado baseado no agente autônomo concorrente, *Encontro de Robótica Inteligente / XXVI Congresso da Sociedade Brasileira de Computação* pp. 146–152.

Kok, J. R., Vlassis, N. and Groen, F. (2003). Team description uva trilearn 2003., *RoboCup 2003 Symposium*.

Price, A. M. A. and Toscani, S. S. (2001). *Implementação de Linguagens de Programação: Compiladores*, 1rd edn, Ed. Sagra Luzzatto.

Tanenbaum, A. S. (2003). *Sistemas Operacionais Modernos*, 2rd edn, Prentice-Hall.

Teixeira, C. P., de Santana Jr., O. V. and da Costa, A. C. P. L. (2008). Mecateam 2008: Programação concorrente para futebol de robôs simulado, *VIII Escola Regional de Computação Bahia, Alagoas e Sergipe* .