

Autonomous Agents and Multiagent Systems

2nd Semester - 2015/2016

Agents in Wolf Pack



Group 17 - <Taguspark>

*<André Lima>* - <70572>

*<Samuel Gomes>* - <76415>

*<André Fonseca>* - <84977>

# Abstract

A solução de problemas multiagentes, principalmente quando envolvem aprendizagem por reforço têm enorme potencial mas são de difícil implementação. Este projeto visa criar uma solução a um problema multiagente do género de perseguição. Implementámos 3 tipos de soluções: reativas, deliberativas e de aprendizagem. Os resultados foram satisfatórios principalmente com a solução deliberativa (e com aprendizagem?). Com estas técnicas comprovamos como é possível resolver de forma rápida e eficiente este problema multiagente.

**Keywords:** multiagentes, arquiteturas reativas, arquiteturas deliberativas, arquitetura de aprendizagem, Q-learning modular

# Index

1. Introduction 4

2. The Scenario 4

3. Agent Architectures / Algorithms 4

3.1 Reactive Architecture 4

3.2 Deliberative Architecture 4

3.3 Hybrid Architecture 5

3.4 Coordination, Cooperation, Negotiation 5

3.5 Emotions Component 5

3.6 Learning Component 5

4. Comparative Study 5

5. Conclusions 6

6. References 6

# Introduction

In this section you should describe the purpose of the work, the problem that is being addressed, the approach and structure of the report.

**Notes:**

* The Introduction *is not* a copy of the abstract;
* Reports should not exceed 20 pages, including the cover, abstract, index and references, according to the format shown in this document.
* All references used in the development of the project must be clearly marked and placed in the References section.

# The Scenario

Este projeto é composto por 2 tipos de agentes num total de 5 agentes: 4 agentes do tipo lobo e um agente do tipo ovelha. Ambos os agentes têm os partilham as mesmas ações: mover para cima, mover para baixo, mover para a esquerda, mover para a direita. Só é possível mover para tais posições se estiverem desocupadas.

O agente ovelha é extremamente simples capaz apenas de se mover de forma aleatória pelo mapa. Acrescentámos também uma opção para uma versão reativa da ovelha capaz de fugir aos lobos próximos. Nesta situação a ovelha tem como sensor um campo de visão capaz de distância *d* capaz de identificar os lobos.

Os lobos são muito mais complexos. Na sua versão reative têm apenas uma variável associada, o seu *field of view* (não confundir com memória visto que não guarda informação apresentada ao longo do tempo. Esta é constante para cada lobo). Com ela podem identificar-se uns aos outros e à ovelha se estiverem dentro do alcance do mesmo.

Quando deliberativos, os lobos têm uma ação extra (*o ouvir)* que transmite uma mensagem aos outros lobos sobre que lado planeiam flanquear a ovelha. Para poderem saber o plano de outros lobos e a quem já enviaram mensagem, têm agora memória sobre estes vários valores.

Por último, na sua versão com aprendizagem, os lobos perdem novamente a capacidade de enviar mensagens mas têm agora conhecimento dos múltiplos *Q-values* possíveis para cada par “outro lobo-presa”.

# Agent Architectures / Algorithms

In this section you should describe the several architectures and algorithms according to what is required in the project’s description, *e.g.*, reactive, deliberative / BDI, hybrid, etc. The text should present a conceptual description mentioning aspects of implementation only if necessary for the understanding of the concepts. Some of the aspects to describe in each subsection are:

## Reactive Architecture

* The *<perception> \* → <actuator>* rules developed for each type of agent;
* The description of the arbitration mechanism of rules, *i.e.*, how they are ordered;
* The justification for any internal state variable that was used.

Para o problema proposto, os nossos agentes, os lobos, possuem um estado interno. Um campo de visão (field of view ou fov) que é, no máximo e em cada dimensão, metade do tamanho do mundo nessa dimensão. Este campo de visão é feito como um quadrado de dimensão d x d à volta do agente. Usando este estado interno, os agentes conseguem detetar se a presa está dentro do seu campo de visão. Usámos um número que representa d como estado interno, esse número foi usado como estado interno porque é uma característica dos agentes.

Em relação às regras perceção -> ação, estas foram desenvolvidas igualmente para todos os lobos.

Neste caso cada lobo verifica se a presa está no campo de visão, e tenta ir na sua direção.

Em último caso, quando algum lobo não consegue ver a presa, apenas escolhe aleatoriamente uma de cinco ações: andar em frente, para trás, para os lados ou ficar parado.

## Deliberative Architecture

No que diz respeito a esta parte do projeto, chegámos à conclusão que uma arquitetura BDI seria demasiada complexa para o tipo de agentes que considerámos.

Assim não seguimos a estrutura BDI dado que a implementação desta técnica se revelou opcional.

A nossa implementação resume-se a três módulos:

- A comunicação entre os agentes de modo a obterem um consenso sobre o seu sub-objetivo.

- Procura de caminhos: recorremos ao algoritmo A Star, tal como na aula de laboratório 3, para procurar caminhos mais curtos até cada uma das posições adjacentes ao lobo. Assim, usámos como heurística para atualizar os custos, a distância euclidiana entre pontos dado que mapeamos a solução ótima ao caminho mais curto. Os nós da procura são implementados como uma lista com 3 campos que representam o f, o g e uma posição na grelha (representada por uma lista com coordenadas).

- Estado Idle: quando um lobo não tem informação necessária para poder cumprir o seu objetivo, este entra num estado que consiste em vaguear pelo mundo em zigzag de modo a voltar a obter informação o mais rapidamente possível. Esse movimento é gerado a partir de operações locais que usam os limites do mapa para poderem mudar de direção quando conveniente. Esta operação têm em conta o field of view do agente, dado que quanto maior o fov do agente menor a frequência com que precisa de passar pelo centro do mapa (dado que adquire mais dados sobre o mundo a cada momento).

O agente deliberativo utiliza todos estes módulos de forma a minimizar o tempo demorado a encontrar a presa. A prioridade para o agente neste sistema é formar um plano em conjunto com os outros lobos. Isto é, obter uma posição adjacente à ovelha para a qual ele e só ele é responsável de ir e depois manter-se nessa posição. Se o agente não tem plano, assim que ele vê a ovelha, este é formado, e assim que possível tenta comunica-lo com os outros lobos. Se não consegue encontrar a ovelha, usa o movimento definido no estado Idle. Por último, se vê outros lobos e não lhes enviou mensagem sobre o seu objetivo atual, realiza tal tarefa nesse “tick”.

## Coordination, Cooperation, Negotiation

No sistema deliberativo, os lobos são capazes de comunicação de forma a coordenarem entre eles um plano de ataque. Cada lobo tem conhecimento sobre o flanco pela qual pretende atacar a presa, os flancos escolhidos pelos outros lobos atacar e informação sobre se os outros lobos ouviram ou não a sua mensagem.

Quando um lobo encontra pela primeira vez a ovelha, este decide encurralá-lo pelo lado que lhe está mais próximo. No entanto, se tem conhecimento que outro lobo já escolheu esse plano de ataque, decide por o segundo melhor, terceiro ou até mesmo quarto, dependendo dos planos já usados.

Assim que o lobo encontra um outro lobo, se ainda não lhe enviou a mensagem do seu plano de ataque, então envia essa mensagem para ter a certeza que o outro lobo sabe. Enviar a informação é algo penoso ao lobo pelo que ele tem de “parar e uivar” gastando um turno nessa ação. Decora também que a mensagem foi enviada àquele colega para não a enviar novamente sem necessidade.

Existem situações em que ocorrem conflitos e dois lobos escolhem atacar o mesmo flanco. Para resolver esse impasse, quando um lobo recebe uma mensagem que indica que já alguém escolheu o seu plano, o lobo desiste da sua ideia em prol da do colega e tem de escolher um flanco novo dos ainda disponíveis. Naturalmente, isso significa que terá novamente de avisar todos os outros lobos do seu plano novamente.

Esta última ação por parte dos lobos é bastante interessante visto que leva a um novo comportamento emergente em que, quando próximos de apanhar a ovelha, os lobos trocam algumas mensagens entre si até terem planeado completamente quem ataca por que lado.

## Learning Component

If learning techniques were used in the project, detail the aspects on which learning was incorporated, what learning algorithms were used and how they were implemented in the specific case of the project.

Devido ao aumento exponencial do espaço de estados de” Q-learning” quando aumenta o número de agentes, implementámos uma técnica diferente: “Q-learning modular”.

**Note:** This section is especially important for the project “*Wolf Pack”*.

# Comparative Study

In this section you should present the comparative study made between the several approaches explored in the creation of agents. In particular, the study should be able to provide conclusions about which of the approaches explored contributed more to the "success" of the agents.

Some of the aspects to describe this section are:

* The approaches that will be compared in the tests;
* The metrics used to determine the "success" of each agent / team;
* The tests that were designed;
* The results obtained in each studied condition (charts, tables, etc.);
* A statistical analysis of the results;
* Conclusions (limitations / advantages of each approach, etc.).

**Notes:**

* Any conclusions must be substantiated by the results of the experiments;
* This section applies to all projects, *i.e.*, even if the requested comparative study does not refer to the several architectures used, the general indications described above remain valid, only the object of the comparative study changes.

# Conclusions

This section should summarize the proposed solution to the problem presented in the Introduction. In particular, you should present an interpretation of the results described in the Comparative Study section.

**Note:** It is also expected in this section to describe how the work presented could be extended in order to motivate future work.

# References

This section should list all the references considered relevant to the understanding of the work presented. Here are some examples of reference formatting for several types of publication:

**For a book:**

1. D. Patterson and J. Hennessy, *Computer Organization and Design: The Hardware/Software Interface*, San Francisco, CA, USA: Morgan Kaufmann Publishers, 1997.

**For a journal article:**

1. B. Kernighan and S. Lin., An Efficient Heuristic Procedure for Partitioning Graphs. *The Bell System Technical Journal*, Vol. 49, No. 2, pp. 291-307, February 1970.

**For a paper included in the proceedings of a conference:**

1. F. Fallah, S. Devadas, and K. Keutzer. OCCOM: Efficient Computation of Observability-Based Code Coverage Metrics for Functional Simulation. In *Proceedings of the Design Automation Conference*, pp. 152–157, June 1998.

**For a dissertation:**

1. D. Cheng. *Power Estimation of Digital CMOS Circuits and the Application to Logic Synthesis for Low Power*. PhD thesis, University of California at Santa Barbara, December 1995.

**For a technical report:**

1. E. Sentovich. *SIS: A System for Sequential Circuit Synthesis.* University of California, Berkeley, April 1992.

**For a webpage:**

1. *Instituto Superior Técnico, Official webpage*. Retrieved from: <http://tecnico.ulisboa.pt/>. Last accessed April 2016.