STUDIES OF DYNAMICS OF PHYSICAL AGENT ECOSYSTEMS

Israel MUÑOZ MORENO

ISBN: 84-688-0973-X
Dipòsit legal: GI-424-2002

UNIVERSITAT DE GIRONA

Departament d’Electrònica, Informàtica i Automàtica

THESIS

STUDIES OF DYNAMICS OF PHYSICAL AGENT ECOSYSTEMS

Israel Muñoz Moreno

Director de Tesis:
Josep Lluís de la Rosa Esteva

Girona, February 2002

Abstract

This thesis addresses the problem of learning in physical heterogeneous multi-agent systems (MAS) and the analysis of the benefits of using heterogeneous MAS with respect to homogeneous ones.

An algorithm is developed for this task; building on a previous work on stability in distributed systems by Tad Hogg and Bernardo Huberman, and combining two phenomena observed in natural systems, task partition and hierarchical dominance. This algorithm is devised for allowing agents to learn which are the best tasks to perform on the basis of each agent's skills and the contribution to the team global performance. Agents learn by interacting with the environment and other teammates, and get rewards from the result of the actions they perform. This algorithm is specially designed for problems where all robots have to co-operate and work simultaneously towards the same goal. One example of such a problem is role distribution in a team of heterogeneous robots that form a soccer team, where all members take decisions and co-operate simultaneously. Soccer offers the possibility of conducting research in MAS, where co-operation plays a very important role in a dynamical and changing environment. For these reasons and the experience of the University of Girona in this domain, soccer has been selected as the test-bed for this research. In the case of soccer, tasks are grouped by means of roles.

One of the most interesting features of this algorithm is that it endows MAS with a high adaptability to changes in the environment. It allows the team to perform their tasks, while adapting to the environment. This is studied in several cases, for changes in the environment and in the robot's body. Other features are also analysed, especially a parameter that defines the fitness (biological concept) of each agent in the system, which contributes to performance and team adaptability.

The algorithm is applied later to allow agents to learn in teams of homogeneous and heterogeneous robots which roles they have to select, in order to maximise team performance. The teams are compared and the performance is evaluated in the games against three hand-coded teams and against the different homogeneous and heterogeneous teams built in this thesis This section focuses on the analysis of performance and task partition, in order to study the benefits of heterogeneity in physical MAS.

In order to study heterogeneity from a rigorous point of view, a diversity measure is developed building on the hierarchic social entropy defined by Tucker Balch. This is adapted to quantify physical diversity in robot teams. This tool presents very interesting features, as it can be used in the future to design heterogeneous teams on the basis of the knowledge on other teams.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Introduction

Over the past decade agents Multi-Agent Systems has become one of the most active research subfields in Artificial Intelligence (AI). In recent years agents have played a key role in the development of the Internet, where users are continuously benefiting from these technologies in many different ways.

Agent is a vague concept, there is no clear definition, and each researcher defines this term depending on his/her own interests. Despite the lack of agreement on what an agent is, all works share the same idea. An agent is seen as an autonomous entity (program, system, car...), intelligent and that tends to communicate and co-operate with other agents, in order to accomplish its goals.

[Wooldridge 1999] gives a general definition of agent:

" *An agent is a computer program that is situated in some environment, and that is capable of autonomous acting in this environment in order to meet its design principles*".

This definition can be extended to define which are the conditions for calling an entity, an intelligent agent [Wooldridge and Jennings 1995]:

- *Reactivity*: intelligent agents are able to perceive their environment, and respond in a timely fashion to changes that occur in it in order to satisfy their design principles.

- *Pro-activeness:* intelligent agents are able to exhibit goal-directed behaviour by taking the initiative in order to satisfy their design principles.

- *Social ability:* intelligent agents are capable of interacting with other agents (and possibly humans) in order to satisfy its design principles.

One of the most important aspects in agents is social ability. Social ability can be understood [Wooldridge 1999] as the necessity to negotiate and co-operate with other agents to achieve its goals.

Other fields have also benefited from this area of research and/or have contributed to it. One of the most interesting areas that have contributed to the field of MAS is biological systems.

Phenomena observed in nature have helped many times researchers to design intelligent systems, which meet the previous requirements. The reason for this interest in these systems is that they present features that are considered desirable for the design of intelligent systems. This has prompted researchers to study such systems and develop models that emulate the behaviour observed in nature.

On the other hand, one of this fields that has benefited from the development in MAS is multi robot systems. Multi Robot Systems has drawn a lot of attention over the past decade and it has become one of the most attractive research fields. Initially, research in the field of robotics focused on a single robot, but in the late eighties and early nineties this interest started to shift to systems composed of several robots. These systems have been applied to perform many different types of tasks, mapping and exploration, transport, rescue.... Despite the number of applications, most research in this subfield of robotics has centred on homogenous robots. In the last few years, there is an increasing interest in developing systems composed of heterogeneous components, but there is still a lot of work to be done in this area.

This thesis brings together concepts from natural systems, MAS and robotics. This thesis focuses on the study of homogeneous and heterogeneous robots and develops an algorithm that allows agents to organise in a team depending on their different physical features and the environment. For this analysis, the domain of soccer has been selected. Soccer has become in the past years one of the most attractive test-beds to test and develop multi-robot systems, as it presents some unique features. Soccer allows robots to work in dynamic and adversarial environments, where co-operation and teamwork play a very important role. Over the past years several competitions have taken place all around the world, in which teams made up of several robots play against other teams.

For this work a more specific definition of the term agent is used. This definition is based on [Stone 1998] and it is selected, because this term is more concrete and it is related somehow to this thesis. According to [Stone 1998]:

"An agent is an entity with perceptions, goals, cognition, actions, and domain knowledge, situated in an environment"

We would like to add to this definition that

"....agent behaviour is conditioned by its physical body, namely, dynamical constraints."

This means that some state variables (speed and acceleration) change over time according to physical laws.

1.2 Thesis Objectives

This thesis aims at studying several aspects related to the field of heterogeneous MAS, where physical constraints determine agent abilities to interact with the environment. This type of agents is also called physical agents. Chapter 2 defines more in depth the objectives of this thesis, but in this section these objectives are outlined:

1) Development of an algorithm to allow heterogeneous MAS to learn how tasks have to be distributed among the members of the team.

This point is addressed using a novel approach. This new approach is based on two nature principles, dominance hierarchies and task partition in insect and animal societies. They are used to define a new model, building on a previous work by [Hogg and Hubermann 1991], where a model is presented to model interactions in a group of heterogeneous agents. This model is adapted and some of the concepts are redefined for the purposes of this thesis. This algorithm is specially conceived for domains, where agents have to co-operate and work together towards the same goal, by performing different task simultaneously. In this type of problems, tasks can be assimilated to the roles that have to be filled to achieve the final goal. One example of this type of problem is the one analysed in this thesis, role distribution in a team of robots.

2) Analysis of the impact of physical heterogeneity in the performance and adaptability of the team.

A set of 5 different robots is defined in this thesis. These robots are used to build several teams of homogeneous and heterogeneous of robots. These teams are tested in different environmental conditions and several properties are evaluated, in order to compare properties between homogeneous and heterogeneous teams of robots.

3) Define a measure of robot physical diversity.

A measure has been defined to allow researchers to carry out a rigorous analysis of the impact of physical heterogeneity in the system. The metric developed in this thesis is based on the work of [Shanon 1949] on information entropy and [Balch 2000] on hierarchical social entropy. This thesis extends this later work and defines a set of benchmarks to evaluate different skills of robots, which are used later to measure team physical diversity.

1.3 Thesis Organisation

This thesis is organised in different chapters that address some of the aspects already mentioned.

***Chapter 2. Related Work*.** This chapter starts by explaining the general problem that has motivated this thesis. Next, the literature related to this thesis is reviewed. It centres mainly on two different aspects, natural systems and heterogeneity. Natural systems analyses how natural phenomena have been used as design principles for MAS, describing some of the most interesting applications. Heterogeneity reviews some work done in the field of heterogeneous multi-robot systems and some of the properties that have been observed in systems composed of heterogeneous components. At the end of this chapter, the objectives of this thesis are defined and motivated on the basis of the open problems found in the literature.

Chapter 3. Model Definition. This chapter describes how the problem of learning role distribution in a physical MAS is approached. The solution to this problem builds on a previous work, which is reviewed. All the aspects related to this algorithm are analysed from a general perspective. This approach is compared later to other existing algorithms used for similar tasks.

***Chapter 4. Experimental Set-up*.** This chapter describes the experimental set-up of this thesis. It details also some of features of the soccer domain and analyses some of the tools that can be used to conduct research in this domain. It also analyses how the different parts of the model, described in the previous chapter, have been implemented in the domain of soccer.

Chapter 5. Model Analysis. This chapter analyses some of the properties and features of this model. This analysis focuses especially on the process of self-organisation and on the property of team adaptability. The model defined in chapter 2 endows MAS with adaptability and robustness to changes in the environment and physical changes in the members of the system. It also studies how the formation of a hierarchical dominance affects the performance and adaptability of the system.

***Chapter 6. Heterogeneity metric*.** This chapter defines the heterogeneity metric developed in this thesis. It starts by reviewing in depth the works by [Shanon 1949] and [Balch 2000] to define a metric of heterogeneity. This last work is extended to measure physical diversity. Next each team diversity is measured using this technique and set of benchmarks are defined, where each robot skills is evaluated.

Chapter 7. Heterogeneity analysis. This chapter focuses on the analysis of performance and task distribution in homogeneous and heterogeneous teams in several types of games. In total five different teams of homogeneous teams are analysed, while 6 different heterogeneous teams are created. All these team are evaluated against three hand-coded teams and they also play against the rest of homogeneous and heterogeneous teams. The results are also analysed using the physical heterogeneity metric defined in chapter 5.

Chapter 8. Conclusions.

Chapter 9. Future Work.

Bibliography

Annexe. This annex describes the different parts and the implementation of a team of robots in soccer. It specially details the roles that have been defined and how the different actions are selected depending on the state of the environment.

1.4 Term Use

In this thesis there are some concepts that are used indistinctly. There are two cases where this can be observed. In the first case, the term physical agent and robot are used in many cases as synonyms. These two terms do not have always the same meaning, but they are used in this thesis indistinctly. Here, a robot is considered an autonomous physical system, which has been conceived, from the point of view of the intelligence, as an agent. This definition can be extended to multi-robot system and physical multi-agent systems. This latter term is more general than a multi-robot system, but it can also used to refer to such systems when they are designed from an agent approach.

The second group of term that are also used indistinctly are tasks and roles. Task is a general term that may include different concepts related to the responsibilities of an agent or group of agents. Role is a more specific term and can be understood as a set of tasks that has to be accomplished by the agent filling a given role.

Chapter 2

Related Work

2.1 Introduction

The development of agent based technologies and their deployment in real world applications has started to change how users interact with the virtual world. These technologies will become more important as agent technologies are included in smart devices and cellular phones. In the future, everyone will be able to buy toasters, refrigerators, microwaves, washing machines... that will make use of agent technologies. Agents will act on behalf of their users, for example an agent built-in in the washing machine will surf the net to find more efficient ways of washing clothes; an agent acting on behalf of the refrigerator will order food from the supermarket depending on the products already stored in it. These are only two examples of the benefits for the users that the development of agent technologies will bring. The existence of these agents acting on behalf of people, organisations and devices will give rise to many different problems and will bring new challenges to researchers.

Some of these questions have been already raised by the European Union under an initiative called Universal Information Ecosystems (UIE) http://www.cordis.lu/ist/fetuie.htm. This initiative aims at exploring new technologies that will allow individuals and organisations to benefit from the global information infrastructure (INTERNET). This project defines the concept of infohabitant, this concept defines the virtual entities that will behalf of their owners, as well as smart appliances,... The main contribution of these technologies should be:

"The development of an environment that supports the dynamic creation of new types of relations and activities and, in doing so, creates value and degrees of scalability, sustainability and robustness that are well beyond what can be envisaged today"

Systems with similar features already exist in the real world. Maybe the best known example of this category is the Stock Market. The stock markets are composed of a large number of components, which act on behalf of their customers, and that sell and buy stocks in a changing environment, which is modified as a result of the decisions of each stock broker or external factors, out of the control of the members of the market. As a result of the interactions among the members of the market and the changing environment, the collective behaviour of the system, which can be represented by the market indexes, shows complex behaviours that may lead to market crashes or financial bubbles. One of the features of these behaviours is the difficulty to predict them. This kind of unexpected behaviours may be also observed in the future when millions of agents will be able interact in the same environment.

All these systems share some common features, as it may be observed in a stock market: ([Wooldridge 2000] also mentions some of these aspects):

- ***Asynchron interactions.*** Components may interact with other components at any given time.

- ***Heterogeneous components.*** Heterogeneity is a key feature of these systems, components may be different at different level: strategies, protocols, programming languages... In the stock market heterogeneity is responsible for the stabilisation of the supply and demand of shares.

- ***Components change over time.*** Components may change due to the experience gathered over the time, which leads to an improvement in their negotiation strategies, market analysis...

- ***Changing environment.*** The environment may change as a result of the changes in the components of the system: new components are added, components may learn from experience, or the decisions of the members of the environment, as their actions affect the state of the environment.

- ***Large number of members.*** Such systems are composed of thousands, millions of components that interact and change the environment. These components can also join or quit the market at any given time.

- ***Partial information of the system.*** It is not possible for any component to have complete, up-to-date information about the state of the system. [Wooldridge 2000]. At the stock market brokers do not have information about the shares price in the future, which events will affect the economy, which strategies follow other brokers...

- ***Emergence.*** The structure of the system is not predetermined, but emerges as a result of interaction and self-organisation on behalf of the system [Wooldridge 2000].

According to the observed problems in the stock markets and some of the problems already studied in the field of agent-based technologies, such systems and their components should be able to deal with different aspects

- ***Integration of agents of different types***, which have been developed with different technologies and may use different communication protocols.

- ***Stability of the system***. The term stability is related to several different concepts. In the stock market the stability of the whole system is related to the market indexes. One example in virtual world are auctions of goods, with several agents bidding for goods, services... One possible concern for the members of the system is the stability of the price of goods.

- ***Performance of the whole system***. How users may benefit from participating in this environment, with respect to the existing technologies that serves the same purpose.

- ***Adaptability***. members of the system should be able to adapt to changing environments, so that they may benefit from the new opportunities that the environment offers.

- ***Co-operation among components of the system*** with different capabilities, in order to benefit from the joint resulting capabilities of the group.

This thesis was originally motivated by the problems and questions associated to systems containing large collections of agents, and it initially centered on the problem of stability, building on the work by [Hogg and Huberman 1991], where the problem of the stability of the decisions of a large group of agents is studied. Although this was the main motivation for this thesis, later the research focused on a different problem; but always bearing in mind some of the problems and challenges related to this general setting.

Building on [Hogg and Huberman 1991], a model for stablishing and learning agent relationship is developed. This model includes two phenomena observed in animal and insect societies, task partition and dominance hierarchies. It models how members of these societies interact with other members of the group and the environment, and how they compete for bounded resources to survive in the environment, this process results in an organisation of tasks and responsibilities of each member within the group. Natural mechanisms observed in animal societies show that complex organisational behaviours, for instance partition of tasks within a group, might be achieved by means of simple mechanisms. One of the features of these mechanisms is the scalability, millions/thousands of entities are able to co-ordinate in efficient way. Also these mechanisms provide the system with robustness and adaptability. These properties have been already mentioned as some desirable features of the systems containing large numbers of agents. Some of these properties are explored at a different scale and are studied in an environment that shares many properties with this general setting. Another interesting aspect of this algorithm is the adaptability of the team to adapt to changes in the environment or in the components of the team.

The test-bed selected is soccer. Soccer shares most of the general properties already detailed, except for the number of agents, as the size of the teams tends to be small (five players in each team in this thesis). Soccer has drawn a lot of attention over the past years as a platform to conduct research in the field of Multi-Agent Systems. Soccer is a test-bed for small and medium teams of agents. One of the main features of this test-bed is the need for co-operation with other agents, in a changing and adversarial environment. These features make it extremely attractive for researchers interested in the deployment of Multi-Agent Systems (MAS).

With this algorithm, the goal is to study the impact of heterogeneity in physical agent teams.

This thesis integrates aspects from different fields of research in AI and related fields. This chapter reviews literature from these fields, but it only focuses on the general concepts of these research fields and on those works tightly related to this thesis.

This chapter is organised into the following sections:

Nature Inspired Systems. This section studies how natural systems, in particular animal and insect societies have inspired the development of algorithms, mechanisms... in the field of AI. This section focuses specially on the concepts from nature that have inspired this work. This section centers on the following specific topics, ecosystems, self-organisation, swarm intelligence and hierarchical societies.

Soccer. This section reviews some general concepts on soccer, as this is the test-bed used for this research. Research on soccer related to the aspects studied in the different parts of this chapter is detailed in the corresponding section.

Heterogeneity. This section analyses some key aspects of heterogeneous systems, as this is the type of Multi-Agent Systems that are studied in this thesis. This section reviews some of the properties found in this kind of system, and some of the metrics developed for measuring heterogeneity in such systems.

Heterogeneous Multi-Robot Systems. This section studies how heterogeneous robot systems have been designed and which mechanisms have been developed for the efficient co-ordination of such systems. It also reviews some applications, especially in the field of robotic soccer.

Learning in MAS. This section concerns learning in MAS. In this section three different aspects are analysed related to this thesis, learning in soccer, learning in heterogeneous MAS, and co-learning.

Physical Agents. Related to the section on heterogeneity on physical systems, this section studies some key concepts on physical agents, and how the knowledge on the body of a physical entity, for instance a robot, may be beneficial for the co-ordination with other agents.

At the end of this chapter the objectives of this thesis are motivated in relation to the state of the art.

2.2 Nature Inspired Systems

2.2.1 Introduction

Nature has many times inspired researches to find solutions to complex problems and has motivated the development of new AI techniques. Some examples of research inspired by nature phenomena are Neural Networks, which imitates the human brain, Genetic Algorithm/Programming, which draws on genetic evolution to solve complex problems, or Swarm Intelligence, which develop algorithms and mechanisms to carry out complex tasks inspired by the complex behaviours observed in insect and animal societies, emerging from many simple individual behaviours.

This section reviews some phenomena observed in animal and insect societies, on which this thesis is based, as well as some applications of these phenomena in the field of MAS and robotics.

[Parunak 1997a], [Parunak 1997b] and [Brueckner 2000] argue that it is possible to build MAS, which are able to show complex behaviours by means of simple agents, instead of using complex agents. The power of this approach lies in the dynamics of the interactions of these agents. These properties can be found in natural ecosystems, for example in insect colonies. Insect colonies are made up of millions of simple entities, which show robust, flexible and adaptive behaviour.

Their approach proposes a set of design principles for designing multi-agent systems, some of them, the most interesting ones, are reviewed here:

- Agents should be **Small in Mass**
  - Small components are easier to design
  - Favour specialised agents vs. general ones
  - Richer space of possible behaviours

- Agents should be **Small in Time**
  - Agents should be able to forget and to improve by learning and adapting

- Agents should be **Small in Scope**
  - Limited sensing, although high-bandwidth is possible
  - Small impact from the activities of an agent on the environment

- Support Agent Diversity
  - Diverse agents cover more environments’ state

- Diverse agents can provide better performance and be more robust.

- Redundancy
- Introduce redundant elements in the ecosystem

- Decentralisation
- Agents must explore the environment by themselves encountering other agents as a source of information or services

- Feedback and Reinforcement
- Knowledge, skills or behavioural patterns that produce positive results should be reinforced

- Share information
- Agents pass information to each other at different levels.

- Randomisation
- A random factor in the agent decisions to prevent emergence of negative synchronism.

- Plan and Execute Concurrently
- Agents should respond to changes in the environment instead of planning and executing

They adopt these principles to design a MAS for shop floor planning. This design is based on how ants connect their nests with available food sources.

2.2.2 Self-Organising Systems

Self-organisation is a phenomenon intrinsic to many different kinds of systems. Self-organisation was first introduced in the field of physics and chemistry to describe how complex behaviour emerged from the local interactions of distributed individuals, giving rise to macroscopic behaviours. Perhaps the best know example is the perfect gas equation of state. This equation describes how gases behave at a macroscopic level, while particles behave chaotically and interact among them at a microscopic level. Today self-organisation can be found in many different fields: biology (insect and animal societies), economics (a stock market is a system made up of millions of individuals), chemistry (thermodynamics), physics...

[Yovits, Jacobi and Goldstein 1962] are the first to define the term self-organisation as:

"*A self-organising system is a system that changes its basic structure as a function of its experience and environment*"

In [Lucas 1997] self-organisation is defined as:

" The evolution of a system into an organised form in the absence of external constraints. A move from a large region of state space to a persistent smaller one, under the control of the system itself".

For [Rocha 1999]:

"*A self-organising is seen as the process by which systems of many components tend to reach a particular state, a set of cycling states, or a small volume of their state space (attractor basins), with no external interference. This attractor behaviour is often recognised at a different level of observation as the spontaneous formation of well organised structures, patterns, or behaviours, from random initial conditions (emergent behaviour).*"

The concepts of attractor and emergence are part of the terminology used in the self-organisation. Attractor is defined by [Lucas 1997] as:

"as a preferred position for the system, such that it the system is started from another state it will evolve until it arrives at the attractor, and will then stay there in the absence of other factors".

and emergence

"the appearance or feature not previously seen".

[Kauffmann 1991] thinks that self-organisation is an "*inherent property of some complex systems*" and that some complex biological systems tend towards self-organisation. [Bonabeu, Theraulaz et al 1997a] claim "*that complex collective behaviours may emerge from interactions among individuals that exhibit simple behaviours"* in insect societies (ants, bees...). Some examples can be found in the next section on Swarm Intelligence.

In AI and multi-agent systems these concepts have been also adopted. For [Mataric 94] emergent behaviour is characterised by:

" Global states or time-extended patterns that are not explicated programmed in but result from local interactions between a system's components... These types of systems are referred to as "self-organising" because of their apparent ability to create order".

Emergent Behaviours can be seen in some cases a result of a self-organising process. From a different point of view [Brooks 1989], but based on the same idea, thinks that possibly human behaviour is chaotic from which "*order appears to emerge*".

Robotics have benefited from these properties, especially in the field of swarm intelligence. For [Unsal 1993] a self-organising system has some advantages over "pre-programmed systems". A self-organising system is based on individuals requiring simple programming and communications. This allows to design and co-ordinate systems composed of large number of individuals in a simple way. Moreover these systems are very reliable and adaptive.

For [Parunak, Brueckner and Sauter 2001] emergent behaviour has two important features, this behaviour may be more complex than the behaviour of any individual component, and it may be qualitatively different than individual behaviours.

However, for the same authors [Parunak and Brueckner 2001] the natural tendency of a group of autonomous processes is to disorder. This poses a threat to multi-agent systems. One possible way of solving this problem is by constraining the behaviour of these agents, this is the typical approach, so that the system exhibits only a part of its potential. This possible solution bounds the potential applications of MAS.

However, this problem may be solved by adding information to the group. This information can lead to an increased organisation in the system if it is added in the right way. How this information should be added and how researcher may take advantage of these features is not clear, but the authors think that concepts of thermodynamics can play an important role in the future to exploit this feature in MAS.

2.2.3 Ecosystems

According to Encyclopaedia Britannica, an ecosystem is:

"The complex of living organisms, their physical environment, and all their interrelationships in a particular unit of space.

The principles underlying the study of ecosystems are based on the view that all the elements of a life-supporting environment of any size, whether natural or man-made, are parts of an integral network in which each element interacts directly or indirectly with all others and affects the function of the whole..."

Ecosystems are complex biological systems in which adaptation is an essential characteristic [Devine, Paton and Amos 1997]. Natural ecosystems are composed of members of different species that usually compete for bounded resources, normally food. The ability of each individual to obtain resources determines its possibility to survive in the ecosystem. Some mathematical models have been created to describe how the populations (number of members of a given class) of species change over time, depending on the available food, population of other species… in the ecosystem. One of the best known ecological models is the Replicator Dynamics. It models the rate of increase/decrease of the species that compose the ecosystem, depending on the relative success (ability to get food, to reproduce…) of each species with respect to the others. This model is studied in more depth in chapter 3, as it is part of this thesis.

These ecological/biological concepts have given rise to the development of optimisation algorithms known as evolutionary algorithms. In these algorithms solutions to the problem are equivalent to species in the ecosystem, the most successful species (the success of a species is measured by the fitness value) tend to combine sexually (crossover) to create new ones. One of the most famous evolutionary algorithms is Genetic Algorithm (GA) [Goldberg 1989].

These biological concepts of competition for resources among the components of the system were adopted in what it was named Computational Ecosystems. Computational Ecosystems were developed mainly in the late eighties and early nineties. The idea behind this term is the allocation of computational resources to competing tasks that have to be completed on idle heterogeneous workstations existing in a network [Huberman and Hogg 1993], this process of competition for resources leads to the formation of complex behaviour at the macroscopic level. This process is also known as a self-organisation process (see previous section). This resource allocation problem becomes a very complex one, when the number of tasks and workstation increases, for which the optimal solution is difficult to find. A widespread solution to this problem was the development of distributed approaches.

On the other hand, Computational Ecosystems present some differences with respect to natural ecosystems. Local rules of each individual are defined by the designer, in biology these rules are fixed by genetics. Moreover the goal of the components of a computational ecosystem is to complete tasks as soon as possible, unlike biological ecosystems, where individuals try to survive as long as possible.

Apart from using concepts from biological ecosystem, concepts from market economies mechanisms are adopted for the allocation of resources. The reason for using this kind of mechanisms is the efficiency of resource allocation in market economies; [Fergurson 1989] proves that this approach is more efficient than non-economic approaches. These mechanisms are basically the use of currency, pricing and auctions; see [Fergurson et al 1996] for a detailed survey of these mechanisms.

Several applications of these concepts have been developed over the past decade. For example Spawn [Waldspurger et al 1992], which is one of the most known computational ecosystems. Concepts from computational ecosystems have been adopted to solve problems where resources are bounded, but they are not necessarily computational resources. For example in [Clearwater and Huberman 1994] a similar approach is used to manage air transport in buildings environments, or in [Ygge and Akkermanas 1996], [Ekbom and Astor 1996], where these mechanisms are applied to power supply management. In a more recent work, [Wooldridge 2000] proposes the creation of an agent-based computational economy on the Internet, in which software agents will sell and buy computational resources on behalf of their owners, in order to process complex tasks that can be only processed on supercomputers.

This proposal of creating a computational economy on the Internet may be seen as an example of systems composed of a large collection of agents that will act on behalf of their owners. The author of this paper sketches out some of the challenges that may face designers building such systems, among them some of the problems already mentioned (stability).

[Hogg and Huberman 1991] developed a model to study computational ecosystems. This model describes how agents choose among several available resources based on the perceived pay-off for using these resources. Pay-off increases as more agents make use of these resources, as co-operation among them arise, up to a given value, but at the same time conflicts also increase with the number of agents. In addition these agents access the information on resource usage with delays, this results in a chaotic resource usage, deteriorating the system performance. By means of a reward mechanism the resource usage of the system stabilises near the optimal. This model was later analysed by [Glance 1993]. Hogg and Huberman's work is reviewed in depth in chapter 3, as this thesis is inspired by this work.

These ideas were developed mainly in the late eighties and early nineties. However the use of biological and economic concepts for resource allocations is drawing a lot of attention, as all the experience gathered in this area is being used to develop mechanisms for agent based infrastructure on the Internet (negotiation, auctions...). Several groups around the world are working on this issue. This thesis also benefits from this area as one of the firsts problems that dealt with the problem of having large population of agents was studied by [Hogg and Huberman 1991]

2.2.4 Swarm Intelligence

Swarm Intelligence has become a very popular area of research over the past years. Swarm Intelligence is defined as follows [Bonabeau, Dorigo and Theraulaz 1999]:

"Any attempt to design algorithms or distributed problem-solving devices inspired by the collective behaviour of social insect colonies and other animal societies"

According to the authors a social insect colony is a decentralised problem-solving system comprised of many simple interacting entities. Elements exploit explicit and implicit communication and agent-agent and agent-environment to achieve complex behaviours. They can solve problems in a very flexible and robust way, and the mechanism allows adaptation to changing environments. Robustness is also a feature of these systems; they are endowed with the ability to work even though some members may fail to perform their tasks.

Observing and analysing insect colonies can help to design artificial problem-solving systems that through self-organisation are able to solve complex problems. The most common way of achieving this behaviour is by trying to build models of how these systems work. These models can be used later to create MAS with agents endowed with the same set of behaviours as the elements of such systems, and achieving complex behaviours through self-organisation and interaction of the components of the system.

Several researchers have been trying to build up such systems over the past years achieving important results. Perhaps the most popular societies are ants, which have been studied in depth and many algorithms and applications have been developed by studying their social behaviour.

Perhaps the most popular application is Ant Colony Optimisation (ACO) [Dorigo, Maniezzo and Colorni 1996]. ACO is general-purpose algorithm to solve different combinatorial optimisation problems. This algorithm mimics the behaviour of real ants. These ants are very simple agents with very simple capabilities that try to emulate how ants build paths from the nest to the source of food in an optimal way. These heuristics have achieved remarkable results in complex problems as job-shop scheduling. These results has prompted many researchers to apply these ideas to solve combinatorial problems in other domains, for example network routing [Bonabeau et al 1998]... In [Maniezzo and Carbonaro 1999] several applications of this algorithm used to solve other complex problems can be found.

Ant Colonies have also been a very important approach for designing multi-robot systems. Swarm Intelligence in robotics means to use a group of simple robots (simple sensors, computational resources, behaviours...) to achieve a reliable, fault-tolerant and more flexible behaviour. Some applications of this approach [Unsal and Bay 1994] and [Stilwell and Bay 1993], show that this is possible, as in the problem of material transport, where several robots are needed, by using a population of robots with limited sensory and communication capabilities and behaviours. This complex behaviour is also achieved by means of a self-organising process. (See Self-Organisation section for more details).

One of the most interesting aspects of insect colonies with respect to this thesis is task partitioning. One example can be found in bees where foragers collect nectar, which is transferred to bees working within the nest to store it. Task partitioning [Ratnieks and Anderson 1999] is a phenomenon in which a piece of work is divided among two or more workers. This is a widespread phenomenon in insect societies as ants, bees, wasps, termites [Bonabeau et al 1997a], resulting usually in a benefit for the whole society. The most important sign of division of labour is the existence of castes, there are three kinds of castes: physical, behavioural and temporal [Martinoli, Goodman and Holland 2001]. The most interesting with respect to this work, is the specialisation of workers in some tasks as a result of the worker experience or physical abilities. [Ratnieks and Anderson 1999] survey some literature on this issue, they concluded that task partitioning by individual differences can occur as a result of several factors:

- ***Age***, in the example of the bees, foragers and bees working within the nest have different age.
- ***Size***, in Daceton armigerum medium-sized workers usually capture pray, but if any of these workers finds on its way back to the nest a larger worker, the pray is transferred to it.
- ***Strength***, in Pheidole pallidula minor workers pin down ants and physically stronger members decapitate these ants.
- ***Skills***, in specie of ants, leaves are cut into small pieces by arboreal cutters, the rest of ants pick up the dropped leaves.
- ***Experience***, in Lasius fuliginosus preys are transferred to workers familiar with a particular patrolling area and which are able to run much faster.

The advantage of workers specialised on a task and having above average skills for this task is that these workers will be more efficient than the average ones. On the other hand the costs of the specialisation of below-average workers can reduce the average efficiency. The problem is very similar with workers with different experience.

One example of the usefulness of studying task partition in insect societies is the work by [Bonabeau et al 1997b]. They study the colonies of Eusocial wasps to formulate an adaptive task allocation algorithm for mobile agents.

These phenomena show that physical heterogeneity affects how tasks are partitioned across the different members of the group, resulting in an increase in the performance of the whole community, as components carry out those tasks for which they are better fitted. This aspect is studied in this thesis for the case of a team of heterogeneous robots. From the point of view of insect castes, these robots are heterogeneous at three different levels, size, strength and skill.

2.2.5 Hierarchical Societies

Animals, insects, human... who live in social groups, establish a social structure called dominance hierarchy within their group. This hierarchy serves to maintain order, reduce conflicts and promote co-operation among group members. A position within the dominance hierarchy is established by each member of the group, based on the outcomes of the interactions between these members. The most dominant animals, called also alpha members, can control access to valued items such as food, den sites and mates. A Danish scientist, Thorlief Schelderup-Eppe, formulated this concept in the 1920's, after observing hens and noticing that they lived in hierarchies. The process of formation of a social hierarchy is a progressive one, where individuals fight several times with other members of the group, usually the one that wins most of the fights against a given member of the group, is dominant with respect to the loser. In many species the likeliness of winning a fight depends also on past experiences, one member that has win most of the past fights (against different members of the group), it is more likely to win the next one. There are always new members trying to challenge the dominant one [Bonabeau et al 1999], but usually the dominance-submission relationship remains stable over a period of time.

Dominance behaviour is present in many animal societies (hens, cows, fish, crabs, primate, wasps...). One of the best known examples of social hierarchies is the wolf (canis lupis). Wolves compete for rank within the pack, for mates and for any scarce resources such as food [Ryon 2000]. Usually the dominant wolf is the first to have access to food and reproductive opportunities. Some researches suggest that dominance hierarchies serve to maintain order and conserve energy within the pack, resulting in a more efficient relation within the group.

In the literature there are numerous examples of this phenomenon. Some researchers claim that physical differences affect largely the process of formation of a hierarchy. [Beaugrand and Cotnoir 1996] study the role of individual differences in the formation of differences of male green swordtail fish, although the final result is also conditioned by previous experience, results show that large individuals are more likely to win the fights. Physical superiority can be found also in crayfish [Goessman, Hemelrijk and Huber 2000]. In wasps [Bonabeau et al 1999] dominance order among female members determines the division of labour among team members. The dominant female remains on the nests where she lays the eggs and contributes to build the nest, while the others perform other tasks outside the nest. The results of the contests are conditioned by the physical superiority of the members.

This last example is extremely interesting within the scope of this thesis, as the phenomenon of task partitioning and hierarchical dominance are combined for assigning responsibilities among the members of a group. These two processes are combined and emulated, as it has been already mentioned at the beginning of this chapter, to task distribution in teams of heterogeneous components.

Concepts from hierarchical societies have been used already in robotics and multi-agent systems.

[Tomlinson, Blumberg and Rhode 2000] propose dominance hierarchies as a way of implementing MAS and a means of understanding them. Coding domination hierarchies can be used to:

- Organise relations among autonomous entities trying to accomplish their goals at the same time under limited resources (computing power, money...)
- Assist in the formation of alliances among agents
- Help humans interact with a system, as they can work in a more human way.

Although the ideas seem promising, they are still been developed in a project called Alpha Wolf [Tomlinson and Blumberg 2001], which is still on the early stages. In this project they attempt to model the dominance and submission in a wolf pack. They claim that understanding social behaviour in the world, using the wolf as an example, will allow researches to build socially intelligent entities.

In robotics, concepts from dominance hierarchies have been adopted by Maja Mataric and by some of her students. Next, some of this work is reviewed as it is of great interest for this thesis.

[Mataric 1995] uses a set of basic behaviours inspired by animal behaviours (such as follow an member of the team, move and avoid other team members...) to achieve complex behaviours in a group of physical mobile robots. She tested two behaviours, dispersion, move away from any robot, and aggregation, the opposite behaviour. She analyses the numbers of steps necessary to reach the final state in which all robots are together or are at a given distance from each other. Two different methods were applied to reach the final state. The first method consisted of allowing all robots to move independently, and the second method consisted of establishing a dominance hierarchy where dominant agents moved while others waited for their turn. In the case of aggregation results show an improvement with respect to the first method, although this difference is not statistically significant. For the second behaviour there is no difference between the two approaches.

Later [Goldberg and Mataric 1997] apply dominance hierarchies to a clean-up task for conflict resolution. Several robots wander around an enclosed area where their task is to find several pucks (small metal cylinders) to pick them up using a gripper and carrying them to a drop-off area. These robots are programmed with several identical behaviours (avoiding, wandering...). Several strategies are compared for dealing with conflict resolution. Experimental results show that establishing a dominance hierarchy takes more time to complete the task of collecting all pucks but results in fewer conflicts.

In a later work by Mataric and some of her students [Vaughan et al 2000] the same approach is applied to a task of robot transportation. One of the problems of the environment is the lack of space in the corridors, only one robot can pass at the same time through the doorway. Several strategies, among them a hierarchical dominance one, are analysed in case of conflict between two robots trying to pass through the doorway. In the case of a dominance hierarchy, a position in the hierarchy is assigned to all robots beforehand. Other strategies are random, conflicts are solved randomly, and personal space. Results show that the dominance hierarchy does not offer an advantage in the experiments. They argue that dominance hierarchies in nature reflect the fitness of the competing individuals, and should be used in heterogeneous systems with non-uniform abilities. They claim also that the process of the formation of a dominance hierarchy on heterogeneous robots should be a progressive one, where the probability of winning/loosing depends on the past competitions. This process would lead to the emergence of a dominance hierarchy.

Results of these works seem to show that stabilising a dominance hierarchy among a group of mobile robots to perform different tasks has no remarkable benefit on the performance of the whole group. One of the problems seen in these works is the methodology used to create the dominance hierarchy. The hierarchy is defined arbitrarily and does not allow any kind of flexibility among the members of the environment. In the latest work they, claim that dominance hierarchies should be applied to heterogeneous robots and the creation of the hierarchy should be as a result of the interaction of all members. However, no further work has been found attempting to develop these ideas for conflict resolution.

2.3 Soccer Domain

Soccer has become one of the most important test-beds for MAS research over the past decade. [Barman et al 1993] developed Dynamo, the first robot soccer platform used as a test-bed to experiment on physical MAS agents in 1993. In 1996 the first FIRA competition took place in Korea and in 1997, RoboCup in Japan. The University of Girona participated at the first FIRA competition in 1996 and the first RoboCup competition in Japan. Since 1999 the robotic team of the University of Girona has participated in all RoboCup symposia that have been held every year, including the soccer competition.

According to [Kitano et al 1997], RoboCup is an attempt by using soccer game as a representative domain where wide-range of technologies can be integrated, as well as new technologies can be developed. The RoboCup envisions a set of longer-range challenges over the next few decades, each league have its own long-term goals. The most ambitious one, it is the development of a team of humanoids by 2050 to play a game against a human soccer team and win the game. A possible list of research areas in RoboCup is:

- Agent architecture in general
- Combining reactive approach and modelling/planning approach
- Real-time recognition, planning and reasoning
- Reasoning and action in dynamics environment
- Sensor fusion
- Multi-agent systems in general
- Behaviour learning for complex tasks
- Cognitive modelling in general

Research on MAS in the soccer domain can be conducted either on real robots or on simulators. On the one hand real robots allow to work on MAS on real systems, coping with the limitations of real systems and problems associated to them. On the other hand, the type of algorithms, strategies... that can be tested must be simple, as malfunctions, communication problems, physical limitations... affect real robots. Simulators overcome the limitations of real robots allowing researchers to develop and test more complex algorithms strategies. The most widely used simulator is SoccerServer [Noda 1998]. This simulator allows researchers to work with teams of up to 11 players with bounded computational resources, noisy perception of the world, communications and stamina. This simulator is the official simulator for the Soccer Simulation League in RoboCup. Another simulator built on Robocup is JavaSoccer [Balch 97], which is based on the Small Size League of Robocup, where each team is made up of up to 5 robots. JavaSoccer is implemented on Java, this means that it may be run on several platforms. This simulator is open source and allows to modify its code to tailor it to everyone’s needs.

One of the problems in the Soccer Domain is the evaluation of the skills of an individual player or a team. This is a difficult task, as the performance of each team may change depending on many different aspects, specially the opponent. One possible way of evaluating these teams objectively is by defining a set of benchmarks, where several skills of these teams can be evaluated. [Johnson, de la Rosa and Kim 1998a] and [Johnson, de la Rosa and Kim 1998b] propose the realisation of several benchmarks to analyse the performance of several solutions implemented. These benchmarks have different purposes:

- To set rigorous scientific standard for research into robot football.
- To encourage teams to work on the same problems and to allow comparison without any explicit match of teams each other.
- To collect and publish data on robot control and ball control, and co-ordination.
- To enable scientific analysis of the performance of teams worldwide.
- To enable any particular team to gauge its performance against these standards.

- To provide a *simple* baseline from which new scientific benchmarks can be defined.
- To create proper families of benchmarks useful for adaptive/emerging techniques of AI.
- Testing individual or collective behaviour.
- Testing with real or simulated robots in a real or simulated environment.
- Generated benchmarks have to be easy to build.

Based on these ideas a new benchmark has been developed for this thesis (see chapter 4 and 6, for more details), called Implicit Opponent [de la Rosa, Duhaut and Muñoz 2000], which consists of an inclined field and a group of static opponents.

2.4 Heterogeneity

2.4.1 Introduction

Heterogeneous MAS has drawn the attention of researchers over the past years and it is becoming an important research subdomain. This section reviews some of the properties and aspects of systems made up of heterogeneous components, as well as some examples of heterogeneous robotic systems.

According to Merrian Webster the term **heterogeneous** means:

"Consisting of dissimilar elements or parts; not homogeneous"

and the definition for **homogeneous**

"Of the same or similar nature or kind. Uniform in structure or composition throughout"

[Good 2000] argues that heterogeneity can be a better approach to solve problems by heterogeneous MAS for different reasons:

- Biological systems are rarely truly homogeneous. Components of MAS built on natural principles are designed as homogenous systems; this approach may be missing some of the properties of these systems. (see section on Swarm Intelligence for some examples that corroborate this statement).

- Heterogeneous systems can allow task specialisation allowing much simpler components.

In Multi-Agent Systems, heterogeneity can be analysed from different points of view. [Stone and Veloso 2000] analyse these systems from a machine learning perspective. From this perspective agents can differ in:

- Goals
- Actions
- Domain Knowledge

[Lander 1994] classifies agent heterogeneity in the context of team problem solving:

Representation heterogeneity in

- Architecture
- Algorithm

- Language
- Inference engine
- Hardware requirements

Knowledge heterogeneity in

- Declarative knowledge
- Evaluation criteria for solutions
- Priorities
- Goals
- Capabilities

Working with heterogeneous agents whatever the domain is, becomes a much more complex task, as designer has to cope with additional problems as a result of dealing with heterogeneous components. As components are heterogeneous the designer must design, test... each agent individually. All agents must be tested together and tuned, and all (at least several) possible combinations of these agents must be tried out, in order to get the best of the whole system. However, according to [Good 2000] the components may be much simpler than in homogeneous systems.

2.4.2 Properties

Heterogeneous systems share some common properties, in this subsection some literature on this issue is reviewed. These properties can not be extended to all implementations involving heterogeneity, but it has been found that the following properties are common to many applications.

2.4.3 Stability

There are a few examples in the literature that show how systems composed of heterogeneous agents can help a MAS to reach a stable configuration. The concept of stability depends on the problem, but usually what is stabilised is the resource usage or some component that is used by the members of the system. Stability of the system is tightly related to the global performance of the system, as it is a condition for a system to work properly. This can be seen in the example of the stock market. Unstable stock markets (volatility, bubbles, crashed...) usually affect negatively to the economy by introducing a greater uncertainty in the national and world economy, as history shows.

Perhaps a very clear example already reviewed in a previous section is [Hogg and Hubermann 1991], where agent heterogeneity, coupled with a reward mechanism, is responsible for the stabilisation of the resource usage of the system. As a result of the stabilisation the global performance of the whole system is improved. In this case diversity is created by using agents with different perceptions of the state of the environment (different time delays to access information on the state of the environment and different individual bias for each kind of resource). [Thomas and Sycara 1997] use the same model and find that distorting the decision making process of each agent, by adding different noise to each agent’s perception of the environment, also stabilises the system.

In [Arthur 1994] a set of people (agents) try to predict the attendance of a bar using the information available on the attendance of the previous weeks. Each agent applies different predictors to this task. Using the information on past attendees each agents decides to go to the bar or stay at home using those predictors, choosing the one that have performed better in the past, among the predictors available for each agent. Heterogeneity in this case helps stabilising the attendance after some sharp oscillations at the beginning. Results show that the average attendance tends to converge to the optimal value through a self-organising process

A similar case can be found in economic systems. [Steiglitz, Honig and Cohen 1995] simulate a large collection of agents that sell and buy resources (as food) produced by themselves. The price of these resources oscillates around a value when only one type of agents is considered. When a new type of agents, called speculators, are introduced into the system, the prices stabilise showing very small oscillations

2.4.4 Performance

Performance is perhaps the most interesting property in heterogeneous MAS. If such systems are able to complete tasks in a more efficient way and the problem of designing, testing and tuning a heterogeneous MAS does not implies considerably additional costs, these systems may prove more interesting for carrying out tasks.

The following examples show that for many applications the impact of using heterogeneous components results in an improvement of the performance of the whole system, even though this is not true for every application. From the point of view of physical heterogeneity only one work has been found that tries to study in depth the impact of it. This last point shows that there are still many things to do to study the possible benefits of physical heterogeneity.

According to [Parunak 1997a] heterogeneous agents should perform better than homogeneous ones, as they cover more of the environment's state. Environment's state contains information on opportunities that should be exploited, and threats that should be avoided. The larger state space covered the better the system can exploit the opportunities/avoid the threats. From the same perspective, both researches have worked together, [Brueckner 2000] claims that heterogeneity enables the whole system to try out different approaches in solving the same system, thus resulting in a possible better performance. One of the problems of trying out different approaches is that new solutions can result in a lower performance of the whole system, as the number of combination increases. In addition, heterogeneous agents are more adaptive than homogenous ones, if an effective use of heterogeneity is made; these systems are able to maintain behavioural patters to accomplish its tasks in a changing environment.

[Hong and Page 1998a] and [Hong and Page 1998b] study the use of diverse problem solvers to solve a multidimensional problem. These problem solvers differ in their heuristics and the internal representation of the problem. Initially these strategies are tested independently, and later they are tested working together on the same problem. Results show that using a set of the best individual problem solvers, the performance is lower than using a group of random problem solvers. The reason for this surprising result is that the best problem solver approach the solution in a similar way, and random problem solvers benefit from having different strategies.

In a different kind of domain [Balch 1998] studies the impact of robotic behavioural diversity in robotic soccer, co-operative moving and foraging. He uses reinforcement learning for behavioural configuration. Results show that for the first two cases behavioural diversity affects positively the performance of the system, while for the third one, diversity is correlated negatively with performance. [Bongard 2000] also comes to the same result for the foraging problem, using a different method for learning behaviours.

[Potter, Meeden and Schultz 2001] co-evolve high level controllers, using an evolutionary algorithm, for a groups of robots in the domain of herding, where one or more robots must herd another robot into a confined space. They argue that the need for heterogeneity depends on the number of skills required by the domain. Results show that in easy tasks heterogeneous and homogenous behaviours result in similar results, but when complexity is increased in this domain by adding a predator, heterogeneous behaviours perform much better than homogeneous ones.

[Kurabayashi et al 1999] study the performance of a group of robots, in simulation, using different algorithms. The goal of these robots is to visit several target points; these target points can contain garbage, objects... Each robot knows the location of each point beforehand; a point is excluded from the points to visit once it has been visited. There are two classes of algorithms, reactive and planning. Reactive robots try always to get to the nearest point given its present position. Planning robots determine the order to visit the points using an algorithm, if one of the points has been already visited it does planning again. Results show that a mixture of reactive and planning robots perform better than only planning ones.

[Parker 1994] analyses the performance of several groups of robots, ranging from homogeneous to heterogeneous. Results show that performance diminishes as heterogeneity increases among the robots. Although all robots are heterogeneous, as each class of robot can only perform a subset of total number of tasks, heterogeneity is also changed by increasing the time difference to complete a given tasks among all robots with the capability to perform the same task.

2.4.5 Diversity Metrics

In order to study rigorously the impact of heterogeneity, it is important to have a metric to evaluate the heterogeneity of a group of agents. Several researchers have tried to develop a measure of diversity to study groups composed of individuals. This measure is very useful for studying several fields, ecology, sociology... and in the past years robotics.

[Shanon 1949] was the first scientist to develop a measure of diversity, when he sought to evaluate the uncertainty of an information source. This measure is known as the information entropy. Diversity depends on the number of different classes of elements that made up the system, and the number of elements in each group. [Balch 2000] extends this work to measure robot group behavioural diversity. One of the contributions of this work is that it takes into account differences among the different classes that constitute the analysed society. He calls this new methodology Hierarchic Social Entropy. The diversity metric of Balch and Shanon are analysed in depth in chapter 6, as they are used in this work.

[Bongard 2000] defines a diversity measure in the field of Genetic Programming for agent groups with evolved behaviours through evolutionary algorithms. This measure is based on a fitness function that is normally used in this type of algorithms to evaluate the facility of the group to accomplish the assigned task. Different values of fitness between two groups imply behavioural differences between them. In the evolutionary context, is easier to evaluate behavioural difference at the level of the fitness of each group.

[Parker 1994] defines heterogeneity for a group of heterogeneous robots that are able to carry out a different subset of tasks, where the time a robot needs to complete a given tasks depends on the type of robot. Heterogeneity is measured as the maximum difference in time, in percentage terms, that any two robots, with the ability to perform the same task, need to complete their task.

Although the metric developed by [Parker 1994] is the only that deals with physical heterogeneity, from the point of view of this work the one that seems more rigorous and includes some of the aspects of heterogeneity, is the one developed by [Balch 2000]. Although this metric only concerns behavioural diversity it can be also adapted to measure physical diversity, as it is explained later.

2.5 Heterogeneous Multi-Robot Systems

2.5.1 Introduction

Multi-robot systems have become a very active topic over the past decade. There are several advantages in using several agents for solving a task instead of one single agent. [Ali 1999] summarises some reasons for using a multi-robot system:

- A larger range of possible tasks
- Greater efficiency
- Robustness
- Lower economic cost
- Ease of development

A central idea is the ability of a multi-robot system to perform any task that a single robot can do, but a single robot can not accomplish tasks that a multi-robot system can do.

[Cao, Fukunaga and Kahng 1997] compare robotic homogeneity and heterogeneity as

" A group of robots to be homogeneous if the capabilities of the individual robots are identical, and heterogeneous otherwise".

Heterogeneity in multi-robot systems can be achieved by two different means, either designing identical robots but with different behaviours or designing robots with different capabilities. Usually robots with different capabilities also present different behaviours. Robots with different capabilities can also be analysed at two different levels. [Parker 1998b] robots may differ in which tasks they are able to accomplish, and second in how well they perform the same task. This thesis centers on the latter case.

Physical heterogeneity adds complexity, since task allocation becomes more difficult. Task allocation will depend on each robot capabilities, [Simmons et al 2000]. For many tasks the use of heterogeneous robots is necessary because of the difficulties of constructing a single robots with the necessary size, strength... For [Parker 1994] the range of capabilities needed to perform a complex task by a group of robots can be partitioned across several more simple robots, instead of having a group of robots with identical, but complex capabilities.

This section focuses on heterogeneous multi-robot systems with different capabilities. Some works that include behavioural diversity, which are of interest in this work, have been already described in previous sections.

2.5.2 Co-ordination Approaches

Several co-ordination approaches have been developed over the past years to exploit the full potential of heterogeneous robot teams. In this subsection some of these approaches are reviewed.

The first serious work in this field was done by [Parker 1994], who developed an architecture for heterogeneous robots. This architecture is called ALLIANCE and is behaviour-based. It is designed for the co-ordination of several heterogeneous robots with overlapping capabilities, fault tolerant, adaptive action selection for small-medium sized robotic teams. Robots select actions depending on several factors, requirements of the mission, the activities of other robots, the current environmental conditions, and their internal states. All these factors are combined by means of a complex mathematical mental model. Later she added a learning capability to this architecture, creating L-ALLIANCE, its specific features are detailed in the next section.

The use of economic concepts for the co-ordination of heterogeneous multi-robot systems is becoming a very common approach to co-ordinate such systems. Results so far show that this approach works quite satisfactorily and its implementation is much simpler than that proposed by ALLIANCE. This approach usually consists in holding auctions among the members of the team, in order to assign incoming tasks to idle robots. These auctions are usually based on utility functions that describes the ability of the robot to complete a given task. One of the drawbacks of such approaches is that introduces several centralised processes into the system, as all robots must be notified of the auction and one of the members of the team must act as auctioneer. Another drawback of these approaches is that members' abilities may change over time, to solve this problem a learning mechanism should be developed to continuously learn these abilities, otherwise the mechanism might result extremely inefficient. Next, some applications of economic approaches are presented

[Dias and Stentz 2000] propose an economic method for the co-ordination of a group of robots, specially designed to deal with the co-ordination of heterogeneous groups. This approach is based on economic concepts, in particular, on free market economies. In free market economies individuals act selfishly, for their own interest, but the aggregated effect of this behaviour is a highly productive society. This approach can be applied to groups of homogeneous and heterogeneous agents, but it is specially devised for heterogeneous agents. The overall goal is to execute a plan such that the profit is maximised, by reducing cost (co-operating with other robots...) and maximising revenues (time to complete a given task...). Robots compete for performing tasks depending on the calculated cost of carrying out a given task and the expected profits. Once a robot has been assigned to a task, it can negotiate with other robots to perform these tasks in co-operation. This methodology allows to take into account each robot's capabilities in the process of task assignment, as capabilities are included in the estimation of costs and revenues. [Thayer et al 2000] apply this approach to the task of mapping. Results show that the cost of visiting several predetermined locations is minimised through the process of bidding (competing for tasks) and negotiating among the robots.

A similar method is developed by [Gerkey 1998], [Gerkey and Mataric 2000] and [Gerkey and Mataric 2001], called MURDOCH. Given a set of resources and a sequence of tasks to be completed, the goal is to allocate tasks to a heterogeneous group of robots, each one having different skills, in a distributed manner and efficient manner, trying to minimise resource usage, task completion time and communication. The process of task allocation starts when a task is introduced into the system. One agent acts as an auctioneer by publishing an announcement message for the task. This message is addressed to all robots capable of performing this task, as this task details the resources needed for its accomplishment. As there may be several robots available capable of this task, each robot evaluates its fitness for this tasks using an indicated metrics, which combine several aspects (position of the robot...). The robots with the highest fitness are awarded the task with a time limit, this allows to reassign the tasks if one robot malfunctions; but a different robot has to supervise the task. Sometimes a task may demand more than one robot. Results, in a task of box pushing among heterogeneous robots, show that this system is extremely reactive to changes in the environment and failures of robots.

Tightly co-operating architectures have also proved useful for the co-ordination of complex tasks by heterogeneous robotic systems.

[Simmons et al 2000c] use three heterogeneous robots for the assembly of large-scale structures, in particular, the assembling of steel structure of a large building. In this problem the motivation for using heterogeneous robots is that all the capabilities that are needed can not be constructed in one robot. This system is made up of three robots that combine heavy lifting capability, precise manipulations, and the estimation of the position of the different parts of the whole system. This is achieved by using three different robots (an overhead crane, a mobile manipulator and a roving eye), each one performing one of the tasks, and co-operating through explicit co-ordination using a layered architecture. Each robot can interact with the other robots through direct connections at each of the three layers that compose the architecture.

[Chaimowicz et al 2001] proposes another architecture for tightly coupled multi-robot co-operation. This architecture is tested on three heterogeneous robots that work together to carry a large object in an environment with objects. These robots have different sensing capabilities, driving mechanisms and operating systems. Each robot makes use of communication and simple control algorithms to solve complex tasks. Its key feature is the dynamical role assignment, one of the robots fills the robot of leader, usually the robot that is best suited in terms of sensor power, manipulation capabilities... while the other robots fill the role of followers. Robots can also exchange roles if the follower can not follow the leader or if it knows a better way to perform the task. This allows robots to adapt to unexpected events, obstacles, malfunctions... The leader plans a trajectory, which is broadcasted to the followers, so that the followers have to control their trajectory based on this plan and feedback from their position, velocity and sensors... Results demonstrate the usefulness of this architecture, its flexibility and its performance in several scenarios.

2.5.3 Heterogeneous Multi-Robot Applications

One of the tasks that has drawn the interest of researchers working on heterogeneous robot systems is map making and exploration.

Millibots [Grabowski et al 2000] is an example of the benefits of having teams of heterogeneous robots. Millibots are small robots that collaborate to map and explore unknown environments. These robots are heterogeneous and have different sizes and sensors. Although each individual robot can have limited capabilities they can accomplish the same tasks through co-operation and are more reliable. In case one robot fails, few capabilities are lost and the robots can continue the task with the remaining robots, which normally have overlapping capacities. This team is made up of middle size robot, which have a range of over 100 miles and are able to transport a number of smaller robots. Middle size robots can serve as the leader and co-ordinator of the small robots. Small robots can access areas not accessible to larger robots. Tasks among the small robots are achieved through specialisation and collaboration. Specialisation means that each robot is equipped for a particular aspect of the task, instead of having robots with every sensor, computation, and communication capability. Collaboration allows the robots to overcome their limited capabilities.

A similar project that combines small size robots and large robots for exploration purposes has been developed by the University of Minnesota [Rybski et al 2000]. This system is used for missions of Reconnaissance and Surveillance for military and civil organisations. Two kinds of robots made up the system: rangers are the largest robots and are able to transport the small ones (called scouts) over distances of several kilometres and deploy them. Small robots are provided with sensors and limited computational capabilities. Rangers process data from scout robots, and act as co-ordinators of the whole team.

[Singh and Fujimura 1993] approaches the problem of map making using co-operating heterogeneous robots, which vary in size and speed. Robots co-operate in order to develop an efficient solution Robots with greater speeds and sensor ranges traverse the region as far as possible and robots with small sizes are used to explore areas where obstacle impose size constraints on big sizes.

Also [Simmons al 2000a] deploy a set of heterogeneous robots for tasks of exploration and mapping. Robots explore the environment in a co-ordinated way, avoiding interference between them. This is accomplished by means of a technique that tries to maximise information gain (map knowledge). Each robot constructs a “bid” consisting of the utility of moving to a new location to explore an unknown territory (this can be seen also as a economic approach to the problem); this utility includes information on the optimal path to reach the new location and the sensor range of each robot, thus including the different capabilities of each robot. In a later work [Simmons et al 2000b] use the maps created in the exploration and mapping stages to deploy the same robots in a co-ordinated way. These robots also are assigned to different locations based on their different capabilities, trying always to minimise the cost of performing the tasks given the constraints of the task.

The architecture ALLIANCE has extensively been applied to several real applications, as multi target observation [Parker 1999], waste cleanup, box pushing [Parker 1998b]...

2.5.4 Heterogeneous Teams in RoboCup

Robocup has seen in the last years an increasing participation of heterogeneous robotic teams. In the small size league and middle size league, many teams have been playing over the past years with teams made up of heterogeneous robots. Most of these teams play with four (three for the middle size) identical robots and use a different player to act as a goalie, for example RogiTeam, the team of the University of Girona, normally plays with a special robot playing as a goalie. In Robocup 2001 FU-Figthers participated with a team with several types of players. They used two different robots to fill the roles of goalie and defender, they both had and elongated shape, while the roles of midfielders and attacker were filled by omnidirectional robots and fast two-wheels robots, every one equipped with a kicking device. In the same Robocup at the simulation league teams started to play with heterogeneous players, players differ in speed, stamina...

[Castelpietra et al 2000] have participated in last years in Robocup in the middle-size league with a team of heterogeneous robots, each robot designed by a different Italian university. Robots differ in many aspects, mechanics, sensors, computer hardware, software, and design. All this affects each robot’s capabilities to play soccer. These robots communicate and co-ordinate with each other through the software architecture called ETHNOS. Messages are exchanged among the members of the team in a similar way to MURDOCH. Co-ordination among the robots starts by selecting a formation depending on the environmental conditions. Next, robots have to agree on how they will fill the roles of this formation. This is achieved by communicating the utility values of each robot with respect to each role. These values are calculated by several utility functions, specific for each role, that each robot evaluates given the information on the environment and each robot’s capabilities. A strategy assigns roles to robots by using the utility values. Roles are reassigned and utility values are recalculated 10 times per second.

2.6 Learning in Multi-Agent Systems

2.6.1 Introduction

Learning in MAS has drawn a lot of attention over the past years, and it is today one of the most important topics in MAS. This section will focus on three different learning problems in MAS, Learning in heterogeneous MAS, learning in soccer and co-learning.

Today learning is applied to many different kinds of problems. Perhaps one of the most interesting surveys on this issue appeared over the past years is [Stone and Veloso 2000]. This section only will focus on the issues that are strongly related to this work.

2.6.2 Learning in General

Over the past half-century many learning techniques have been developed. These techniques are mainly classified as supervised and unsupervised learning. The first class of techniques has in common that the learning process is achieved by learning through pairs of inputs and outputs. The system tries to learn the mapping from inputs to outputs. Perhaps the best known technique in this group is Neural Networks. The second class of techniques learns through the inputs (rewards) they get from the environment. Using the information on the inputs, agents update their internal states always trying to take the actions that maximise/minimise the inputs they get from the environment. Reinforcement Learning (RL) is an example of this category.

Basically all the examples presented in this section learn through Reinforcement Learning techniques or Genetic Programming, for this reason a short overview of each technique will be given.

RL allows an agent to learn which action to take, given the present state of the system by trial-and-error interactions with a dynamic environment. RL makes use of statistical techniques and dynamical programming to estimate the utility of taking an action, from the available set, for the different states of the world. The basic model of RL works as follows, on each step the agent receive an input from the environment containing information on the state of the system. Based on this information the agent chooses an action. As a result of this action the state of the environment changes and the values of this state transition is communicated to the agent through a scalar, known as the reinforcement signal. The agent should learn to choose actions that tend to increase the sum of reinforcement signals. There are several RL algorithms that try to maximise the reinforcement signal, depending on the kind of information available, problem, number of states... For more information there are several tutorial and surveys available on Internet, perhaps one the most complete surveys is [Kaelbling, Littman and Moore 1996].

Genetic programming (GP) is a very popular technique, which is based on genetic algorithms (GA) [Goldberg 1989]. GP uses evolution to optimise computer programs or algorithms to solve a given problem. Initially a set of possible solutions is chosen (randomly) to solve the problem. For each possible solution a fitness value is calculated, which is a measure of how well each evolved algorithm is solving the problem. The best solutions are combined to find new ones by means of two different operators, crossover, several solutions are combined to form new solutions, and mutation, some elements of the solution are randomly changed. At any given time the solution is the program value with the highest value of fitness. (GA) is a much simpler technique than GP but follows the same process to find a numerical solution to a problem. (More information http://www.geneticprogramming.com/Tutorial/) These programs quite often encode behaviours for the agents that made up the system.

Aspects from these two learning techniques are combined together in this thesis to develop an algorithm to learn in heterogeneous multi-agent systems. How they are combined, is detailed in the next chapter.

2.6.3 Learning in Soccer Domain

Soccer has become one of most interesting test-beds for testing and developing new learning strategies. Most of the research has been carried out on the simulation domain, even though some applications with physical robots have also been done. Learning in soccer can be found at two different levels, learning individual skills (as kicking, passing, dribbling...) and learning team strategy. Most of the learning applications in this domain have been done using Genetic Programming or Reinforcement Learning.

Some examples show that evolutionary techniques can be used to learn low-level skills, for example [Akiyama 2000] uses GP to learn to dribble and kick the ball, and [Matsumura 1998] to kick the ball.

RL have been also used to learn basic behaviours on real robots, [Nakaruma 1997] applies Q-learning, a type of RL, to learn to kick the ball and keep the goal on real robots. It can be also used to learn very basic co-operative tasks in simulation, as to choose between two actions, pass the ball to a team-mate or to kick the ball in the presence of an opponent near the goal [Kostiados and Hu 1999].

Some teams have started by learning basic skills and building on these learned basic skills, they have learnt the team strategy. Perhaps the best known example in the soccer community is the work done by [Stone 1998]. A basic skill as intercept the ball is learned using Neural Networks, once a player knows how to intercept the ball, then learns to evaluate if a pass will be successful using decision trees. Finally the whole teams learns to decide to which player pass the ball using Reinforcement Learning, in order to maximise goal difference.

In GP there are two examples, [Luke et al 1997] and [Andre and Teller 1998], that managed to learn individual skills and team strategy using this technique.

[Luke et al 1997] use Genetic programming to evolve behaviours for each player, in the RoboCup Server domain. These behaviours are evolved using a set of atomic functions; some of them evolved previously using Genetic Programming. Initially they make behaviours evolve that are used by all players. Later they build a pseudo-heterogeneous team, instead of evolving a behaviour for each player; they evolved six different behaviours, each one containing one or two robots. Results in competition showed that this technique worked quite well, even though the team that participated in RoboCup 1997, was a homogeneous one.

[Lubbers and Spaans 1998] use GA in a different way, they try to learn confidence values to decide which action to choose given the world model perceived by each agent.

[Salustowicz, Wiering and Schmidhuber 1998] apply to learn how to behave different types of techniques. They compare evolutionary learning techniques with RL techniques. Results show that evolutionary techniques learn better and faster than the RL technique. In this case all team members share the same policy.

2.6.4 Learning in Heterogeneous MAS

Learning in heterogeneous MAS is still an open problem. This problem present some additional features with respect to learning in MAS as the state space is larger. This section will focus only on algorithms that learn in systems made up of heterogeneous components or that heterogeneity has been evolved as the result of a learning process. For the former case, there are very few examples, for the later one there are a few more.

[Lander 1994] presents a knowledge-based technology, called TEAM, in which a group of heterogeneous agents, each with an area of expertise, tries to solve a given problem with a set of specifications. Each of these agents works on a part of the problem, the overall solution is made up of solutions to subproblems. Each agent plays an organisational role in this search for a solution. An agent can choose to initiate a new design, extend a partial design or critique an existing plan. This process involves a framework where negotiations can take place among these agents. All this procedure is applied to a steam condenser, as an example, in which a set of agents (heat exchanger, pump, motor, V-belt, Shaft, Platform) has to act in order to accomplish a set of specifications. Each agent has a set of capabilities and has to propose solutions to the specifications dealing at the same time with the constraints. This work is extended later, [Nagendra, Lesser and Lander 1995] and [Nagendra, Lesser and Lander 1996], in a new framework called L-TEAM. L-TEAM allows the robots to learn their organisational roles from experience. After the learning process each agent knows what to do, which role to play, according to the problem to be solved. A simple RL technique is applied to learn in this problem. This learning process shows an improvement in results in comparison to results in TEAM framework.

[Parker 1994] L-ALLIANCE is an extended version of ALLIANCE that preserves the features of ALLIANCE and adds an on-line mechanism that allows a robot to learn about the time itself and other team members need to complete a given task. Each robot observes its team-mate when it performs a given task, using the data collected from several observations, each robot tries to figure out an estimation of the time each robot needs to complete a given task. This learned knowledge is used later to determine which robot carries out a given action, always trying to minimise the time that is needed to complete the task assigned to the whole robot team. She studies several possible strategies to include this learned knowledge in the process of decision making. Experiments show an increase in team performance when compared with ALLIANCE.

Some works prove that heterogeneity can be the a result of an evolution process:

[Bongard 2000] develops a new genetic programming technique, known as the Legion System. This technique apart from evolving behaviours also evolves heterogeneity depending on how the task is collectively performed. He illustrates this technique in two different tasks that ends up in homogenous and heterogeneous populations.

[Potter, Meeden, Schultz 2001] uses an evolutionary algorithm to evolve neural high-level controllers for a group of robots in the domain of herding. This algorithm is able to evolve heterogeneous controllers successfully, but heterogeneity is imposed from the very beginning of the process of learning by defining a different neural network for one of team members, or changing the initial position of one robot.

In soccer there are also some examples that show that behavioural heterogeneity emerges from learning.

[André and Teller 1998] use a similar approach to [Luke et al 1997] to evolve the team's individual skills and agent behaviour for the simulation competition. Unlike this previous work the evolved team presents different behaviours for each player.

One of features of evolving heterogeneous behaviours with these techniques is that the time needed for this task increases dramatically.

[Balch 1998] uses RL to learn robot strategies from a set of common skills, known as motor schemas. Results show that the player strategies depend on the type of learning selected. Global learning, all agents are rewarded/punished when a goal is scored/received, results in heterogeneous strategies, and performing better than homogenous strategies, learned through local reinforcement. Agents are rewarded depending on its position when a goal was scored/received.

2.6.5 Learning and Co-learning

Co-learning is a subfield of learning that has attracted a lot of attention over the past few years. [Shoham and Tennenholtz 1994] defines the notion of co-learning

"A process in which several agents simultaneously try to adapt to one another's behaviour so as to produce desirable global system properties"

As each agent updates its behaviour over time and all agents are simultaneously learning, this process is called co-learning.

Today this problem is especially interesting for designing agents participating in auctions on the Internet. Agents act selfishly trying always to get the maximum profits out of the auctions. For this purpose, agents have to learn from other agents [Hu and Wellman 1998] and adapt continuously their strategies. But it is also attractive in many other fields and problems. [Wolpert and Tumer 2000] describes some possible fields of applications of co-learning, control of communication satellites, routing over a network, vehicular traffic control...

The existence of several agents, each simultaneously learning, poses a serious problem to the success of the learning process, as agents' decisions change the state of the environment, leading agents to learn in non-stationary environments [Claus and Boutilier 1997]. Most of the existing learning techniques have been developed to work in stationary environments, but they do not guarantee many of their properties under such circumstances. This issue has been analysed by several researches in the domain of competitive and collaborative games, [Shoham and Tennenholtz 1994], [Claus and Boutilier 1997] and [Wellman and Hu 1998]. They have studied the conditions for the convergence of the process of co-learning, and although these conditions have been found for simple cases, they generally conclude that these conditions do not hold for more complex games.

Some researches that have studied this problem conclude that the resulting behaviour of the whole collection of agents tends to result in highly non-linear systems and nontrivial results, even if the learning strategy is quite simple [Shoham and Tennenholtz 1994].

One of the problems of research in game theory is that they do not reflect the complexity of many real problems, where the relationships among the agents that may compose a system, are much more complex than those represented in those games. [Wolpert and Tumer 2000] develop some of the ideas existing in co-learning in what they call COIN, "Collective Learning". They define collective learning as a large multi-agent system where:

- There is little to no centralised, personalised communication and/or control
- There is a world utility that rates the possible histories of the full system.

In COIN each agent runs a reinforcement learning algorithm, aiming at optimising the world utility function. The idea behind these concepts is the development of an approach for designing MAS that does not need detailed hand-tuning of the agents and of the interactions of these agents, resulting in highly non-robust systems and with limited applicability. Instead of relying on the traditional approach, COIN aims at designing the problem implicitly, benefiting from the adaptive character of the Reinforcement Learning algorithms of each of these agents, leading at the same time to an optimisation of a global utility function.

Some reviewed literature on this issue shows that this approach has been applied successfully to some domains, but they are still quite simple.

2.6.6 Credit Assignment Problem

One of the problems of learning algorithms in MAS is the credit assignment problem. The problem of assigning credit or blame for overall performance changes to each member of the system activities that contributed to that changes [Weiss 1996]. This problem comprises two subproblems, the problem of finding which action caused the system to change its performance and the problem of what decisions led to choose this action. All these become much more complex when the members of the systems are completely heterogeneous.

In evolutionary algorithms there are two kind of problems, the "free-rider problem" and the "forgotten genius". In the first problem teams that are generally good may contain poorly performing components, these components can slow down or stop the process of evolution. In the second problem exceptional components may belong to bad performing teams, resulting in a lost of these component through the evolving process.

2.7 Physical Agents

Most research on robotics has considered the process of decision making in robotics, as a process that starts by perceiving the state of the environment through the sensors and reasoning from this state and from the internal states of the robots. This process leads to choose an action among the ones available for the robot. Very few researchers have paid attention to the importance of the dynamical and physical aspects of the robots, as a result of having a physical body.

According to [Asada et al 1997], physical bodies play an important role in agents to achieve goals in a dynamic real world, on which the traditional AI research has not paid so much attention. Physical bodies have an important role in enabling the system to interact with physical environments, that makes the system learn from the environment and develop its internal representation. The meanings of "having a physical body" can be summarised as follows:

- Sensing and acting capabilities are not separable, but tightly coupled.

- To accomplish the given tasks, the sensor and actuator spaces should be abstracted under resource bounded conditions (memory, processing power, controller, etc.).

- The abstraction depends on both the fundamental embodiments inside the agents and the experiences (interactions with their environments).

- The consequences of the abstraction are the agent-based subjective representation of the environment, and its evaluation can be done by the consequences of behaviours.

Perhaps one of the most interesting contributions to this topic, it is the work by [Zhang and Mackworth 1995]. They define a robot, although this can be extended to other systems, as a hybrid intelligent dynamical system, consisting of a controller coupled to its body. At the same time a robot is symmetrically coupled to a dynamic environment. To model the dynamics of they system and environment they developed Constraint Net (CN), a unitary framework to model dynamical systems. CN can deal with discrete and dynamical systems as well as synchronous and asynchronous event structures. CN is implemented as modules with I/O ports. Each module performs a transduction from its input traces to output traces, subject to the principle of causality: an output value at any time can depend only on the input values before, or at that time. A robot situated in an environment can be modelled as three machines; each of them modelled separately:

- Robot body
- Robot Controller
- Environment

Having a CN model and the required properties of a controller, specified as a set of constraints, it is feasible (if it is possible) to find in an automatic way a controller with the given specifications. [Zhang and Mackworth 1994]. In [Zhang and Mackworth 1998] these ideas are applied and implemented in a robot controller for RoboCup

[Stilwell and Bay 1993] develops a material transport system using robots with limited communication and sensorial capabilities. Its task is to transport a pallet to a predetermined location. In order to complete this task successfully they make use of the knowledge on the dynamical behaviour of the pallet and of each robot, to design a controller that allows them to transport this item measuring only, with a sensor, the forces applied in the horizontal plane. This strategy allows them to accomplish this task without communication.

[Fierro et al 2001] and [Chaimowicz et al 2001] achieve complex co-operation among a group of robots using information on controllers. [Fierro et al 2001] control a group of robots that follow a predetermined trajectory while maintaining a given formation at the same time. This complex behaviour is achieved by equipping each robot with range sensors and defining a set of controllers that are activated depending on the position of each robot and the destination point, this data is obtained through the sensors. The controller is designed, so that the stability of the whole system is guaranteed. [Chaimowicz et al 2001] achieve co-operation of heterogeneous robots using several controllers for a task of transporting a large object. In case that one of the robot is not able to keep up with the leader, in terms of speed, trajectory... , then the two robots switch roles in order to be both able to achieve the same speed, trajectory...

Also in [de la Rosa et al 1999] the problem of having a physical body is considered. Based on [Shoham 1993], capacities are used to represent the knowledge on the dynamical features of the robot. Capacities based on dynamics mean having information about controllers and how some variables (i.e. speed) change over time. All this information is used in negotiation processes among several agents (deliberative agents) to perform a joint task. Some examples show the convoying of two vehicles or passing a ball [de la Rosa et al 2000]. [Oller, de la Rosa, del Acebo 1999] presents an agent architecture, called DPA, to design agents with dynamics. This architecture has three different levels: control, supervision and agent. Dynamic knowledge has a different representation for each level; being adapted it for a better use for each different level.

2.8 State of the Art Analysis

The literature review shows a growing interest in the study and deployment of physical heterogeneous MAS. So far the interest of researchers in this area has focused on systems made up of identical components but programmed with different behaviours. In the past only a few researchers has shown interesting in building teams with heterogeneous physical components, the most important contribution to this domain was done by [Parker 1994]. In recent years some researchers has centered on the problem of building such teams, but they have paid little attention to the problem of co-ordinating such systems; mainly the development of algorithms for the co-ordination of such systems so that they can benefit from heterogeneity. We think that in this latest area, there is still a lot of work to do, so that the full potential of using heterogeneous systems can be exploited.

One possible way of exploiting the full potential of these systems is by learning how a given task can be performed efficiently by a group of heterogeneous robots, by partitioning the tasks across these robots. This problem has been studied in teams of homogenous robots; but this problem becomes much difficult when the members of team are heterogeneous, as the state space increases considerably. The literature review shows that very few researchers have worked on the problem of learning in heterogeneous MAS. In the field of robotics the only work that tries to tackle the problem of learning in MAS is [Parker 1994] with her behaviour-based architecture L-ALLIANCE. Although results show an improvement in performance with respect to a similar architecture without the learning capability, the way this problem is approached is quite simple, it is only based on the time a given robot needs to complete each task, and does not take into account several aspects that may affect considerably the performance of the whole system:

- ***Global Performance.*** The strategy of assigning tasks depending on the time necessary to complete them many not result in the best global performance, as one or more members of the team may contribute better to the global solution by completing other tasks, for which they are not so well fitted.

- ***Evaluation Metric.*** Time is not the only metric that can be considered for the problem of selecting a task. There are other aspects that can be included, energy consume, quality of the result, efficiency…

- ***Problem Complexity.*** The complexity of this problem can be considered as a co-learning problem. In order to find which is the best task distribution that maximises team performance, the process requires continuous interaction with the environment and other team members, so that the different solutions can be tried out.

One additional question related to this problem, it is to know whether a group of heterogeneous robots may increase the performance of the whole system, with respect to a group of homogeneous robots. To this question there are very few answers. From the literature analysed in this chapter the only answer is that physical heterogeneity is beneficial when all the capabilities needed for completing a given task can not be integrated in one robot. This is the case for [Simmons et al 2000c], which use three heterogeneous robots for the assembly of large-scale structures. For the case of having a set of heterogeneous robots that may complete all the tasks or a subset of them with different efficiency, it is not clear which is the answer. According to [Parker 1994] the time needed to complete a task increases as the heterogeneity among their members increase.

With respect to the analysis of the related work, this thesis contributes to the state of the art mainly in three different aspects:

- Development of a learning algorithm for heterogeneous MAS. This algorithm allows agents to learn which are the best tasks to complete based on each agent skills, other agent's interests and the global team performance. This algorithm is built on a previous work by [Hogg and Hubermann 1991] and combines concepts from Reinforcement Learning and Evolutionary Algorithms. This learning technique builds on the concepts of ecosystems, in which a set of different species compete for bounded resources and the ability to self-organise of insect and animal societies. In this approach, species can be assimilated to different types of robots that compete to complete different tasks. This algorithm also keeps the adaptability present in nature, which enables agents to adapt to changes in the environment and/or in its own bodies. Tasks in the test-bed selected for this research are assimilated to roles in a team of robots.

- Analysis of the benefits of using teams of heterogeneous robot to accomplish a team task, as robotic soccer. One of the main interests of this domain for the analysis of heterogeneity analysis is the need to co-operate among the members of the team, in order to avoid conflicts. In this environment every agent can accomplish every task, but the efficiency to complete the task depends on the physical capabilities of each agent that change depending on the size speed and mechanical devices of each robot. Team performance is also evaluated from the point of view of physical diversity.

- A methodology to evaluate physical diversity in teams of heterogeneous robots is developed building on the work by [Balch 2000]. Several benchmarks are designed for this task, so that different types of agent can be evaluated under controlled environmental conditions. The results for every agent in each of the benchmarks are used to compute a measure of physical diversity by means of the methodology proposed by [Balch 2000].

In addition, the development of this new algorithm contributes also to the question raised in COIN, the design of MAS benefiting from the adaptive character of having several learning agents. Results will show that this learning algorithm complies with some of the design questions of COIN, achieving high robust and adaptable behaviour, and with little need of human hand tuning and intervention.

Chapter 3

Model Definition

3.1 Introduction

This chapter presents one of main contributions of this thesis, the development of an algorithm to learn in heterogeneous MAS. This algorithm is later applied to several teams of robots that learn how to partition and complete a global task efficiently. After the different selected teams have completed this process, the results of it are used to study how robot physical heterogeneity affects the different properties of such systems. These properties are compared across the different teams of heterogeneous and homogeneous.

This algorithm is built on a previous work by [Hogg and Hubermann 1991], which has already been outlined in the previous chapter. This chapter starts by reviewing this previous work and motivating the choice of this approach for the studying the problems proposed in this thesis. Next, all the different aspects of the algorithm and related to it are explained. Chapter 5 analyses later, some of the properties of the algorithm.

3.2 Motivation

In [Hogg and Hubermann 1991] a mechanism is presented to model the interactions of a large number of agents that compete for using bounded resource. This paper shows how this model works with simple but interesting examples.

This mechanism starts by defining resource choice for a group of agents. This is modelled by the following equation:

$$\frac{df_r}{dt} = a\left(r_r - f_r\right) \qquad (1)$$

f_r is the fraction of agents using resource r, in the examples used in this paper only two resources are considered. a is the rate at which agents re-evaluate their resource choice and r is the probability that an agent will prefer resource 1 to 2. f_r is also called the population (number) of agents of type r, as this works is based on ecological concepts.

In order to obtain the value of r, the difference of performance obtained by using resource 1 ($G_1(F(f_1))$) and resource 2 ($G_2(F(f_2))$) is computed. These performances may be obtained evaluating real actions, as done in this thesis, time required to complete the task... In this paper, the authors use an algebraic function to model performance based on the following idea:

The performance of a group of agents using the same resource increases (up to a certain value), as new agents start using the same resource, due to the emergence of co-operation among the agents using this resource. Once the number of agents using this resource reaches a given fraction, performance begins to decrease because of the increasing number of conflicts, due to the large amount of agents using the same resource. In this example these two concepts are modelled with these two equations:

$$
\begin{aligned}
G_1 &= 4 + 7 \cdot f_1 - 5.333 \cdot f_1^2 \\
G_2 &= 7 - 3 \cdot f_2 \qquad (2) \\
f_1 &= 1 - f_2
\end{aligned}
$$

Using these pay-offs, the expected performance for using each resource can be computed. The values of r for each resource are calculated using the equation (3). The greater the difference of the expected performances is, the greater the value of r is, for the resource that has the greatest value of pay-off. The uncertainty (σ) is modelled as a typical deviation on performance to decide when clearly resource 1 is going to perform better than resource 2.

$$r = \frac{1}{2} \cdot \left(1 + erf\left(\frac{G_1(F(f_1)) - G_2(F(f_2))}{2s} \right) \right) \qquad (3)$$

Using this model they conduct several experiments to study several properties of such systems.

The first experiment is conducted using only one kind of agents. When these agents work with perfect information, data on current resource usage is up to date and available at any moment, the resource usage of the group of agents stabilises (Figure 3.1), after a transient period, near the optimal resource allocation (determined by the present pay-off functions). When these agents work with incomplete information, this is done by supplying delayed information on the state of the environment, they show a chaotic behaviour (Figure 3.2), leading to a very poor performance of the overall system

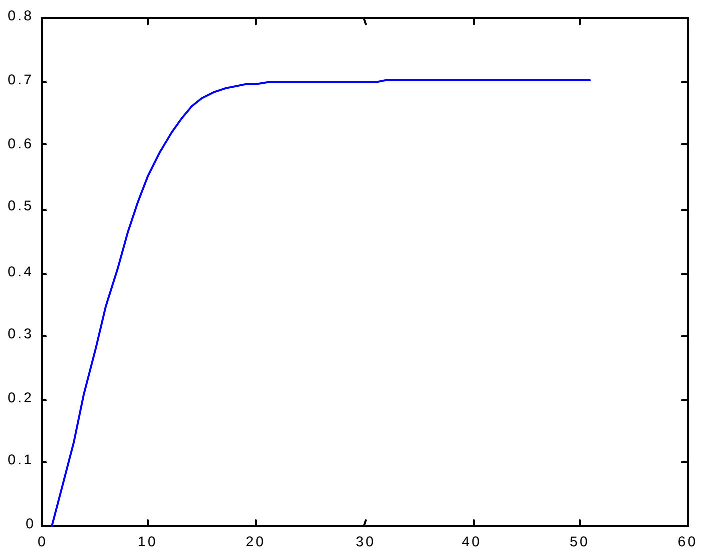


Figure 3.1 One agent with perfect information

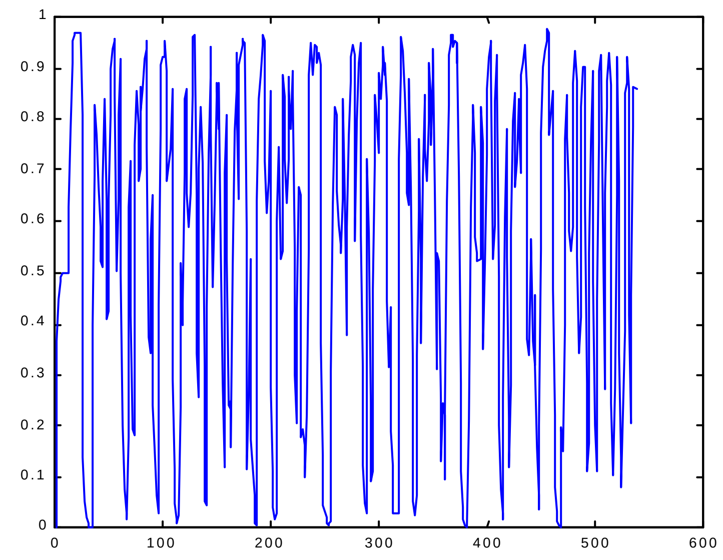


Figure 3.2 One agent with delayed information

The same problem is extended to a group of heterogeneous agents, with 40 different types of agents. Each of them has incomplete information on the state of the system, by perceiving the state of the system with a different delay. Figure 3.3 shows that resource usage also oscillates and does not reach a stable configuration.

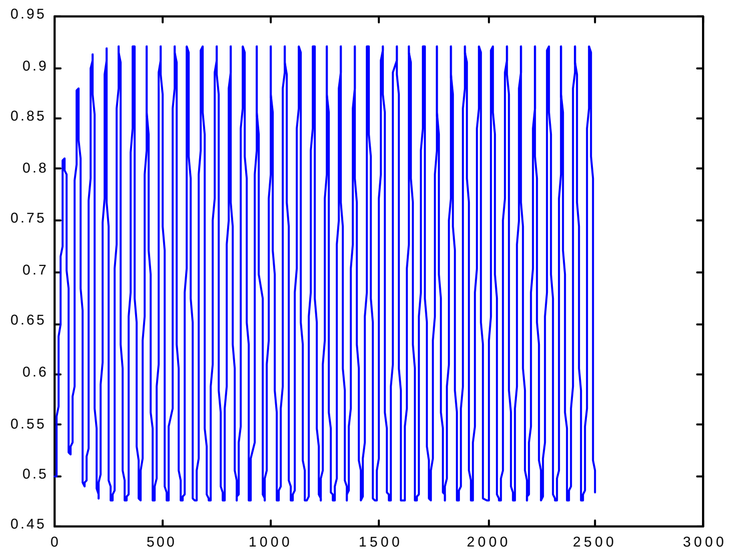


Figure 3.3 Oscillating resource choice

Next, this model is improved adding a new term (equation 4). This term rewards those agents that perform better, this means agents are able to get more pay-offs, and penalise those that perform worse. This process consists of increasing/decreasing the fraction of agents of a given type in the system depending on their relative performances (equation 5 and 6).

These are the resulting equations:

$$\frac{df_{rs}}{dt} = \boldsymbol{a}\left(f_s^{type}\,\boldsymbol{r}_{rs} - f_{rs}\right) + \boldsymbol{g}\left(f_r^{res}\,\boldsymbol{h}_s - f_{rs}\right) \tag{4}$$

$$\frac{df_s^{type}}{dt} = g\left(\eta_s - f_s^{type}\right) \qquad (5)$$

The relative performance, how good/bad with respect to the average, is computed in η_s:

$$\eta_s = \frac{f_{1s} \cdot G_1 + f_{2s} \cdot G_2}{f_1^{res} \cdot G_1 + f_2^{res} \cdot G_2} \qquad (6)$$

s denotes type of agent. In this example, each type of agent has a different perception of the system; type of agent k perceives the system with a delay of k samples. The value of k ranges from 10 to 50.

The fraction of agents of type s (f_s^{type}) in the system at each simulation step is modelled by equation 4, depending on their relative performances.

f_r^{res} is the fraction of agents, considering all types of existing agents, using resource r.

The value of r is computed by each type of agent using the information available, delayed depending on the type of agent (k, $k+1$, $k+2$... samples of time)

$g\left(f_r^{res}\eta_s - f_{rs}\right)$ is the term that rewards those agents that are doing better, in which g is a parameter similar to a.

Results show that by adding this new term the system stabilises near the optimal resource allocation. It also presents a robust behaviour when perturbations are introduced into the system (Figure 3.4). The reward mechanism and diversity are responsible for the stabilisation of the system and improving the overall system performance. Results (refer to the paper) show how the populations of the different types of agents change over time and also stabilises.

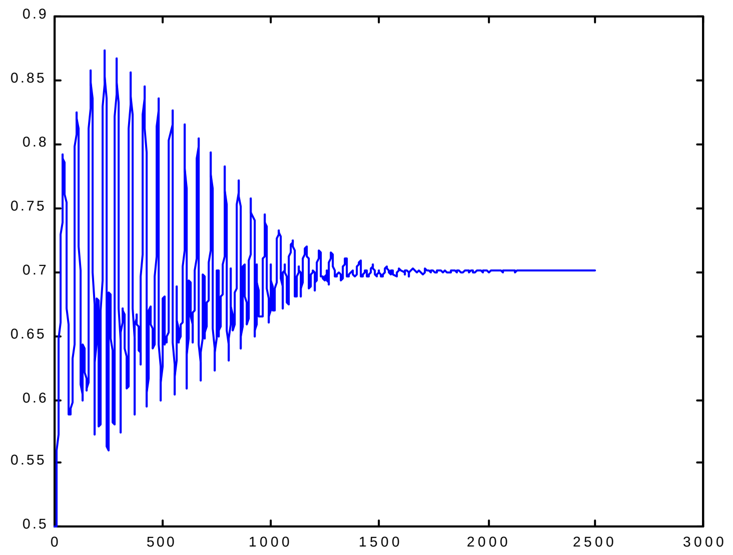


Figure 3.4 Stabilisation of resource usage in a group of agents

This work presents some interesting features:

- The whole system benefits from heterogeneity, allowing the whole system to stabilise its resource usage, thus improving the performance of the whole system. In this work, this result would not be possible in a group of homogeneous agents.

- It presents an interesting and simple mechanism to deal with heterogeneity in multi-agent systems, by means of a rewarding mechanism that allows competition among the different classes of agents.

We are going to show in this thesis that some of the ideas from this work can be applied to design heterogeneous MAS, in order to develop a mechanism that deals with heterogeneity in the system and that exploits the full potential of it. However, this model presents several limitations, as it has been defined for studying the problem of chaos in distributed systems, but it is not intended to build and model interactions in MAS, as this thesis plans. This thesis adopts this model, keeping some of its original features, to develop a new mechanism that allows to deal with the heterogeneity in a simple way in MAS. This new mechanism includes some of the aspects of the original model, but it gives a new dimension to some of the concepts of it.

This new model aims at modelling agent interactions in heterogeneous MAS (in homogeneous is also possible), and it is specially conceived for the task of learning in a heterogeneous MAS. With this new model we aim at contributing to this issue that stills represents an open problem in the AI community.

3.3 Redefining the Model

3.3.1 Introduction

This section analysis the different aspects related to this new model that have been adapted from the work by [Hogg and Hubermann 1991]. This section details the new meaning for the parameters of this model, f_{rs} and f_s. Although the equations of the model have been barely changed, the meaning of these two concepts have been redefined. As a result of this change, the model represents now different concepts, which are related to the tasks to be completed and the different agents that compose the MAS. The concept of diversity has also been changed, and now diversity concerns physical heterogeneity. The term resource has also been adapted and now is related to the tasks that must be completed by a team of robots. This new model keeps the mechanism that allows to deal with heterogeneity in the system, but changing the rest of the components of it.

Finally, this model was reviewed and some changes were introduced as a result of the observations made by Dr Tad Hogg, one of the authors of the original model, during its visit to the University of Girona in June 2001.

3.3.2 New Model

In the new model f_{rs} and f_s represent different concepts:

- f_{rs} is defined as the number of agents of type s using resource r. Now this parameter reflects the preference of a given agent for the tasks that have to be completed to accomplish the goals of the team. This value is linked to the decision-making of each agent and how they interact with other team members. This term is now redefined as the preference of agent s for task r.

- f_s is defined as the number of agents of type s in the system and is now represented by N_s. The number of this type of agents changes depending on the relative performance of each type of agents. This parameter is redefined as the fitness of agent s in the system.

This value gives a measure of how well an agent is doing in the system with respect to other team members.

- Agents re-evaluate their choices using the available information on the state of the system and the utility, in the form of payoffs, for selecting one of the resources. This term is now defined as the rewards each agent may obtain for completing a given task r, after performing an/a set of actions or after a given time has elapsed.

After redefining some aspects of the model, the general setting for this model is the following: a set of agents $S=\{s_1, s_2 \ldots s_n\}$ compete for a set of resources $R=\{r_1, r_2, \ldots r_n\}$. Agents may be heterogeneous, they may present different physical features, and resources may represent different aspects, but it can be understood in most cases as tasks, type of tasks to be completed or roles to be filled in a team of agents. The number of tasks/roles/... must be bounded and promote competition among the different team members, in order to benefit from the concept of competition for resources. Agents use the information about tasks/roles/..., in the form of rewards for using them, to update the preferences for each r and fitness of each agent. These two parameters condition the individual decisions of each agent and the decisions of team as a group.

This model is especially useful for problems, where co-operation and teamwork plays a very important role. In these situations all agents work at the same time and simultaneously towards the same goal. One of these examples is soccer, where tasks can be divided into roles, where each agent that fills a given role has a set of responsibilities that has to accomplish. In this domain all agents work simultaneously to achieve the goal of the team, which is to win the game.

The following model now defines preferences and fitness.

Preferences (f_{rs})

Given an utility for each task/role/... (G_{rs}) each agent s can compute its current preferences $ñ_{rs}$ for each task/role/... r. This value is used to update f_{rs}, that can be understood as a weighted average of each agent preferences over time. The average is helpful by several reasons:

- Agents should be able to interact several times with the different tasks/roles/... to evaluate the possible rewards they can get.
- The preferences for each role are not only determined by the physical features of the agents but also by the preferences of the rest of the team members. This process can be understood as co-learning, which usually leads to complex dynamics in the preferences of agents.

On the other hand:

- Every agent should be able to update quickly preferences, when their preferences change as a result of a change in the environment.

This value is used for the decision making process, high values of f_{rs} mean a high preference of agent s for role r resulting in a higher likelihood to select this task/role/.... These preferences can change also as a result of the changes in the value N_s.

$$\frac{df_{rs}}{dt} = \boldsymbol{a}\left(N_s \boldsymbol{r}_{rs} - f_{rs}\right) \tag{7}$$

$$\boldsymbol{r}_{rs} = \frac{G_{rs}}{\sum_{r=1}^{R} G_{rs}} \tag{8}$$

$$\sum_{r=1}^{R} \sum_{s=1}^{S} f_{rs} = 1 \tag{9}$$

***Fitness** (N_s)*

N_s is used to compare each agent individual performance with respect to the whole team and can be understood as the *fitness* of each agent in the group. Values of N_s above the average means that agent s is better adapted to the environment and is getting more rewards from it. This value is updated depending on $ç_s$ (N_s average $ç_s$ over time), the current *fitness* using the latest rewards obtained by each agent over t_w. The motivation for using this average is the same as the one for updating preferences.

$$\frac{dN_s}{dt} = g\left(\boldsymbol{h}_s - N_s\right) \tag{10}$$

$$\boldsymbol{h}_s = \frac{G_s^{total}}{\sum_{s=1}^{S} G_s^{total}} \tag{11}$$

$$\sum_{s=1}^{S} N_s = 1 \tag{12}$$

***Current Preferences /Fitness** ($ñ_{rs}$, $ç_{rs}$)*

This parameter ($ñ_{rs}$) is calculated in the original work using the function *erf*. This function returns $ñ_{rs}$ depending on the difference in pay-offs for using each resource. This function acts as a threshold, not considering significant the difference of rewards, when their values are under a given value. This is determined by ó. One of the problems of this function is that it is only designed to deal with two sources of rewards. For this reason and in order to benefit from this feature (whose benefits are detailed in other chapters) a new function has been defined, which preserves the features of *erf*.

This new function re-scales the rewards an agent obtains from each task/role/... (G'_{rs}) by means of the function *sclp*. This function works as a threshold, values under a given value are not considered significant of agent preferences and a constant value (V) is assigned to G'_{rs}. Otherwise the value G'_{rs} is proportional to (k_5) .

$$G_{rs} = sclp\left(G'_{rs}\right)$$

$$sclp : [0, \infty) \rightarrow [0, \infty)$$

$$G_{rs} \rightarrow sclp\left(G'_{rs}\right) = \begin{cases} V_1 & , G'_{rs} \leq threshold \\ \dfrac{G'_{rs} - threshold}{k_5} & , otherwise \end{cases} \tag{13}$$

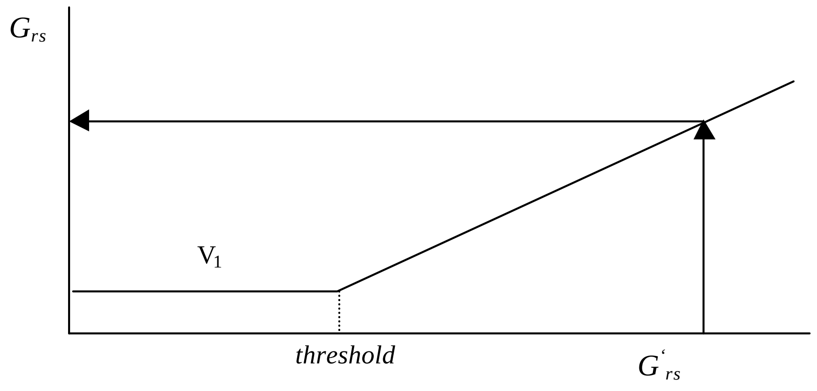


Figure 3.5 sclp function

This threshold helps the team to adapt to changes in the environment. Agents have to get enough rewards, so that the can keep their current preferences. If there is a change in the environment, agents start getting fewer rewards for some of the tasks/roles/...; thus preferences also change, allowing each agent to interact with the environment if this change is important. This threshold speeds up this process by discarding all the rewards that are below a given value.

Unlike in the original work, fitness is now computed using only the rewards each agent obtains from each task/role/.... In this case, the current fitness given the latest rewards ($ç_{rs}$) is also re-scaled by replacing the rewards from each task/role/... by the total rewards obtained by a given agent. The parameters that govern this function *sclp* can in this case take different values.

This model emulates two phenomena observed in natural systems, especially in insect and animal societies. These phenomena are task partitioning and dominance hierarchies (see chapter on related work).

- ***Task partitioning*** Agents try to choose tasks that are better correlated with their skills, as it occurs in task partitioning in many insect societies. f_{rs} models the interest of an agent *s* to perform a tasks/role/…, this interest is calculated using the rewards each agent obtains after completing the assignment. The correlation between the robot skills and the task is measured indirectly by the rewards. Usually the process of task partition is the result of a competition for tasks among the different society members [Bonabeau et al 1999]. Members that lose the competition usually have to focus on different tasks. This process is also modelled by f_{rs}. If two agents are interested in the same task/role/... this is assigned to the robot that presents the highest f_{rs}, while the other has to focus on other resources (tasks/roles/...).

- ***Dominance hierarchies*** are related to the fitness value of the model. In animal societies members compete for bounded resources, females… Usually this process results in a team organisation, where some members have priority access to resources, reproduction… Dominance hierarchies serve to maintain order and conserve energy within the group, resulting in a more efficient relation within the group. In a group of physical agents this parameter helps to maintain order, when conflicts arise within the group, and results in the formation of hierarchical dominance.

The following sections review some of the aspects related to this model: resources, rewards, heterogeneity and decision-making.

3.3.3 Resources

Resource is used in the work by [Hogg and Hubermann 1991] as a computational resource, as this work concerned ecological ecosystems, where a set of tasks had to be completed on idle computational resources. This works adapts this term and now represents the tasks that members of a team have to complete, in order to accomplish the goals of the team. Tasks in this case can be understood as a general concept with different implications:

- ***Tasks.*** f_{rs} can represent the preference of agent s for performing task r. This is the simplest case, where rewards are used to determine which is the most preferred task for a given player. If the number of tasks is high, the process of exploration and determination of the most preferred task may take very long. There are two possible ways of solving this problem, the first one by grouping tasks in roles. The second one by defining the different skills needed to complete tasks.

- ***Roles.*** f_{rs} can represent the preference of agent s for the role r. Roles allows to partition the responsibilities of a team into the members that compose the team. These responsibilities usually results in performing a set of tasks associated to the each role. This allows to complete a task by a team of robots, while each robot focuses on a part of each task.

- ***Skills.*** f_{rs} can represent also the preference of agent s for tasks involving the use of skill r. When deciding which task to perform, agents may compute a preference for a task on the basis of the skills involved to complete it.

This model, as it is defined, focuses basically on tasks that require one agent to be completed. In order to model tightly coupled co-operation, some tasks may require the use of two or more agents, this model can include additional aspects, but they are not included in this thesis, as they are beyond the scope of it.

3.3.4 Rewards

Rewards play an important role in this work. They are used to find out agent’s preferences for each resource and compute fitness. Rewards are usually given to agents depending on the effects of the actions on the state of the environment. Agents can be rewarded at two different levels.

- ***Individually.*** Rewarding an agent individually allows to evaluate each agent skills, and helps to find out which is the best player to perform each individual task. One of the problems of this approach is that this policy does not guarantee the best team performance, as agents may contribute better to the team by performing other tasks.

- ***Globally.*** Rewarding each agent depending on team performance can help to overcome some problems that may arise when considering agents individually. This process may also help to distribute agents not only according to their skills, but also to their contribution to the team performance, as preferences are also conditioned by the team global performance.

In some cases, it is difficult to define a global performance of the team or it may be difficult to perceive this global performance by the components of the team.

Rewards can measure different types of results related to task completion; this usually depends on the domain where the team is working. One of the most usual measures of individual and team performance is the time needed to complete a task/ a set of tasks. In some cases, the energy needed to complete a task/a set of tasks can also be used to define a measure of individual/team performance. In some domains individual/team performance can take different forms. For example in a rescue mission, the goal is to rescue alive as many people as possible. In soccer, teams can be rewarded depending on the team performance measured by the goal margin.

3.3.5 Heterogeneity

This thesis focuses on heterogeneous MAS, in particular in physical heterogeneous MAS. This type of systems, in most cases, is equivalent to heterogeneous teams of robots (this is the case studied in this thesis). Heterogeneous robots can be created including different types of features. In the chapter on related work, several examples have been presented where robots are heterogeneous at different levels. The following list summaries some of the features that may differentiate robots.

- ***Sensors***. Robots may have different sensing ranges; they may use different types of sensors, infrared, cameras… Some robots may also be fitted with GPS systems.

- ***Actuators***. Robots may be fitted with different types of motors, stepping motors, DC, AC … motors. Motors may also have different power, response…

- ***Size, Shape, Weight*** Robots movements and actions are conditioned by their physical limitations. Small size robots are able to explore area that big robot can not access an area, due to the size. Shape may prove more efficient for some tasks, for example in pushing an item. Heavy robots may require powerful actuators to move or it may limit the speed of the robot.

- ***Mechanical devices***. Robots may have some mechanical devices to pick up items, grasp big objects. Especially to manipulate objects

- ***Communication Capabilities.*** Robots also may differ in communication capabilities; some agents may be able to communicate with their supervisors through satellite communications. Other some robots may not be able to communicate, or only with close teammates.

3.3.6 Decision Making

MAS built using this model must include the values of f_{rs} and N_s in the process of decision making. How preferences and fitness are included in this process depends on the approach used. In the literature there are many different approaches, from very complex ones, as the one created by [Parker 1994] or very simple ones, where robots are purely reactive.

- ***Preferences.*** They should be used to determine which agents have to perform the different tasks, fill roles, take actions. If the decision-making depends, for instance, on the construction of utility functions preferences should be used to create these utility functions.

- ***Fitness.*** Fitness is already included in the preferences, as the values of preferences also increase/decrease depending on the fitness. Fitness can also be included in the decision making process, so that agents with high fitness values can participate more actively in the tasks to be completed, so that their better skills might be better exploited.

This model is defined, so that agents can co-operate at a very basic level, especially to avoid conflicts among the robots. Tightly coupled co-operation among heterogeneous robots represents a more complex problem, where additional factors must be included. This aspect is beyond the scope of this thesis, even though in the section on future works this aspect is addressed.

3.4 Model Analysis

This section compares this model with other existing learning strategies and analyses some of the features of this algorithm with respect to other works found in the literature.

Learning in heterogeneous MAS is a much more difficult task than learning in homogeneous MAS, as the state space increases due to the heterogeneous components existing in the system. Chapter 2 has reviewed the literature related to this issue. Research conducted in this area shows that there is still a lot of work to be done in this area. In fact, very few people have focused on this kind of problems heterogeneous MAS. This algorithm tries to contribute to this problem by emulating natural processes, where tasks and responsibilities are distributed through a process of interaction among the members of the group. Nature provides researchers with many examples of complex behaviour, which arises from the interactions of simple individuals. As it has been reviewed task partition and hierarchical dominances are common in animal and insects societies. This model includes these two concepts, and emulates the process of task partition and competition for bounded resources. In this case, the whole system self-organises from the interactions among the members of the system. This process ends up in an assignment of robots to tasks, resulting in a near optimal assignment, as in this process the global team performance is also included. These aspects are studied later in this thesis.

From a more theoretical point of view, this algorithm combines concepts from Reinforcement Learning and Evolutionary algorithms. The update function of preferences is very similar to a simple example of a Q-learning [Christoper, Watkins and Dayan 1992], which provides an estimate of the value of performing an action a. An agent updates its estimate *Q(a)* using the rewards *r* it gets for taking each action *a* as follows:

$$Q(a) \leftarrow Q(a) + a \cdot (r - Q(a)) \tag{14}$$

á is the learning rate that determines the importance of the new sample.

However, the equivalent formula presents some different features. The rewards an agent gets for using a given role are transformed by a function that acts as a threshold, instead of computing directly the estimation. This threshold adds some interesting features to the update rule:

- This allows to deal with the preferences for each role simultaneously, instead of updating each estimation when the information on the role is available.

- The function discards all rewards below a predetermined value. This helps the agent to adapt to changes in the environment or changes in its own physical features, as rewards may change due to a change in the environment. Therefore, each agent needs to obtain enough rewards from environment to keep their current preferences. This feature allows agents to adapt when the environment changes.

The other term that adds also interesting features is known as fitness. It can be easily understood as the concept of fitness found in genetic algorithm and genetic programming, where this term is used to evaluate several candidate solutions to the problem. In this case this term is used to evaluate each agent individually with respect to other team members. The formula to update fitness values is very similar to the equation known as Replicator Dynamics [Sigmund 1998].

$$\frac{dx_i}{dt} = x_i \cdot \left(G_i(x) - \bar{G} \right) \tag{15}$$

This formula models how animal populations x_i (members of an specie) change over time, according to their relative success in the system of each specie with respect to other species in the system. The members of a given specie increase their population, if they are performing relatively better in the system.

Fitness is responsible for the creation of dominance hierarchy in this work, where agents with higher values of fitness tend to win more conflicts. This results in several features:

- Agents better adapted to the environment, tend to participate more in the decisions and completion of the tasks, thus benefiting from their better skills.

- Team can adapt to changes in the environment more easily.

These aspects are studied later in this thesis.

Although the learning algorithm is based on simple concepts, computing a estimation of choosing a given task, the interaction of the agents that made up the system gives rise to complex dynamics, as it is observed in natural systems. This process usually results in a complex non-linear and self-organising behaviour. [Shoham and Tennenholtz 1994] observed some of these features when they applied a simple Q-algorithm to estimate the utility of taking a given action in the domain of co-operative games among several agents. Despite the simplicity of the algorithm and domain, the results showed that this process resulted in a complex, unexpected and highly non-lineal behaviour. [Parunak 1997] and [Parunak, Brueckner and Sauter 2001] have stressed the importance of exploiting the interactions among groups of simple agents to achieve complex global behaviours, as in this case.

One of the main problems of such settings is the complexity to study the properties of these emerging behaviours from an analytical point of view. Among these properties, the convergence of the algorithm. [Boutlier 1997] studied the convergence of a Q-learning in the context of co-operative games. They concluded that the conditions for the convergence of Q-learning were difficult to determine, even though they were able to define some conditions for the convergence of the algorithm. This problem is even more difficult to solve in domains where the interactions among the members of a team and the domain are much more complex.

Chapter 4

Implementation

4.1 Introduction

The research conducted in this thesis are implemented and tested in the domain of soccer. The algorithm described in a previous chapter is applied to a group of heterogeneous and homogeneous agents that learn which roles they have to select depending on their skills and the environment, so that the global task is completed efficiently. Several previous works have shown that this domain offers an interesting environment for multi-agent co-operation, autonomous robots, team-work… in which interesting results has been achieved. Some of these results have been successfully extended to other domains, network routing [Stone 1998], RoboCup Rescue [Kitano et al 1999] (rescue in after earthquakes using the research developed in RoboCup). These are only two examples of the benefits of using soccer as research domain.

One of the main problems in soccer domain is how to evaluate and test all the research developed for this domain. [Johnson, de la Rosa and Kim 1998] lay out some conditions to conduct research in soccer domain. It is important to define standard benchmarks, in order to be able to compare results from several simulations and the impact of different strategies. Also it is important to be able to work under controlled conditions. If these conditions are not well known (noise, robots malfunctions, interferences...), results might not be suitable for analysis. These aspects should be considered when deciding whether to work with real robots or simulation. On one hand real robots offer the possibility of working under real conditions and constraints, but on the other hand, it does not allow to conduct massive analysis and results may be distorted by uncontrolled external factors. Simulation overcomes these problems, allowing to work under know conditions and to conduct large number of experiments, which in real robots might take years. For all these reasons, the experiments of this thesis are conducted in simulation, using JavaSoccer [Balch 1997] in which conditions are well known and can be controlled. An estimate of the time that would be needed to conduct all the experiments of this thesis with real robots amounts to almost 10 years.

Three hand-coded teams have been selected to play against test several homogenous and heterogeneous teams of robots, in order to study some aspects of these systems. These teams present different levels of difficulty, (difficult, intermediate and easy). In addition a benchmark, detailed later in this chapter, [de la Rosa, Duhot and Muñoz 2000], has been defined to conduct the necessary experiments to measure diversity in a team of heterogeneous robots. The reason for using a benchmark, instead of playing against the hand-coded teams, is that this benchmark may represent the skills of a robotic team and simplify this analysis (see chapter on measuring heterogeneity for more details). It would be very difficult to evaluate a player individually when playing against a team of 5 robots.

This chapter reviews several aspects related to the experimental conditions of this thesis. It describes the features of the simulator JavaSoccer and the game rules that are implemented in the simulator. It also defines the benchmark, known as the implicit opponent.

Finally this chapter details in depth how the different aspects related to the model defined are implemented in the example selected for this thesis.

4.2 RoboCup Simulators

JavaSoccer [Balch 1997] is a simulator to experiment in Soccer that conforms with RoboCup rules. It has been written in Java and as a result of it can be run on almost every computer. This simulator enjoys a widespread popularity in the Robocup Community. Several teams use it to test their strategies, before they are implemented on real robots. Also it is also a tool used by many universities in courses of AI, in which teams are created using different approaches. One of the most interesting aspects of this tool, is the possibility of modifying the source code. This allows researches to modify some of the features of the simulator, in order to adapt it to each one's needs.

There are other tools available for the Robocup community for research purposes; this is the case of Soccer Server [Noda 1998]. Soccer Server is the simulator used at Robocup events, as it is the official simulator for the Simulation League. It is written in C and deals with the complexity of the soccer domain. Soccer server allows up to 11 players for each team, communications among agents are restricted to a small number of messages and the agents have to interact with a noisy and partly observable environment. One of the problems when conducting research on this simulator is its complexity, it takes at least one year to learn to write complex strategies and run the simulator. It also demands several powerful computers to run simultaneously two teams of 11 players. Besides Soccer Server has not been able to work with heterogeneous players until the summer 2001. This feature was introduced one year later after this research was started. For all these reasons JavaSoccer was selected for this research, instead of SoccerServer.

Unlike SoccerServer, JavaSoccer allows for complete environmental information, but the programmer can manage access to this information. This is what has been done in this work. The standard configuration of this simulator represents a field of the Small Size League. This league is played by up to 5 robots on each team that must fit inside a cylinder of 180 cm of diameter. The size of the field is 274.5 cm long and 152.5 cm wide; the goals are 50 cm wide. Players are detected by means of a overhead camera, although some teams also use on-board vision (http://www.itee.uq.edu.au/~wyeth/f180rules.htm for more information on the rules of this league). The simulation is performed in discrete steps. Although the time elapsed at each step is unknown by the user, the simulator has been modified to run all simulations with steps of 50 ms. In this simulator a robot's control system interacts with a defined sensor-actuator interface. At each step, each agent processes their sensory data and issue actuator commands. The simulator processes all the commands proposed by the agents, using a model of robot-ball-collisions, in case of collisions among the players, and of the dynamical behaviour of the ball and the players. Next, it updates the state of all the elements of the system (ball and players). In case a goal is scored by one of the teams or the ball gets stuck, it is moved to a new position.

Each agent is provided with the following sensors:

- ***Velocity sensor:*** present heading and speed of the robot
- ***Kicking sensor:*** possibility of kicking given the position and heading of the robot and ball

- ***Ball position sensor:*** vector pointing to the position of the ball
- ***Own goal sensor:*** vector pointing to the center of the robot's goal
- ***Opponent goal sensor:*** vector pointing to the center of the opponent's goal
- ***Team sensor:*** array of vectors pointing to the teammates
- ***Opponent sensor:*** array of vectors pointing to the opponents
- ***Score sensor:*** indicates whether the team or the opponent has scored
- ***Robot ID:*** A unique integer from 1 to the size of the team identifying the player.

Using this information, new information can be computed, for example speed of the ball, the speed of the opponents, teammates. The access to this information can be restricted by setting a perception range.

In Small Size league all this information is available by a usual team participating at the competition. Unlike JavaSoccer, a team of real robots works with noisy information, due to the lens distortion. Even though vision systems have seen a very important development over the past years in Robocup, and today many teams have vision systems with position errors of a few millimetres and up to 60 frames per second.

After this information has been processed, each agent can act on the environment by means of:

- ***Set Linear Speed:*** A real value from –1 to 1 is sent to the robot/s drive motor, indicating how fast the robot should move. (-1 and 1 are relative to each robot's maximum speed).

- ***Set Heading:*** a real value from 0 to 2π is sent to the robot's controller indicating the desired heading for the robot.

- ***Kick:*** if the ball is within a given distance of the robot, the ball is kick at a given speed (depending on the power of the kicking device) in the direction of the robot's heading. The resulting direction of the ball is affected by a gaussian noise ranging from -10º to +10º.

In addition, each player can send, at each simulation step, messages containing information on the world state, individual beliefs... to its teammates. Depending on the player, this message is received by its mates at the same simulation step, or one step later.

Although the players can have complete information of the world, in this thesis some of these capabilities have been restricted or are not used by the agents. The only information globally available is the score and the event of scoring a goal. Communications are restricted to short messages at each decision step, longer messages are sent every 100 seconds, and are broadcasted to each teammate. The communications are reliable, how communications capabilities should be implemented to be fully reliable is not within the scope of this work.

In order to achieve more realistic robot behaviour, commands sent to the simulator by the players are distorted by a random noise depending on the speed. This is no applied to hand-coded teams, described later in this chapter.

4.3 Game Rules

Some modifications have been introduced to adapt this simulator to the present Robocup rules and to solve some existing problems in the simulator.

- ***Field size.*** The field size until the year 2001 was 274.5 cm long and 152.5 cm wide. With a goalmouth of 50 cm. In RoboCup 2002 the field and the goals will be enlarged, so that more room is available to play. The reason for this change is that the field has become too crowded and hinders co-operation among players. Although the robots used in this thesis are relative small (robots can be up to 18 cm in diameter), the size of the official field is also a problem. For this reason, the field has been enlarged; the present size is 340 cm long and 220 cm. The goalmouth is still 50 cm. The new size of the field allows more co-operation among the players.

- ***Fouls.*** According to the SmallSize League Rules. A free kick is awarded when one player pushes an opponent; there are more than two attackers in the defence zone or an opponent touches the goalie. A Penalty-Kick is awarded to the attacking team when one defender (apart from the goalie) enters the defence zone and affects the course of the game. In a real game, some of these rules are extremely difficult to enforce and usually cause long discussions. In simulation, if the rules were enforced as they are, many plays would result in free kicks and penalty kicks, due to the strategy of some teams. For this reason some of these rules have been modified. The new rules depend of the type of game played. There are two different types, the first one concerns a game between a learning team and a previously hand coded team, playing always with the same strategy. The second type of game concerns a game between two learning teams. The reason for having different rules is that hand-coded teams do not have an obstacle avoidance skill and usually play defend several players inside the goal. They also tend to push the goalie when attacking. In the second type of game, players have better skills; they have an obstacle avoidance skill and defend with only one goalie.

  Learning-HandCoded: The learning team is not punished for pushing the goalie, this usually does not occur. However the hand-coded team is punished when one player has been touching the goalie for over 0.5 seconds inside the penalty (Fig.1) area S_1. In this case the ball is placed randomly on the line X_l.

  Learning-Learning: Both teams are punished when one player has been touching for over 0.5 seconds inside the penalty (Fig.1) areas (S_1 and S_2.) In this case, the ball is placed randomly on the closest *X* line to the penalty area where this foul occurred.

- ***Ball positioning***. In the original program ball is always placed at midfield, after 60 seconds time have elapsed without the ball being kicked or 5 seconds after the ball was moved for the last time. According to the rules, ball is positioned at midfield after a goal is scored. In case the ball is stuck, the referee can move the ball to a new position using a stick. In case the ball is difficult to unstuck the game is stopped and the ball is repositioned in a neutral location. Some modifications have been introduced in the simulator to conform to most of these rules.

  1. Each time the ball gets stuck or it is stopped below the line X_l or above a X_r position the ball is repositioned randomly on this line.
  2. In any other event the ball is repositioned 20 cm off (y-axis) the last position.
  3. After 60 samples have elapsed the ball is repositioned following the points 1 and 2. This prevents the ball from getting stuck between two players.

  If a goal has been scored the position where the ball is placed after this kind of event also depends on the kind of game played,

  Learning-HandCoded: After a goal has been scored the ball is repositioned randomly on the line X_l. This solves one of the existing problems when starting the kick-off from midfield. The strongest team gains easily control of the ball and as a result of this play; this team usually has very good chance to score a goal with no defenders.

  Learning-Learning: After a goal has been scored the ball is repositioned randomly on the line X_l or X_r. Depending on the team that has scored the goal. If the team that defends the left goal scored the goal, the ball is placed randomly on the line X_l. If the other team scores the goal, the ball is placed randomly on the line X_r.

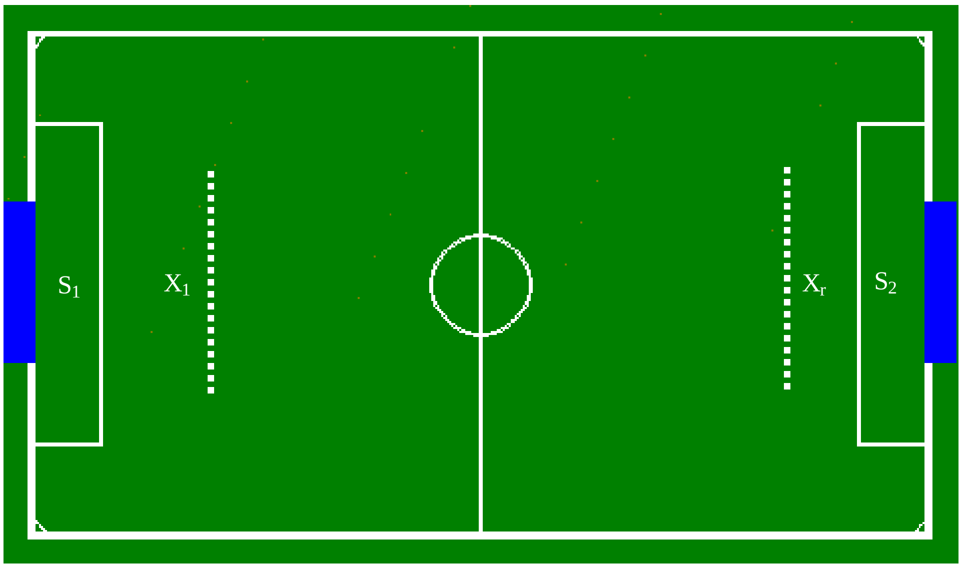


Figure 4.1 Field zones and ball repositioning

4.4 Hand-Coded Teams

Several hand-coded teams have been selected to conduct part of the experiments, in order to study the features of teams composed of heterogeneous and homogeneous physical agents. These teams are BrianTeam, SchemaDemo and ou7kou and represent several levels of difficulty. The two first teams are included in the JavaSoccer package and the third one was hand-coded by several students of AI course. The first team is regarded as an easy team; the second one can be as an intermediate team, and third one as a difficult one.

BrianTeam. This team may be considered an easy team. It usually loses all the games by a wide goal margin. This team follows a very simple strategy, if a player has control of the ball, it has several choices. It shoots the ball, if no opponent robot can intercept the ball. In any other case, it can pass the ball to a teammate if it is in a better position or may decide to dribble. This team defends or attacks depending on the position of the ball. If it is on its own side of the field it defends, if the ball is on opposite side of the field it attacks. It also defends, if an opponent is within a given distance from the ball.

SchemaDemo. This team may be considered an intermediate team. Even though almost all teams are able to win this team, the goal margin tends to be quite narrow in some cases. This team also follows a simple strategy. The players of this team try always to get behind the ball and move it forward, while keeping the ball away from the opponents. One of the most interesting aspects of the strategy of this team, is the formation they keep while moving with the ball, or attempting to regain control of the ball. This formation hinders the opponent to regain control of the ball, as with this formation the ball is extremely protected.

ou7kou. This team may be considered a difficult team, only a few teams are able to win a game against it. The roles designed in this thesis are based on ou7kou, even though this team implements a much more simple strategy. This team consists of 5 roles, goalie, defender, right/left midfielder and striker. They agree on their co-operative actions following a similar strategy to the one used in this thesis. One of the most important differences with this work, apart from the number of roles, is that they do not have implemented an obstacle avoidance skill. In addition their commands are not affected by gaussian noise.

4.5 Benchmark Generator: Implicit Opponent

4.5.1 Introduction

The principle of this benchmark is to use a team of robots against a fictitious (implicit) opponent that is modelled by an inclined field with variously shaped static obstacles (Figure 4.2 and Figure 4.3). For this reason this benchmark is called implicit opponent. The goal of the team will be to play against this implicit opponent and for instance play a normal match, or do other type of plays. This will be further discussed later.

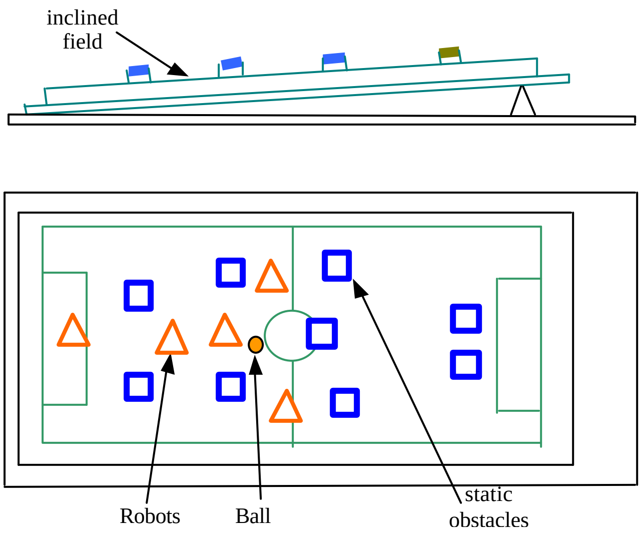


Figure 4.2 Proposed benchmark set-up

This benchmark is a generalisation of benchmarks that were proposed by [Johnson, de la Rosa and Kim 1998b]. The expected interesting features of this benchmark are:

- The implicit opponent's team behaviour is clear and predictable.
- Any number of robots from 1 can evolve in any F-xxx platform both in real and in simulation. The F-xxx contains any RoboCup and FIRA platforms.
- It contains requirements for control, co-ordination, and co-operation.
- Continuous interest in the evolution of the benchmark: this is good for continuous and dynamic movements. Its provides with good data for learning and emergent behaviours.
- These good features allow a good reference benchmark so that all existing teams can be evaluated/compared in a robust and general way.

4.5.2 Formulation of the Benchmarks Generator

This benchmarks generator will be based on the principle of an implicit opponent.

4.5.3 Purpose

The purposes of the benchmark generator are:

- To set rigorous scientific standard for research into robot football.
- To encourage teams to work on the same problems and to allow comparison without any explicit match of teams each other.

- To collect and publish data on robot control and ball control, and co-ordination.
- To enable scientific analysis of the performance of teams worldwide.
- To enable any particular team to gauge its performance against these standards.
- To provide a *simple* baseline from which new scientific benchmarks can be defined.
- To create proper families of useful benchmarks useful for adaptive/emerging techniques of AI.
- Testing of individual or collective behaviour.
- Testing with real or simulated robots in a real or simulated environment.
- Generated benchmarks have to be easy to build.

4.5.4 Description of Parameters

Every benchmark **B** with a name "name" will be defined by the tuple $< \mathbf{F}, \mathbf{P}, \mathbf{P_r}, \mathbf{P_o}, \mathbf{T_e}, \mathbf{F_b}, \mathbf{C}, \mathbf{J} >$. The performance of the team at every experiment on the benchmark will be contained in instances of **J**. All these terms are defined as follows:

The Formula

The formula **F** is both the 2-D terrain and type of football robots both in real and in simulation that will be used to implement a benchmark. We will use the nomenclature of RoboCup. So **F** are:

- **F-1 Simulator League**
- **F-180 Small Size League**. Small real robots.
- **F-2000 Middle Size League**. Middle real robots.
- We define the simulation league that uses JavaSoccer as the **F-2 Simulator League**.
- We define the MIROSOT league as the **F-100**

The origin of every formula is (0,0) metres; point that will be situated in the top left corner.
In the future surely there will be other formulae. The formula **F** defines the limits of the terrain, and the type of football robots. For instance, today the size of the terrain in F-180 and F2 is a Ping-Pong table, F-2000 is of 16 Ping-Pong tables, F-1 is the size of 105x68 metres, F-100 is a little more that half Ping-Pong table. However, our *implicit opponent* benchmark will be conceived to be equal and perfectly used in any formula. This is very important, especially to prototype results in simulation (F-1) and then easily exports them to the other real implementations (F-180 and F-2000).

The Angle of Inclination

It consists in inclining the terrain of the platform formula **F** in some angle **P**. In this benchmark generator the angle will be normalised in a set of positions using radians in a range that clear dynamics of the ball and robots will be observable. For instance $\mathbf{P} \in [0, 0.2]$ radians. This makes the ball move continuously down the angle against the goalkeeper of the tested team. The behaviour of a free ball is simply its dynamics driven by the gravity's force. This will introduce continuous movement of the ball, which will behave as if a hidden *implicit opponent* team of agents was moving there.

The Pattern of Evaluated Robots

The pattern of evaluated robots $\mathbf{P_r}$ consist of a certain number of robots $\mathbf{N_r}$ of the evaluated team, that will be any number from 1 with $\mathbf{N_r} \in [1, \infty]$, the shape $\mathbf{S_r}$ ={round, square, others} the robots are made of, the total area occupied by the robots $\mathbf{A_r}$ and finally a pattern of their initial conditions, that is the initial geometric positions on the 2-D terrain, that will consist in a list of **<x, y,q>** positions, with units in metres and refereed to the same origin of the formula **F**. Thus, a pattern of evaluated robots $\mathbf{P_r}$ will be defined by the tuple $\mathbf{<N_r, S_r, A_r, <x_1, y_1, q_1 >, ..., <x_{Nr}, y_{Nr}, q_{Nr} >>}$

The Pattern of Opponents

The Pattern of Opponents $\mathbf{P_o}$ consists of a certain number of obstacles $\mathbf{N_o}$, what shape $\mathbf{S_o}$ ={round, square, others} the obstacles are made of, total area occupied by the opponents $\mathbf{A_o}$ and finally a pattern of their geometric positions on the 2-D terrain, that will consist in a list of **<x, y>** positions, with units in metres and refereed to the same origin of the formula **F**. Thus, a pattern of opponents $\mathbf{P_o}$ will be defined by the tuple $\mathbf{<N_o, S_o, A_o, <x_1, y_1, q_1 >, ..., <x_{No}, y_{No}, q_{No} >>}$

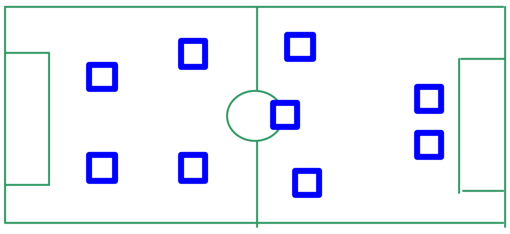


Figure 4.3 An Example of a pattern of opponents, with 9 obstacles

Note that there will be $\mathbf{N_o} \in [0, \infty]$ obstacles.
In the case of $\mathbf{N_o}$=0, if the interpretation of N_o is the number of opponents, then No=0 is only possible when **P**=0 (the case that there is no opponent).

4.5.5 The Time to Do the Experiment

It is necessary to limit the time $\mathbf{T_e}$ to do experiments in the benchmark for their better comparison. Units are in seconds. Its value can range [0, ∞].

4.5.6 The Features of the Ball

The features $\mathbf{F_b}$ related to the ball have to be declared: type of the ball $\mathbf{T_b}$ {light = ping-pong, average = golf, heavy = pin-ball}, whether the ball can or cannot rebound in walls, **CAN** = {rebound, not rebound}, the random function **R** that will randomly generate <x, y,θ> positions (units in metres and origin the same of formula **F**) to place the ball in the field whenever the ball get stuck and the seed **SD** for the random function, which will be an integer. Then $\mathbf{F_b}$ will be defined by the tuple $\mathbf{<T_b, CAN, R, SD>}$

4.5.7 The Constraints

A first simple constraint for coercing the co-operation skills of the evaluated team is introduced. It consists of limiting the time $\mathbf{T_p}$ any agent is allowed to have the ball under its control before getting rid of it or passing it to another agent. This time will be $\mathbf{T_p} \in [0, \infty]$ seconds.

A second constraint is the norm of velocity of robots of the evaluated team, $||\mathbf{V_r}||$, that is the norm of the angular and linear speed or robots. Declaring this constraint is important since there will be some types of benchmarks where this magnitude will be lower than the maximal velocity of the ball $||\mathbf{V_o}||$ "driven" by the implicit opponents. In this situation, $||\mathbf{V_r}|| < ||\mathbf{V_o}||$, co-operation is strictly necessary. In the situation $||\mathbf{V_r}|| < ||\mathbf{V_o}|| / \mathbf{N_r}$, that is the maximal velocity of any robot of the evaluated team is lower than the maximal velocity divided by the number of evaluated robots, then the co-operation of all robots will be strictly necessary. This measure will be in units of metres/second and ranges [0, ∞].

A third constraint is the norm of velocity of the shooting velocity $||\mathbf{V_s}||$ for any robot of the evaluated team. This measure will be in units of metres/second and ranges [0, ∞].

A third constraint is a prohibited zone **Z** that consists of a square where the robots cannot enter in. Units are in meters, and the square is declared by its four corner points **Z (<x_1, y_1>, <x_2, y_2>, <x_3, y_3>, <x_4, y_4>)**. The origin is the same origin of the formula F we were using.
Then the constraints **C** is a tuple of **< T_b, $||V_r||$, $||V_s||$, Z >**

4.5.8 The Performance Index

The performance index **J** is based on measurable features, which could include the following:

- The total amount of goals: **G_i** for the implicit opponent, **G_e** for the evaluated team.
- The differential amount of Goals **G_e - G_i**
- The (average **aTis**, best **bTis**, worst **wTis**, etc) **time** that the implicit opponent team needs to score (that is, the time the evaluated team receives a score):
- The (average **aTes**, best **bTes**, worst **wTes**, etc) **time** the evaluated team needs to score.
- The average distance **aDb** and standard deviation that the evaluated team maintain the ball from its local goal area.
- The ratio **R_p** of the number of successful passes versus the unsuccessful passes.
- The (average **aTet**, best **bTet**, worst **wTet**, etc) **time** that the evaluated team can maintain the ball in a target area.
- Others to be declared in the future.

The performance index **J** will be generated by the joint evaluation of these measurable features.

4.5.9 Benchmark Implementation

- A special strategy has been coded for the static players, necessary for the benchmark (see below), called *Ghost.java*. *Ghost.java* has no decisions programmed, as a result of it players do not move from their initial positions.

- A constant acceleration has been included in the model of the ball in order to simulate an inclined field, according to the benchmark (see below benchmark). This acceleration changes depending on the inclination of the field. The equation for this acceleration is:

$$a = g \cdot sin(a) \qquad (16)$$

g is the gravitatorial acceleration
a is the inclination of the field

These modifications have been also included in class *GolfBallS.java*.

4.5.10 Summary

$$\textbf{B "name" < F, P, P}_r\textbf{, P}_o\textbf{, T}_e\textbf{, F}_b\textbf{, C, J >} \qquad (17)$$

- **P** is an angle $\in$ [0, 0.2] radians.

- **P_r** is a pattern of evaluated robots and is the tuple
$$\langle N_r, S_r, A_r, \langle x_1, y_1, q_1\rangle, ..., \langle x_{Nr}, y_{Nr}, q_{Nr}\rangle\rangle \qquad (18)$$

- **P_o** is a pattern of opponents and is the tuple
$$\langle N_o, S_o, A_r, \langle x_1, y_1, q_1\rangle, ..., \langle x_{No}, y_{No}, q_{No}\rangle\rangle \qquad (19)$$

- **T_e** is a time to do experiments $\in$ [0, ∞].

- $\mathbf{F_b}$ contains the features of the ball and is the tuple

$$\mathbf{<T_b, CAN, R, SD>} \quad (20)$$

- **C** is the constraints and is the tuple

$$\mathbf{< T_b, ||V_r||, ||V_s||, Z >} \quad (21)$$

- **J** is the performance index

Several benchmarks are created later to evaluate each class of robot individually. This is discussed later in the chapter on measuring robotic heterogeneity.

4.6 Model Implementation

4.6.1 Introduction

This section details how the different parts of the model have been implemented. Agents are rewarded depending on the short and long term effects of their actions in the environment. Teams are also rewarded depending on their global team performance, score margin, when compared to past performances. Physical heterogeneity is created by building robots with different size, actuators and control strategy. Decision-making is implemented by means of a consensus technique that has been successfully applied to a team of real soccer robots [de la Rosa et al 1997]. The parameters of the model are easily integrated in this decision-making strategy.

4.6.2 Resources-Roles

One possible way of achieving teamwork in a team of agents is by partitioning the responsibilities into the agents that compose the team. This is can be achieved by defining a set of roles. Roles allow to define which are the responsibilities that a player filling a given role has to accomplish, in a team that works towards the same goal. This approach have been used by many teams that have participated in Robocup, as human teams split up responsibilities by defining beforehand each player duties. The use of roles is a mechanism that allows the co-ordination of the team members that work towards the same goal, allowing each agent to accomplish a part of the task and avoiding at the same time conflicts among agents.

A role defines a number of responsibilities for the agent that decides to play that role and it is responsible for fulfilling them [Cheng and Padgham 1997]. In order to meet these responsibilities a strategy is chosen, depending on the role of the agent and the conditions of the environment. Roles may be rigid completely specifying an agent's behaviour, or may be flexible leaving a certain degree of autonomy to the agent filling the role [Stone and Veloso 1999].

This approach presents several problems,

- Inflexibility to short-term changes (one robot is non-operational)
- Inflexibility to long-term changes (a route is blocked)
- Lack of facility to reassigning roles.

Some of these problems can be overcome by creating complex mechanisms that allow dynamical role assignments.

Normally a role consists of several hand-coded rules that determine what agents have to do given the present environment (in soccer, position of the ball, position of teammates, opponents) and have to deal with the lack of perfect information in the system. Each role can contain a different number of rules and kinds of actions. How some actions are performed usually changes depending on the role.

In this work each role contains a limited number of high level actions

if *condition1 and/or condition2 and/or........* then *action1*

These rules propose actions to agents depending on the position of the ball and the player on the field, which is taking the decision. Each condition is a fuzzy variable, whose values are defined by a fuzzy set. Each rule has a certainty (*j*) associated (*j Î [0,1]*), this value depends on the level of activation of each conditions and the operators (*AND/OR*). Conditions and operators are combined together using fuzzy algebra and implemented by means of a possibilistic approach. Two or more agents can use the same role at the same time, as long as it consists of more than one action. Before a decision is taken, a process consisting in two confidence parameters revises these certainties: Prestige and Necessity (see section on Consensus Technique). These two parameters can increase/decrease the certainty *j*. Initially each robot takes the decision with the highest revised certainty, but in case of conflicts, for instance two agents had chosen the same action, the agent with the highest certainty would win, while the other agent would have to choose a different action.

Each agent gets rewards when it makes use of these roles (see Rewards) depending on the resulting performance of taking that action. Eleven roles have been coded using fuzzy rules,

- ***Goalie (GL)***, its task is to keep the goal. It tries to block all shots coming from the opponents and tries always to optimise the area covered of the goal with respect to the position of the ball. The goalie never leaves the penalty area.

- ***Fullback (FB)***, its task in defensive plays, is to block any shot and pass from the opponent when attacking, regain control of the ball and prevent other players from having good shooting positions and entering the penalty area. In offensive plays, its mission is to regain control of the ball to start a new attack in midfield.

- ***Right/Left Defender (RD/LD)***, they share the tasks of the fullback, but they are also responsible for moving the ball to midfield by dribbling.

- ***Defensive Midfielder (DM),*** its task is to regain control of the ball in defence, so that a new attack from the own field may be started by kicking away the ball. In offensive plays its task is to regain control of the ball in midfield.

- ***Right/Left Midfielder (RM/LM)***, its task is to regain control of the ball in defensive/offensive situations in order to move the ball by dribbling towards offensive positions.

- ***Center Forward (CF),*** its task is to regain control of the ball in defence/offense and to move the ball from midfield positions, wingers and striker, by dribbling. The player , which fills this role can also shoot the ball from outside the penalty area if it has a good chance.

- ***Striker (ST)***, its task is to score goals by dribbling and shooting. It also has to attempt to intercept the ball when coming from the defenders of its own team, so that it can have a good chance of scoring goal in the presence of no/few defenders.

- ***Right/Left Winger (RW/LW)***, they share tasks with the striker and their mission is also to help the striker, by regaining the ball near the penalty area and passing the ball to the striker. They are also allowed to try to score goals by shooting or dribbling, if they have a good chance.

Figures 4.4 and 4.5 show the zones covered by each role when the ball is on the own/opponent field. White areas denote common influence areas to adjacent roles.

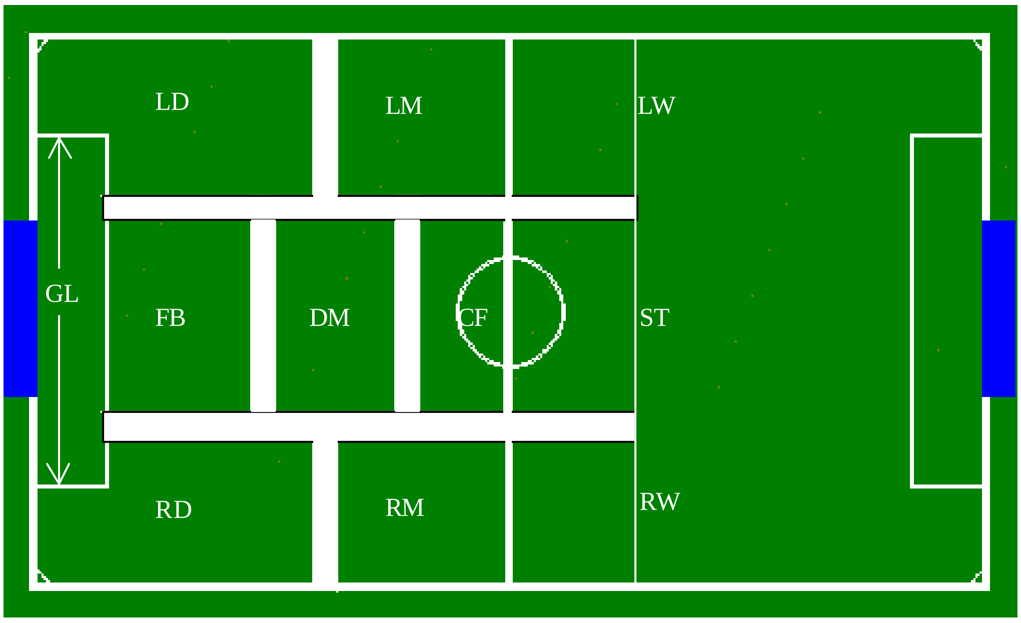


Figure 4.4 Roles and Areas in Defence

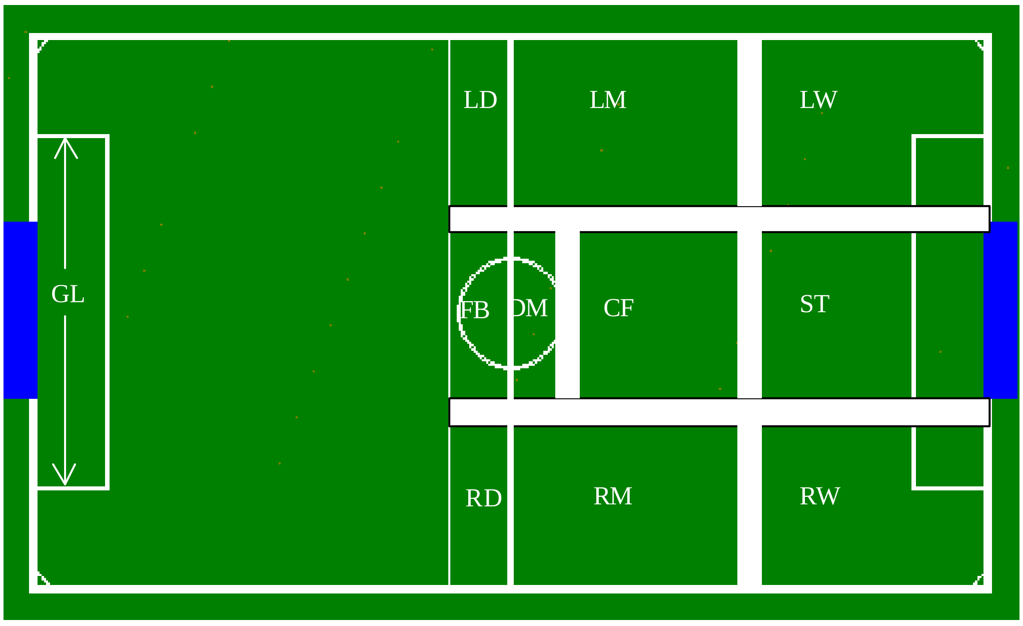


Figure 4.5 Roles and Areas in Attack

It has been mentioned previously that one of the features of ecosystems is that species compete for bounded resources, in this case, where an ecosystem is emulated, agents have to deal with an environment with bounded resources. It seems a contradiction that the amount of roles (11) exceeds considerably the number of players (5). Bounded resource availability is achieved by designing roles with a limited (high level) number of actions, only six actions have been defined, and most of them shared by all/several roles, leading to many possible conflicts among the agents. However the number of roles allows to conduct a deeper analysis of several aspects of heterogeneity and adaptability, there are enough roles to have several kinds of team's formation (offensive, defensive...). The six actions that have been defined are:

- ***Defend (DF)***, action only defined in the role of goalie and that aims at blocking any attempt from an opponent to score a goal by dribbling or shooting.

- ***Kick Ball (KB),*** action that can be taken by every role that aims at kicking away, shooting the ball or dribbling using the kicking device of each robot. This action can only be taken when the ball is within a given distance from the kicking device.

- ***Go to Ball (GB),*** action used to approach the ball on the own side of the field and at the same time trying to stay between the ball and the own goal.

- ***Attack (AT),*** action used to approach the ball on the opponent side of the field and at the same time trying to aim at the opponent goal.

- ***Regain Ball (RB),*** action aimed at regaining control of the ball when the ball is behind the player.

- ***Stay in Area (SA),*** action taken when the ball is far from the area covered by the role. This action allows the player to try to get as close as possible to the ball, but always trying not to interfere with other team-mates, and at the same time not leaving its assigned area.

As it has been mentioned, only one player can take the same decision at the same time to avoid conflicts. In addition, the following actions **Attack**, **Go to Ball** and **Regain Ball** are also conflictive actions because all these actions share the same goal, getting to the ball. For this reason only one of these actions can be taken at the same time.

Table 4.1 summarises the actions associated to each role. In Annexe a more detailed explanation of roles and actions can be found together with some more specific details on team design.

ROLE	SYMBOL	DF	KB	GB	AT	RB	SA
Goalie	GL	X	X				
Fullback	FB		X	X		X	X
Left/Right Defender	LD/RD		X	X		X	X
Defensive Midifielder	DM		X	X		X	X
Left/Right Midfielder	LM/RM		X	X	X	X	X
Center Forward	CF		X		X	X	X
Left/Right Winger	LW/RW		X		X	X	X
Striker	ST		X		X	X	X

Table 4.1 Actions and Roles

4.6.3 Rewards

Rewards are related to the actions they take and their impact of these actions on the environment. Each agent gets rewards, if any, for the roles they use after performing the action and evaluating the result of it. If this action has been performed on the transition areas between the roles, rewards are split up between the two roles. Agents can perceive and evaluate the short-term results of their actions using their sensors. Long term results of actions, score/receive a goal, are broadcasted to all robots when they occur.

Every t_w seconds (in this work 100 seconds) agent s adds all rewards obtained over the past t_w seconds of time for each role r (R_{rs}), as each agent can make use of all available roles. Once the overall rewards are computed, they are increased/diminished depending on the score, and some additional data related to each player. In order to exchange data among team members, they make use of their communication capabilities every t_w seconds. The amount of data can be considered small and amounts to less than 100 bytes/second for each agent.

Rewards are awarded taking into account several aspects:

Position of the player and role

Any time an agent takes a decision, its position and the role it is using are compared. If the player is within the area covered by the role, a reward is given if it results from that action. If an action results in a reward, and the agent is outside the area for the role that is using, it is not rewarded, as it is the result of an unplanned action. Agents can be pushed by other team-mates or opponents outside its assigned area, they can also have to use other roles as a result of the limited number of actions or move to a different position, as they are avoiding other players that are on their way. Agents keep track of the number of times a decision is taken at the right place, as this information is used later.

Short Term Rewards

All roles except the role of goalkeeper are rewarded depending on the result of their short-term actions. This kind of rewards are planned so that the own players may be able to evaluate the results of their actions, making use of their sensors and taking into account their limited perception of the world.

Short term rewards consist in giving rewards to agents, depending on the ability of each player to move forward the ball, by kicking, passing, dribbling... within a few seconds (2 seconds) after gaining control of the ball. If the number of seconds is small the robots are able to perceive the result of their actions, as the position of the ball is within its perception range. If after this time the player is still in control of the ball, the rewards are computed, and the whole process starts again. This rewarding mechanism may also include the possibility of punishing the robot for its actions, if the ball is behind the initial position after the predetermined time has elapsed. One of the drawbacks of punishing robots is that players with very bad skills will not be able to get rewards, leading to the non-convergence of the algorithm for these kind of players.

Although this mechanism seems quite simple, results show that it works quite efficiently and allows simultaneously the evaluation of several robot skills that define the ability of a robot to play soccer:

- ***Regain control of the ball***. The only possible way for a robot of being rewarded is by gaining control of the ball. As its ability for this task increases, it is likelier for it to be rewarded.

- ***Dribbling ability***. Once the robot has gain control of the ball, one possible way of getting rewards is by moving forward with the ball. If the robot has a good dribbling skill the rewards it may get increases considerably.

- ***Obstacle Avoidance***. Obstacle avoidance is tightly related to the dribbling ability. As the opponent loses control of the ball, it tries to regain control of it as fast a possible, making difficult, for the robot in control of the ball, to advance towards the opponent goal. If the ability of this robot to avoid obstacles is very poor, it will very difficult for it to get rewards.

- ***Passing ability***. Players with powerful kicking devices are able to pass the ball to players that are in more offensive positions. Although kicking the ball is an easy task for this kind of players, they have to kick the ball when no opponent is close enough to block the ball, as the ball may be blocked or deflected by an opponent. If they are able to combine these two aspects, they can get an important amount of rewards.

These rewards are given as follows:

- All roles are rewarded, except the role of goalie.
- All roles are identically rewarded.
- Negative rewards are also given, only if the play starts near the opponent goal this kind of rewards are not given. (It is difficult for a player to get rewards if it starts the play on the opponent penalty area).

$$rst_{rs} = St_r \cdot \left(ball\_x_{t+2} - ball\_x_t\right) \tag{22}$$

St_r is a constant, and it is identical for each role.

Long Term Rewards (Goals scored)

Long term rewards concerns scored goals. Although allowed goals may also be included in long-term rewards, analysis conducted during this stage of the work proved that punishing robot for goals allowed was a difficult task. Many times robots responsible for failing to accomplish their tasks, resulting in a goal scored by the opponent team, were no punished by their actions. This led to exclude this event from this group of rewards. However allowed goals are included when rewarding/punishing players filling the role of goalie.

For the case of scored goals, all players may be rewarded, except the goalie. The reason for rewarding all kind of roles for this event is that scoring a goal usually is the result of a co-operative action. A defender may regain the ball in defence, a midfielders may move the ball forward by dribbling, and the striker may finally score a goal *(t_last_goal).* Players are rewarded depending on the time elapsed since they had control of the ball for the last time *(t_last_touch)*. As the time elapsed increases, the amount of rewards decreases. Also how rewards diminish over time change also depending on the role (*Kt*$_r$), as the long term effects of an action may take more time, for example, for the role of fullback than for the role of striker.

The values of Lt_r are identical for each role.

$$rlt_{rs} = Lt_r \cdot e^{\left(\frac{t\_goal - t\_last\_touch}{Kt_r}\right)} \tag{23}$$

One of the problems of this type of rewards is that some agents may be rewarded for taking wrong actions, which have made more difficult to score a goal. However result show that this rewarding mechanisms works satisfactorily.

Goalkeeper ability

Goalkeeper ability is only evaluated when a player is inside the penalty area using the role of goalie. There are three different results that are rewarded.

- The ball is moving into the goal and the goalkeeper is able to push the ball forward

- The ball is moving into the goal and the goalkeeper deflects the ball into a non-dangerous direction.
- An opponent is pushing the ball by dribbling into the goal and the goalkeeper is able to block the ball.

These actions are rewarded depending on the speed of the ball. Players that are able to make saves when the ball is moving fast have to be better rewarded as they show a better skill for this tasks than those that make saves at lower speeds.

$$rg_{rs} = R_r \cdot bvel \tag{24}$$

On the other hand a goalie also must be evaluated from the point of view of the goals that it allows. Goals are used to decrease rewards at the end of the period of time used to compute rewards (t*w*). If the opponent scores a goal, the player that scored the goal decreases their rewards as follows:

If a goal is allowed, the player acting as goalkeeper, if there is one, is punished.

$$Rg_{rs} = Rg_{rs} \cdot na_s^{-nr} \tag{25}$$

goalie n_r=n_1

na_s : Number of goals allowed when agent *s* was inside of the influence area of the role of goalie.

Rewarding agents for the number of goals scored helps to increase the preferences of agents skilled to play in offensive positions for offensive roles. Although all kinds of rewards detailed so far are important, more important is to know which are the players that can score goals, as this is the goal of soccer. A similar situation explains the punishment for the role of goalie. In this case, the role that is specially punished is the goalie. This role can get rewards from several kind of situations and it is highly rewarded for it, but if a goal is usually is scored by the opponent team, this may mean that this player is not the right player for this role of goal For this reason the goalie is harshly punished when this occurs.

Once all rewards obtained by using actions are added at the end of *tw* seconds (here capital letter R stands for the sum of all individual rewards for agent *s* and role *r*),

$$R_{rs} = \left(Rst_{rs} + Rlt_{rs} + Rg_{rs} \cdot na_s^{-n_r}\right) \tag{26}$$

A set of parameters and transformations are applied to R_{rs} before a final value is obtained to update the model.

Reward Estimation

Using additional information each agent tries to figure out an estimation of the possible rewards (PG_{rs}) for each role. This aims at estimating the rewards that a given player would have obtained if had made use of a role *r* at each decision step over the analysed period. In order to compute these estimations, several aspects are considered.

- As each agent keeps track of its decisions, it can compute, every t_w seconds, the number of times that the role r was correctly used (P_{rs}). Using the values for all the roles, each agent can compute the fraction (p_{rs}) of the number of times that this role *r* was correctly used.

$$p_{rs} = \frac{P_{rs}}{\sum_{r=1}^{R} P_{rs}} \tag{27}$$

- Estimation of the possible rewards each agent can get from each role is sometimes based on incomplete information as agents only have made use of some roles a few times. *ì*$_{rs}$ is a conservative confidence parameter depending on the amount of information available for calculating PG_{rs}.

$$m_{rs} = \frac{1}{1 + k_1 \cdot e^{-f \cdot k_2}} \tag{28}$$

ì$_{rs} \in$ *[0,1],* k_1, k_2 are constants

- All roles, except the goalie, are rewarded identically. However the area (Sf_r) covered by each role is different, as it can be seen in figures 5 and 6. In order to evaluate each role, considering that each one covers an identical area, a new parameter is defined (sc_r)

$$sc_r = \left(\frac{R \cdot Sf_r}{\sum_{1}^{r=R} Sf_r} \right)^{-1} \tag{29}$$

R is the number of roles.

Finally, all these parameters are used by each agent to compute an estimation of the rewards, which it can expect to obtain for using each of the roles. G_{rs} is then computed as follows,

$$PG_{rs} = \frac{R_{rs}}{p_{rs}} \cdot m_{rs} \cdot sc_r \tag{30}$$

Team's Performance

So far agents are able to figure out the rewards they can get for using a given role, but all agents working as a team have a common goal, to win with the greatest score margin. If they are not able to win the game, they should be able at least to lose the game to with the smallest differential of score. For this reason rewards are also subject to changes depending on the overall team performance, measured by the score margin. The concept is quite simple, if the team performance is increasing/decreasing; rewards are increased/decreased depending on the score differential.

One of the main problems for using learning algorithms in soccer domain is the huge state space of this domain; it becomes even larger when the players in the environment present diverse skills. It is also a highly stochastic domain and the opponent can change its strategy over the game. One additional problem is to know who is to be awarded/punished for scoring/receiving goals; this strategy may reward agents that are not contributing to the overall team performance.

The process of learning form the score starts by compering the partial score of the past t_w seconds (ds_k) with a weight average ($dgav_k$) of the past partial scores, this gives an idea of how the team has been performing in the past. This value is calculated as follows:

$$dsav_k=b_1 \cdot ds_{k-1}+b_2 \cdot dsav_{k-1} \tag{31}$$

$b_1,b_2 \in [0,1] \quad \wedge \quad b_1+b_2=1$

Rewards are increased/decreased depending on the values of ds_k and $dgav_k$. Only if the value of ds_k is greater than $dsav_k$ and less than zero, then rewards are not changed. Although in this situation there is an improvement on team performance, the goal of the team is to win by the greatest score margin. It is possible that the opponent is an excellent team and the learning team does not have a chance to win. However by imposing this condition, a team will expend more time exploring the state space, trying to find test more solutions, as this rule will slow down the process of convergence of the algorithm. For the same reason if the value of $dgav_k$ is less than zero and ds_k greater than $dgav_k$ 0 will replaced $dgav_k$.. All this process is calculated by the function *learn.*

$$G'_{rs} = learn(PG_{rs})$$

$$learn : [0,\infty) \to [0,\infty)$$

$$G'_{rs} \to learn(PG_{rs}) = \begin{cases} PG_{rs} \cdot (1+k_3 \cdot (ds_k - dsav_k)), & ds_k > dsav_k \\ \dfrac{PG_{rs}}{(1+k_4 \cdot (ds_k - dsav_k))}, & dsav_k > ds_k \\ PG_{rs}, & \text{otherwise} \end{cases} \tag{32}$$

k_3 and k_4 are constants. k_4 may contain the same value as k_3. These formulas include the case for an identical score.

We think that there are other possible methods for accomplishing the same goal, but after analysing several possibilities we have come to the conclusion that this strategy keeps things simple and results in an important increase of team performance. One of approaches that was discarded after a long evaluation, was the inclusion into the function *learn,* of the role usage of each agent. Rewards for each role, from the point of view of each agent, were increased/decreased depending on the role usage. In case the performance improved, roles that were used more often saw their rewards increase more than other roles; the opposite when the overall performance worsened. Results showed a poorer performance than this simpler approach.

4.6.4 Heterogeneity

In robotic soccer and in RoboCup in particular, teams of heterogeneous robots is becoming a common feature in some teams of middle-size league. In the small size league, many teams play with four identical robots and use a slightly different robot to play as goalie. In 2001 in Seattle, FU-Figthers played with several kinds of robots, holonomic, non-holonomic, cylindrical, rectangular... Depending on the features of each robot, they filled different roles during the game.

In this work diversity will be created designing robots with different physical features and from the experience of RogiTeam in RoboCup. There are several features that can be modified in each robot in order to achieve different behaviours:

- ***Size (Radius of the robot)***
- ***Linear/Angular Speed***
- ***Kicking Device Power***
- ***Dynamical Behaviour***

Although another possible way of building heterogeneous robots is by changing the shape, the simulator conditions the shape. Since the simulator used in this thesis does not allow to change this feature, the shape of each robot is cylindrical.

From the experience of RogiTeam in building robots, big robots are heavier, as they carry more batteries and more material is needed. This affects the speed of the robot and its dynamical behaviour. From the experience on designing controllers, controllers tend to be more accurate when they are designed to controlling a robot that moves slowly, this feature is also affected by the accuracy of the vision system. On the other hand when it is designed to control robots at faster speeds (and smaller) tend to be more inaccurate. Bigger robots can also contain enough room to fit a powerful kicking device inside, while this is not so easy for small robots, whose kicking devices usually are not so powerful.

Using these ideas, heterogeneous players are created by building robots ranging from fast, inaccurate, and with small kicking devices to small, accurate and with powerful kicking devices. In this work inaccurate robots means robots that continuously oscillate and that have problems to reach the end point (see figure below). The left figure shows the possible behaviour of a slow robot, while the left figure shows the behaviour of a fast robot.

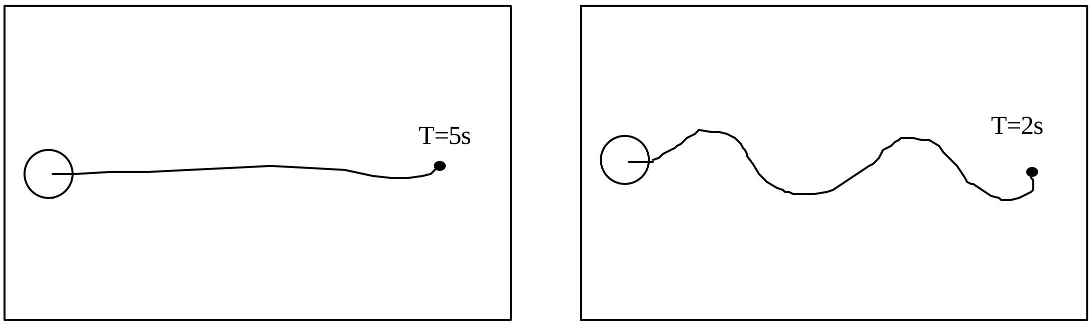


Figure 4.6 Examples of dynamical behaviour of a robot

Dynamical features also determine the behaviour of the robots when avoiding a teammate or opponent and the ability to control the ball. Although fast players can also move slowly, their controllers have been designed to move fast, for this reason all players when moving will run always at the same speed.

All teams will be composed of five robots, each one having the features of one of the following kinds of robots:

Agent	Radius (cm)	Translation Velocity (m/s)	Turning Velocity (rad/s)	Kick Speed (m/s)
A1	7.0	0.25	5.28	0.50
A2	6.5	0.30	5.78	0.35
A3	6.0	0.35	6.28	0.30
A4	5.5	0.40	6.78	0.30
A5	5.0	0.45	7.28	0.20

Table 4.2 Robot Features

All these players with different physical features present a wide range of different behaviours and skills for playing soccer. Although all robots can perform all the tasks of a team playing soccer, they accomplish these tasks with different efficiency, as a result of their skills and environment. All players present different set of skills that allow them to contribute to the team work in different ways.

4.6.5 Decision-Making

Agents co-operate in order to accomplish their goals, this technique of co-operation focuses on reducing the number of conflicts among several agents, for instance preventing two or more agents from taking the same decision. A consensus technique was used in soccer robotics [de la Rosa et al 1997] in order to reduce the amount of conflicts and it is implemented by RogiTeam [de la Rosa et al 2000] on its real robots. The procedure is the following: initially each robot takes the action with the highest value of activation (*j''*) after the revision process imposed by the consensus technique. In case two or more agents intend to take the same decision, the one with the highest value of certainty wins. This process is repeated until every agent have taken a different action. Agents make use of their communication capability to exchange information and to reach an agreement on their decisions.

The revision process is based on two parameters, *Prestige* (*P*) and *Necessity* (*N*), which modify the value of certainties (*j*). *P* is related to each agent preferences. Both values have unique values depending on each agent and role (here both values are used without subindex for the sake of the explanation, see later on this section for more details).

Prestige performs a linear transformation over *j* (level of activation of a fuzzy rule leading to an action).

$$j'=P(j)=P\,x\,j \qquad (33)$$

Then, *Necessity* performs a non-linear transformation over *j'*

$$j''=N(P(j)) \qquad (34)$$

This consensus process can be implemented through different approaches, probabilistic, possibilistic and conservative. Using the probabilistic approach, the one used in this work, the resulting formula is:

$$j''= P\,x\,j + N\,x\,sigm(P\,x\,j - q) - P\,x\,j\,x\,N\,x\,sigm(P\,x\,j - q) \qquad (35)$$

q is a threshold $\left(q = \frac{0.1}{N}\right)$

Prestige can be understood as the confidence that an agent has in one role. Higher values of Prestige mean high confidence values for these roles. Prestige is a conservative parameter, if an agent has a low confidence on one role, the value will very low. Here this value is assimilated to the preferences of each agent.

Necessity is a parameter that increases *j '* according to the *necessity* of the source of information that is being revised. This is a non-linear parameter that modifies agent decisions based on their preferences (*j '*); in order to prevent agents from settling into one/several roles too quickly. Its goal is similar to that of Boltzmann exploration used in reinforcement learning to explore the environment. As agents interact with the environment and increase their preferences for a role/roles the effects of this parameter on agents decision making are negligible, but it helps when the environment changes as agents start changing their preferences allowing some of them to explore again the state space.

4.6.6 Model and Decision Making

In order to deal with the idea of preference for one role, the value of frs is used to assign the parameter *Prestige* (*P*) , for the certainty revision process. A high preference of agent *s* for role *r* results in high values of P_{rs}.

$$P_{rs} = \frac{1}{1 + \exp\left(\frac{\frac{1}{R \cdot S} - f_{rs}}{k_6}\right)} \tag{36}$$

k_6 is a constant.

R is the number of roles and *S* the number of agents.

The values of Necessity are computed depending on the role usage. As every agent interactswith the roles and it starts to show more preference for one or more roles the effects of this parameter tend to disappear.

$$N_{rs} = N_{max} \cdot \exp(-k_7 \cdot f_s^{total} \cdot \frac{p_{rs}}{f_{rs}}) \tag{37}$$

k_6 and k_7 are constants and $f_s^{total} = \sum_{r=1}^{R} f_{rs}$

4.6.7 Implementation

This section explains how everything works together, step by step, to have a clear idea of how all these parts explained in depth fit together.

Step1: Variables assignment

- The number of own players, opponent players, features of each robot is described in a configuration file.

- Values of Necessity start usually with a value of 0.5. Preference values, as they are fractions, have to add up 1. Considering this constraint, agents/roles can be initialised at different values. This allows the designer to code preference of an agent(s) for one (or more) of the roles.

Step2: Game starts and decisions are taken.

- **Position in the field:** Initially each player has a position in the field depending on their position in the file configuration file.

- **Decision making:** Using initial values for Prestiges (normally 1/(R·S) and Necessity (normally 0.4) each player evaluates and revised each rule for each role, in order to find which action has a highest revised certainty. Agents propose this action to their mates. If there are no conflicts with other agents, they transform this decision into a lineal and turning velocity. In case two or more agents want to take the same decision the one with the highest uncertainty wins, if there is a draw the one with a lower robot identification wins (this is quite unlikely). The other agents have to take their second best choice and go through the same process and check if there are conflicts with other agents. In order to proceed in an orderly manner, the agents with the highest uncertainty start the process, and in turn agents with lower certainties continue the process. JavaSoccer allows the user to design agents with communication capabilities. This process is repeated at each sample time in JavaSoccer (≈0.5 seconds) for *t* seconds.

- **Collecting rewards:** Each agent, using its perception capability, computes rewards for each action, depending on their positions and taking into account the roles that it is using. Also agents are informed when a goal is score by its own team or by the opponent.

- **Keeping track of decisions:** Every time a decision is made, it is saved together with the role and if the player was inside the area assigned to the role.

- **Communication of information:** Every t_w seconds, agents broadcast the information of their individual performances in an orderly manner on their total rewards.

- **Reposition of players:** Each time a goal has been scored, players are repositioned according to two criteria:

    1. If $0 \leq t < t_1$ players are repositioned randomly in the initial positions
    2. If $t \geq t_1$ players are repositioned in the initial positions according to their closer roles, in case there are conflicts the same rule as in decision making is applied.

Step 3: Updating Prestige and Necessity values for the next t_w seconds.

- Using information coming from each agent populations (values of f_{rs}) are updated, using rewards for each agent and role. The Euler method has been used to approximate this function. *Dt* for the approximation of each function changes through the time:

$$\begin{array}{ll} 0 \leq t < t_1 & Dt = Dt_1 \\ t_1 \text{ £ } t < t_2 & Dt = [Dt_1, Dt_2] \\ t_2 \leq t < \text{¥} & Dt = Dt_2 \end{array} \qquad (38)$$

Initially *Dt* is a very small value (0.005), after some time this value increases up to a certain value. This change allows the agents to go through many different situations initially until they "decide" to fill a role

- Using information on individual role usage the new values of Necessity are computed.
- All population values and role usage for each agent are saved in a file.

***Step 4:* Go to decision making in Step 2.**

This model deals with the way decision making is done at two different levels. Each agent s tends to use the roles with the highest preferences (f_{rs}). In case two agents think that role r is the best for them, then N_s plays a decisive role, as it has been mentioned before. As this parameter increases/diminishes the values of f_{rs}, agents with higher N_s tend to win these conflictive situations, resulting in a hierarchical structure of the system, where some agents are dominant with respect to others in case of conflictive situations.

4.7 MAS Analysis

In a previous chapter the model defined in this thesis has been classified according to a setting defined by [Stone and Veloso 2000]. This section attempts to compared not only the model, but also the domain and problem selected in this thesis, with respect to the principles defined by [Parunak 1997] (already reviewed in the related work section) to design a MAS based on nature principles. The reason for this analysis is that this model emulates some natural phenomena observed in natural systems.

Generally speaking, this MAS meet most principles, but the most important ones are not completely satisfied by the team of robots.

Although there are some more principles, only the most relevant to this work are reviewed.

- ***Agents are Small in Mass, in Time and Scope***

The number of agent is very small (5) with respect to the environment.

- ***Agents are Small in Time***

Agents are able to learn and adapt to the environment, as it is shown in this section.

- ***Agents are Small in Scope***

The components of the MAS and their actions can cause important changes on the environment, for instance scoring a goal. On the other hand these agents have limited sensing, they only can perceive objects within a short range.

- ***Support Agent Diversity***

Agent diversity is achieved by designing agents with different physical features.

- ***Redundancy***

Agents can complete all tasks, they can use any role and select and perform all available actions.

- ***Decentralisation***

Agents explore the environment by themselves and teammates are used as a source of information on decisions.

- ***Feedback and Reinforcement***

Agents are rewarded depending on how they perform, the whole team is also rewarded/punished depending on how well they are performing. This process of reinforcing agents results in a process of self-organisation in the MAS

- ***Share information***

Agents share information using their communication capabilities.

- ***Randomisation***

Parameter *N* allows agents to explore the environment introducing some kind of randomness in the decision making of each agent.

- ***Plan and Execute Concurrently***

Agents perceive the environment and act depending on their perceptions. A step later, they analysed the environment and they can change its previous decision.

Chapter 5

Model Analysis

5.1 Introduction

This chapter continues the analysis of the algorithm developed to allow a group of heterogeneous agents to learn which are the best tasks they can perform, in this thesis represented by a group of roles, on the basis of each agent skills and the global team performance. It studies some of the properties of this algorithm and analyses how the parameters that define this model affect the features of a team of robots, mainly performance, adaptability, convergence...

This chapter starts by studying the process of self-organisation and the process of adaptability to changes in the environment. This is analysed by means of three examples. The first example studies how a heterogeneous MAS learn and self-organises when playing against a fixed opponent, the second one analyses the same case but when both teams are learning, and finally the third one studies how the team evolves when the environment changes over time.

Later, a set of parameters is analysed, especially from the point of view of adaptability and algorithm convergence. One of the most interesting parts of this model is studied, a parameter that represents the fitness of each agent in the system. This parameter is analysed in terms of team adaptability and team performance.

Finally, the last section outlines some general recommendations for the design of MAS using the approach selected for this thesis. This last section attempts to generalise some recommendations, so that this model might be used to design MAS, and they can benefit from the interesting features that it presents.

5.2 Dynamics Analysis

5.2.1 Introduction

In this section the convergence of this algorithm is analysed. The next section analyses how the convergence can be conditioned by some of the parameters.

As it has been already mentioned in chapter 3, this algorithm can be considered a combination of a simple reinforcement learning and evolutionary algorithms. The power of this algorithm lies in the resulting dynamics of the interactions of the members of the system. This aspect is underlined by [Parunak 1997] and other authors in the subfield of Swarm Intelligence as a desirable feature for MAS.

In this section we do not pretend to do an exhaustive theoretical analysis of this algorithm. Many researchers have already studied Reinforcement Learning and Evolutionary Algorithms and some of the properties are well known in the AI community. This section attempts to study some of particularities that the combination of these two algorithms present.

From the point of view of RL, if only one agent is considered, an agent can be able to figure out which is the best role to select under different environmental conditions, in case it has enough time to explore the environment and can interact with the different roles. This analysis becomes more difficult, as more agents are added to the system and everyone is trying to learn independently from each other. This gives rise to very complex dynamics, also studied by [Shoham and Tennenholtz 1994] and can be considered a co-learning process, where each agent is changing the state of the environment, as they learn. This section analyses this process of self-organization, when several agents learn concurrently. This is studied in three different scenarios, learning, co-learning and adaptation. These cases are exemplified by means of three examples.

In order to facilitate the reader's understanding of each example, a table, which contains the basic information on the state of the system, has been defined.

A1			A2			A3 (1)			A4			A5		
ST	86%	20.3	LM	95%	21.3	RM (2)	92% (3)	21.1 (4)	MD	98%	24.7	GL	100%	23.0
0.1971			0.1993			0.1991 (5)			0.2043			0.2001		
55-1 (6)			90 (7)											

Table 5.1 Table for State Description

(1) Denotes the name of each agent. In the examples selected for this chapter, all teams are composed of 5 fully heterogeneous robots. *An* denotes for this example a robot of type *n*. For more details see chapter 4, where the features of each type of robot are described.

(2) Represents the roles that are used more often by the indicated agent

(3) Indicates in percentage terms, the number of times that the role in (2) has been used (after the revision imposed by the consensus process)

(4) is the preference that agent *n* has for the role in (2). This value changes according to the model defined and the rewards each agent obtains from the environment. Although according to the model definition in chapter 3, this value must be smaller than 1, for notation purposes this value is rescaled from 0 to 100.

(5) is the fitness of agent *n* in the system. This value also changes depending on the model, and depends on the information on rewards exchanged by the agents.

(6) is the accumulated score over 10 past steps.

(7) are the number of steps elapsed after the beginning of the experiment. Each step represents 100 seconds.

Although the following examples are illustrated by means of heterogeneous teams, these processes do not vary significantly for homogeneous ones. Later in this chapter, some differences in adaptability between heterogeneous and homogeneous MAS are analysed from a general point of view.

5.2.2 Learning

The process of learning can be divided into three different phases. The designer can determine these phases by setting the different parameters of the model. Due to the large number of simulations to be performed in the course of this thesis, some parts of this process are artificially conditioned by the designer to speed the simulations up. In this case, these changes concern the role of goalie and the learning rate.

The process of learning usually starts by assigning to each agent the same preference and fitness, so that they start under the same conditions. Each simulation consists of 180 steps; during this time the values of preferences and fitness are updated, using the rewards obtained over each step. At the end of these 180 steps, this time is considered enough for the algorithm to converge and the values of preferences and fitness are not updated any more. The performance of the learned team is tested over 60 additional steps. Over the first 90 steps each agent acts as goalie the same number of times, the reason for this exception is that the learning process takes place very fast. As a result of it a robot may converge very fast towards the role of goalie. The role of goalie presents a particularity, in case a robot has the same preference for each role, it always chooses the role of goalie. This ensures the presence of a goalie in the team formation. If enough time was available, this would not be necessary. In many examples of learning in the soccer domain, the goalie is treated as a special case. For the other roles this problem is not so crucial, because it is possible to interact several times with each of them.

Phase 1. Interaction with the Environment

During the first phase agents interact with the environment and tend to fill all roles several times. As initially preferences are very similar, the exploration of the environment is determined by the Necessity, one of the parameters that define the consensus process. The use of this parameter ensures that all agents interact several times with all roles. At this point in the process the team presents a chaotic behaviour, where agents start using a given role, but a few seconds later, they may switch to a different one. As a result of these interactions and the amount of rewards agents obtain, they change their preferences. These initial preferences depend on several factors:

- ***Opponent skills***. The disorganised behaviour that the system presents may keep the team away from reaching attacking positions, if the opponent, for instance has very good attacking skills. This may force agents to focus on defensive/midfield roles at the beginning.

- ***Physical features***. Each robot's physical features determine the amount of rewards that each player may obtain; the skills needed to fill a role depend usually on the opponent (see chapter 7 for more information).

- ***Other teammates***. Other teammates can usually have a similar preference for the same role. This situation can make more difficult the use of a given role by other agents, as the number of conflicts increase (two or more robots decide to choose the same action in the same field area). This may decrease the probability that other players select this role.

Also during this part of the learning process the goalie is used in turn by all agents, as it has been already mentioned. All this process takes place at a very low speed as the learning rate (see next section on parameter analysis) is set at a very small value, so that agents have enough time to explore the environment. This part of the learning process may take around 40-50 steps (≈ 100 seconds), according to how the parameters have been selected.

Phase 2. Competition and Search of Opportunities

During this second part of the learning process, agents start to focus on a small number of roles, even though they are still exploring the environment. Agents usually focus on two or three different roles, whose preferences for them are slightly higher than for the rest of the roles. One of the things that usually takes place at this point in the game is that several agents may have interest for the same role. This is due to the same factors detailed previously. This gives rise to a competition for roles, which emulates the competition for bounded resources that can be observed in animal societies. Agents that have a similar interest for the same role, tend to fill the same role in turn, as the Necessity parameter allows agents to explore other roles, when the preferences are still very similar. This process may lead to different results, depending on different factors inherent to the team:

- One of the players finally settles into this role, because it has been able to get more rewards than the other teammates competing for the same role. This leads to higher preference values and increases the possibility of winning in case of conflicts for this role. These preferences may also increase due to higher fitness values; this parameter favours those agents that are performing better in the environment. Other agent/agents that were interested for the same role has/have to start exploring again the environment or focusing on other roles, for which its/their preferences are also high.

- One/several players withdraw from this fight, as they have found that other roles provide them with more rewards.

- A change in the environment as a result of the decisions of other teammates. Agents not participating in this competition for this role have started to fill other roles that make decrease the possibility of obtaining rewards for this role. This may lead these agents to explore other roles, as the preferences for this role may start to fall. Two robots competing for the role of left-winger when there is no striker can exemplify this, as they are able to score many goals from this position. If one player suddenly starts filling this latter role, this may decrease the chances of obtaining rewards from this role.

This process of competition may take from 40 to 50 steps and usually finishes when all agents start to have a clear preference for one of the roles and start using them over 50% of the times.

Phase 3. Consolidation

During the last phase in the learning process, agents tend to increase the preferences for the role that at the end of previous phase were using most of the time. In some cases, some players may finally switch to a different role; there are several reasons to explain this change:

- Other agent's preferences. This problem has been already mentioned in the previous phase.

- The increase in the order of the elements of system may help the team to play in a more organised way, allowing players to control the ball better and to play in a more coordinated way. This new situation may help the team to get to the opponent goal with the ball under control and thus changing agent's preferences.

- As agents converge to one role, they may explore from this position other close roles. This is explained by the fact that many roles overlap and some level of excitation remains in the system. Sometimes these agents may find that it is possible to obtain more rewards from one of the adjacent roles. This may give rise to changes in the team formation, as one agent may finally switch to a new role, leading other agents to explore and change to other roles, due to the change in the environment. Fortunately this situation hardly occurs.

At the end of this process a collection of agents is assigned to different roles, although they may also use other roles as a result of the conflicts among the different players. As a result of the whole process, the team presents a team formation adapted to the opponent and with a very efficient role distribution among the different heterogeneous players. This last part of the process can take 40-50 steps.

The following figure shows how the score for a learning team and a hand coded team changes over time. The learning team is represented by the black line and the opponent by the dotted line. Although initially the opponent performs better than learning team and the goal margin is increasing, the learning team is able to self-organised and increase the number of goals scored, while the opponent only is able to score very few goals after the step 20.

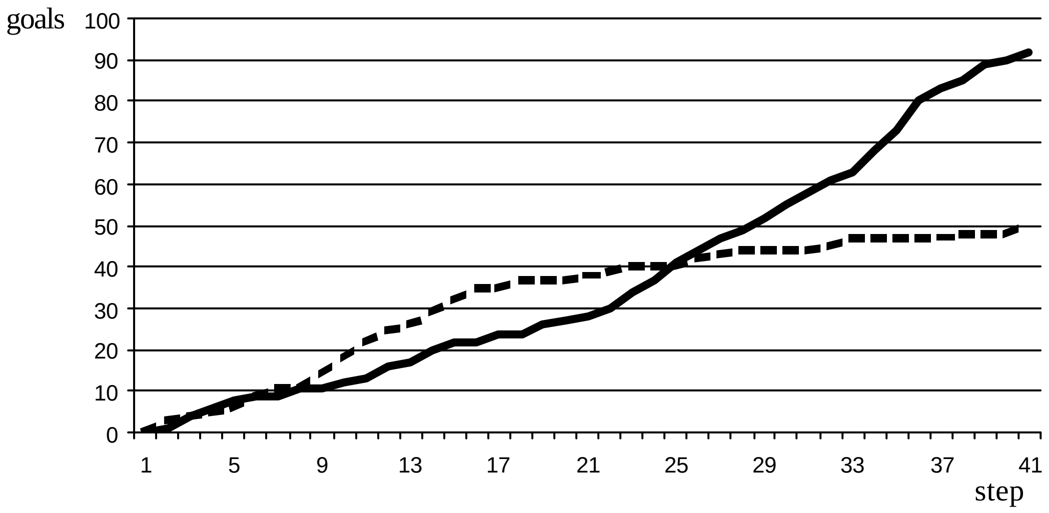


Figure 5.1 Score for Learning vs Hand-Coded Team

The following example illustrates the process explained at the beginning of this section. This example analyses a game between a learning team and a fixed opponent, in this example this team is Brian Team. The details on this team can be found in chapter 4. The learning team is made up of 5 fully heterogeneous players. This process is evaluated at different time steps.

A1			A2			A3			A4			A5		
MD	35%	18.5	GL	20%	17.9	ST	29%	18.3	GL	20%	18.1	RD	24%	18.2
GL	20%	17.9	RD	18%	18.2	GL	20%	18.1	LM	15%	18.2	GL	21%	18.3
			ST	18%	18.2				RM	12%	18.3			
0.2			0.2			0.2			0.2			0.2		
58-0			60											

After 60 steps, agents are still exploring the environment and playing in different roles. Agents' preferences show that these values are still very similar (even identical) with respect to the initial preferences; these low values allow the robots to continue to explore the environment. At this point in the game there are two aspects that are interesting from the point of view of the competition for roles, apart from the role of goalie. A1 presents the highest preference (18.5 vs 18.2) for a given role; the role of MD, the preference for this is role is correlated to its physical features. Its high speed allows it to move the ball very fast from defence to offense. In the same example there are two agents, A2 and A3, that are showing a growing interest for the role of ST (18% and 29%). In this case, these preferences are also related to the skills of each agent. Both agents have good abilities to play in offensive positions against this opponent.

A1			A2			A3			A4			A5		
MD	34%	18.6	LM	55%	19.2	GL	19%	18.0	GL	20%	18.5	MD	34%	18.7
GL	21%	17.9	GL	19%	17.6	RD	14%	18.1	RD	18%	18.3	GL	20%	18.2
						ST	13%	18.4						
0.2			0.2003			0.2			0.2			0.1999		
58-14			90											

After 90 steps, the role of goalie is definitely filled by A4, as its preference for this role is higher than for any of the members of the team. The competition for the role of striker has been won by A3 and sees its preference for this role increase, while A2 has found a new role (LM) that provides it with a large amount of rewards. This role is also tightly coupled to this type of player, as the abilities required to fill this role, specially good dribbling skills, are very similar to those needed for offensive positions. While the competition for the role ST has finished, a new competition for the role of MD is taking place between A1 and A5. Both players are using this role the same number of times and it’s the most preferred role for both agents, with very similar preference values. A5 is able to obtain many rewards in this role by kicking away the ball. The team performance has deteriorated in the past steps; this is especially true for the number of goals allowed, as a result of the continuous fights for roles.

A1			A2			A3			A4			A5		
RM	34%	18.5	LM	89%	19.3	ST	51%	19.0	GL	100%	23.6	MD	85%	20.1
LD	19%	18.2				LW	14%	18.2						
MD	13%	18.4												
0.2005			0.2			0.1995			0.2001			0.2000		
52-0			120											

After 120 steps, the competition for the role of MD has ended and A5 finally fills this role. As a result of this process, A1 has started to explore other possible roles. At this point in the game all the players, except A1, are presenting high preference values for one of the roles and are consolidating their positions in the team. However, A1’s preferences are still very similar to the initial ones and is still exploring several possible roles. The fitness is still very similar for all the players. The goal margin has improved as a result of the greater order in the system.

A1			A2			A3			A4			A5		
RM	100%	20.3	LM	100%	20.2	ST	86%	20.8	GL	100%	29.1	MD	89%	23.2
												FB	11%	21.7
0.1993			0.1991			0.1976			0.2018			0.2022		
84-1			240											

After 240 steps, all players have converged to one role. A1 has finally found a role to play and is acting as RM. Agents’ preferences are the same as in previous period evaluated, but their values have increased over time. Fitness values have increased for the roles of GL and MD. This allows A5 to use other close roles, as FB. The team performance has considerably increased, by 30 goals, due to the result of the process of self-organisation of the team, where players have converged to different roles, correlated with their skills, and the level of organisation in the system has improved.

This role distribution may be regarded as one of the best possible against this type of opponent, according to some tests performed by assigning robots to roles manually. Results for the process of learning show that if enough time is available, robots tend to converge to a near optimal role distribution. The use of the goal margin as measure of team performance contributes considerably to this result. Initially this aspect was not included leading to a very poor team performance, as agents filled those roles from which they got most rewards. Against easy teams team formation was composed only of attackers, but there were not enough players in midfield, leading to very low team performances. With the inclusion of this learning feature, results started to show a considerable improvement in the goal margin.

5.2.3 Co-learning

Co-learning is a very similar process to the one detailed for the case of learning, with the difference that teams are not playing against opponents with fixed policies, but against a team that is also learning. The main change can be observed during the second and third phase. Usually one of the teams tends to converge more rapidly towards a team formation than the other. This usually occurs as a result of an improvement in team performance (score margin) over the previous time steps. This leads the second team to adapt to the team that has started to converge and it usually starts to converge to a team formation that usually results in an improvement in its own team performance, even though it can be still losing the game. The convergence of the second team may lead the first team to adapt to the changes in the second team, even though this does not always occur. Usually this last change consists in a change of role in one of the players. Both teams tend towards a fixed team configuration after this process finishes, although it is possible to observe small variations in the team formations over time. In this thesis teams are evaluated after 240 steps. This time is considered enough for both teams to converge to a final formation.

This example shows how two different teams evolve when they try to learn from each other. The first one (denoted by A) is composed of a team of fully heterogeneous robots; the second one (denoted by B) is a team of homogeneous robots. As in the previous example, the first 90 periods all robots fill the role of goalie the same number of times.

A1			A2			A3			A4			A5		
LM	25%	18.2	GL	24%	18.1	GL	21%	18.0	GL	20%	18.1	MD	26%	18.2
GL	20%	18.1	MD	15%	18.2	CF	19%	18.2	MD	15%	18.2	RW	25%	18.1
												GL	19%	18.7
0.2			0.2			0.2			0.2			0.2		
23-30			60											

B1			B2			B3			B4			B5		
CF	26%	18.4	LM	23%	18.5	GL	21%	18.0	FB	20%	18.2	RD	21%	18.1
MD	20%	17.9	LW	22%	18.4	RM	6%	18.1	GL	20%	18.0	GL	19%	19.0
			GL	20%	17.8									
0.2			0.2			0.2			0.2			0.2		
30-23			60											

After 60 steps, both teams are still interacting with the environment, and all robots fill in turn the role of goalie. Although the preferences for the different roles are still very similar, results show that some players in the homogenous team are showing greater preferences than the players in the heterogeneous one. Player’s preferences focus on midfielder roles, as most part of the game takes place in midfield, where both teams attempt to regain and control the ball. The score shows that the homogenous team is leading the game. This aspect can be explained, as a result of the types of players of the heterogeneous team, where all players initially fill the role of goalie, but not every robot is equally fit for this role.

A1			A2			A3			A4			A5		
LM	77%	18.6	RM	81%	18.6	LD	83%	18.4	MD	90%	18.8	GL	100%	24.3
0.1997			0.1996			0.1994			0.1997			0.2015		
22-20			120											

B1			B2			B3			B4			bB5		
MD	82%	19.2	LW	80%	18.9	RM	81%	18.6	RW	43%	18.2	GL	100%	24.0
									LM	36%	18.2			
0.1997			0.1998			0.1995			0.1992			0.2017		
20-22			120											

After 120 steps, players are converging to roles. The homogenous team has been the first one to converge and the heterogeneous one is attempting to adapt to this team strategy. The role of goalie is filled by only one robot A5/B5 in both teams. Although the most used roles are midfield ones, the homogenous team has two robots filling partly attacking positions (B2 and B4), while the heterogeneous team is made up of defenders and midfielders. The reason why the heterogeneous team has a defender (A3) is that the homogenous is playing with several attackers, as a result of the faster convergence of the homogeneous team. The score margin has decreased almost to 0, because the role of goalie in the heterogeneous team is now filled by a good goalkeeper.

A1			A2			A3			A4			A5		
LM	61%	18.8	RM	78%	18.8	LD	89%	18.2	MD	92%	19.2	GL	100%	34.5
LW	26%	18.8												
0.1990			0.1987			0.1972			0.1988			0.2062		
17-18			180											

B1			B2			B3			B4			B5		
MD	91%	19.7	LW	84%	18.7	RM	82%	18.8	LM	93%	18.9	GL	100%	31.1
0.1990			0.1978			0.1985			0.1976			0.2071		
18-17			180											

After 180 steps, the players continue to converge to roles. A few changes can be noticed at this point in the game. Both teams present a very similar team formation. In the heterogeneous team, A1 is starting to play in more offensive positions and in the homogenous one B4 is finally playing as a LM, helping B2 in midfield. Fast players (A1 and A2) in the heterogeneous team are responsible for filling attacking roles, while A3 and A4 are playing in midfield/defence. The score shows that both teams are scoring/allowing practically the same number of goals.

A1			A2			A3			A4			A5		
LW	88%	19.5	RW	86%	19.4	LD	89%	18.2	MD	92%	20.6	GL	100%	41.2
0.1975			0.1974			0.1944			0.1976			0.2130		
21-22			240											

B1			B2			B3			B4			B5		
MD	92%	19.7	LW	88%	18.6	RM	80%	19.1	LM	95%	19.1	GL	100%	36.4
0.1982			0.1953			0.1975			0.1963			0.2127		
22-21			240											

After 240 steps, both teams have finally converged to roles. Simulation show that teams have finally stabilised in these formations. The heterogeneous team presents a formation with two attackers (A1,A2), one midfielder (A4) and one defender (A3), while the homogenous team presents three midfielders (B1,B3,B4) and one attacker (B2). This difference is explained by the presence of a defender (A3) in the heterogeneous team. A3 and A4 allows this team two play with two attackers by regaining control of the ball in midfield and passing this ball to these players, this also allows a better defence against the homogenous team, which presents a team with slightly better attackers. The score margin has also stabilised near 0.

In general, fitness values are better for the robots filling the roles of goalkeeper and midfield-defender. The number of goals scored during the game shows that the attacking actions are frequent; this favours the role of goalie with respect to the fitness value, if this player is accomplishing its task satisfactorily. For the role of midfield defender, this is explained by the importance of this role in midfield, where most part of the game takes place.

5.2.4 Adaptation

According to [Belew 1996]:

"Adaptation is not only the capacity to change but also the additional requirement that this change represents an improvement"

In this domain an adaptation process may be the consequence of two different events, a change in the skills of the opponents, usually an improvement, as a result of a learning process. On the other hand, it may be also the result of a mechanical, electrical... problem in one of the robots. This problem may have two different results; either the robot cannot keep moving, or it can continue to participate in the game, but their skills are different and the robot performance has deteriorated. Later in this chapter, several situations are analysed. The process of adaptation consists mainly of three phases. At the end of this process the team has a new configuration and the performance of the team has in general improved.

Phase 1. Fitness and preferences change

In order to keep their current preferences, agents need to obtain continuously rewards. If a change in the environment occurs, agents may not be able to keep these preferences, because the new environment may change the amount of rewards they were obtaining. For example, if the opponent improves its strategy, and the team is playing with three attackers, these players will have problems to obtain rewards, as a robot playing in midfield will have more problems to reach the attacking positions. This kind of situations gives rise to a change in agent's preferences. In this process the function ***scalp*** plays a very important role. Since all preferences are updated simultaneously, when rewards decrease for the role that this player is using most of the time, then the preferences for other roles tend to increase simultaneously. If the fall in rewards is considerable, as a result of a change in the environment, then preferences will tend to converge towards the same value. This will allow agents to explore other roles, and thus adapt to the environment by selecting a more suitable role for the current conditions. This process offers some advantages with respect to other possible methods, for instance reset values for all agents, as it allows adaptation of some of the members of the team, while other robots continue to carry out the same tasks. This helps to keep the team working quite satisfactorily while adapting at the same time. Changes in the environment also affect the fitness of agents, as the amount of rewards they obtain also change. The presence of fitness helps in three different ways:

- Speeding up the process. An important fall in the fitness of a given player results in a rapid decrease in the preferences of this agent. This helps to start the exploration much faster as the absolute difference in preferences are smaller than for average values of fitness.

- Replacing damaged robots. Damaged robots see how their fitness and preferences fall considerably. This allows other players to fill the role of this player, if it represents a key position in the team formation, by increasing the preferences for all roles, as a result of an increase in fitness.

- Increasing the fitness of other players. In an environmental change there are some players that see their fitness increase, especially as a result of its position in the field. This helps these players to increase its dominance in the team. This can lead these robots use partly and temporarily other roles, which should be filled by other players as a result of the new environment, while the other players start to adapt and explore the environment again. (more details on fitness later in this chapter).

Phase 2. New roles Selection

Once preferences tend to balance, agents start exploring other roles, this process occurs in two different ways, depending on the fitness of each agent. For low values of fitness this process starts by exploring most of the unused roles. For values slightly under the average, the exploration usually consists in using the roles adjacent to the one used previously.

Phase 3. Consolidation

This process finishes in a similar way as in the one described in the learning processes. (See previous example).

This example illustrates the process of adaptation when one of the robots breaks down. In this example, the robot that breaks down is the one filling the role of striker and it is of type 1 (0.45 m/s). As a result of the mechanical problem, the robot is still able to play, but it can only move at a speed of 0.2 m/s and a turning velocity of 4.28 rad/s.

A1			A2			A3			A4			A5		
ST	86%	25.0	LM	95%	25.0	RM	92%	25.0	MD	98%	25.0	GL	100%	25.0
LW	16%	17.0												
0.2			0.2			0.2			0.2			0.2		
30-0			0											

This is the starting team formation. With this team formation, and before robot A1 breaks down, this team is able to win BrianTeam by a goal margin of 75-80 goals. Right after this event, the same team is only able to win by a margin of 30-35 goals. After all the adapting process has been completed, the team is able to win by a goal margin of 70-75 goals. The system adaptation is analysed in periods of 90 steps.

A1			A2			A3			A4			A5		
ST	86%	20.3	LM	95%	21.3	RM	92%	21.1	MD	98%	24.7	GL	100%	23.0
LW	7%	17.8												
0.1971			0.1993			0.1991			0.2043			0.2001		
35-1			90											

After 90 steps, the team has started to evolve. The most important changes can be perceived in A1 and A4. A1 sees decrease its fitness, as a result of its mechanical problems. Their preferences also have started to change, even though it is still filling the role of striker, the preference for this role has decreased, as it can not get enough rewards to keep the same preferences as before. A4 see its fitness increase as a result of the new skills of A1. Since A1 moves slowly and has more problems to control the ball, this result in an increase of activity for the player that plays behind it, this is the case for A4. The other players continue to use the same roles, although their preferences also start to change. The score differential margin is 34, if may be seen that team performance after the robot’s breakdown is related to problems in offense. As the result shows, the opponent has only been able to score one goal.

A1			A2			A3			A4			A5		
ST	86%	19.0	LM	70%	20.6	RM	91%	20.7	MD	93%	23.5	GL	100%	23.0
LW	5%	17.7	LW	28%	20.6									
0.1944			0.2008			0.1997			0.2073			0.1979		
40-0			180											

After 180 steps, the team continues to evolve. A1’s preferences continue to balance, and its fitness decrease. As in last period, A4 see its fitness value increase for the same reason as before and the preferences for the role of MD decrease. However, the changes for A4 cannot be considered very important. The most important changes during this last period concerns A2. This player has seen its preference for the role of LW increase, because it is possible for another player to fill an attacking role as a result of the position of the damaged robot. The score margin over this period has improved slightly, even though this change cannot be considered significant.

A1			A2			A3			A4			A5		
ST	81%	18.1	LW	75%	21.3	RM	82%	20.1	MD	88%	24.8	GL	100%	24.0
LM	8%	17.4	LM	22%	21.1				FB	5%	20.3			
0.1915			0.2017			0.2008			0.2095			0.1995		
59-1			270											

After 270 steps, the system evolves in the same direction followed from the very beginning; A1’s fitness and preference for the role ST are decreasing. On the other hand, A2 is using the role of left-winger around 75% of the time. This results in a greater number of goals, increasing from 40 goals to 59. At this point in the game, A2’s preference for the role of ST, the one that A1 is currently using, is 18.2. This means that A2’s preference for this role is greater than A1’s one. This is explained by the different fitness values for each robot. However, A2 is not using this role because its preferences for other roles are much higher.

A1			A2			A3			A4			A5		
ST	69%	17.5	LW	72%	21.5	RM	72%	20.8	MD	90%	26.6	GL	100%	23.0
LW	16%	17.2	LM	14%	21.4	RW	11%	19.4	FB	6%	21.2			
			ST	8%	18.7									
0.1890			0.2022			0.2017			0.2119			0.1965		
66-1			360											

After 360 steps, A1’s preferences for roles are almost identical, with a slightly higher preference for the role of ST. The fitness for this agent has continued to decrease this value. A2 and A3 have started to compete for the attacking roles. Their preferences for the role of ST continue to be higher than A1’s preference. For A2 this situation gives rise to start using the role of ST, and at the same time A2 sees its fitness increase, while A3 continues to fill the roles of midfielder, although it uses from time to time attacking roles. A4 remains in the same position, although with a high value of fitness. The score margin shows that the scored goals have slightly increased due to the presence of better-fit players filling attacking roles.

A1			A2			A3			A4			A5		
ST	44%	17.0	LW	82%	21.8	RM	86%	20.8	MD	76%	27.0	GL	100%	23.0
RD	14%	17.0	ST	15%	20.1	RW	7%	19.7	LM	16%	20.7			
0.1871			0.2026			0.2009			0.2133			0.1961		
76-1			450											

After 450 steps, the team formation is undergoing an important change. At this point in the simulation A1's preferences are almost alike, it has started a process of exploring other roles. The ST role is only chosen by this agent 44% of the times. On the other hand A2 its increasing the usage of ST. The difference between the roles of LW and ST has decreased dramatically over the previous period. The other agents fill approximately the same roles as before, with the exception of the role A4, as it partly fills the role that was filling before A2. The fitness for each agent has stabilised, with the exception of A1, whose value is still falling. The score shows that this change in the offense has resulted in a wider goal margin.

A1			A2			A3			A4			A5		
LM MD	61% 26%	17.0 17.0	ST LW	84% 13%	22.0 20.8	RM RW	88% 6%	20.8 20.4	MD LM	74% 17%	27.0 20.7	GL	100%	23.0
0.1863			0.2021			0.2016			0.2142			0.1950		
77-1			540											

After 540 steps, the system reaches its final configuration. Finally A2 fills the role of ST, while A1 fills the role initially used by A2, together with A4. A1 also uses the role MD 26% of the times. This can be explained by the preferences of A4, although it is not the most preferred role by this agent, its high fitness allows this player to fill partly in this role. A3 continues to play in midfield in the role of RM. The role of goalie, A5, is not affected by the changes in the team.

The simulation shows that this formation only presents small oscillations over the time and in the role usage. Also the score stabilises around a goal margin of 75-80.

5.3 Parameter Analysis

5.3.1 Introduction

The model defined in this thesis consists of several parameters. This section addresses the problem of setting the values for these parameters, from the point of view of algorithm convergence and team adaptability. Some parameters can be adjusted using the knowledge the designer has on the system; others need some testing and tuning. Most of them affect the speed of the convergence of the system.

Usually high rates of learning may result in sub optimal solutions, and bad team performances, as agents tend to exploit rapidly the knowledge they have on the environment. On the other hand, slow rates can guarantee a better result, as agents are able to spend more time exploring the environment and thus having more time to know which is the role to select. However, this process can also get stuck in local minima. This problem is common to many learning algorithms, and the solution is usually a compromise between exploration and exploitation.

At the end of this section the adjusted values of the parameters for this thesis are detailed.

5.3.2 Necessity and Prestige

Necessity and Prestige are related to agent's preferences and team decision-making. These two parameters condition the converge process and the level of excitation in the system.

Necessity

As it has been already mentioned, one of the main questions that has to be addressed is the problem of exploration vs exploitation. In the problem studied in this thesis, if agents have little time to explore the environment, they will settle into the first role from which they get an important amount of rewards. On the contrary, if they have more time, they will be able to figure out better, which are the best roles for them. This process is determined by the parameter

a and necessity. a affects the rate at which agent revaluate their preferences, this parameter is analysed later.

Necessity is a parameter introduced by [de la Rosa 1993], in addition to a usual confidence parameter (in this work known as Prestige), to revise the certainty of several sources of information. Results showed that the introduction of this parameter improved the system performance. In this work Necessity is used in a similar way as the Botzmann exploration strategy, widely used in the field of reinforcement learning. This strategy allows agents to choose actions probabilistically.

Necessity aims at helping agents to explore all existing possibilities, in this case roles. When agent preferences are very similar for all roles, necessity increases the likeliness of using a role that has not been previously used. This parameter helps agents to explore all roles before they settle into one role. One of the features of this parameter is that it is present at each decision step during the game. This is one important advantage as the effects of this parameter can help the agent to adapt when the preferences for each role tend to balance again. These features pose a problem, choosing too high values of Necessity prevents agents from converging, as they will continuously explore the environment or it will slow down the convergence process. It must be tuned, so that agents explore the environment when the preferences are very similar, and when the differences among the preferences reach a given value the effects of this parameter must start to disappear. This value has to be also related to the possible values of Prestige, as Necessity affects the revision process on the certainties of each action how prestige.

Prestige

Prestige is tightly related to agent’s preferences. They are related by an exponential function, whose gradient is one of the parameters that has to be selected. This gradient affects the process of convergence and adaptability. A steep gradient helps the system to converge by making agents settle into the roles, for which they have a slightly greater preference. However, this may not give enough time for agents to explore the environment and may prevent agents from adapting, because preferences must be in this case almost identical, before an agent starts again to explore the environment. If the gradient is rather smooth, an agent has more time to explore the environment, as the preference values needed to settle into one role are higher. On the other hand, the latter option may hinder the convergence, because agents may need high values of preferences for settling into one role.

Variables that govern these two parameters can be easily adjusted by testing and observing how the system evolves for different values of them. It is easier to set a value for the variables that link the preferences and prestige, which strikes a balance between exploration, convergence and adaptation. Once this has been done, Necessity can be adjusted, so that this parameter affects the whole process up to a certain value of prestige.

5.3.3 Function *sclp*

This function replaces the process presented in the original work [Hogg and Huberman 1991] by means of the function erf. It aims at linking pay-offs and interest for roles/tasks/... . This function, as it is originally defined in [Hogg and Huberman 1991], is only prepared to deal with two resources. The new function ($sclp$) created for this thesis attempts to include all the interesting features of this function but allowing at the same time the use of more resources. This function has two main goals, the first one is to normalise the values of rewards with respect to agent preferences, and the second and most interesting one, to evaluate rewards obtained by an agent and consider whether they are significant. This second case is of a great importance for the adaptability of the team. The value selected as a threshold affects the convergence speed of the system.

sclp helps in two different ways:

- The reason for using such a function can be easily understood by means of an example. A given agent gets for using a role r_1 a reward of 0.001 and for using r_2 a reward of 0.01 over a period of t_w seconds. Usually agents can get up to 5 in rewards when an agent is performing well when filling a given role. Using the present rewards an agent can conclude that it is 10 times better to use r_1 than r_2, even though normal rewards can be around 5. If it makes use of this function, and the threshold is set at 0.5, it will conclude that both roles are identical in terms of preferences. This function represents an improvement in the process of choosing among several possibilities.

- In the second case, the threshold helps in two different situations, in the process of learning and in the process of adapting to the opponent. In both cases this process works very similarly. A change in the environment may affect the rewards some agents get when making use of some roles. This threshold balances the preferences, as the rewards decrease, thus if the threshold is correctly defined, then small number of rewards is not considered significant. This allows agents to explore again the environment some after the change.

This process is not only applied to compute the preferences of each agent, but also to compute the fitness. This latter case is not studied specifically in this analysis.

The value of the threshold affects the adaptability and convergence of the system. An excessively low value for the threshold can considerably help the convergence process, as all rewards are considered significant. On the contrary, this can slow down or prevent the team from adapting, as agents may consider significant small amounts of rewards, thus keeping the agent’s preferences. As in the previous case a compromise has to be found between the two aspects. The function slope (after the threshold) is related to the speed in the increase in agents’ preferences, thus affecting the convergence speed. Steep slopes may give rise to rapid increase in agent preferences and thus limiting the time to explore the system, while smooth slopes may slow down this process

5.3.4 Rewards

Rewards do not affect the process of convergence or adaptability, as rewards are rescaled by the function *sclp*. However, when adjusting the parameters that govern this function, the amount of possible rewards must be included.

When setting the values for rewards two aspects must be considered:

- Goalie and field players are evaluated in two different ways, this does not allow comparing easily the rewards for each role. As the role of goalie is one of the most key roles in the team, it must be filled by a player that presents good skills for this position. For this reason, the rewards an agent can get for using this role should be higher than for field roles. However, in order to prevent to give many rewards to agents not fitted for this role, in case the opponent scores a goal, these rewards should be considerably lower.

- One of the parameters that have to be adjusted in the rewards is the maximum time elapsed after a player touched the ball for the last time and the play resulted in a goal (agents obtain also rewards depending on the elapsed after a goal was scored). This parameter can be easily tuned experimentally by estimating the time that the ball may need to move from the area covered by a role to the goal, and increasing this time by a factor.

5.3.5 Learning from the score

Learning from the score allows to reward agents as a team, depending on how well they are globally performing. One of the main problems is finding which are the best values that ensure good results. Results show that this learning feature adds robustness to the process of self-organization. Some experiments showed that this parameter allowed to vary rewards by ±50% (considering different rewards for all roles with the exception of the goalie), while the learning process tended to converge to similar team formations.

The system is rewarded depending on current and the weighted average of the past performances of the team. This performance is measured in this case by the score difference. The most important aspect of this learning feature is to determine how important are the past results. There are two situations that can exemplify the importance of this aspect:

- Only very recent results are considered ($b_1 \approx b_2$). This allows to punish/reward those agents immediately after the team performance improves or deteriorates. On the other hand, this strategy can slow down the process of adaptability, because the score margin used as a reference may indicate quite fast the current team performance, not allowing to punish agents after a change in the environment.

- Past results are considered ($b_1 << b_2$). This helps the system to adapt as the present team performance can be compared over a long period of time with the one previous to the environmental change. This allows to punish all the team. On the other hand, this strategy may reward some agents when the team is performing worse than in previous steps, as the average is computed using old information on the score.

Like in other parameters, it is necessary to strike a balance between team adaptability and team reaction to changes in the environment. It is also possible to use two score averages, the first one including the recent team performance, and the second one including part of the previous scores. In order to compare the past performance of the team with respect to the present one, the index that presents the greatest value should be selected. This strategy allows to combine both aspects.

5.3.6 a g

These two parameters govern the rate at which agents revaluate their preferences and fitness.

a Determines how fast agents revaluate their preferences. High values of a leads the system to a fast convergence, but agent may not have enough time to explore the environment. On the other hand low values leads the system to a very slow convergence, but agents then may have enough time to interact with the environment.

g determines how fast agent’s fitness evolves over time. This parameter is not so critical as a, because a conditions how changes in fitness affect preferences. In order to help the team to adapt, g should be several times greater than a. However, high values of g may give rise to sharp oscillations in the fitness and preferences of each agent, hindering the convergence and team adaptability.

a should be tuned striking a balance between convergence and exploration. Once this value is determined g should be several times (5-10) the value of a, but always bearing in mind the problems that can arise out of high values of g.

5.3.7 m

This parameter decreases the reward estimation depending on the information available for its computation. This parameter plays a very important role in team adaptability, but it may also cause undesirable effects in the convergence. As roles overlap, agents tend to obtain rewards for several roles, even though the role that they are filling is the one that provides them with more rewards. If a change in the environment occurs, some agents may start to obtain more rewards from these close roles, especially as the game may be played in a different place on the field. If this function considers significant a small role usage, this can help to increase preferences for other close roles and thus helping the team to adapt. If this function is extremely conservative, this change in the environment may not materialise in a change in rewards, slowing down the whole process. On the other hand, if this function considers as significant very small role usages, agents can start to oscillate among several roles. This function must be carefully selected, so that the system is able to adapt, but preventing oscillation between several roles.

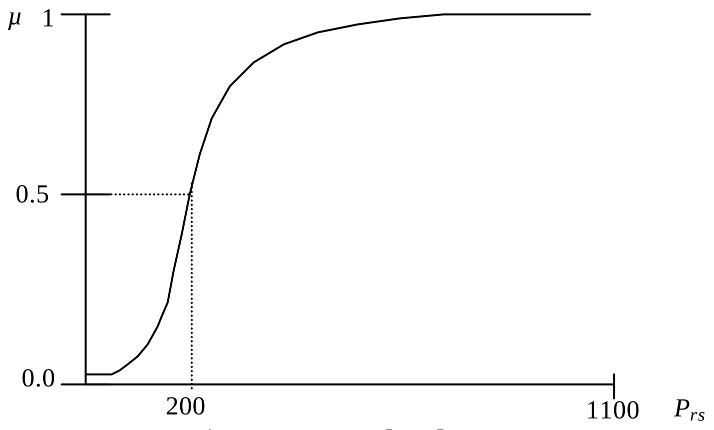


Figure 5.2 μ and Role Usage

These parameters that govern this function may be tuned based on the experience. According to the experience of the designer, making estimations of rewards with a value of P_{rs} below 100 can be considered as no significant. With a value of over 300 the estimation can be considered as significant for making estimations. This is how the parameters of this function are defined.

5.3.8 Parameter Values

The following table displays the values for some of the parameters of the model, only the most important ones are included.

Concept	Parameter	Value
Model	α	0.075
	γ	0.3
	tw	100 seconds
sclp	*threshold*	0.5
	V_1	100
Rewards	*Str*	1.5
	Ltr	1.5
	Rr (goalie)	3.5
	n_1	10
Learning from Score	b_1	0.8
	b_2	0.2
Prestige	k_6	0.008
Necessity	N_{max}	0.30
	k_7	0.006
Parameter m	k_1	65
	k_2	0.01

Table 5.2 Parameter Values

5.4 Team Adaptability and Fitness

5.4.1 Introduction

This section studies in depth one of the main features of this model, the adaptability to changes in the environment. This process has been already introduced in a previous section. This section studies how the system evolves for different types of changes in the environment. This process is also analysed from the point of view of the fitness of each agents. Results show that the presence of this parameter in the system can help the process of adaptability.

Changes may be due to two different aspects, a change in the opponent or a in a robot's individual skills, as a result of a mechanical problem. This section describes two scenarios of each type and studies the impact of the fitness parameter in the adaptability. At the end of this section a point addresses the issue of adaptability in homogeneous and heterogeneous teams from a general point of view

Fitness gives a measure of how well each agent is performing in the system with respect to other teammates. High values of fitness represent an agent that is performing much better than other teammates. This affects the system in two different ways:

- Player with high values of fitness tends to win conflicts with other teammates, when they decide to choose the same action. In other words, this means that these players are dominant with respect to others.

- Players with high values of fitness tend to extend its dominance and make also use of other roles, more often than other team members. However, this depends also on how the roles have been defined. In some cases team design can promote agents to interact with other roles.

In this section fitness is represented by the slope that defines the function ($sclp$) that links the rewards with the fitness of each agent. Steep slopes lead to high values of fitness for the best agents in the team. The experiments of this section have been analysed for three different cases, fitness is not considered (∞), a medium value (1/100) and a steep slope (1/50). All the examples analysed where fitness is included are computed using a slope of 1/100 and a threshold of 0.5.

In the following examples the Necessity parameter is set at 0, this means that the decision-making of each agent is not affected by this parameter. This helps to show that team adaptability is possible with this model, without the need of any source of excitation.

5.4.2 Change in the Opponent I

This case studies the adaptability of a team to changes in the environment as a result of a rapid deterioration in the opponent performance. This is not a very common case, but it can help to illustrate the adaptability of a team using this model.

The starting formation for this experiment is extremely defensive, and consists of a goalie (A5), three defenders, a fullback (A4), a left and right defender (A1,A2) and a midfielder (A5). This is tested against BrianTeam. Under these circumstances the team is able to win by a goal margin of 15 goals. Most of these goals are scored by the opponent in its own goal, as there are no offensive players.

A1			A2			A3			A4			A5		
LD	89%	25.0	RD	85%	25.0	MD	100%	25.0	FB	99%	25.0	GL	100%	25.0
0.2			0.2			0.2			0.2			0.2		
Approx 15-0			0											

The first case analysed is when the fitness of each agent is not included.

A1			A2			A3			A4			A5		
LD	76%	20.9	RD	79%	20.1	MD	99%	22.3	FB	79%	18.0	GL	100%	25.0
LM	22%	20.3	RM	16%	19.5				RM	8%	18.0			
0.2			0.2			0.2			0.2			0.2		
27-0			75											

After 75 steps, the players' preferences show that the team is adapting, as players start filling other roles. While preferences for the initial roles are decreasing, those are increasing for some of the attacking roles. A1 and A2 are using roles adjacent to their initial positions (LD⇒LM and RD⇒RD). A4 is still playing as a fullback; but its preferences have decreased considerably. This leads this player to start to explore and use other roles. In this case, it cannot start this process by using close roles, as the role of MD is filled by A3, and these player preferences for this role have barely changed. The scores shows an improvement in team performance as a result of the new roles used.

30 steps later the team reaches its final team formation.

A1			A2			A3			A4			A5		
LM	97%	20.1	RM	83%	18.8	MD	98%	22.3	LW	78%	19.5	GL	100%	25.0
									ST	20%	19.4			
0.2			0.2			0.2			0.2			0.2		
84-0			300											

After 300 steps, players have fully converged to their final roles. A1 and A2 play as a LM and RM, the roles closer to their starting positions. A4 finds a new role, as the role of MD is already used by another player, and fills the role of LW, an attacking position, and partly the role of ST. Finally A3 continues to fill the same role as in the beginning. The role of MD is a key position in the team formation when playing against Brian Team. Team performance shows a very important improvement, the score margin has jumped from 27 to 84. This follows from the offensive team configuration that team has reached.

The same situation is analysed in a team where fitness is considered.

A1			A2			A3			A4			A5		
LD	79%	20.5	RD	80%	19.7	MD	89%	22.3	FB	79%	17.9	GL	100%	25.0
LM	19%	20.4	RM	15%	18.3	CF	11%	18.3	MD	6%	18.0			
0.2			0.1990			0.2020			0.1973			0.2016		
58-0			75											

After 75 steps, the team shows the same team formation as in the previous case. The preferences and roles used are very similar (in percentage terms), with the exception of A3. Its high value of fitness (0.2020), allows this player to fill other roles as CF, an attacking position. This results in a rapid improvement in team performance. (58 goals vs 27 without fitness). On the other hand, A4 shows a very low value of fitness, as its position in the field, playing in the role of FB does not allow it to obtain many rewards.

A1			A2			A3			A4			A5		
LM	90%	21.4	RD	66%	18.2	MD	79%	23.9	RW	70%	17.7	GL	100%	22.6
			MD	11%	17.9	RM	16%	19.6	ST	11%	17.6			
0.2026			0.1986			0.2076			0.1936			0.1980		
72-0			150											

After 150 steps, the team evolves in a way that the final team formation will differ from the case where the fitness is not considered. These differences are related to A2 and A4. A2 does not continue to use the RM role, its adjacent role. This role is partly filled by A3, as its fitness is increasing. A4 has started to use also offensive roles; but its preference for these roles are lower than those for A2 (18.0 vs 17.6), as a result of the fitness values. This allows later A2 to fill these roles, as it is able fitted to complete the tasks associated with this role. This can be observed in the next step analysed.

The goal margin shows that the team performance is still improving, increasing from 58 to 72.

A1			A2			A3			A4			A5		
LM LW	87% 9%	22.7 21.1	ST RW	80% 17%	19.3 19.1	MD RM	77% 16%	25.5 20.8	RM MD	77% 10%	17.3 17.2	GL	100%	19.0
0.2065			0.1987			0.2131			0.1891			0.1926		
86-0			300											

After 300 steps, player's preferences have converged to their final values. As it has been already mentioned before, A2 fills the attacking positions (ST,RW), instead of A4, since its fitness is higher than in A4. A4 fills the role that in the first example is filled by A2. A1 and A3 fill the same positions as in the previous case, but their high value of fitness allows to use other roles without giving rise to conflicts. The final team performance shows a very similar result to the one observed in the first example, where fitness is not considered, but globally the system now adapts much faster.

The following table displays the information on the team performance for different values of fitness. The team performance is evaluated at the step 75. This information gives an idea of how fast the team is adapting to the environment. 35 simulations have been performed for each case. Fitness is represented by the slope that defines the function *sclp*.

Fitness	Score Margin Mean	Variance
∞	3.56	1.01
1/100	4.37	1.02
1/50	4.62	1.61

Table 5.3 Fitness and Score Margin

The team tends to improve performance faster as the slope of the function *sclp* increases. This also results in a greater dominance of robots with high fitness. The mean difference between the two cases, where the fitness is/is not included is statically significant. While the difference between the two cases where fitness is included, the test shows that the difference is not significant, although this value is greater for the case where the fitness is better considered. Results show also, although this information is not displayed here, that the possibility of other role distribution also increases with the fitness. This allows to try out several possible role distributions in the team that may result in a better team performance, as the new environment may not require the same type of agents for the some of the roles. In the case, where fitness is not included, the team evolves always towards the same role distribution, as the one shown in the example analysed.

5.4.3 Change in the Opponent II

This example analyses the process of adaptation when the opponent improves their team strategy, for example as a result of a learning process. The team that adapts is supposed to have been playing against a very easy team, before this change in the environment occurs. The opponent selected for this experiment in SchemaDemo.This team presents better attacking skills than BrianTeam, this allows to play against a team that represent an improvement in team skills.

The starting team formation is composed of a goalie, a center forward, a striker and a right/left winger.

A1			A2			A3			A4			A5		
RW	100%	25.0	LW	100%	25.0	ST	99%	25.0	CF	99%	25.0	GL	100%	25.0
0.2			0.2			0.2			0.2			0.2		
4-50			0											

After the change in the environment the team presents a very poor global team performance. It is only able to score 4 goals, but the opponent scores 50 goals. The first case studies this process for a team where the fitness is not considered.

A1			A2			A3			A4			A5		
RW	92%	18.6	LW	88%	18.5	ST	90%	18.6	CF	83%	18.7	GL	100%	25.0
									MD	12%	17.9			
0.2			0.2			0.2			0.2			0.2		
14-40			90											

After 90 steps, agent's preferences tend to decrease, with the exception of A4 that has started to use the role of MD. All players continue to fill their initial roles, although they see their preferences increase for those roles close to their positions in the field. The score margin presents a very poor team performance, although the score margin has improved.

A1			A2			A3			A4			A5		
RM	92%	18.6	LW	82%	18.6	MD	96%	19.3	LM	91%	18.6	GL	100%	27.
			ST	10%	18.0									
0.2			0.2			0.2			0.2			0.2		
37-22			240											

After 240 steps, the team presents its final formation. A4 finally plays as LM, although it seemed that it was going to fill the role of MD. This role is finally filled by A3, which has seen their preferences for this role increase much faster than A4's ones. A1 has moved to a close role, a midfield role from RW to RM, and plays now as A1, while A2 still plays in its team position, even though it is also acting as ST, 10% of times. This change in team formation results in an improvement in team performance. Now the score margin is positive and it shows a clear improvement in offensive performance.

The case where fitness is included presents some changes with respect to this first example.

A1			A2			A3			A4			A5		
RW	88%	18.8	LW	94%	19.0	ST	78%	19.0	CF	59%	18.6	GL	100%	25.0
						LM	9%	18.2	MD	29%	20.2			
						RM	7%	18.3						
0.1957			0.1953			0.1991			0.1998			0.21		
6-44			240											

After 90 steps all players have seen their preferences decrease as a result of the environmental change. At this point in the game there are two players that have started to explore other roles. These players are A3 and A4. A4 has increased its preference for the role MD considerably. This role is a key component in a normal team formation. On the other hand A3 is also increasing its preferences for midfield roles. Due to the problems that most players have to get rewards, as the ball rarely reaches the attacking positions, agents have seen their fitness fall, with the exception of A5. This is the only player that is contributing somehow to the team. The goal margin shows a slight improvement in team performance, but the team performance is still very poor.

A1			A2			A3			A4			A5		
RW	94%	19.5	LW ST	87% 10%	20.0 18.6	RM	93%	20.2	MD LM	83% 11%	30.0 20.6	GL	100%	25.0
0.1911			0.1925			0.1965			0.2143			0.2056		
37-17			240											

After 240 steps the team presents a new formation, which remains stable over time. A4 is finally filling the role of MD, the one that this player has started to use in previous evaluation steps. In addition, it also plays the role of left midfielder. This has two effects; A3 finally acts as RM, after exploring RM and LM. The new position in the field of A3 and A4, filling the midfield roles, allows A1 and A2 to continue to play in offensive positions. A3 and A4, as a result of this change in roles, have seen their fitness increase, specially for A4 that becomes the most dominant team member, while the rest of the team see their fitness decrease as a result of their positions in the field. This new formation improves the score margin, and allows the team to win by a goal margin of 20 goals.

Unlike the previous example, in this case the fitness value does not affect the speed of the process, but it introduces a new interesting feature. For the case where the fitness is not included the final role distribution presents, in most cases, a very similar to the one observed in the example detailed previously. In this case, only two players usually tend to fill the role of MD. This situation changes for the case where fitness is included and the slope is 1/50. As the final role distribution for MD shows, this value allows almost all team agents to explore and (one of them) settle into this role. This feature is very important in adaptation in heterogeneous teams, as it is explained in chapter 7, where homogeneous and heterogeneous teams are compared. The best-fitted player for a given role changes depending on the opponent. When the environment changes, some players tend to adapt by filling close roles. In some situations, these players are not the best fitted ones for these roles, but they are the first to play in these positions because of their previous position in the field.

Player	∞	1/100	1/50
A1	11%	15%	19%
A2	11%	1%	26%
A3	45%	48%	17%
A4	33%	36%	38%

Table 5.4 Fitness and Role Distribution for MD

Due to the high learning rate, the system is not able to benefit from this feature and the performance after 300 steps is better for the case where fitness is not included.

5.4.4 Goalie Malfunction

This example analyses how the team evolves, when the player filling the role of goalie breaks down. The first case studies how the system evolves when fitness is identical for each player and do not change over time. This case is evaluated against SchemaDemo.

All experiments start from the following initial state. The team is composed of a goalie, a midfield defender, a left and right midfielder and a left-winger.

A1			A2			A3			A4			A5		
GL	100%	30.0	LW	99%	30.0	LM	100%	30.0	MD	100%	30.0	RM	100%	30.0
0.2			0.2			0.2			0.2			0.2		
Approx 13-36			0											

In this example, A1 has a mechanical problem and as a result of it, it can only move at 15 cm/s (from 45 cm/s). Although it is moving slower, it can still participate in the game filling the role of goalie. The score shows that after the mechanical problem, the team loses by a margin of 13-35.
When the fitness parameter is not considered, the team is not able to fully adapt.

A1			A2			A3			A4			A5		
GL	100%	19.9	LW	90%	20.8	LM	94%	21.0	MD	99%	22.9	RM	97%	22.0
0.2			0.2			0.2			0.2			0.2		
20-32			150											

After 150 steps, the team formation has not changed. Agent's preferences for the roles they are using tend to decrease, but this does not result in a change in agent preferences. This process can be perceived especially in A1, as a result of its mechanical problems. The score margin has slightly improved, even though this change may be explained by the domain, as soccer presents highly stochastic environments.

A1			A2			A3			A4			A5		
GL	100%	18.8	LW	94%	19.7	LM	96%	20.5	MD	97%	22.5	RM	95%	20.7
0.2			0.2			0.2			0.2			0.2		
14-38			300											

After 180 steps, the team formation remains stable, although preferences for the currently used roles in general tend to decrease. The analysis of additional data shows that these values stabilise around the current values, presenting small oscillations. The score margin has increased again, and it is similar to the one found at the beginning of the simulation.

When the fitness is introduced, the team is able to adapt. The following example illustrates this process.

A1			A2			A3			A4			A5		
GL	100%	18.6	LW	86%	18.1	LM	90%	19.0	MD	98%	24.2	RM	85%	19.0
0.1968			0.1968			0.1993			0.2067			0.2004		
61-63			120											

After 120 steps, players' preferences decrease, with the exception of A4 whose preferences have barely changed. The most important changed can be perceived in the fitness of each agent. While for A3 and A5, these values change slightly, A1 and A2 see their fitness decrease. For A1, this follows from the physical change that its body has experienced and for A2 from the role that it is filling. Most of the work of the team focuses on defensive and midfield tasks. This is reflected in A4, with a high value of fitness. The score shows that the number of goals scored by the opponent is slightly greater than the number of goals scored by the learning team. Studying other simulations show that this value tends to range from –20 to 0 at this point in the game, showing a great variability.

A1			A2			A3			A4			A5		
GL RD	80% 10%	17.9 17.7	LW	73%	17.8	LM	82%	19.0	MD GL	78% 15%	23.4 19.3	RW GL	74% 5%	18.2 18.1
0.1962			0.1962			0.1994			0.2086			0.1997		
17-24			150											

After 150 steps, the fitness for A1 and its preference for the role GL is very low. This enables other team members to fill the role of goalie, as their preferences for this role are higher. This leads A1 to start to interact with other roles. This process can be explained by the different fitness of each agent that allows agents with high fitness values to increase the preferences for all roles. This is the case for A4 (18.1>17.9) and A5 (19.3>17.9), even though in the case of A4 its preference is much higher for the role of MD (23.4). On the other hand, A2 has also seen an important decrease in its fitness values and preferences. Although it is still playing as LW, this percentage is falling. The same process can be observed in A3. The score margin shows deterioration in team performance, as a result of the constant changes in the position of goalie.

A few steps later A5 finally fills the role of goalie, even though A4 has a greater preference for this role. The reason why A4 is not filling this role is that this agent has a much greater preference for the role of MD (23.4>>19.3). Some simulations show that A4 is the player that finally fills this role, this usually depends on the rewards each agent is able to obtain when filling this role.

A1			A2			A3			A4			A5		
LD	42%	17.5	RD	70%	17.7	LW	70%	18.4	MD	75%	26.4	GL	100%	22.9
LM	41%	17.6				ST	20%	18.2	RM	14%	19.9			
0.1929			0.1937			0.1994			0.2125			0.2015		
27-20			210											

After 210 steps, A5 sees its fitness and preference for GL increase, instead of decreasing, as it was the case of A1.The rest of the team is exploring the environment as A5 left its role of right midfielder. This role is partly filled by A4, as a result of the increase in its fitness. A1, A2 and A3 are exploring other possible roles. The score margin is now positive as a result of the new goalkeeper.

A1			A2			A3			A4			A5		
LM	85%	17.3	RD	71%	17.7	ST	76%	18.3	MD	72%	28.9	GL	100%	24.2
									RM	19%	21.6			
0.1901			0.1925			0.1987			0.2174			0.2014		
32-23			300											

After 300 steps, all players have converges to new roles, after the major reorganisation. Finally, A3 is filling the role of ST. Experiments show that this player is very well fit for this task. A2 is playing in defence positions (RD) and helps A4 in midfield. A4 has seen increase considerably its fitness value and it is acting as MD, but near 20% of the times as right midfielder. A1 has settled into the role of left midfielder, it contributes better to the team in this role than acting as a goalie. The score shows a slight improvement in team performance, with respect to the period evaluated; even though this improvement may not be considered very significant.

This example has been run 25 times for the two different cases where fitness is included, and a third one, where this parameter is not considered. The system state is evaluated after 300 steps. Results displayed in table 5.5 show that the number of times the damaged goalie is replaced after 300 steps, increase with the fitness, while in the case where fitness is not considered A1 is never replaced.

Fitness	% Goalie Replacement
∞	0%
1/100	62%
1/50	85%

Table 5.5 Fitness and Goalie Replacement

Results also show that A1, when it is replaced by a different player in the role of GL, usually tends to fill the former position filled by this player. If the player acting, after this process, as a goalie was filling a position in the attacking line, a different player, and better fitted for these tasks, moves to this position, while A1 fills the role that the new attacker has left, tending to play in the same side of the field (right/left), but in more defensive positions.

One of the problems when replacing the goalie is to know if these changes will improve the team performance. This is difficult to know and it has not been addressed in this thesis, even though these examples show that when the goalie is replaced the team performance increases.

5.4.5 Striker Malfunction

This example analyses the process of team adaptability, when the player filling the role of striker is damaged. This example has already been analysed in the first section of this chapter to illustrate the process of adaptability. In that example, where fitness is included, the team is able to adapt by replacing player A1 in the role of striker. This section analyses this case for a team where fitness is not included.

All the experiments that analyse the adaptability after the breakdown in A1 start initially from the same team configuration. This team formation represents a typical team configuration when a team of fully heterogeneous players, play against BrianTeam, the team selected for this case.

A1			A2			A3			A4			A5		
ST LW	86% 16%	25.0 17.0	LM	95%	25.0	RM	92%	25.0	MD	98%	25.0	GL	100%	25.0
0.2			0.2			0.2			0.2			0.2		
39-0			0											

A1 acts as striker, although it can also use other close roles, as roles overlap. A2 and A3 fill the roles of right and left midfielders, while A4 act as midfield-defender and A5 a goalie.

A1			A2			A3			A4			A5		
ST	83%	19.8	LM	89%	20.1	RM	94%	21.3	MD	99%	23.0	GL	100%	27.0
0.2			0.2			0.2			0.2			0.2		
47-0			150											

After 150 steps, agents' preferences tend to decrease, tending towards the average values of preferences (0.01818). This is process is more noticeable in A1, the player that is damaged. The score shows a slight improvement in team performance, as A2 has also started to play, a very small percentage of times, as left-winger, helping A1 in attacking plays.

A1			A2			A3			A4			A5		
ST	84%	19.3	LW	73%	20.1	RM	93%	20.4	MD	97%	23.0	GL	100%	33.9
0.2			0.2			0.2			0.2			0.2		
48-1			300											

After 300 steps preferences have almost stabilise with the exception of A2, which is playing now as left winger, helping A1 in attacking tasks. The score margin has not changed from the previous period, where it improved with respect to the previous period analysed.

A1			A2			A3			A4			A5		
ST	81%	18.9	LW	84%	21.2	RM	91%	20.4	MD	97%	22.7	GL	100%	34.6
0.2			0.2			0.2			0.2			0.2		
52-1			450											

After 450 steps, the team's preferences have stabilised on the previous values, agent's preferences present only small changes. The score shows that the goal margin has slightly increased. However, this improvement cannot be considered significant.

Unlike the example where the fitness is included and A1 is replaced by another player in the role of striker, A2 is not able to replace A1; it only can fill another attacking role, and leaves the midfield with only two players. This situation leads to a lower performance of the team.

Fitness	% Striker Replacement	Mean	Variance
∞	0%	47.1	60
1/100	75%	75.2	43
1/50	75%	72.8	48

Table 5.6 Fitness and Striker Replacement

Table 5.6 shows the results for the different values considered for fitness, after repeating each case 20 times. When the fitness is not included, this player (A1) in never replaced in the role of ST and the team presents a very poor performance, when compared to the two cases where this parameter is included. In this latter case, both teams are able to adapt the same number of times after 300 steps. Team performance shows also an important and significant improvement with respect to the first case.

5.4.6 Fitness and Adaptability Problems

The previous examples have shown the benefits of using the concept of fitness in the adaptability of the team. But in some cases, a high dominance of some of the members of a team can prevent the whole team from adapting. The following case exemplifies this situation. The slope consider in this example is approximately 1/10.

A1			A2			A3			A4			A5		
RW	93%	25.0	LW	94%	25.0	ST	95%	25.0	MD	100%	25.0	GL	100%	25.0
0.2			0.2			0.2			0.2			0.2		
20-20			0											

This example represents a situation where the opponent skills improve considerably. Before this change in the opponent team, the team considered is playing against an easy opponent, playing with three attackers, one midfielder and a goalie. When this team is evaluated against the opponent, the score shows a draw between the two teams

A1			A2			A3			A4			A5		
RM	93%	20.3	LM	98%	21.0	ST LW	85% 10%	19.0 18.7	MD	94%	24.0	GL	100%	34.0
0.2			0.2			0.2			0.2			0.2		
31-10			240											

After 240 steps in the example of a team where the term fitness is not considered, the two robots playing in wings have moved back to midfield positions. This allows to control better the ball in midfield and to reach attacking positions with the ball controlled, helping A3 to score goals. The score margin, evaluated over past 10 evaluation periods, shows a margin of 21 goals. However, in the case where the fitness is considered the team shows a very different final team formation.

A1			A2			A3			A4			A5		
RW	94%	16.6	LW	96%	25.4	ST	90%	16.3	MD	68%	27.3	GL	100%	25.0
									LM	20%	23.1			
									RM	5%	22.2			
0.1685			0.1685			0.1689			0.2409			0.2534		
20-17			0											

After 240 steps only two changes can be observed in the team formation. The important increase in the fitness values for A4 and A5. In A4 this increase in fitness results in the use of other roles. Roles that in the example where fitness is not considered are filled by A1 and A2. In this case A4 dominance prevents A1 and A2 from filling these positions, leading to a lower team performance, as the score shows. The score margin is only 3 goals.

5.4.7 Heterogeneous vs. Homogeneous Teams.

All the experiments in this section have been conducted using a team of heterogeneous robots. The same experiments have been repeated for a homogenous team, and results turn out to be very similar. However, in particular situations some differences can be observed. In the example of a damaged striker, a different player may initially replace this player. But as the system evolves, a different player that performs better in this role may replace this player later. On one hand, this increases the time needed to complete the adaptation process, on the other hand the team may see their improvement increase, as robot have reorganised to make a better use of their skills. This process has been observed in some particular cases, such as the damaged striker.

From a more general point of view heterogeneous teams tend to be more adaptive than homogeneous ones. The reason is that heterogeneous team present a diverse set of skills. When the environment changes, the skills needed for the different tasks may change. In the case of soccer this may mean, for example, that the type of agent that was filling the role of striker is not the best player after the change in the environment, as the skills required for each role may change with the environment or it can contribute better to the team performance filling other roles (This is analysed in-depth in the chapter where homogeneous and heterogeneous teams are analysed). In this situation a team of fully homogeneous robots it can do nothing to solve this problem. On the other hand, the heterogeneous team can try out several approaches to adapt to the new environment and make a better use of the team skills. However, this process may take longer, as the number of different solutions that ca be tested increase with the number of diverse components.

5.5 Dominance Hierarchy and Performance

5.5.1 Introduction

This section studies the impact of agent's fitness on team performance. The previous section has studied how this parameter affects the adaptability of the whole system. As it has been already studied agents with higher values of fitness tend to use other close roles, and usually tend to win more conflicts when two or more players are taking conflictive actions. This section focuses on two different aspects, on one hand it analyses how the team performance changes depending on the fitness, on the other hand it studies by means of two different examples, where fitness plays an important role, how this parameter may improve or deteriorate team performance. The goalie is excluded from this analysis, since the player filling this role is not allowed to leave the penalty area.

Two different aspects, the role and the player condition fitness. There are roles that tend to participate more during the game as a result of its position on the field. This allows it to obtain more rewards than other players and increase its fitness. On the other hand the ability to obtain rewards, regardless of the role, depend on each player skills and how these skills are appropriate for the role filled.

5.5.2 Fitness Analysis

The following table describes an example where the fitness is not considered. This example is used to compare the team formation and the performance for different cases where the fitness is included. With respect to the following examples this team formation presents a small difference A3 tend to fill more often the role of ST than one of wingers (LW). This small variation does not affect significantly the performance of the whole team and serves well the purpose of this analysis. Numbers in brackets denote the number of conflicts (for instance two or more players decide to take the same action) won by the agent in percentage terms. With the exception of A5, the results show that there are players that tend to win more conflicts than others (A3), even though the fitness is the same, this value also depends on the position of the player in the team

A1			A2			A3			A4			A5		
MD	99%	22.5	RM	80%	19.6	ST	83%	20.2	LM	87%	19.8	GL	100%	29.9
			RW	10%	19.0	LW	13%	18.4	MD	5%	18.1			
0.2 (49%)			0.2 (47%)			0.2 (64%)			0.2 (23%)			0.2 (100%)		
49-4			240											

The first example analyses how the introduction of the concept of fitness may benefit the team performance.

A1			A2			A3			A4			A5		
MD	67%	27.3	RM	91%	18.6	LW	83%	19.3	LM	79%	18.0	GL	100%	29.2
RM	16%	22.1				ST	13%	18.3	MD	5%	17.7			
LM	8%	21.1							CF	5%	17.7			
CF	5%	18.8												
0.2208 (98%)			0.1922 (5%)			0.1968 (66%)			0.1950 (4%)			0.1954 (100%)		
69-4			240											

In this example there is an agent A1 that presents a very the high value of fitness. For this player this results in using other roles, as the values of preferences for these roles are also high and also much higher than for other players filling these roles. There are two cases that illustrate this process. A1 makes use sometimes of the roles of RM and LM, these roles are usually filled by A2 and A4. As A1's preferences for RM is 22.1 (> 18.6 for RM) and 21.1 (>18.0). In these situations A2 and A4 have to choose a different action from these roles, or use a different role. This fitness allows this robot to win almost all conflicts (98%) that may arise when two or more robots compete to execute the same action. On the other hand, the dominance of A1 results in a very low number of conflicts won by A2 (5%) and A4 (4%) This dominance can result in a more efficient way of completing tasks. One example to explain this benefit is the following. If A1 has gained control of the ball in midfield, it can attempt to move the ball forward, if on its way towards more offensive positions finds and obstacle it may have to avoid it, this may make A1 move to the left/right entering the area defined for a different role. In this situation it may decide to move back to its area leaving the ball unattended until a teammate filling this role regains control again of the ball. While a teammate attempts to regain control of the ball, the ball may be already under the control of the opponent. In case the A1's fitness is the present value, it can continue the play, controlling the ball, and leaving the ball near other players or attacking positions.

The score margin for this example is 65, while average of a team where the fitness is not considered is 43.7 goals. This represents a significant improvement. Although the fitness is not considered, there are players that are dominant with respect to others, as a result of the position they fill in the team. These players tend to win more conflicts than others, although in percentage terms is much lower. For the experiments performed under the same conditions as in this example, the average for the dominant players is around 75%. For the homogenous case this value is slightly above 50% (54%). For a balanced team, all players must win/lose the same number of conflicts. In this case, this value should be 50%.

A1			A2			A3			A4			A5		
MD	45%	25.2	LW	85%	18.1	RW	71%	18.6	RM	71%	17.8	GL	100%	29.5
LM	16%	21.6	LM	5%	17.4	ST	19%	18.1	LD	8%	17.5			
ST	12%	19.1				CF	6%	17.7	FB	8%	17.5			
RM	12%	19.2												
0.2208 (92%)			0.1910 (10%)			0.1953 (45%)			0.1929 (3%)			0.2 (100%)		
49-11			240											

This example illustrates the problem that may appear for having high values of fitness. There are some differences with respect to the previous example that leads to an important fall in team performance (from 65 to 38). Although the team formation is very similar to the previous example, A1 fills the role of MD, while A2 and A4 fill the roles of RM and LM. The most important difference can be observed in A1. In this example A1's preference for MD is lower than in the other examples. This situation leads to a lower number of times that this player uses its main role (from 67% to 45%). Moreover A1 uses other roles more often, increasing the number of conflicts with the players already using these roles. One additional problem in this example is that A1 fills also the role ST (12%). While the roles of LD and RD are adjacent to MD, ST is not so close. At the same time, using the role ST interferes with A3, which is responsible for scoring goals. The number of conflicts won by this player fall from 66% to 45%, as a result of the A1 dominance. This interference results in a lower performance in the attacking positions. One of the problems of having dominant robots is that the number of technical problems may increase. This type of robots may need more batteries, as they are continuously moving from one place to another. For the same reason these robots may break down more easily, as they make continuous use of their motors, mechanical parts...

5.5.3 Fitness and Performance

This section analyses how the performance of a two teams, playing against different opponents, changes depending on the dominance of the best-fitted players in the system, represented by high values of fitness. These two teams are tested playing against several hand-coded teams (BrianTeam, SchemaDemo and ou7kou).

In this analysis a new parameter is used to evaluate the dominance of the most fitted robots. Robot dominance is also measured as the standard deviation of the differences between the conflicts won and lost by each of the players (the goalie is excluded from this analysis) over all the games played for each case analysed. In systems where all components tend to win and lose the same number of times, this parameter presents a value close to 0 for all players, as the difference between the conflicts win and lose are close to 0. If in a system there are one or more components that win most of the conflicts, then there are others that tend to lose in all of them. In this case, the variance increases, as the difference of win and lose conflicts show a greater dispersion, even though the average is 0 when all players are considered. This value gives a clearer picture of how dominant are the players with the highest values of fitness.

In order to allow more interaction among the robots that compose the team some of the experimental conditions have been changed with respect to the experiments in chapter 7.

Fitness	Score Margin Mean	Variance	Std Deviation Conflicts
∞	97.3	130	289
1/70	101	80	402
1/25	102.9	110	539
1/1	85.2	180	795

Table 5.7 Fitness and Team Performance vs BrianTeam

Fitness	Score Margin Mean	Variance	Std Deviation Conflicts
∞	43.7	130	183
1/70	47.4	80	330
1/25	45.6	110	644
1/1	42.2	180	818

Table 5.8 Fitness and Team Performance vs SchemaDemo

Fitness	Score Margin Mean	Variance	Std Deviation Conflicts
∞	5.6	40	174
1/70	7.9	40	365
1/25	9.7	30	536
1/1	4.9	90	781

Table 5.9 Fitness and Team Performance vs ou7kou

Results displayed in the tables show that for moderate values of fitness, the team performance increases with respect to the case where fitness is not included, but as the dominance of the best-fitted agents increase from these values, performance tends to fall. In the first example analysed, the performance of the team for 1/1 is significantly lower than when the model does not include the fitness. This table also shows that even when fitness is not included there are robots that are dominant with respect to others. When analysing in depth the collected data, this result is explained by the position that some players fill in the team. One of the problems when tuning the function *sclp*, in order to calculate the fitness of each agent, is that it is difficult to know beforehand which is the best value, as this depends on the environment.

The influence of fitness in team performance has been analysed in a team of big size robots. The results displayed in the follow tables show that performance tend to decrease as the fitness increases. This results is explained by the physical interference in the team., due to the size of the robots. Dominant team members tend to collide or get stuck among several team mates, as they attempt to avoid other players, resulting in clear fall in team performance.

Fitness	Score Margin Mean	Variance	Std Deviation Conflicts
∞	74.4	65.3	224.4
1/50	70.9	71.3	349.7
1/25	66.5	72.2	611.4

Table 5.10 Fitness and Team Performance vs BrianTeam

Fitness	Score Margin Mean	Variance	Std Deviation Conflicts
∞	36.1	59.5	169.5
1/70	35.8	55	355.6
1/25	32.1	60.9	594

Table 5.11 Fitness and Team Performance vs SchemaDemo

Fitness	Score Margin Mean	Variance	Std Deviation Conflicts
∞	2.6	12.7	268.8
1/25	2.5	13.5	465
1/1	1.4	14.9	640

Table 5.12 Fitness and Team Performance vs ou7kou

When physical intereference does not represent a problem, the use of fitness results in an improvement team performance, as it has been also observed in other cases analysed, but not displayed here.

This model allows to stablish a hierarchical dominance, but unlike [Mataric 1995] and [Vaughan et al 2000], this results in an improvement of the performance of the whole system when physical interference does not play an important role. This model allows to set a hierarchical dominance and presents some advantages with respect to [Mataric 1995] and [Vaughan et al 2000]. The resulting structure is not rigid as in these works, and it is the result of the interaction of the components of the system and it measures the different abilities of the components of the system.

5.6 General Analysis

5.6.1 Introduction

This chapter has analysed how this model works, and how MAS may benefit from it. Although this model has been analysed in the domain as soccer, many results and recommendations can be extended to other domains, as they may present many similar features with soccer, or some concepts can be easily adapted. This section tries to give some hints on how MAS can benefit from the features presented in this chapter

5.6.2 Competition

Competition is the key point that allows self-organisation (task-partition) and group adaptation in natural systems. This concept can help MAS to adapt and to self-organise as this chapter has shown, and can help heterogeneous MAS to learn efficiently. In a more general case, MAS can benefit from these features, if the process of decision-making is designed in a way that allows agents to interact with each other and give rise to high number of conflicts. However, in domains where agent co-operation plays a very important role, competition may lead to low group performance, as some examples in this section has shown.

5.6.3 Environment Exploration

Exploration does not concern the process of exploring the environment at the beginning of a learning process, but how agents may explore to and adapt to changes in the environment when they are already filling one role (in a more general setting carrying out tasks, missions...). In the example selected for this thesis this is achieved by designing roles that overlap. In other domains and problems things cannot be designed in such a way. One possible way of solving this problem in other domains is by evaluating several aspects when carrying out a task... . For example, agent's preferences might represent interest for tasks involving a set of skills (preference for *r* may mean, interest for doing a tasks involving the skill *r*). Each task could be represented by a set of different skills and once this task is completed agents would update their preferences depending on the skills of this task. This methodology can help each robot to adapt to changes in the environment and in their own bodies.

5.6.4 Fitness

This chapter has proved the benefits of introducing the concept of agent's fitness, helping to the system adaptation and in some cases also to the performance. In a more general case, the introduction of this concept must contribute to the system in a similar way, by allowing those agents that are better adapted to the environment to participate more often in tasks than those with lower values of fitness. On one hand the whole system may benefit from having robots better fit for the environment. On the other hand, it clearly helps to the adaptability of the MAS as the several examples showed. However this can cause some problems. When working with physical systems, an excessive participation of one of the components may give rise to a rapid deterioration of the physical components. This chapter has also shown that high agent fitness can lead to lower team performance and problems in team adaptability.

5.7 Summary and Conclusions

This chapter has analysed the most important features of using the ecosystem approach for modelling agent's relations and learning in heterogeneous MAS. This approach has proved to be highly adaptable and present many features observed in natural systems, from which this smodel attempts to benefit. Fitness has turned out to be a very important factor in system adaptability and system performance, while competition contributed to an efficient role distribution in a team of heterogeneous players. At the end of this chapter some general conditions have been sketched out, which MAS should meet, to present these properties

This approach has proved to meet some of the requirements of COIN [Wolpert and Tumer 2000] and [Parunak 1997] for MAS, where. Interactions among simple agents give rise to complex behaviours, from which the system can benefit. This approach also facilitates the task of MAS engineering, as it is described in COIN. Once agents are designed and their interactions defined, the system can self-organise autonomously to act efficiently in the environment or to adapt to changes in it, without the need of human intervention.

Chapter 6

Heterogeneity Metric

6.1 Introduction

This chapter describes the methodology used in this work to evaluate the physical diversity of a group of heterogeneous physical agents. This metric is based on the Hierarchic Social Entropy [Balch 2000] that was designed to evaluate behavioural diversity among a group of agents. This methodology is adapted to evaluate physical diversity, even though its author claims that this is a difficult task. The degree of diversity is determined by means of several benchmarks, which allows to evaluate each robot at different game situations. This chapter is organised as follows: the first section reviews social entropy and motivates his choice, the second section describes how this methodology is adapted to the purposes of this work and finally the last section presents the results of the experiments for each robot.

6.2 Diversity Metrics

6.2.1 Social Entropy

Social entropy is based on [Shanon 1949] information entropy. This technique dates back to early days of cybernetics and was developed to quantify the uncertainty of an information source. The concepts developed by Shanon has been used by other researchers and adopted by other fields of research.

The information entropy of a system X is defined as:

$$H(X) = -K \cdot \sum_{i=1}^{M} p_i \cdot \log_2(p_i) \qquad (39)$$

M is the number of overlapping subsets.
p_i is the proportion of agents in the ith subset.
K is set by Shannon to 1.

Information entropy meets several properties. The two most interesting properties show that H is minimised when the system is homogeneous, and it is maximised when there are an equal number of agents in each subset. This value also increases as the number of subsets with equal number of agents increases.

The following example helps to clarify how the formula works.

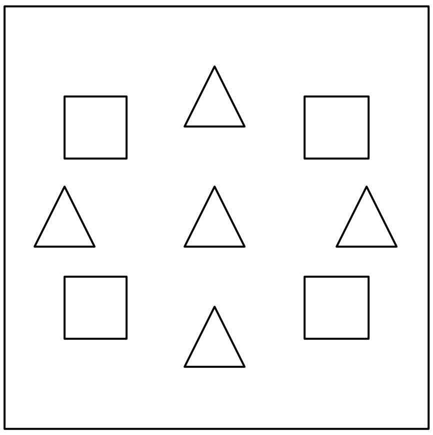


Figure 6.1 Example for Simple Entropy

The diversity value for this society (of 9 elements) is 0.99.

$$H\left(\frac{5}{9},\frac{4}{9}\right) = -\frac{5}{9}\log_2\left(\frac{5}{9}\right) + -\frac{4}{9}\log_2\left(\frac{4}{9}\right) = 0.99$$

For [Balch 2000], there are two limitations of this theory when used to measure diversity:

- ***Loss of information***. Several (including also an infinite number) societies can match any particular value of diversity. A single value does not give any information on the classes of agents and the number in each class.

- ***Lack of sensitivity to the degree of difference between agents***. This theory does not consider how different are the classes that made up the system.

6.2.2 Hierarchic Social Entropy

To overcome these drawbacks he defines a new metric, hierarchic social entropy. This new metric includes the differences among the subsets that compose the society. The most interesting feature of it is the inclusion of a clustering algorithm. This diversity measure is defined as follows:

$$S(R) = \int_0^\infty H(R,h)dh \qquad (40)$$

R is the society under evaluation
h is a parameter of the clustering algorithm indicating the maximum difference between two agents.
H(R,h) is the simple entropy for the clustering at level *h*.

This a simple example presented in the original paper, which is helpful to explain hierarchical social entropy.

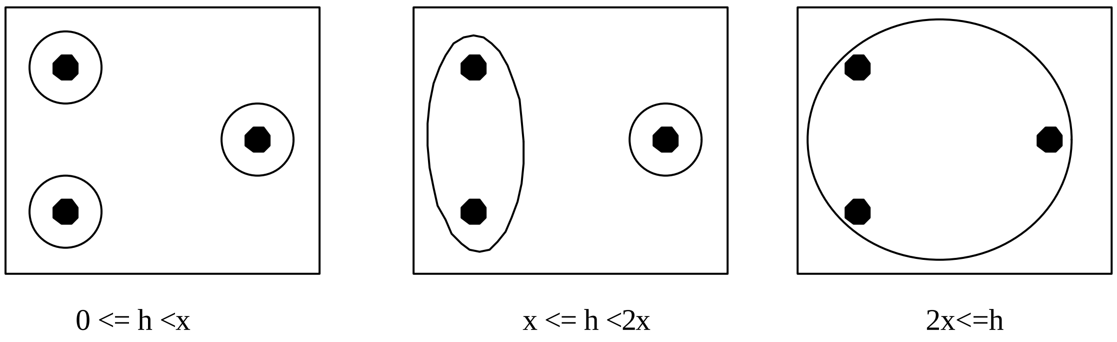


Figure 6.2 Example for Hierarchic Social Entropy

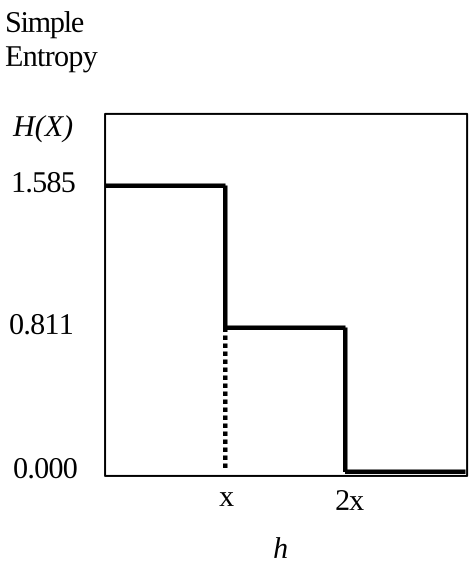


Figure 6.3 Simple Entropy and *h*

$$S(R)=\int_0^\infty H(R,h)dh=\int_0^x H\left(\frac{1}{3},\frac{1}{3},\frac{1}{3}\right)+\int_x^{2x} H\left(\frac{2}{3},\frac{1}{3}\right)+\int_{2x}^\infty H\left(\frac{3}{3}\right)$$

h is a parameter of the clustering algorithm. This algorithm clusters the elements of the society depending on the distance among them. Distance is in this case a scalar value and defines the behavioural difference between two elements, for this thesis this value defines physical difference between the robots of a team. Shanon's social entropy is evaluated for each value of h. When social entropy is evaluated for small values of *h* (in this example 0<= h<x), each component of the society form an independent cluster, in this situation the value of social entropy is the highest possible for this society. As *h* increases (x <= h <2x) some members of the society are clustered together, thus decreasing the value of social entropy. For large values of h (2x<=h) all members of the society are grouped together, and the degree of diversity in the group is 0.

The algorithm used for clustering is called C_u, (as it is defined in [Balch 2000])

1) Initialise *N* clusters with $c_i=\{r_i\}$

2) For each cluster c_i

   (a) For each r_j (except r_i) in R:
   (b) If (Distance $(r_j,r_k) \leq h$) for every r_k in c_i add element r_j to cluster c_i.

3) Discard redundant clusters

The upper limit of the integral can be ser at 1.0 and normalising all the differences using the greatest difference as 1.

Some of the properties of this metric are:

- It is a continuous ratio measure
- It has an absolute 0 and equal units
- It can distinguish differences between societies regardless of scale
- It accounts for continuous differences between elements in the society, unlike simple entropy

We think that this work can be very useful for our research, since it defines a methodology to quantify diversity in a group of heterogeneous robots. From the point of view of this thesis, this methodology is used to measure the physical difference in a group of heterogeneous robots.

6.3 Evaluating Physical Difference

6.3.1 Introduction

For Tucker Balch, one possible way of evaluating behavioural difference in a group of robots is by building an "evaluation chamber", exposing each robot to different situations and recording its traces. After the experiments, the resulting traces can be compared and the behavioural difference quantified. He claims that such a chamber is almost impossible to build, and as a result of it, he approaches the problem of quantifying behavioural difference from a different point of view, by comparing the policies of each agent, which define the behaviour of each robot.

Physical difference can also be quantified in an indirect way, by evaluating agent's performance in different environmental situations, as this result is conditioned by the physical abilities of each agent. In many cases, this can be done by measuring the time needed to complete all the different tasks, energy needed, quality of the result... In some situations, this type of information cannot be easily obtained or is difficult to evaluate. In this case, one possible way of obtaining this information is by defining a set of benchmarks, which represent some standard and usual situations that agents may face when acting in the environment. These benchmarks can be finally used to evaluate each agent independently under controlled environmental conditions.

The latter solution is the one selected to quantify physical diversity in soccer domain. One of the problems of measuring directly player performance in this domain is that it may prove difficult to evaluate a player in different game situations when playing against a team of five players. As the ball would be continuously under control of the opponent or some game situations would be difficult to evaluate. If the robot is evaluated in the context of playing in a team, results would not reflect the individual skills of each player and the presence of other players would distort the results. For these reasons several benchmarks are created, so that each player can be evaluated in different game situations and under controlled experimental conditions. This set of benchmarks is built on [Johnson et al 1998], where a set of benchmarks are designed to evaluate team skills, and are implemented using the benchmark implicit opponent [de la Rosa et al 2000], already described in chapter 4. These benchmarks aim at evaluating how each player perform under the following common game situations in soccer:

- Goalkeeper
- Defence
- Midfield
- Offense

The benchmark implicit opponent is used to include the skills of each opponent into the different benchmarks. This benchmark allows to model team skills by adjusting the inclination of the field and by placing obstacles. The parameters of this benchmark are tuned, so that the score of a (long) game between a given team and the implicit opponent ends in a draw.

6.3.2 Results Evaluation

These benchmarks are designed to evaluate different agent abilities, even though the objective is to calculate physical diversity. Although each benchmark has its own features, and aim at evaluating different aspects, all are implemented using the benchmark implicit opponent as the basis for its design. Each benchmark has been designed, so that the different aspects of the game can be included in each benchmark.

Results of each benchmark are recalculated to have equivalent evaluations for all benchmarks, in order to compare physical difference. Each result is compared with a maximum and minimum value that depends on the benchmark, thus the evaluation of each benchmark is a value that ranges between 0 and 1. The maximum and minimum values may be determined from two different points of view:

- ***Global evaluation***. The evaluation of each robot may be based on the global performance of this robot on each benchmark. This performance is compared with respect to an upper and lower bound. For example, in the benchmark designed to test the ability of a player in the role of goalie, the upper bound corresponds to a player that does not allow any goal and the lower bound corresponds to a goalie that is scored all the times. This evaluation allows to analyse and compare each robot with respect to all the possible existing robots in the world, and future ones. One of the problems of this approach is that sometimes is difficult to determine for some benchmarks the upper and lower bound. In other cases, it may also not reflect the diversity when comparing several groups of robots, as the lower and upper bound may correspond to very extreme cases, while robots may perform in a narrower range.

- ***Relative evaluation***. The evaluation of each robot may be based on the teammates' performances. The result of a given robot on a benchmark depends on itself, but also on how the other teammates perform against the same benchmark. In this case, the upper is determined by the best performance among the robots that are being considered, and the lower bound, for worst performance of this group of robots. This approach solves the problem of finding and upper and lower bound for each benchmark; but on the other hand poses a new problem. It is not possible to compare the diversity of two teams, whose entropy has been determined at two different moments in time. Since these two teams have a measure of diversity based on their members.

In this thesis, the latter case has been selected for evaluating the entropy of each team, since it does not aim at comparing teams from different designers. Besides, it allows to have a clearer picture of the degree of diversity of each team. In this case, the upper and lower bound corresponds to the best and worst performance for each of the benchmarks and every opponent considered. Later, the greatest behavioural difference is selected as the upper limit for the integral. In addition, each benchmark defines the possible global upper and lower bounds.

6.3.3 Goalie

This benchmark aims to evaluate the ability of each agent to keep the goal. This benchmark consists in placing the ball at different positions and letting the ball roll down the field. The inclination of the field corresponds to one found for the opponent selected. Apart from the inclination of the field some obstacles are placed near the goal in order to deflect, in some situations, the ball. All the starting positions of the ball guarantees that the ball ends up into the goal (without a goalie). Each test starts with the ball moving down from one of the predetermined positions, and ends when the ball is in the goal or hits one of the walls near the goal. If the ball gets stuck and it does not move for five seconds the experiments it is also considered finished. It is considered that the goalie has blocked the ball. If the ball does not end up into the goal, the result of the experiment is considered a save. Next, the ball is repositioned again at one of the predetermined positions, and the player starts to move again from its last position. All the set of positions is repeated several times.

The metric to evaluate robots in this benchmark is the percentage of saves,

$$b\_g = \frac{n\_saves}{n\_tests} \tag{41}$$

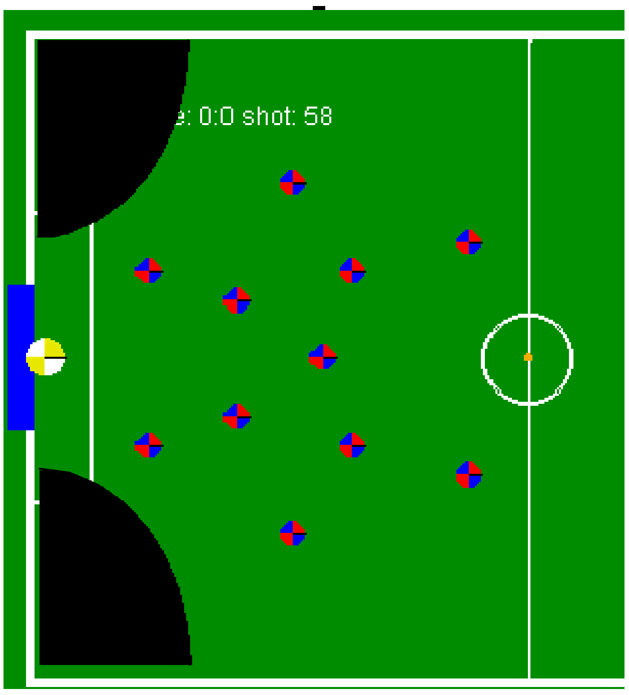


Figure 6.4 Goalie Benchmark

From the point of view of a global evaluation, the minimum value for this benchmark is 0 saves, and the maximum value is when the number of saves equals the numbers of tests.

$$bg = \frac{b\_g - b\_g_{\min}}{b\_g_{\max} - b\_g_{\min}} \tag{42}$$

6.3.4 Defence

This benchmark aims to evaluate the ability of a player to defend its own goal, by blocking or deflecting the ball. In this benchmark, the ball is positioned at different locations in midfield. At the beginning of each test, the robot starts from a position near its penalty area and the ball rolls down from one of the predetermined positions in midfield. From these positions the ball always ends up inside the goal, if the player does not block it or deflect it. Once the ball starts to move, the player attempts to intercept and block the ball or deflecting it into a non-dangerous trajectory. If the ball finally does not end up inside the goal, the result is considered a success; otherwise, if a goal is scored, the test is considered a failure.

The metric used to evaluate robots in this benchmark is similar to that of the goalie. It is the percentage of times that the ball has been block of deflected by the robot and it is considered a "save".

$$b\_d = \frac{n\_saves}{n\_tests} \tag{43}$$

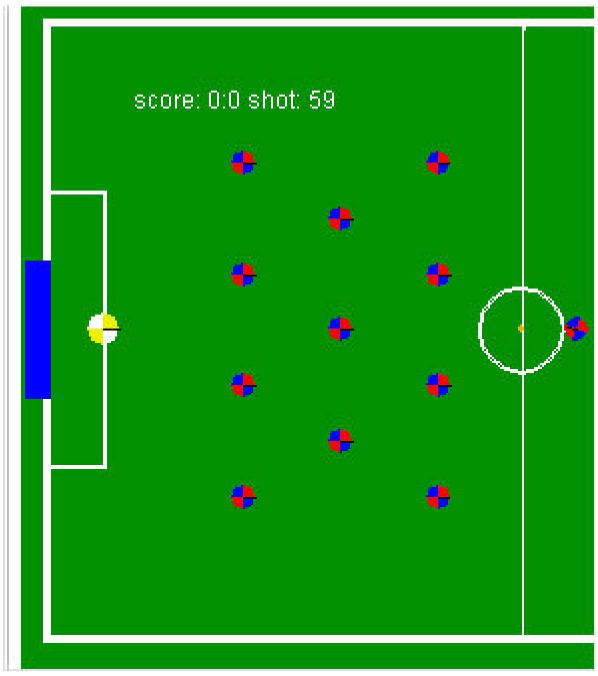


Figure 6.5 Defence Benchmark

From the point of view of a global evaluation, the minimum value and maximum value is the same as in the first benchmark, 0 saves for the lower bound and the number of tests for the upper bound.

$$bd = \frac{b\_d - b\_d_{\min}}{b\_d_{\max} - b\_d_{\min}} \tag{44}$$

6.3.5 Midfield

This benchmark aims to evaluate the ability of a player to move the ball from defence to attacking positions, so that the offense can start attacking the opponent goal. As in the other experiments, ball is positioned at several positions and this sequence of points is repeated several times. This benchmark starts by placing the ball at one of the predetermined positions. This ball does not start to move form this position until a few seconds have elapsed. The player that is being evaluated starts always from the same position. The goal of this benchmark is to reach the attacking positions with the ball (defined by the right white line), starting from the present position. If this point is reached it is considered a success, otherwise, if the player is only able to reach an intermediate position, the result is considered a failure.

The metric used to evaluate a player in this benchmark, is the number of times the player has managed to reach the right black line, with respect to the number of trials.

$$b\_m = \frac{n\_succes}{n\_tests} \tag{45}$$

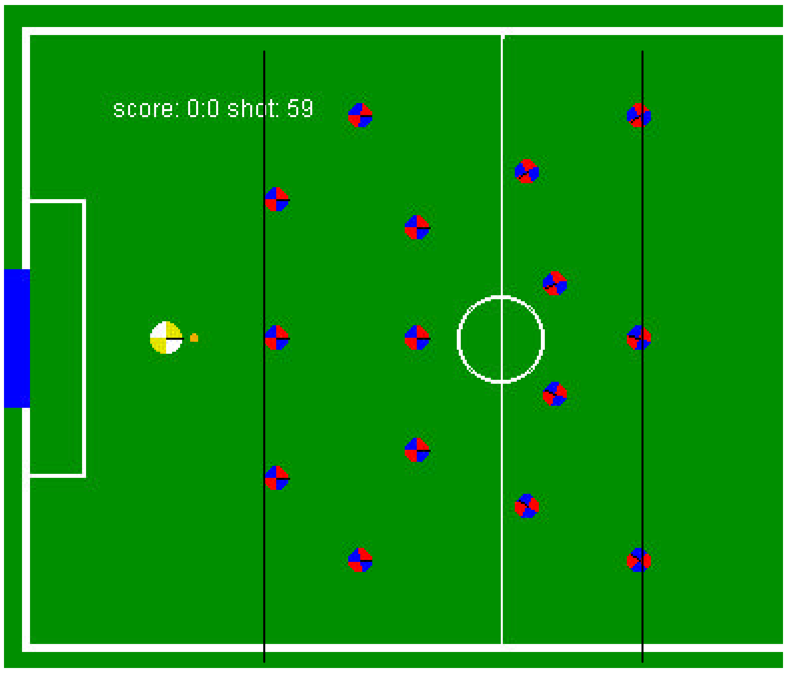


Figure 6.6 Midfield Benchmark

From the point of view of a global evaluation, the minimum value for this benchmark is 0, the player has never been able to reach the right line, and the maximum value equals the number of tests, this means that the evaluated player has been able to reach the second line in every test.

$$bm = \frac{b\_m - b\_m_{\min}}{b\_m_{\max} - b\_m_{\min}} \tag{46}$$

6.3.6 Offense

This benchmark aims to evaluate the ability of a player to score goals. As in the other experiments, ball is positioned at several positions and this sequence of points is repeated several times. In this case the ball is placed across the opponent field. The benchmark starts by placing the ball at one of the predetermined points. The ball does not start rolling down until it is touched by the player, which starts from midfield. If the ball ends up into the opponent goal before 60 seconds have elapsed, then it is considered a success. If during this period of time ball crosses the midfield line, it is replaced at the same position. In case the ball gets stuck, it is repositioned a few centimetres away, but it starts moving straightaway. After the 60 seconds the ball is repositioned at a new point and the robot starts from midfield again. Unlike other benchmarks, the ball is repositioned always at the same position over a period of time. The reason for this difference with respect to other benchmarks is that the obstacles cover practically completely the goal. In this situation, agents have to dribble several obstacles that stand on its way. As a result of it, the number of goals scored would be very low and not significant of agents' skills. This also represents much better the tasks of the attacker, which has to dribble several opponents and always try several times, before it scores a goal.

The metric used to evaluate a player in this benchmark is the number of goals scored.

$$b\_o = \frac{n\_goals}{n\_tests} \tag{47}$$

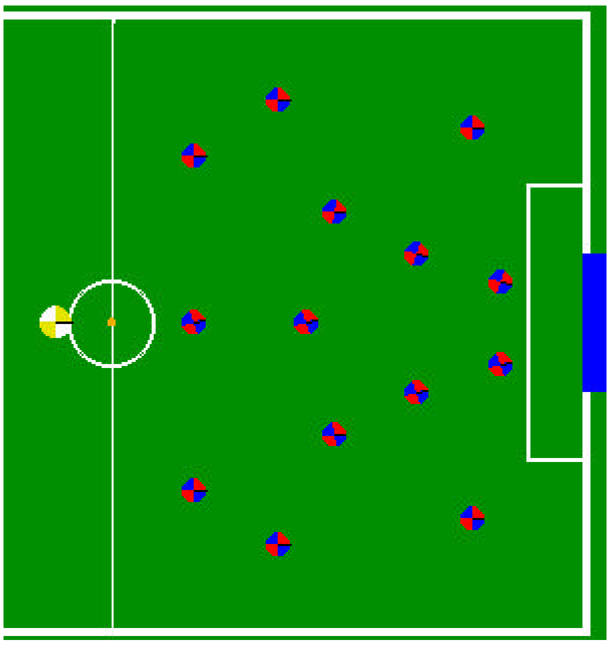


Figure 6.7 Offense Benchmark

From the point of view of a global evaluation, the minimum value for this benchmark is 0, the player has never been able to score a goal, and the maximum value equals the number of tests, in this case the evaluated player has been able to score a goal in all tests.

$$bo = \frac{b\_o - b\_o_{\min}}{b\_o_{\max} - b\_o_{\min}} \tag{48}$$

6.3.7 Entropy

Once each player has been evaluated on each benchmark and for different opponents, it is possible to compute the physical difference and the maximum physical difference (P_h_{max}) between two players. This value is necessary to fix the upper limit for the simple entropy (equation 2).

Physical difference between two players (*a* and *b*) for the opponent *o* is defined as follows:

$$P_o\_h_{ab} = \sqrt{(bg_a - bg_b)^2 + (bd_a - bd_b)^2 + (bm_a - bm_b)^2 + (bo_a - bo_b)^2} \tag{49}$$

In order to normalise values for all the different opponents, the last formula can be reformulated as follows:

$$P_o\_h_{ab} = \sqrt{\left(\frac{bg_a - bg_b}{\Delta bg_{\max}}\right)^2 + \left(\frac{bd_a - bd_b}{\Delta bd_{\max}}\right)^2 + \left(\frac{bm_a - bm_b}{\Delta bm_{\max}}\right)^2 + \left(\frac{bo_a - bo_b}{\Delta bo_{\max}}\right)^2} \tag{50}$$

Where Dbx_{max} denotes the maximum difference between two players for all environments considered for a given benchmark. The introduction of these values helps to compare the results for the different opponents considered.

The maximum difference between two players is defined as follows,

$$P\_h_{\max} = \max(P_o\_h_{ab}) \tag{51}$$

For each pair of players a and $b \wedge a$ ¹ b and each opponent o.

From the point of view of a global evaluation, the maximum difference equals the square root of the number of benchmarks, and physical difference is not normalised as in (12).

Once all the physical differences between each possible pair robots has been computed, and knowing P_h_{max}, the value of hierarchic social entropy for each team can be determined, using the clustering algorithm and computing the simple entropy for each level of the clustering.

6.3.8 Teams and Equal Diversity

One of the limitations of this diversity measure for some purposes is that the same value of diversity may correspond to different teams. This is especially true for homogeneous teams, where teams' diversity equals 0. Having a measure of diversity can be useful to build a team of heterogeneous robots, if it is combined with other information, as it allows to know which are the features that present the best teams (see next chapter). In order to benefit from this feature, it is necessary to introduce a new metric that allow to discriminate teams from a different point of view

Several approaches have been analysed to solve this problem. The one that suits better the needs of this research is to classify teams according to some of their features. In this particular case, teams are regarded as offensive or defensive and this parameter is defined as team profile

Teams are classified depending on the number and types of robots that compose each team and how each agent has performed individually against each of the benchmarks. Each benchmark describes a different situation of the game representing offensive/defensive plays. Using this information, it is possible to establish a scale that defines each benchmark as representative of offensive game situations or rather defensive. The goalie and offense may be regarded as completely defensive and offensive, while the defence and midfield may be regarded as partly defensive for the first case and partly offensive for the latter one.

According to these explanations, these benchmarks can be classified as offensive/defensive

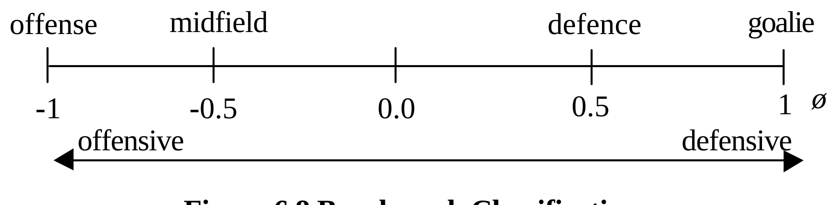


Figure 6.8 Benchmark Classifications

In order to compute the offensive/defensive degree of each team, the result for each benchmark can be multiplied by the values of the scale. This value can also be computed depending either on the global upper and lower limits or on the relative values for each benchmark. In the latter case, the maximum and minimum values correspond to the best and worst individual performance for each benchmark. In this case, all robots are regarded as defensive of offensive with respect to the other members analysed, even though all players may be either globally offensive or defensive.

The robot profile for player *k* (*rpro*$_k$) can be calculated as follows:

$$rpro_k = \sum_{i=1}^{i=B} y_i \cdot average(b_i) \qquad (52)$$

B denotes the number of benchmarks considered

The average is computed for every opponent analysed in benchmark *i*. The reason for using the average value is detailed later in this chapter (see section on diversity and environment).

Once the value for each player is known, the team profile (*tpro*) can be easily obtained using the information on the number and types of the different members.

$$tpro = \frac{1}{R} \cdot \sum_{i=1}^{i=R} rpro_i \qquad (53)$$

R is the number of robots that compose the team
i denotes each robot in the team, but it does not represent type of robots.

From a more general point of view, a team of physical agents might be classified depending on some of the features that the tasks present. For example, in the domain of exploration and surveillance this can be determined by the range of the robots. In other examples the size of the robot may play an important role, as there may be some areas only accessible to small robots...

One of the limitations of this approach is that tasks are only analysed in a binary scale, either defensive or offensive, and in some cases this scale might not be enough for other types of features considered. This aspect has not been addressed in this thesis and belongs to the future work.

6.4 Teams Diversity

6.4.1 Introduction

After defining the methodology to evaluate teams of heterogeneous members, this section focuses on the analysis of each type of agent used to build teams of homogeneous and heterogeneous robots.

This process starts by determining the skills of each opponent with respect to the implicit opponent. These teams, already described in chapter 4, are BrianTeam, SchemaDemo and ou7kou. One of the main problems existing in these teams is that they lack a good obstacle avoidance skill. Simulations show that they get easily stuck, even with small obstacles. This presents a serious problem, as in the benchmark the difficulty is mainly described by the inclination of the field and the obstacles. The position of the obstacles is used later for the benchmarks, even though some changes are also possible (for example, benchmark for the goalie).

Since it is not possible to place obstacles to evaluate the opponents, a compromise has been found, each team is evaluated without obstacles. For each benchmark, the obstacles, which are needed, are placed on the field, but the size of these obstacles is considerably reduced. These obstacles are placed; so that robots cannot succeed in each benchmark simply by following a straight line.

The following table shows the inclination found for each selected opponent,

Team	Inclination of the field
Brian Team	3.0º
Schema Demo	4.0º
Ou7kou	4.5º

Table 6.1 Teams vs Implicit Opponent

Two additional cases have been analysed, in order to evaluate how team diversity changes depending on the difficulty of the opponents. These are the case for a very easy opponent (inclination 2º) and a very difficult opponent (inclination 5º). This is evaluated for the case of a fully heterogeneous team.

6.4.2 Benchmark Definition

This section describes each benchmark used to evaluate robots using the implicit opponent notation.

Goalie

< F-180, P={0.0523,0.0698,0.0785} rad, P_r, P_o, T_e=¥ seconds, F_b, C, J={Gi} >

P_r is a pattern of evaluated robots and is the tuple
<N_r=1 robot, S_r=round, A_r ={0.0095,0.104,0.0113,0.0122,9,0132}m^2,<-1.5, 0.0>>

P_o is a pattern of opponents and is the tuple
<N_o=2 robots, S_o=round, A_o=3.53m^2,<-1.8, -1.2>,<-1.8,1.2>>

F_b contains the features of the ball and is the tuple
<T_b=golf, CAN=rebound, R=<0.3:0.10:1.20,-0.25:0.05:0.25>*,SD=3>

C is the constraints and is the tuple
< T_b=¥, $||V_r||$= (), $||V_s||$= (), Z=() >

*0.3:0.10:1.20 denotes all the points starting in 0.3 an increasing by 0.10 up to 1.20

Defence

< F-180, P={0.0523,0.0698,0.0785} rad, P_r, P_o={}, T_e=60 seconds, F_b, C, J={Gi} >

P_r is a pattern of evaluated robots and is the tuple
<N_r=1 robot, S_r=round, A_r ={0.0095,0.104,0.0113,0.0122,9,0132}m^2,<-1.5, 0.0>>

F_b contains the features of the ball and is the tuple
<T_b=golf, CAN=rebound, R=<0.0,-0.225:0.05:0.225>,SD=3>

C is the constraints and is the tuple
< T_b=¥, $||V_r||$= (), $||V_s||$= (), Z=() >

Midfield

< F-180, P={0.0523,0.0698,0.0785} rad, P_r, P_o, T_e=60 seconds, F_b, C, J=b_m* >

P_r is a pattern of evaluated robots and is the tuple
<N_r=1 robot, S_r=round, A_r ={0.0095,0.104,0.0113,0.0122,9,0132}m^2,<-1.5, 0.0>>

P_o is a pattern of opponents and is the tuple
**<N_o=22 robots, S_o=round, A_o=0.075m^2,<-0.35, -0.75:0.25:0.75>,
<-0.05,-0.85:0.25:-0.65>,<0.25,-0.75:0.25:-0.75>>**

F_b contains the features of the ball and is the tuple
<T_b=golf, CAN=rebound, R=<-0.5,-0.15:0.03:0.15>,SD=3>

C is the constraints and is the tuple
< T_b=2 seconds, ||V_r||= (), ||V_s||= (), Z=() >

*it is defined in the corresponding benchmark

Offense

< F-180, P={0.0523,0.0698,0.0785} rad, P_r, P_o, T_e=¥ seconds, F_b, C, J= G_e >

P_r is a pattern of evaluated robots and is the tuple
<N_r=1 robot, S_r=round, A_r ={0.0095,0.104,0.0113,0.0122,9,0132}m^2,<-1.5, 0.0>>

P_o is a pattern of opponents and is the tuple
**<N_o=18 robots, S_o=round, A_o=0.070m^2,<1.6,-0.40>,<1.6,0.40>,<1.5,-0.5>,
<1.5,-0.15>,<1.5,0.15>,<1.5,0.5>,<1.3,0.0>,<1.2,0.3>,<-1.2,-0.3>,<1.1,-0.9>,
<1.1,-0.5>,<1.1,0.5>,<1.1,-0.9>,<0.8,-0.8>,<0.8,-0.4>, <0.8,0.0>,<0.8,0.4>,<0.8,0.8> >**

F_b contains the features of the ball and is the tuple
**<T_b=golf, CAN=rebound, R=<0.4,0.55>,<0.4,±0.25>,<0.4,±0.20>,<0.4,±0.15>,
<0.4,±0.10>,SD=3>**

C is the constraints and is the tuple
< T_b=2 seconds, ||V_r||= (), ||V_s||= (), Z=() >

6.4.3 Bechmark Results

Once the benchmark implicit opponent has been tuned for each of the opponents, it is possible to evaluate each type of player. In order to get a clear and unbiased evaluation of each player, 1000 tests are conducted on each benchmark.

	Golie	Denfense	Midfield	Offense
Brian Team	825	902	150	102
SchemaDemo	690	886	65	90
Ou7kou	679	813	45	64
2º	960	948	331	132
5º	654	653	31	33

Table 6.2 Results for player A0

	Golie	Denfense	Midfield	Offense
Brian Team	456	957	134	322
SchemaDemo	358	919	66	276
Ou7kou	385	891	44	220
2º	572	957	300	414
5º	386	840	42	212

Table 6.3 Results for player A1

	Golie	Denfense	Midfield	Offense
Brian Team	521	980	109	354
SchemaDemo	416	927	50	254
Ou7kou	459	896	50	206
2º	662	999	278	446
5º	488	851	34	92

Table 6.4 Results for player A2

	Golie	Denfense	Midfield	Offense
Brian Team	655	989	89	358
SchemaDemo	556	922	61	202
Ou7kou	529	901	36	190
2º	765	999	218	481
5º	532	851	21	140

Table 6.5 Results for player A3

	Golie	Denfense	Midfield	Offense
Brian Team	735	990	143	284
SchemaDemo	619	927	64	188
Ou7kou	610	886	55	166
2º	891	1000	320	506
5º	531	786	41	132

Table 6.6 Results for player A4

A different type of robots have been also evaluated and analysed in this section. These robots present the same features as the one already analysed, but with the difference of the radius. In this case the radius has been set at 6 centimetres for all robots. A3 is not analysed here, as its radius is already 6 cm. These new robots allow to analyse some interesting aspects in the next chapter.

	Golie	Denfense	Midfield	Offense
Brian Team	788	973	122	102
SchemaDemo	677	895	51	61
Ou7kou	629	857	42	50
2º	961	999	276	124
5º	606	643	27	53

Table 6.7 Results for player A0 and 6 cm radius

	Golie	Denfense	Midfield	Offense
Brian Team	523	979	170	406
SchemaDemo	451	938	88	314
Ou7kou	451	897	57	300
2º	627	962	365	424
5º	461	889	57	220

Table 6.8 Results for player A1 and 6 cm radius

	Golie	Denfense	Midfield	Offense
Brian Team	588	986	145	376
SchemaDemo	514	947	82	300
Ou7kou	488	913	52	234
2º	665	984	306	440
5º	498	882	48	198

Table 6.9 Results for player A2 and 6 cm radius

	Golie	Denfense	Midfield	Offense
Brian Team	736	982	110	350
SchemaDemo	582	839	64	188
Ou7kou	612	733	50	188
2º	845	1000	267	480
5º	554	749	28	144

Table 6.10 Results for player A4 and 6 cm radius

Using the information obtained in each benchmark, the robot profile for each type of robot can be computed:

Team	A1	A2	A3	A4	A5
Normal	-0.338	-0.127	0.213	0.453	0.770
Radius 6 cm	-0.423	-0.137	0.213	0.156	0.876

Table 6.11 Robot profile

Chapter 7, where several homogeneous and heterogeneous robot teams are analysed, presents the values of diversity for each of team built using the robots evaluated in this section.

6.4.4 Diversity and Environment

Physical diversity presents some features that are not observed in behavioural diversity. The most interesting one is related to the change of physical diversity depending on the environment conditions. As the previous results have shown, agent’s skills do not manifest in the same way for different environments. This results in different diversity values.

Table 6.11 presents the diversity degree for a team of fully heterogeneous robots when they play against different opponents. The first result that can be observed is that heterogeneity decreases as the opponent difficulty increases. When this difficulty reaches a given point team diversity tends to stabilise.

Team	Diversity
2º	1.62
Brian Team	1.31
SchemaDemo	1.05
Ou7kou	1.11
5º	1.11

Table 6.12 Diversity and Opponent

There is a case that has not been analysed but the results can be somehow speculated. If the diversity of a team is evaluated against a very difficult team, for example a team that is modelled by means of the implicit opponent with an inclination of 90º, this an hypothetical case, Robots will perform very similarly against this opponent, as their skills will not allow them to succeed in any test, this will probably result in a lower degree of heterogeneity than for the cases analysed.

Using the information on the real experiments and the case analysed, it is possible to define the possible function that describes how the degree of diversity changes in this particular case (Figure 6.9).

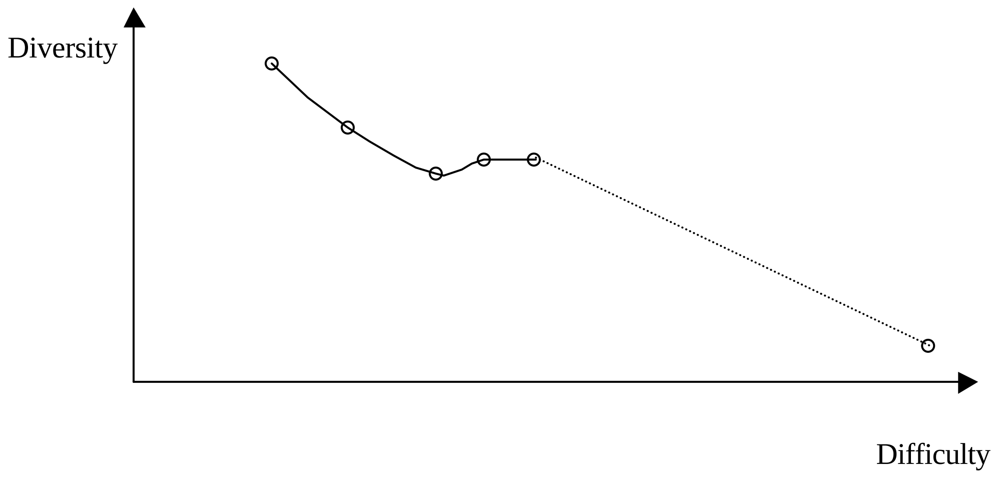


Figure 6.9 Diversity and Environment

From these results it can be concluded that diversity it is difficult to measure. One possible way of solving the problem of changing diversity is by computing an average value of the diversity for every possible environment. As this is quite difficult to do, because of the large number of environments, a possible solution is to evaluate each robot in a set of environments that can be representative of different typical degree of complexity. In the case of soccer, this can be achieved by evaluating the different robots against teams that may represent three levels of difficulty, easy, intermediate and difficult, as it is done in this chapter. If the designer knows the fraction of each of these degrees of difficulty that the team can face, it is also possible to compute a weighted average.

6.5 Summary and Conclusions

This chapter has described an approach to quantify physical heterogeneity in teams of robots. This approach is based on [Balch 2000] and evaluates each agent skills by means of a set of benchmarks. Agent's skills are a manifestation of the impact of the physical features in robot behaviour. In order to evaluate these skills a set of benchmarks have been defined.

Another interesting feature of robot physical diversity is that it changes depending on the environment, in soccer represented by the opponent team. One possible way of quantifying physical diversity is by evaluating each robot in several different typical environments and averaging diversity for the different cases. Finally this methodology has been used to set the diversity for the different teams of heterogeneous robots that are analysed and detailed in the next chapter.

In a more general setting, each robot skills can be evaluated from different points of view. In some cases it is possible to use the information associated to a give task. For the example the time needed, a measure of the quality of the result, energy consumed... In some cases this may not be evaluated and each robot should be tested against several benchmarks that represent different environmental situations.

Chapter 7

Heterogeneity Analysis

7.1 Introduction

This chapter analyses the impact of physical heterogeneity in robotic teams. It studies the heterogeneity from the point of view of team performance and task partition, and compares the results among several homogenous and heterogeneous teams. All teams are made up of five robots.

Several teams have been built to play against different robotic teams, in the domain of soccer. The robots used to build these teams are endowed with different skills. These skills allow each robot to perform all tasks included in the different roles, but the level of efficiency depends on the body of the robot and the environment. In this chapter, the algorithm defined and analysed in previous sections is applied to the task of learning the role's distribution in teams of homogeneous and heterogeneous teams. The experiments conducted to analyse these aspects can be divided into three different groups.

The first set of experiments aims to evaluate a set of homogenous and heterogeneous teams against a set of hand-coded teams. These teams have been already used to help to measure physical diversity and are used as a benchmark. Each homogeneous team is composed of five robots, each one, built using a different type of robot (see chapter on the implementation for more details). The hand-coded teams present several levels of difficulty: easy, intermediate and difficult. Using the results on the games for the homogeneous teams, a set of four heterogeneous teams is designed. These teams attempt to combine several features that have been observed in the fully homogeneous ones. These new teams are also tested against the same hand-coded teams.

In the second set of experiments, all homogeneous and heterogeneous teams play against each other. This section aims to study which is the best team among the two groups, and how heterogeneity conditions performance and task partition, when several teams built using the same set of available robots and having available the same set roles play against each other. Based on these results a new heterogeneous team is created. This team is built so that it is able to beat the best team from the competition among the different teams. This new team is also tested against the hand-coded teams.

In the third set of experiments a special case of heterogeneity is analysed. The robots used for the two first sections are designed in a way that each robot is endowed with different but balanced skills. In the case analysed in this section, there are some types of robots that are much better skilled than others, but the average skills are similar to the other heterogeneous teams. This team is tested against the hand-coded teams and the homogenous and heterogeneous learning teams.

After these three sections, results from the experiments are analysed from the point of view of the metric defined to measure physical heterogeneity. This section also raises the question of the design of heterogeneous robot teams on the basis of the diversity and the features of the robots of each team.

Finally the last section draw conclusions from the different experiments and attempts to describe which are the benefits of using heterogeneous MAS. These conclusions must help designers to build and design such systems.

7.2 Analysis Methodology

All the experiments are conducted under the same conditions. In each example one/two teams learn over a period of 180 steps, while the last 60 steps are used to evaluate the performance of each one of these teams using the learned role distribution. For the hand-coded teams each game is repeated 80 times, while for the second set of experiments each game is repeated 50 times. For all these simulations several types of data are collected and analysed. This information is displayed in several types of tables.

The first type of table (Table 7.1) contains the basic information on the score and the number of goals scored/allowed. (1) details the team that is analysed. The second column (2) displays the arithmetical mean of the score margin, while the third column (3) contains the information on the (corrected) standard deviation of the score margin. In (4) the confidence interval for the mean is calculated at a 95% level. (5), (6), (7) and (8) contain the information on the average values of the goals scored and allowed. The first two columns of this group display the number of goals scored/allowed for all the game, while the last two columns contain the information on the average values for the evaluation period.

Team	Mean	Standard Deviation	Interval estimation	Goals Scored	Goals Allowed	Goals Scored	Goals Allowed
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)

Table 7.1 Table for Results

The second type of table (Table 7.2) displays the information on the global role usage during the evaluation period. (1) indicates the team analysed, while (2) shows the information on the role usage for each role. Two letters represent each role. The roles and its corresponding letters are described in the chapter on the implementation. (2) is a value between 0 and 5. This gives information on how often a role is used by a player, as 5 is the number of robots that compose each team. If a role in (2) displays a value of 1, this means that this role is filled by one agent in average during the evaluation period. In some case some roles may take the value of 0.90, this also may indicate that this role is usually filled in each game, but that the player that plays in this role may be also using other close roles. This may occur as a result of the conflicts that arises among the different players, leading some agents to use other roles.

Team	GL	FB	RD	LD	MD	RM	LM	CF	ST	RW	LW
(1)						(2)					

Table 7.2 Table for Team Formation

The third type of table (Table 7.3) is only used to analyse heterogeneous teams. It displays the information on the role distribution. (1) defines the team analysed, (2) the type of robot and (3) the name of the role used (also represented by the two letters). (4) displays the role usage of the role in (3) for the type of agent in (2). In case there are several robots of the same class, the average value for the different robots are displayed in this table.

Team	P1 (2)	P2	P3	P4	P5
(1)	(3) (4)				

Table 7.3 Table for Role Distribution

From a statistical point view, results are analysed at a level of confidence of 95%. The results for the experiments against hand-coded teams are analysed by means of an analysis of variance (ANOVA), this allows to define which are the best robots in each group. This analysis is divided into two different groups, the homogenous and heterogeneous teams. Next, the average results for the homogenous and heterogeneous teams are compered using a t-test.

For the second set of experiments, the score is tested against the hypothesis that the result is a draw (score differential equals 0). If this hypothesis is rejected with the available data, then a victory is given to the team that has score more goals or a defeat to the team that scored less goals over the evaluation period. If the hypothesis is not rejected the result is considered a draw.

In order to facilitate the understanding and the explanation, each team is represented by the types of players and the number of player of each type. Later in this chapter, these results are represented using the diversity measure for each of the teams. For example a team composed of 1 robot of type 1, two robots of type 2, one robot of type 4 and one of type 5 is represented by the following set of numbers 12011. There is a homogeneous team for each type of robot, this means that there are 5 teams of homogeneous robots. The teams made up of robots of type *n* are denoted as team *Hn*. The information about the types of robots used to build the teams can be found in chapter 4, where their physical features are detailed in depth.

The slope of the function *sclp*, which links performance and fitness, is set at 1/100, so that fitness does not affect team performance in big size robot teams, but preserving at the same time team adaptability.

7.3 Hand-Coded Teams

7.3.1 Introduction

This section studies a set of different homogeneous teams when they play against three hand coded teams selected. Once the homogeneous teams have been tested, four heterogeneous teams are built attempting to combine some of the best performing robots found in the homogeneous teams. These teams are also evaluated against the hand-coded teams.

The homogeneous teams analysed are the following

Team	Team Code	Player 1	Player 2	Player 3	Player 4	Player 5	Diversity Value	Team Profile
H1	50000	A1	A1	A1	A1	A1	0	-0.34
H2	05000	A2	A2	A2	A2	A2	0	-0.13
H3	00500	A3	A3	A3	A3	A3	0	0.21
H4	00050	A4	A4	A4	A4	A4	0	0.45
H5	00005	A5	A5	A5	A5	A5	0	0.77

Table 7.4 Homogeneous Teams

7.3.2 Homogeneous Teams vs Hand-Coded Teams

Table 7.5 shows the results obtained for all the homogeneous teams when playing against **BrianTeam**. An Analysis of Variance has been performed on the means of the score. Results for this analysis conclude that all the means are significantly different.

Team	Mean	Standard Deviation	Interval Estimation	Goals Scored	Goals Allowed	Goals Scored	Goals Allowed
H1-50000	62.3	108.5	64.6 – 59.9	248	46	70.1	7.7
H2-05000	72.8	79.7	74.8 – 70.8	261.8	28.6	77.0	4.2
H3-00500	77.2	80.4	79.2 – 75.2	252.3	18.3	79.9	2.7
H4-00050	68.6	59.4	70.3 – 66.8	193.2	12.6	69.8	1.2
H5-00005	58.1	53.5	59.8 – 56.5	170.6	11.9	59.0	0.8
Average	67.8	-	-	225.2	23.5	71.1	3.3

Table 7.5 Results for Homogeneous Teams vs BrianTeam

From this table, the team composed of robots of type 3 offers the best performance among all five teams. This team is followed by H2, which is also able to win by a goal margin over 70. H2 and H3 are the two teams that have been able score significantly more goals, although this value is slightly lower for H2. The third team that shows a better performance is H4. This team is able to win by a larger score margin than H1, although this player seems to be better fit to play against this opponent, as fast players in this case tend to score more goals. However, its bad defensive skills results in a lower score margin. H5 is the team that wins by the narrowest margin, although it is the team that has allowed the fewest goals. However the difference in the number of goals between H4 and H5 (0.4) can not be considered statically significant. It follows from the simulations against **BrianTeam** that big robots are able to defend better, but their performance attacking is rather poor. On the other hand, fast players, with the exception of H1, tend to score more goals, although they also allow more goals.

Team	GL	FB	RD	LD	MD	RM	LM	CF	ST	RW	LW
H1-50000	1.0	0.03	0.06	0.03	0.96	0.75	0.93	0.03	0.83	0.17	0.21
H2-05000	1.0	0.04	0.03	0.02	0.97	0.80	0.93	0.04	0.87	0.08	0.22
H3-00500	1.0	0.05	0.02	0.05	0.99	0.84	0.95	0.05	0.89	0.05	0.11
H4-00050	1.0	0.06	0.05	0.05	1.01	0.80	0.90	0.08	0.86	0.06	0.13
H5-00005	1.0	0.1	0.03	0.04	0.95	0.90	0.96	0.05	0.70	0.08	0.19

Table 7.6 Team Formations for Homogeneous Teams vs BrianTeam

Table 7.6 presents the team formations during the evaluation period. All types of agents present similar formations with the exception H5. A typical team is composed of a goalie, a midfield-defender, a right and a left midfielder and a striker. Sometimes the striker is helped by a right/left winger. In other cases the player filling the role of striker also uses one of the wingers. This is especially true for H1 and H2, since the players of these teams are able to move fast and can cover more ground. H5 is the kind of robot that plays less often in offensive positions, as a result of its poor dribbling skill. Since it is a slow robot, it loses quite often control of the ball when it attempts to dribble an opponent.

The results of the ANOVA analysis for the hand-coded team **SchemaDemo** (Table 7.7) shows that there are only two pairs of means, whose difference can be considered significantly different. These two pairs are H1-H5 H2-H5. If in this analysis H1 and H3 were the only teams analysed, a t-test would show that the means are statically different.

Team	Mean	Standard Deviation	Interval Estimation	Goals Scored	Goals Allowed	Goals Scored	Goals Allowed
H1-50000	15.3	44.5	16.8 – 13.8	92.5	64.1	27.1	11.7
H2-05000	14.9	40.6	16.3 – 13.5	77.7	65.2	24.7	9.8
H3-00500	13.1	42.7	14.6 – 11.6	66.2	55.7	22.6	9.5
H4-00050	13.5	37.6	14.9 – 12.2	62.2	57.9	23.3	9.8
H5-00005	10.9	37.2	12.3 – 9.6	67.3	60.5	22.1	11.2
Average	13.5	-	-	73.2	60.7	24.0	10.4

Table 7.7 Results for Homogeneous Teams vs SchemaDemo

Results for the homogeneous teams show that the two teams with the fastest robots H1 and H2, are the players that are able to obtain better results, even though this result is not significant. The opponent and the skills of these players can explain these results. As it has detailed in a previous chapter, **SchemaDemo** is able to control the ball very efficiently. Right after one of the players has gained control of the ball, the rest of the team moves in formation around this player, in order to prevent opponents from regaining control of the ball. A similar strategy is used to regain control of the ball. Another feature of this team is that all five robots defend on the penalty zone at the same time. These two features benefit players that are able to move fast and are small, this size allows these players to move through the opponents that defend the goal and the ball. This opponent strategy affects specially the performance of team H5, the team with the biggest robots, which is the one that wins by the narrowest score margin. The number of goals allowed by the homogeneous players are very similar for all the teams, unlike the previous case analysed, where the number of goals allowed is correlated to the size of the robots. This difference is explained by the fact that fast players tend to keep the ball away from their own goal, while slow players tend to cover better the goal, but the opponent has more chances to score goals.

Team	GL	FB	RD	LD	MD	RM	LM	CF	ST	RW	LW
H1-50000	1.0	0.04	0.08	0.14	0.98	0.84	0.88	0.06	0.29	0.36	0.33
H2-05000	1.0	0.04	0.11	0.06	0.99	0.84	0.88	0.07	0.34	0.34	0.33
H3-00500	1.0	0.04	0.06	0.10	1.01	0.82	0.83	0.07	0.49	0.24	0.34
H4-00050	1.0	0.05	0.08	0.07	1.03	0.81	0.83	0.08	0.51	0.24	0.30
H5-00005	1.0	0.07	0.07	0.05	0.97	0.87	0.87	0.07	0.24	0.43	0.46

Table 7.8 Team Formations for Homogeneous Teams vs SchemaDemo

The team formations (Table 7.8) resulting from the learning process are very similar to those observed in **BrianTeam**. The typical team formation is composed of a goalie, a midfield-defender, a right and a left midfielder, and an attacker. The main difference with respect to **BrianTeam** can be found in the offensive positions. In this case, striker positions are not so often filled as in the previous case. This is explained by the formation that **SchemaDemo** presents in defence and the problem that many teams have to enter the penalty area through the center part of the field. This is especially true for the fast robots, H1 and H2, and H5. For H5 the resulting team formation is quite interesting, it presents the highest role usage for attacking position, even though it has very poor dribbling skills. However, its powerful kicking device allows it to shoot and score goals from outside the penalty area, as all rivals are usually inside the area and do not attempt to get to the goal. H4 and H5 tend to attack through the striker position; this is explained by the combination of two different factors. On one hand H4 has also a powerful kicking device. On the other hand both players are able to move relatively fast and their bigger size allows them to keep more time the control of the ball, even if they collide with one of the players inside the penalty area

The Analysis of Variance performed on the means of the score margin for **ou7kou** (Table 7.9) shows that all the differences are significant with the exception of H1-H2.

Ou7kou is the most difficult opponent selected for these experiments. Results show that teams with very poor defensive skills tend to allow many goals. The number of goals scored by the opponent tends to decrease as the size of the robot increases. The score difference is also correlated with the size, results improve as the radius increases. It is clear that in this case the size plays a very important role in the results. H1 and H2 are the two teams that are able to score more goals. However, these teams have obtained the worst results. This is explained by the fact that the only way to defend their goal is by attacking, this means trying to keep away the ball from its penalty area as long as possible. For the middle and big robots the results change slightly. In this group of robots, the one that is able to score more goals, is A5, this represents the slowest robots among all the players. Although the difference is not very significant, its powerful kicking device allows it to score goals from outside the penalty area, as it has been remarked in the previous case. This feature also helps this robot to kick the ball away from its own field, keeping the opponent and the ball away from the penalty area..

Team	Mean	Standard Deviation	Interval Estimation	Goals Scored	Goals Allowed	Goals Scored	Goals Allowed
H1-50000	-23.7	75.6	- 21.8 - -25.7	47.4	160.4	12.3	36.0
H2-05000	-22.1	73.9	-20.6 - -24.1	27.8	130.2	7.05	29.2
H3-00500	-13.1	28.4	-11.9 - -14.3	13.4	87.5	4.1	17.3
H4-00050	-5.7	34.3	-4.4 - -7.0	13.6	67.1	2.9	8.9
H5-00005	-2.3	13.4	-1.5 - -3.1	19.8	54.1	5.4	7.2
Average	-13.4			24.4	99.9	6.3	19.8

Table 7.9 Results for Homogeneous Teams vs ou7kou

Team	GL	FB	RD	LD	MD	RM	LM	CF	ST	RW	LW
H1-50000	1.0	0.04	0.87	0.77	0.67	0.47	0.67	0.07	0.13	0.22	0.39
H2-05000	1.0	0.08	0.90	0.83	0.65	0.31	0.56	0.07	0.08	0.27	0.25
H3-00500	1.0	0.06	0.91	0.83	0.73	0.35	0.48	0.07	0.09	0.24	0.24
H4-00050	1.0	0.09	0.93	0.85	0.69	0.42	0.43	0.08	0.09	0.19	0.23
H5-00005	1.0	0.05	0.88	0.81	0.85	0.53	0.44	0.06	0.07	0.12	0.19

Table 7.10 Team Formations for Homogeneous Teams vs ou7kou

The resulting team formations (Table 7.10) for this opponent change substantially with respect to the previous opponents. Teams adopt a more defensive formation as a result of the attacking skills of **ou7kou**. A typical team formation is composed of a goalie, a left and a right defender and a mifield-defender. The rest of the roles are partly filled. Usually the player filling the role of middle defender tends to partly use either the right or left midfielder. Players acting as left and right defender can also partly use these roles in midfield. The remaining robot may act as midfielder (right or left) and also fill one of the attacking positions, although this is in general not so usual. Formations also change depending on the team. A1 team shows higher preferences for offensive roles. Attacking positions for this type of player add 0.74, while A5 only adds 0.39. These preferences tend to decrease as size increases and robots move slower, increasing at the same time the preferences for defensive roles. For teams of fast robots, the strategy consists in pushing the ball away from the penalty area, as their defensive skills are rather poor.

7.3.3 Building Heterogeneous Teams

Using the results for the homogeneous teams, a set of four heterogeneous teams is designed to compare their results with the homogeneous ones. The heterogeneous teams are created following several criteria:

- Teams must be composed at least of three different types of robots
- As many as 2 robots of each type of robot
- 2 teams composed of rather attacking players (robots that score more goals)

- 2 teams composed of rather defensive players (robots that score less goals)

These teams attempt to combine players that have performed well in some of the positions, but always meeting the criteria defined above

Team	Team Code	Player 1	Player 2	Player 3	Player 4	Player 5	Diversity Value	Team Profile
HT1	11111	A1	A2	A3	A4	A5	1.26	0.19
HT2	11201	A1	A2	A3	A3	A5	1.08	0.15
HT3	01112	A2	A3	A4	A5	A5	1.06	0.42
HT4	00122	A3	A4	A4	A5	A5	0.73	0.53

Table 7.11 Heterogeneous Teams

The first team is the most obvious one. It is a team composed of the five different robots used in this thesis. It presents the highest heterogeneity value, among the all-possible teams that can be built using the available robots. The second team is the one that presents the most offensive robots, as a robot of type 4 is replaced by a robot of type 3. This player is the one that is able to score more goals against one of the opponents. On the other hand team 3 replaces a robot of type 1 by the most defensive player A5, as A1 is the player that presents the worst team performance in defence. Team 4 is the one that is composed of most defensive players, as there is only one player that can be considered a good attacker, A3.

7.3.4 Heterogeneous Teams vs Hand-Coded Teams

All the heterogeneous teams defined based on the homogeneous teams are tested in this section against the same three hand coded opponents.

Team	Mean	Standard Deviation	Interval Estimation	Goals Scored	Goals Allowed	Goals Scored	Goals Allowed
HT1-11111	72.6	112	75 – 70.3	230.9	20.55	74.2	1.6
HT2-11201	70.1	92.3	72 – 67	227.5	22.3	71.9	1.8
HT3-01112	71.1	98.1	73 – 69	215.2	14.9	72.1	1.0
HT4-00122	69.3	115	71 – 67	202.9	14.2	70.2	0.9
Average	70.8	-	-	219.1	18.0	72.1	1.3

Table 7.12 Results for Heterogeneous Teams vs BrianTeam

After conducting an analysis of variance (ANOVA) for the results against **BrianTeam** (Table 7.12), mean differences are not statistical significant. Almost all players present a very similar result

All the score margin means are below the best homogenous team performance, H3. The team performances are very similar to those that present H2 in terms of score differential, which performs significantly worse than H3. Among the heterogeneous teams, the team that wins by the widest score margin is HT1. Although the differences are not significant, it performs slightly better than team 2, which is composed of two players of type A3. This is explained by the fact that a defensive player as A4 contributes better to the defensive and midfield tasks, while agents of type A3 plays better in offense and RM and LM. Another interesting result that can be observed in these games is that the number of goals scored and allowed barely change across all the teams, unlike the homogenous teams, where these values change considerably from one team to another.

If the average of the means for the homogeneous and heterogeneous teams is compared, this value is greater for the heterogeneous teams, and the result is statically significant when performing a *t-test* on the average of the means for the homogenous and heterogeneous teams.

Teams' formations (Table 7.13) are very similar to those of the homogenous teams, with the exception of H5. Although there are teams that are made up of very defensive players (HT4), the team formation does not vary significantly as in H5.

Team	GL	FB	RD	LD	MD	RM	LM	CF	ST	RW	LW
HT1-11111	1.0	0.08	0.04	0.05	0.96	0.85	0.94	0.04	0.81	0.07	0.16
HT2-11201	1.0	0.06	0.03	0.07	0.97	0.80	0.93	0.04	0.82	0.10	0.18
HT3-01112	1.0	0.09	0.05	0.08	0.95	0.86	0.88	0.04	0.83	0.10	0.12
HT4-00122	1.0	0.08	0.04	0.05	0.96	0.85	0.93	0.07	0.80	0.08	0.13

Table 7.13 Team Formations for Heterogeneous Teams vs BrianTeam

Team	A1		A2		A3		A4		A5	
HT1-11111	LM	40%	RM	25%	ST	35%	GL	39%	GL	32%
	RM	23%	ST	24%	GL	24%	MD	18%	MD	32%
	MD	21%	LM	21%	RM	14%	LM	15%	RM	15%
HT2-11201	LM	31%	LM	31%	GL	26%			GL	45%
	MD	27%	MD	21%	ST	22%			MD	23%
	RM	18%	ST	19%	LM	15%			RM	13%
HT3-01112			LM	35%	ST	41%	GL	24%	GL	37%
			ST	23%	RM	22%	MD	19%	MD	21%
			MD	17%	MD	17%	LM	19%	RM	20%
HT4-00122					ST	43%	MD	23%	GL	26%
					LM	28%	GL	21%	RM	23%
					MD	12%	LM	15%	LM	20%

Table 7.14 Role Distribution for Heterogeneous Teams vs BrianTeam

Table 7.14 details the role distribution for the different types of players. The information on role usage shows that there is a strong correlation between what it has been observed in the homogenous team, and how the different types of players fill the different roles. The role ST is usually filled by agents of type 2 and 3. The homogenous teams H2 and H3 are able to score more goals than any other team. These two types of players tend to fill this role over 80% of the times; an agent selects this role. When they do not fill attacking roles they usually focus on LM or MD, where the speed and a having a good dribbling skill is also very important. These results also show that big robots (robots of type A4 and A5) play usually as a goalie, specially the latter one. In homogenous teams, these two types of players present the lowest number of allowed goals. In some simulations agents of type 3 also fill this role, which is also a good goalie against this opponent. Agents of type 1 tend to fill midfield roles; this follows from its physical features. Although this is a fast robot and can dribble very fast, its physical features (strong oscillations) makes it lose control of the ball quite often and performs worse that A2 and A3 in this role. Not losing the ball is especially important for the attacking roles, as they may miss good chances to score a goal. For this reason results show that this player tend to fill roles in midfield (LM, RM and MD) where losing control of the ball is not so critical. The MD is also filled by robots with powerful kicking devices; this position allows them to have very good chances to start an attacking play by kicking the ball. Big robots do not usually fill attacking roles, although they have also to play in midfield positions, usually filled by fast robots, in the team composed of defensive robots.

The ANOVA test for **SchemaDemo** (Table 7.15) shows that mean difference between HT1, HT2, HT4 with respect to HT3 are statically significant.

Game results show a very similar team performance for all the teams, with the exception of HT 3. This is explained by the absence of a player of type 1, which plays a very important role in midfield. However if a new defensive player is added (A4 in HT4) the performance increases again. In this case, the score difference improves as a result of the kicking devices of four of the players and their ability to obstruct and block any shots from the opponent. Performances for three of the heterogeneous teams are slightly higher than the best of the homogenous ones. However this mean difference is not statically significant. The number of goals scored and allowed show again very uniform values across all the teams. Among these teams, HT4, the team with the most defensive players, is the team that is able to score more goals during the evaluation period, but at the same time is the second team that allows more goals. The high number of goals allowed is correlated with the robot type A5, as H5 shows. The high number of goals scored is explained by the kicking devices of some of team members combined with a robot with a good dribbling skill, A3.

Team	Mean	Standard Deviation	Interval Estimation	Goals Scored	Goals Allowed	Goals Scored	Goals Allowed
HT1-11111	16	45.8	17.5 – 14.5	73.6	58.3	25.2	9.2
HT2-11201	16.4	46	17.9 – 14.9	75.8	57.1	25.7	9.3
HT3-01112	13.1	47.8	14.7 – 11.6	71.4	58.6	23.8	10.7
HT4-00122	16.5	46.9	18.0 – 15.0	74.8	58.4	27.1	10.6
Average	15.5	-	-	73.9	58.1	25.4	10.0

Table 7.15 Results for Heterogeneous Teams vs SchemaDemo

The global mean difference between the homogenous and heterogeneous teams is greater for the latter one, but after performing a *t-test* this difference can not be considered statistically significant, even though the *t-value* for this difference is slightly under the value that can be considered significant (1.91<1.99).

Team	GL	FB	RD	LD	MD	RM	LM	CF	ST	RW	LW
HT1-11111	1.0	0.05	0.10	0.08	1.00	0.79	0.85	0.08	0.35	0.35	0.36
HT2-11201	1.0	0.05	0.05	0.07	0.98	0.75	0.86	0.06	0.37	0.37	0.44
HT3-01112	1.0	0.06	0.09	0.07	0.99	0.81	0.80	0.08	0.36	0.35	0.40
HT4-00122	1.0	0.06	0.05	0.06	1.00	0.72	0.86	0.07	0.39	0.38	0.41

Table 7.16 Team Formations for Heterogeneous Teams vs SchemaDemo

Team formations (Table 7.16) are also very similar to those in the homogenous teams made up of fast players, but with the difference that again the heterogeneous teams present very uniform values, specially for the attacking positions. Unlike the homogenous teams where these values change considerably depending on the team.

Table 7.17 details the role distribution for each team. These results again are related to those of the homogenous teams and to the physical features of each player. The role of GL is again filled by big agents (A4 and A5) and also by A3. With respect to the previous example, agents of type A5 are not the ones that act more often as goalie. In this case, these agents are A3 and A4. The results on the homogenous teams show that this player is not among the best goalies. This opponent demands a goalie of big size, but also a player, which is able to move relatively fast. Although the presence of other types of agents help this player to improve its performance as goalie, agents A3 and A4 seems to meet these requirements. Offensive roles is filled by two different types of roles, players with dribbling skills (A2 and A3), as in the previous example or players with powerful kicking devices (A5). For the first type of agents they tend also to fill midfield roles, as these positions also require a good dribbling skill or a powerful kicking device. These midfield positions also allow excellent shooting chances to score goals by kicking the ball. For the MD role, a key role in the team structure, the results are similar to those of **BrianTeam**. On one hand A1 plays quite oft in this position, as it is a fast player, on the other hand players with powerful kicking devices (A5 A4) also tend to fill these roles.

Team	A1		A2		A3		A4		A5	
HT1-11111	MD	28%	LM	27%	GL	30%	GL	34%	GL	21%
	RM	23%	MD	17%	RM	18%	MD	16%	MD	32%
	LM	19%	RM	15%	ST	16%	RM	15%	RW	12%
HT2-11201	MD	35%	LM	26%	GL	35%			GL	21%
	LM	22%	RM	19%	LM	16%			MD	32%
	RM	21%	RW	15%	LW	15%			RW	10%
HT3-01112			MD	23%	RM	22%	GL	33%	GL	24%
			LM	23%	LM	19%	MD	17%	MD	22%
			RM	23%	RW	10%	LW	13%	RM	15%
HT4-00122					RM	23%	GL	28%	GL	20%
					LW	14%	MD	18%	MD	19%
					ST	13%	LM	15%	LM	18%

Table 7.17 Role Distribution for Heterogeneous Teams vs SchemaDemo

Team	Mean	Standard	Interval	Goals	Goals	Goals	Goals
		Deviation	Estimation	Scored	Allowed	Scored	Allowed
HT1-11111	-5.7	34.3	-4.4 - -7.0	26.2	80.9	7.1	12.8
HT2-11201	-8.7	47	-7.1 - -10.2	25.7	90.3	7.6	16.3
HT3-01112	-6.4	21.8	-5.4 - -7.5	19.2	69.1	4.5	11.0
HT4-00122	-4.2	15	-3.4 - -5.1	18.2	60.5	4.7	8.9
Average	-6.2			22.3	75.2	6.0	12.2

Table 7.18 Results for Heterogeneous Teams vs ou7kou

The Analysis of Variance for the hand-coded team **ou7kou** (Table 7.18) determines that the mean differences between HT4 and HT2, HT4 and HT3, HT1 and HT2 and HT3 and HT2 are significant. This difference is not significant, according to this test, for the rest of the possible pairs of means.

In this case the best teams are HT1 and HT4, the latter case offers the best performance, although it is not significant, but it is significantly worse when compared to the best of the homogenous teams. Results show that the number of goals tend to increase with the number of offensive players, but that the allowed goals also tend to decrease with teams of attacking players. In this case, the number of goals change in a similar way as in the case of the homogeneous robots, even though this change is not so considerable. This results show a considerably smaller difference between the best and the worst team (4.5<21.4) than in the homogeneous teams .

If the average of the means of the score margins are compared, the performance is statically significantly greater for the heterogeneous teams.

Team	GL	FB	RD	LD	MD	RM	LM	CF	ST	RW	LW
HT1-11111	1.0	0.05	0.89	0.83	0.75	0.28	0.44	0.07	0.15	0.26	0.28
HT2-11201	1.0	0.05	0.89	0.80	0.78	0.31	0.42	0.06	0.16	0.22	0.31
HT3-01112	1.0	0.06	0.90	0.82	0.82	0.40	0.47	0.08	0.09	0.20	0.17
HT4-00122	1.0	0.09	0.88	0.81	0.81	0.41	0.51	0.07	0.05	0.16	0.21

Table 7.19 Team Formations for Heterogeneous Teams vs ou7kou

The team formations (Table 7.19) present a similar pattern to those observed in the homogenous teams. Unlike the previous examples, the teams change their formation depending on the number of defensive skilled players present in the team. Teams tend to be more offensive, as the number of fast players increase in the team. This can be also observed in the homogeneous teams

Table 7.20 shows the role distribution for the heterogeneous teams. As in the previous examples this distribution is tightly related to the results of the homogenous teams. However, some differences can be observed. The role of goalie is again filled by robots of big size, A4 and A5, and sometimes also A3. Although the results for A3 and A4 show a high number of goals allowed, they also tend to fill these roles. The presence of other types of players in the team helps these players to improve their performance in this position. When playing against **ou7kou**, fast players tend to fill the defensive roles. This is explained by the fact that the opponent usually attacks trough the center field, allowing fast robots to move the ball towards midfield and starting an attacking play from this position. The role MD is in this example usually played by agents with powerful kicking devices; fast robots in this case do not show the same interest for this role, as in the previous examples. There are only two types of players that play in attacking positions. A3, although this role usage is only representative in the first team HT1, it can be also observed in the other teams (this information is not included in this table). Players of type A5 can also score goals from midfield positions.

Team	A1		A2		A3		A4		A5	
HT1-11111	LD	32%	RD	52%	GL	27%	GL	33%	GL	36%
	RD	32%	LD	16%	MD	16%	RD	25%	MD	26%
	MD	12%	RM	7%	RW	11%	MD	12%		
					LW	10%				
HT2-11201	LD	29%	RD	49%	GL	29%			GL	37%
	RD	29%	LD	16%	MD	13%			MD	28%
	MD	14%	MD	11%	LD	17%				
HT3-01112			RD	44%	RD	20%	GL	24%	GL	26%
			LD	24%	LD	16%	MD	16%	MD	22%
			MD	10%	GL	13%	LD	15%	RM	17%
HT4-00122					RD	20%	GL	24%	GL	25%
					LD	20%	RD	20%	MD	18%
					MD	17%	MD	13%	LD	12%

Table 7.20 Role Distribution for Heterogeneous Teams vs ou7kou

7.3.5 Results Summary

Results in this section show that heterogeneous teams perform globally better than the homogeneous ones. Statically tests have demonstrated this result against two teams (**BrianTeam** and **ou7kou**). Against **SchemaDemo** the *t-test* show that the difference is slightly under the significance level. However, in two cases (**BrianTeam** and **ou7kou**), the best of the homogeneous teams against each of these two teams performs significantly better than the best of the heterogeneous teams. In a third case, this difference can not be considered significant.

In the homogeneous teams, it can be observed that the best team against a given hand-coded team turns out to be the worst one against a different team. This is the case for H1, where it is the best team against **SchemaDemo** and the worst against **ou7kou**. In the heterogeneous teams, this is not observed, as these teams perform very similarly against the different hand-coded teams.

Heterogeneous teams also present more uniform results in the number of goals scored and allowed. The differences are not usually significant across all the different teams. However for the homogeneous ones, results show that the difference in the number of goals scored/allowed between the best and the worst team present very important differences.

From the point of view of team formations, the different heterogeneous teams converge to very similar team configurations when playing against the same opponent. In the homogeneous ones, team formation change depending on the features of the robots that compose the team.

Another feature observed in the results is that robots in heterogeneous team usually fill different roles depending on the opponent. This aspect is analysed more in depth in the experiments of the next section.

7.4 Learning vs Learning Teams

7.4.1 Introduction

This section studies the homogenous and heterogeneous teams when they play against each other. This analysis allows a better understanding of the benefits of heterogeneity when all these players play against a team that presents identical team strategies. Also the results of these games are used to determine which team is the best among the ones defined for these experiments. Later based on the results of these games a new team heterogeneous is built, which is a combination of the best of the heterogeneous and homogeneous teams. Team’s results are evaluated according to the following methodology:

- If the mean of the score is statistically significant above 0, a victory is awarded. **(W)**
- If the mean of the score is not statistically significantly above/below 0, a draw is awarded. **(D)**
- If the mean of the score is statistically significant below 0, a defeat is awarded. **(L)**

7.4.2 Homogeneous vs Homogeneous

The first group of teams evaluated is the homogeneous one. The result of these games give an idea of which is the best in this group of teams. Table 7.21 displays the results for these games. According to these results the best homogeneous team is H1. It is able to win 3 games and tie against A2. Teams that contain slower robots tend to perform worse.

Team	Mean	Standard Deviation	Interval Estimation	Goals Scored	Goals Allowed	Goals Scored	Goals Allowed
H1-H2	1.2	52.2	3.2 - -0.9	99.0	95.0	28.2	27.0
H1-H3	1.4	22.7	2.9 - 0.1	86.9	75.8	23.8	22.4
H1-H4	6.5	43.8	8.3 - 4.6	83.0	57.2	22.3	15.8
H1-H5	5.3	40.6	7.1 - 3.5	78.5	56.5	20.6	15.3
H2-H3	-1.9	21	-0.65 - -3.2	65.7	65.8	17.9	19.8
H2-H4	4.2	21.2	5.5 - 2.9	64.3	44.5	16.7	12.5
H2-H5	3.1	33	4.8 - 1.5	62.7	48.2	16.3	13.2
H3-H4	-0.2	15.3	1.3 - -0.9	37.8	36.1	8.3	8.5
H3-H5	3.3	17.9	4.5 - 2.1	49.9	30.7	10.9	7.6
H4-H5	-7.5	22.2	-6.4 - 8.6	25.1	49.3	4.9	12.4

Table 7.21 Results for Homogeneous Teams vs Homogeneous Teams

H1, the winning team among the homogeneous, is composed of very fast robots, which have problems to control of the ball, as they tend to oscillate as they move. This high speed allows this team to regain control of the ball in midfield, and attack continuously the opponent goal. In this set of experiments the key point in the strategy proves to be the control of midfield, teams that usually control this part of the field, are able to obtain good results. This table shows that the best team among the homogeneous ones is the one able to score more goals, but it is also one of the teams that allows more goals, as this type of player present very poor skills for defensive tasks. As teams are composed of slower and bigger robots, the number of scored goals decreases, but also the number of allowed goals, with some exceptions.

Team	W	L	D
H1-50000	3	0	1
H2-05000	2	1	1
H3-00500	2	1	1
H4-00050	1	3	0
H5-00005	1	3	0

Table 7.22 Global Results for Homogeneous Teams

Team	GL	FB	RD	LD	MD	RM	LM	CF	ST	RW	LW
H1-50000	1.0	0.07	0.12	0.12	0.75	0.47	0.51	0.08	0.31	0.77	0.82
H2-05000	1.0	0.04	0.07	0.13	1.01	0.26	0.33	0.07	0.43	0.82	0.85
H1-50000	1.0	0.07	0.12	0.12	0.75	0.47	0.51	0.07	0.31	0.77	0.81
H3-00500	1.0	0.04	0.07	0.13	1.00	0.26	0.33	0.07	0.43	0.82	0.85
H1-50000	1.0	0.09	0.06	0.05	0.86	0.58	0.52	0.08	0.62	0.56	0.59
H4-00050	1.0	0.14	0.12	0.18	0.95	0.35	0.30	0.14	0.39	0.67	0.76
H1-50000	1.0	0.10	0.12	0.13	0.86	0.58	0.65	0.10	0.47	0.46	0.53
H5-00005	1.0	0.09	0.12	0.13	0.96	0.41	0.38	0.14	0.30	0.79	0.68
H2-05000	1.0	0.05	0.11	0.15	0.99	0.51	0.48	0.09	0.28	0.63	0.71
H3-00500	1.0	0.05	0.09	0.13	1.02	0.40	0.37	0.12	0.26	0.82	0.74
H2-05000	1.0	0.08	0.14	0.10	0.97	0.54	0.51	0.11	0.30	0.63	0.62
H4-00050	1.0	0.07	0.08	0.19	1.02	0.58	0.41	0.10	0.16	0.68	0.71
H2-05000	1.0	0.07	0.10	0.14	0.97	0.67	0.68	0.10	0.37	0.45	0.45
H5-00005	1.0	0.08	0.16	0.10	1.03	0.49	0.54	0.10	0.26	0.63	0.61
H3-00500	1.0	0.12	0.12	0.12	1.03	0.77	0.78	0.11	0.16	0.36	0.43
H4-00050	1.0	0.09	0.11	0.13	1.04	0.58	0.59	0.12	0.20	0.64	0.50
H3-00500	1.0	0.09	0.14	0.10	0.97	0.65	0.69	0.13	0.37	0.47	0.39
H5-00005	1.0	0.19	0.20	0.13	0.96	0.57	0.62	0.13	0.12	0.59	0.49
H4-00050	1.0	0.12	0.17	0.14	1.04	0.83	0.79	0.15	0.17	0.29	0.30
H5-00005	1.0	0.12	0.11	0.10	0.96	0.56	0.53	0.19	0.22	0.67	0.54

Table 7.23 Team Formations for Homogeneous Teams vs Homogeneous Teams

Results in team final team formation show that the team configuration (Table 7.23) depends on the teams that are playing. Against small and fast players, H1 usually plays with a formation of one/two midfielder and two attackers, but when the game is played against a team composed of big and slow robots, this formation changes to a more defensive one, usually playing with two midfielders and one/two attackers. H2 and H3 also present a similar team formation when they play against the same teams. The formation for slow robots usually presents a very offensive team formation against almost every team. This is explained by the fact that these players are able to reach attacking positions, simply by kicking away the ball. In other cases the size of big robots causes physical interference, this makes robots play far from each other. It is also easier for these teams to score many goals, not just by dribbling, but kicking the ball from the wings

7.4.3 Heterogeneous 1 vs Homogeneous

Team	Mean	Standard Deviation	Interval Estimation	Goals Scored	Goals Allowed	Goals Scored	Goals Allowed
11111-50000	2.9	51.7	4.9 – 0.9	88.0	88.9	24.4	21.5
11111-05000	7.1	52.8	9.2 – 5.1	82.6	69.4	23.2	16.1
11111-00500	4.9	29.8	6.5 – 3.4	52.6	41.2	14.1	9.2
11111-00050	6.8	25.7	8.2 – 5.4	52.5	33.5	13.8	7.0
11111-00005	1.9	26.6	3.3 - 0.4	50.7	46.2	13.3	11.4

Table 7.24 Results for Heterogeneous 1 vs Homogeneous Teams

The first of the heterogeneous teams evaluated (Table 7.24) is able to win all games against the homogeneous teams. The score margin is very narrow when this team plays against H1 and H5. In both cases this is explained by the type of robots that compose each team, in the latter case the big size of the robots usually causes many problems to some of the teams, due to the physical interference in the field. The number of scored goals show that this team is able to score the same number of goals as H1 when playing against H2, but the goals scored tend to decrease and be quite different from H1 in the rest of the games. The main difference with respect to A1 lies in the number of allowed goals, as HT1 show better defensive skills. The score margin tends to improve with respect to A1 when playing against fast robots (A2, A3) but it falls for big robots (A5), as a result of the problem already mentioned, the physical interference

The team formation (Table 7.25) changes depending on the size of the opponents, in a similar way as in the case of homogenous robots (H1 and H2). As robots tend to move slower and its size increases, the formation tends to become more defensive. However, this team presents against all opponents a more defensive team formation than H1. While H1 is composed of fully offensive players, this team presents three relative fast players, but two players with powerful kicking devices. However, these robots do not allow this team to play using the same attacking strategy as the one it shows against H1, H2 and H3.

Team	GL	FB	RD	LD	MD	RM	LM	CF	ST	RW	LW
HT1-11111	1.0	0.08	0.13	0.12	0.99	0.29	0.35	0.12	0.33	0.81	0.78
H1-50000	1.0	0.07	0.15	0.14	0.88	0.51	0.54	0.08	0.31	0.63	0.69
HT1-11111	1.0	0.09	0.11	0.13	0.99	0.29	0.24	0.12	0.40	0.80	0.83
H2-05000	1.0	0.11	0.13	0.12	0.82	0.55	0.52	0.11	0.22	0.70	0.72
HT1-11111	1.0	0.11	0.15	0.13	1.02	0.58	0.55	0.13	0.22	0.61	0.50
H3-00500	1.0	0.07	0.15	0.17	1.03	0.68	0.66	0.11	0.08	0.58	0.47
HT1-11111	1.0	0.13	0.13	0.19	0.92	0.69	0.66	0.18	0.24	0.39	0.47
H4-00050	1.0	0.10	0.15	0.10	0.95	0.58	0.45	0.13	0.23	0.75	0.54
HT1-11111	1.0	0.11	0.15	0.11	0.80	0.80	0.85	0.13	0.30	0.32	0.32
H5-00005	1.0	0.10	0.13	0.11	0.97	0.49	0.44	0.12	0.36	0.67	0.59

Table 7.25 Team Formations for Heterogeneous 1 vs Homogeneous Teams

Table 7.26 displays the number of times, in percentage terms, that each robot has filled the different roles. A2 and A3 are the type of robots that usually fill the attacking roles, this is also observed in the games against the hand-coded teams. This is explained by the combination of a good dribbling skills and speed. Occasionally slow robots also fill these role, as their powerful kicking device allow them to score goals from outside the goal area, while A2 and A3 usually score goals by dribbling defenders and the goalie. Although A1 is also fit to play these kinds of roles (as HT1 shows), the competition for these roles is usually won by A2 and A3. For this reason A1 usually focuses on other roles that also demand dribbling skills and speed. One of the most interesting aspects in the results can be observed in A1 and A2 and their interest for MD. These two players tend to fill the role of MD as the opponent is composed of fast robots. This can be observed when playing against H1 and H2. As it has been already remarked, fast players tend to control the midfield when playing against other teams. However, this situation tends to change as they play against bigger robots. As the team are composed of bigger robots, A1 and A2 (specially for the former one falling from 40% to 10%), focuses on other roles, while slower and bigger robots fill this role. In these games the size and the kicking device become more important to control midfield.

Opponent	A1		A2		A3		A4		A5	
H1-50000	MD	33%	MD	27%	MD	24%	GL	81%	GL	19%
	LM	19%	RW	16%	RW	20%			MD	18%
	RW	15%	RM	16%	LW	20%			LM	11%
H2-05000	MD	38%	MD	46%	LW	30%	GL	42%	GL	55%
	RW	20%	RW	25%	RW	20%	LW	15%	RW	7%
	LW	21%	LW	12%	ST	11%	ST	10%		
H3-00500	MD	33%	MD	27%	MD	24%	GL	81%	GL	19%
	LM	20%	RM	16%	RW	20%			MD	18%
	RM	18%	RW	16%	LW	20%			LM	11%
H4-00050	LM	31%	MD	26%	LW	17%	GL	61%	GL	39%
	RM	24%	LM	22%	MD	17%	MD	16%	MD	21%
	MD	12%	LW	16%	RW	15%				
H5-00005	LM	30%	LM	24%	LM	23%	GL	65%	GL	35%
	RM	28%	RM	21%	MD	22%	MD	13%	MD	23%
	MD	12%	MD	20%	ST	14%			RW	7%

Table 7.26 Role Distribution for Heterogeneous 1 vs Homogeneous Teams

Finally A4 and A5 tend to fill the roles of goalie, especially A4, for which it seems to be better fit, unlike the games against some hand-coded teams. This is explained by the fact that this player combines speed and size, speed is very important for the goalie when the opponent attempts to dribble the goalie to score a goal. The attacking roles present this feature. However, size becomes more important when the opponent attempts to score goals by pushing, instead of dribbling (this is observed in most hand-coded teams).

7.4.4 Heterogeneous 2 vs Homogeneous

Team	Mean	Standard Deviation	Interval Estimation	Goals Scored	Goals Allowed	Goals Scored	Goals Allowed
01121-50000	1.2	38.8	2.3 - 0.1	90.1	82.5	22.9	21.6
01121-05000	3.3	38.5	5.0 - 1.5	77.1	70.2	20.9	17.6
01121-00500	0.6	34.5	2.2 - -1.1	56.7	56.3	15.0	14.4
01121-00050	5.8	38.6	7.6 - 4.1	52.0	36.4	13.6	7.8
01121-00005	2.1	29.1	3.6 - 0.5	53.1	47.5	13.2	11.1

Table 7.27 Results for Heterogeneous 2 vs Homogeneous Teams

This team is composed of four attacking players, and only one defensive player. Results (Table 7.27) show that this team is globally performing worse than the first heterogeneous team. The presence of a more offensive player does not represent an improvement in the team performance and the number of scored goals does not show any kind of improvement. This team is only able to win 4 games and tie one game. These are the games played against H3. This apparently contradictory result is explained by the fact that there are already in the team good attacking players. However, defensive players can help to the team in midfield and defence, better than players of type 3 do, by blocking shots from the opponent and making it difficult for the opponent to dribble.

Team formation (Table 7.28) shows a similar configuration to the one observed in HT1. When playing against fast teams the team formation is rather offensive, but as the robots become bigger this team formation tends to evolve towards a more defensive configuration.

Table 7.29 displays the role distribution for this team. The main difference that can be observed in the team formation with respect to the previous case (HT1) is that players of type A3 also fill the role of goalie. However, results show that the presence of this type of player in this role does not affect significantly the number of goals allowed. In this team, some of the results observed in previous cases are also present in the role distribution in this team. Fast agents tend to fill the roles (A1, A2) of MD, when the opponent tend to play also fast, while this position is filled by slower and bigger robots, as the opponent presents a bigger size and a powerful kicking device. Attacking roles, as in the previous example, are also filled by agents of type A2 and A3, while the rest of the team usually plays in midfield positions.

Team	GL	FB	RD	LD	MD	RM	LM	CF	ST	RW	LW
HT2-01121	1.0	0.08	0.15	0.13	0.99	0.33	0.34	0.11	0.30	0.80	0.77
H1-50000	1.0	0.07	0.13	0.11	0.87	0.51	0.48	0.10	0.35	0.73	0.65
HT2-01121	1.0	0.09	0.11	0.16	0.98	0.35	0.38	0.10	0.33	0.77	0.73
H2-05000	1.0	0.08	0.11	0.15	0.87	0.42	0.63	0.10	0.34	0.73	0.57
HT2-01121	1.0	0.10	0.13	0.11	0.91	0.62	0.60	0.13	0.29	0.56	0.55
H3-00500	1.0	0.07	0.13	0.09	0.87	0.40	0.55	0.13	0.38	0.67	0.71
HT2-01121	1.0	0.12	0.13	0.13	0.94	0.64	0.75	0.18	0.24	0.45	0.43
H4-00050	1.0	0.09	0.13	0.14	0.93	0.59	0.48	0.11	0.26	0.60	0.67
HT2-11120	1.0	0.14	0.16	0.12	0.90	0.77	0.81	0.13	0.32	0.32	0.32
H5-00005	1.0	0.11	0.12	0.11	0.97	0.42	0.42	0.14	0.29	0.74	0.68

Table 7.28 Team Formations for Heterogeneous 2 vs Homogeneous Teams

Opponent	A1		A2		A3		A4	A5	
H1-50000	MD	46%	MD	35%	GL	25%		GL	50%
	RW	14%	RW	26%	RW	19%			
	LW	14%	LW	20%	LW	18%			
H2-05000	MD	42%	MD	41%	GL	28%		GL	44%
	LW	18%	RW	26%	LW	20%			
	RW	12%	ST	10%	RW	17%			
H3-00500	RM	28%	MD	27%	GL	33%		GL	33%
	LM	23%	RM	16%	MD	14%		MD	26%
	RW	17%	RW	13%	RW	12%			
H4-00050	LM	27%	RM	24%	GL	26%		GL	48%
	MD	21%	LM	19%	MD	17%		MD	27%
	RM	19%	RW	21%	LW	13%			
H5-00005	LM	35%	RM	25%	GL	30%		GL	40%
	RM	21%	LM	23%	MD	16%		MD	29%
	MD	15%	RW	12%	RW	13%			

Table 7.29 Role Distribution for Heterogeneous 2 vs Homogeneous Teams

7.4.5 Heterogeneous 3 vs Homogeneous

Team	Mean	Standard Deviation	Interval Estimation	Goals Scored	Goals Allowed	Goals Scored	Goals Allowed
01112-50000	-0.3	45.6	1.7 - -2.2	76.3	81.5	21.8	22.1
01112-05000	1.4	39.0	3.1 - -0.4	58.7	66.5	17.1	15.7
01112-00500	0.2	44.9	1.8 - -1.4	51.1	51.7	12.7	12.5
01112-00050	4.3	20.6	5.6 - 3.0	41.3	30.4	9.8	5.5
01112-00005	-1.8	19.3	-0.5 - -3.0	42.9	45.4	9.8	11.6

Table 7.30 Results for Heterogeneous 3 vs Homogeneous Teams

The third heterogeneous team (Table 7.30) is only able to win one game, tie three and lose one. This team is composed of three rather defensive players. However, this feature does not result in an improvement in the number of goals allowed with respect to the previous heterogeneous teams. In most cases, the addition of this player results in a very small improvement. This is explained by the fact that there are already enough players to accomplish the tasks related to these players, when compared to HT1, and that the addition of a new defensive player only contributes to decrease the attacking skills of this team. Therefore this aspect results in a lower number of scored goals, leading this team to obtain very poor results against most of the opponents.

The team formation (Table 7.31) presents very similar role distribution to those observed in the previous examples. The team tends to present a more offensive team formation when playing against fast robots than against defensive ones.

Team	GL	FB	RD	LD	MD	RM	LM	CF	ST	RW	LW
HT3-01112	1.0	0.12	0.15	0.12	0.93	0.37	0.33	0.12	0.26	0.83	0.77
H1-50000	1.0	0.06	0.10	0.07	0.88	0.57	0.57	0.12	0.45	0.60	0.58
HT3-01112	1.0	0.10	0.18	0.12	1.0	0.35	0.42	0.12	0.13	0.81	0.77
H2-05000	1.0	0.8	0.12	0.11	0.94	0.54	0.53	0.08	0.32	0.67	0.61
HT3-01112	1.0	0.10	0.15	0.15	1.00	0.58	0.63	0.14	0.17	0.55	0.54
H3-00500	1.0	0.07	0.10	0.14	1.00	0.39	0.53	0.11	0.31	0.71	0.65
HT3-01112	1.0	0.14	0.15	0.13	0.96	0.65	0.76	0.20	0.19	0.43	0.39
H4-00050	1.0	0.10	0.12	0.15	0.97	0.70	0.61	0.19	0.20	0.35	0.61
HT3-01112	1.0	0.13	0.11	0.12	0.98	0.75	0.77	0.16	0.25	0.40	0.34
H5-00005	1.0	0.11	0.12	0.11	0.97	0.46	0.59	0.16	0.23	0.70	0.56

Table 7.31 Team Formations for Heterogeneous 3 vs Homogeneous Teams

Opponent	A1	A2		A3		A4		A5	
H1-50000		RW	33%	RW	33%	GL	54%	GL	23%
		LW	33%	LW	17%			MD	30%
		MD	11%	MD	11%			LW	10%
H2-05000		MD	32%	RW	41%	GL	55%	GL	45%
		LW	25%	MD	22%			MD	23%
		RW	21%					LW	17%
H3-00500		MD	37%	MD	35%	GL	42%	GL	28%
		LM	19%	RW	20%	LM	16%	LW	15%
		RM	17%	LM	15%	RM	15%	MD	10%
H4-00050		LM	23%	LM	29%	GL	64%	MD	23%
		MD	22%	RM	21%	MD	17%	GL	16%
		RM	21%	RW	17%			LM	11%
H5-00005		LM	26%	RM	28%	GL	52%	MD	25%
		MD	27%	LM	26%			GL	21%
		RM	16%	RW	14%			RM	12%

Table 7.32 Role Distribution for Heterogeneous 3 vs Homogeneous Teams

The role distribution (Table 7.32) presents some significant changes with respect to the previously studied examples. When playing against H1, the absence of a player of type A1 allows players of type A2 and A3 to continue to fill the attacking roles, while in midfield this player is replaced by a player of type A5. This leads to a lower team performance, as a less fitted player fills this position. Although for fast opponents, the presence of players of types A2 and A3 in offense is relatively important, the attacking roles are also filled by players of type A5. Although this allows the team to score goals from outside the area, against this kind of opponents is more effective to score goals by dribbling the goalie and defenders. As the team plays against teams with bigger robots, the role distribution presents a similar pattern, as the one seen in previous examples. Big robots tend to fill the role MD, although fast robots still continue to play this role, while the robot of type 2 and 3 fill other roles in midfield, sometimes also playing in attacking positions. Agents of type 4 and 5 fill the role of goalie, and as in the previous cases A4 seems to be better fitted for this role.

7.4.6 Heterogeneous 4 vs Homogeneous

Team	Mean	Standard Deviation	Interval Estimation	Goals Scored	Goals Allowed	Goals Scored	Goals Allowed
00122-50000	-3.4	56.2	-1.3 - -5.5	63.8	80.0	17.9	21.3
00122-05000	-1.1	38.7	0.7 - -2.9	53.2	63.4	14.7	15.8
00122-00500	5	23.8	6.4 - 3.6	47.5	39.6	12.6	8.6
00122-00050	7.6	34.6	9.3 - 6.0	50.2	24.7	12.4	4.8
00122-00005	-1	27.6	0.5 - -2.5	40.7	42.1	9.3	10.3

Table 7.33 Results for Heterogeneous 4 vs Homogeneous Teams

The fourth heterogeneous team (Table 7.33) is composed of four big size robots and a good attacking player. Despite adding a new defensive player, this team performs better than HT3, it wins 2 games, loses 1 games and tie 2. This team performs relatively well against teams of middle size and middle speed, while for fast and big size robots it only can tie, and loses against the fastest one. The reason for this improvement is that the presence in the field of an important number of big size robots can make it difficult for the opponent to get to the own penalty area. This team presents the best results in defence, although the difference between the number of goals allowed between this team and HT1 is not very significant. On the other hand this improvement in defence does not allow this team to score more goals in offense than other teams. In some of the games, this decrease in the number goals is very significant with respect to HT1.

The team formation (Table 7.34) presents some interesting variations with respect to other heterogeneous teams analysed. This team usually presents a very similar formation to the one observed in H5, playing with one goalie, two attackers and only two midfielders. The most interesting change can be observed in the game against H5. Unlike other cases where H5 has in average 1.5 players in offense against the different types of opponent, both teams, when playing against each other, present in their team formation only one attacker. This is as a result of the size of the robots in each team. These two teams tend to block the shots from each other, and therefore they can not take advantage of their kicking devices. In addition, they also physically interfere with other players in the field.

Team	GL	FB	RD	LD	MD	RM	LM	CF	ST	RW	LW
HT4-00122	1.0	0.10	0.13	0.14	0.95	0.42	0.39	0.12	0.27	0.75	0.74
H1-50000	1.0	0.07	0.10	0.06	0.84	0.62	0.55	0.09	0.49	0.54	0.64
HT4-00122	1.0	0.12	0.13	0.13	0.95	0.35	0.41	0.16	0.27	0.75	0.72
H2-05000	1.0	0.08	0.11	0.12	0.88	0.61	0.65	0.13	0.33	0.56	0.53
HT4-00122	1.0	0.1	0.13	0.15	0.95	0.43	0.45	0.13	0.19	0.78	0.69
H3-00500	1.0	0.09	0.16	0.15	1.02	0.78	0.71	0.13	0.23	0.34	0.39
HT4-00122	1.0	0.11	0.13	0.13	0.93	0.50	0.46	0.15	0.38	0.63	0.58
H4-00050	1.0	0.13	0.19	0.15	0.96	0.75	0.73	0.16	0.19	0.36	0.38
HT4-00122	1.0	0.10	0.12	0.10	0.93	0.69	0.75	0.13	0.39	0.43	0.35
H5-00005	1.0	0.13	0.12	0.14	0.98	0.65	0.76	0.16	0.19	0.46	0.41

Table 7.34 Team Formations for Heterogeneous 4 vs Homogeneous Teams

The role distribution (Table 7.35) shows, as in the previous examples. that the agent of type A3 fills the attacking roles, specially the role of ST. As robots increase in size this agent plays more in midfield positions. The role of MD is played by agents of type A4 and type A5, while one of type A4 also fills one of the attacking roles. The lack of offensive players when playing against fast teams prevents this team very often from reaching the opponent area, as the fastest players of the rival control midfield. Also this problem also affects the attacking roles, while one of this roles is usually filled by a good attacking robot (A3), one of the usually two attacking positions is played by a robot of type 4, and sometimes also of type 5. In this team, it is not clear which is the best player in the role of GL, as both players, globally filled this role the same number of times.

Opponent	A1	A2	A3		A4		A5	
H1-50000			LW	40%	GL	31%	MD	19%
			RW	22%	MD	17%	GL	19%
			ST	10%	RW	15%	RM	12%
H2-05000			LW	34%	MD	26%	GL	40%
			RW	25%	RW	18%	MD	12%
			ST	9%	LW	11%	LM	8%
H3-00500			LW	34%	GL	34%	GL	16%
			RW	24%	RW	15%	MD	34%
			LM	12%	LW	13%	LW	13%
H4-00050			LW	25%	GL	27%	MD	30%
			MD	14%	MD	24%	GL	17%
			LM	14%	LM	11%	RW	11%
H5-00005			MD	29%	GL	22%	GL	28%
			RM	16%	MD	28%	MD	14%
			LM	14%	LM	16%	LM	15%

Table 7.35 Role Distribution for Heterogeneous 4 vs Homogeneous Teams

7.4.7 Heterogeneous vs Heterogeneous

Table 7.36 shows the results for the games among the heterogeneous robots. The best team in this set of games is the fully heterogeneous one, which is able to win all games. The second one is the team with more offensive players, HT2. While the most defensive team, HT4, is able to win the third homogeneous team, HT3. Although HT1 is the team that is able to win the three games, the second team wins the game against the most defensive one with the widest score margin, and significantly greater than the one achieved by HT1. In this particular case the presence of two players of type 3 helps this team to obtain a better result. The result between the HT3 and HT4 is explained by the fact that the fourth heterogeneous robots is able to obtain a slightly better result when playing against teams composed of big size robots and has better defensive skills.

Team	Mean	Standard Deviation	Interval Estimation	Goals Scored	Goals Allowed	Goals Scored	Goals Allowed
11111-11201	3.2	52	5.3 - 1.2	70.3	59.9	17.4	14.2
11111-01112	5.2	53	7.3 - 3.2	68.0	52.2	17.4	12.2
11111-00122	5.6	39.4	7.3 - 3.8	65.6	51.3	17.3	11.8
11201-01112	2.9	54.3	5.0 - 0.8	59.6	51.1	15.6	12.7
11201-00122	8.0	28.2	9.5 - 6.5	58.1	39.1	15.0	7.0
01112-00122	-2.2	36.5	-0.5 - -3.9	47.0	61.6	12.5	14.7

Table 7.36 Results for Heterogeneous vs Heterogeneous

Team formation (Table 7.38) in these games do not present very important variations, unlike other games, where the team formation depends on the size of the opponent; but there are also some exceptions. While HT1 presents a very attacking team formation, HT2 plays with two players in midfield. Despite consisting of 4 attacking players, the presence of one big size robot in midfield does not require additional players, while the second requires an additional player to play in this key area. The second significant difference can be found in the game between HT3 and HT4. Although HT4 seems to be the most defensive team among the heterogeneous ones, it presents the most offensive team formation of all teams analysed in this section. This is explained by the fact that this team consist of 4 big size robots fitted with powerful kicking devices, while HT3 team consists only of three robots with these features.

Team	W	L	D
HT1-11111	3	0	0
HT2-11201	2	1	0
HT3-00121	1	2	0
HT4-01112	0	3	0

Table 7.37 Global Results for Heterogeneous Teams

Team	GL	FB	RD	LD	MD	RM	LM	CF	ST	RW	LW
HT1-11111	1.0	0.08	0.14	0.12	0.97	0.35	0.36	0.15	0.29	0.74	0.81
HT2-11201	1.0	0.15	0.17	0.16	0.88	0.48	0.60	0.14	0.20	0.56	0.66
HT1-11111	1.0	0.10	0.11	0.11	0.97	0.41	0.45	0.15	0.30	0.69	0.71
HT3-01112	1.0	0.19	0.17	0.23	0.92	0.45	0.49	0.12	0.14	0.68	0.70
HT1-11111	1.0	0.07	0.09	0.08	0.95	0.35	0.62	0.14	0.35	0.69	0.66
HT4-00122	1.0	0.23	0.18	0.20	0.86	0.53	0.48	0.09	0.18	0.61	0.62
HT2-11202	1.0	0.08	0.11	0.11	0.98	0.53	0.49	0.11	0.26	0.66	0.68
HT3-01112	1.0	0.13	0.17	0.19	0.97	0.51	0.53	0.16	0.15	0.60	0.59
HT2-11201	1.0	0.10	0.12	0.10	0.97	0.52	0.47	0.14	0.31	0.63	0.64
HT4-00122	1.0	0.09	0.21	0.26	1.02	0.65	0.65	0.11	0.18	0.51	0.42
HT3-01112	1.0	0.11	0.20	0.19	0.99	0.60	0.54	0.12	0.17	0.53	0.55
HT4-00122	1.0	0.12	0.17	0.20	0.75	0.33	0.38	0.16	0.26	0.79	0.84

Table 7.38 Team Formations for Heterogeneous Teams

Role distribution (Table 7.39) also presents some variations with respect to other analysed games. In the first game, between the two teams with fast robots (HT1-HT2), the difference in role distribution in midfield is very notorious. While in the fully heterogeneous team, the role MD, one of the key roles in a team, is filled by the fastest robot types (A1 and A2), the same role is only filled by a big size robot and robots of type 3. In this case fast players tend to play in the roles of left/right midfielder and left/right wingers. In previous games analysed for this team (HT2), results show that fast players (A1 and A2) tend to fill the role of MD, when playing against teams with fast players. Although HT1 can be considered also a team of fast robots, the combination of a fast robot and a big size robot in midfield (A4/A5) turns out to be very efficient and allows the control of midfield, leading to HT2 to play in this area with a different strategy. This team formation in midfield for HT2 can also be observed against the rest of the heterogeneous opponents. This can also be explained by the combination of different skills. For HT3 and HT4 role distribution among their team members do not present very important changes. Fast robots tend to fill the roles of attackers, while big size robots tend to act as goalie or play in midfield.

Team	A1		A2		A3		A4		A5	
HT1-11111	MD	39%	MD	39%	LW	28%	GL	54%	GL	46%
	LW	18%	RW	24%	RW	20%	LW	15%	RM	12%
	RW	13%	LW	15%	ST	13%				
HT2-11201	RM	27%	RW	25%	GL	24%			GL	50%
	MD	20%	LW	18%	LW	20%			MD	27%
	LM	16%	RM	17%	MD	15%				
HT1-11111	MD	45%	MD	35%	RW	27%	GL	58%	GL	42%
	RW	15%	LW	21%	LW	19%	LW	16%	MD	11%
	RM	12%	RW	18%	LM	17%				
HT3-01111			LW	22%	RW	23%	GL	47%	MD	30%
			RW	21%	LW	14%	LW	11%	GL	25%
			RM	16%	MD	13%	RW	11%		
HT1-11111	LW	25%	MD	44%	RW	24%	GL	59%	GL	41%
	MD	21%	ST	13%	LW	22%	LM	10%	LM	12%
	RW	19%	LW	10%	LM	20%				
HT4-00122					RW	21%	GL	25%	MD	38%
					LW	18%	LW	18%	GL	25%
					LM	15%	LM	14%		
HT2-11201	LW	22%	RW	18%	MD	24%			GL	44%
	LM	18%	MD	16%	GL	15%			MD	22%
	RM	17%	RM	14%	RW	15%				
HT3-01112					MD	26%	GL	24%	GL	26%
					RW	22%	MD	25%	LM	13%
					LW	17%	RW	13%	LW	10%
HT2-11201	MD	28%	RW	22%	GL	32%			GL	36%
	RM	18%	LM	22%	MD	16%			MD	16%
	LW	17%	MD	21%	LW	24%				
HT4-00122					RW	20%	GL	29%	GL	20%
					RM	18%	MD	17%	MD	26%
					LW	16%	RM	11%	RM	14%
HT3-01112			MD	38%	MD	27%	GL	24%	GL	26%
			LM	15%	RW	20%	MD	15%	MD	13%
			RM	14%	RM	18%	RM	15%	LW	10%
HT4-00122					MD	24%	GL	27%	GL	23%
					RW	23%	LW	15%	LW	15%
					LW	23%	RW	15%	MD	12%

Table 7.39 Role Distribution for Heterogeneous Teams

7.4.8 Final Results

After all the games have been played, the classification (Table 7.40) shows that among the four best teams, there are 3 heterogeneous ones, while HT3 presents very poor results. In general, teams composed of fast robots tend to perform better than slow robots, in the games analysed. However, HT4 performs among the best, even though is composed of slow robots, and in particular of two robots of type 4, which are not so good as a team when compared to other types of players. This also can be observed in HT1, which contains a player of type 4, but it performs much better than the rest of the teams,

Team	W	L	D	Points
HT1	8	0	0	24
HT2	6	1	1	19
H1	4	2	3	15
HT4	3	3	2	11
H2	2	3	3	9
H3	2	3	3	9
H5	2	5	1	7
HT3	1	4	3	6
H4	1	7	0	3

Table 7.40 Global Results for Homogeneous and Heterogeneous Teams

Using the results on the individual team performances, a new team is created (HT5), so that it can beat the best of the teams of the designed so far. The new team is a team composed of two player of type A1, one player of type A2, one player of type A3 and finally a player of type A5. This team includes a new player of type A1, as this team has turned out to be the best among the homogeneous teams. This new team is tested against the hand-coded teams and all learning teams.

Team	Team Code	Player 1	Player 2	Player 3	Player 4	Player 5	Diversity Value	Team Profile
HT5	21101	A1	A1	A2	A3	A5	1.07	0.04

Table 7.41 Heterogeneous 5 Team Compositon

7.4.9 Heterogeneous 5 vs All Teams

Results displayed table 7.42 show that this new heterogeneous team, perform slightly worse than HT1 against the hand-coded teams. However, the only significant mean difference can be observed in the games against **ou7kou**. The number of goals scored and allowed present very small changes with the exception of the results for **ou7kou**. The number of scored goals as well as the number of allowed goals are greater for HT5, but this is especially significant for the number of goals allowed. This result is explained by the introduction of an offensive player, which helps the team to improve its offensive skills, but this also results in a very poor performance in defence.

Opponent Team	Mean	Standard Deviation	Interval Estimation	Goals Scored	Goals Allowed	Goals Scored	Goals Allowed
BrianTeam	71.1	83.8	73 - 69	233.0	25.0	73.3	2.2
SchemaDemo	14.9	42.6	16.4 - 13.5	77.1	59.8	24.8	9.9
ou7kou	-11.3	56	-9.7 - -12.9	32.0	102.7	8.5	19.8

Table 7.42 Results for Heterogeneous 5 vs Hand-Code Teams

Results for HT5 against the homogeneous and heterogeneous teams (Table 7.43) show that this team is able to win against all the homogeneous and heterogeneous teams, with the exception of H1, against which it is only able to tie. This team presents a very similar performance to the one observed in HT1 when playing against homogeneous teams, with the exception of H5, against which HT5 performs much better. However, against the heterogeneous teams, the score margin is greater and in some cases significantly better than for HT1. The results for the games between HT1 and HT5 show that the latter one is able to win by a significant goal margin. From the point of view of the number of goals scored/allowed, both teams present similar values, with some exceptions. In the games against HT3 and HT4, this team is able to score more goals and especially for the latter case, where it allows significantly fewer goals than HT1. In this team, the introduction of a more offensive player, replacing a defensive one, does not result in an important increase in the number of allowed goals. However, against some teams this results in a greater number of goals scored, quite the opposite of HT2, where the introduction of a more offensive player does not result in a better team performance. This result is explained by the fact that players of type A1 not only play well in attacking positions, but also in midfield unlike A3.

Team	Mean	Standard Deviation	Interval Estimation	Goals Scored	Goals Allowed	Goals Scored	Goals Allowed
21101-50000	1.4	47.2	3.4 - -0.5	87.4	93.9	23.8	22.4
21101-05000	5.5	31.1	7.1 - 3.9	81.1	74.9	22.0	16.6
21101-00500	4.6	45.8	6.5 - 2.7	65.1	55.6	18.3	13.7
21101-00050	6.9	38.4	8.7 - 5.1	60.9	38.1	15.4	8.6
21101-00005	5.3	31.3	6.9 - 3.7	65.3	45.1	17.0	11.7
21101-11111	3.0	28.9	4.6 - 1.5	65.4	60.5	17.4	14.3
21101-11201	4.1	40	5.8 - 2.6	69.0	60.9	18.6	14.5
21101-01112	7.1	30.6	8.7 - 5.6	66.2	49.3	18.1	11.0
21101-00122	10.2	29.0	11.7 - 8.7	40.3	66.1	18.1	7.9

Table 7.43 Results for Heterogeneous 5 vs All Teams

The team formations (Table 7.44) against the hand-coded teams does not show very important variations with respect to results observed in HT1. Only the role configuration for **BrianTeam** shows a slightly greater use of attacking roles.

Opponent Team	GL	FB	RD	LD	MD	RM	LM	CF	ST	RW	LW
BrianTeam	1.0	0.06	0.05	0.09	0.94	0.75	0.90	0.04	0.86	0.10	0.21
SchemaDemo	1.0	0.07	0.07	0.14	0.98	0.75	0.85	0.07	0.29	0.35	0.43
ou7kou	1.0	0.05	0.89	0.81	0.74	0.36	0.43	0.06	0.15	0.24	0.27

Table 7.44 Team Formations for Heterogeneous 5 vs Hand-Coded Teams

Teams	GL	FB	RD	LD	MD	RM	LM	CF	ST	RW	LW
HT5-21101	1.0	0.07	0.17	0.11	0.99	0.38	0.35	0.10	0.24	0.83	0.76
H1-50000	1.0	0.07	0.15	0.12	0.96	0.37	0.44	0.09	0.27	0.83	0.70
HT5-21101	1.0	0.09	0.12	0.15	0.99	0.33	0.25	0.12	0.38	0.78	0.79
H2-05000	1.0	0.05	0.18	0.22	0.99	0.41	0.42	0.09	0.21	0.74	0.69
HT5-21101	1.0	0.11	0.13	0.11	0.97	0.47	0.48	0.15	0.28	0.65	0.65
H3-00500	1.0	0.06	0.14	0.16	1.01	0.39	0.39	0.10	0.23	0.76	0.75
HT5-21101	1.0	0.12	0.14	0.15	0.97	0.49	0.62	0.17	0.20	0.63	0.51
H4-00050	1.0	0.08	0.18	0.15	1.03	0.46	0.48	0.10	0.18	0.66	0.68
HT5-21101	1.0	0.12	0.15	0.14	0.95	0.55	0.56	0.14	0.32	0.55	0.52
H5-00005	1.0	0.11	0.16	0.16	0.96	0.38	0.47	0.15	0.22	0.75	0.64
HT5-21101	1.0	0.09	0.10	0.15	0.98	0.38	0.41	0.12	0.26	0.74	0.77
HT1-11111	1.0	0.10	0.14	0.20	1.00	0.47	0.49	0.11	0.22	0.65	0.62
HT5-21101	1.0	0.08	0.13	0.14	0.99	0.32	0.38	0.12	0.25	0.81	0.78
HT2-11201	1.0	0.09	0.17	0.21	0.99	0.48	0.41	0.14	0.19	0.67	0.66
HT5-21101	1.0	0.09	0.14	0.12	0.96	0.41	0.36	0.15	0.31	0.71	0.75
HT3-01112	1.0	0.12	0.22	0.18	1.02	0.42	0.56	0.13	0.12	0.63	0.60
HT5-21101	1.0	0.11	0.13	0.17	0.96	0.45	0.52	0.14	0.22	0.67	0.63
HT4-00122	1.0	0.14	0.14	0.19	0.97	0.63	0.52	0.12	0.15	0.59	0.55

Table 7.45 Team Formations for Heterogeneous 5 vs All Teams

Team formations against the learning teams (Table 7.45) present a very similar formation to the one observed in teams composed of rather offensive players. The role usage tends to increase for midfield and defensive roles, as the number of defensive players in the opponent increases. This phenomenon has been also observed in other teams. However, the team formation against the defensive teams (H5, HT3 and HT4) presents a more offensive configuration than for other similar types of teams.

The role distribution (Table 7.46) does not present very important changes with respect to HT1, with the exception of the agent of type 3 that also plays as goalie. For this type of player this is especially significant for **SchemaDemo** and **ou7kou**, teams that present good attacking skills, require also fast goalkeepers. A1 plays usually as right/left midfielder/defender. When this team plays against **SchemaDemo**, A1 fills the role of MD, this aspect has been also observed in HT1. Agents of type A2 and A3, tend to fill also the attacking roles, against the easiest teams, even though A3 in this case also acts as GL, decreasing the possibility of playing in attacking positions.

Opponent Team	A1		A2		A3		A4	A5	
BrianTeam	LM	28%	ST	24%	ST	29%		GL	58%
	RM	22%	LM	22%	GL	27%		MD	26%
	MD	20%	MD	17%	RM	14%			
SchemaDemo	MD	30%	GL	21%	GL	45%		GL	23%
	RM	22%	LM	20%	LM	15%		MD	23%
	LM	18%	RM	16%	LW	11%		LM	15%
ou7kou	RD	36%	LD	21%	GL	41%		GL	42%
	LD	23%	MD	21%	LW	10%			
	LM	12%	RD	14%	RW	8%			

Table 7.46 Role Distribution for Heterogeneous 5 vs Hand-Coded Teams

Opponent Team	A1		A2		A3		A4	A5	
H1-50000	MD	26%	MD	31%	GL	40%		GL	60%
	LW	24%	RW	25%	RM	18%		RW	10%
	RW	21%	LW	20%					
H2-05000	MD	29%	MD	30%	GL	45%		GL	55%
	RW	24%	LW	24%	RW	10%			
	LW	23%	LW	18%					
H3-00500	MD	26%	MD	31%	GL	76%		GL	24%
	RW	20%	LW	21%				RW	25%
	LW	19%	LM	16%				LW	14%
H4-00050	MD	31%	MD	22%	GL	63%		GL	37%
	LM	19%	RW	23%					
	RW	15%	LW	20%					
H5-00005	MD	32%	LW	26%	GL	61%		GL	39%
	RM	17%	MD	21%				RW	10%
	LM	15%	RW	15%					

Table 7.47 Role Distribution for Heterogeneous 5 vs Homogeneous Teams

Role distribution against the homogeneous teams (Table 7.47) shows that players of type A1 and A2 control midfield in all cases. Unlike HT1, where the types of players that fill this role tend to change depending on the number of big size robots in the opponent team. Against these teams, these players are also responsible for offensive positions, while players of type A3 and

A5 tend to fill the role of goalie. Although A3 does not seem to be the best type of robot to act as goalie, results show that the number of goals allowed do not increase significantly.

Teams	A1		A2		A3		A4		A5	
HT5-21101	MD	29%	LW	27%	GL	50%			GL	50%
	RW	22%	MD	27%	LW	10%				
	LW	18%	RW	20%	RW	10%				
HT1-11111	MD	32%	MD	29%	MD	24%	GL	62%	GL	37%
	RM	16%	RW	22%	LW	22%			LM	10%
	LW	15%	LW	12%	RW	26%				
HT5-21101	RW	29%	MD	34%	GL	49%			GL	51%
	MD	23%	LW	26%						
	LW	20%	RW	15%						
HT2-11201	MD	41%	MD	27%	GL	28%			GL	40%
	LW	17%	RW	22%	LW	17%			LM	10%
	RM	12%	RM	12%	RW	14%				
HT5-21101	MD	32%	LW	33%	GL	59%			GL	41%
	RW	20%	MD	23%	LW	11%				
	LW	15%	RW	19%						
HT3-01112			MD	29%	RW	21%	GL	66%	MD	27%
			RW	20%	LM	19%	LM	12%	GL	17%
			RM	18%	LW	17%			LW	11%
HT5-21101	MD	30%	MD	24%	GL	57%			GL	43%
	RW	18%	LW	22%						
	LW	15%	RW	19%						
HT4-00122					MD	34%	MD	25%	GL	21%
					LW	20%	GL	27%	MD	15%
					RW	17%	LM	14%	LM	11%

Table 7.48 Role Distribution for Heterogeneous 5 vs Homogeneous Teams

Team	W	L	D	Points
HT5	8	0	1	25
HT1	8	1	0	24
HT2	6	2	1	19
H1	4	2	4	16
HT4	3	4	2	11
H2	2	4	3	9
H3	2	4	3	9
H5	2	6	1	7
HT3	1	5	3	6
H4	1	8	0	3

Table 7.49 Global Results for Homogeneous and Heterogeneous Teams

Against the heterogeneous teams (Table 7.48), role distribution is also very similar to the one observed in HT1, with A3 showing a greater interest for attacking roles. The other heterogeneous teams present also a similar role distribution against this type of teams. However, in this case HT2 do no present the same team configuration as the one observed against HT1, where fast robots do not show a similar interest for the roles in midfield, as a result of the combination of players of different type in midfield.

Once the team classification is updated (Table 7.49) with the results of the new games, HT5 replaces HT1 at the top of this classification, but leading only by 1 point.

7.4.10 Results Summary

Results presented in this section have shown that the three best teams are heterogeneous, and that among the five best teams, four are also heterogeneous. Among the homogeneous teams, the teams composed of fast players tend to perform better, as the speed of the robots allows them to have a better control of midfield. The better performance observed in heterogeneous teams is explained by two factors:

- Players with different skills can perform more efficiently than teams of homogeneous teams, as the skills required for the roles are different. Teams of heterogeneous members may present players better fitted for the different roles than homogeneous ones, resulting in an improvement in team performance. While big size robots can perform quite well in defensive positions, fast robots tend to play better in attacking positions. If these players are replaced by intermediate robots (middle size and average speed), even though their average skills are the same, they do not perform in attacking and defensive positions as well as a combination of fast and small-size robots and slow and big size robots.

- Results show that role distribution in the heterogeneous teams change depending on the opponent. This is explained by the fact that the skills required to play in a given position depend on the environment, in this case the opponent. This can be observed in the role of MD, one of the key roles of the team, which is filled in some cases by a player of type A1 (small size and fast), when playing against teams composed of fast robots. However, this position in the team is filled by a robot of type A5 (big size and slow) when the same team plays against an opponent composed of more defensive players. This feature is especially beneficial for heterogeneous teams, as they present different skills that can be recombined in different ways, in order to adapt to the opponent.

On the other hand, results also reflect that the design of a heterogeneous team based on the results of the homogeneous teams is not a trivial task. Results show that the addition of players that individually may perform very poor, may result in an improvement in the team performance when compared to teams of theoretically better individual robots. This result can be observed in the game between HT1 and HT2, and HT3 and HT4. The presence of a poorly skilled robot with other better robots results in some cases in a better team performance, as a result of the combination of two different types of skills in a field area, especially interesting in midfield, as the results show for the games between HT1 and HT2. However, the combination of fast players (two/three) and defensive players (two/three) usually results in a better team performance than the homogeneous ones.

Although heterogeneous teams perform globally better, the number of goals that a heterogeneous team scores is significantly lower than the number of goal scored by the best of the homogeneous teams. This is especially true, when H1 is compared with the heterogeneous teams. From the point of view of the number of goals allowed, heterogeneous teams tend to allow fewer goals than homogeneous teams, even though they can be composed of one or two defenders. This is explained by the control of midfield that keeps the opponent away from the penalty area.

Team formation usually change depending on the opponent and the type of players in the team. This has also been observed against the hand-coded players, but in this set of experiments these variations can be better perceived.

7.5 Unbalanced Heterogeneous Team

7.5.1 Introduction

This section analyses the heterogeneity from a different point of view. So far robots have been built, so that they present balanced features. In this situation, all robots contribute to the overall team performance with their skills.

This example analyses the impact of heterogeneity, when there is a subset of team members that are clearly better than other players. In this example, the robots selected are identical to the robots in HT1, but with the difference that the radius is set at 6 cm for all the robots. The resulting team consists of 2 very good skilled robots (1 and 2), two relatively poor skilled robots (4 and 5) and fifth one (3) that presents the same skills as before. This team is tested against the three hand-coded teams selected for this thesis, and later, against all the homogeneous and heterogeneous teams. This team is denoted in this section as HT1s, and the different players also are denoted as *Ans*.

Team	Team Code	Player 1	Player 2	Player 3	Player 4	Player 5	Diversity Value	Team Profile
HT1s	11111	A1s	A2s	A3s	A4s	A5s	1.4	0.09

Table 7.50 Unbalanced Heterogeneous Team Compositon

7.5.2 Experiments

In all three games against the hand-coded teams (Table 7.51), this team performs worse than the equivalent heterogeneous balanced team. In addition, the difference in mean with respect to HT1 for **BrianTeam** and **ou7kou** are statically significant. If these results are compared to all designed heterogeneous teams, in most cases this team performs worse than these teams and in many cases this difference is also statically significant. If these results are compared to the ones obtained by homogeneous teams, this team performs approximately like the average homogeneous team.

Opponent Team	Mean	Standard Deviation	Interval Estimation	Goals Scored	Goals Allowed	Goals Scored	Goals Allowed
BrianTeam	64.7	108	67 - 62	226.0	22.8	67.7	3.0
SchemaDemo	14.6	38.5	16 - 13.2	199.0	61.8	25.2	10.6
ou7kou	-12.2	46	-10.6 - -13.7	27.0	110.1	8.3	20.5

Table 7.51 Results for Unbalanced Heterogeneous vs Hand-Coded Teams

Team	Mean	Standard Deviation	Interval Estimation	Goals Scored	Goals Allowed	Goals Scored	Goals Allowed
H1-50000	2.1	45.7	4.1 - 0.3	76.3	79.4	22.5	20.4
H2-05000	4.0	36.3	5.7 - 2.3	72.3	68.9	22.5	18.5
H3-00500	- 1.6	27.7	-0.1 - -3.1	52.2	59.5	13.7	15.2
H4-00050	3.3	30.5	4.8 - -1.7	49.5	37.9	12.2	8.9
H5-00005	-0.3	37	1.4 - -2.0	50.2	46.1	12.3	12.6
HT1-11111	-5.0	49.5	-3.0 - -7.0	55.5	67.9	12.8	17.8
HT2-11201	-3.9	48.1	- 1.9 - -5.9	57.0	63.7	13.7	17.6
HT3-01112	5.0	36.9	6.7 - 3.3	60.6	45.2	16.2	11.2
HT4-00122	7.8	28.6	9.4 - 6.3	60.8	36.0	16.4	8.6
HT5-21101	-6.7	46.5	-4.8 - -8.7	61.9	75.9	14.2	20.9

Table 7.52 Results for Unbalanced Heterogeneous vs All Teams

Against the homogeneous and heterogeneous teams (Table 7.52), results for this team change considerably depending on the opponent. It is only able to win 5 games; it loses 4 of them and ties 1 game. These results show that this team is basically able to win those teams that have either poor attacking (H4, HT3 and HT4) skills or defensive skills (H1, H2). Against teams that present more balanced skills, the team loses by a wide goal margin (HT1, HT2 and HT5). Despite having players better fitted for attacking positions, the number of goals scored show a decrease with respect to the two best heterogeneous teams HT1 and HT5. With respect to the number of goals allowed, it also performs worse than many heterogeneous teams.

This team presents a very similar team formations (Table 7.53) to the one observed in H1, when it plays against **BrianTeam** and **SchemaDemo**. However, the formation against **ou7kou** presents some differences when it is compared to the other teams. This team has the most offensive team formation observed in all the heterogeneous and homogeneous teams. The reason for this configuration is that this team is able to reach easily the opponent goal, because of the players with very good dribbling skills (A1, A2). This aspect can be better observed in the role distribution.

Opponent Team	GL	FB	RD	LD	MD	RM	LM	CF	ST	RW	LW
BrianTeam	1.0	0.12	0.11	0.12	0.94	0.61	0.72	0.10	0.76	0.26	0.26
SchemaDemo	1.0	0.12	0.13	0.15	0.99	0.61	0.69	0.15	0.22	0.48	0.45
ou7kou	1.0	0.07	0.85	0.77	0.86	0.20	0.24	0.06	0.15	0.41	0.39

Table 7.53 Team Formations for Unbalanced Heterogeneous vs Hand-Coded Teams

Team	GL	FB	RD	LD	MD	RM	LM	CF	ST	RW	LW
HT1s	1.0	0.10	0.15	0.14	0.99	0.28	0.40	0.16	0.19	0.84	0.75
H1-50000	1.0	0.06	0.19	0.15	0.99	0.47	0.38	0.08	0.31	0.68	0.69
HT1s	1.0	0.11	0.14	0.12	0.99	0.41	0.26	0.13	0.18	0.84	0.82
H2-05000	1.0	0.06	0.08	0.13	1.00	0.51	0.49	0.09	0.35	0.63	0.65
HT1s	1.0	0.12	0.15	0.17	1.00	0.51	0.51	0.16	0.15	0.63	0.60
H3-00500	1.0	0.06	0.14	0.14	1.01	0.50	0.47	0.10	0.23	0.66	0.70
HT1s	1.0	0.12	0.13	0.16	0.99	0.63	0.61	0.18	0.14	0.51	0.53
H4-00050	1.0	0.08	0.18	0.13	1.03	0.46	0.55	0.12	0.27	0.68	0.50
HT1s	1.0	0.11	0.14	0.16	0.98	0.74	0.76	0.15	0.18	0.40	0.38
H5-00005	1.0	0.11	0.14	0.11	0.98	0.49	0.50	0.13	0.21	0.66	0.67
HT1s	1.0	0.16	0.14	0.19	0.89	0.65	0.63	0.16	0.22	0.48	0.48
HT1-11111	1.0	0.09	0.17	0.16	0.97	0.39	0.36	0.13	0.34	0.70	0.69
HT1s	1.0	0.12	0.17	0.18	1.01	0.52	0.55	0.14	0.16	0.61	0.54
HT2-11201	1.0	0.08	0.11	0.11	0.98	0.41	0.44	0.15	0.25	0.74	0.73
HT1s	1.0	0.11	0.15	0.18	0.97	0.44	0.42	0.17	0.21	0.66	0.69
HT3-01112	1.0	0.14	0.15	0.14	0.97	0.54	0.58	0.10	0.19	0.62	0.57
HT1s	1.0	0.12	0.15	0.14	0.95	0.53	0.51	0.19	0.22	0.60	0.59
HT4-00122	1.0	0.11	0.22	0.22	1.02	0.59	0.56	0.12	0.17	0.48	0.51
HT1s	1.0	0.19	0.17	0.20	0.85	0.57	0.55	0.17	0.16	0.55	0.59
HT5-21101	1.0	0.08	0.15	0.14	0.99	0.34	0.32	0.11	0.38	0.75	0.74

Table 7.54 Team Formations for Unbalanced Heterogeneous vs All Teams

Team formations (Table 7.54) for this HT1s show that they are very similar to those observed in many teams composed of fast robots. Players tend to play in attacking positions when they play against teams made up of fast robots. On the contrary, when they play against teams with big size robots, the team formation changes to a more defensive one. However, results show that this latter case is also observed against the two best heterogeneous teams HT1 and HT5. This result is explained by the problems that this team has to control midfield. Although it has better players to play in these positions, HT1 and HT5 present the best attacking and defensive skills. This makes it difficult for this team to defend and attack; as against these two opponents all positions are important. When playing against players with very good attacking/defensive skills the best team players attempt to neutralise the opponent in those positions, where it plays better.

The role distribution against the hand coded teams (Table 7.55) shows that the most fitted players tend to fill the most important roles of the team, which change depending on the opponent. The results for **BrianTeam** show that these roles are MD and ST. They are filled by the most fitted players A1, A2 and A3, specially the first two robots. The role of goalie, when playing against this opponent, does not represent a very critical position. For this reason the least fitted robots (A4 A5) tend to play in this position.

For **SchemaDemo** the role distribution presents a very important change. The role of goalie becomes one of the key roles, as this team presents a very good attacking strategy. For this reason, the best robots (A1, A2 and A3) fill again the key roles. They also act quite often as MD. In this role there is also an important change in the skills required, as it is a robot with a powerful device, the one that usually plays this role.

Against **ou7kou** the role distribution changes again and the most important roles in the team formation are RD and LD. These players play an important role in this position as they help the team to move the ball forward, and prevent the opponent from getting into the penalty area, as they combine speed and good dribbling skills. In this case, the role of GL is filled by A4 and the role of MD by A5, as before. One of the most important differences with respect to the rest of the heterogeneous and homogeneous teams is the roles filled by A3, apart from the role of GL. This player acts quite often as attacker; this is explained by the position of A1 and A2 as defenders, which allows the team to move very fast the ball to midfield. This gives very good score chances to the attacker, as these players usually get to midfield positions faster than the opponents. This team formation also explains that this team is the second one that is able to score more goals against **ou7kou**.

Opponent Team	A1		A2		A3		A4		A5	
BrianTeam	MD	26%	MD	28%	LM	24%	GL	24%	GL	48%
	ST	24%	ST	23%	ST	21%	MD	19%	LM	10%
	RM	16%	LM	17%	MD	13%	RM	11%		
SchemaDemo	GL	23%	GL	29%	GL	32%	MD	18%	MD	31%
	MD	20%	MD	23%	LM	17%	LM	16%	RM	18%
	RW	13%	RM	12%	RM	14%	LW	16%	LM	10%
ou7kou	RD	31%	RD	28%	GL	23%	GL	47%	MD	38%
	MD	18%	LD	27%	LW	14%	LD	15%	GL	16%
	LD	14%	RW	8%	RW	13%	MD	11%	RW	8%

Table 7.55 Team Formations for Unbalanced Heterogeneous vs Hand-Coded Teams

Role distribution against the learning teams (Table 7.56) shows some of the aspects already observed in the team formations and against the hand-coded teams. Best-fitted agents tend to fill the key roles, while the other players play in other positions. Against H1 the best fitted players (A1 and A2) focus on MD, key role for the control of midfield, and on the wingers, also important to score goals. The other players tend also to fill these roles but not so often as A1 and A2. A4 and A5 finally act most of the times as goalie. Against H2, these roles are LW and RW, while MD is filled by other players, especially by A4. Against the rest of the homogeneous teams the task of these players focuses on midfield roles, while the attacking positions are filled by other players, even by slow robots (A5 against H5). with powerful kicking devices. In these cases the best-fitted robots are responsible for reaching the attacking positions with the ball controlled, while other players attempt to score goals by kicking the ball.

Opponent Team	A1		A2		A3		A4		A5	
H1-50000	RW	31%	MD	23%	LW	28%	GL	41%	GL	28%
	MD	26%	RW	29%	MD	18%	LW	12%	MD	22%
	LW	15%	GL	17%	LM	12%	MD	11%		
H2-05000	LW	37%	RW	36%	GL	31%	MD	32%	GL	25%
	MD	20%	LW	18%	RW	20%	GL	22%	MD	16%
	RM	10%	MD	12%	MD	19%	RW	11%		
H3-00500	MD	29%	MD	23%	GL	37%	LW	20%	MD	23%
	RM	16%	RW	21%	MD	14%	GL	18%	GL	15%
	LM	17%	GL	22%	LW	13%	RW	12%	RW	11%
H4-00050	LM	26%	RM	22%	MD	21%	GL	42%	MD	28%
	MD	24%	RW	17%	LW	18%	LW	11%	GL	21%
	RM	15%	LM	15%	RM	16%	MD	10%		
H5-00005	MD	29%	LM	26%	GL	26%	GL	25%	MD	28%
	RM	22%	GL	23%	RM	17%	MD	18%	RW	11%
	LM	18%	RM	19%	LM	14%	LM	11%	LW	10%

Table 7.56 Team Formations for Unbalanced Heterogeneous vs Homogeneous Teams

Against the heterogeneous teams (Table 7.58) a similar result can be observed. A1, the best-fitted player of this team, usually plays in midfield positions. This player is also helped in these tasks by one of the worst players in the team, A4 and A5, in most cases the role MD is filled by one of these players. The other two players that present good skills tend to play in attacking positions, even though they usually also act as goalie and midfielder. Another interesting aspect that can be observed is that A5 does not act as goalie when playing against the heterogeneous teams. In this case this role is usually filled by a slow robot (A4), but especially by two good robots as A2 and A3, as the opponents present better attacking skills.

Team	W	L	D	Points
HT5	9	0	1	28
HT1	9	1	0	27
HT2	7	2	1	22
HT1s	5	4	1	16
H1	3	3	4	13
H3	3	4	3	12
HT4	3	5	2	11
H2	2	5	3	9
H5	2	6	2	8
HT3	1	6	3	6
H4	1	9	0	3

Table 7.57 Global Results for Homogeneous and Heterogeneous Teams

Team	A1		A2		A3		A4		A5	
HT1sup	RM	24%	GL	28%	LM	17%	GL	28%	MD	26%
	LM	22%	MD	17%	RW	14%	MD	22%	GL	15%
	MD	14%	LW	14%	LW	14%			RW	13%
HT1-11111	MD	47%	LW	21%	LW	23%	GL	55%	GL	45%
	RW	14%	MD	20%	RW	21%	RW	13%	RM	9%
	LW	12%	RW	19%	ST	16%				
HT1sup	RM	25%	GL	21%	MD	28%	GL	36%	MD	26%
	LM	21%	RW	14%	GL	23%	MD	12%	RW	12%
	MD	18%	LW	15%	RW	17%	LW	12%	LW	12%
HT2-11201	MD	38%	MD	29%	GL	30%			GL	40%
	LW	18%	RW	27%	LW	16%				
	RM	13%	LW	15%	RW	15%				
HT1sup	MD	22%	GL	22%	GL	22%	GL	39%	MD	38%
	RM	19%	RW	21%	LW	29%	MD	13%		
	LW	14%	MD	18%	RW	17%				
HT3-01112			MD	44%	RW	24%	GL	65%	LM	18%
			RM	17%	MD	24%			GL	17%
			LW	14%	LW	16%			LW	22%
HT1sup	RM	26%	GL	22%	GL	29%	GL	27%	MD	33%
	LM	20%	RW	21%	LM	14%	LW	22%	GL	12%
	MD	20%	MD	15%	LW	14%	LM	22%		
HT4-00122					MD	22%	GL	33%	GL	16%
					RM	19%	MD	30%	LM	15%
					LW	18%			RW	12%
HT1sup	RM	25%	GL	26%	MD	18%	GL	30%	MD	33%
	LM	21%	MD	17%	GL	17%	MD	16%	RW	10%
	RW	16%	RM	13%	LW	15%	LW	16%		
HT5-21101	MD	32%	LW	25%	GL	53%			GL	47%
	RW	22%	RW	20%						
	LW	17%	MD	20%						

Table 7.58 Team Formations for Unbalanced Heterogeneous vs Heterogeneous Teams

Once these results are included in the team classification, HT1s ranks fourth (Table 7.57). Despite this position in the classification, this team is far from the two best teams.

7.5.3 Results Summary

Unbalanced robot skills show that in a domain as soccer, this feature does not result in a better team performance. While the best fitted robots tend to fill the key roles in the team, performing better in these positions than players with more balanced skills, the performance of the poor skilled robots do not contribute to improve the global team performance and in some cases, this performance is worse. This type of heterogeneous team only performs well when it plays against opponent that present either good attacking or defensive skills. In these cases, the best-fitted robots focuses on the key roles of the teams, neutralising the opponent best skills in offense or defence. However, when it plays against teams that present good performance in attack and defence, this team presents a very poor global performance. This result can be also observed against the hand-coded teams, against which this team is the worst of the heterogeneous teams.

7.6. Performance and Heterogeneity

7.6.1 Introduction

This section analyses team results from the point of view of the diversity measure for physical heterogeneous MAS developed in chapter 6. Team performance is represented according to team diversity and team profile. This aspect is analysed for the two types of experiments, hand-coded teams and learning teams. In both cases, results are normalised with respect to the maximum and minimum team performance.

7.6.2 Hand-Coded Teams

In this set of experiments results are normalised according to the goal margin.

Figure 7.1, 7.2 and 7.3 display how team performance changes depending on team diversity and team profile. The results for the 11 teams have been represented and a surface has been interpolated using these results. According to the figures, team performance tends to increase with diversity up to a given value. After this value teams present a worse performance. The highest values of diversity correspond to the team of unbalanced agent skills. From the point of view of team profile, there are two different types of teams that perform well. On the one hand, defensive teams (0.5) that present a moderate diversity (0.7), on the other hand teams that present a team profile near 0, but with a high diversity. Teams that are rather offensive tend to show a worse performance, even though their diversity is high.

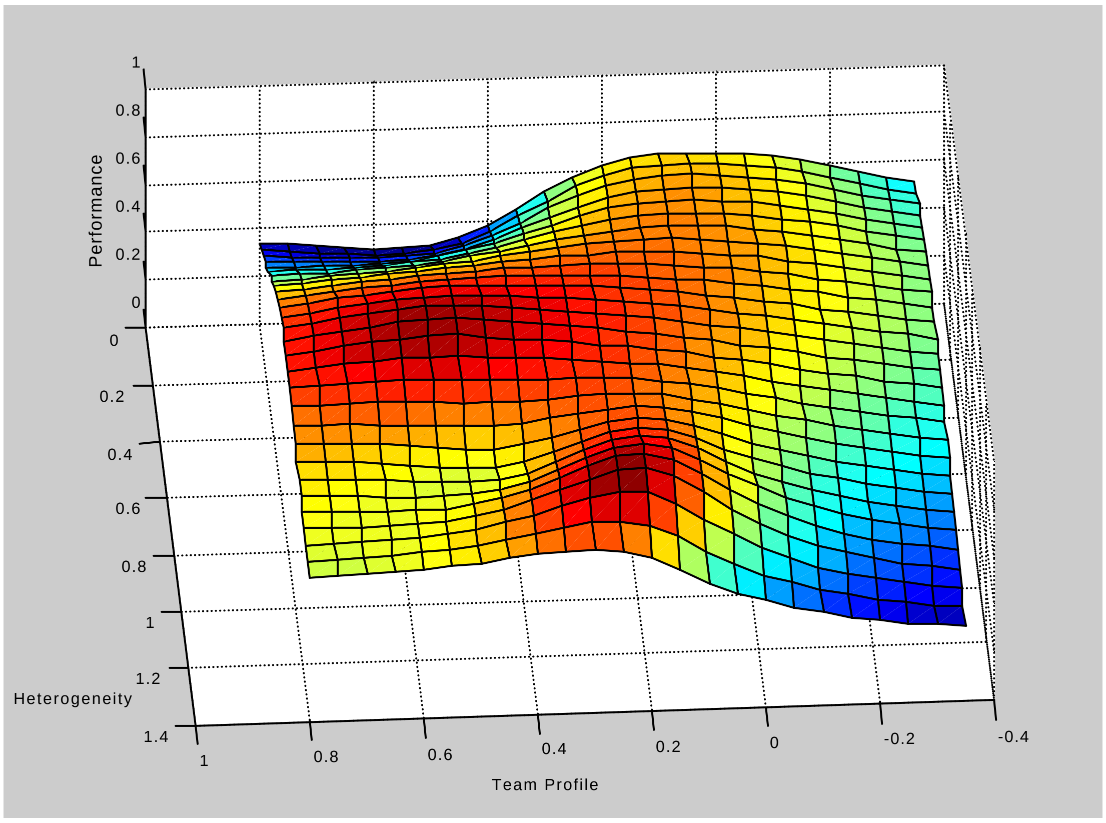


Figure 7.1 Team Performance vs Team Diversity and Team Profile

7.6.3 Learning Teams

In the set of experiments among the learning teams, results have been normalised according to the number of point obtained by each team and setting the performance to 0 for the team with the lowest number of points and 1 for the team with more points.

Results show that performance also increases globally with team diversity. In this case performance also decreases for the case of maximum diversity that corresponds to the team of unbalanced skills. From the point of view of team profile, performance is higher for teams of slightly offensive players, while defensive teams tend to show very poor team performance, even for relatively high values of diversity.

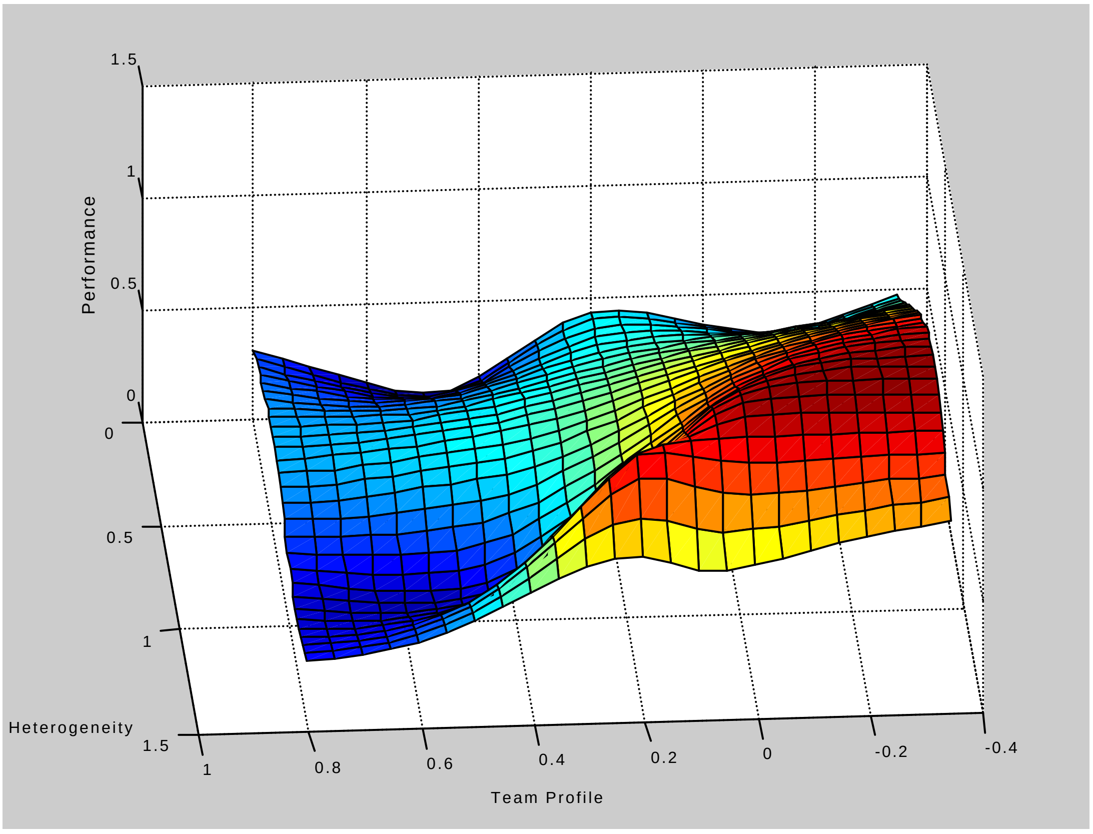


Figure 7.2 Team Performance vs Team Diversity and Team Profile

7.6.4 Heterogeneous MAS Design

One of the most interesting results of diversity and team performance is that the best performing teams tend to present similar values of diversity and team profile. In the case of the hand-coded teams, the best performing teams can be found in two different areas. These results may make possible the design of heterogeneous teams on the basis of the knowledge on some teams. This knowledge can facilitate the task of building a good performing team by discarding some of the possible teams, thus limiting the number of existing candidates. This task becomes more difficult, as the number of heterogeneous components available increase. However, the design of a team of robots out of an available pool of robots is beyond the scope of this thesis and belongs to the future work.

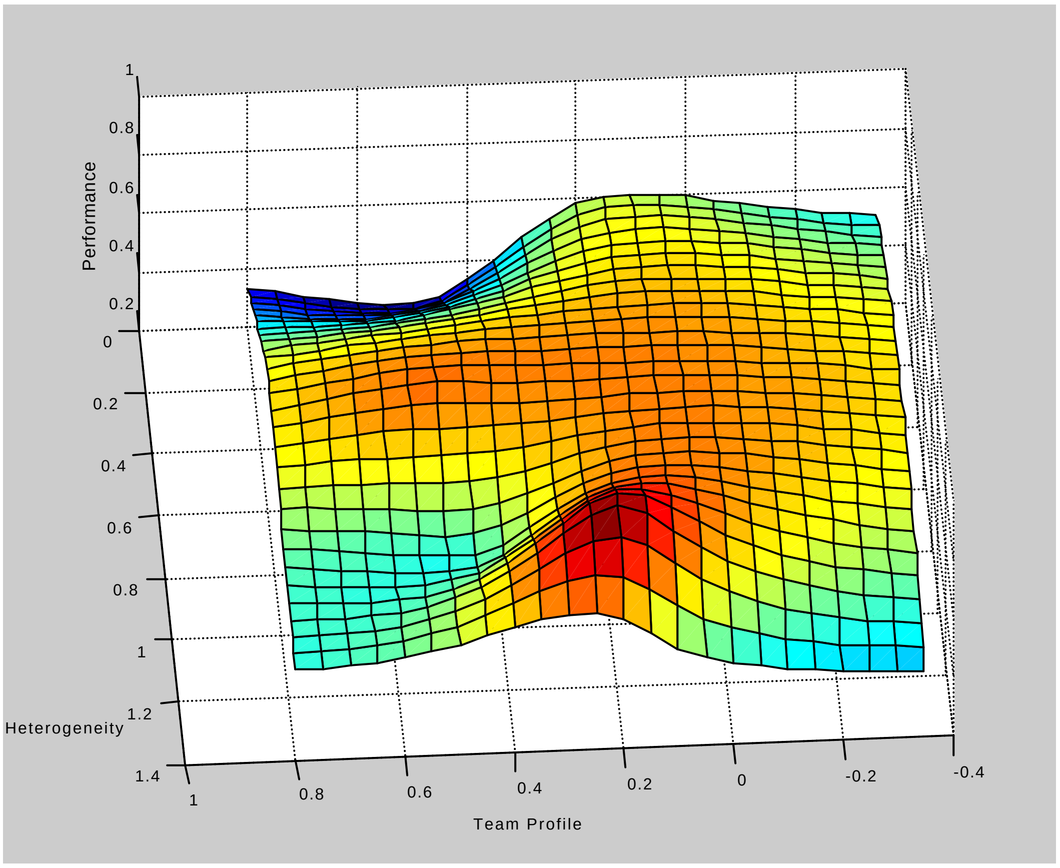


Figure 7.3 Global Team Performance vs Team Diversity and Team Profile

7.7 Summary and Conclusions

This section attempts to draw conclusions from the different experiments performed. It tries also to generalise some of the results that have been found for the domain of soccer. For the study of heterogeneous and homogeneous robot systems, soccer can be considered a domain where co-operation among team members plays a very important role and agents are situated in an adversarial and changing environment.

Generally speaking, heterogeneous robot system performs better than the homogeneous ones in this domain. As it has been seen in the previous section team performance increase with heterogeneity up to a given value. This result is explained by two different reasons:

- Different skills. A heterogeneous team present different skills. In the domain where the analysis have been conducted, results show that the different roles, require different player skills. The only possible way of having the largest possible set of skills is by having a team of heterogeneous components.

- Flexibility. Requirements to fill each role not only depend on the role, but also on the environment. Several examples ha shown that role distribution in a team of heterogeneous teams depended on the opponent. In some case a player filling the role of attacker against one opponent, was acting as midfield defender (a player between midfield and defence) against a different opponent. Flexibility can be only be provided by heterogeneous teams.

Despite the advantages of heterogeneous teams, these teams may not always offer the best performance, even though only a subset of the heterogeneous teams has been evaluated. This is explained by the fact that in some cases one robot is the best to perform the most important tasks, while for the rest of the tasks is not so important which agent takes care of them. However, the performance for heterogeneous teams does not present very important variations for changes in the environment. Some of the examples show that some homogeneous teams perform among the best teams against a given opponent, but when the same team plays against a different opponent it may perform among the worst teams. In the examples for the group of heterogeneous teams analysed show that these teams always perform much better than the worst of the homogenous teams. The difference between the best and worst heterogeneous team is considerable smaller than the one observed in homogenous teams. These features in heterogeneous teams are especially important for domains where the environmental conditions can change dramatically. In domain where the environment does not present important variations, a team of robots can be designed, so that it performs quite efficiently for all the environments.

Another interesting feature than can be concluded from these results is that in domains where tightly co-operation among agents is a key factor, the design of heterogeneous teams is a complex task. Some examples in this chapter show that is not possible to predict the result of a heterogeneous team, using the information available from the results of the homogenous ones. There are some examples where this aspect can be found (HT1/HT2, HT3/HT4). The introduction into one of the teams of a robot that presents a very low performance as a team (H4), improves the team performance when compared to another team consisting of a the same number of players, with the exception of this player. The co-operation among two different types of robots may give rise to unexpected features, and in this case this results in a better team performance.

Related to this aspect, results also show that in some cases some players that are not well fitted for a given role, co-operation and the presence of other players can lead this player to play in this position, not affecting the team performance. Teammates can improve a robot performance in a given position through co-operation and task distribution.

The combination of a diversity metric and other possible measures of team features may contribute to the design of heterogeneous teams, on the basis of the information available on some of the teams already built. Graphical results for performance and diversity can help to build such systems, as it is possible to know which degree of diversity and team features present better team performance. Using this information, it is possible to test several candidate teams and discard other teams. Results for the teams analysed in this chapter show that the best teams tend to show similar diversity degrees and team features.

Results also show that domains where co-operation is important and the participation of available robots in the tasks of the team are very important feature, a bad and unbalanced distribution of skills across a team of robots may result in a very poor team performance. In the example analysed in this chapter, while some agents can perform some tasks better than other robots, the fact that other team members are worse skilled results in a globally lower team performance.

Chapter 8

Conclusions

8.1 Introduction

This thesis has contributed to the field of physically heterogeneous MAS in several ways. The first contribution has been the development of a model that allows agents to learn to select which are the best tasks they can perform, in soccer the roles they can fill, on the basis of their individual skills and team global performance. The second one has been the analysis of heterogeneous MAS and the comparison with homogeneous teams in co-operative and adversarial domains, as soccer. In addition, a methodology has been defined to measure physical diversity, using and adapting an entropy metric for behavioural heterogeneous robotic teams. Results show that this methodology can be a useful tool to build teams of heterogeneous robots.

8.2 Algorithm

The algorithm developed in this thesis emulates two phenomena observed in natural systems, task partition and the formation of hierarchical dominances and it is based on a previous work by [Hogg and Huberman 1991]. This algorithm has proved to work quite efficiently in the learning task and has been later used to analyse and evaluate physical heterogeneity in robotic teams.

This algorithm allows agents to learn, in the form of rewards, which are the best tasks to perform depending on their skills and their contribution to the global team performance. Usually a competition among the different agents that compose the system arises, as several members may find equally interesting some of the tasks. This gives rise to a self-organising process, where the system evolves from a fully disorganised team to a fully organised one. This process results in the assignment of a set of roles to a set of heterogeneous robots. It emulates successfully the phenomena observed in nature, where complex behaviours arise out of the interactions among the members of the system.

This algorithm has also proved to be suitable to endow MAS with a high adaptability and robustness to changes in the environment and in the team members. Results have shown that the team is able to adapt when one of the team members is damaged or the environment changes. The presence of a parameter called fitness, which measures how well is each team member performing with respect to the team average has proved to be very important in this process, as adaptability is possible in many cases thanks to this parameter. This parameter can also contribute in some cases to the improvement in team performance, as it allows the best-fitted team members to participate more actively in the tasks of the team, and thus fully exploiting the skills of these team members.

In addition, this algorithm also contributes to other issues and questions raised in MAS: On one hand [Parunak et at 2000] claims that the full potential of MAS is not exploited, which lies in the interactions of the members of the system. This algorithm has proved that a system composed of relatively simple agents can exploit their interactions, giving rise to a complex process of self-organisation and adaptation for which they were not programmed. On the other hand, [Wolper et at 2000] propose for the design of MAS, the use of distributed and concurrent learning algorithms, in order to avoid human intervention in the task of adjusting and tuning MAS: This thesis has also contributed to this issue, as this algorithm allows MAS to organise and adapt, once agents have been built and their interaction defined, without the need of human intervention, as the experiments have shown.

8.3 Heterogeneity Analysis

The impact of heterogeneity in physical heterogeneous MAS has been analysed from two different points of view, performance, task partition and adaptability. For this purpose soccer has been selected as test-bed. Soccer offers the possibility to conduct research in MAS in an adversarial and changing environment, but where co-operation among team members also plays a very important role. In this thesis a group of homogeneous and heterogeneous teams have been analysed, robots presented features, that ranged from slow, big and with a powerful kicking device robots to fast, small robots. These features have allowed to build robots with different skills to play in soccer.

A first set of experiments has been conducted to test these two types of teams (5 homogeneous and 4 heterogeneous) against three hand-coded teams. Results in these experiments have shown that heterogeneous teams perform globally better than the homogenous ones, although in some cases one of the homogeneous teams has performed better against some of the teams selected. Results for the heterogeneous team also show that heterogeneous team tend to perform more uniformly than homogeneous ones, presenting smaller differences in the performance across the different teams.

A second set of experiments has analysed all the teams together by organising a league among all the teams. Among the nine teams that have initially been defined (4 heterogeneous and 5 homogenous) 3 heterogeneous teams have qualified among the four best teams. Results in this case have also shown other interesting aspects. Role distribution among the heterogeneous team change depending not only on the skills of each robot, but also on the opponent. This represents that heterogeneous teams not only globally perform better than homogenous ones because of presenting different skills for different tasks, but also due to possibility of combining these skills in different ways, depending on the environment. Another interesting feature observed is that the combination of players that play individually much better, do not always results in a better team performance, as some examples showed in this section. This is explained by the combination of different skills that for some tasks may result more efficient than the use of two players, which individually are better, but with similar skills.

In a third set of experiments a special case of a heterogeneous team has been analysed, where robot skills are unbalanced. This team consisted of two very good robots, an average robot, and two poorly skilled robots. Results have shown that this team was not able to perform better than the other heterogeneous teams, and globally performs very similarly to an average homogeneous team. This result is explained by the need of having skills fairly distributed, in cases where co-operation in adversarial domains plays a very important role.

In order to evaluate the results according to the heterogeneity of each team, a methodology have been developed to measure entropy in such systems building on a previous work by [Balch 2000], where a measure of behavioural diversity is established. This has been achieved by defining a set of different benchmarks, using as a basis a benchmark called the implicit opponent, which allows to measure each team abilities and represent them in a simple way to test other teams or individual robots.

Using this metric, the correlation between heterogeneity and performance has been analysed. In general, performance improves as heterogeneity increases, although there are also homogeneous teams, which presented excellent performances. The combination of the diversity metric with a measure of some of the team features can help designers to build heterogeneous teams of robots. Results have shown, in the domain selected for this research, that it is possible to find which is the best possible team on the basis of the information on the performance of several teams and their diversity, as the best performing teams usually present similar features.

From the point of team adaptability heterogeneous teams are more adaptable than homogenous teams. Experiments show that diversity allows teams to take advantage of the robot's individuals skills. In case of an environment change robots may recombine their skills to adapt better to the environment. On one hand, this feature is not possible in homogeneous teams, as all robots present the same skills. On the other hand this process may take longer for heterogeneous teams, as the team may test several solutions to adapt to the environment. One of the problems of this approach is that the resulting team formation may not present the best team performance, as the possible number of combinations increases considerably with the number of heterogeneous components.

Generally speaking, heterogeneity presents more flexibility and adaptability than homogeneous teams, and perform globally better, although homogeneous teams may perform better in some particular cases, as the skills required may be similar for the different tasks.

Chapter 9

Future Work

This thesis has contributed to several aspects in the field of physically heterogeneous MAS. However, this field of research presents more problems that those analysed in this thesis. The model developed to study and work with these systems has proved quite efficient to deal with this feature in MAS. For this reason, we think that this model can be used to study further aspects related to this field. The following list gives only a few ideas:

- Heterogeneity has only been studied in the soccer domain, which presents some features that allow to extend these results to other domains. However, there are other types of domains that may also benefit from this thesis. One of the domains, which is going to benefit from this thesis, is the rescue of people in natural disasters, especially in earthquakes, by a heterogeneous teams of soccer. This domain is also known as Rescue and has been proposed as a new domain to apply the research developed for the field of soccer.

- Physical heterogeneity has been studied in this thesis mainly in the context of actuators and robot size. Although this is one of types of heterogeneity most common in robot systems, there are other types of heterogeneity that may condition teamwork. Among all the different types of heterogeneity, one of the most interesting ones from the point of view of how it can affect team organisation are the types of sensors and sensor range.

- This thesis has only focused on one type of decision-making. However, this is not the only type of decision-making that can be found in teams of robots. One of the most interesting aspect to study in the future is how this model and other decision-making processes should be integrated, so that the system could benefit from the features observed in the examples studied in this thesis.

- This thesis have developped a methodology to measure physical diversity. Results have shown that the combination of degree of diversity and the features of team can help designers to build heterogeneous systems, on the basis of some known examples. This aspect should be studied more in depth, as it can a possible solution to the problem of building heterogeneous teams from a pool of available robots.

- This model also can be applied in the context of the problems that have motivated originally this thesis, information ecosystems. Although this domain presents different features from those, where physical heterogeneous MAS are common, it shares some common interests, especially those related to the heterogeneity in the system and how adaptability and self-organisation is achieved.

From the point of view of the model, there are several aspects that can be improved.

- The model developed in this thesis is only useful for domains where co-operation is important, but it does not play a central role. This model can be easily adapted and extended to include in the process of task assignment tightly coupled co-operation. This type of co-operation proves to be very important in heterogeneous robot systems, as the literature review shows. Preferences not only might concern tasks, but also which teammates are the most desirable ones to perform a joint-task. However, this change cannot be effective when the number of components in the system increases considerably.

- One of the problems of this model is that agents require communication capabilities to exchange data, so that they can update their fitness values using the information on other teammates. In domains where the number of component is high, communication is limited or the distance among the team members can be greater than the communication range, this task may prove difficult or may result in a high number of communication collisions. In this case they should be able to work with incomplete information, from the point of view of fitness, and new methods to exchange and deal with this uncertainty should be found.

ANNEXE

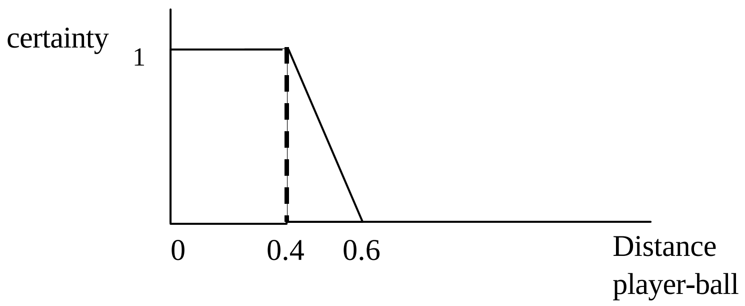
I. INTRODUCTION

This annex describes in detail how the different roles are implemented. It also details the different fuzzy sets that have been created to define each role and the actions that each robot can select when filling a given role. Roles design and team implementation is completely built on a previous work done by undergraduate students. Only a few changes have been introduced to adapt it to this work. These changes have resulted in an increase in the number of roles and actions.

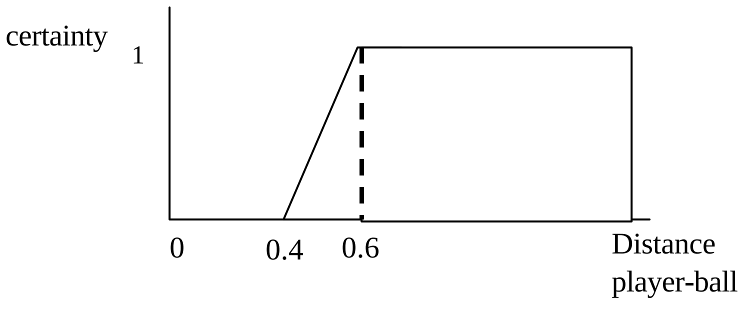
II. RULES

Rules contain information on which action to take, according to the position of the players and the ball on the field. Theses rules have been implemented using fuzzy sets, which transform positions and distances into a value, ranging from 0 to 1. In each rule, several variables, represented by means of these sets, and logic operators are combined together to determine which action to take. Next, all these variables are described.

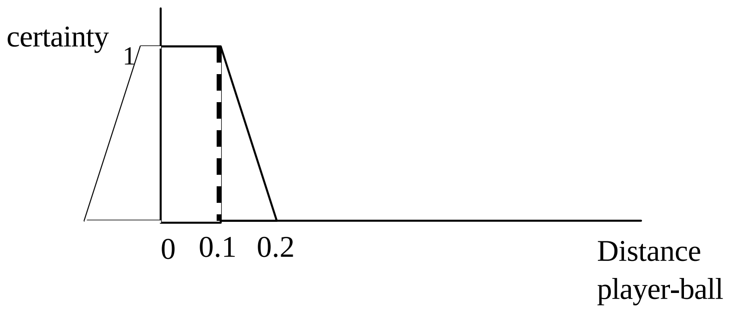
- *be_near_ball*



- *be_far_ball*



- *have_ball*



- *clear_kick:* This variable defines the chance of scoring a goal given the position of the ball and the robot. If the player decides to kick the ball and the ball is inside the white are, it means that the player has a very good chance of scoring a goal (certainty 1). If the ball is inside the black area, it means that is going to be difficult to score a goal, although it is not impossible (certainty [0,1]). Outside these two zones, it is unlikely that the action results in a goal (certainty 0).

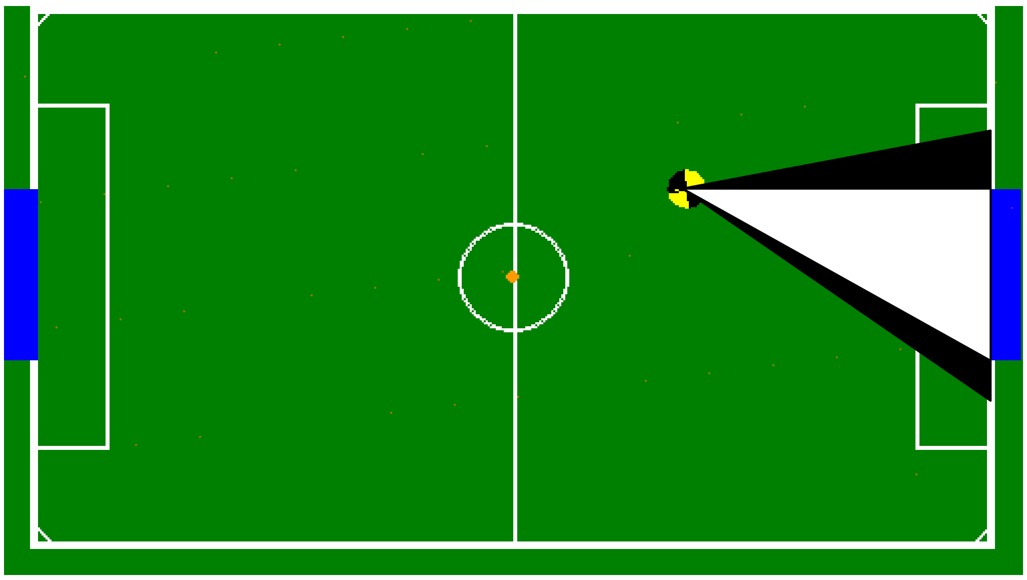
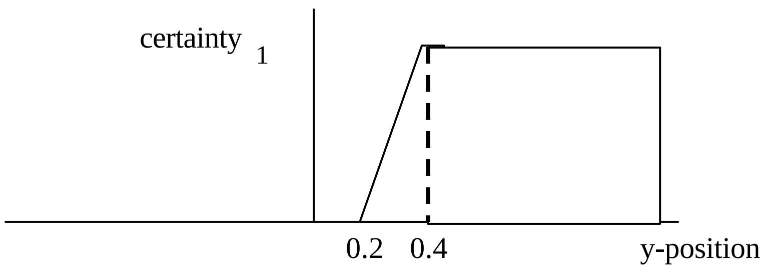
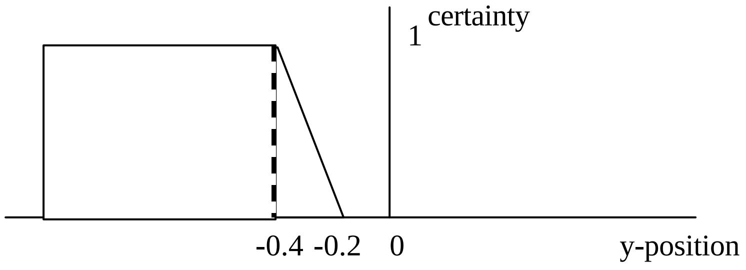


Figure I clear_kick

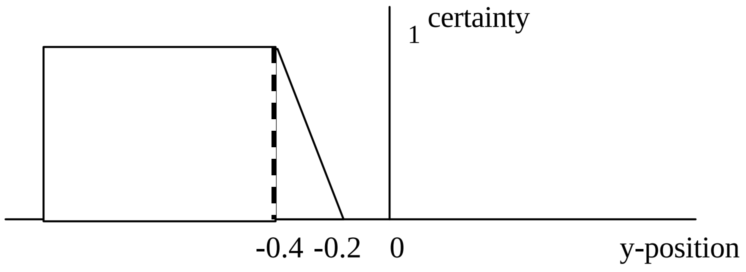
- *be_right_side:* if the team defends the east goal, then be at the right side of the field is:



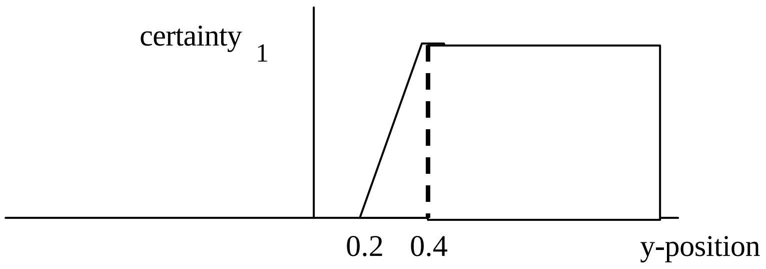
If the team defends the west goal, then:



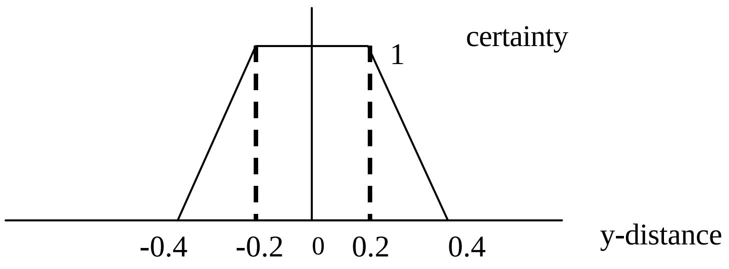
- *be_left_side*: if the team defends the east goal, then be at the left side of the field is:



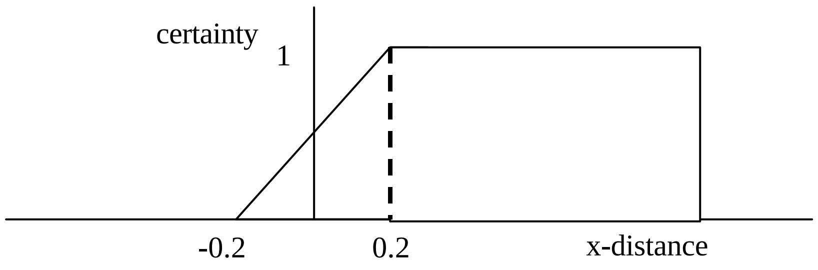
If the team defends the west goal, then:



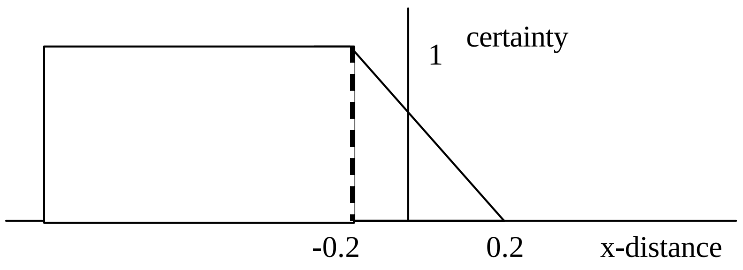
- *be_central_side:* this variable is complementary to *be_right_side* and *be_left_side* and represents the zone between the two sides.



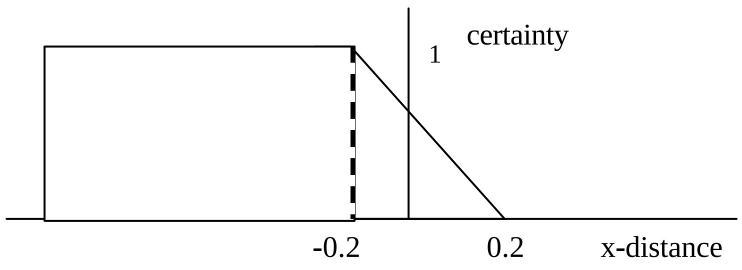
- *be_own_side:* if the team defends the east goal



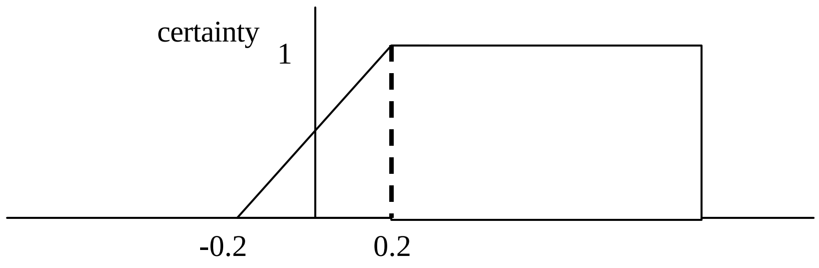
If the team defends the west goal



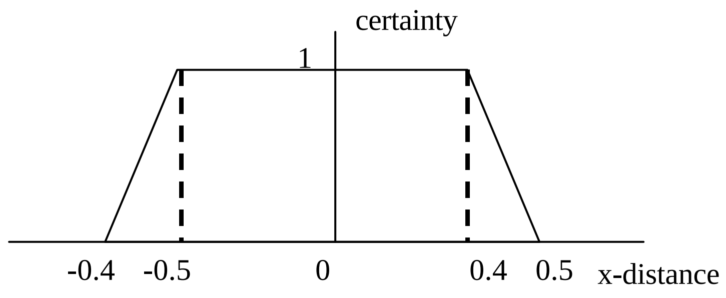
- *be_opponent_side:* if the team defends the east goal



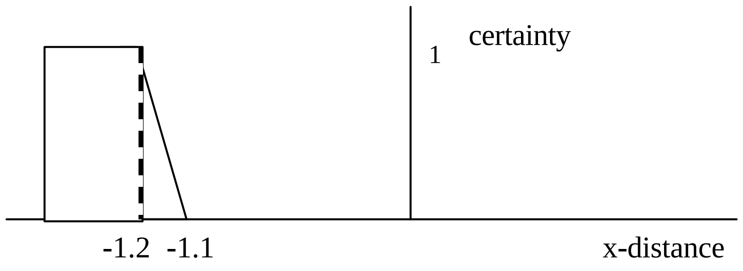
If the team defends the west goal, then:



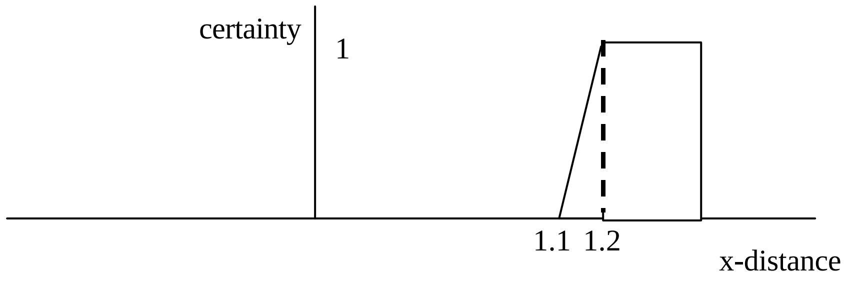
- *be_midfield:* This variable is complementary to *be_own_side* and *be_opponent_side*. It represents the area between these two field zones.



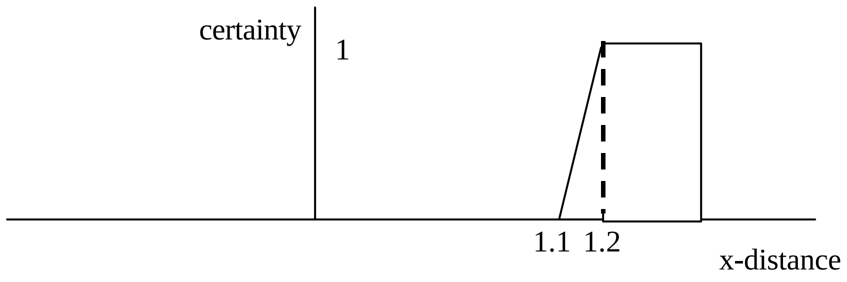
- *be_own_penalty_area:* if the team defends the east goal



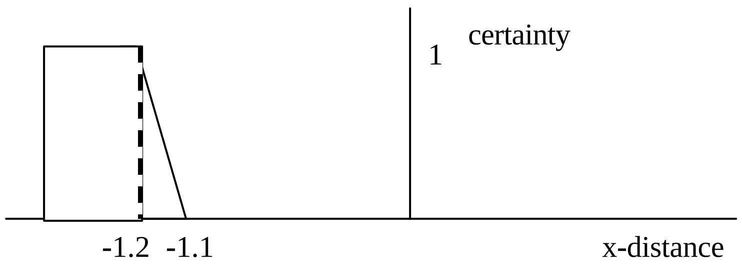
if the team defends the west goal



- *be_opponent_penalty_area:* if the team defends the east goal



if the team defends the west goal



- *be_in_front*: Given the position of two players, or one player and the ball, and the goal that the team is defending, *be_in_front* means that the player/ball is in front of the ball/player. This variable can only take two values, 0 or 1.

III. ROLES

Eleven roles have been coded using the fuzzy variable defined in the previous section and the actions described in the next one. With the some exceptions, each role consists of four different rules. In most cases, one rule is used to kick the ball, a second one to get to ball, a third one to regain control of the ball and finally the last one, positions the robot within its area, while keeping it close to the ball.

- ***Goalkeeper (GL)***

R1: *have_ball* and *clear_kick* $\Rightarrow$ ***Kick_ball***

R2: ***Defend***

- ***Fullback (FB)***

R1: *have_ball* and *ball_be_in_front* and *ball_be_own_filed* and *ball_be_central_side* $\Rightarrow$ ***Kick_ball***

R2: (*ball_be_in_front* and *ball_be_own_side* and *ball_be_central_side*) $\Rightarrow$ ***Go_to_ball***

R3: *be_in_front_ball* and *ball_be_own_side* and *ball_be_central_side* and not (*ball_be_own_penalty_area*) $\Rightarrow$ ***Regain_ball***

R4: or(*ball_be_far*, or(*ball_be_own_penalty_area*, or(not(*ball_be_right_side*) ,not(*ball_be_own_side*))))$\Rightarrow$ ***Keep_position***

- ***Right defender (RD)***

R1: *have_ball* and *ball_be_in_front* and *ball_be_own_side* and *ball_be_right_side* $\Rightarrow$ ***Kick_ball***

R2: (*ball_be_in_front* and *ball_be_own_side* and *ball_be_right_side*) $\Rightarrow$ ***Go_to_ball***

R3: *be_in_front_ball* and *ball_be_own_side* and *ball_be_right_side* and not(*ball_be_own_penalty_area*) ⇒ ***Regain_ball***

R4: or(*ball_be_far*, or(*ball_be_own_penalty_area*, or(not(*ball_be_right_side*) ,not(*ball_be_own_side*))))⇒ ***Keep_position***

- ***Left defender (LD)***

R1: *have_ball* and *ball_in_front* and *ball_be_own_filed* and *ball_be_left_side* ⇒ ***Kick_ball***

R2: (*ball_be_in_front* and *ball_be_own_side* and *ball_be_left_side*) ⇒ ***Go_to_ball***

R3: *be_in_front_ball* and *ball_be_own_side* and *ball_be_left_side* and not(*ball_be_own_penalty_area*) ⇒ ***Regain_ball***

R4: or(*ball_be_far*, or(*ball_be_own_penalty_area*, or(not(*ball_be_left_side*) ,not(*ball_be_own_side*))))⇒ ***Keep_position***

- ***Defensive mifielder (DM)***

R1: *have_ball* and *ball_be_in_front* and or(*ball_be_own_side*, *ball_be_midfield*) and *ball_be_central_side* ⇒ ***Kick_ball***

R2: *be_in_front_ball* and *ball_be_central_side* and or(*ball_be_own_side,ball_be_midfield*) ⇒ ***Go_to_ball***

R3: *ball_be_in_front* and or(*ball_be_own_side,ball_be_midfield*) and *ball_be_central_side* and not(*ball_be_own_penalty_area*) ⇒ ***Regain_ball***

R4: or(*ball_be_far*, or(*ball_be_own_penalty_area*, or(not(*ball_be_central_side*) , *ball_be_opponent_side*)))⇒ ***Keep_position***

- ***Right mifielder (RM)***

R1: *have_ball* and *clear_kick* and *ball_be_right_side* and or (*ball_be_own_side, ball_be_midfield*) ⇒ ***Kick_ball***

R2: *ball_be_in_front* and *ball_be_midfield* and *ball_be_right_side* ⇒ ***Attack***

R3: *ball_be_in_front* and *ball_be_own_side* and *ball_be_right_side* ⇒ ***Go_to_ball***

R4: *be_in_front_ball* and *ball_be_midfield* and *ball_be_right_midfielder* ⇒ ***Regain_ball***

R5: or(*be_far_ball*, or(not(*ball_be_right_side*) , not(*ball_be_own_side*)))⇒ ***Keep_position***

- ***Left mifielder (LM)***

R1: *have_ball* and *clear_kick* and *ball_be_left_side* and or (*ball_be_own_side*, *ball_be_midfield*) $\Rightarrow$ ***Kick_ball***

R2: *ball_be_in_front* and *ball_be_midfield* and *ball_be_left_side* $\Rightarrow$ ***Attack***

R3: *ball_be_in_front* and *ball_be_own_side* and *ball_be_left_side* $\Rightarrow$ ***Go_to_ball***

R4: *be_in_front_ball* and *ball_be_midfield* and *ball_be_left_midfielder* $\Rightarrow$ ***Regain_ball***

R5: or(*be_far_ball*, or(not(*ball_be_left_side*) , not(*ball_be_own_side*)))$\Rightarrow$ ***Keep_position***

- ***Center forward (CF)***

R1: *have_ball* and *clear_kick* and *ball_be_center_side* and or (*ball_be_opponent_side*, *ball_be_midfield*) $\Rightarrow$ ***Kick_ball***

R2: *ball_be_in_front* and *ball_be_center_side* and or(*ball_be_midfield*, *ball_be_opponent_side*) $\Rightarrow$ ***Attack***

R3: *be_in_front_ball* and *ball_be_center_side* and or(*ball_be_midfield*, *ball_be_opponent_side*)$\Rightarrow$ ***Regain_ball***

R4: or(*be_far_ball*, or(not(*ball_be_center_side*) , not(*ball_be_midfield*)))$\Rightarrow$ ***Keep_position***

- ***Right winger (RW)***

R1: *have_ball* and *clear_kick* and *ball_be_right_side* and *ball_be_opponent_side* $\Rightarrow$ ***Kick_ball***

R2: *ball_be_in_front* and *ball_be_right_side* and *ball_be_opponent_side* $\Rightarrow$ ***Attack***

R3: *be_in_front_ball* and *ball_be_right_side* and *ball_be_opponent_side*$\Rightarrow$ ***Regain_ball***

R4: or(*be_far_ball*, or(not(*ball_be_right_side*) , *ball_be_own_field*))$\Rightarrow$ ***Keep_position***

- ***Left winger (LW)***

R1: *have_ball* and *clear_kick* and *ball_be_left_side* and *ball_be_opponent_side* $\Rightarrow$ ***Kick_ball***

R2: *ball_be_in_front* and *ball_be_left_side* and *ball_be_opponent_side* $\Rightarrow$ ***Attack***

R3: *be_in_front_ball* and *ball_be_left_side* and *ball_be_opponent_side*⇒ ***Regain_ball***

R4: or(*be_far_ball*, or(not(*ball_be_left_side*) , *ball_be_own_field*))⇒ ***Keep_position***

- ***Striker (ST)***

R1: *have_ball* and *clear_kick* and *ball_be_center_side* and *ball_be_opponent_side* ⇒ ***Kick_ball***

R2: *ball_be_in_front* and *ball_be_center_side* and *ball_be_opponent_side* ⇒ ***Attack***

R3: *be_in_front_ball* and *ball_be_center_side* and *ball_be_opponent_side*⇒ ***Regain_ball***

R4: or(*be_far_ball*, or(not(*ball_be_center_side*) , *ball_be_own_field*))⇒ ***Keep_position***

IV ACTIONS

Although eleven roles have been defined, only six actions have been described. Most of the actions are shared by several roles. However, the rules that activate these actions depend on the role.

Kick_ball
The player goes at full speed to the ball and kicks it.

Go_to_ball
The player goes to the ball and stops behind it. Next he can *Attack* or *Kick_ball*. The player positions 5 cm away from the ball and pointing at the same direction as the vector coming from the own goal (white line in Figure II). In order to get to the ball as fast as possible, the robot move at full speed. This action allows the robot approach the ball while defending its own goal and it is included in the defensive roles.

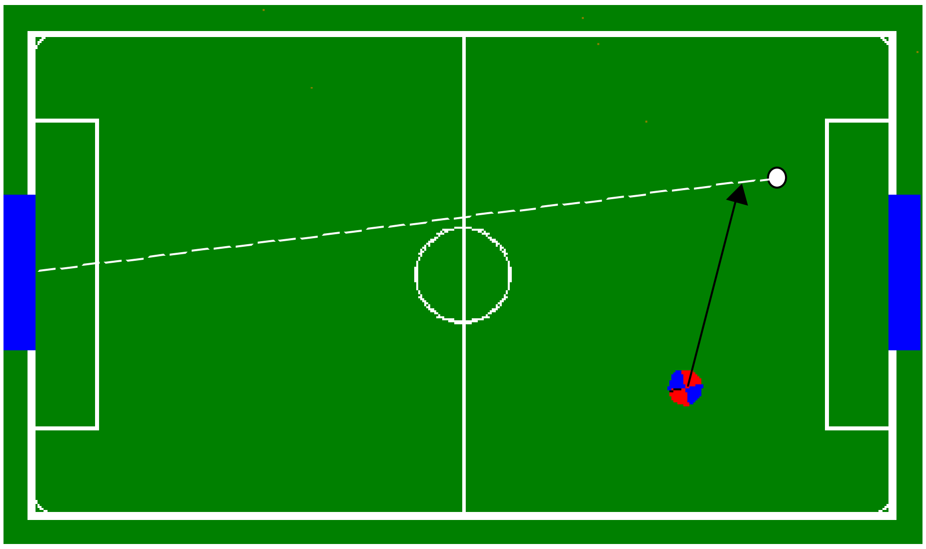


Figure II Go_to_ball

Regain_ball

Regain_position is a defensive action, very similar to *Go_to_ball*. Unlike *Go_to_ball* the distance is increased up to 20 cm, in order to get behind the ball and kick the ball afterwards. In order to get to the ball as fast as possible, the robot moves at full speed.

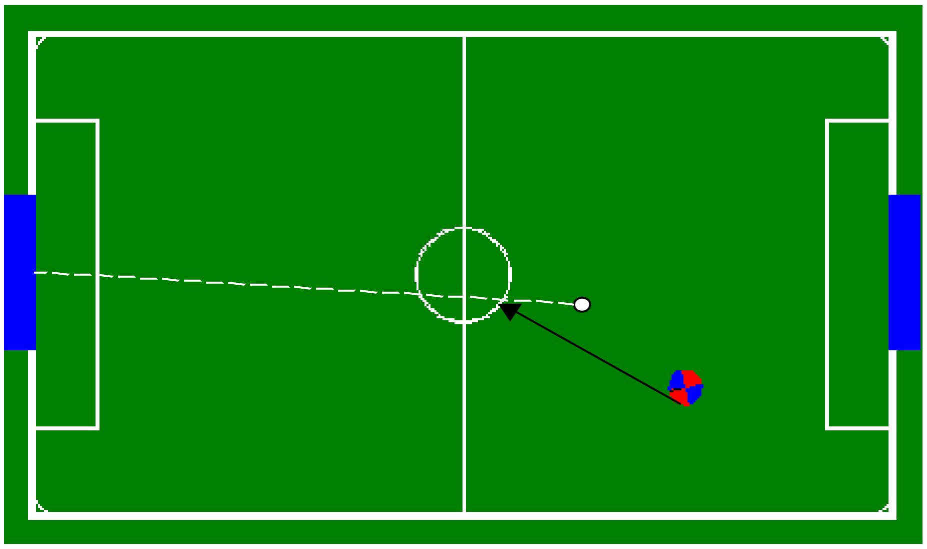


Figure III Regain_ball

Defend

The goalkeeper is the only player that performs this task. The player moves to a position between the ball and the goal, at a maximum distance of 35 cm from the goal. The grey area shows the zone where the player moves when performing this action. The player moves at full speed until it reaches the final position.

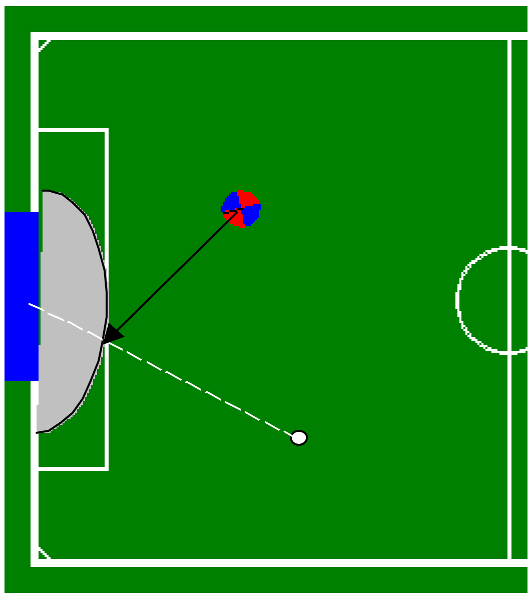


Figure IV Defend

Attack

The robot positions at 5 cm behind the ball, while pointing at the middle of the opponent goal. This allows the robot to start a new attacking play by kicking or pushing the ball. This action is performed at full speed by the robot.

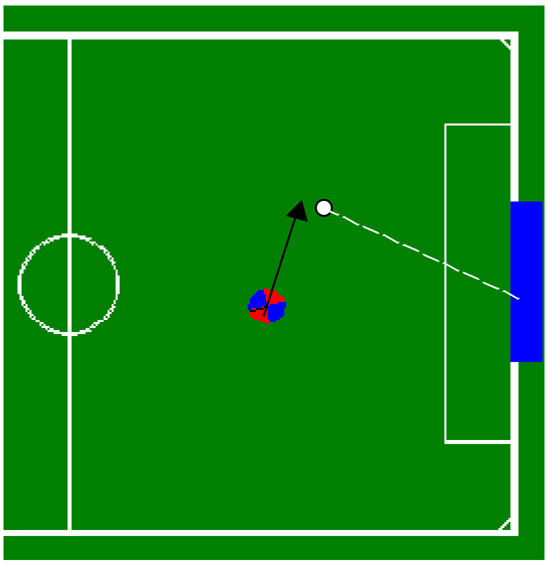


Figure V Attack

Keep_zone

This action positions the robot within its assigned area, so that it is close to the ball. The robot is usually positioned at the closest point from the ball within the assigned area.

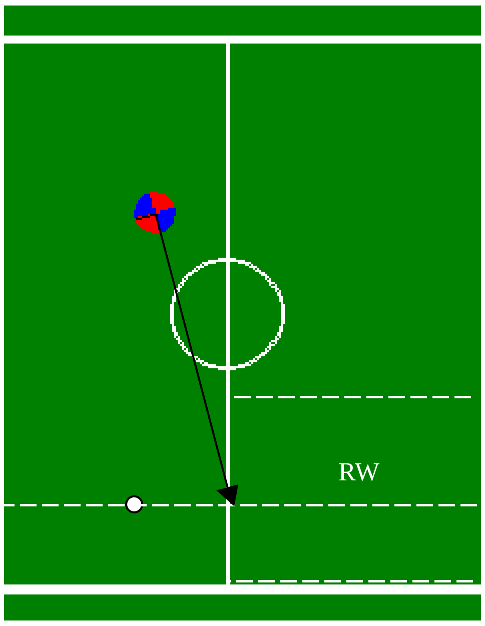


Figure VI Keep_zone

V OBSTACLE AVOIDANCE

One of the most important skills, in order to have a good robotic team, is obstacle avoidance. Obstacle avoidance should be used to avoid opponents and teammates. For this work, a very simple obstacle avoidance skill has been developed. This obstacle avoidance skill consists of two rules:

Opponents

If ((*distance_opponent*<*min_distance_opponent*)&(*end_point_left*)) **then** *turn_left*
If ((*distance_opponent*<*min_distance_opponent*)&(*end_point_right*)) **then** *turn_right*

Teammates

If ((*distance_teammate*<*min_distance_teammate*)&(*end_point_left*)) **then** *turn_left*
If ((*distance_teammate*<*min_distance_teammate*)&(*end_point_right*)) **then** *turn_right*

distance_opponent / *distance_teammate* are the closest opponent/teammate that stand on the way to a given point. These rules are activated when a minimum distance is surpassed (*min_distance_opponent/ min_distance_teammate*). The distances for the opponents and the teammates are different. The distance between teammates is shorter, because collisions among teammates can be tolerated. *end_point_left*/*end_point_right* mean that from the position of the player is easier to get to the end point, by avoiding the obstacle turning on the left/right.

BIBLIOGRAPHY

[Adorni et al 2000] Adorni G., Bonarini A., Clemente G., Nardi D., Pagello E. and Piaggio M., ART'00- Azzura Robot Team for the Year 2000. In proceedings of RoboCup 2000: Robot Soccer World Cup IV, Springer Verlag pp559-562.

[Akiyama 2000] Akiyama H., A Team, In proceedings of RoboCup 2000: Robot Soccer World Cup IV, Springer Verlag pp401-404.

[Ali 1999] Ali K. S., MultiAgent Telerobotics, Matching Systems to Tasks, PhD dissertation, Georgia Institute of Technology, June 1999.

[Andre and Teller 1998] Andre D. and Teller A., Evolving Team Darwing United, In Proceedings of the second RoboCup Workshop, 1998.

[Arthur 1994] Brian A.W., Inductive Reasoning and Bounded Rationality (The El Farol Problem), American Economics Review (Papers and Proceedings), 84, 406, 1994.

[Asada et al 1997] Asada M., Kuniyoshi Y., Drogoul A, Asama H., Mataric M., Duhaut D., Stone P. and Kitano H. The RoboCup Physical Agent Challenge, First RoboCup Workshop in the XV IJCAI-97 International Joint Conference on Artificial Intelligence, pp.51-56, 1997.

[Balch 1997] Balch T, JavaSoccer, RoboCup-97: Robot Soccer World Cup I in Lecture Notes in Computer Science, pp 181-187, 1997.

[Balch 1998] Balch T., Behavioral diversity in Learning Robot Teams, PhD Thesis Georgia Tech 1998.

[Balch 2000] Balch T., Hierarchic social entropy: an information theoretic measure of robot team diversity Autonomous Robots, July 2000.

[Barman et al 1993] Barman R. A., Kingdon S. J, Mackworth A., Pai D., Sahota M., Wilkinson H., and Zhang Y. Dynamite: A Testbed for Multiple Mobile Robots, Proceedings of the IJCAI-93 Workshop on Dynamically Interacting Robots, 1993.

[Beaugrand and Cotnoir 1996] Beaugrand J. P. and Cotnoir P.A., The role of individual diffrences in the formation of triadic dominance orders of male green swordtail fish, Behavioural Processes, 38 (196) 287-296.

[Bonabeau et al 1997a] Bonabeau E., Theraulaz G., Deneubourg J-L., Aron S. and Camazine S. Self-Organisation in Social Insects, Trends Ecological Evolution. 12, 183-193, 1997.

[Bonabeau et al 1997b] Bonabeu E., Sobkowski A., Theraulaz G. and Deneubourg J-L. Adaptative Task Allocation Inspired by a model of Division of Labour in Social Insects, , Bio Computation and Emergent Computing, pp 36-45, 1997.

[Bonabeau et al 1998] Bonabeau E., Henaux F., Guerin S. Snyers D., Kuntz P. and Theraulaz G., Routing in telecommunications networks with smart ant-like agents Intelligent, Agents for Telecommunications Applications '98 (IATA'98).

[Bonabeau et al 1999] Bonabeu E., Theraulaz G. and Deneubourg, Dominance orders in animal societies: the self-organisation revisited, Bulletin of Mathematical Biology,1999.

[Bonabeau, Dorigo and Theraulaz 1999] Bonabeau E., Dorigo M and Theraulaz J-L, Swarm Intelligence: From Natural to Artificial Systems, Oxford Univeristy Press.

[Bongard 1999] Bongard J.C., Divide and Conquer: A Novel Approach to the Design of Heterogeneous Multi-Robot Systems, MSc, University of Sussex 1999.

[Bongard 2000] Bongard J.C., The Legion System: A Novel Approach to Evolving Heteogenity for Collective Problem SolvingGenetic Programming, Proceedings of EuroGP'2000

[Brooks 1989] Brooks R.A., How To Build Complete Creatures Rather Than Isolated Cognitive Simulators, Architectures for Intelligence, K. VanLehn (ed), Erlbaum, Hillsdale, NJ, pp. 225--239. Fall 1989.

[Brueckner 2000] Brueckner S.A., Return from the Ant, PhD Dissertation Humboldt University 2000.

[Cao, Fukunaga and Kahng 1997] Cao U.Y., Fukunaga A.S. and Kahng A.B., Cooperative Mobile Robotics: Antecedents and Directions, Autonomous Robots, 1, 1-23, Kluwer Academis Publishers, Boston.1997

[Castelpietra et al 2000] Castelpietra C., Iocchi L., Nardi D., Piaggio M., Communication and Coordination among Heterogeneous Mid-size Players: ART99, In Proceedings of the 4th International Workshop on RoboCup, pages 149--158, Melbourne, Australia, August 2000

[Chaimowicz et al 2001] Chaimowicz L., Sugar T., Kumar V. and Campos M.F., An Architecture for Tightly Coupled Multi-Robot Cooperation, Proceedings of the 2001 IEEE International Conference on Robotics and Automation, pp. 2292-2297. Seoul – Korea, May, 2001.

[Cheng and Padgham 1997] Cheng S. and Padgham L., From Roles to Teamwork: a framework and architecture, Applied Artificial Intelligence Journal 1997.

[Christoper, Watkins and Dayan 1992] Christoper J., Watkins C.H. and Dayan P., Q-learning, Machine Learning, 8:279:292, 1992.

[Claus and Boutilier 1997] Claus C. and Boutilier C. The dynamics of reinforcement learning in cooperative multiagent systems. In Proceedings of the Tenth Innovative Applications of Articial Intelligence Conference AAAI-97, pages 746-752, Madison, Wisconsin, USA, July 1998.

[Clearwater and Huberman 1994] Clearwater S. and Huberman B., Thermal Models for Controlling Building Environments. Energy Engineering, 91(3):25--56, 1994.

[Dias and Stentz 2000] Dias M. B. and Stentz A., A Free Market Architecture for Distributed Control of a Multirobot System, 6th International Conference on Intelligent Autonomous Systems (IAS-6), July, 2000, pp. 115-122

[De Wilde 2000] De Wilde P., Diversity of Software Agents and Population Dynamics, http://www.ee.ic.ac.uk/philippe/

[Dorigo, Maniezzo and Colorni 1996] Dorigo M., Maniezzo V. and Colorni A., Ant System: Optimization by a Colony of Cooperating Agents, IEEE Transactions On Systems, Man, Ans Cybernetics-Part B: Cybernetics, Vol 26, N01, February 1996.

[Ekbom and Astor 1996] Ekbom K. and Astor E., Investigations of Multi-Agent Systems in Computational Markets and Ecosystems, Research-Report 14-96, University of Karlskrona, October 1996.

[Fegurson 1989] Fergurson D.F., The Application of Microeconomics to the Design of Resource Allocation and Control Algorithms, PhD Dissertation Columbia University 1989.

[Ferguson et al 1996] Fergurson D.F., Christos N., Jakka S. and Yechiam Y. Economic Models for Allocating Resources in Computer Systems, In Scott Clearwater, editor, Market-Based Control: A Paradigm for Distributed Resource Allocation, Scott Clearwater. World Scientific, Hong Kong, 1996

[Fierro et al 2001]. Fierro R, Das A. K., Kumar V. and Ostrowski J. P., Hybrid Control of Formations of Robots, IEEE Int. Conf. on Robotics and Automation, Seoul, Korea, May 2001, pp. 157-162.

[Gerkey and Mataric 2000] Gerkey B.P. and Mataric M.J., Murdoch: Publish/Subscribe Task Allocation for Heterogeneous Agents, Proceedings Agents 2000, Barcelona Spain June 2000

[Gerkey and Mataric 2001] Gerkey B.P. and Mataric M.J., Sold!. Market methods for multi-robot control, Technical Report IRIS-01-399, Institute for Robotics and Intelligent Systems, University of Southern California

[Glance 1993] Glance N.D., Dynamics with Expectations, PhD Dissertation Stanford University 1993.

[Goessman, Hemelrijk and Huber 2000] Goessmann C., Hemelrijk C. and Huber R., The Formation and Maintenance of Crayfish Hierarchies, Behavioral Ecology Sociobiology 2000, 48 418:428.

[Goldberg 1989] Goldberg D.E., Genetic Algorithms in Search, Optimization and Machine Learning.

[Goldberg and Mataric 1997] Goldberg D. and Mataric M.J., Interference as a Tool for Designing and Evaluating Multi-Robot Controllers, Proceedings AAAI-1997, 637-642..

[Good 2000] Good B.M., Evolving Multi-Agent Systems: Comparing Existing Approaches and Suggesting New Directions, Mster's thesis, University of Sussex, 2000.

[Grabowski et al 2000] Grabowski R., Navarro-Serment L.E., Paredis C.J.J. and Khosla P.K., Heterogeneous Teams of Modular Robots for Mapping and Exploration, 2000

[Hogg and Huberman 1991] Hogg T. and Huberman B. A., Controlling Chaos in Distributed Systems, IEEE Transactions on Systems, Man, and Cybernetics, Vol. 21, No. 6, November/December 1991.

[Hong and Page 1998a] Hong L. and Page S.E., Diversity and Optimality, http://ishi.lanl.gov/Documents1.html.

[Hong and Page 1998b] Hong L. and Page S.E, Problem Solving by Heterogeneous Agents, http://ishi.lanl.gov/Documents1.html.

[Hu and Wellman 1998] Hu J. and Wellman M.P., Online Learning about Other Agents in a Dynamic Multiagent System, Proceedings of the 2nd International Conference on Autonomous Agents (Agents'98), pp 239-246, ACM Press 1998.

[Huberman and Hogg 1993] Huberman B.A. and Hogg T., The Emergence of Computational Ecologies, SFI 1992 Lectures in Complex Systems, 163-184, Addison-Wesley 1993.

[Johnson, de la Rosa and Kim 1998a] Johnson J., de la Rosa J.Ll., and Kim J.H., “Benchmark Tests in the Science of Robot Football” Proceedings IEEE of Mirosot-98, pp. 85-89, Paris 1998. R. J. Stonier (ed). Univ. Central Queensland.

[Johnson, de la Rosa and Kim 1998b] Johnson J., de la Rosa J.Ll., and Kim J.H., “Benchmark Tests of Robot Soccer Ball Control Skills” Proceedings IEEE of Mirosot-98, pp. 91-93, Paris 1998. R. J. Stonier (ed). Univ. Central Queensland.

[Kaelbling, Littman and Moore 1996] Kaelbling L.P., Littman L.M and Moore A.W, Reinforcement Learning: A Survey, Journal of Artificial Intelligence Research, pp 237-285, May 1996.

[Kauffmann 1991] Kauffmann S.A, Antichaos and Adaptation, Scientific American, pp. 78-84, August 1991.

[Kitano et al 1997] Kitano H., Stone P., Veloso M., Coradeschi S., Osawa E. and Matsubara H., The RoboCup Synthetic AgentChallenge 97, XV IJCAI-97 International Joint Conference on Artificial Intelligence, Vol 1, pp.24-29, Nagoya, August 1997.

[Kitano et al 1999] Kitano H., Tadokoro S., Noda I., Matsubara H., Takahashi T., Shinjou A. and Shimada S., RoboCup Rescue: Search and Rescue in Large-Sclae Disasters as a Domain for Autonomous Agents Research, Systems, Man, and Cybernetics, 1999. IEEE SMC '99 Conference Proceedings. 1999 IEEE International Conference on , Volume: 6 , 1999 Page(s): 739 -743 vol.6

[Kostiados and Hu 1999] Kostiadis K. and Hu H., Reinforcement Learning and Co-operation in a Simulated Multi-agent System, Proc. of IROS'99, Korea, October 1999.

[Kurabayashi et al 1999] Kurabayashi D., Ota J., Arai T. and Noguchi K., Behavior of a Group Composed of Robots with Heterogeneous Motion Algorithms, Proceedings of the 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems.

[Lander 1994] Lander S.E., Distributed Search and Conflict Management Among Reusable Heterogeneous Agents, PhD Thesis, University of Massachusetts, May 1994.

[Lucas 1997] Lucas C., Self-Organization FAQ
http://psoup.math.wisc.edu/archive/sosfaq.html, 1997.

[Luke et al 1997] Luke S., Hohn C., Farris J., Jackson G. and Hendler J., Co-Evolving Soccer SoftBot Team Coordination with Genetic Programming, Proceedings of the First International Workshop on RoboCup, at the International Joint Conference on Artificial Intelligence pp 398-412.

[Lubbers and Spaans 1998] Lubbers J. and Spaans R.R., The Priority/Confidence Model as a Framework for Soccer Agents, In Proceedings of the second RoboCup Workshop, 1998.

[Maniezzo and Carbonaro 1999] Maniezzo V. and Carbonaro A., Ant Colony Optimaztion: An Overview Proceedings of MIC'99, III Metaheuristics International Conference, Brazi

[Martinoli, Goodman and Holland 2001] Martinoli A., Goodman R. and Owen H., Swarn Intelligence Coures, California Institute of Technology 2001.
http://www.micro.caltech.edu/Courses/EE141/course.html

[Mataric 1994] Mataric M.J, Interaction and Intelligent Behaviour, PhD dissertation, MIT, May 1994.

[Mataric 1995] Mataric M.J., Designing and Understanding Adaptive Group Behavior, Adaptive Behavior 4:1, pp 51-80, 1995

[Matsumura 1998] Matsumura T., Description of Team Erika, In Proceedings of the second RoboCup Workshop, 1998.

[Miller and Drexler 1988] Miller M. S. and Drexler K. E., Markets and Computation: Agoric Open Systems, The Ecology of Computation, B. A. Huberman, Ed Amsterdam: North-Holland, pp 133-176, 1988.

[Nakaruma 1997] Nakaruma T., Development of Self-Learning Vision-Based Mobile Robots for Acquiring Soccer Robots Behaviors. In proceedings of RoboCup-97: Robot Soccer World Cup I, Springer Verlag 1997, pp257-276.

[Nagendra, Lesser and Lander 1995] Nagendra M.V., Lesser V.R and Lander S.E. "Learning Experiments in a Heterogeneous Multi-agent System", IJCAI-95 Workshop on Adaptation and Learning in Multiagent Systems, Montreal, Canada, August, 1995.

[Nagendra, Lesser and Lander, 1996] Nagendra M.V., Lesser V.R., and Lander S.E.. Learning organisational roles in a heterogeneous multi-agent system. In Adaptation, Coevolution and Learning in Multiagent Systems: Papers from the 1996 AAAI Spring Symposium, pages 72-77, Menlo Park,CA, March 1996. AAAI Press.

[Noda et al 1998] Noda I., Matsubara H., Kazuo H. and Frank I., Soccer server: A Tool for research on multi-agent systems, Applied Artificial Intelligence, 12:233:250, 1998.

[Oller, de la Rosa and del Acebo 1999] Oller A., de la Rosa J. Ll. and del Acebo E., DPA: Arquitecture for Co-operative Dynamical Physical Agents, MAMAAW’99, June 1999.

[Parker 1994] Parker L.E., Heterogeneous Multi-Robot Co-operation, PhD Thesis M.I.T, May 1994.

[Parker 1998a] Parker L.E., Toward the automated synthesis of co-operative mobile robot teams, Proceedings of SPIE Mobile Robots XIII, vol. 3525, pp 82-93, 1998.

[Parker 1998b] Parker L. E., Adaptive Heterogeneous Multi-Robot Teams, Neurocomputing, special issue of NEURAP '98: Neural Networks and Their Applications, vol. 28

[Parker 1999] Parker L.E., A Case Study for Life-Long Learning and Adaptation in Co-operative Robots Teams, Proceedings of SPIE Sensor Fusion and Decentralised Control in Robotic Systems II, vol. 3839,pp. 92-101, 1999.

[Parker 2000] Parker L. E., Current State of the Art in distributed Autonomous Mobile Robotics, Distributed Autonomous Robotic Systems 4, Lynne E. Parker, George Bekey, and Jacob Barhen (eds.), Springer, 2000: 3-12.

[Parunak 1997a] Parunak H.D., "Go to the Ant": Enginnering Principles from Natural Multi-Agent Systems, Annals of Operations Research 75 (1997) pp.66-101, special issue on AI and Management Science

[Parunak 1997b] Parunak H.D., Toward the Specification and Design of Industrial Synthetic Ecosystems, Fourth International Workshop on Agent Theories, Architectures, and Languages (ATAL'97)

[Parunak 1999] Parunak H.D., From Chaos to Commerce: Practical Issues and Research Opportunities in the Non-linear Dynamics of Decentralized Mananufacturing Systems, A keynote presentation at IMS'99..

[Parunak and Brueckner 2001] Parunak H.D. and Brueckner S., Entropy and Self-Organization in Multi-Agent Systems, Autonomous Agents '01, May 28- June 1, Montreal Canada.

[Parunak, Brueckner and Sauter 2001] Parunak H.D., Brueckner S. and Sauter J., ERIM's Approach to Fine-Grained Agents, Proceedings of the NASA/JPL Workshop on Radical Agent Concepts (WRAC'2001), Greenbelt, MD, 19-21 September 2001.

[Potter, Meeden, Schultz 2001] Potter M.A., Meeden L.A. and Schultz A.C., Heterogeneity in the Coevolved Behaviors of Mobile Robots: The Emergence of Specialists. Proceedings of the Seventeenth International Conference on Artificial Intelligence, August 4-10, Seattle, Washington, USA.

[Ratnieks and Anderson 1999] Ratnieks F.L.W. and Anderson C., Task partitioning in Insect Societies, Insectes Sociaux, 46 (1999) pp 95-108.

[Reynolds 1987] Reynolds C.W., Flocks, Herdes and Schools: A Distributed Behavioural Model, Computer Graphics, vol 21, nº4,pp 25-34, July 1987.

[Rocha 1999] Rocha L. M., Syntactic Autonomy or Why There is no Autonomy without Symbols and how Self-Organisation Systems Might Evolve Them, New York Academy of Sciences, 1999.

[de la Rosa 1993] de la Rosa J. Ll., Heuristics for Co-operation of Expert Systems, Application to Process Control, PhD Thesis, Universitat Autonoma de Barcelona (UAB), 1993.

[de la Rosa et al 1997] de la Rosa J. Ll., Oller A., Vehí J. and Pujol J. Soccer Team based on Agent-Oriented Programming, Robotics and Autonomous Systems. Ed.Elsevier, Vol 21, pp.167-176, October 1997.

[de la Rosa et al 1999] de la Rosa J. Ll., García R., Innocenti B. and Muñoz I. Rogi Team Real: Research on Physical Agents, 3rd Workshop on RoboCup, 16th IJCAI, July 31-August 6, 1999.

[de la Rosa et al 2000] de la Rosa J. Ll., Innocenti B., Figueres A. and Oller A. An example od Dynamical Agents, Workshop in EuroRobocup 2000, Amsterdam, June 2000.

[de la Rosa, Duhot and Muñoz 2000] de la Rosa J.Ll., Duhaut D. and Muñoz I., Experimental Basis for Benchmarking RoboCup WAF 2000 , Tarragona, September 2000.

[Rybski et al 2000] Rybski P., Papanikolopoulos N.P., Stoeter S.A., Krantz D.G., Yesin K.B., Gini M., Voyles R., Hougen D.F., Nelson B. and Erickson M.D., Enlisting Rangers and Scouts for Reconnaissance and Surveillance, IEEE Robotics ans Automation Magazine, December 2000.

[Ryon 2000] Ryon J., Social Organization of Wolves, Canadian Centre for Wolf Research. http://www.wolfca.com/SocialOrg.html 2000.

[Salustowicz, Wiering and Schmidhuber 1998] Salustowicz R.P., Wiering M.A. and Schmidhuber J., Learning Team Strategies: Soccer Case Studies, Machine Learning 1998.

[Schaerf, Shoham and Tennenholtz 1995] Schaerf A., Shoham Y. and Tennenholtz M., Adaptive Load Balancing: A Study in Multi-Agent Learning, Journal of Artificial Intelligence Researc 2 (1995) pp. 475-500.

[Shannon 1948] Shannon C.E., A Mathematical Theory of Computation, The Bell System Technical Journal, Vol 27 pp379-423 and 623-656, July-October 1948.

[Shoham 1993] Shoham Y., "Agent-oriented programming". Artificial Intelligence, 60:51-92, 1993.

[Shoham and Tennenholtz 1994] Shoham Y. and Tennenholtz M., Co-Learning and the Evolution of Social Activity, Technical Report STAN-CS-TR-94-1511.

[Singh and Fujimura 1993] Singh K. and Fujimura K., Map Making by Cooperating Mobile Robots, In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) 1993.

[Sigmund 1998] Sigmund K., The Population Dynamics of Comflict and Cooperation, IR-98-102, International Institute for Applied Systems and Analysis.

[Simmons et al 2000a] Simmons R., Apfelbaum D., Burgard W., Fox D., Moors M., Thrun S. and Hakan Y., Coordination for Multi-Robot Exploration and Mapping. Proceedings of the National Conference on Artificial Intelligence (AAAI) 2000.

[Simmons et al 2000b] Simmons R., Apfelbaum David, Fox Dieter, Goldman Robert P., Zita K., Musliner D.J., Pelican M. and Thrun S., Coordinated Deployment of Multiple, Heterogeneous Robots, In Proceedings of Conference on Intelligent Robotics and Systems (this volume), Takamatsu Japan, 2000.

[Simmons et al 2000c] Simmons R., Singh S. , Ramos J. and Smith T., Coordination of Heterogeneous Robots for Large-Scale Assembly, Proceedings of the International Symposium on Experimental Robotics (ISER), Hawaii, December 2000

[Steiglitz, Honig and Cohen 1995] Steiglitz K., Honig M. L. and Cohen M.A., Computational Market Model Based on Individual Action, A Paradigm for Distributed Resource Allocation, Scott Clearwater, World Scientific, Hong Kong 1995.

[Stilwell and Bay 1993] Stilwell D.J. and Bay J.S., Toward the Development of a Material Trnasport System using Swarms of Ant-like Robots, IEEE Conf. On Robotics and Automation , Atlanta, GA, 1993, pp 766-771.

[Stone 1998] Stone P., Layered Learning in Multi-Agent Systems, PhD Thesis, Carnegie Mellon University, December 1998.

[Stone and Veloso 1999] Stone P. and Veloso M. Task Descomposition, Dynamic Role Asignment, and Low-Bandwith Communication for Real-Time Strategic Teamwork, Artificial Intelligence 1999.

[Stone and Veloso 2000] Stone P. and Veloso M., Multiagent Systems: A survey from a Machine Learning Perpective, Autonomous Robotics volume 8, number 3, July 2000.

[Tambe et al 2000] Tambe M., Pynadath D. V., Chauvat N., Das A. and Kamink G.A., Adaptive Agent Integration Architectures for Heterogeneous Team Members, Proceedings of the Fourth International Conference on MultiAgent Systems (ICMAS-2000)

[Thayer et al 2000] Thayer S. M., Dias M., Nabbe B., Digney B., Hebert M. and Stentz A., Distributed Robotic Mapping of Extreme Environments, Proceedings of SPIE: Mobile Robots XV and Telemanipulator and Telepresence Technologies VII, Vol. 4195, November, 2000.

[Thomas and Sycara 1997] Thomas J.D. and Sycara K., Hetereogenity, Stability and Efficiency in Distributed systems, Proceedings of the Third Annual Conference on Multiagent Systems (ICMAS '98), IEEE, November, 1997, pp. 293 - 300.

[Tomlinson, Blumberg and Rhode 2000] Tomlinson B., Blumberg B. and Rhodes B., How Is an Agent Like a Wolf?: Dominance and Submission in Multi-Agent systems, In Proceedings of the International ICSC Symposium on Multi-Agents and Mobile Agents in Virtual Organizations and E-Commerce (MAMA'2000). December 11-13, 2000. Wollongong, Australia. CD-ROM.

[Tomlinson and Blumberg 2001] Tomlinson B. and Blumberg B., Social Behavior, Emotion and Learning in a Pack of Virtual Wolfs. 2001 AAAI Fall Symposium "Emotional and Intelligent II: The Tangled Knot of Social Cognition". November 2-4, 2001, North Falmouth, MA.

[Unsal 1993] Unsal C., Master Thesis: Self-Organisation in Large Populations of Mobile Robots, Virgina Polythenic Institute, May 1993.

[Unsal and Bay 1994] Unsal C. and Bay J. S., Spatial Self-Organization in Large Populations of Mobile Robots, 1994 IEEE International Symposium on Intelligent Control, 16-18 August. Columbus, Ohio, USA.

[Vaughan et al 2000] Vaughan R.T., Stoy K., Sukhatme G.S. and Mataric M.J., Go ahead, make my day: Robot Conflict Resolution by Aggressive Competition, Proc. 6th Int. Conf. Simulation of Adaptive Behaviour, Paris, France 2000.

[Waldspurger et al 1992] Waldspurger C.A., Hogg T., Huberman B., Kephart J.O. and Stornetta W. Spawn: A Distributed Computational Economy, IEEE Transactions on Software Engineering, Vol.18 No2, pp. 103-117, Feb. 1992.

[Weiss 1996] Weiss G., Adaptation and Learning in Multi-Agent Systems: Some Remarks and a Bibliography, Adaption and Learning in Multi-Agent Systems, Lecture Notes in Artificial Intelligence, Volume 1042 1996.

[Wellman and Hu 1998] Welman M. P. and Hu J., Conjectural Equilibrium in Multiagent Learning, Machine Learning 33, pp 1-23, Kluwer Academic Publishers 1998.

[Wolpert and Tumer 1999] Wolpert D.H. and Tumer K., An Introduction To Collective Intelligence, In J. M. Bradshaw, editor, Handbook of Agent Technology. AAAI Press/MIT Press 1999.

[Wooldridge and Jennings 1995] Wooldridge M. and Jennings N.R., Intelligent Agents: Theory and Practice, The Knowledge Engineering Review, volume 10, pp 115-152, 1995.

[Wooldridge 2000] Wooldridge M., Engineering the computational economy, IST-2000: Proceedings of the Information Society Technologies Conference, Nice, France, November 2000.

[Ygge and Akkermanas 1996] Ygge F. and Akkermanas F., Power Load anagement as a Computational Market, Proceedings of the 2nd International Conference on Multi-Agent Systems ICMAS'96, Kyoto , Japan, December 1996.

[Yovits, Jacobi and Goldstein 1962] Yovits M.C, Jacobi G.T. and Goldstein G.D., Self-Organizing Systems, Self-Organising Systems, Washington D.C., McGregor&Werner, 1962.

[Zhang and Mackworth 1994] Zhang Y. and Mackworth A.K., Specification and verification of constraint-based dynamic systems, Principles and Practice of Constraint Programming, Lecture Notes in Computer Science number 874, pp 229-242, 1994.

[Zhang and Mackworth 1995] Zhang Y. and Mackworth A.K., Constraint Nets: A Semantic Model for Hybrid Dynamic Systems, Theoretical Computer Science 130, pp 211-239, 1995.

[Zhang and Mackworth 1998] Zhang Y. and Mackworth A.K., A Multi-level Constraint-based Controller for the Dynamo98. In Proceedings of the second RoboCup Workshop 402-409, 1998.