that as the number of requests increases, the case-base learns properly the new information and hence, the number of routes that can be composed with the cases increases (and an estimation is not necessary). Figure 5, which shows the relation between the number of cases in the case-base and the percentage of estimated routes, also supports this conclusion. Finally, the percentage of distrust from which an agent can commit itself to performing a service was also checked (modifying the confidence values from 70%, 80% and 90%). As expected, bigger confidence percentages resulted in agents committing themselves to performing more tasks (Figure 6). However, in such cases the percentage of services accepted and completed on time decreases, since the agent committed itself to the performance of a big amount of services (Figure 7).

6 Conclusions

A module to analyse whether an agent has enough time to perform a service has been developed. A CBR approach for deciding if an agent can commit itself to performing some service without exceeding the maximum time assigned for performing that service has been used. From now, we have focused our work on the design and implementation of the different phases of the CBR cycle for the mobile robot problem. This work has been tested and evaluated by using a simulated scenario. The results obtained support the expectations.

References

1. Winikoff, M.: Designing commitment-based agent. In: International Conference on Intelligent Agent Technology, pp. 18–22 (2006)
2. Julián, V., Botti, V.: Developing Real-Time Multi-agent Systems. Integrated Computer-Aided Engineering 11(2), 135–149 (2004)
3. Aamodt, A., Plaza, E.: Case-based reasoning; Foundational issues, methodological variations, and system approaches. AI Comm. 1(7), 39–59 (1994)
4. Dean, T., Boddy, M.: An analysis of time-dependent planning. In: Proceedings of the seventh National Conference on AI, pp. 49–54 (1988)
5. Hart, P.E., Nilsson, N.J., Raphael, B.: A formal basic for the heuristic determination of minimum cost paths. IEEE Transactions on SSC 4, 100–107 (1968)
6. Navarro, M., Julián, V., Soler, J., Botti, V.: jART: A Real-Time Multi-Agent Platform with RT-Java. In: Proc. 3rd IWPAAMS 2004, pp. 73–82 (2004)
7. RTJ: The Real-Time for Java Expert Group, http://www.rtsj.org
8. Webots: http://www.cyberbotics.com/

Hybrid Multi-agent Architecture (HoCa) Applied to the Control and Supervision of Patients in Their Homes

Juan A. Fraile[1], Javier Bajo[1], and Juan M. Corchado[2]

[1] Pontifical University of Salamanca, Compañía 5, 37002 Salamanca, España
[2] Departamento de Informática y Automática, Universidad de Salamanca, Plaza de la Merced s/n, 37008 Salamanca, España
{jafraileni,jbajope}@upsa.es, corchado@usal.es

Abstract. This paper presents a Hybrid Multi-Agent Architecture for the control and supervision of dependent environments. HoCa (Home health Care) integrates various modules to allow an agile exchange of information, minimizing the necessary message traffic and optimizing the efficiency. HoCa is based on an Ambient Intelligence model and integrates a management system of alerts and an automated identification, localization, and movement control system. The core of the architecture is formed by both deliberative agents and reactive agents that interact to offer efficient services. The architecture has been successfully tested in a real environment.

Keywords: Dependent environments, Ambient Intelligence, Multiagent Systems, Home Care.

1 Introduction

In recent years, there has been an important growth in the field of Ambient Intelligence (AmI) [1] [13], involving major changes in the daily lives of people. The vision of Ambient Intelligence implies the creation of intelligent spaces where users interact in a natural way with computational systems and communication technologies, which become invisible and ubiquitous. The technology is adapted to individuals and their context, acting autonomously, and facilitating their daily tasks. One of the main aims of Ambient Intelligence focuses in building systems that support the activities of daily living in an efficient way. For example, activities related to the home automation. These systems can be developed under various scenarios and are known as "ubiquitous intelligent environments" [9], where it is possible to solve the challenge of developing strategies for early problems detection and prevention in automated environments. The society will benefit from the technological advances provided by the AmI, but particularly dependent people will see more improved their quality of life, thanks to future generations of products based on these advances.

The complexity of these systems requires advanced control architectures, which have gained increasing importance in recent years. These architectures must provide novel structures, information exchange mechanisms and computer resources management. Besides, most of the current systems require concurrent work to provide real

H. Geffner et al. (Eds.): IBERAMIA 2008, LNAI 5290, pp. 193–202, 2008.

time solutions. This paper presents Home Care architecture (HoCa), a novel architecture specifically designed to be implemented in Ambient Intelligence environments. HoCa integrates an alert management system and an identification, location and movement control mechanism based on a multi-agent system. These mechanisms facilitate most of the common tasks required in home care environments, as patients monitoring and tracking, as well as quick response to problematic situations. HoCa must be flexible enough to allow communication between remote modules, adaptation to computational requirements as well as to support the rapid integration of new modules and sensors. Multi-agent systems are very appropriate to satisfy these needs, due to their characteristics [14]. A multiagent system consists of a software agent's networks that interact to solve problems that are beyond individual capabilities. One of the main contributions of HoCa is the use of both reactive and deliberative agents, which facilitates advanced reasoning capabilities together with real-time reactive behaviours. These are two important characteristics that must be taken into account in the development of intelligent environmets.

The paper is organized as follows: The first section presents the problem that prompted this work. The third section presents the proposed architecture, and the fourth section gives the results and conclusions obtained after applying the proposed architecture to a real case in an environment of dependence.

2 General Description of the Problem

The use of intelligent agents is an essential component for analyzing information on distributed sensors [14]. These agents must be capable of both independent reasoning and joint analysis of complex situations in order to be able to achieve a high level of interaction with humans [3]. Although multi-agent systems already exist and are capable of gathering information within a given environment in order to provide medical care [6], there is still much work to be done. It is necessary to continue developing systems and technology that focus on the improvement of services in general. After the development of the internet there has been continual progress in new wireless communication networks and mobile devices such as mobile telephones and PDAs. This technology can help to construct more efficient distributed systems capable of addressing new problems [7].

Hybrid architectures try to combine deliberative and reactive aspects, by combining reactive and deliberative modules [5]. The reactive modules are in charge of processing stimuli that don't need deliberation, whereas the deliberative modules determine which actions to take in order to satisfy the local and cooperative aims of the agents. The aim of modern architectures like Service Oriented Architecture (SOA) is to be able to interact among different systems by distributing resources or services without needing to consider which system they are designed for. An alternative to these architectures are the multi-agent systems, which can help to distribute resources and to reduce the centralization of tasks. Unfortunately the complexity of designing multi-agent architecture is great since there are not tools to either help programme needs or develop agents.

Multi-agent systems combine aspects of both classic and modern architectures. The integration of multi-agent systems with SOA has been recently investigated [2]. Some researchers focus on the communication among these models, whereas others focus on the integration of distributed services, especially web services, in the agents' structure [4] [12]. These works provide a good base for the development of multi-agent systems. Because the majority of them are in the development stage, their full potential in a real environment is not known. HoCa has been implemented in a real environment and not only does it provide communication and integration among distributed agents, services and applications, but it also provides a new method for facilitating the development of multi-agent systems, thus allowing the agents and systems to function as services. HoCa also implements an alert and alarm system across the agent's platform, specially designed to be used by mobile devices. The platform agents manage this service and determine the level of alert at every moment so that they can decide who will receive the alert and when. The alerts are received by the subscribed in the system that have associated such alerts. Hours sending alerts depend on the system parameters and the urgency of the warning. In order to identify each user, HoCa implements a system based on Java Card [15] and RFID (Radio Frequency IDentification) microchip technology in which there will be a series of distributed sensors that provide the necessary services to the user.

3 HoCa Architecture

The HoCa model architecture uses a series of components to offer a solution that includes all levels of service for various systems. It accomplishes this by incorporating intelligent agents, identification and localization technology, wireless networks and mobile devices. Additionally, it provides access mechanisms to multi-agent system services, through mobile devices, such as mobiles phones or PDAs.

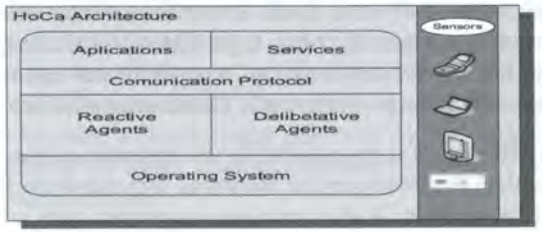

Fig. 1. HoCa Framework

Access is provided via wi-fi wireless networks, a notification and alarm management module based on SMS (Short Message Service) and MMS (Multimedia Messaging System) technologies, and user identification and localization system based on Java Card and RFID technologies. This system is dynamic, flexible, robust and very adaptable to changes of context.

HoCa architecture describes four basic blocks that can be seen in Figure 1: Applications, Services, Agents Platform and Communication Protocol. These blocks constitute the whole functionality of the architecture. The applications represent all programs that can be used to exploit the system functionalities. The services represent the functionalities that the architecture offers. The Agents Platform and the

Communication Protocol are explained in detail in the following subsections. Moreover, the identification and alert systems integrated within HoCa are also described, presenting their appropriateness for home care environments.

3.1 Agents Platform in HoCa

The agents platform is the core of the architecture and integrates two types of agents, each of which behaves differently for specific tasks, as shown in Figure 2.

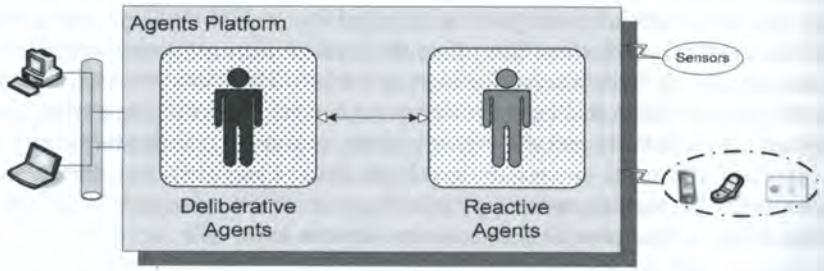

Fig. 2. Agents platform structure in the HoCa architecture

The first group of agents is made up of deliberative BDI agents, which are in charge of the management and coordination of all system applications and services. These agents are able to modify their behaviour according to the preferences and knowledge acquired in previous experiences, thus making them capable of choosing the best solution. Deliberative agents constantly deal with information and knowledge. Because they can be executed on mobile devices, they are always available and they provide ubiquitous access for the users. There are different kinds of agents in the architecture, each one with specific roles, capabilities and characteristics. This fact facilitates the flexibility of the architecture to incorporate new agents. However, there are pre-defined agents which provide the basic functionalities of the architecture:

- CoAp Agent: This agent is responsible for all communications between applications and the platform. Manages the incoming requests from the applications to be processed by services. It also manages responses from services to applications. CoAp Agent is always on listening mode. Applications send XML messages to the agent requesting for a service.
- CoSe Agent: It is responsible for all communications between services and the platform. The functionalities are similar to CoAp Agent but backwards. This agent is always on listening mode waiting for responses of services. Manager Agent indicates CoSe Agent the service that must be invoked. Then, CoSe Agent sends an XML message to the service.
- Directory Agent. Manages the list of services that can be used by the system. For security reasons, the list of services is static and can only be modified manually, however services can be added, erased or modified dynamically. The list contains the information of all trusted available services.
- Supervisor Agent. This agent supervises the correct functioning of the agents in the system. Supervisor Agent verifies periodically the status of all agents registered in the architecture by means of sending ping messages. If there is no response, the agent kills the agent and creates another instance of that agent.
- Security Agent. This agent analyzes the structure and syntax of all incoming and outgoing XML messages. If a message is not correct, the Security Agent informs the corresponding agent that the message cannot be delivered.
- Manager Agent. Decides which agent must be called taking into account the users preferences. Users can explicitly invoke a service, or can let the Manager Agent decide which service is better to accomplish the requested task. If there are several services that can resolve the task requested by an application, the agent selects the optimal choice using a strategy based on the analysis of the efficiency of the services. An optimal choice service has higher and better performance that other. Manager Agent has a routing list to manage messages from all applications and services.
- Interface Agent. This kind of agent has been designed to be embedded in users' applications. Interface agents communicate directly with the agents in HoCa so there is no need to employ the communication protocol, but FIPA ACL specification. The requests are sent directly to the Security Agent, which analyzes the requests and sends them to the Manager Agent.

The second group is made up of reactive agents. Most of the research conducted within the field of multi-agent systems focuses on designing architectures that incorporate complicated negotiation schemes as well as high level task resolution, but don't focus on temporal restrictions. In general, the multi-agent architectures assume a reliable channel of communication and, while some establish deadlines for the interaction processes, they don't provide solutions for limiting the time the system may take to react to events. It is possible to define a real-time agent as an agent with temporal restrictions for some of its responsibilities or tasks [11]. From this definition, we can define a real-time multi-agent system (Real Time Multi-Agent System, RT-MAS) as a multi-agent system in which at least one of the agents is a real-time agent. The use of RT-MAS makes sense within an environment of critical temporal restrictions, where the system can be controlled by autonomous agents that need to communicate among themselves in order to improve the degree of system task completion. In this kind of environments every agent requires autonomy as well as certain cooperation skills to achieve a common goal.

3.2 HoCa Communication Protocol

Communication protocol allows applications, services and sensors to be connected directly to the platform agents. The protocol presented in this work is open and independent of programming languages. It is based on the SOAP standard and allows messages to be exchanged between applications and services as shown in Figure 3.

However, interaction with environmental sensors requires Real-time Transport Protocol (RTP) [10] [5] which provides transport functions that are adapted for applications that need to transmit real-time data such as audio, video or simulation data, over multicast or unicast network services. The RTCP protocol is added to RTP, allowing a scaleable form of data supervision. Both RTP and RTCP are designed to work independently from the transport and lower network services. They

Fig. 3. Communication using SOAP messages in HoCa

are in charge of transporting data with real-time characteristics, and of supervising the quality of service, managing the information for all the entities taking part in the current session.

The communications between agents within the platforms follows the FIPA ACL (Agent Communication Language) standard. This way, the applications can use the platform to communicate directly with the agents.

3.3 Location and Identification System in HoCa

This system incorporates Java Card [15] and RFID [8] technologies. The primary purpose of the system is to convey the identity of an object or person, as with a unique serial number, using radio waves. Java Card is a technology that permits small Java applications (applets) to be run safely in microchip smart cards and similar embedded devices. Java Card gives the user the ability to program applications that can be run off a card so that it has a practical function in a specific application domain. The main features of Java Card are portability and security; it is described in ISO 7816. The data are stored in the application and the Java Card applications are executed in an isolated environment, separated from the operating system and from computer that reads the card. The most commonly used algorithms, such as DES, 3DES, AES, and RSA, are cryptographically implemented in Java Card. Other services such as electronic signature or key generation are also supported.

RFID technology is grouped into the so-called automatic identification technologies. But RFID provides more information than other auto-identification technologies, speeds up processes without losing reliability, and requires no human intervention.

The combination of these two technologies allows us to both identifiable element, and to locate it, by means of sensors and actuators, within the environment, at which time we can act on it and provide services. The microchip, which contains the identification data of the object to which it is adhered, generates a radio frequency signal with this data. The signal can be picked up by an RFID reader, which is responsible for reading the information and sending it, in digital format, to the specific application.

3.4 Alert System in HoCa

The alert system is integrated into the HoCa architecture and uses mobile technology to inform users about alerts, warnings and information specific to the daily routine of

the application environment. This is a very configurable system that allows users to select the type of information they are interested, and to receive it immediately on their mobile phone or PDA. It places the information to be sent into information categories. The users determine the information they are interested in. The system automatically sends the information to each of the users as soon as it is available.

4 Using HoCa to Develop a Multi-agent System for a Home Care Dependent Environment

Ambient Intelligence based systems aim to improve people quality of life, offering more efficient and easy to use services and communication tools to interact with other people, systems and environments. One of the most benefited segments of population with the development of these systems is elderly and dependent people. Agents and multi-agent systems in dependency environments are becoming a reality, specifically on health care. Most agents-based applications are related to the use of this technology in patients monitoring, treatment supervision and data mining.

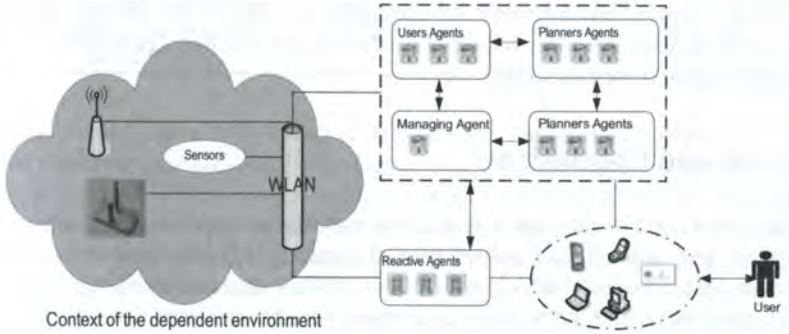

Fig. 4. HoCa structure in a dependent environment

HoCa has been employed to develop a multi-agent system aimed to enhance assistance and care for low dependence patients at their homes. Main functionalities in the system include reasoning, planning mechanisms, management alerts and responses in execution time offered to certain stimuli, as shown in Figure 4. These functionalities allow the system the use of several context-aware technologies to acquire information from users and their environment. Among the technologies used are mobile systems for alerts service managing across PDA and mobile phones, Java Card elements for identification and presence detectors and access control.

Each agent in the system has its own functionalities. If an agent needs to develop a task in collaboration with other agent a request form is sent. There are priority tasks that a set of agents can perform. This ensures that the priority tasks are always available. There are four specific types of deliberative BDI [14] agents included in the case study which collaborate with the predefined agents in the HoCa platform:

- User Agent manage user personal data and their behavior. They are responsible through the system, identify and locate implemented by the architecture. They

determine the status of the user and offering services in the environment as a correct temperature, automatic lighting, access blocking or opening, etc.

- SuperUser Agent runs on mobile devices and inserts new tasks into the ManagerALZ Agent to be processed by a reasoning mechanism. It also needs to interact with the User Agents to impose new tasks and receive periodic reports, and with the ScheduleUser Agents to ascertain plans' evolution.
- SheduleUser Agent schedules the users' daily activities obtaining dynamic plans depending on the tasks needed for each user. It manages scheduled-users profiles, tasks, available time and resources. Every agent generates personalized plans depending on the scheduled-user profile.
- ManagerALZ Agent runs on a Workstation and plays two roles: the physical security role that monitors the users and the managerAlz role that handle the data and the tasks assignment for the medical staff. It must provide security for the users and ensure the tasks assignments are efficient.

On the other hand there are a number of reactive agents that work in collaboration with the deliberative agents. These agents are in change of control devices interacting with sensors (access points, lights, temperature, alarms detection, etc.). They receive information, monitor environment services and also check the devices status connected to the system. All information is treated by the reactive agent and it is sent to the manager agent to be processed.

5 Results and Conclusions

HoCa has been used to develop a system for monitoring dependent patients at home. The testing environment has evolved from the one used for the previous ALZ-MAS architecture [6]. The ALZ-MAS architecture allows the monitoring of patients in geriatric residences, but home care is carried out through traditional methods. As mentioned in [6], ALZ-MAS present certain advantages with respect to previous architectures. However, one of the problems of ALZ-MAS was the number of agents crashed during tests. HoCa solves this problem using replicated services and reducing the crashes an average of 9%. Moreover, HoCa incorporates mobile SMS technology for managing service alerts through PDAs and mobile phones, which provides a remote alert system, and Java Card technology for identification and access control, which improves the previous RFID location technology. The environment includes reasoning and planning mechanisms, and alert and response management. Most of these responses are reactions in real time to certain stimuli, and represent the abilities that the reactive agents have in the HoCa architecture based platform. Real-time systems require an infrastructure characterized by their response time for computing and communication processes [5]. The time is considered as a critical parameter [10] to react to sensor values, i.e. when a door has to be automatically open or closed after detecting a patient identification. In this kind of situations the reactive agents receive behaviors from the deliberative agents and take control of the actions carried out.

One of the main contributions of the HoCa architecture is the remote alert system. We implemented several test cases to evaluate the management of alerts integrated into the system. This allowed us to determine the response time for warnings generated by the users, for which the results were very satisfactory, with response times

shorter than those obtained prior to the implementation of HoCa. The system studies the information collected, and applies a reasoning process which allows alerts to be automatically generated, trying to avoid false alarms. For these alerts, the system does not only take response time into account, but also the time elapsed between alerts, and the user's profile and reliability, in order to generalize reactions to common situations. The results show that HoCa fits perfectly within complex systems by correctly exploiting services and planning mechanisms.

Table 1. Comparison between the HoCa and the ALZ-MAS architectures

Factor	HoCa	ALZ-MAS
Average Response Time to Incidents(min.)	8 minutes	14 minutes
Assisted Incidents	12	17
Average number of daily planned tasks	12	10
Average number of services completed daily	46	32
Time employed by the medical staff to attend to an alert (min.)	75 minutes	90 minutes

Table 1 presents the results obtained after comparing the HoCa architecture to the previously developed ALZ-MAS architecture [6] in a case study on medical care for patients at home. The case study presented in this work consisted of analysing the functioning of both architectures in a test environment. The HoCa architecture was implemented in the home of 5 patients and was tested for 30 days. The results were promising. The data shown in Table 1 are the results obtained from the test cases. They show that the alert system improved the communication between the user and the dependent care services providers. The user identification and location system in conjunction with the alert system has helped to notably reduce the percentage of incidents in the environment under study. Moreover, in addition to a reduction in the number of incidents, the time elapsed between the generation of a warning and solution decreased significantly. Finally, due to the many improvements, the level of user satisfaction increased with the introduction of HoCa architecture since patients can live in their own homes with the same level of care as those offered at the residence.

References

1. Anastasopoulos, M., Niebuhr, D., Bartelt, C., Koch, J., Rausch, A.: Towards a Reference Middleware Architecture for Ambient Intelligence Systems. In: ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications (2005)
2. Ardissono, L., Petrone, G., Segnan, M.: A conversational approach to the interaction with Web Services. Computational Intelligence 20, 693–709 (2004)
3. Bahadori, S., Cesta, A., Grisetti, G., Iocchi, L., Leone1, R., Nardi, D., Oddi, A., Pecora, F., Rasconi, R.: RoboCare: Pervasive Intelligence for the Domestic Care of the Elderly. AI*IA Magazine Special Issue (January 2003)

4. Bonino da Silva, L.O., Ramparany, F., Dockhorn, P., Vink, P., Etter, R., Broens, T.: A Service Architecture for Context Awareness and Reaction Provisioning. In: IEEE Congress on Services (Services 2007), pp. 25–32 (2007)
5. Carrascosa, C., Bajo, J., Julian, V., Corchado, J.M., Botti, V.: Hybrid multi-agent architecture as a real-time problem-solving model. Expert Systems With Applications 34(1), 2–17 (2008)
6. Corchado, J.M., Bajo, J., de Paz, Y., Tapia, D.: Intelligent Environment for Monitoring Alzheimer Patients, Agent Technology for Health Care. Decision Support Systems 34(2), 382–396 (2008)
7. Corchado, J.M., Bajo, J., Abraham, A.: GERAmI: Improving the delivery of health care. IEEE Intelligent Systems. Special Issue on Ambient Intelligence (March/April 2008)
8. ITAA. Radio Frequency Identification. RFID...coming of age. Information Technology Association of America (2004), http://www.itaa.org/rfid/docs/rfid.pdf
9. Kleindienst, J., Macek, T., Seredi, L., Sedivy, J.: Vision-enhanced multi-modal interactions in domotic environments. IBM Tecnologías de Voz y Sistemas. República Checa (2004)
10. Jacobsen, V., Fredrick, R., Casner, S., Schulzrinne, H.: RTP: A transport Protocol for Real-Time Applications. RFC 1889. Lawrence Berkeley National Laboratory, Xerox PARC, Precept Software Inc., GMD Fokus (January 1996), http://www.connect.org.uk/teckwatch/cgi-bin/rfcshow?1889 (accessed October 16, 1996)
11. Julian, V., Botti, V.: Developing real-time multi-agent systems. In: Proceedings of the Fourth Iberoamerican Workshop on Multi-Agent Systems (Iberagents 2002), Málaga (2004)
12. Ricci, A., Buda, C., Zaghini, N.: An agent-oriented programming model for SOA & web services. In: 5th IEEE International Conference on Industrial Informatics (INDIN 2007), Vienna, Austria, pp. 1059–1064 (2007)
13. Richter, K., Hellenschmidt, M.: Interacting with the Ambience: Multimodal Interaction and Ambient Intelligence. In: W3C Workshop on Multimodal Interaction, July 19-20 (2004) (position paper)
14. Tapia, D.I., Bajo, J., De Paz, F., Corchado, J.M.: Hybrid Multiagent System for Alzheimer Health Care. In: Rezende, S.O., da Silva Filho, A.C.R. (eds.) Proceedings of HAIS 2006. Ribeirao Preto, Brasil (2006)
15. ZhiqunChen (Sun Microsystems). Java Card Technology for Smart Cards. Addison Wesley/Longman ISBN 0201703297

Mixing Greedy and Evolutive Approaches to Improve Pursuit Strategies

Juan Reverte, Francisco Gallego, Rosana Satorre, and Faraón Llorens

Department of Computer Science and Artificial Intelligence
University of Alicante
{jreverte,fgallego,rosana,faraon}@dccia.ua.es

Abstract. The prey-predator pursuit problem is a generic multi-agent testbed referenced many times in literature. Algorithms and conclusions obtained in this domain can be extended and applied to many particular problems. In first place, greedy algorithms seem to do the job. But when concurrence problems arise, agent communication and coordination is needed to get a reasonable solution. It is quite popular to face these issues directly with non-supervised learning algorithms to train prey and predators. However, results got by most of these approaches still leave a great margin of improvement which should be exploited.

In this paper we propose to start from a greedy strategy and extend and improve it by adding communication and machine learning. In this proposal, predator agents get a previous movement decision by using a greedy approach. Then, they focus on learning how to coordinate their own pre-decisions with the ones taken by other surrounding agents. Finally, they get a final decission trying to optimize their chase of the prey without colliding between them. For the learning step, a neuroevolution approach is used. The final results show improvements and leave room for open discussion.

Keywords: Multi-agent systems, communication, coordination, neuroevolution.

1 Introduction

The Predator-prey problem (or pursuit domain) is a well-known testbed for multi-agents systems. It consists of world where a group of agents (called predators) aim to chase and surround another agent (called prey) that tries to evade them [1]. The goal of predator agents is to surround (capture) prey without touching it (i.e. occuping the adjacent cells), whilst the goal of the prey, as expected, is not to be captured.

This problem has been adressed many times in literature. Initially, Korf [7] proposed a greedy without inter-agent communication. His approach was to use a fitness function that combined 2 forces: each predator was "attracted" by the prey and "repelled" from the closest other predator. This solution keeped predators away from other predators while they got closer to the prey; the idea

H. Geffner et al. (Eds.): IBERAMIA 2008, LNAI 5290, pp. 203–212, 2008.