REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)* 03-11-2004	**2. REPORT TYPE** Final Report	**3. DATES COVERED** *(From – To)* 1 October 2003 - 03-Jan-05

4. TITLE AND SUBTITLE Situation Assessment and Information Fusion: an experimental framework for performance evaluation	**5a. CONTRACT NUMBER** FA8655-03-1-3065
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Dr. Daniele Nardi	**5d. PROJECT NUMBER**
	5d. TASK NUMBER
	5e. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Università di Roma, Via Salaria 113, ROMA, 00198, Italy	**8. PERFORMING ORGANIZATION REPORT NUMBER** N/A
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) EOARD, PSC 802 BOX 14, FPO 09499-0014	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S) SPC 03-3065

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This report results from a contract tasking Università di Roma as follows: The Grantee will perform high-level Information Fusion research based on multi-agent information monitoring and fusion. The goal is to develop situation assessment techniques and strategies which focus on evolving, dynamic scenarios such as natural disasters (e.g. earthquakes or other catastrophic events). Complete details of expected results and technical contributions are found the research proposal.

15. SUBJECT TERMS

EOARD, Information Fuion, RoboRescue

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18, NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE	UL		PAUL LOSIEWICZ, Ph. D. [illegible]
UNCLAS	UNCLAS	UNCLAS			19b. TELEPHONE NUMBER *(Include area code)* +44 20 7514 4474

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39-18

Situation Assessment and Information Fusion: an experimental framework for performance evaluation GRANT: # 033065 Addendum

I hereby certify that there were no subject inventions to declare during the performance of this grant. Moreover, I confirm that the Acknowledgement of Sponsorship and Disclaimer in the publications of the results of the project follow the indications reported below:

1) Acknowledgement of Sponsorship: The Grantee is responsible for assuring that an acknowledgement of Government support will appear in any publication of any material based on or developed under this project, in the following terms: "Effort sponsored by the Air Force Office of Scientific Research, Air Force Material Command, USAF, under grant number FA8655-03-1-3A46. The U.S. Government is authorized to reproduce and distribute reprints for Government purpose notwithstanding any copyright notation thereon."

2) Disclaimer: The grantee is responsible for assuring that every publication of material based on or developed under this project contains the following disclaimer: "The views and conclusions contained herein are those of the author is and should be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research or the U.S. Government."

Rome, 30th December 2004

[illegible]

Prof. Daniele Nardi

Dipartimento di Informatica e Sistemistica
Universita' di Roma "La Sapienza"
Via Salaria 113, 00198 ROMA, Italy

tel. +39-06-49918330, fax +39-06-85300849
http://www.dis.uniroma1.it/~nardi
email:nardi@dis.uniroma1.it

AQ F05-05-0754

Situation Assessment and Information Fusion: an experimental framework for performance evaluation

Calisi, D., Farinelli, A., Iocchi, L., Nardi, D., Patrizi, F.
Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Salaria 113, 00198 Roma, Italy
last-name@dis.uniroma1.it

October 8, 2004

Abstract

The goal of this project is to address information fusion and situation assessment techniques in a multi-agent system. In the first six months, several different Multi-Agent Systems (MAS) have been developed using the Cognitive Agent Development Kit, and tested using the RoboCup Rescue Simulator. In the second part of the project, in order to address more realistic scenarios, we focused on perception unreliability and active perception, extending our method to take into account those issues and applying the method in a novel experimental domain composed by physical robotic agents.

1 Introduction

The main goal of the project is to address information fusion and situation assessment techniques in a multi-agent system. For this reason, our first step has been to provide the Cognitive Agents (CADK, see [2, 3]) with information fusion capabilities and to test them in the RoboCup Rescue Simulator [1, 6]. In particular, through the use of the CADK we developed several kinds of MAS, testing and comparing them.

However, in order to face more complex and realistic scenarios, we applied our method also in a different experimental domain, composed by physical robotic agents acting in a dynamic environment.

This new experimental domain allows us to consider unreliability in the information gathering process and asynchrony in communication. Moreover, in this new scenario, the relationship among information fusion, global situation assessment and cooperation capability turns out to be even more strict than in simulated environments, and has to be taken into account in order to increment global system efficiency and reliability. Finally, since low level actions (e.g. navigation in the environment) are much more complex and not guaranteed to succeed, information gathering and task assignment processes should be designed to carefully avoid conflicts among action executed by robotic agents.

As described in the first report, the main goals achieved in the first part of the project are as follows:

- **Refinement and extension of the CADK** with specific techniques for information fusion that allow for the integration of information (at different levels of abstraction) coming from several agents in the environment;
- **Experimental evaluation** of the proposed techniques in the scenario built on the simulator and the development of systematic methodologies for an effective evaluation of the information fusion techniques and strategies, as well as the impact of situation assessment on the overall system performance.

In this second part, as stated in the Technical Plan, we probed the following issues:

- **Treatment of errors in information gathering**, taking into account scenarios where information gathering is inherently unreliable;
- **Active perception** exploiting the trade-off between acting and gathering information from the environment.

In the following we describe the results achieved for each of the goals described above.

2 Results

2.1 CADK refinement and extension

In the first six months we addressed the problem of integrating symbol-level data coming from different sources [7] to assess the current environment state and to improve the task allocation process. In fact, by integrating information coming from the other agents, each agent improves its own knowledge and is able to take decisions in a cooperative fashion, thus increasing the overall MAS utility.

In the paper *Task Assignment with dynamic token generation* (see publication list), we address the problem of assigning tasks to a team of robots and present a distributed approach, involving environmental state information exchange among agents. To evaluate the proposed solution, MAS with different features have been implemented through the CADK, focusing the attention on communication and situation assessment issues.

In the second part of the project, after integrating the results achieved in the first part, we focused on extending CADK functionalities in order to address unreliable information and active perception.

2.2 Experimental evaluation

We have investigated techniques of information fusion and situation assessment [5, 4] that can be successfully applied to the rescue domain, by evaluating the results obtained in several simulations. The first step in this direction has been the adoption of an appropriate evaluation methodology to consider both efficiency and reliability of the performance of a MAS with information fusion, situation assessment and task assignment capabilities. We have analyzed the relationship among assessment of situation, efficiency of the cooperation realized by task assignment strategies and communication features.

The paper *Experiments with the RoboCup Rescue Simulator in a post Earthquake emergency Italian Scenario* (see publications) presents the methodology by which we have evaluated and compared MAS. A set of experiments are discussed to validate the methodology and to show that additional features are required to be considered when evaluating performance. Moreover, the interdependencies among information fusion, global situation assessment and cooperation capabilities are discussed.

2.3 Introduction of unreliable information

In this second part of the project we tested our method also in another experimental domain, which allows us to take into account unreliability of information gathering.

In the paper *Task assignment with dynamic perception and constrained tasks in a Multi-Robot System*, we extend the previous task assignment algorithm by introducing unreliability, both on perception and in communications. We describe our design choices for the appli-

cation of our method in a Multi-Robot system, precisely characterizing robots capabilities.
We successfully extend our task assignment approach to take into account possible constraints among tasks to be accomplished. Moreover, we discuss the extension of our method to consider failure in object perceptions and we report quantitative results obtained in a simulation environment of our reference scenario and qualitative results for the experiments performed with real robots.

2.4 Active perception

With "active perception" we refer to the ability of explicitly gathering information from the environment through specific actions.
For example, if a robot does not know with enough precision its position inside the working environment, it might start an active localization procedure (e.g. looking for some known marker inside the environment). Since information gathering actions have a cost (e.g. time needed to execute the action), there is a trade off between the cost to perform a perception action and gaining useful information that help the robot in fulfilling its task.

In the paper *Task assignment with dynamic perception and constrained tasks in a Multi-Robot System* we provide agents with active perception, allowing them to take more consistent actions that improve the overall system performance. We also find active perception more relevant in robotic domains than in simulated ones.

2.5 Other

With respect to the previous report, our task assignment and information gathering method has been deeply investigated, using the RoboCup Rescue simulator. The method demonstrated to be scalable and robust in various operative conditions, changing agents capabilities, agent number and their distribution inside the city map (see *Distributed Task Assignment for Real World Environments* in publications).

3 Publications

Below we list the publications related to the field of research of the present project:

- A. Biagetti, A. Farinelli, L. Iocchi, D. Nardi, F. Patrizi. Experiments with the RoboCup Rescue Simulator in a post Earthquake emergency Italian Scenario. Presented in *Second International Workshop on synthetic simulation and robotics to mitigate earthquake disaster (SRMED 2004), Lisbon, July 2004* and in *IEEE International workshop on safety, security and rescue robotics, Bonn, May 24-26, 2004 (SSRR 2004).*

- A. Farinelli, L. Iocchi, D. Nardi, F. Patrizi. Task Assignment with dynamic token generation. In *Proc. of Int. Workshop on Monitoring, Security, and Rescue Techniques in Multiagent Systems (MSRAS)*, Plock, Poland, 2004.
- A. Farinelli, L. Iocchi, S. Lo Cascio, D. Nardi. Situation Assessment and Information fusion: an experimental framework for performance evaluation. In *AFOSR Workshop on Information Fusion 29-30 June, 2004, Stockholm, Sweden, in conjunction with the 7th International Conference on Information Fusion, Fusion 2004,*
- A. Farinelli, L. Iocchi, D. Nardi, V. Ziparo. Task assignment with dynamic perception and constrained tasks in a Multi-Robot System. Submitted.
- A. Farinelli. Distributed Task Assignment for Real World Environments. *Ph.D Thesis. In preparation.*

References

[1] Robocup Rescue Web Site. http://robomec.cs.kobe-u.ac.jp/robocup-rescue.

[2] F. D'Agostino, A. Farinelli, G. Grisetti, L. Iocchi, and D. Nardi. Design and implementation of a Rescue-Agent Development Tool. In *Proceedings of the Workshop on Planning and real-time monitoring of rescue operations*, 2002.

[3] A. Farinelli, G. Grisetti, L. Iocchi, S. Lo Cascio, and D. Nardi. Using the robocup-rescue simulator in an italian earthquake scenario. In *Proc. of the 1st International Workshop on Synthetic Simulation and Robotics to Mitigate Earthquake Disaster*, 2003.

[4] D.L. Hall and J. Llinas (Eds.), editors. *Handbook of Multisensor Data Fusion.* CRC Press, 2001., 2001.

[5] J. Salerno. Information fusion: a high-level architecture overview. In *IEEE Int. Conf. Information. Fusion*, AnnaPolis, July 2002.

[6] S. Tadokoro and et al. The robocup rescue project: a multiagent approach to the disaster mitigation problem. *IEEE International Conference on Robotics and Automation (ICRA00), San Francisco*, 2000.

[7] L. Valet, G. Mauris, and P. Bolon. A statistical overview of recent literature in information fusion. *IEEE Aerospace and Electronics Systems Magazine*, 16(3):7 – 14, 2001.

Experiments with the RoboCup Rescue Simulator in a post Earthquake emergency Italian Scenario

Angelo Biagetti
Dipartimento dei Vigili del Fuoco,
del Soccordo Pubblico e della Difesa Civile
Ministero dell'Interno

Alessandro Farinelli, Luca Iocchi, Daniele Nardi, Fabio Patrizi
Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Salaria 113, 00198 ROMA, Italy
[farinelli|iocchi|nardi|patrizi]@dis.uniroma1.it

Abstract

In this paper we present the achievements of a research project, based on the RoboCup Rescue simulator, carried out in Italy in collaboration with the Italian Fire Department. The overall goal is to devise tools to allow for monitoring and supporting decisions which are needed in a real-time rescue operation in a post earthquake emergency scenario. As for the experiments, we have addressed the problem of task allocation by providing an experimental analysis of different strategies in different operative conditions. Moreover, we are currently extending the experimental analysis to feature-level information acquisition and integration. Finally, we discuss the limitation and potential for application of simulation based tools and, the RoboCup Rescue Simulation, in particular, based on the experience gained through the collaboration with the Italian Fire Department.

Keywords: *Resource coordination and information management in disaster scenarios, fire fighting and collapsed buildings (search and rescue, chemicals), distributed intelligence.*

1 Introduction

The RoboCup-Rescue Project started in 1999 with the goal of developing a comprehensive urban disaster simulator (see [8]). It aims at producing a software environment useful for both testing intervention strategies in a virtual world and supporting decisions in case of real disasters such as earthquakes or big fires.

The presentation addresses two issues: (1) extensions and supporting tools for the deployment of the RoboCup Rescue Simulator, (2) experiments and evaluation methodologies based on the simulations. With respect to the first issue we have developed some useful tools, that are shared with the RoboCup Rescue community in order to define new scenario and visualize agents' views as well as to design agents with information fusion, planning and coordination capabilities. In particular, we have modeled the Italian city of Foligno based on real data from the earthquake of the fall 1997 [4]. With regard to the second issue, we describe a methodology for evaluation of multi-agent system in this scenario that takes into account not only the efficiency of a system, but also its robustness when operation conditions and environment change, as well as other features, such as the ability to acquire a precise and coherent representation of the disaster scenario.

2 The RoboCup-Rescue Simulator: an application to the earthquake of Umbria and Marche

Below we sketch the overall structure of the simulator to provide some indications on the components that need to be developed in order to apply the simulator to a specific disaster scenario. For a detailed description of the simulator see [11].

The RoboCup-Rescue Simulator has a distributed architecture, formed by several modules, each of them being a separate process running in a workstation on a network. The following are the main components of the simulator: i) *Geographic Information System* - The GIS module holds the state of the simulated world. Before simulation begins, it is initialized by the user in order to reflect the state of the simulated area at a given time, then it is automatically updated at each simulation cycle by the kernel module. ii)*Kernel* - This module is connected to any other module. At each step it collects the action requests of the agents and the output of the simulators, merging them in a consistent way. Then the kernel updates the static objects in the GIS and sends the world update to all the connected modules. iii)*Simulators* - Fire-simulator, Collapse-simulator, Traffic-simulator, etc. are modules connected to the Kernel, each one simulating a particular disaster feature (fire, collapses, traffic, etc.). At the beginning of every simulation cycle, they receive from the kernel the state of the world, then they send back to the kernel the pool of GIS objects modified by the simulated feature (for example, a pool of burned or collapsed buildings, obstructed roads, etc.) iv)*Agents* - Agent modules are connected to the kernel and represent "intelligent" entities in the real world, such as civilians, police agents, fire agents, etc. They can do some basic actions, such as extinguishing a fire, freeing obstructions from roads, talking with other agents, etc. Agents can also represent non-human entities: for example they can simulate a police-office, a fire station, an ambulance-center, etc. v) *Viewers* - their task is to get the state of the world, communicating with the Kernel module, and graphically displaying it, allowing the user to easily follow the simulation progress.

In order to use the RoboCup-Rescue simulator in the context of the present project several issues must be taken into account.

The first issue we have addressed is the identification of the domain that should be based both on the availability of data and on suitability of the RoboCup Rescue simulators in modeling such an area. With respect to the simulation scenario a GIS editor for building the scenario for the RoboCup Rescue simulator has been implemented and data about the city of Foligno and the earthquake of Umbria and Marche (1997) have been acquired.

The second issue is the modeling of the agent system, which involves the communication infrastructure that allows the agents to exchange information during the simulation and the agent behaviour. An extension to the original RoboCup Rescue simulator that has been considered fundamental for our aims has been implemented in order to allow civilians to communicate with the coordinating agents of the Rescue forces. In addition, based on the models built in collaboration with the VVF, a basic set of agents have been designed; in particular, a prototype demonstrator, including firemen, police force and ambulances has been set-up to simulate the rescue scenario after earthquake. This has been achieved by providing a Cognitive Agent Development Kit (CADK, [2]), which allows for the specification of the behaviour of agents in a declarative fashion.

Finally, some issues concerning the viewers have been addressed. In particular, we have provided the system with the capability of visualising the specific views of individual agents to enable for evaluating the ability to reconstruct the situation after the disaster. Moreover, we have addressed the construction of a 3D model for providing a 3-D view of the simulation.

3 Evaluation Methodology

Evaluation of Multi-Agent Systems in the RoboCup Rescue domain is important not only within the RoboCup Rescue simulation competitions, but also for evaluating actual plans to be used during rescue emergencies.

Evaluation of MAS in the RoboCup Rescue domain is currently carried out within international contests [1, 7], by rating each competing rescue team with a score representing the result of its activity in a simulated scenario.

In the real world, however, events not always develop in a known and predictable way, since unexpected changes in the operative conditions and failures can occur at every time. The evaluation rule used in the RoboCup Rescue simulation competitions is applied under standard fixed conditions (only a particular known *configuration* is used) and thus it does not take into account the ability of a MAS to work under troublesome conditions and its ability to adapt to non-standard operative conditions (different *configurations*).

Below we summarize an evaluation method (see [3]), based on [6], which allows the analysis of a rescue team in a more realistic way, by analyzing the performance of a MAS in terms of efficiency under normal conditions, as well as in terms of reliability under changing working conditions.

3.1 Experimental settings

To acquire a measure of the reliability and the robustness of a MAS, a series of simulations are to be accomplished under changing operative conditions. These tests give a measure of the system adaptability to unexpected situations.

The operative parameters that can change during a simulation are different and we have grouped them in three classes, denoting the kind of agent capabilities that are affected: 1) Perception, 2) Action Execution, 3) Cooperation.

For each of these classes, we have selected a specific parameter that has been used for generating different operative *configurations*, and in this way we have characterized the following three tests, that have been performed under different configurations: i) **visibility test** that is performed by executing simulations to probe the activity of a system under different visibility conditions; ii) **disabled agents test** in which we have modeled those situations where agents may become suddenly not operational, making possible to analize the reactions of the MAS against new configurations of each force; iii) **noisy communication test** that introduces errors in the communication channel.

3.2 Performance measures

The performance of a rescue Multi-Agent System is measured in terms of efficiency and reliability. The efficiency is directly evaluated by the formula (also used in RoboCup-Rescue tournaments 2003):

$$V = (P + S/S_0) * \sqrt{B/B_0}$$

where P is the number of living agents, S is the remaining hit points (health level) of all agents, S_0 is the total hit points of all agents at initial, B is the area of houses that are not burnt and B_0 is the total area of houses at the beginning of the experiment; the higher the value of V for a rescue system, the better the results of the rescue operation.

The reliability, that is not considered in RoboCup Rescue Competitions, describes how much system efficiency is affected by the variation of *configurations*, and how much it depends on the values V assumed in the simulation sequence of a single test. To compute reliability we perform different simulations for the same *scenario*, changing a given set of parameters in the configuration worsening the abilities of the agents. Reliability is evaluated with the linear regression slope formula:

$$LRS = \frac{\sum_{i=0}^{N-1}(x_i - x_m) * (y_i - y_m)}{\sum_{i=0}^{N-1}(x_i - x_m)^2}$$

where (x_i, y_i) are the coordinates of a point in a Cartesian system, (x_m, y_m) the average values of these coordinates, N the number of points considered. To acquire the reliability value, this formula can be simplified with $x_i = i$ and $y_i = V(i)$, since each point of the graph represents the value of V obtained in the i-th *configuration*. Usually, the result is a negative value, since the effectiveness of the agents decreases with more difficult operative conditions. A small absolute value means a good degree of reliability of the system to adverse situations.

Obviously, this process can be applied to relevant measures other than the value of V. Depending upon the aspects of the system (i.e. task allocation or information fusion) to be analyzed, one can focus on different parameters, such as the number of messages exchanged among agents or the time during which agents are allocated, etc.

3.3 Performance comparison

Measures of efficiency and reliability of a single Multi-Agent System are of little significance if not compared with the results obtained from simulations of other rescue systems. Performance comparisons allow to establish the effectiveness of a new technique over the previous ones, or over the state-of-the-art.

Here we recall an example of the performance evaluation executed on four different rescue-systems (see [4] for a detailed description), created with the CADK tool (see also [2] for detatils) and differing for the information integration and resource allocation techniques employed.
To compare the performance of these four rescue systems, a set of experiments has been performed. The results are summarized by the diagrams in Figure 1.

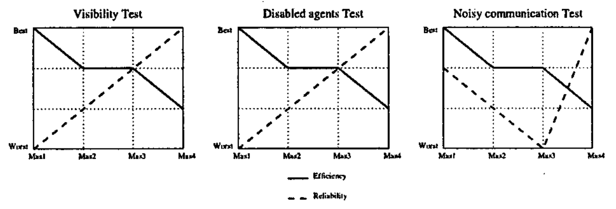

Figure 1: Performance comparison

In each test there is a rescue system which gets the highest value of efficiency and another one which obtains the highest score in reliability. Often in these tests the same rescue system is not the best for the two measures, since usually sophisticated techniques that improve efficiency turn out to be less robust to non-standard operative conditions, as the graphical representation of Figure 1 suggests. It is not obvious to identify which system has the best overall performance. In the visibility test, MAS1 is the best system in terms of efficiency, but it gets the worst rating about reliability. MAS2 and MAS3 have the same efficiency value, and are jointly ranked in the second place. MAS4, which is the worst system in terms of efficiency, is the best one with respect to reliability. The diagram shows also that MAS3 may be regarded as the best compromise between efficiency and reliability, since it is second in both of the two measures. In the noisy communication test, MAS1, which has the best efficiency value, is also a good system in terms of reliability, ranking in the second place; in this case, it seems to be superior to the other ones.

This example shows that the choice of the best system is hard to cast in absolute terms. Depending on the application, the system which offers the best score with respect to efficiency, reliability, or a (weighted) combination of the two may be selected. Indeed, the choice of a measure to select the best solution is a non-trivial task.

4 Task Assignment

In this section we show the performance analysis of a team of fire brigades with different task assignment strategies. The problem of task assignment in our simulation domain consists in the selection of agents to solve the three typical issues of the domain: (i) save civilians, (ii) extinguish fires, (iii) clean roads.

In order to study the task assignment problem in our target domain, we have focussed our attention on the issue of extinguishing fires by the fire force. The scenario is constituted by a city map in which fire agents and intervention places can be allocated on different regions of the map following different distribution. As we will see, different maps are suitable for different task allocation strategies.

In the first allocation strategy (named *Greedy-local perception* or GLP) no communication among agents is held and each agent goes to the nearest fire it can perceive. Notice that, given the strict communication limitation enforced in the RoboCup rescue competitions, several top performing team use similar approaches.

The second strategy is based upon a distributed mechanism, that regulates access to roles through the use of tokens. Each agent, when a new fire is perceived, creates a token referring to that role and decides whether to execute that role or pass the token to one of the team mates (using a round robin policy); each fire perceived through vision is registered in a list of known fires, hence only new fires are considered. This is useful to avoid too many agents going toward the same fire. Among the fires an agents knows, only the one at which the agent is most capable at is taken, the others are sent out. The capability of each agent is computed considering the distance from the fire. We referred to this strategy as *Token Passing* or TP. In both the strategies it could happen that two agents are competing for the same fire. However for the RoboCup rescue scenario this is not a problem, since agents help each other in fighting the same fire. Problems can arise if too many agents fight the same fire, because agents can block each other in narrow passages.

We have considered two scenarios: each one includes 18 fires and 10 agents, but in the first one fires and fire brigades are spread almost uniformly all across the city map (see Fig. 2), while in the second one fires are concentrated in one region of the map and fire brigades start from two regions of the map (see Fig. 3).

Figure 2: Scenario 1– Uniformly distributed agents and fires

Figure 3: Scenario 2– Fires and agents concentrated in few regions of the map

From the results of each simulation, we have extracted the percentage of saved buildings, the extinguish time (time steps needed to put out all the fires), and finally the messages exchanged among agents.

	TP Sc. 1	GLP Sc. 1	TP Sc. 2	GLP Sc. 2
safe build.%	95.97	96.02	95.63	93.7
Ext. time	40.07	40	32.35	59
Msgs	359	0	811.35	0

Table 1: Task assignment results

Comparing the GLP allocation and the TP one we see that while the performance are similar in terms of saved buildings percentage and extinguish time, the GLP allocation does not need any message exchange, therefore it can be considered a better choice to the TP for such situations. The main reason to explain this result is that communication in the RoboCup Rescue simulator requires a considerably high amount of time: in fact it is possible to send at most one message each time step, which resemble approximately one message per minute in the real world, therefore when fires and fire brigades are nicely spread all over the city map, GLP allocation results actually as a very good strategy.

In the second scenario, where fires are concentrated in a particular region of the map, and fire brigades are concentrated in two other regions, communication is obviously more important and actually the TP approach consistently outperforms the GLP, while keeping a very low communication overhead. Notice that the scenario reported in figure 3, is much more likely to happen in a real rescue situation because, generally fire brigades starts their missions from specific fire centers while fires are mostly concentrated in few regions of the city map.

Summarizing, the experiments performed clearly show that the evaluation of a specific allocation strategy in the rescue domain should consider not only the efficiency and reliability of the strategy, but also particular features of the environment in which the allocation should work. As an example, in our particular reference scenario, the distribution of fires and fire brigades across the city map, is a fundamental characteristic to consider in the choice of the allocation strategy. In particular, if fires and fire brigades are spread on the city map in a uniform fashion, with a simple policy like the GLP it is possible to obtain good performance while avoiding messages exchange. However, for different distributions such as the one represented in Figure 3 more complex strategies are needed.

5 Information Fusion

In order to reach good performance in post-earthquake disaster situation agents need to exhibit both planning and cooperation capabilities, since the abilities of a single individual agent are often not enough for fighting an expanding disaster. However, another aspect to be considered, while developing a team of rescue agents is the need of integrating partial and noisy information coming from the agents, in order to assess a global situation, on which to perform the resource allocation. Indeed, the system performance is deeply influenced by the knowledge of the scenario. The interactions between information acquisition and integration and the process of decision making is clearly addressed in various models that have been proposed for information fusion [5, 10]. Specifically, at the stage of situation assessment not only is relevant the knowledge about the scenario, but also the effects and performance of actions taken in the scenario. It is worth noticing that, since agents may take specific actions to gather information (differently from passive sensors), the scenario is significantly different from that of a distributed sensor agent network [9, 13].

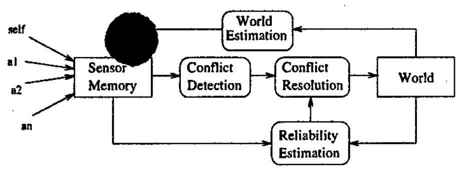

Figure 4: Information Integrator Module

The goal is to reconstruct the new state of the world starting from the previous states and the incoming information.

There are different levels at which the integration can be performed [12], depending on the properties of the data being integrated:
-*sensor-level data*: numeric values directly extracted from sensors;
-*feature-level data*: aggregation of numeric data representing specific features;
-*symbol-level data*: high level items related to agent knowledge.
In our approach the information integration is performed at symbol and feature level, in terms of properties of the world objects, since the RoboCup Rescue domain, in which we tested the system, is well suited for this level of information fusion.

In Figure 4 the functional structure of the Information Integration module of our CADK is sketched. The information provided by the sources, that can be on-board agent sensors, external sources (i.e. messages about the world situation coming from other agents) as well as agent expectations about the current situation are collected in a data structure (Sensor Memory). Then the possible conflicts arising from the comparison of the collected reports are detected and solved, by taking into account the reliability of the sources. Also the reliability is evaluated, by taking into account the evolution of the reconstructed situation. By choosing the level of information being integrated, as well as the conflict resolution policies, it is possible to implement a wide set of information integration strategies.

Information Fusion technology has the primary goal of improving the cognitive skills of a system, by integrating and correlating elements of different kinds. Roughly speaking, this amounts to obtaining from the raw data directly gathered from sensors compact, coherent and understandable information.

According to this view, an evaluation methodology is required, to measure the effectiveness of the fusion process, which addresses at least the following aspects: (i) quality of information, measured with respect to reality; (ii) robustness to noise and communication errors; (iii)computational effort.

To evaluate an information fusion system at data and object level, several techniques, based on the variation of some parameters and on the comparison of the obtained results [5] are used. However, the problem of evaluating the performance of a system for situation assessment and higher fusion level has received less attention, while we believe it is a very relevant issue in this framework.

6 Conclusions and future work

We have presented a methodology for evaluating the performance of a Multi-Agent Systems in the framework of a simulation environment for post earthquake scenarios. The aim of the work is to address the problem of devising methods and evaluating the performance of specific aspects of MAS. In particular, we reported some results on task assignment and presented a proposal for extending the approach to high level information fusion. Our idea is to attempt first to address the two aspects separately, in order to refine the experimental methodology and understand the specific features of each of them. However, in practice they coexist and deeply influence each other, since the management and coordination of a Multi-Agent System requires good knowledge of the situation where the operations take place. Therefore, after completion of the first phase, we plan to combine them and extend the approach to deal with both of them at the same time. In this extended framework the critical question of action vs perception, will be addressed in the context of a simulated MAS.

In addition, we are developing this work in collaboration with the Italian Fire Department, which provides us with technical expertise on rescue operations that we take into account in the design of both the simulation system and the MAS. In this respect, we are planning to use the tools developed in this project to build a scenario embodying features that are substantially different from those currently available, and to compare implemented strategies with those actually deployed by the forces of Italian Fire Department to make our working hypotheses in the simulation more accurate with respect to actual emergency scenarios.

References

[1] Robocup Rescue Web Site. http://robomec.cs.kobe-u.ac.jp/robocup-rescue.

[2] F. D'Agostini, A. Farinelli, G. Grisetti, L. Iocchi, and D. Nardi. Design and implementation of a Rescue-Agent Development Tool. In *Proceedings of the Workshop on Planning and real-time monitoring of rescue operations*, 2002.

[3] A. Farinelli, G. Grisetti, L. Iocchi, S. Lo Cascio, and D. Nardi. Design and evaluation of multi agent systems for rescue operations. In *Proc. of International Conference on Intelligent Robots and Systems (IROS'03)*, 2003.

[4] A. Farinelli, G. Grisetti, L. Iocchi, S. Lo Cascio, and D. Nardi. Using the robocup-rescue simulator in an italian earthquake scenario. In *Proc. of the 1st International Workshop on Synthetic Simulation and Robotics to Mitigate Earthquake Disaster*, 2003.

[5] D.L. Hall and J. Llinas (Eds.), editors. *Handbook of Multisensor Data Fusion*. CRC Press, 2001., 2001.

[6] G. A. Kaminka, I. Frank, K. Arai, and K. Tanaka-Ishii. Performance competitions as research infrastructure: Large scale comparative studies of multi-agent teams. *Journal of Autonomous Agents and Multi-Agent Systems*, 2002. To appear.

[7] H. Kitano and et al. Search and rescue in large scale disasters as a domain for autonomous agents resarch. *IEEE International Conference on System, man and Cybernetics (SMC99), Tokyo*, 1999.

[8] Hiroaki Kitano and Satoshi Tadokoro. RoboCup-Rescue: A grand challenge for multi-agent and intelligent systems. *AI Magazine*, 22(1):39–52, 2001.

[9] A. Knoll and J. Meinkoehn. Data fusion using large multi-agent networks: an analysis of network structure and performance. *Proceedings of International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI '94).*, pages 113 – 120, 1994.

[10] J. Salerno. Information fusion: a high-level architecture overview. In *IEEE Int. Conf. Information. Fusion*, AnnaPolis, July 2002.

[11] Robocup Rescue Web Site. http://robomec.cs.kobe-u.ac.jp/robocup-rescue.

[12] L. Valet, G. Mauris, and P. Bolon. A statistical overview of recent literature in information fusion. *IEEE Aerospace and Electronics Systems Magazine*, 16(3):7 – 14, 2001.

[13] Zili Zhang and Chengqi Zhang. Result fusion in multi-agent systems based on owa operator. *Proceedings of Australasian Computer Science Conference, (ACSC2000)*, pages 234 – 240, 1999.

Situation Assessment and Information Fusion: an experimental framework for performance evaluation

Alessandro Farinelli, Luca Iocchi, Sergio Lo Cascio, Daniele Nardi
Dipartimento di Informatica e Sistemistica
Via Salaria 113
00199, Rome, Italy
[farinelli|iocchi|locascio|nardi]@dis.uniroma1.it

Abstract – *The field of Information Fusion is recently focusing on issues arising in the high level steps of the process. One approach that is gaining attention in this context is based on the techniques developed in the field of Multi Agent Systems. In this paper we present a methodology and some preliminary results for evaluating the performance of a Multi Agent System developed to simulate the intervention in a post earthquake emergency scenario, derived from the RoboCup Rescue simulator. Specifically, we propose an evaluation methodology suitable both for task assignment and feature-level information fusion. As for Task Assignment we present an experimental analysis of different strategies in different operative conditions; while for feature-level information fusion we specify our evaluation methodology and indicates lines to follow in the experimental analysis to be performed.*

Keywords: Information fusion, performance evaluation, task assignment

1 Introduction

The problem of integrating information coming from different sources at different times has been addressed in many domains, ranging from information agents for the Web, database integration, to sensor fusion for mobile robotics, etc. Specific techniques have been developed for these applications (see for example [1, 2]) and the research on information fusion has produced useful reference models [3]. However, the goal of effectively monitor the situation in highly dynamic scenarios, such as, for example, those arising in post-earthquake or other catastrophic events, where a large variety of sensor devices and information sources are available, is still far from being achieved.

The development of suitable frameworks and methodologies to evaluate the performance of complex systems acting in such real-world scenarios, is a fundamental and challenging research issue. In fact, the inherent complexity of the tasks that an intelligent system is expected to perform make it very difficult to apply standard techniques for testing, validation and measure of performances [4]. In this context the availability of standard reference scenarios is receiving increasing attention, and, more specifically, synthetic simulation scenarios are considered for the evaluation of Multi-Agent approaches [5].

Multi-Agent Systems (MAS) [6] are a very powerful and flexible tool to build such scenarios, and they are becoming increasingly popular for such kind of applications. Specifically the RoboCup Rescue simulator [7, 8, 9] is a powerful and flexible environment based on a MAS approach to develop simulation scenarios. Tools with similar features are also being adopted in [10]

In particular, in previous projects [11, 12] we have adapted the RoboCup Rescue simulator for developing a reference scenario to address the study of the techniques for information fusion, situation assessment and agent coordination, in a post-earthquake emergency situation. The scenario includes teams composed by human operators, information agents, and (mobile) sensors, that gather information in a specific geographic area and event, in order to reconstruct a global representation of this environment, that is critical for an effective operation. The basic paradigm underlying the proposed approach is Multi-Agent System [13], that allows for both the modeling and the implementation of the information fusion process [3, 11]. More specifically, an experimental framework can be realized to simulate different scenarios that are of practical interest for the implementation, experimentation and evaluation of information fusion techniques.

The availability of a simulator makes it possible to devise methodologies for the systematic evaluation and comparison of information fusion techniques and strategies, which is often difficult to achieve. For example, it is possible to measure through simulations the time required to obtain a specific information or a complete picture of the given scenario, depending on several factors: the information fusion techniques, the exploration strategy adopted by the agents, the reliability of communication, the noise of the information coming from external sources, etc. Moreover, one can use the simulator to identify problems and bottlenecks that are specific to the domain under consideration.

In this paper we present the methodology we developed to evaluate performance of a Multi-Agent System in rescue domains. Our main idea is that such evaluation should not be the measure of a specific objective function (as currently done in the RoboCup Rescue competitions), but it should rather consider several parameters under different and varying operation conditions [14]. In fact, besides the efficiency of a Multi-Agent System, in a rescue domain it is also important to measure the robustness when operation conditions change, as well as the variance of the results tervention strategies. To this end we follow the proposed methodology in the experimental analysis of a team of fire brigades with different task assignment strategies. Moreover, we specify our evaluation methodology for the comparison of different fusion techniques, indicating the lines to follow in the future experimental analysis.

2 RoboCup-Rescue Simulator

In order to make the paper self contained below we sketch the basics of the RoboCup Rescue simulator [9]. The RoboCup Rescue simulator is a distributed framework, formed by several modules, each of them being a separate process running in a workstation on a network. The main components of the simulator are:

1) a *Geographic Information System (GIS)*, holding the state of the simulated world: before the simulation begins, it is initialized by the user in order to reflect the state of the simulated area at a given time, then it is automatically updated at each simulation cycle by the kernel module;

2) a *Kernel*, that collects the output of the simulators and the action requests from the agents, merging them in a consistent way and updating the information in the GIS; it is connected to all the other modules and sends the world updates to all of them;

3) *Simulators*: Fire-simulator, Collapse-simulator, Traffic-simulator, etc. that are modules connected to the *Kernel*, each one simulating a particular disaster feature (fire, collapses, traffic, etc.): at the beginning of every simulation cycle, they receive from the kernel the state of the world and send back to the *Kernel* the pool of GIS objects modified by the simulated feature;

4) *Agents*: agent modules represent "intelligent" entities in the real world (such as fire agents, police agents, medical staff, civilians, etc.), that can perform basic actions (such as extinguishing fires, freeing obstructions from roads, save victims, talking with other agents, etc.);

5) *Viewers*, that allows for displaying the state of the world and for following the simulation progress.

The simulation is realized in a discrete time system framework, where the state evolution in a certain instant is calculated on the basis of the state in the previous instant, and of the inputs (represented by the agents' action). Thus, starting from a given *initial situation*, that is generally provided by the user possibly through a graphical interface (see [15]), the simulation evolves according to both the process evolution computed by the internal *Simulators* and the actions performed by the *Agents* in the world.

For the development of the agents used in the simulations we devised a tool, the Cognitive Agent Development Kit (CADK) [15, 12], to easily create Multi-Agent teams with different capabilities and peculiarities; such a tool is a fundamental feature because allows the developers to easily change the agents capabilities in order to experiment several different coordination and information fusion techniques. The agents developed with this tool have the following characteristics: (i) they can act autonomously in the environment by selecting the actions to be performed according to the information acquired from the environment; achieve a common goal; (iii) they can exchange information about the environment in order to reconstruct a global situation by using appropriate information fusion techniques.

In this paper we use the term *scenario* to denote a specific set up of the environment comprising the map used in the simulation and the initial situation. During the simulation, the evolution of the scenario is characterized by a sequence of situations that corresponds to the time steps of the simulation. The map we are using is a map of an Italian town (Foligno), with different initial distribution of fires and agents. Another important issue that must be taken into account when performing experiments and evaluation of performance with the RoboCup-Rescue simulator is the parameters that are used for describing the capabilities of the agents (for example, the range of vision, the ability of communicating, etc.). For distinguishing different settings, we will use the term *configuration* to denote a particular setting of the parameters that determine the abilities of the agents.

3 Evaluation Methodology

Evaluation of Multi-Agent Systems in the RoboCup Rescue domain is important not only within the RoboCup Rescue simulation competitions, but also for evaluating actual plans to be used during rescue emergencies.

Evaluation of MAS in the RoboCup Rescue domain is currently carried out within international contests [7, 8], by rating each competing rescue team with a score representing the result of its activity in a simulated scenario. The one with the highest measured score is the contest winner.

In the real world, however, events not always develop in a known and predictable way, since unexpected changes in the operative conditions and failures can occur at every time. The evaluation rule used in the RoboCup Rescue simulation competitions is applied under standard fixed conditions (only a particular known *configuration* is used) and thus it does not take into account the ability of a MAS to work under troublesome conditions and its ability to adapt to non-standard operative conditions (different *configurations*).

The evaluation method proposed here is based on [5] and allows the analysis of a rescue team in a more realistic way, by analyzing the performance of a MAS in terms of efficiency under normal conditions, as well as in terms of reliability under changing working conditions.

3.1 Experimental settings

To acquire a measure of the reliability and the robustness of a MAS, a series of simulations are to be accomplished under changing operative conditions. These tests give a measure of the system adaptability to unexpected situations.

The operative parameters that can change during a simulation are different and we have grouped them in three classes, denoting the kind of agent capabilities that are affected: 1) Perception, 2) Action Execution, 3) Cooperation.

For each of these classes, we have selected a specific parameter that has been used for generating different operative *configurations*, and in this way we have characterized different configurations.

The visibility test. In outdoor environments, visibility conditions are extremely variable. Rescue operation can be needed every time of the day, also in the night. Thus, it is necessary to probe the activity of a system also in these situations. In this test the changing configurations depend on the perception range of each agent: the visibility test is performed by executing different simulations, each with decreasing perception radius, modeling activities under different visibility conditions (i.e. twilight, night time, fog).

The disabled agents test. In a real emergency situation, it can happen that an agent suddenly become not operational for some reason (for example a mechanical failure of its vehicle or its equipment); this test analyzes the reactions of the MAS against new configurations in which some of the operative agents are disabled. The disabled agents test thus considers varying numbers of disabled agents on each force.

The noisy communication test. Agent cooperation in the rescue domain is mainly attained by radio communications among coordination centers and between a coordination center and the operative agents. In real conditions communication transmissions are not free from network failures, or human misunderstandings. This test verifies the robustness of an analyzed MAS by introducing errors in the communication channel, thus preventing messages to reach their destination. The errors in the communication channel are simulated by the random loss of an increasing percentage of messages.

3.2 Performance measures

The performance of a rescue Multi-Agent System is measured in terms of efficiency and reliability. The **efficiency** is directly evaluated by the formula (also used in RoboCup-Rescue tournaments 2003):

$$V = (P + S/S_0) * \sqrt{B/B_0}$$

where P is the number of living agents, S is the remaining hit points (health level) of all agents, S_0 is the total hit points of all agents at initial, B is the area of houses that are not burnt and B_0 is the total area of houses at the beginning of the experiment; the higher the value of V for a rescue system, the better the results of the rescue operation.

The **reliability** describes how much system efficiency is affected by the variation of *configurations*, and how much it depends on the values V assumed in the simulation sequence of a single test. To compute reliability we perform different simulations for the same *scenario*, changing a given set of parameters in the configuration worsening the abilities of the agents. Reliability is evaluated with the linear regression slope formula:

$$LRS = \frac{\sum_{i=0}^{N-1}(x_i - x_m) * (y_i - y_m)}{\sum_{i=0}^{N-1}(x_i - x_m)^2}$$

system($(\overset{i,\, y_i}{x_m, y_m})$ t[illegible]verage values of these coordinates, N the number of p[illegible]s considered. To acquire the reliability value, this formula can be simplified with $x_i = i$ and $y_i = V(i)$, since each point of the graph represents the value of V obtained in the i-th *configuration*. Usually, the result is a negative value, since the effectiveness of the agents decreases with more difficult operative conditions. A small absolute value means a good degree of reliability of the system to adverse situations.

Obviously, this process can be applied to relevant measures other than the value of V. Depending upon the aspects of the system (i.e. task allocation or information fusion) to be analyzed, one can focus on different parameters, such as the number of messages exchanged among agents or the time during which agents are allocated, etc.

3.3 Performance comparison

Measures of efficiency and reliability of a single Multi-Agent System are of little significance if not compared with the results obtained from simulations of other rescue systems. Performance comparisons allow to establish the effectiveness of a new technique over the previous ones, or over the state-of-the-art.

In this section, it is shown an example of the performance evaluation executed on four different rescue-systems, created with the CADK tool. The analyzed MAS differ for the information integration and resource allocation techniques employed, as shown in the following table:

Allocation\fusion	no fusion	simple
Static	MAS 1	MAS 2
Dynamic	MAS 3	MAS 4

To compare the performance of the these four rescue systems, we perform a set of experiments, whose results are shown in the left tables of Figure 1.

In each test there is a rescue system which gets the highest value of efficiency and another one which obtains the highest score in reliability. Often in these tests the same rescue system is not the best for the two measures, since usually sophisticated techniques that improve efficiency turn out to be less robust to non-standard operative conditions. To provide better intuition to the previous results, a graphical representation is given in the right side diagrams of Figure 1. It is not obvious to identify which system has the best overall performance. In the visibility test, MAS1 is the best system in terms of efficiency, but it gets the worst rating about reliability. MAS2 and MAS3 have the same efficiency value, and are jointly ranked in the second place. MAS4, which is the worst system in terms of efficiency, is the best one with respect to reliability. The diagram shows also that MAS3 may be regarded as the best compromise between efficiency and reliability, since it is second in both of the two measures. In the noisy communication test, MAS1, which has the best efficiency value, is also a good system in terms of reliability, ranking in the second place; in this case, it seems to be superior to the other ones.

MAS1	23.3	–2.65
MAS2	18.8	–2.06
MAS3	18.8	–1.99
MAS4	11.6	–1.67

Disabled agents	Effic.	Rel.
MAS1	23.3	–6.18
MAS2	18.8	–5.32
MAS3	18.8	–5.09
MAS4	11.6	–4.88

Noisy communic.	Effic.	Rel.
MAS1	23.3	–3.18
MAS2	18.8	–3.95
MAS3	18.8	–4.04
MAS4	11.6	–2.79

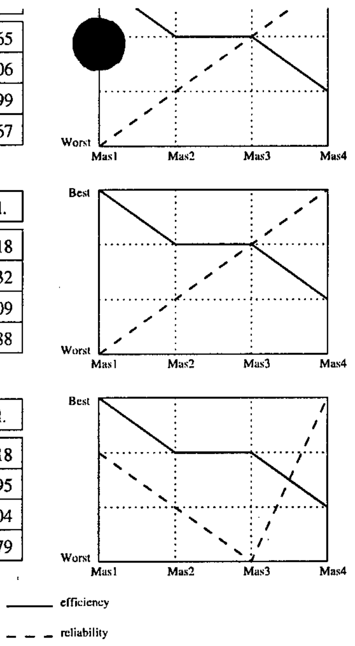

Fig. 1: Performance comparison

This example shows that the choice of the best system is hard to cast in absolute terms. Depending on the application, the system which offers the best score with respect to efficiency, reliability, or a (weighted) combination of the two may be selected. Indeed, the choice of a measure to select the best solution is a non-trivial task.

3.4 Variance analysis

Event evolution in a distributed simulation depends not only on the initial setup and on the behavior of the MAS, but also on the system where the simulation is running. When a simulation can be run along a distributed system (as in the case of the RoboCup-Rescue simulator), there are two main causes of unpredictable event development: (i) network reliability and (ii) processor speed. The first issue is crucial as the simulation system is composed by several modules (seven for the simulation and six for the agents), each of which communicates with the kernel module massively over the network at each simulation step. The protocol used by the modules to communicate is derived from UDP, thus causing some messages not to arrive at destination (ever or just in time). Processor speed is also crucial for simulation evolution since each agent has a limited time to plan an action; if the agent has not issued an action within this time it loses the turn. Hence, more sophisticated action planning techniques may cause the loss of action cycles, if not supported by adequate CPU power of the host machine.

For these issues the simulator system itself does not allow to replicate identical experiments. In this context, a statistical analysis of the results' variance of a RoboCup-Rescue simulation can be useful for the following two reasons: (i) it allows the analysis of rescue simulation system by giving a good estimation of the reliability of a particular hardware and software simulation environment, since low variance indicates few losses in network messages and agents' team's behavior: it is advisable to have low variance, since it means high robustness to unexpected event evolution.

As a sample analysis of RoboCup-Rescue variance we have executed 20 runs on the same scenario using the same initial conditions and the same rescue team; the results are summarized in the Table 1.

	avg.	var.	perc. var
Evaluation	50.59	2.8	5.5
Saved civilians	23.75	3.5	14.7
Extinguished fires	16.45	6.2	37.7
Cleaned roads	313.6	28.0	8.9

Table 1: Measured parameters over 20 runs

In this case we can see that the analyzed rescue team is not so stable for the high variance of saved civilians, extinguished fires and cleaned roads; the worst parameter is the number of extinguished fires, which indicates poor performance of the fire force.

The results show that variance analysis is crucial when working with the RoboCup simulator, and should be always considered in order to show significant stable results.

4 Task Assignment

In this section we show the performance analysis of a team of fire brigades with different task assignment strategies. The problem of task assignment in our simulation domain consists in the selection of agents to solve the three typical issues of the domain: (i) save civilians, (ii) extinguish fires, (iii) clean roads.

In order to study the task assignment problem in our target domain, we have focussed our attention on the issue of extinguishing fires by the fire force. The scenario is constituted by a city map in which fire agents and intervention places can be allocated on different regions of the map following different distribution. As we will see, different maps are suitable for different task allocation strategies.

In the first allocation strategy (named *Greedy-local perception* or GLP) no communication among agents is held and each agent goes to the nearest fire it can perceive.

The second strategy is based upon a distributed mechanism, by which each agent autonomously evaluates, among its known fires, the best one to extinguish. Not selected fires are sent toward other agents. Each agent, when a new fire is perceived, decides whether to fight that fire or communicate the presence of the fire to one of the team mates (using a round robin policy), each fire perceived through vision is registered in a list of known fires, hence only new fires are considered. This is useful to avoid too many agents going toward the same fire. Among the fires an agents knows, only the one at which the agent is most capable at is taken, the others are sent out. The capability of each agent is computed considering the distance from the fire. We referred to this strategy as *Token Passing* or TP. In both the strategies it could happen that two agents are competing for the same fire. However for the RoboCup rescue scenario this is same fire. Problems can arise if [illegible] many agents fight the same fire, because agents can b[illegible] each other in narrow passages.

In order to overcome problems due to possible lost messages we use a *time_to_live* for the fire perceived and consider valid old perceptions only up to some time steps. Each time a fire is perceived we check if it is already present in the known fire list, if not a new task is associated to this fire a to the related time stamp. If the fire is present but the time stamp is older than a given threshold (7 time steps in the simulations), we re-consider the fire for execution and update its time stamp. Another problem we have experienced are roads being blocked by other agents; this frequently causes a relevant delay in the mission execution and, therefore low performance. We decided to detect blocking situation and integrate this information in each agent's capability computation.

We have considered two scenarios: each one includes 18 fires and 10 agents, but in the first one fires and fire brigades are spread almost uniformly all across the city map (see Fig. 2), while in the second one fires are concentrated in one region of the map and fire brigades start from two regions of the map (see Fig. 3).

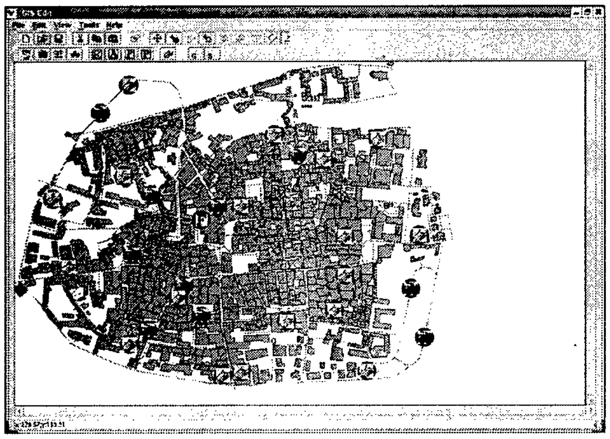
Fig. 2: Scenario 1– Uniformly distributed agents and fires

From the results of each simulation, we have extracted the percentage of saved buildings, the extinguish time (time steps needed to put out all the fires), the messages exchanged among agents, the average task changes per agent, the average number of tasks refused by all agents (each task has a visited agents list) and, finally, the percentage of time the agents have been used till the extinguish time.

We have tried two kinds of simulations: a centralized one, were all the simulators agents and the kernel run on the same machine, and a distributed one were the computation is done on several machines. The algorithm used in both settings is exactly the same, however, computation due to messages exchange have a relevant influence on the simulation, resulting in higher variance of the results, hence we decided to use the distributed setting in the experiments.

Comparing the GLP allocation and the TP one we see that while the performance are similar in terms of saved

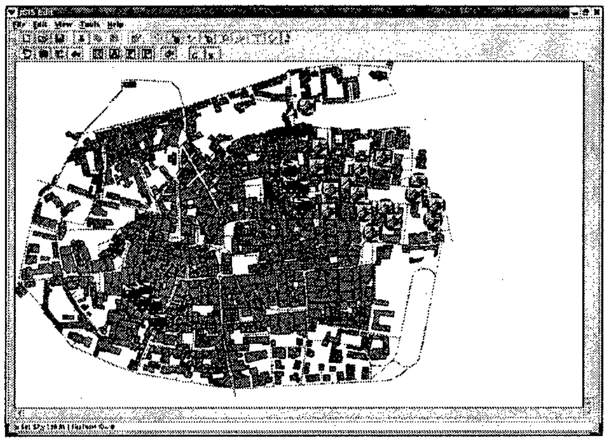
Fig. 3: Scenario 2– Fires and agents concentrated in few regions of the map

	avg.	var.	perc. var
Perc. of saved buildings	95.97	~0	~0
Ext. time	40.07	0.37	0.94
Messages to fire center	0	0	undef
Messages among agents	359	6.04	1.7
Task changes per agent	1.42	0.13	9.3
Total tasks refused	4	0	0
Perc of active time	66.08	0.44	0.67

Table 2: Token Passing - Scenario 1

	avg.	var.	perc. var
Perc. of saved buildings	96.02	~0	~0
Ext. time	40	~0	~0
Messages to fire center	0	0	undef
Messages among agents	0	0	undef
Task changes per agent	0	0	undef
Total tasks refused	0	0	undef
Perc of active time	41.25	~0	~0

Table 3: Greedy Local Perception - Scenario 1

tion does not need any message [illegible]ange, therefore it can be considered a better choice to [illegible]P for such situations. The main reason to explain this result is that communication in the RoboCup Rescue simulator requires a considerably high amount of time: in fact it is possible to send at most one message each time step, which resemble approximately one message per minute in the real world, therefore when fires and fire brigades are nicely spread all over the city map, GLP allocation results actually as a very good strategy.

In the second scenario, where fires are concentrated in a particular region of the map, and fire brigades are concentrated in two other regions, communication is obviously more important and actually the TP approach consistently outperforms the GLP, while keeping a very low communication overhead. Notice that the scenario reported in figure 3, is much more likely to happen in a real rescue situation because, generally fire brigades starts their missions from specific fire centers while fires are mostly concentrated in few regions of the city map.

	avg.	var.	perc. var
Perc. of saved buildings	95.63	0.21	0.22
Ext. time	32.35	1.31	4.1
Messages to fire center	0	0	undef
Messages among agents	811.35	63.73	7.9
Task changes per agent	1.95	0.18	9.4
Total tasks refused	4.71	1.04	22
Perc of active time	63.42	0.05	0.08

Table 4: Token Passing - Scenario 2

	avg.	var.	perc. var
Perc. of saved buildings	93.7	~0	~0
Ext. time	59	~0	~0
Messages to fire center	0	0	undef
Messages among agents	0	0	undef
Task changes per agent	0	0	undef
Total tasks refused	0	0	undef
Perc of active time	43.05	~0	~0

Table 5: Greedy Local Perception - Scenario 2

Summarizing, the experiments performed clearly show that the evaluation of a specific allocation strategy in the rescue domain should consider not only the efficiency and reliability of the strategy, but also particular features of the environment in which the allocation should work. As an example, in our particular reference scenario, the distribution of fires and fire brigades across the city map, is a fundamental characteristic to consider in the choice of the allocation strategy. In particular, if fires and fire brigades are spread on the city map in a uniform fashion, with a simple policy like the GLP it is possible to obtain good performance while avoiding messages exchange. However, for different distributions such as the one represented in Figure 3 more complex strategies are needed.

As previously desc[illegible] in order to reach good performance in the post-earthquake disaster situation agents need to exhibit both planning and cooperation capabilities, since the abilities of a single individual agent are often not enough for fighting an expanding disaster. However, another aspect to be considered, while developing a team of rescue agents is the need of integrating partial and noisy information coming from the agents, in order to assess a global situation, on which to perform the resource allocation. Indeed, the system performance is deeply influenced by the knowledge of the scenario. The interactions between information acquisition and integration and the process of decision making is clearly addressed in various models that have been proposed for information fusion [1, 3]. Specifically, at the stage of situation assessment not only is relevant the knowledge about the scenario, but also the effects and performance of actions taken in the scenario. In this section, we specifically focus on the problem of information acquisition and integration, given a set of communicating *information sources*, while the problem of simultaneous information acquisition and task allocation will be briefly discussed in the next section.

In general, an agent can be equipped with a large variety of sensors and/or sensing capabilities, each one providing a different type of input. In our framework we have several communicating agents of various types with possibly different sensing capabilities, and it is also possible to simulate sensor agents. It is worth noticing that, since agents may take specific actions to gather information (differently from passive sensors), the scenario is significantly different from that of a distributed sensor agent network [16, 17].

The goal is to reconstruct the new state of the world starting from the previous states and the incoming information.

There are different levels at which the integration can be performed [18], depending on the properties of the data being integrated:
-*sensor-level data*: numeric values directly extracted from sensors;
-*feature-level data*: aggregation of numeric data representing specific features;
-*symbol-level data*: high level items related to agent knowledge.
In our approach the information integration is performed at symbol and feature level, in terms of properties of the world objects, since the RoboCup Rescue domain, in which we tested the system, is well suited for this level of information fusion.

In Figure 4 the functional structure of the Information Integration module of our CADK is sketched. The information provided by the sources, that can be on-board agent sensors, external sources (i.e. messages about the world situation coming from other agents) as well as agent expectations about the current situation are collected in a data structure (Sensor Memory). Then the possible conflicts arising from the comparison of the collected reports are detected and solved, by taking into account the reliability of the sources. Also the reliability is evaluated, by taking into account the evolution of the reconstructed situation. By

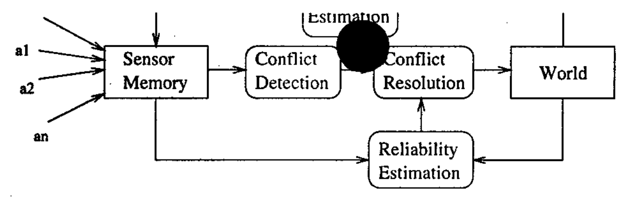

Fig. 4: Information Integrator Module

choosing the level of information being integrated, as well as the conflict resolution policies, it is possible to implement a wide set of information integration strategies.

5.1 Evaluation of Information Fusion

Information Fusion technology has the primary goal of improving the cognitive skills of a system, by integrating and correlating elements of different kinds. Roughly speaking, this amounts to obtaining from the raw data directly gathered from sensors compact, coherent and understandable information.

According to this view, an evaluation methodology is required, to measure the effectiveness of the fusion process, which addresses at least the following aspects: (i) quality of information, measured with respect to reality; (ii) robustness to noise and communication errors; (iii)computational effort. In the following, we further discuss each of them pointing out the parameters that we plan to use in order to asses them through simulations.

Information quality. Information fusion process involves data acquisition from several sources and a state reconstruction which is as close as possible to reality; thus, *amount*, *precision* and *reliability* of information are relevant parameters. In order to measure them it is useful to compare the reconstructed state and the real state. While amount and precision of information can be directly measured, reliability is more involved, and we determine it by considering false alarms and failures. A *false alarm* comes up when a property estimation gives a critical value, while the real measure is not. On the other hand, we have a *failure* when a real property has a critical value not detected by the system.

Robustness. In some practical fields robustness means working with continuity, even when a source node or a fusion node are temporarily or permanently down. In a more general framework, the fusion system should be able to work even in case of noise or communication errors over networks. Moreover, robustness is an important feature in situations when conflicts arise (i.e. when information sources provide contradicting data).

Computational effort. Computation capabilities are usually limited, therefore it is compulsory to ensure that all fusion nodes are able to process data given the application constraints. For this reason it is necessary to analyze the relation between input and computation effort, with regard to the response time to obtain the output.

and object level, se[illegible] techniques, based on the variation of some parameters [illegible] on the comparison of the obtained results [1] are used. However, the problem of evaluating the performance of a system for situation assessment and higher fusion level has received less attention. We are planning to use our simulation framework, in order to make some steps in this direction. Specifically, based on the parameters above described, we want to examine the behavior of the information fusion system, by adopting the same methodology described in the previous section for task assignment. To this end we evaluate the global system load and the performance, by first providing ideal conditions: no errors in sensors, and in the communication system. Then in order to verify robustness we perform the following tests, which again correspond to affecting the perception, action and cooperation capabilities of the agents. (i) Sensor malfunctionings which can give faulty information. By introducing world estimation mistakes by sensors, it is possible to have a robustness measure of the information fusion process with respect to noisy information. (ii) Communication errors: some messages can be lost. Introduction of noise in the communication network by admitting loss of messages can give an estimation of the fusion process sensitivity to communication errors. (iii) Temporal or permanent inactive sensors. Alteration of network sensors can give a measure of system ability to adapt to unpredictable situations.

6 Conclusions and future work

We have presented a methodology for evaluating the performance of a Multi-Agent Systems in the framework of a simulation environment for post earthquake scenarios. The aim of the work is to address the problem of devising methods and evaluating the performance of specific aspects of MAS. In particular, we reported some results on task assignment and presented a proposal for extending the approach to high level information fusion. Our idea is to attempt first to address the two aspects separately, in order to refine the experimental methodology and understand the specific features of each of them. However, in practice they coexist and deeply influence each other, since the management and coordination of a Multi-Agent System requires good knowledge of the situation where the operations take place. Therefore, after completion of the first phase, we plan to combine them and extend the approach to deal with both of them at the same time. In this extended framework the critical question of action vs perception, will be addressed in the context of a simulated MAS.

In addition, we are developing this work in collaboration with the Italian Fire Department, which provides us with technical expertise on rescue operations that we take into account in the design of both the simulation system and the MAS. In this respect, we are planning to build a new map with a scenario embodying features that are substantially different from those currently available, and to compare implemented strategies with those actually deployed by the forces of Italian Fire Department to make our working hypotheses in the simulation more accurate with respect to actual emergency scenarios.

[1] D.L. Hall and J. Llinas (Eds.)[illegible]ors. *Handbook of Multisensor Data Fusion*. CRC Pre[illegible]001., 2001.

[2] K. Sycara and M. Lewis. From data to actionable knowledge and decision. In *IEEE Int. Conf. Information. Fusion*, AnnaPolis, July 2002.

[3] J. Salerno. Information fusion: a high-level architecture overview. In *IEEE Int. Conf. Information. Fusion*, AnnaPolis, July 2002.

[4] Performance metrics for intelligent systems workshop. http://www.isd.mel.nist.gov/research_areas/ research_engineering/Performance_Metrics/PerMIS_2003/.

[5] G. A. Kaminka, I. Frank, K. Arai, and K. Tanaka-Ishii. Performance competitions as research infrastructure: Large scale comparative studies of multi-agent teams. *Journal of Autonomous Agents and Multi-Agent Systems*, 2002. To appear.

[6] J. Ferber. *Multi-Agent Systems*. Addison-Wesley, 1999.

[7] Robocup Rescue Web Site. http://robomec.cs.kobe-u.ac.jp/robocup-rescue.

[8] H. Kitano and et al. Search and rescue in large scale disasters as a domain for autonomous agents resarch. *IEEE International Conference on System, man and Cybernetics (SMC99), Tokyo*, 1999.

[9] S. Tadokoro and et al. The robocup rescue project: a multiagent approach to the disaster mitigation problem. *IEEE International Conference on Robotics and Automation (ICRA00), San Francisco*, 2000.

[10] J. Llinas. Information fusion for natural and man-made disasters. In *IEEE Int. Conference on Information Fusion*, AnnaPolis, July 2002.

[11] F. D'Agostino, A. Farinelli, G. Grisetti, L. Iocchi, and D. Nardi. Monitoring and information fusion for search and rescue operations in large-scale disasters. In *IEEE Int. Conf. Information Fusion*, pages 672–679, AnnaPolis, July 2002.

[12] A. Farinelli, G. Grisetti, L. Iocchi, S. Lo Cascio, and D. Nardi. Using the robocup-rescue simulator in an italian earthquake scenario. In *Proc. of the 1st International Workshop on Synthetic Simulation and Robotics to Mitigate Earthquake Disaster*, 2003.

[13] K. Sycara. Multiagent systems. *AI Magazine*, 19(2), pages 79–92, 1998.

[14] A. Farinelli, G. Grisetti, L. Iocchi, S. Lo Cascio, and D. Nardi. Design and evaluation of multi agent systems for rescue operations. In *Proc. of International Conference on Intelligent Robots and Systems (IROS'03)*, 2003.

[15] F. D'Agostino, A. Farinelli, G. Grisetti, L. Iocchi, and D. Nardi. Design and implementation of a Rescue-Agent Development Tool. In *Proceedings of the Workshop on Planning and real-time monitoring of rescue operations*, 2002.

[16] A. Knoll and J. Meinkoehn. Data fusion using large multi-agent networks: an analysis of network structure and performance. *Proceedings of International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI '94).*, pages 113 – 120, 1994.

[17] Zili Zhang and Chengqi Zhang. Result fusion in multi-agent systems based on owa operator. *Proceedings of Australasian Computer Science Conference, (ACSC2000)*, pages 234 – 240, 1999.

recent literature i[illegible]formation fusion. *IEEE Aerospace and Electronics Syst[illegible]agazine*, 16(3):7 – 14, 2001.

Task Assignment with Dynamic Token Generation

Alessandro Farinelli, Luca Iocchi, Daniele Nardi, and Fabio Patrizi

University of Rome La Sapienza,
Dipartimento di Informatica e Sistemistica
Via Salaria 113, Rome, Italy
lastname@dis.uniroma1.it

Summary. The problem of assigning tasks to a group of agents acting in a dynamic environment is a fundamental issue for a MAS and is relevant to several real world applications. Several techniques have been studied to address this problem, however when the system needs to scale up with size, communication quickly becomes an important issue to address; moreover, in several applications tasks to be assigned are dynamically evolving and perceived by agents during mission execution. In this paper we present a distributed task assignment approach that ensure very low communication overhead and is able to manage dynamic task creation. The basic idea of our approach is to use tokens to represent tasks to be executed, each team member creates, executes and propagates tokens based on its current knowledge of the situation. We test and evaluate our approach by means of experiments using the RoboCup Rescue simulator.

1 Introduction

The problem of assigning tasks to a group of agents or robots acting in a dynamic environment is a fundamental issue for Multi Agent Systems (MAS) and Multi Robot Systems (MRS) and is relevant to several real world applications. Many techniques have been studied to address this problem in different scenarios, providing solutions that in different ways approximate the optimal solution of the Generalized Assignment Problem (GAP), which consists in assigning a predefined set of tasks (or roles) to a set of agents maximizing an overall utility function that takes into account the capabilities of all the agents in the team.

While GAP requires the definition of a static set of tasks, that must thus be known in advance, in many application domains, tasks to be accomplished are not known a priori, but are discovered dynamically during the execution of the mission. Furthermore, when the system needs to scale up with size, communication quickly becomes an important issue to address.

The problem of dynamic task assignment has been studied and experimented by many researchers both in MRS (e.g. [3, 16, 10]) and in MAS (e.g. [6, 4, 7, 13]) communities. Several different aspects of the problem have been investigated and several approaches proposed. However, the growing complexity of missions in which robots and agents are involved pushes toward the development of novel solutions for task assignment, which are able to address the more challenging issues posed by the applications. For example, auction based approaches to task assignment, have been proved to fail in the RoboCup Rescue domain, due to high communication requirements [8].

In this paper we present a distributed task assignment approach that is able to dynamically discover new tasks to be accomplished according to the situation perceived by the agents during the execution of their activities, and to ensure very low communication overhead. We focus on task assignment for teams operating in environments that need to meet (soft) real time constraints in their mission execution, where agents involved have similar functionalities but possibly varied capabilities. The reference scenario we are interested in has the following characteristics: i) the domain and the number of agents involved pose strict constraints on communications; ii) agents may perform one or more tasks, but within resource limits; iii) too many agents fulfilling the same task lead to conflicts that needs to be avoided; iv) tasks are discovered during mission execution.

The basic idea of our approach is derived from previous work based on token passing [12]. Tokens are used to represent tasks that must be executed by the agents, and each team member creates, executes and propagates these tokens based on its knowledge of the environment. The basic approach is based on the assumption that one token is associated to every task to be executed and that the token is maintained by the agent that is performing such a task, or passed to another agent if the agent that has the token is not in the condition of performing it.

In the case of dynamic discovery of the tasks to be performed and thus of dynamic token generation, the token passing approach must be appropriately extended in order to limit the number of tokens associated to the same task. Indeed, in our reference scenario optimal performance is obtained when there is a limited number of agents cooperating to execute the same task; when too many agents operate on a single task the overall performance decreases, since they ignore other tasks that evolve in a dynamic environment. The algorithm presented in this paper allows every agent to generate tokens dynamically whenever a task to be accomplished is perceived, while limiting the number of tokens associated to the same task and minimizing the bandwidth (i.e. communication messages among agents) required.

We test and evaluate our approach by means of experiments on a simulated scenario, that models a team of fire-fighters engaged in fighting fires in a city. To this end, we use the RoboCup Rescue simulator, that models the evolution of fires in the buildings of a city, city traffic, fire-fighters actions of extinguishing fires and communication among them. In this scenario, the location of the fires are not known a priori and the fire-fighter agents find them during their activities; in addition fires may unpredictably spread over adjacent buildings if not extinguished in time. Moreover, communication constraints are very strict, since messages are both limited and costly (in terms of simulation time steps).

The results that are reported in this paper show that the proposed extension of the token passing approach provides good performance in this scenario, while maintaining a very low communication bandwidth and thus significantly increasing the scalability of the system. Therefore, the proposed approach is specifically well-suited for large scale teams operating in dynamic environment, as compared to other dynamic task assignment methods that require a wider communication bandwidth.

2 Problem Definition

The definition of the problem considered in this paper is derived from the GAP problem [14], which consists in assigning a set of tasks (or roles) $R = \{r_1 \dots r_m\}$ to a set of agents (or entities) $E = \{e_1 \dots e_n\}$ with different capabilities for each task $Cap(e_i, r_j) \in [0,1]$ (i.e. a reward for the team when agent e_i performs task r_j), different resources needed by the agents for performing each task $Resources(e_i, r_j)$, and the resources available for an agent $e_i.resources$. An allocation matrix A is used for establishing task assignment: $a_{i,j} = 1$ if and only if the agent e_i is assigned to task r_j. The goal for the GAP problem is to find such an allocation matrix, that maximizes the overall capability function:

$$f(A) = \sum_i \sum_j Cap(e_i, r_j) \times a_{i,j}$$

subject to:

$$\forall i \sum_j Resources(e_i, r_j) \times a_{i,j} \leq e_i.resources$$

$$\forall j \sum_i a_{i,j} \leq 1$$

For example, in the rescue scenario that we have considered in our experiments, tasks are fires to be extinguished and agents are fire fighter brigades. The capability of a fire fighter to extinguish a fire, maybe dependent on several parameters, however a good approximation could be to consider the capability as a function of distance from the fire; clearly, if the nearest fire fighter is allocated to each fire the team gain a reward in terms of total traveled distance and time to extinguish all the fires. Resources are represented by the amount of water needed to put out fires.

The above formulation is well defined for a static environment, where agents and tasks are fixed and capabilities and resources do not depend on time. However, in several applications it is useful or even necessary to solve a similar problem where the defined parameters changes with time.

For example, in the above mentioned rescue scenario, all the defined parameters clearly depends on time, (e.g. fire fighters capabilities are strongly dependent on the environment evolution). Indeed several methods for dynamic task assignment implicitly take into consideration such an aspect, providing solutions that consider the dynamics of the world and derive a task allocation that approximate solutions of the GAP problem at each time steps (see for example [3, 16, 10, 8]).

The method described in this paper follows the line described above, and aims at solving the GAP problem when the set of tasks R is not known a priori when the mission starts, but it is discovered and dynamically updated during tasks execution.

To describe our method we will use the following notation. We denote that the set R depends on time with $R(t) = \{r_1 \dots r_{m(t)}\}$, where $m(t)$ is the number of tasks considered at time t, and we express the capabilities and the resources depending on time with $Cap(e_i, r_j, t)$, $Resources(e_i, r_j, t)$, and $e_i.resources(t)$. The dynamic allocation matrix is denoted by A_t, in which $a_{i,j,t} = 1$ if and only if the agent e_i is assigned to task r_j at time t. Consequently, the problem definition is to find a dynamic allocation matrix that maximizes the following function

$$f(A_t) = \sum_t \sum_i \sum_{j=1}^{m(t)} Cap(e_i, r_j, t) \times a_{i,j,t}$$

subject to:

$$\forall t \forall i \sum_{j=1}^{m(t)} Resources(e_i, r_j, t) \times a_{i,j,t} \leq e_i.resources(t)$$

$$\forall t \forall j \in \{0, \dots, m(t)\} \sum_i a_{i,j,t} \leq 1$$

3 Token Generation for Tasks Allocation

The main idea of the token passing approach is to regulate access to tasks execution through the use of tokens, i.e. only the agent currently holding the token can execute the task. Following this approach the communication needed to guarantee that each task is performed by one agent at time is dramatically reduced (see [2]).

If a task can benefit from the simultaneous execution of several agents, we can decide to create several tokens referring to the same task. However, when tokens are generated and perceived by agents during mission execution conflicts on tasks may arise. In this paper we will deal with two kinds of conflicts: the first one is due to the fact that the same task can be perceived by several agents during the missions, and if no explicit procedure is used the allocation process has no control on the maximum number of agents operating on such a task; this can lead to a consistent waste of resources and result in poor performance. The second type of conflict arises when an agent accomplishes a task and other tokens referring to the same task are still active, causing agents to waste precious time in trying to accomplishing terminated tasks.

We explicitly address these problems by proposing an extension to the algorithm presented in [2]. In the following, a *Task* refers to the physical object or event that the agent perceives and that implies an activity to be executed (e.g. a fire to be extinguished), therefore given a perceived object o we define the related task $T(o)$; a *Token* comprises the physical object related to the task and an identification number, that identifies different tokens for the same task, therefore given a task $T(o)$ we may have a number s of tokens $TK(o,1)...TK(o,s)$. The main idea of the proposed algorithm is that when an agent perceives a task, it records this information in a local structure and announces the presence of the task to all its team mates. Only the agent that first perceives a new task (e.g. a fire) creates one ore more tokens for it; conflicts that might arise due to simultaneous perception are addressed and solved as explained later. Whenever, an agent accomplishes a task it announces to the entire team the task termination, and each of the team members removes the tokens referring to the accomplished task from their local structures.

Using this approach conflicting tokens can still be created for two main reasons: i) **Contemporary task discovery**: two agents e_1 and e_2 perceive a new task t, creating a set of tokens $Tk(t,1)...Tk(t,s)$ exactly at the same time, such that both agents will have different tokens referring to the same task. ii) **Messages asynchrony** Assume we have three agents e_1, e_2, e_3; if e_1 immediately after the creation of a new set of tokens $Tk(t,1)...Tk(t,s)$ decide to send one of them, say $Tk(t,j)$, to agent e_3, this token will not be found in the local structure of e_1 when the announce messages of e_2 arrives and therefore will not be deleted; for e_3 we can have two situations: a) the message referring to token $Tk(t,j)$ arrives before the announce message of e_2 b) the announce message of e_2 arrives before the message referring to token $Tk(t,j)$. In both these situations the token $Tk(t,j)$ will not be deleted, and the conflict will not be solved. Both these problems have been addressed and solved in our approach as explained later in this section.

In the algorithm the following data structures are used: i) Known Tasks Set (KTS) is a set containing at each time step all the tasks that has been perceived by all the agents; ii) Token Set (TkS) is the set of tokens each agent currently holds; iii) Temporary Token Set (TmpTkS) is a set containing the tokens created by the agent in the current time step; iv) Accomplished Tasks Set (ATS) is a set containing at each time step all the tasks that have been accom-

Algorithm 1: Procedures for on line token generation

```
OnPercReceived($task$)
(1)  if ($task \notin KTS$)
(2)    $KTS = KTS \cup task$
(3)    $TmpTkS = TmpTkS \cup T(task, 1) \cup ... \cup T(task, s)$
(4)    send(Msg(Announce,$task$))

OnMsgReceived($Msg$)
(1)  if $Msg.type == AccomplishedTask$
(2)    $ATS = ATS \cup Msg.task$
(3)  if $Msg.type == Announce$
(4)    if ($Msg.task \notin KTS$)
(5)      $KTS = KTS \cup \{Msg.task\}$
(6)    else
(7)      if $Msg.senderId \geq MyId$
(8)        $TmpTkS = TmpTkS \setminus \{T | \forall j T(Msg.task, j)\}$
(9)        if $CurrentTask == Msg.task$
(10)         StopCurrentTask()
(11) if $Msg.type == Token$
(12)   $TkS = TkS \cup Msg.Token$

OnTaskAccomplishment($task$)
(1)  $ATS = ATS \cup task$
(2)  send(Msg(AccomplishedTask,$task$))

TokenManagement()
(1)  $TkS = TkS \setminus ATS$
(2)  $TokenSet$ = ChooseTokenSet(TkS)
(3)  $SendTokenSet = TkS \setminus TokenSet$
(4)  Send(Msg(Token,$SendTokenSet$))
(5)  $TkS = TkS \cup TmpTkSet$
(6)  StartTask(ChooseTask(TokenSet))
```

plished by all the agents each of this data structure is local to one agent. v) A message has three fields: *type* $\in \{announce, accomplishedTask, token\}$, ***task*** that contains information about the perceived task (e.g. fire position), valid when *type* is *announce* or *accomplishedTask*; finally the ***token*** field is valid only when the message is of type *token* and contains information about the token (e.g. task position, Id number, visited agents etc.); whenever an agent detects a new task through its perception it adds the new task to the KTS, creates s tokens referring to the task and adds them in the TmpTkS, then it *Announce* the new task to all its team members (Algorithm 1 OnPercReceived). Each team member when accomplishes a task sends an accomplished message to all its team mates and update its ATS (Algorithm 1 OnTaskAccomplishment). Each team member when receiving a message updates its local structures as explained in Algorithm 1, OnMsgReceived. Whenever a task is perceived, a new token is generated only if that task is not present in the *KTS*. After tokens have been processed (Algorithm 1, TokenManagement) the *TmpTkS* is copied in the *TkS* . Assuming that messages cannot get lost, Algorithm 1 guarantees that when an agent a perceives a task T, that has already been discovered before (i.e. that is present in the *KTS*), it will not create new tokens for it, correctly assuming that someone else already has the token(s) for T.

Notice that OnPercReceived, OnMsgReceived and OnTaskAccomplishment are asynchronous procedures, triggered by particular events; theoretically all the possible interleaving of their execution could occur, however, if we assume that each procedure is atomic (which is a reasonable assumption since no synchronization among agents is involved), we can guarantee that there will never be two tokens referring to the same task in the system for a time longer than the time required for the *Announce* messages to reach all the team members. In fact, as explained above conflicting tokens may be created in case of **Contemporary task discovery** or due to **Messages asynchrony**.

The problem of **Contemporary task discovery** is considered and solved by procedure OnMsgReceived: when agents receive the announce messages the one with a lower static priority, represented in the procedure by the lower Id number, will delete the token for task t from *TmpTkS*, solving the conflict; if t is already being executed by the agent with lower static priority, it will stop its execution yielding to the higher priority agent the possibility to execute the task. The problem arising due to **Messages asynchrony** is avoided thanks to the distinction between temporary tokens (stored in *TmpTkS*) and normal tokens (stored in *TkS*). In fact, assuming that the time needed for an announce message to reach all the agents is less than one simulation step (i.e. assuming that messages are synchronized with agents execution) the use of a Temporary Token Set guarantees that the conflicts will be detected and avoided. Otherwise, a higher communication overhead is needed in order to recover from such conflicts.

Setting a static fixed priority among agents can obviously result in non optimal behavior of the team, for example assuming that $Cap(e_1, r_j, t_i) > Cap(e_2, r_j, t_i)$ following the static priority, we yield to the less capable agent the access to the task r_j. However, while theoretically the difference among capabilities can be unbounded, generally, when tasks are discovered using perception capabilities agents perceive tasks when they are close to the object location, (e.g. if two fire fighters perceive the same fire their distance from the fire is comparable) and therefore the loss of performance due to the use of a fixed priority is limited.

Once a token has been created and added to the *TkS* the token-based access to values requires that each agent decides whether to execute the tasks represented by tokens it currently has or to pass the tokens on. The token management procedure of Algorithm 1 describes how tokens are processed: each agent erases from its *TkS* the accomplished task set *ATS*, then it chooses a set of tokens it can execute (ChooseTokenSet(TkS)). Each agent follows a greedy policy in this decision process, i.e. it tries to maximize its utility given the tokens it currently can access and its resource constraints. However, each agent in its decision consider whether it is in the best interest of the team for it to execute the tasks represented by its tokens. The key question is whether passing the token on will lead to a more capable team member taking on the token. Using probabilistic models of the members of the team and the tasks that need to be assigned, the team member can choose the minimum capability the agent should have in order to take on a token. Each agent sends the remaining tokens to its team mates, following a round robin policy and copies the $TmpTkS$ in the TkS. Finally, each agent chooses the best task (e.g. for fire fighters could be the nearest fire) among the $TokenSet$ it currently has (ChooseTask(TokenSet)) and starts the task execution.

4 Experiments and Results

We tested our task assignment approach in the RoboCup Rescue environment [5]. RoboCup Rescue provides an ideal simulation environment to test allocation strategies for team comprised of rescue agents. We focus on a real city map of Foligno in Italy [9], so as to test the performance of our approach in a realistic disaster rescue environment and where agents must navigate narrow streets and passages. Here, a team of fire brigades must fight fires in real-time, while facing the uncertainty of fire spreading and the dynamism that arises due to several factors: (i) agent has a limited view of the world, and do not know in advance fires initial positions (ignition points); (ii) the way fires spread can not be precisely predicted; (iii) agents can be blocked in narrow passages.

To show that the algorithm presented in section 3 does actually avoid conflicts of both types, we implemented three different kinds of allocation strategies. The first strategy, referred to as Token Passing (TP), is a plain implementation of the token based approach algorithm, no announce procedure is used, but agents record in a Known Fire List the fires they perceive to avoid that different agents create two tokens for the same fire. This strategy does not enforce any constraint on the maximum number of agents simultaneously fighting the same fire. The second strategy, referred to as TP with Announce (TPA-n), makes use of the announce procedure to enforce that no more than n agents are simultaneously fighting the same fire, however this strategy does not address the second kind of conflict type, therefore situations in which agents can try to fight already extinguished fires may arise. The third strategy, referred to as TPA-n with AccomplishedTask (TPAA-n), makes use of the *announce* and *AccomplishedTask* messages, avoiding both types of conflicts.

In all the strategies the processing token procedure is the same and the capability to execute a task is computed considering the distance between the fire fighting agent and the fire, and whether the agent is blocked in a narrow passage. If an agent is blocked, it sends out the task it is currently executing and choose a different task from its set. The set of tasks to be executed is computed choosing the nearest fire f and keeping up to K fires whose distance from f is lower than a fixed Threshold T. The Threshold T and the number of tokens each agent can retain is statically defined, and is computed considering global information, such as the number of agents involved in the simulation and their distribution on the map. For a detailed discussion on how this static values can be computed we refer to [11].

We tested each strategy in different operative conditions, changing the extinguish power the fire fighting agents have. We start each simulation from the same initial configuration, comprised of 10 fire fighting agents and 18 ignition points distributed as shown in Figure 1;

In this experiments we assume that messages cannot be lost, and that their delay is not higher than a simulation step (i.e. agent execution is synchronized with message passing), moreover we set the number of tokens to be created for each task to be a fixed number (three in the performed experiments); while it is possible to dynamically change this number during mission execution depending on the environment situation, in these experiments we focus on studying how conflicts influence performance of fire fighting agents, leaving the problem of how many tokens would be needed for each task and how to deal with possible lost messages and unpredictable delays to later investigation.

We extracted from the performed experiments the *extinguish time*, as the time needed to put out all the fires, the number of *point to point messages* exchanged among agents per time step, the number of *broadcast messages* sent by agents per time step, the total *traveled distance* per agent and finally the total number of *conflicts*, as the number of times during the entire simulation that more than three agents have the same fire as target.

	TP	TPA-3	TPAA-3
Ext. Time	67 [0.7]	59.63 [16.6]	50.62 [2.5]
Ptp Msg per time step	1.4 [0.04]	1.8 [0.56]	1.7 [0.053]
Bcast Msg per time step	0 [0]	0.68 [0.63]	1.63 [0.13]
Trav. Dist. per agent	2495 [198]	3201 [527]	2221 [195]
Conflicts	26.62 [1.92]	0 [0]	0 [0]

Table 1. Results obtained averaging 10 simulations

In Table 1 we report results obtained from the simulations performed. Each reported value is the average obtained from ten repetitions of the simulation with the same operative conditions, along with the computed standard deviation (reported between brackets). From the table it is possible to see that the TPAA-3 strategy consistently outperform the TP strategy with a higher but still acceptable amount of messages. Moreover, the traveled distance for each

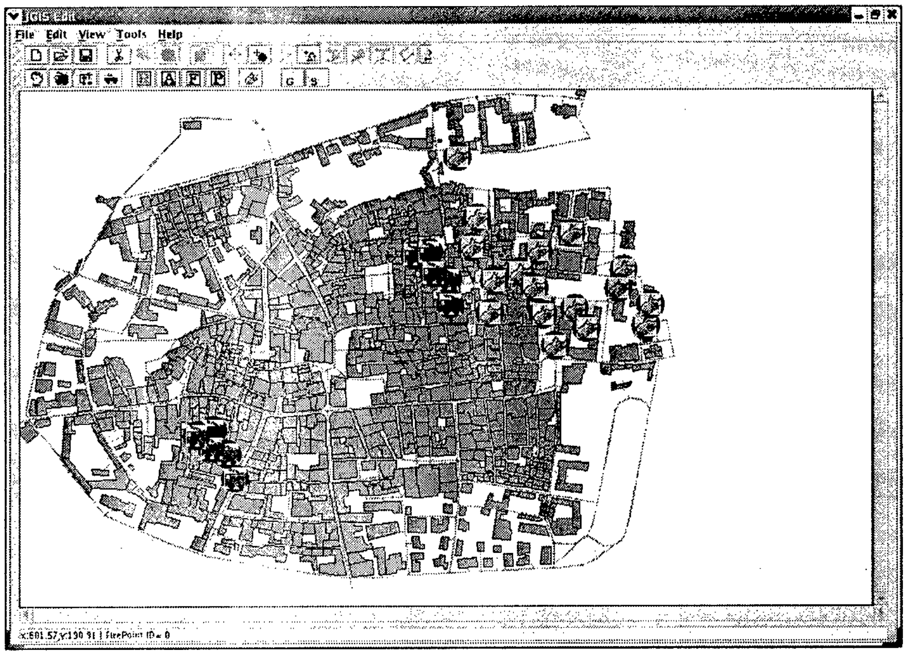

Fig. 1. Foligno Map used in the experiments

agent is smaller on average, showing that better results are reached with a smaller waste of resources. The performance of TPA-3 strategy are on average in the middle with respect to TP and TPAA-3, however this strategy is characterized by a very high variance specially regarding the extinguish time and traveled distance. The high variance is due to the fact that the strategy does not avoid the second type of conflicts, possibly generating consistent resource wasting.

In the performed experiments we have used values for extinguish power ranging from 6000 (water unit per minute) and up to model situations where it is useful that the agents allocation is balanced among the different tasks. Indeed, we found that the similar relationships among strategies hold increasing the extinguish power from 6000 (results reported in table 1) to 8000 and 10000.

5 Conclusions and Future works

Task allocation is a very widely studied area and several approaches have been presented in literature addressing different issues and techniques ranging from forward looking optimal model [8], to market or auction based techniques [16, 4], to symbolic matching [15] and Distributed Constrained Optimization

Problem based algorithms [6]. However, the growing complexity of application for MAS and MRS requires novel solutions for task assignment, which are able to address specific features posed by the domain, such as dynamic tasks evolution, strict constraints on communication and soft real time constrained to be met.

Token based approach have been proved to be well suited for task allocation in such scenario [13, 1], however the specific problems of dynamic token generation and conflicts resolution have not been considered yet. In this paper we take a step in this direction proposing an extension to the token approach able to address this issue while keeping a reasonably low communication overhead. Moreover, we present first experimental results obtained for our approach, showing that it is actually applicable in a rescue scenario and is able to resolve conflicts improving the performances of the rescue teams.

Several other issues need to be further addressed, in particular we intend to test our algorithm with different types of rescue teams, such as ambulances or police force. The ambulance case is particularly interesting because it is important to enforce the constraint that only one agent can take care of a civilian, since no further benefit can be given to the team by having more than one ambulance trying to pick up a civilian, therefore we plan to further test our approach with ambulances. When dealing with different forces type constrained tasks comes into play, for example an ambulance agent could need to have a blocked road freed to pick up a civilian by a police agent, and an evaluation of our approach in such situation is particularly interesting. Finally, in our working scenario we assumed that no messages can be lost, this is quite a strong assumption, that can be easily violated in real world applications, therefore an interesting extension of our method will be devoted to explicitly deal with such situations.

Acknowledgment

This effort was partially funded by the U.S. Air Force European Office Of Scientific Research under grant number 033065 and by project "Simulation and Robotic Systems for intervention in emergency scenarios" within program COFIN03 of the Italian MIUR, grant number 2003097252

References

1. A. Farinelli, P. Scerri, and M. Tambe. Allocating and reallocating roles in very large scale teams. In *First Int. Workshop on Synthetic Simulation and Robotics to Mitigate Earthquake Disaster*, Padua, Italy, July 2003.
2. A. Farinelli, P. Scerri, and M. Tambe. Building large-scale robot systems: Distributed role assignment in dynamic, uncertain domains. In *Representation and approaches for time-critical decentralized resources/role/task allocation (AAMAS WorkShop)*, 2003.

3. B. Gerkey and J. M. Matarić. Multi-robot task allocation: Analyzing the complexity and optimality of key architectures. In *Proc. of the Int. Conf. on Robotics and Automation (ICRA'03)*, Taipei, Taiwan, Sep 14 - 19 2003.
4. L. Hunsberger and B. Grosz. A combinatorial auction for collaborative planning. In *Proceedings of the Fourth International Conference on Multi-Agent Systems (ICMAS-2000)*, pages 151–158, 2000.
5. H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, E. Osawa, and H. Matsubara. RoboCup: A challenge problem for AI. *AI Magazine*, 18(1):73–85, Spring 1997.
6. Roger Mailler, Victor Lesser, and Bryan Horling. Cooperative negotiation for soft real-time distributed resource allocation. In *Proceedings of AAMAS'03*, 2003.
7. P. J. Modi, H. Jung, W. Shen, M. Tambe, and S. Kulkarni. A dynamic distributed constraint satisfaction approach to resource allocation. In *Proc of Constraint Programming*, 2001.
8. R. Nair, T. Ito, M. Tambe, and S. Marsella. Task allocation in robocup rescue simulation domain. In *Proceedings of the International Symposium on RoboCup*, 2002.
9. D. Nardi, A. Biagetti, G. Colombo, L. Iocchi, and R. Zaccaria. Real-time planning and monitoring for search and rescue operations in large-scale disasters. Technical report, University "La Sapienza" Rome, 2002. `http://www.dis.uniroma1.it/~rescue/`.
10. L. E. Parker. ALLIANCE: An architecture for fault tolerant multirobot cooperation. *IEEE Transactions on Robotics and Automation*, 14(2):220–240, 1998.
11. P. Scerri, A. Farinelli, S. Okamoto, and M. Tambe. Allocating roles in extreme team. In *AAMAS 2004 (Poster)*, New York, USA, 2004.
12. P. Scerri, A. Farinelli, S. Okamoto, and M. Tambe. Token approach for role allocation in extreme teams: analysis and experimental evaluation. In *13th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE-2004).*, Modena, Italy, 2004.
13. P. Scerri, D. V. Pynadath, L. Johnson, Rosenbloom P., N. Schurr, M Si, and M. Tambe. A prototype infrastructure for distributed robot-agent-person teams. In *In Proceedings of AAMAS*, 2003.
14. D. Shmoys and E. Tardos. An approximation algorithm for the generalized assignment problem. *Mathematical Programming*, 62:461–474, 1993.
15. G. Tidhar, A. S. Rao, and E. A. Sonenberg. Guided team selection. In *Proceedings of the Second International Conference on Multi-Agent Systems*, 1996.
16. R. Zlot, A Stenz, M. B. Dias, and S. Thayer. Multi robot exploration controlled by a market economy. In *Proc. of the Int. Conf. on Robotics and Automation (ICRA'02)*, pages 3016–3023, Washington DC, May 2002.

Task Assignment with dynamic perception and constrained tasks in a Multi-Robot System*

A. Farinelli, L. Iocchi, D. Nardi, V. A. Ziparo
Dipartimento di Informatica e Sistemistica
University of Rome "La Sapienza"
Via salaria, 113 00198 Rome, Italy
last-name@dis.uniroma1.it

Abstract— **In this paper we present an asynchronous distributed mechanism for allocating tasks in a team of robots. Tasks to be allocated are dynamically perceived from the environment and can be tied by execution constraints. Conflicts among team mates arise when an uncontrolled number of robots execute the same task, resulting in waste of effort and spatial conflicts. Conflicts are due both to the distributed perception of tasks and to wrong data association of robots for moving objects. The proposed approach is able to successfully allocate roles to robots avoiding conflicts among team mates and maintaining low communication overhead. We performed test in simulated environment to collect quantitative data in different operative conditions, and on real robots.**

Index Terms— **Multi-Robot Systems, Coordination, Task Assignment.**

I. Introduction

The problem of assigning tasks to a group of entities acting in a dynamic environment is a fundamental issue for both Multi Agent and Multi Robot Systems and is relevant to several real world applications. Many techniques have been studied to address this problem in different scenarios, providing solutions that in different ways approximate the optimal solution of the Generalized Assignment Problem (GAP): assigning a predefined set of tasks (or roles) to a set of entities, maximizing an overall utility function that takes into account the capabilities of all the entities.

Solutions proposed for MAS range from forward looking optimal model [7], to market or auction based techniques [13] and Distributed Constrained Optimization Problem based algorithms [6]. As for MRS, proposed approaches include i) Sequential task assignment [5] where tasks are allocated to robot sequentially as they enter the system. ii) Iterative task assignment [1], [12] where all tasks present in the system are allocated from scratch at each time step. iii) Reactive task assignment [8] where each member of the team decides whether to employ itself in accomplishing a task, without (re)-organizing the other members activity.

The growing complexity of applications for MAS and MRS require novel solutions for task assignment, which are able to address specific features posed by the domain, such as dynamic tasks evolution, strict requirements on communication and constraints among tasks to be executed. In most real world applications involving MRS, tasks to be assigned cannot be inserted into the system in a centralized fashion, but are perceived by each entity during mission execution. This issue has a big impact on the task allocation process and at the same time is strictly dependent on perception capabilities of entities involved. Despite its importance, the integration of dynamic task perception and distributed task assignment has received little attention in the MRS community.

In this paper we take a step in this direction by presenting an approach to distributed task assignment able to assign dynamically perceived tasks in a team of robotic agents, while avoiding conflicts among team members. Since we focus on a team of physical robots, conflicts on tasks to be accomplished can cause significant inefficiencies due to the complexity of acting in a real world environment. Moreover, perception capabilities play a fundamental role in the design of the coordination approach. In particular, in several robotic scenarios distinguishing objects with similar shape and color among them is not trivial. This requires to consider, in the data association process, properties that can change with time (such as object position in the working environment). Finally, tasks to be accomplished may be tied by constraints, requiring the agents to synchronize the execution of their tasks.

Summarizing, the reference scenario we are interested in has the following characteristics: i) tasks are discovered and created during mission execution; ii) tasks may need multiple agents to perform them synchronizing their actions; iii) agents may perform one or more tasks, but within resource limits; iv) too many agents fulfilling the same task lead to conflicts that needs to be avoided; v) properties that distinguish tasks can vary over time.

The basic idea of our approach is derived from previous works on token passing for task assignment which have been proved to be well suited for task allocation in similar scenarios [9]. Tokens are used to represent tasks that must be executed by the agents, and each team member creates, executes and propagates these tokens based on its knowledge of the environment. The basic approach relies on the assumption that one token is associated to every task to be executed and that the token is maintained only by the agent that is performing such a task. If the agent is not in the condition of performing the task it can decide to pass the token on to another team member.

This paper makes use of the token passing approach, by introducing the new concept of dynamic token generation, i.e. tokens are not statically predefined, but generated online during mission execution as result of robots perceptions. Moreover, the proposed method assigns tasks using only a broad knowledge of team mates and sharing a minimal set of information, thus ensuring low communication overhead and scalability of the approach. Finally, the approach provides for a distributed asynchronous algorithm, that is able to guarantee a conflict free allocation of exactly n agents for each task.

We tested our approach with a team of AIBO robots in a foraging task (see [3] for a description of MRS testbeds). Robots have to collect several objects scattered in the environment. The collection of each object requires that exactly two robots help each other to grab it (a helper robot and a collector robot), while, after the grabbing phase, only one robot is needed to transport the object. Objects number and position in the environment is not known, thus enforcing task discopvery through perception, and objects are identical so they can only de distinguished by their position in the environment, thus presenting properties that change over time.

It is important to highlight that this is a quite complex scenario for MRS coordination, that takes into account specific characteristics of a real-time environment (e.g. indistinguishable objects), that are not usually considered in Multi-Agent Systems. Moreover, these features have required the definition of a new task assignment problem and associated solutions, that are not taken into account in previous works (e.g. [2], [9]).

For this scenario we have performed experiments both in a simulated environment and with real robots. The reported results show that the proposed approach is able to allocate exactly n robots to each task thus avoiding possible conflicts with other team mates, while maintaining a very low communication bandwidth. The paper is organized as follow: next section defines the problem we are addressing, Section III presents the algorithm used and Section IV shows the experimental setting and the obtained results. Section V draws conclusions and sketches future work.

II. Problem Definition

The problem of assigning a set of tasks to a set of entities can be easily framed as a GAP [10]. However, while the GAP is well defined for a static environment, where agents and tasks are fixed and capabilities and resources do not depend on time, in several real world applications it is useful or even necessary to solve a similar problem where the defined parameters changes with time.

Indeed several methods for dynamic task assignment implicitly take into consideration such an aspect, providing solutions that consider the dynamics of the world and derive a task allocation that approximate solutions of the GAP problem at each time steps [4], [7], [8], [13].

The problem we will address in this paper differs from the GAP formulation in two main respects: i) Tasks to be accomplished can be tied by constraints, ii) The set of tasks R is not known a priori when the mission starts, but it is discovered and dynamically updated during task execution.

To take in to account those differences we will use the following notation: $E = \{e_1 \dots e_n\}$ denotes the set of entities. While in general also entities involved in the task assignment process can vary over time, in this contribution we focus on a predefined static set of entities. T represents the set of tasks and is dependent on time with $T_t = \{t_1 \dots t_{m(t)}\}$, where $m(t)$ is the number of tasks at time t. Each task is a set of roles or operations $T_i\{r_i^1 \dots r_i^k\}$, with k varying from task to task. For example, the Task *lift object O* comprises two roles: *collect O* and *support collection of O*. Notice, that each operation can comprise a set of sub-operations and so on; for the sake of simplicity we will consider only two levels of the possible hierarchy. Each entity has different capabilities for each task and different resources available. We express the capabilities and the resources depending on time with $Cap(e_i, r_j^k, t)$, $Res(e_i, r_j^k, t)$, and $e_i.res(t)$. Where $Cap(e_i, r_j^k)$ represents the reward for the team when agent e_i performs role r_j^k at time t, $Res(e_i, r_j^k)$ represents the resources needed by e_i to perform r_j^k at time t, while $e_i.res(t)$ represents the available resources for e_i at time t.

A dynamic allocation matrix is used to establish task assignment, denoted by A_t; in A_t, $a_{i,j,k,t} = 1$ if and only if the agent e_i is assigned to task r_j^k at time t. Furthermore, to represent constraints, we define $\bowtie$ as the set of relationships which hold among roles. Consequently, the problem is to find a dynamic allocation matrix that maximizes the following function

$$f(A_t) = \sum_t \sum_i \sum_j \sum_k Val(e_i, r_j^k, \bowtie, t) \qquad (1)$$

with:

$$Val(e_i, r_j^k, \bowtie, t) = \begin{cases} Cap(e_i, r_j^k, t) & if\ Cond \\ 0 & otherwise \end{cases} \qquad (2)$$

subject to:

$$\forall t \forall i \sum_{j=1}^{m(t)} Res(e_i, r_j^k, t) \times a_{i,j,k,t} \le e_i.res(t) \qquad (3)$$

$$\forall t \forall j \in \{0, \dots, m(t)\} \sum_i a_{i,j,k,t} \le 1 \qquad (4)$$

where $Cond$ is true when constraints relative to r_j^k are satisfied.

Constraints can possibly be of several types (OR, XOR, AND), in this paper we focus only on AND constraints. When an AND constraint holds among a group of roles, each agent cannot perform any role if all the other roles are not being performed simultaneously by some other team mates. We write an AND constraint as $AND^i = \{r_v\}$

therefore $\bowtie = \{AND^1 \dots AND^s\}$. If a role $r_v \in AND^z$ then the equation describing $Cond$ is:

$$Cond \equiv \sum_i \sum_{r_j^k \in AND^z} a_{i,j,k,t} = |AND^z| \qquad (5)$$

Notice that if the role is unconstrained, $|AND^z| = 1$, then $Val(e_i, r_j^k, \bowtie, t) = Cap(e_i, r_j^k, t) \times a_{i,j,k,t}$, as above.

Agents have only a local view of the environment, thus we define two sets: $LOT_{i,t}$, which is the task set locally known to agent i at time t (Locally Observed Tasks), and $GOT_t = \bigcup_i LOT_{i,t}$ which is the union of tasks locally known to each agent i at time t (Globally Observed Tasks). The $LOT_{i,t}$ is built by each agent based on its local perception thus we can write $LOT_{i,t} = Mem(LOT_{i,t-1}, O(i,t))$ where $O(i,t) : A \times Time \rightarrow \mathcal{P}(T \times \{0,1\})$. Given an agent a and a time step t $O(i,t)$ returns a set of pairs $< T, 1 >$ if task T is active and visible for agent a at time step t, and $< T, 0 >$ if task t is visible for the agent but is not active. Notice that assuming that the observation function is able to distinguish between active and non active tasks is quite a strong assumption; for example, in a robotic foraging task where objects are similar in shape and color and the distinction is made using object position, knowing if an object is being moved by another team mate is not a trivial problem. Thus, in this contribution we consider cases where the observation function can fail in deciding whether a detected object is to be considered active, and explicitly address this problem in the coordination method. An active task is a new perceived task that need to be accomplished.

The Mem function integrates observation for an agent during the mission execution. We assume that the Mem function add newly discovered tasks and is able to remove non active tasks from the Locally Observed Tasks set. We define $LRS_{i,t} = \{r_j^k | a_{e_i, r_j^k} = 1\}$ (Local Roles Set) and $GRS_t = \bigcup_i LRS_{i,t}$ which are the currently assigned roles (Global Roles Set). An *allocation* for a D-GAP problem is the set $Alloc = \{LRS_{i,t}\}$. The global constraints 4 can be then expressed as

$$\bigcap_i LRS_{i,t} = \emptyset \qquad (6)$$

We define a non conflicting allocation as an allocation for which the following holds:

$$\forall t \bigcap_i LRS_{i,t} = \emptyset \qquad (7)$$

III. Token Passing Approach for Role Allocation

The main idea of the token passing approach is to regulate access to task execution, through the use of tokens, i.e. only the agent currently holding the token can execute the task. Following this approach the communication needed to guarantee that each task is performed by one agent at time is dramatically reduced.

If a task can benefit from the simultaneous execution of several agents, one possible strategy is to create several tokens referring to the same task. However, when tokens are perceived and generated by agents during mission execution conflicts on tasks may arise due to the fact that several agents may perceive the same task; and thus create an uncontrolled number of tokens leading too many agents to execute the same role. Moreover, an explicit procedure is needed to enforce AND constraints among roles. In the following, we present and discuss our approach to ensure that exactly n agents perform the same role simultaneously, thus solving both the issues previously highlighted.

A *Task* is characterized by the physical objects (or events) that the agent perceives, therefore given a perceived object o we define the related task $T(o)$. We associate a token to each role comprising a task to be accomplished, thus for the object o we will have $T(o) = \{T(o)^1 \dots T(o)^k\}$. In particular, in our reference scenario, when a new object obj is found we need two robots cooperating to grab the object, therefore we have a task $Move(obj)$ and two roles (and thus two tokens) for this task: $\{collect(obj), support(obj)\}$.

To prevent possible conflicts, that may arise during mission execution, we have to guarantee that no more than n tokens are created for the same role. The main idea of the proposed algorithm is that when an agent perceives an object, it records this information in a local structure and announces the presence of the object to all its team mates. Moreover, whenever an agent accomplishes a task, it announces to the entire team task termination, and each of the team members removes the tokens referring to the accomplished task from its local structures.

Using this approach conflicting tokens can still be created for three main reasons: i) **Contemporary task discovery**: two agents e_1 and e_2 perceive a new task t, creating a set of tokens $Tk(t,1) \dots Tk(t,s)$ exactly at the same time, such that both agents will have different tokens referring to the same task. ii) **Messages asynchrony**: messages are not guaranteed to arrive in a predefined order; Suppose an entity observes a new object, creates a new token for a specified role and decides to pass the token on. If the token reach a team member before the new perception announcement, it can decide to perform the role possibly conflicting with other team mates. iii) **Errors in active object detection**: if the observation function fails in the recognition of an active object, a team members can decide to allocate itself to a role that is already being carried on by someone else.

In the following, we describe our approach to avoid conflicts during mission execution and further details of the token passing process.

A. Distributed conflict detection

Contemporary task perception and message asynchrony are addressed using a distributed approach for conflict

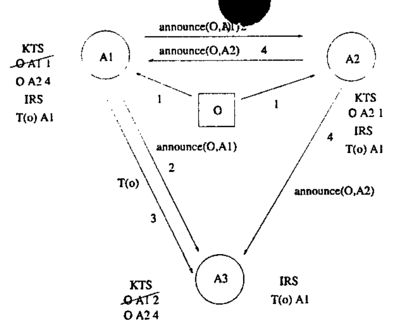

Fig. 1. Distributed conflict detection

detection and a predefined global policy that orders all the team mates. Each agent maintains in a local structure called Invalid Token Set (ITS) all the detected conflicting tokens. Each new announce is registered in the Known Task Set (KTS) maintaining the announcer agent information. Tokens are added in the ITS structure when an announce message for an already known object is received and the announcer agent has a lower priority as compared with the agent that already announced the object detection. If a higher priority agent announce the same object, then the creation the KTS is updated. An example of possible execution of this procedure is reported in Figure 1. Numbers on arrows correspond to time step at which messages are received by agents. The square represents the object and arrows starting from the square represent object perception. KTS is the Known Task Set and IRS is the invalid role set. Notice that KTS contains also information on the creator agent. As it is possible to see all agents share the same IRS without need of any further communication.

Setting a static fixed priority among agents can obviously result in non optimal behavior of the team; for example, assuming that $Cap(e_1, r_j^k, t_i) > Cap(e_2, r_j^k, t_i)$ following a static priority based on id, we yield to the less capable agent the access to the task r_j^k. While in principle the difference among capabilities can be unbounded, generally, when tasks are discovered using perception capabilities, agents perceive tasks when they are close to the object location, (e.g. if two robots perceive the same object their distance from the object is comparable); therefore, the loss of performance due to the use of a fixed priority is very limited.

B. Avoiding failures in active task recognition

Suppose that when a collector robot moves an object, another robot observes the same object, This robot will consider the object as a new task to accomplish starting a new allocation process for the task and possibly conflicting with the collector robot.

To address this problem we partition the working environment using a regular grid. While an object is perceived inside a cell of the grid it is considered as the same task. Suppose a robot observes the moved object and starts the allocation of a new task; when the collector robot is reached by an announce message for the object creation, it can detect that the new task is actually being performed by itself and declare the new created task as invalid. The collector robot will then announce the task invalidation to all its team mates sending an InvalidTask message in broadcast. Tasks are considered invalid only for a given amount of time. This is needed to avoid that collector robots passing near other (active) objects invalidate them for all the mission execution.

C. Token passing process

While we do not report here the complete algorithm we give some further details to clarify how the token passing process is used to assign roles to team members.

Once a token has been created and added to the TkS the token-based access to values requires that each agent decides whether to execute the tasks represented by tokens it currently has or to pass the tokens on. Each agent follows a greedy policy in this decision process, i.e. it tries to maximize its utility given the tokens it currently can access, its resource constraints and a broad knowledge on team composition (see [9] for further details).

When roles are tied by AND constraints, following this procedure will lead to potential deadlocks or inefficiencies. For example, consider two tasks, r_j and r_k, that need to be simultaneously performed. When a team member a accepts task r_j, it may reject other tasks that it could potentially perform. If there is no team member currently available to perform task r_k, a must simply wait. Thus, an explicit procedure to enforce the AND constraints among roles is needed. The general idea (as described in [9]) is to use potential tokens to represent tokens that are tied by AND constraints. Potential tokens retain agents; when an agent receives a potential token it can perform other roles (i.e. the potential token does not have impact on the current resource load of the agent). However, when a lock message arrives the agent is forced to execute the role. Whenever a constrained task is detected a fixed number of potential tokens are generated and sent out by the team member for each role associated to the task; the agent that created the potential token is then responsible to manage the handling of messages required to ensure that there are enough agents retained to perform the task and then send the lock message. The detailed procedure used to manage the AND roles is described in the algorithm *ANDMonitor* while the normal operations to manage token circulation are reported in algorithm *TokenMonitor*. Notice that the agent that manages the potential token can take on normal tokens just as all the other agent.

IV. Experiments and Results

In order to test our approach implemented our algorithm on the Sony AIBO robots used in the RoboCup legged league competitions [11]. Our reference scenario comprises a set of robots that need to perform a synchronized operation on a set of objects scattered in the environment. For each object two robots are needed to perform the synchronized operation, while only one robot is needed to accomplish the task.

In particular, in our reference scenario two robots are needed to cooperatively grab an object while only one robot is needed to transport it when the grabbing phase is terminated. Each robot knows all the other team mates and it is able to communicate with them through a wireless device. Robots do not know the number of objects deployed in the environment nor their positions.

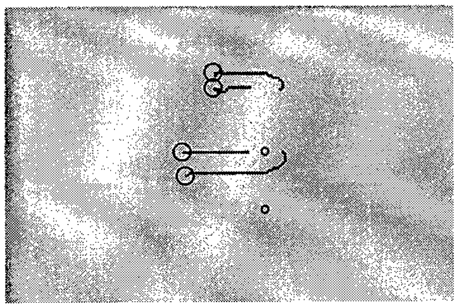

Fig. 2. Simulation environment

A. Simulation results

In order to have a consistent data set to analyze, we used a simulation framework for reproducing our reference scenario and performed our experiments in this environment (see Figure 2).

Each robot can perceive objects (represented as circles) in the environment with a limited field of view. This limitation in the perception leads to interesting situations, where different robots can perceive the same object and thus possibly generate conflicts as previously explained; or a robot can be notified about the presence of an object nearby, which is outside its visual field through a token sent by a team mate.

For a quantitative evaluation of our approach we have identified a set of initial configurations varying the number of robots involved in the foraging task, the position of the agents with respect to the objects to collect and the number of tasks initially perceived by agents. We let the allocation algorithm run until all the tasks have been accomplished and then we measured the time needed to accomplish the task and the number of exchanged messages.

In Table I we report the results obtained for two, four, five and seven robots in three different initial configurations: situation *All* represents the fact that each object can be detected by every agent. Situation *Many* represents the fact that no agent can detect all the objects, but every object can be detected by at least one agent. Finally, in situation *One*, only one agent can see all the objects. The reported results are averaged over ten runs each.

	One	Many	All
Two Agents			
Avg Time	122.44	84.887	111.24
Avg Msg Per Agent	33.15	36.92	44.63
Four Agents			
Avg Time	73.74	62.15	91.81
Avg Msg Per Agent	161	159	151.94
Five Agents			
Avg Time	66.44	47.12	69
Avg Msg Per Agent	70.74	131.98	96.57
Seven Agents			
Avg Time	58.32	58.44	95.42
Avg Msg Per Agent	135.84	121.13	171.19

TABLE I
RESULTS OBTAINED AVERAGING 10 SIMULATIONS

The experiments performed show that our approach ensures no conflicts on tasks and that AND constrained roles do not arise deadlocks, always converging to a valid allocation (i.e. exactly 2 robots for each task). As shown in the table, time needed for agents to accomplish all tasks decreases when the number of agents increases, but when too many robots are present in the working environment performance degrades due to spatial conflicts. Finally, number of messages exchanged remains acceptable in all the operative conditions.

B. Experiments with real robots

Experiments with real robots have been performed in order to validate the behaviour obtained in simulation. Although we have not collected quantitative data for real robot experiments and were not able to test the method with a high n umber of robots, we had experimental evidence of the results presented here also with a real robotic system.

In order to perform experiments with robots we have developed object localization capabilities and synchronized plans for task accomplishment. These features have been used to characterize objects and to handle non-instantaneous actions. We simplified in some way low level actions and used a suitable environment in order to put emphasis on coordination issues. The experimental setting, borrowed from the Robocup Legged League, is a rectangular field with six landmarks in known positions where every landmark is clearly distinguishable with color, while objects to be collected are identical colored wheeled-bars.

The first problem we have addressed was to distinguish objects in terms of their position given noisy perception and unreliable self-localization. This can produce errors leading to false positives in task detection. To avoid this problem, we filtered the absolute position of objects using a very conservative policy and active perception to ensure the correctness of the observations. In fact due to the poor reactiveness in perception, if an object is uncorrectly perceived while moving to the expected position of the current task, situations in which erroneous tasks are generated may occur. Therefore, once the robot is close enough to the object, it interleaves object tracking with landmark pointing action for self-localization in order to identify the object and understand if it is the one associated with his task.

Although the system is robust to false positives we cannot assume their absence. Thus, if during the execution of a role a robot realizes that an announced object is a false positive, it has to invalidate the corresponding task with an Invalid Role message.

Because of the small amount of messages needed by our coordination method we conveniently adopted the TCP protocol, so that we do not have to manage message loss.

Since tasks to be accomplished are complex and we can't assume instantaneous actions nor deterministic effects, the reactive module was developed to execute plans that can identify failures during action or plan execution and specify recovery procedures. Furthermore, advanced concurrency can be handled through synchronizations of parallel actions executed on one or more robots. In particular inter-robot synchronization was achieved providing the robot with actions capable of sending or sensing signals from the net.

The plans in Figures (3) and (4) are shown as example of inter robot synchronization.

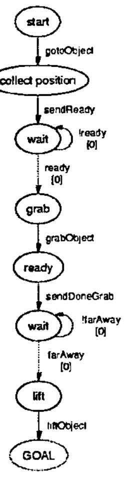

Fig. 3. Plan for the collector

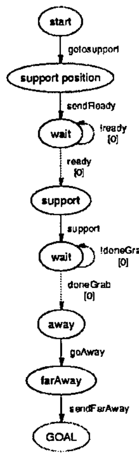

Fig. 4. Plan for the supporter

V. Conclusions and Future Works

In this paper we presented a distibuted asynchronous algorithm for task assignment in dynamic environment. The presented approach is based on token passing for role allocation and successfully achieves the integration of the task assignmnent approach with object perception form the environment. The approach is able to detect and solve conflicts in role execution and to provide correct allocation for constrained roles.

Experiments performed on the simulated environment show that the method succesfully allocate tasks to agents in different operative conditions while maintaning a very low communication overhead. Experiments performed on real robots qualitatively show that the task assign is able to effectively assign tasks to robot in our reference scenario, while avoiding conflicts among team members. The solutions we adopted to implement the described coordination method on the sony AIBO robots are well suited for the referencing scenario.

As future work a interesting extension would be to dynamically change the number of robots involved in the task allocation process and to optimize the allocation process deciding whether to accept tokens and whom to pass tokens based on probabilistic models of team members.

References

[1] C. Castelpietra, L. Iocchi, D. Nardi, M. Piaggio, A. Scalzo, and A. Sgorbissa. Coordination among heterogenous robotic soccer players. In *Proc. of the Int. Conf. on Intelligent Robots and Systems (IROS'00)*, pages 1385–1390, 2000.

[2] F. Cottefoglie, A. Farinelli, L. Iocchi, and D. Nardi. Dynamic token generation for constrained tasks in a multi-robot system. In *International Conference on Systems, Man and Cybernetics*, The Hague,The Netherlands (to appear), 2004.

[3] A. Farinelli, L. Iocchi, and D. Nardi. Multi robot systems: A classification based on coordination. In *IEEE Transactions on System Man and Cybernetics, part B (to appear)*, 2004.

[4] B. Gerkey and J. M. Matarić. Multi-robot task allocation: Analyzing the complexity and optimality of key architectures. In *Proc. of the Int. Conf. on Robotics and Automation (ICRA'03)*, Taipei, Taiwan, Sep 14 - 19 2003.

[5] B. Gerkey and M. J. Matarić. Principled communication for dynamic multi-robot task allocation. In *Proceedings of the Int. Symposium on Experimental Robotics*, Waikiki, Hawaii, December 2000.

[6] Roger Mailler, Victor Lesser, and Bryan Horling. Cooperative negotiation for soft real-time distributed resource allocation. In *Proceedings of AAMAS'03*, 2003.

[7] R. Nair, T. Ito, M. Tambe, and S. Marsella. Task allocation in robocup rescue simulation domain. In *Proceedings of the International Symposium on RoboCup*, 2002.

[8] L. E. Parker. ALLIANCE: An architecture for fault tolerant multi-robot cooperation. *IEEE Transactions on Robotics and Automation*, 14(2):220–240, April 1998.

[9] P. Scerri, A. Farinelli, S. Okamoto, and M. Tambe. Token approach for role allocation in extreme teams: analysis and experimental evaluation. In *13th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE-2004).*, Modena, Italy, 2004.

[10] D. Shmoys and E. Tardos. An approximation algorithm for the generalized assignment problem. *Mathematical Programming*, 62:461–474, 1993.

[11] Robocup Rescue Legged Size League Web Site. http://www.openr.org/robocup/.

[12] B. B. Werger and M. J. Mataric. Broadcast of local eligibility for multi-target observation. In *DARS00*, pages 347–356, 2000.

[13] R. Zlot, A Stenz, M. B. Dias, and S. Thayer. Multi robot exploration controlled by a market economy. In *Proc. of the Int. Conf. on Robotics and Automation (ICRA'02)*, pages 3016–3023, Washington DC, May 2002.