Simultaneous Task Subdivision and Assignment in the FRACTAL Multi-robot System

C. Rossi L. Aldama. A. Barrientos

Grupo de Robtica y Cibernetica, Universidad Politcnica de Madrid, Madrid, Spain (Tel: +34-91.336.3061; e-mail: Claudio.Rossi@upm.es)

Abstract: This paper presents a negotiation protocol for simultaneous task subdivision and assignment in a heterogeneous multi-robot system. The protocol is based on an abstraction of the concept of task that allows it to be applied independently on the actual task, and adopts Rubinsteins's alternate offers protocol extended with a co-evolutionary step in search for the best (counter)-offer. The protocol has been tested on computer simulated application scenarios.

1. INTRODUCTION

Multi-robot systems (MRS) are a very active field of research. A variety of techniques have been proposed in order to approach the problem of coordination in different kinds of applications [Parker, 2003]. Cooperation applications can be roughly divided in two classes: tight cooperation requires a continuous coordination between the robots, like for instance in box pushing and formation keeping. Loose cooperation requires coordination at the beginning of the mission for planning a division of labour, and at given moments of times, when re-planning may be needed. Exploration and mapping are typical applications. Behaviour-based ?] and schemas [Arkin, 1992] are examples of techniques suitable for the first class of coordination problems, while market-based [Dias and Stentz, 2003] and auction [Gerkey and Mataric, 2002] techniques are commonly used in the second class of problems.

Here, we focus in loose cooperation. In this class of problems, a given task has to be partitioned in sub-tasks, and sub-tasks have to be assigned to individual team-members for being executed. Most of the coordination techniques assume that the task subdivision step is done at a high level, and focus on the sub-task allocation problem. Alternatively, an element of the team is given the main task and is responsible of partitioning it, and then the sub-task assignment step is performed. This approach, although applied with success in many applications, has two principal drawbacks: first, it is not really distributed, since the task partitioning is done in one place (either a command and control station or a specific team-member) and second, the partitioning algorithm is usually considered outside the coordination protocol. Often, the details of how the original task is partitioned are not given at all. Moreover, the robots preferences and limitation are considered only in the assignment stage, when the robots decide whether to accept (or opt for) a task or not.

Such features do not suit our need of a fully distributed approach that should consider robots capabilities already at the task partitioning stage. We have developed a new negotiation protocol that performs a simultaneous task subdivision and allocation, taking into account robots preferences.

Negotiations have been widely studied in the context of socio-economic studies [Chatterjee and Samuelson, 1987] using, amongst others, Game Theory [Osborne and Rubinstein, 1994]. An example of recent application is electronic commerce using agents [Sugasaka et al., 2000]. The main problem with game theory approaches is that the theoretical results obtained refer to very simplified models that are not immediately applicable to complex applications. The protocol we propose is based of Rubinsteins alternate-offers protocol [Rubinstein, 1983]. Since such protocol is based on a uni-dimensional good, a search mechanism for the best (counter)-offer had to be devised for the protocol to be applied in real multi-dimensional tasks. To the best of the authors knowledge, the only similar approach has been proposed in Soo and Wu [2000] and Chen et al. [2002], where a co-evolutionary genetic algorithm is used to negotiate the payoff matrix of a coordination game (using a trusted third party), and then find an optimal agreement between the parts reasoning on the matrix.

In this paper we will use the terms robot, vehicle and agent as synonymous, as this makes no difference for the scope of the discussion.

In the following, we will first present our definition of tasks and how agents take into account costs and rewards to evaluate (sub-)tasks. Section 3 briefly describes Rubinstein's alternate offers paradigm and the negotiation protocol we have developed based on this, and its extension to the case of more than two negotiators. Finally, in Section 4 we describe the tests we have performed on different instantiations of tasks and analyze the results.

2. TASKS

In order to design a negotiation algorithm that is general enough to work with different kind of tasks (cf. Fig. 1), an abstract task concept should be defined. A negotiation algorithm based on such an abstraction allows different applications with minimal changes. First of all, let us clarify that, in the context of loose cooperation, with task

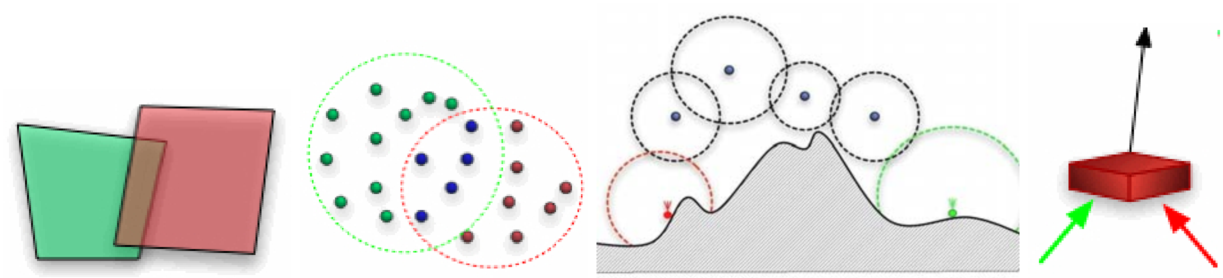

Fig. 1. Examples of instantiation of tasks: Areas, foraging, communication ranges and Box Pushing. In the box pushing, tasks are force vectors. Their union is the projection of the resulting vector in the desired direction of motion, and the intersection its projection in the orthogonal direction.

we mean the object to be divided, and not the activity to be performed on such object. For example, if the task is surveying a given area, we are mainly interested in partitioning the area and assigning sub-areas to the agents.

Of course, the activity the agents will have to perform and their preferences (for instance w.r.t. their capabilities) have an important role in the negotiation. This role is encapsulated in the cost/reward the agents associate to the task (see e.g. Fig.4, where two agents give different values to a given area).

We define a task T as an element of a set $\mathbf{T}$, $T \in \mathbf{T}$. An element of $\mathbf{T}$ is described by a set of k parameters $x \in \mathbb{P}_1^{k_1} \times \ldots \times \mathbb{P}_h^{k_h}, \sum_i k_i = k$, and $\mathbb{P}_i, i = 1 \ldots h$, being parameters types. Without loss of generality we can assume they are all of the same type, as in most practical cases x will be an array of real numbers: $x \in \mathbb{R}^k$. Then we can write $T = T(x)$, that is, we consider that a task T is the product of a function that maps a set of parameters into a task: $T : \mathbb{P}^k \to \mathbf{T}$. A task T has to be divided in R subtasks: $T(x) = \{T_1, \ldots, T_R\}$. Each subtask $T_i, i = 1..R$, can in turn be described by a set of parameters x_i:

$$T(x) = \{T_1(x_1), \ldots, T_R(x_R)\}$$

In general a good subdivision is such that there is minimum overlapping between sub-tasks (ideally null), and such that the subtasks cover the original task. That is,

$$T_i \cap T_j = \oslash, \qquad \forall i, j = 1 \ldots R \qquad and \qquad \bigcup_{i=1}^{R} T_i = T$$

where the operators $\cap, \cup : \mathbf{T} \times \mathbf{T} \to \mathbf{T}$ are to be defined according to the meaning of the task. Note that there can be exceptions, depending on the application. For example, in a communication relay application, the overlapping between the range of action of two robots must not be null. Let $g : \mathbf{T} \to \mathbb{R}$ be a reward function, giving the value of a (sub)task. Then, the function

$$f : \mathbb{P}^k \to \mathbb{R} = T \circ g$$

associates a reward to a set of parameters describing a task. We associate to a subdivision $T = \{T_1, \ldots, T_R\}$ an index called global coverage G, that takes into account the total coverage of the subtasks and their pair wise overlapping.

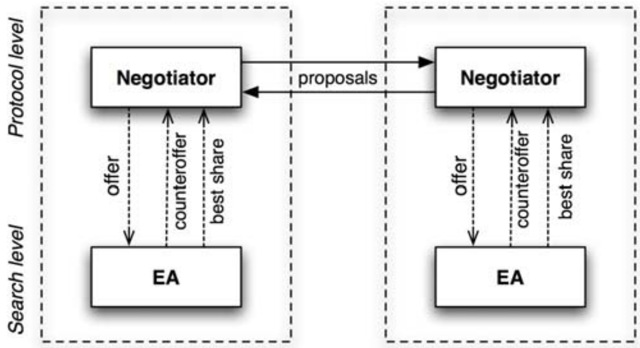

Fig. 2. Architecture of the negotiation module. The search level is implemented with an Evolutionary Algorithm.

$$G = \sum_{r=1}^{R} f(x_r) - \frac{\sum_i \sum_{j \neq i} g(T(x_i) \cap T(x_j))}{2}$$

Then, the problem of task subdivision can be formulated in the following way:

Given a task T and a number R of agents, find the R sets of parameters x_i, $i = 1 \ldots R$, such that G is maximized:

$$max_{x_1 \ldots x_r}(G).$$

Note that G is a global performance index. During the negotiation, each robot will give a different value to the same task, depending on its characteristics (locomotion, sensors, status, etc.). In other words, each robot uses its own reward function g_i to evaluate a task. To this aim, g_i takes into account its internal parameters to evaluate the cost of executing the task, the start-up cost (for instance, to reach the execution site), specific constraints (e.g. forbidden zones, turn angles, sensors), penalty for exceeding task limits (i.e. $T_i \backslash T$) and the reward associated to the task, expressed as function g.

3. TASK NEGOTIATIONS

A given task T can be executed by a team of R robots, after a suitable subdivision of the task has been performed, and an assignment of the subtasks to the robots have been established. Our aim is to perform these two actions simultaneously and in a distributed way. In our system, the number of sub-tasks is determined by the number of robots

```
Receive negotiation request and task info
Calculate first counteroffer (=maximum)
Send ok to negotiate
LOOP
  Receive offer
  Estimate other agent's  delta
  Estimate max possible share
  Search for best share
  IF(best share < estimated max possible share)
     Update counteroffer
     Send counteroffer
  ELSE
     Accept and exit loop
END LOOP
```

Fig. 3. Negotiation protocol. The first counteroffer is the maximum possible for robot i. The counteroffer update tries to produce a value that is close to a factor of δ_i smaller than the previous.

willing to participate to the negotiation. Let us first discuss the case $R = 2$. We assume that robots are not lazy, in the sense that they are willing to perform as much as they can of the given task (hence maximizing their reward), the only limitations being their available resources (endurance, computation power, battery consumption etc.). Thus, in a negotiation, each agent will try to maximise its reward by (i) trying to get an as big as possible subtask and (ii) minimizing overlapping with other agents task. Each agent starts proposing the biggest possible share for itself, and reduces it until the counterpart finds it acceptable. In this way a good near-optimal solution, although not optimum in general, can be achieved.

Global index G is optimized in a distributed way, without even being computed explicitly (see Fig. 8).

In the alternate-offers protocol proposed by Rubinstein, each part of the bilateral negotiation, in turn, propose a subdivision of a uni-dimensional good of size 1. The responder can agree with the subdivision, or disagree with it, and in this case it has to propose a counteroffer. The protocol assumes that each part has a target (desired) reward, and a negotiation cost, that makes the target reward decrease at each step, imposing a time pressure to the reaching of an agreement. Such protocol has interesting theoretical properties. By applying discount factors as negotiation cost, it guarantees a termination and can forecast the final agreement, which will be a perfect equilibrium in the sense of game theory. Let the discount factors be $0 < \delta_1 \leq 1$ and $0 < \delta_2 \leq 1$, and let $y_{t+1} = y_t \cdot \delta, t = 1 \dots n$ be the update rule of target share y. Rubinstein's theory guarantees that one perfect equilibrium point exists such that the share y the initiator agent will get is:

$$y = \frac{1 - \delta_2}{1 - \delta_1 \delta_2}.$$

If the discount factors were known to both parts, each could know *without negotiating at all* which will be its share. However, these are not immediately applicable to the multi-dimensional case. In the uni-dimensional case, when an agent makes a proposal p of what it would like to get, it is immediate that the other would get $1 - p$. In the multidimensional case, given a proposal $x \in \mathbb{P}^k$ on the whole task T_0, an agent shall search the space $T_0 \backslash T(x)$ to decide if the share it would get is acceptable and to generate a counteroffer, since many different configurations are possible.

Thus, we divide the negotiation in two levels: the protocol level and the proposals evaluation and generation level (see Fig. 2). The protocol level is governed by parameters such as impatience to reach an agreement (time pressure as discount factor) and desired target reward. Moreover, at each new offer received, it estimates the other agents discount factor δ in order to estimate the maximum possible share it can expect, according to Rubinstein's theory, and update its desired share accordingly. Supposing agent 1 is the first who makes a proposal, it can estimates $\hat{\delta_2}$ and forecast its share will be

$$\hat{y} = \frac{1 - \hat{\delta_2}}{1 - \delta_1 \hat{\delta_2}}. \quad (1)$$

The proposal generation level searches the space for a good share given a proposal from the other part, taking into account its own resources, parameters and limitations. This level is also responsible for updating counteroffers. In fact, in a multi-dimensional space there are may ways offers can be updated. The update function aims at reducing the offer in such a way to reduce the overlapping. In order to comply with Rubinsteins' theory hypothesis, the new offer should have a dimension $dim(T^t) = \delta \cdot dim(T^{t-1})$. Currently, the search step is performed by an Evolutionary Algorithm, which adopts a specialized mutation operator that generate solutions that comply with this constraint within a given tolerance. Since we have two Evolutionary Algorithms each searching a space that is changed by the solution of the other one, we talk of co-evolution. However, other search methods could be applied.

When an agreement has been reached, the result is a subdivision of the original task and at the same time an assignment of the sub-tasks. Note that in this way one agent does not need to know information about the private characteristics of the team-mates. The only information it needs is their offers, in form of an array of parameters $x \in \mathbb{P}^k$.

In case all agents desire a similar amount of work w, and this is known a priori (e.g., divide the task equally between all participants), the value of w can be used directly instead of $\hat{y}$, and the agents will end the negotiation when such value is reached.

3.1 Extending to more than two negotiators

When there are more than two agents, the proposed negotiation protocol can be extended in several ways. It is important to point out that the difference between two and three negotiators reflects a basic qualitative difference between the types of processes that take place within the negotiation. The involvement of more the three parties can be seen as an extension of a three-party process [Caplow, 1968]. Hence, let us focus on the case $R = 3$.

The simplest extension is to do negotiation rounds where each agent, in turn, makes its proposal. At the end of the round, if all agents are satisfied the negotiation is closed, otherwise another round starts. The problem in this case is the estimation of the expected maximum share $\hat{y}$ (Eq.

Fig. 4. Negotiation with forbidden area (shaded area). Black agent, an aerial vehicle, refuses all proposals including the no-fly zone.

1), which determines the termination of the negotiation. During rounds, each agent can estimate the discount factor of each of the other agents, but there is no straightforward way to combine such discount factors as in Eq. 1. Simply applying Eq. 1 to both the opponent and averaging the results to obtain the estimated share would not work as the sum of all estimations is not guaranteed do be equal to 1. In the current implementation of the protocol, we are adopting a simple heuristic rule derived by the analysis of the outcome during the early testing of the algorithm.

$$\hat{y_j} = \frac{1}{n}\sum_{i\neq j}\left(\frac{1-\hat{\delta}_i}{1-\delta_j\hat{\delta}_i}\right) \qquad j = 1\ldots n. \tag{2}$$

We are currently studying the theoretical properties of the model extended to three parties to formulate more grounded estimations.

4. EXPERIMENTAL RESULTS

In order to assess the effectiveness of the proposed protocol, we have performed several simulations using the robotics tool Webots[Michel, 2007].

Figure 4 show how an hard constraint (a forbidden area) makes the agents agree on a subdivision that exclude such area from the area to be surveyed by the constrained vehicle. The subdivision of an area among three agents is depicted in Figure 5. The figures have been post-processed to enhance visibility in black and white printing.

Table 1 shows that negotiations only take tho order of seconds to conclude and that the global performance index G is optimized. Each figure is the average of 10 runs. Note that not always the optimum value for G is reached, but this is due to time pressure: agents are forced to reach an agreement soon. In applications where more precision is needed, this can be reached by imposing a lower time

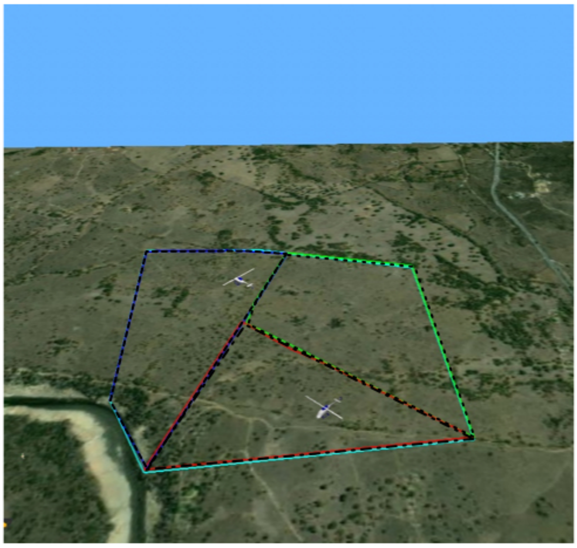

Fig. 5. Negotiation with three agents using rounds.

pressure. In the communications relay experiments (Fig. 6), $dim(T)$ is the distance between the two points to connect, and G is the length of the shortest path between agents obtained, which is longer due to the presence of an obstacle. In this case, the vehicles negotiate positions.

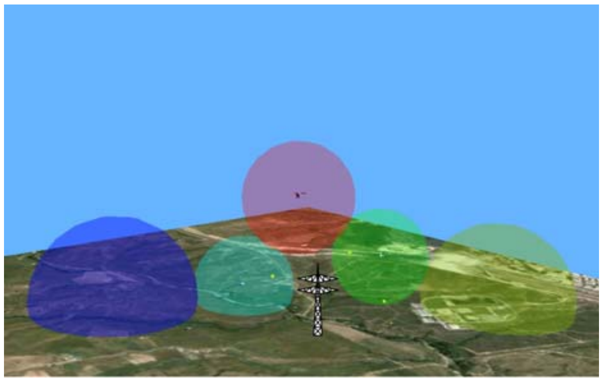

Fig. 6. Communication relays. Three aerial vehicles have to negotiate how to position themselves in order to guarantee communications between two fixed rovers' positions (leftmost and rightmost spheres). Spheres represent signal range. Note the presence of an obstacle in the middle (power tower).

Experiment	*Agents*	*Steps*	*Time (sec.)*	*dim(T)*	*G*
No fly zone	2	28	1.82	0.67	0.66
Area Survey-2	2	53	2.72	1.00	1.02
Area Survey-3	3	82	3.53	1.00	0.99
Comm relay	5	9	0.36	0.90	1.02

Table 1. Summary of experiments. In the "Comm relay" test, two of the agents (source and destination) were constrained to remain in a fixed position.

Figure 8 shows an example of the shares of area partitioning between two agents. The left plot shows how the shares the agents obtain is actually close to values predicted by Rubinstein's theory. The plot on the right shows how the overlapping task is reduced during the negotiation, and

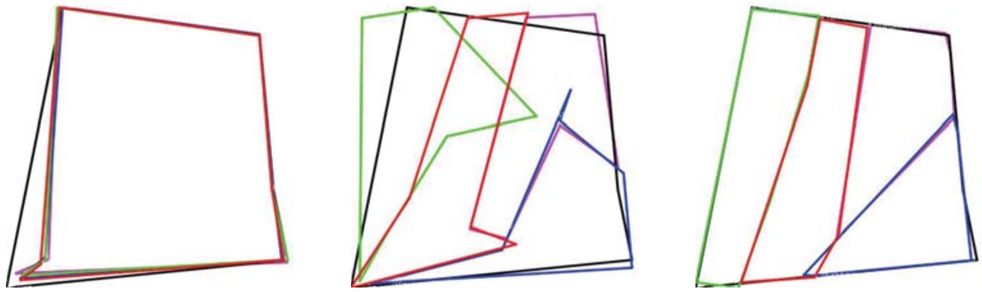

Fig. 7. Detail of negotiation steps between 4 agents: best offers at step 1, 25, 100. The search step move the vertices of the polygons in order to reduce overlapping.

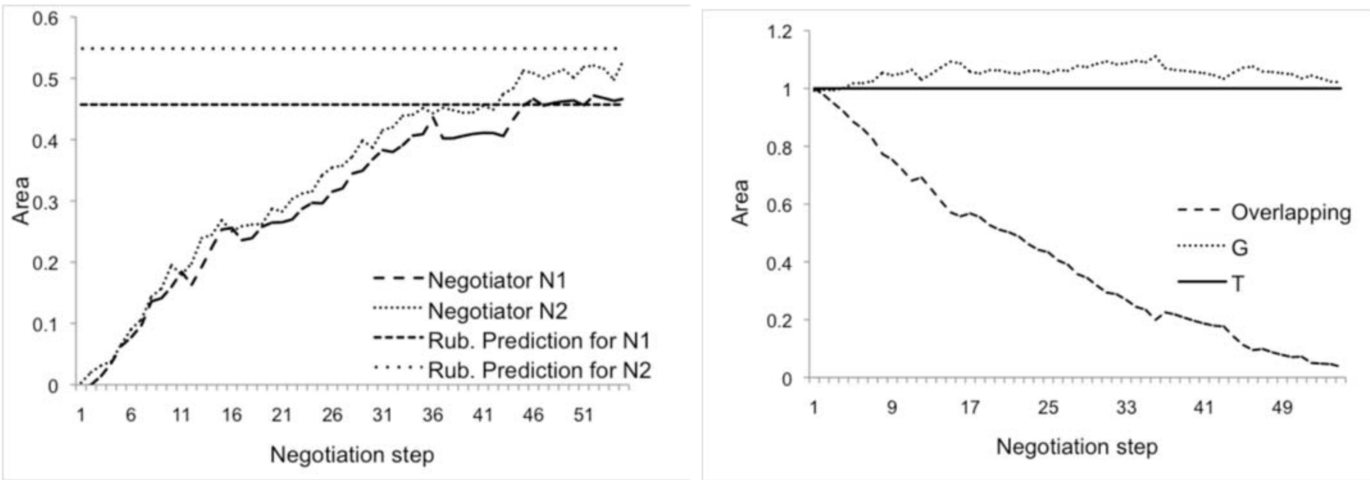

Fig. 8. Negotiation on how to partition an area. Best share and Rubinstein predictions (left). Total area T and global coverage G (right). $\delta_1 = 0.988, \delta_2 = 0.99$, both agents start claiming 1.

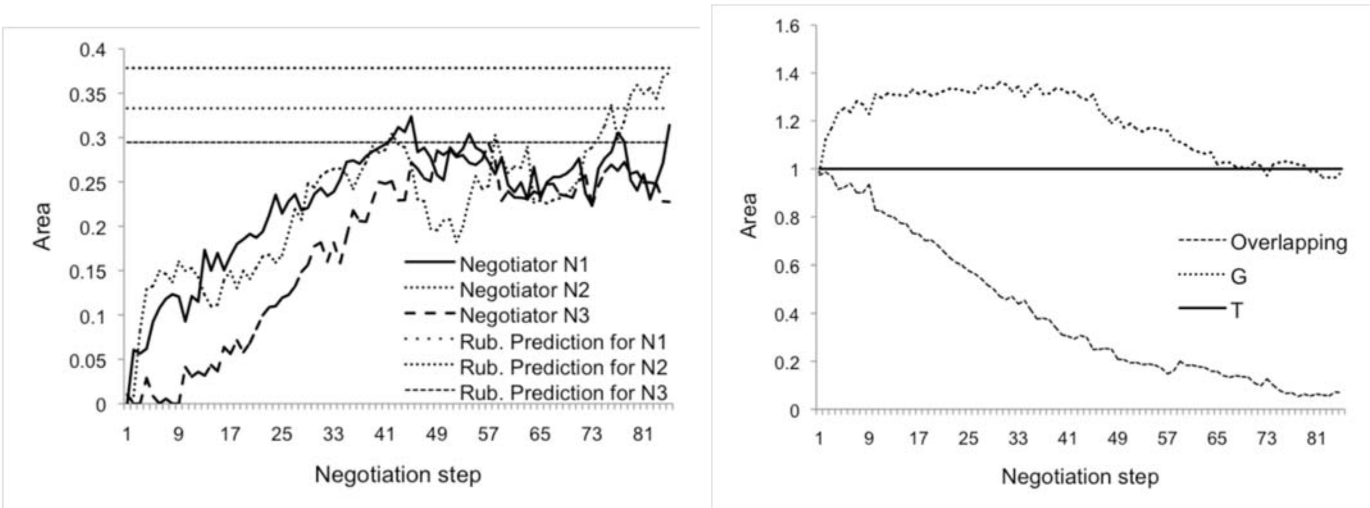

Fig. 9. Example of negotiation with three agents ($\delta_1 = 0.988, \delta_2 = 0.99, \delta_3 = 0.986$). Best share and predictions of Eq. 2 (left). Total area T and global coverage G (right).

how global coverage G is very close to the optimum value (in this, case the total area T). Also, note how global coverage is maintained throughout the negotiation. In other words, the whole area is covered by the two robots at all times. Figure 9 shows similar results for three agents. In the plots, a value of G greater than T means that the current offers cover an area outside the target area. This was allowed in the simulations. This effect can be seen in Figure 7, that shows the detail of the negotiation process between four agents. At the beginning all agents claim the same share. During the negotiations the vertices of the polygons are adapted in order to minimize overlapping, and some of the offers may partially lay outside the target area. Offers can be forced not to lay outside the target task by increasing the penalty factor associated to exceeding the task in the evaluation functions g_i, according to the particular application constraints.

5. CONCLUSION AND FUTURE WORK

The main contributions of this paper are two. First, we propose a formal definition of the concept of task, general enough that allows expressing different problems. The negotiation algorithm implemented using such formulation then applies to a vast variety of multi-robot tasks. Second, a negotiation protocol that takes advantage of theoretical results that guarantee some important properties such as termination and prediction of the outcome.

Experiments conducted in computer simulations show the effectiveness of the proposed approach, and the adherence of the numerical results with the theoretical results coming from game theory.

The main objection that can be done is the need of multiple communications between the agents in order to reach an agreement. This is certainly true with respect to, e.g., the *contract net* protocol [Smih, 1980], which is the base for many protocols that can be found in the literature. However, as mentioned earlier, such protocols assume a task partitioning step that is not taken into account. Such step is mostly centralized and needs information of the full status of the system, and it may require complex algorithms and more computing power. We may say that in our approach we exchange computing power for communications in order to distribute the task subdivision and assignment problem amongst the team members, with the advantage that complete information of the whole system's status is not needed. Thus, it can be concluded that our method is more suitable when a central node with enough power is not available or recommended (for instance, in security applications) and communications can be guaranteed at least during the negotiation process. On the other hand, contract net-based protocol are to be preferred when communications cannot be guaranteed, and if amongst the team members there is a team leader (perhaps a command and control station) that can take the charge of finding a good task splitting and manage the auction.

We are currently working on two fronts. On the practical side, we will deploy the negotiation algorithms on our fleet of aerial and ground autonomous vehicles for testing with real robots, and perform tests with different kind of tasks. Although at the moment all experimentation has been done on simulation, we believe that the deployment of the algorithms on real robots will not influence in an important way the general idea on the negotiation protocol we propose, although some tuning of the penalty and cost parameters may be needed to adjust the algorithm to the real performances of the robots.

As far as the theoretical aspect is concerned, we are investigating the properties of the multiple-rounds extension, in order to have a more grounded estimation of the expected shares on the basis of the estimation of the opponents discount factors in the case $R > 2$.

ACKNOWLEDGEMENTS

The first author acknowledges a Ramon y Cajal research fellowship of the Ministerio de Educacon y Ciencia (MEC) of Spain. This work is funded by project FRACTAL (Fleet of autonomous aerial and ground robots - DPI2006-03444) of the MEC.

REFERENCES

R. C. Arkin. Cooperation without communication: Multiagent schema-based robot navigation. *Journal of Robotic Systems*, 1992.

T. Caplow. *Two Against One: Coalitions in Triads.* Prentice-Hall, Englewood Cliffs, S.J., 1968.

K. Chatterjee and L. Samuelson. Bargaining with two-sided incomplete information: An infinite horizon model with alternating offers. *Review of Economic Studies*, 54, 1987.

J-H. Chen, K-M. Chao, N. Godwin, C. Reeves, and P. Smith. An automated negotiation mechanism based on co-evolution and game theory. In *Proc. of the 2002 ACM symposium on Applied*, 2002.

M Bernardine Dias and Anthony (Tony) Stentz. Traderbots: A market-based approach for resource, role, and task allocation in multirobot coordination. Technical Report CMU-RI -TR-03-19, Robotics Institute, Pittsburgh, PA, August 2003.

B. P. Gerkey and M. J. Mataric. Sold!: Auction methods for multirobot coordination. *Trans. on Robotics and Automation*, 18:5, 2002.

O. Michel. Webots: Professional mobile robot simulation. *J.Adv. Robotics Systems*, 1(1), 2007.

M. J. Osborne and A. Rubinstein. *A course in game theory.* MIT Press, Boston, 1994.

L.E. Parker. Current research in multi-robot systems. *J. Art. Life and Robotics*, 7, 2003.

L.E. Parker. L-alliance: Task-oriented multi-robot learning in behaviour-based systems. *Advanced Robotics*, Special Issue on Selected Papers from IROS'96, 1997.

A. Rubinstein. Perfect equilibrium in a bargaining model. *Econometrica*, 50:1, 1983.

R. G. Smih. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, C-20:12, 1980.

V-W. Soo and S-H. Wu. Negotiation without knowing other agents payoffs in the trusted third-party mediated-game. In *Second workshop on game theoretic and decision theoretic agents*, 2000.

T. Sugasaka, K. Tanaka, R. Masuoka, A. Sato, H Kitajima, and F Maruyama. An agent-based system for electronic commerce using recipes. In *Proc. of the Seventh International Conference on Parallel and Distributed Systems*, 2000.