Multiagent Coordination by Auctioning POMDP Tasks

Matthijs T.J. Spaan
Institute for Systems and Robotics
Instituto Superior Técnico
Lisbon, Portugal
mtjspaan@isr.ist.utl.pt

Nelson Gonçalves
Institute for Systems and Robotics
Instituto Superior Técnico
Lisbon, Portugal
ngoncalves@isr.ist.utl.pt

João Sequeira
Institute for Systems and Robotics
Instituto Superior Técnico
Lisbon, Portugal
jseq@isr.ist.utl.pt

ABSTRACT

We present a procedure for the estimation of bids in auction protocols. This class of protocols is of interest to multiagent systems because they can be used to coordinate the assignment of tasks to agents. The main idea is to take advantage of methods for the synthesis of task execution controllers that rely on the availability of value functions. These provide a natural way to obtain the bid values for a given task. The approach is demonstrated on an active surveillance system, where mobile robots must approach and identify humans, and conduct patrols. The Partially Observable Markov Decision Process (POMDP) framework is used to compute policies for the execution of tasks by each agent, the task bid values are obtained directly from the respective value functions. Several simulation examples are presented for an urban surveillance environment, illustrating the applicability of our ideas.

General Terms

Multiagent systems

Keywords

POMDP, Auction Protocol, Task Assignment

1. INTRODUCTION

We present an approach to the estimation of bids in auction protocols. It is based in the value functions obtained from the design of controllers for the execution of tasks by the agents. From these functions, the fitness of an agent to execute a task (given the state of the environment) can be obtained directly, and without extra effort.

The use of auction protocols was initially proposed by [20] for collaborative, distributed problem solving among a set of agents. In multi-robot systems, these protocols are commonly used to determine the assignment of tasks [5]. They are also used in assignment problems arising in areas such as corporate management [2], and game theory [13]. The main advantages of these protocols are their robustness to individual agent failures and the reduced bandwidth requirements [6]. Another advantage is that the assignment solutions are computed in a distributed manner, and thus can be used by agents with low computational resources.

AAMAS 2009 Workshop on Multi-agent Sequential Decision-Making in Uncertain Domains, May 11, 2009, Budapest, Hungary.

A crucial challenge to applying auction protocols is the estimation of the agents' bid values for each task. Agents must evaluate their fitness for executing a task using only locally available information. In mobile robotic applications, tasks often consist in the execution of a path [6, 7, 5]. Therefore, the bid value on each task can be the path distance, the travel time or a combination of these measures [12]. In general, these are heuristic measures that need to be defined for each task, often in an ad-hoc manner. By comparison, the advantage of using task controllers based on value functions is that bid functions need not be tailored for the application at hand. Instead, they already have been computed, and will reflect better true bid values, since they are derived directly from the task controller. In order to implement our task execution controllers using value functions, they are synthesized through a decision-theoretic approach. In particular, we use Partially Observable Markov Decision Processes (POMDPs) [10], which form a general and powerful mathematical basis for planning under uncertainty.

The remainder of this paper is as follows. In Section 2 we present an overview of the proposed approach. The POMDP framework is reviewed in Section 3, and Section 4 describes the auction protocol. In Section 5 the proposed approach for the estimation of bids is presented. Section 6 shows how the approach can be applied to an active surveillance system, and it is evaluated through simulation. Finally, in Section 7 we discuss the paper.

2. MULTIAGENT TASK COORDINATION

The problems considered in this paper are the assignment of tasks and their execution in a multiagent system. The first problem is formulated as the assignment of tasks with unknown arrival order. The agents can execute only one task at a time, which can be interrupted to begin the execution of another one. In this case, the current progress of the interrupted task is lost. Due to communication and hardware failures, the number of available agents is not known *a priori*. The computation of the assignment solution when the order of task arrivals is known and no failures occur is NP-HARD in general, [9, 4]. The available algorithms proposed for this problem often require computational resources organized in a centralized manner.

The second problem is the synthesis of controllers for the execution of each task by the agents. Each has available a finite, and possibly distinct, set of actions and can perceive features of interest in the environment. This problem is then formulated as computing a controller for the execution of a task by an agent. The tasks are assumed to

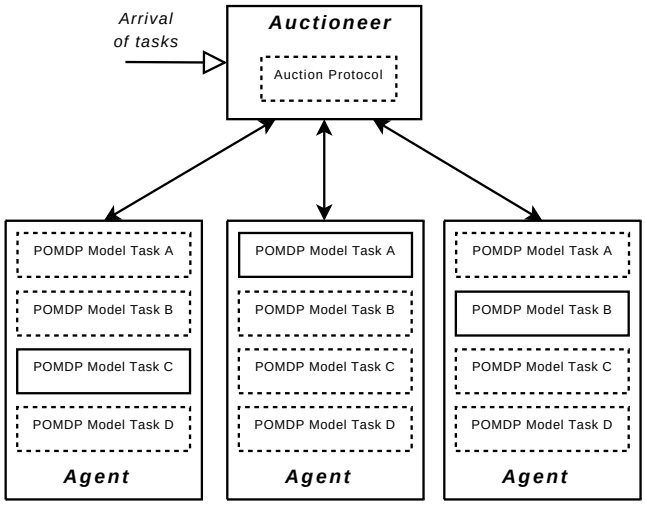

Figure 1: Diagram of proposed solution. Each agent is running a POMDP model for each task in parallel, but only one is active (indicated by a solid box).

be executed by a single agent, without explicit coordination with other agents. In this way, we avoid the severe complexity penalty involved when considering the full joint planning problem (either centralized or decentralized). Coordination is achieved on a task level, by finding the optimal assignment of individual tasks to agents.

A POMDP problem is formulated and solved for each task an agent can execute. The POMDPs at each agent receive the same set of local observations, but between agents no beliefs or other types of information are shared. When an agent is assigned a task, the policy of the corresponding POMDP is enabled and the others disabled. That is, the actions executed by the agent are those determined only by the policy of the assigned task POMDP.

The proposed approach in this paper is represented in the block diagram of Figure 2. It is composed by a central supervisor, denoted the *auctioneer*, and a set of heterogeneous agents. The tasks to be executed by the agent are received by the auctioneer and are then assigned through an auction protocol. Although the solution to the computation of task assignments is centralized, these can arrive in any order and the computational complexity is polynomial. Also, it is robust to communication and individual agent failures. The disadvantage is that in general, an optimal assignment solution is not guaranteed.

3. POMDP BACKGROUND

We will discuss POMDP models and solutions, briefly introducing some general background. A more elaborate POMDP model description is provided by [10], for instance.

A POMDP models the interaction of an agent with a stochastic and partially observable environment, and it provides a rich mathematical framework for acting optimally in such environments. The framework is based on the assumptions that at any time step the environment is in a state $s \in S$ and the action $a \in A$ is taken by the agent. As a result of this action, a reward $r(s,a)$ signal is received by the agent from the environment. And the environment state is changed to the new state s', in accordance to a known stochastic transition model $p(s'|s,a)$. The task of an agent is defined by the reward it is given at each time step. The agent task goal is to maximize the long-term reward signals received. After the environment transition to the new state, an observation $o \in O$ is perceived by the agent. This is conditional on the current environment state, and possibly the action executed, according to a known stochastic observation model $p(o|s',a)$.

Given the transition and observation models, the POMDP can be transformed to a belief-state MDP, where the all past information of the agent is summarized using a belief vector $b(s)$. It represents a probability distribution over S, from which a Markovian signal can be derived for the planning of actions. The initial state of the system is drawn from the initial belief b_0, which is typically included in the POMDP problem formulation. Every time an action a is taken by the agent and observation o is obtained, the agent belief is updated by Bayes' rule; for the discrete case:

$$b_a^o(s') = \frac{p(o|s',a)}{p(o|a,b)} \sum_{s \in S} p(s'|s,a) b(s), \qquad (1)$$

where $p(o|a,b) = \sum_{s' \in S} p(o|s',a) \sum_{s \in S} p(s'|s,a) b(s)$, is a normalizing constant.

In POMDP literature, a plan is called a policy $\pi(b)$ and maps beliefs to actions. The policy can then be used to select the action the agent must execute in order to achieve the task goal. A policy π can be characterized by a value function V^π which is defined as the expected future discounted reward $V^\pi(b)$ the agent can gather by following π starting from belief b:

$$V^\pi(b) = E_\pi \Big[\sum_{t=0}^{h} \gamma^t r(b_t, \pi(b_t)) \Big| b_0 = b \Big], \qquad (2)$$

where $r(b_t, \pi(b_t)) = \sum_{s \in S} r(s, \pi(b_t)) b_t(s)$ following the POMDP model as defined before, h is the planning horizon, and γ is a discount rate, $0 \leq \gamma \leq 1$.

The process of solving POMDPs optimally is hard, and thus algorithms that compute approximate solutions are used. There has been much progress in approximate POMDP solving, see for instance [8, 21] and references therein. Furthermore, if a value function has been computed off-line, the on-line execution of the policy it implements does not require much computational requirements.

4. AUCTION PROTOCOL

The purpose of the auction protocol is to determine the POMDP policy that each agent must execute. This is equivalent, in the context of this paper, to the assignment of tasks to agents. The task generation process is assumed to be exogenous to the multiagent system. For instance, the execution of a task can be triggered by the occurrence of an event. The tasks arrive at the auctioneer at any time instant, but are assigned in a bulk manner at regular intervals. Note that we can also start a round of task assignment on demand, for instance when a high-priority task arrives. Also, a task can be scheduled to be executed at periodic time intervals, such as battery recharge operations.

The task execution requests could also originate from some of the agents or the auctioneer. As an example, the auction-

eer may directly receive event messages and locally favor the assignment of some tasks over others. The priority of each task is dictated by the specific application.

In order to solve the task assignment problem, the auctioneer is only required to know the expected discounted reward values of the POMDP task solutions from each agent, given their individual beliefs. The auction protocol is designed as follows, requesting this information from each agent.

DEFINITION 1 (AUCTION PROTOCOL). *The auction protocol is as follows:*

1) *All of the tasks are announced to the agents by the auctioneer.*
2) *The agents reply with their current expected discount reward $V^{\pi}(b)$ for each task. Hence, this is obtained from the solution V^{π} for the task's POMDP model, and the agent's current belief b.*
3) *The assignment solution is computed by the auctioneer and announced to the agents.*

The assignment solution is determined by solving a mixed integer-linear program (MILP). The number of tasks waiting to be assigned by the auctioneer, at a particular instant, is represented by w. The number of agents that replied with bids is n. The assignment of the tasks to the agents is represented by the matrix Y. The element y_{ij} is one if the i-th agent is assigned the j-th task, and zero otherwise. The optimal assignment solution, Y^*, is then determined from the MILP:

$$\begin{aligned} &\max \sum_i \sum_j \beta_{ij} \cdot y_{ij} \\ &s.t. \\ &\sum_i y_{ij} \leq 1, \quad j = 1, \ldots, w \\ &\sum_j y_{ij} \leq 1, \quad i = 1, \ldots, n \\ &y_{ij} \in \{0, 1\} \end{aligned} \tag{3}$$

where β_{ij} is the value of the bid received from the i-th agent for the execution of the j-th task. The problem constraints state that each task is assigned to at most one agent and vice-verse. Thus, if their numbers are not equal, that is $w \neq n$, some tasks are not assigned or some agents remain idle. This can also occur in heterogeneous multiagent systems, where some agents may not be able to execute some of the tasks. In this case, the agents reply with an arbitrary negative value. From the problem formulation, it is clear that if β_{ij} is negative then the i-th agent is not assigned the j-th task. This is because there is at least one feasible solution without this assignment and with a greater value for the cost functional.

The main advantage of this protocol is that the auctioneer is not required to known the number of available agents or their beliefs. Therefore, the approach is robust to the failure of agents or temporary network shortages. Because if an agent cannot be contacted, the others are still assigned tasks. Also, the communication and computational resources are reduced since only the current expected discounted reward must be reported to the auctioneer. Finally, the coordination of the agents is obtained implicitly through the auction protocol.

Although the assignment problem is a MILP, it can be efficiently solved in polynomial time using the Hungarian algorithm [3]. Therefore, this approach can be applied to small and medium sized problems with tens or hundreds of agents and tasks. In contrast, the auction protocols described in [5] often exhibit exponential complexity. The reason is that in these protocols, agents bid on bundles of tasks instead of the single task case of our protocol. Although the computational complexity is greatly reduced, the solution is only locally optimal. In [11] it was shown that for bundles with a small number of tasks, the assignment solution quality is improved without significantly increasing the computational and communication costs. Nevertheless the problem of computing the bid value is not considered and the tasks to be executed are known in advance. This is not the case in this paper, where the tasks and their arrival order are not known in advance. Also, it is assumed that agents do not accurately know their state. As a result, the estimation of bids for future tasks is complicated by the uncertainty at the current state.

5. POMDPS FOR BID ESTIMATION

In this work, we assume that the agents do not share any information among them. The reason is to reduce the network bandwidth and computational requirements, since the POMDP instances are smaller. It is known that relying on perfect communication can reduce the decentralized planning to a centralized one [17], but the size of the centralized problem still grows exponentially in the number of agents. Another reason is that since the agents are assumed not to be required to coordinate their individual actions in order to execute tasks, their POMDP models in general need not account for other agents. It must be stressed that the execution of tasks could benefit from knowledge on the other agents' beliefs and actions. For instance, if the planned paths of two mobile robots intersect, the collision could be avoided if their beliefs were shared. In order to avoid such potentially dangerous cases, low-level safety controllers are assumed to be available for the execution of actions.

Since multiple independent decision makers are present in the environment, the problem could be modeled as a decentralized POMDP (Dec-POMDP) [1, 19, 14]. However, given their very high complexity class, current algorithms do not scale to the types of applications we are focusing on. In our case, the coordination of the agents is obtained implicitly through the auction protocol and the auctioneer; coordination is considered on the level of task assignments vs. the level of individual agent actions, as is common in Dec-POMDPs.

A POMDP model of a certain task provides both an implementation of the task, i.e., how the agent should act to accomplish the task, as well as a valuation of the agent's fitness to execute it. The latter depends on the state of the environment, and in the POMDP case, on the agent's belief state. For each task, the reward model is such that when the agent accomplishes the goal, for instance reaching an area where to patrol, it receives a single reward of 10. When reaching the goal, the agent is transferred to an absorbing state, in which it receives zero reward, and it only leaves the absorbing state when a new task has been assigned. The use of an absorbing state is crucial, because otherwise the POMDP values can keep on rising, by instructing the agent to remain in a goal state. Although this effect might not

influence the policy implemented by the value function, inflated values are not desirable for our approach, given that we compare values between different POMDPs.

As discussed, in order to evaluate the relative benefit of using agent x over agent y, their valuations should be in the same range. For this reason we employ the same maximum reward in each POMDP model, where the values of each are normalized to [0, 10]. However, we need to be able to express different priorities for tasks, in order to ensure that more important tasks get assigned first; for instance, when there are more tasks than agents available. This is accounted for by multiplying each the bid values by the task priority. Since the bid values are normalized, the result is that each bid is weighted by the respective task priority.

Note that instead of POMDPs also other planning models can be used, as long as they involve computing a value function. An example is the ALLIANCE control architecture [15], where the agent impatience and acquiescence levels determine the task to be executed. Assume that the sum of both levels is equal to some constant. Then the value function can be identified with the agent impatience and the acquiescence with the complementary value.

6. ACTIVE SURVEILLANCE SYSTEM

The presented approach is applied, in simulation, to an active surveillance system. It is composed by a set of mobile robots, an auctioneer and a network of cameras. These are capable of detecting, with some uncertainty, the location in the environment of robots and humans. Upon the detection of a human by the cameras, the auctioneer is notified. Note that here we present a simplified scenario, which can be extended easily to include more events (with different priorities), for instance the detection of fires.

6.1 Experimental setup

The robots have available on-board cameras, which can recognize humans, also with some uncertainty. Each robot can obtain its localization in the environment directly from the camera network. In addition, the robots' on-board power supply is limited and must be recharged after some time has elapsed. The tasks the mobile robots can execute are thus: (i) identification of humans, (ii) meeting a person, (iii) patrol the environment and (iv) recharge their on-board batteries. The first two tasks are assigned only when a person detection event has occurred. In these tasks the robot must approach the desired location and use the on-board sensors either to identify a human or meet it, in order to engage in human-robot interaction. The last two task types are assigned at regular intervals and have a low priority with respect to the first two. In this manner, if no events occur mobile robots can conduct patrols or recharge their batteries.

A set of four robots were simulated (as a unicycle), three modeled after a Pioneer 3-AT robot (indicated by Pioneer A, B and C), and one after an AtrvJr robot. In our current setup, the difference between the Pioneers and the AtrvJr is their maximum speed, which is $1.0\frac{m}{s}$ for the AtrvJr and $0.4\frac{m}{s}$ for the Pioneers. In addition, the camera of Pioneer A had a higher resolution than the cameras on-board the other robots. As a result, this robot could observe a location in the environment from a greater distance than the others.

A topological map of the active surveillance environment is represented in Figure 2. It was obtained from the test site of the URUS project [18], at the UPC campus in Barcelona, Spain. The overall dimensions of the map are 100 by 100 meters, as represented in the figure. The environment was partitioned in smaller regions, with the center of each represented in the map. The tasks are defined as navigation actions, defined using the region centers as way-points. In such a topological map, from each node a robot can only move to nodes connected to it by edges, representing the topology of the environment.

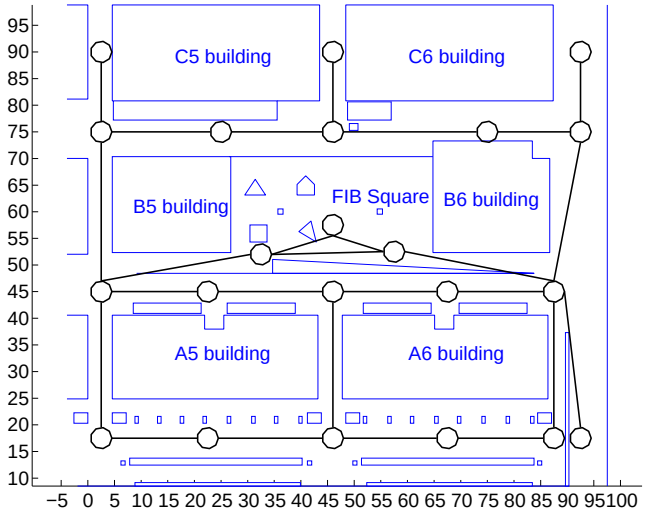

Figure 2: Topological map of the active surveillance environment.

Task	State variables
Patrol SouthWest	Robot position
Patrol NorthWest	Robot position
Patrol NorthEast	Robot position
Patrol SouthEast	Robot position
Meet Person	Robot position, person position
Identify Person	Robot position, person position
Recharge	Robot position, battery level

Table 1: State variables used by different tasks. Positions are represented by nodes in the topological map; battery level consists of four levels, ranging from "high" to "very low".

Each of the tasks mentioned in the previous section have been modeled and approximately solved a POMDP, using Symbolic Perseus [16]. The POMDP models are represented using two-stage dynamic Bayesian networks, and the software allows for exploiting (context-specific) independence between state variables. Table 1 lists the different state variables for each task. We assume the surveillance cameras can localize each robot, but with a particular uncertainty. Also each robot's movement actions are subject to noise. As we are essentially considering long-term plans, the discount rate is set high, $\gamma = 0.99$. Each movement action is penalized with a reward of -0.1.

6.2 Simulation Results

In a first experiment, all of the robots are initially positioned at the region containing the center node, in location $(46, 45)$, and are requested to execute four patrol tasks, one to each corner of the map. The value functions of each robot

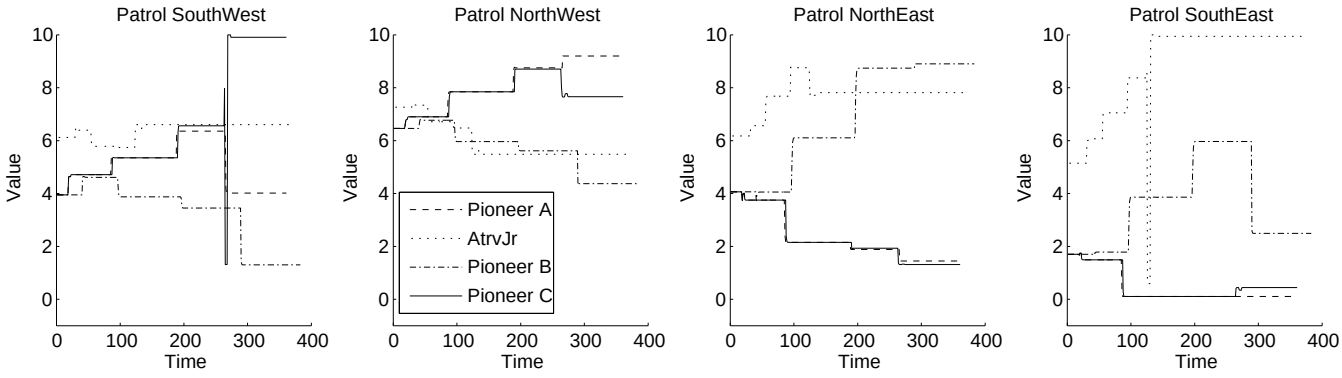

Figure 3: Four different Patrol tasks.

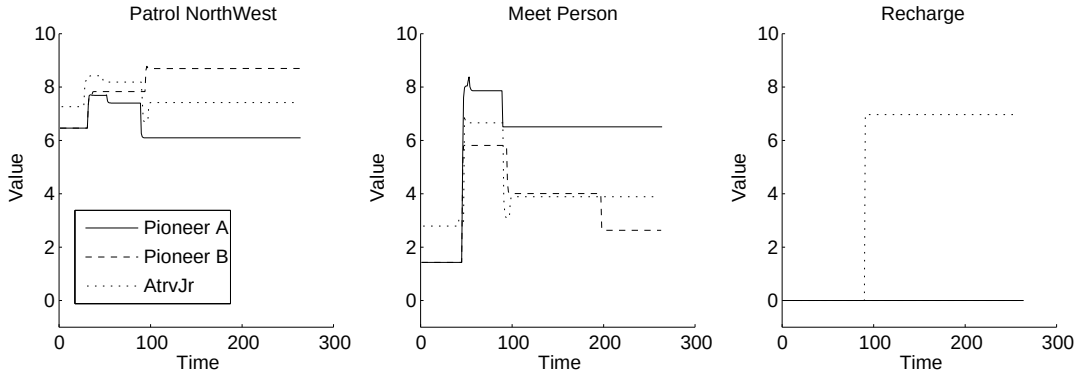

Figure 4: Three Patrol tasks and a Meet person task.

over time are plotted in Figure 3. As the belief of the robot changes while moving through the environment, the value for each of the tasks is updated. The AtrvJr robot has initially the highest value for any task since it is the fastest robot. Nevertheless it can only be assigned one task, in this case the "Patrol SouthEast" task, and the Pioneers are assigned the remaining three tasks: Pioneer A gets "Patrol NorthWest", Pioneer B "Patrol NorthEast", and Pioneer C "Patrol SouthWest". Although the AtrvJr robot has initially a higher value for the "Patrol NorthWest" task, it is assigned a different task. The reason is that the task assignment is determined by maximizing the sum of all bid values (3) and not individual bids.

Until about 100 time units, all Pioneers are still close to their initial starting position. As a result, their values for each task are similar, but a hysteresis mechanism prevents the assignment solution from changing too often. But as the robots move progressively away from the starting point, their values also become more different and the assignment solution stabilizes naturally.

In the second experiment (see Figure 4), three of the robots again start at the same node, but now at location $(46, 75)$. The robots are requested to execute three patrol tasks and also a recharge task. This is only executed when their battery level is low enough. Since the robots start with a full battery, initially each is assigned a patrol task. At about 50 time units, a person is detected in the node at location $(46, 90)$ and a task to meet the person is requested. Since the patrol and recharge tasks have a lower priority, one of the robots, Pioneer A, abandons its patrol task and was assigned the "Meet person" task. The other two robots are assigned two of the patrol tasks and one task is left unassigned. At around 100 time units, the AtrvJr robot, while moving to the patrol task goal, passes by the node containing the battery recharge station. Since the destination of the patrol task is still far and its battery is low, it is assigned the recharge task.

Finally, in the third experiment (Figure 5) robots Pioneer A and B are initially at nodes with locations $(2.5, 17.5)$ and $(87.5, 17.5)$ respectively. The robots where initially requested to execute two patrols tasks, one for each of their respective locations. Since the robots where already at their goals, they did not move. At about 40 time units, a person was detected at node $(46, 17.5)$ and an identify person task was requested. For this task, unlike the meet person, the robot is only required to approach the person close enough to take a clear picture. Although the robot Pioneer A is further away from the person, it has a camera with a better resolution and can take a picture at a greater distance. For this reason it is assigned the "Identify person" task instead of Pioneer B.

These experiments demonstrate that the auction protocol enable the robots to coordinate their task execution without communication of their state or beliefs. Also, the system is able to respond to detected events that occur during the execution of the tasks initially requested. Although in simulation, the presented methodology should transfer well to a real-world scenario, given the robustness with respect to communication failures.

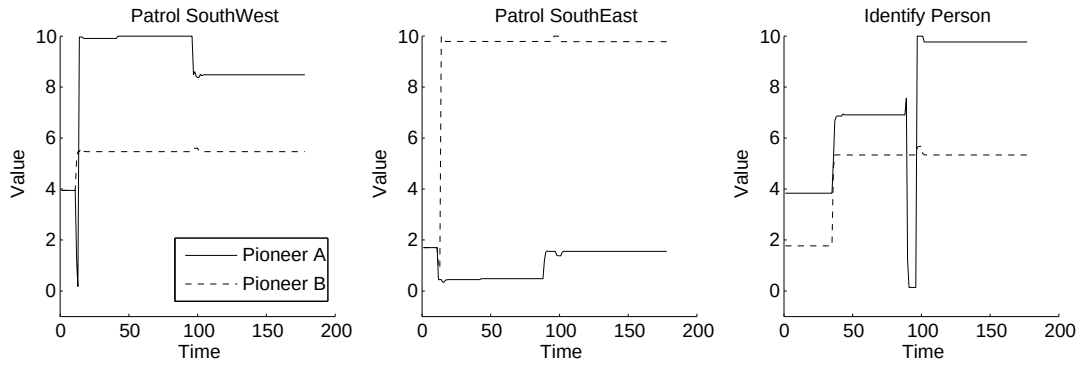

Figure 5: Identify person task and two Patrol tasks.

7. DISCUSSION AND FUTURE WORK

An approach to the assignment and execution of tasks in a multiagent system was presented. The motivation behind the approach was to illustrate the benefits of using auction protocols and the POMDP framework in multiagent systems. The auction protocols enable the coordination of multiple agents under stringent network operation conditions and robustness to individual agent failures. But an important drawback in the use of auctions is the question of how to estimate each agent's bid values.

The synthesis of controllers for the execution of tasks was performed using the POMDP framework. If suitable stochastic models of the environment and the agent observations are available, the synthesis problem can be formulated in a straightforward mathematical manner. The main difficulty of the POMDP approach is to compute a solution in an efficient manner. This is especially the case for almost all problem instances, expect those with relatively small dimensions.

The combination of the two frameworks produced a solution in which the individual drawbacks are mitigated. From the synthesis of controllers using POMDP task models, the values to bid are naturally obtained from the respective expected discounted rewards. Another advantage is that the agent's belief is already factored into this value. As a result, it is not necessary to invest additional time in the design of bid functions for each of the agents' tasks. Furthermore, as they are derived directly from the task controller, they are likely to reflect better true bid values, compared to commonly used heuristic bid functions.

The use of an auction enabled the use of smaller POMDP models than otherwise would be used if all agents and all tasks are considered simultaneously. This because the agents' coordination is implied in the use of the auction protocol and the auctioneer. Therefore, in the controller synthesis problem the other agents and tasks can be abstracted away. This is at the cost of optimality, since in practice the agents can interfere in each others' task execution.

It is possible, depending on the application, to use different auction protocols, such as on-line combinatorial auctions [13], or for agents to bid on bundles of tasks. This would require that the arrival order for tasks is known or that agents could accurately estimate their future rewards. Also, instead of POMDPs other planning models can be used, as long as they involve computing a value function.

A direction of future research is the synthesis of a controller for the auctioneer to determine the task priorities. The purpose is to maximize some performance criteria, such as the minimum assignment delay for some task types. The controller can also be used to determine which tasks to trade with other auctioneers.

Acknowledgments

This work was supported by the European Project FP6-2005-IST-6-045062-URUS, ISR/IST pluriannual funding through the POS_Conhecimento Program that includes FEDER funds, and through FCT grant PTDC/EEA-ACR/73266/2006. Nelson Gonçalves is working under grant SFRH/BD/23804/2005 from Fundação para a Ciência e a Tecnologia.

8. REFERENCES

[1] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein. The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research*, 27(4):819–840, 2002.

[2] M. Brydon. Economic metaphors for solving intrafirm allocation problems: What does a market buy us? *Decision Support Systems*, 42(3):1657–1672, 2006.

[3] R. E. Burkard. Selected topics on assignment problems. *Discrete Appl. Math.*, 123(1-3):257–302, 2002.

[4] M. I. Daoud and N. Kharma. A high performance algorithm for static task scheduling in heterogeneous distributed computing systems. *Journal of Parallel and Distributed Computing*, 68(4):399–409, April 2008.

[5] M. B. Dias, R. Zlot, N. Kalra, and A. Stentz. Market-based multirobot coordination: a survey and analysis. *Proceedings of the IEEE*, 94(7):1257–1270, 2006.

[6] B. P. Gerkey and M. J. Mataric. Sold!: auction methods for multirobot coordination. *IEEE Transactions on Robotics and Automation*, 18(5):758–768, 2002.

[7] B. P. Gerkey and M. J. Mataric. Multi-robot task allocation: analyzing the complexity and optimality of key architectures. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 3, pages 3862–3868, 2003.

[8] M. Hauskrecht. Value function approximations for partially observable Markov decision processes.

Journal of Artificial Intelligence Research, 13:33–95, 2000.

[9] J. Jungwattanakit, M. Reodecha, P. Chaovalitwongse, and F. Werner. A comparison of scheduling algorithms for flexible flow shop problems with unrelated parallel machines, setup times, and dual criteria. *Computers and Operations Research*, 36(2):358 – 378, 2009.

[10] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99–134, 1998.

[11] S. Koenig and C. Tovey. Sequential bundle-bid singlesale auction algorithms for decentralized control. In *Proc. Int. Joint Conf. on Artificial Intelligence*, pages 1359–1365, 2007.

[12] A. R. Mosteo and L. Montano. Comparative experiments on optimization criteria and algorithms for auction based multi-robot task allocation. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3345–3350, 2007.

[13] N. Nisam, T. Roughgarden, E. Tardos, and V. V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, September 2007.

[14] F. A. Oliehoek, M. T. J. Spaan, and N. Vlassis. Optimal and approximate Q-value functions for decentralized POMDPs. *Journal of Artificial Intelligence Research*, 32:289–353, 2008.

[15] L. E. Parker. Alliance: an architecture for fault tolerant multirobot cooperation. *IEEE Transactions on Robotics and Automation*, 14(2):220–240, 1998.

[16] P. Poupart. *Exploiting Structure to Efficiently Solve Large Scale Partially Observable Markov Decision Processes*. PhD thesis, University of Toronto, 2005.

[17] D. V. Pynadath and M. Tambe. The communicative multiagent team decision problem: Analyzing teamwork theories and models. *Journal of Artificial Intelligence Research*, 16:389–423, 2002.

[18] A. Sanfeliu and J. Andrade-Cetto. Ubiquitous networking robotics in urban settings. In *Workshop on Network Robot Systems. Toward Intelligent Robotic Systems Integrated with Environments. Procs. of 2006 IEEE/RSJ International Conference on Intelligence Robots and Systems (IROS2006)*, October 2006.

[19] S. Seuken and S. Zilberstein. Formal models and algorithms for decentralized decision making under uncertainty. *Autonomous Agents and Multi-Agent Systems*, Feb. 2008.

[20] R. G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Trans. Comput.*, 29(12):1104–1113, 1980.

[21] M. T. J. Spaan and N. Vlassis. Perseus: Randomized point-based value iteration for POMDPs. *Journal of Artificial Intelligence Research*, 24:195–220, 2005.