Spatio-Temporal Clustering of Tasks for Swap-Based Negotiation Protocols in Multi-Agent Systems

Matteo Golfarelli and Stefano Rizzi
DEIS, CSITE CNR - University of Bologna, Italy
golfare@csr.unibo.it, srizzi@deis.unibo.it

Abstract. Path planning in robotic agents consists in determining, on a network of places and routes modeling the environment, a sequence of resources to be visited in order to carry out a set of tasks. In multi-agent systems, agents may cooperate to decrease the task execution costs; the Contract Net Protocol is an approach to negotiation of tasks based on the announcement-bid-award mechanism. If the agents do not own money, they can decrease their costs only by swapping tasks with other agents; unfortunately, swapping only single tasks may trap negotiation in local minima of the cost. In this paper we show how the utility of negotiations can be increased by allowing tasks to be swapped in non-disjoint clusters. Clustering of tasks is carried out, in a fuzzy fashion, according to two orthogonal dimensions which consider, respectively, the spatial disposition of the resources within the environment and the temporal progression of their time windows; the approach is scalable to support other dimensions as well.

1 Introduction

Path planning in robotic agents is the search for a route in the environment which enables an agent to achieve a set of inter-related goals. In symbolic approaches, path planning is carried out on a symbolic description of the environment; the main interest is not in achieving motion in the physical space but in determining, on a network of places and routes, a sequence of places to be visited in order to carry out a set of tasks.

When several agents are present in the environment, they must coordinate their activities in order to detect and resolve the conflicts which may arise in planning; besides, they may cooperate to decrease the task execution costs. The cooperation techniques to be adopted depend strongly on the nature of the agents: a *benevolent agent* tends to maximize the global utility of the agent society, whereas a *self-interested agent* is inclined towards maximizing its own profit and thus is open to cooperation only if it is convenient for itself.

In [3] we proposed a negotiation protocol for multi-agent path planning which extends the *Contract Net Protocol* (*CNP*) [10] to environments where the only possible type of contract is the task-swapping. Each agent plans an initial path to execute its tasks, then progressively modifies it and decreases its cost by swapping tasks with other agents. The agents we consider are self-interested, may have different skills and have limited communication capabilities; the tasks to be executed require one or more resources, placed in the environment and available during limited time windows.

In this paper we show how the effectiveness of swap-based negotiations can be enhanced by allowing tasks to be swapped in clusters, in order to climb over local minima in the cost function. Clustering of tasks is carried out, in a fuzzy fashion, according to two orthogonal dimensions which consider, respectively, the spatial disposition of the resources within the environment and the temporal progression of their time windows; the approach is scalable to support other dimensions as well. In order to increase the probability of carrying out convenient swaps, the clusters produced are non-disjointed; thus, the same task can be proposed within different negotiations.

In Section 2 we introduce our hypotheses about agents and tasks. In Section 3 the swap-based negotiation protocol is outlined, while in Section 4 the approach to clustering is described. Finally, some experimental results are presented in Section 5.

2 The agents

By the term *agent* we denote a robot capable of moving autonomously within a structured, partially known environment. In the society we consider, agents are heterogeneous and self-interested; they do not share any physical memory and are not coordinated by any central supervisor. Agents communicate with each other by broadcasting messages; the communication range is limited to a radius ρ, and broadcasting is assumed to be noise-free (no loss of messages). Each agent is capable of performing a finite set of actions (for instance, commute a switch or empty an ash-tray). We assume that agents own neither money nor any other explicit means for utility transfer.

Each agent owns a private, possibly incomplete description of the environment. Knowledge of the environment is structured as a graph whose vertices and arcs represent, respectively, *landmarks* (distinctive or serviceable sites in the environment) and *routes* (feasible trajectories between landmarks) [5]. The agent evaluates every path P belonging to this graph by means of a cost function $cost(P)$ which may express, for instance, its length or the time/fuel required to follow it.

Each agent is assigned a set S of tasks; we assume that each task:

- entails an *action*.
- must be executed either on a *resource* or on a *resource type*. By the term *resource* we mean a landmark where the task must be executed (for instance, "*the director's computer*"); a *resource type* is a set of resources such that the task may be executed indifferently on either of them (for instance, "*any computer*").
- may be constrained to be executed within a given *time window* (typically, because the resource required is not available outside that time window).

We assume that the initial assignment of tasks to agents is carried out randomly.

The primary goal of each agent is to execute its tasks at the lowest possible cost. Thus, each agent must be capable of planning a "cheap" path, $path(S)$, which allows the resources required by the tasks in S to be visited in observance of the temporal constraints posed by time windows; examples of such planning algorithms can be found in [1] [2] [6].

The *relative marginal cost* of the subset of tasks $C \subseteq S$ is defined as:

$$rmc(S, C) = cost(path(S)) - cost(path(S-C))$$

In particular, if S includes a task entailing an action which the agent is not capable of executing, then $cost(path(S))=\infty$.

3 The swap-based contract net protocol

The CNP [10] is a *market-based* approach to negotiation in multi-agent systems. Each agent can formulate a *bid* for each *announcement* received; the contract will be *awarded* to the agent which sends the best bid.

Assuming that the agents do not own money, they cannot sell tasks; thus, the only way an agent can decrease its execution cost is to swap tasks with other agents. Let S_A and S_B be, respectively, the current sets of tasks assigned to agents A and B. The *utility* for A in swapping the subset of tasks $C_1 \subseteq S_A$ with the subset of tasks $C_2 \subseteq S_B$ assigned to B is measured as the difference between the cost of $path(S_A)$ and that of $path(S_A \cup C_2 - C_1)$. A swap is said to be *individual rational* for an agent if its utility is positive.

The swap-based negotiation protocol we proposed in [3] consists of three phases:

Announcement. The announcing agent formulates an announcement and broadcasts it to the other agents. An announcement consists of one task which the announcer is interested in exchanging. The announcer, in selecting the task to negotiate, cannot calculate the utility it will receive; thus, a reasonable heuristic it can follow is to choose the task $s{*} \in S_A$ with the highest relative marginal cost, $rmc(S_A, \{s{*}\})$.

Bid. Each agent receiving the announcement formulates a bid and broadcasts it to the announcer. The bid associated to an announcement consists of one or more tasks, owned by the bidder, which the bidder is interested in exchanging with the task included in the announcement. A swap is proposed only if it is individual rational. The bidder can calculate the utilities it will get by swapping $s{*}$ with each of its tasks; after dropping all the non-individual rational swaps, it will decide how to formulate the bid. The most "politically correct" strategy it can adopt is to include in the bid all the individual rational swaps. This strategy maximizes the probability of success and the utility of the announcer; unfortunately, it may produce little utility for the bidder. Alternatively, the bidder can decide to include only the most convenient swap(s) in the bid: in this case the bidder is inclined to accept a reduction of the probability of success in exchange for a more advantageous deal.

Award. The announcer collects the bids for a fixed time. When this time expires the announcer determines, among all the bids received, the swap having the highest utility. If the utility is positive, then the swap is individual rational for both the announcer and the bidder; an award is broadcasted to the winner and the swap is confirmed.

Since the tasks with the highest relative marginal costs are announced first, each agent has good probability of succeeding in swapping the tasks it is not able to carry out; nevertheless, if no agent within the communication radius can propose individual rational swaps, it may happen that these tasks cannot be negotiated successfully.

Negotiating one task at a time can be insufficient, whenever the cost for carrying out a task depends on the execution of other tasks; this may lead to local optima, where no transfer of a single task enhances the global solution, but transferring a cluster of tasks simultaneously does [9]. A simple example of this situation is shown in Figure 1, where squares represent the resources on which tasks must be executed: the white ones are assigned to agent A, the black ones to agent B; only swapping all the tasks between the two agent is individual rational for both agents, i.e., leads to a reduction of the global cost.

A clustering-based version of the negotiation protocol can be formulated as follows:

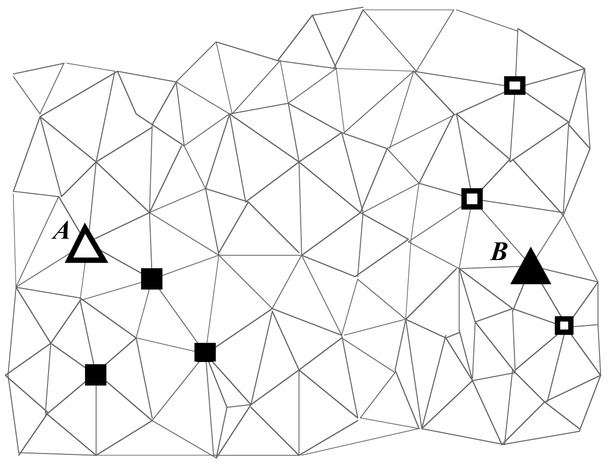

Figure 1. A local minimum for the cost function. The graph representing the environment is drawn in gray.

Announcement. The announcing agent applies the clustering algorithm (see Section 4.3) to the set S_A of its tasks. The announcement consists of the cluster $C^*\subseteq S_A$ with the highest relative marginal cost, $rmc(S_A, C^*)$.

Bid. The bidder applies the clustering algorithm to its tasks and offers the cluster(s) which determine individual rational swaps with C^*.

Award. Same as above, except that the swap involves two clusters.

4 Clustering of tasks

The problem of clustering assumes a significant role in a variety of research areas, ranging from pattern recognition to computer vision [4]. In particular, clustering of tasks is a complex problem, formalized in [8], for which some domain-dependent heuristics have been proposed [7]. The main issue in clustering is defining a measure for the similarity between tasks; since the tasks assigned to the agents are constrained both in space (by the positions of the resources where they must be executed) and in time (by the time windows), the approach to clustering we propose in this paper defines the similarity by taking both the spatial and temporal distances between tasks into account.

After defining a spatial and a temporal distance, we propose an approach to clustering where the space and the time dimensions are integrated in a fuzzy fashion. It is remarkable that, if additional kinds of constraints on tasks are defined within the application domain, the clustering algorithm can be easily scaled to integrate *n* dimensions. Two dimensions which could be considered relate, respectively, to the existence of precedence constraints between tasks and to the fact that tasks may require and/or produce one or more objects. This is an example of a set of tasks, expressed in natural language, for an office-agent: "*Pick up a document from the secretary's desk, stamp it in the director's office and take it back to the secretary; take a copy of the stamped document to the archive. The director leaves at 11 a.m. and the archive is open from 10 a.m.*". Here, the document and its copy are objects produced and required; the secretary's desk, the director's office, any photocopier and the archive are resources; pick up, take, stamp and photocopy are actions; "*until 11 a.m.*" and "*after 10 a.m.*" denote time windows; the fact that the document must have been stamped before it is copied implies a precedence constraint.

4.1 Spatial distance

The *spatial distance* between two resources r' and r'' is defined as the cost of the shortest path between the positions of r' and r'':

$$\delta_s(r', r'') = \delta_s(r'', r') = cost(shortestPath(r', r''))$$

This definition assumes that the cost function is symmetric, i.e., that the cost for covering routes does not depend on the direction. In environments where this is not true (for instance, due to the presence of one-way streets or doors), the spatial distance between two resources can be defined for instance as the average of the costs for covering the shortest paths in the two directions (it is reasonable to assume that every ordered pair of resources is always connected by at least one path with finite cost).

The *spatial distance* between two resource types R' and R'' is defined as the minimum among the spatial distances between their member resources:

$$\delta_s(R', R'') = min_{\{r_i \in R',\, r_j \in R''\}} (\delta_s(r_i, r_j))$$

The *spatial distance* between two tasks is defined as the spatial distance between the two resources (or resource types) on which the tasks must be executed.

Since our approach to clustering is threshold-based, it is necessary to complete the spatial metrics we introduced by defining a *spatial threshold* τ_s, which will be used to distinguish between “near” tasks and “far” tasks. Experiments confirm that a good solution to define the spatial threshold is to make it proportional to the length of the path the agent must cover in order to carry out all the tasks currently included in its S set.

4.2 Temporal distance

Spatial distance is not always sufficient to model similarity between two tasks. In fact, two tasks may be executed on resources which, though being spatially neighboring, have very distant time windows.

Clustering tasks on the temporal dimension means grouping the tasks which must be executed within “near” time windows; however, the definition of distance between time windows is not as immediate as for the spatial case. A time window is an interval $[t_{start}, t_{end}]$, where we assume for simplicity that $(0\leq)t_{start}\leq t_{end}(\leq 24)$. Consider for instance the situation in Figure 2: it is not clear which of the upper two windows is nearer to $[t_1,t_6]$, and whether it is more convenient to cluster $[t_2,t_3]$ with $[t_4,t_5]$ or with $[t_1,t_6]$.

Our approach maps time windows into points on a Cartesian plane, thus transforming the distance between two windows into a Euclidean distance. Consider Figure 3, where t_{start} and t_{end} are represented on two Cartesian axes. Each point in the triangle defined by $t_{start}\leq t_{end}$ represents a time window; the *temporal distance* between two resources is then defined as the Euclidean distance between the two points corresponding to the time windows of the resources:

$$\delta_t\left(\left[t'_{start}, t'_{end}\right], \left[t''_{start}, t''_{end}\right]\right) = \sqrt{\left(t'_{start} - t''_{start}\right)^2 + \left(t'_{end} - t''_{end}\right)^2}$$

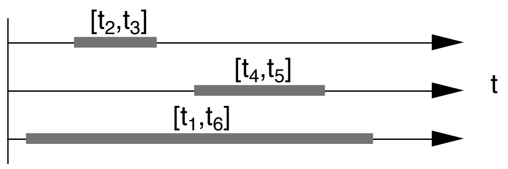

Figure 2. Three time windows.

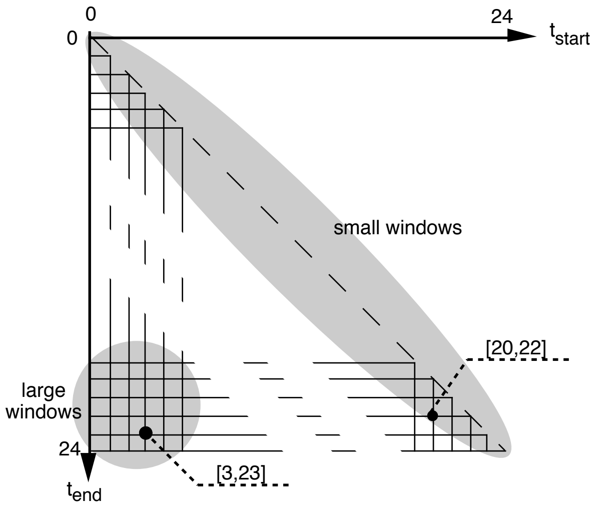

Figure 3. Definition of temporal distance on a Cartesian plane.

It is easy to verify that, in this metrics, two small windows are near if they are overlapping or if one starts immediately after the other; on the other hand, the larger windows are, the nearer they are independently of their starting and ending times.

Assuming for simplicity that all the resources belonging to each resource type have the same time window, it is possible to define the *temporal distance* between two tasks as the temporal distance between the two resources (or resource types) on which the tasks must be executed.

As to the *temporal threshold* τ_t, it can be made proportional to the overall time necessary to cover the path the agent planned to carry out all the tasks currently included in its S set.

4.3 The clustering algorithm

In the specific application domain of task negotiation by CNP, the target of clustering is to determine the sets of tasks to be announced. In order to provide higher flexibility to negotiation, our approach produces non-disjointed clusters; this means some tasks (especially those associated to resource types widely distributed in the environment and/or large time windows) may be offered during several negotiations.

Let n be the number of clustering dimensions (n=2 if only the spatial and temporal dimensions are considered), m be the current number of tasks for the agent considered and $\boldsymbol{K}_x$ (x=1,…n) be the $m\times m$ proximity matrix for tasks computed according to the distance defined for dimension x; the i,j-th element in these matrices denotes the distance between tasks s_i and s_j. Task clustering is carried out in three steps:

1. Each proximity matrix $\boldsymbol{K}_x$ is normalized by means of a sigmoid function:

$$\sigma_x(\delta_x) = \frac{1}{1+\alpha\left(\frac{\delta_x}{\tau_x}\right)^{\sqrt{\tau_x}}}$$

   where δ_x and τ_x are, respectively, the distance and the threshold for dimension x; parameter α is used to normalize the thresholds (see next paragraph).

2. The n normalized matrices are merged into a resulting matrix $\boldsymbol{K}$ by multiplying each n-ple of corresponding cells:

$$K(i,j) = \prod_{x=1}^{n} K_x(i,j)$$

By doing so, a cell in $\boldsymbol{K}$ has value near to 1 only if all the corresponding cells in the component matrices have values near to 1; on the other hand, it has value near to 0 if at least one of the corresponding cells has value near to 0. Of course, if two tasks are distant exactly τ_x for every x, the corresponding element in $\boldsymbol{K}$ must have value equal to the global threshold τ:

$$\prod_{x=1}^{n} \sigma_x(\tau_x) = \tau$$

which can be easily satisfied by imposing

$$\sigma_x(\tau_x) = \sqrt[n]{\tau} \Rightarrow \alpha = \tau^{-\frac{1}{n}} - 1$$

3. Matrix $\boldsymbol{K}$ is clustered as follows:

```
CS←{$C_i$,i=1,...m|$C_i$={$s_i$}};
// CS set of clusters; $C_1$,...$C_m$ clusters; $s_1$,...$s_m$ tasks
for each i,j=1,...m,i≠j do
  if K(i,j)<τ then $C_i$←$C_i$∪{$s_j$};
// τ is the global clustering threshold
for each i=1,...m do
  if ∃$C_j$∈CS|$C_i$⊆$C_j$ then CS←CS-$C_i$;
```

First, one cluster is created for each task in S. Then, each cluster is augmented with the "nearest" tasks. Finally, the clusters totally included in other clusters are eliminated.

The tasks within the clusters produced will be offered together during negotiation.

5 Experimental results

We have carried out a set of experimental tests by simulating negotiation sessions within a society of mobile robots navigating the same environment. The results are mainly influenced by:

- Number of agents, a. The higher a, the higher the probability of carrying out swaps, hence, the higher the reduction of the global cost.
- Communication radius, ρ. The higher ρ, the higher the average number of agents who receive each announcement and can join the negotiation session.
- Number of tasks assigned to each agent, m. The agents entrusted with several tasks have higher probability of carrying out swaps.
- Structure of the environment map. The lower is the average connectivity of the map, the more expensive is for the agents to navigate it, hence, the higher the potential benefits of negotiations.

The results we present refer to a set of tests made with a=8 and ρ=50% (of the total map diameter). If non-clustered negotiations are adopted, the global relative saving (defined as the percentage difference between the total execution cost for the society before and after negotiation) turns out to be 29%; in the clustered case, the global relative saving depends on the clustering threshold τ and on the dimensions adopted for clustering as shown in Table 1.

Table 1. Experimental results

τ	*Spatial (n=1)*	*Spatial+Temporal (n=2)*
0.4	33.2%	33.9%
0.5	33.2%	35.2%
0.6	31.9%	34.4%
0.7	31.9%	32.8%

Overall, the combined use of spatial and temporal clusterings turns out to increase the effectiveness of negotiation by allowing a more significant reduction of the overall execution cost to be achieved.

References

[1] O. Causse and J.L. Crowley, Navigation with constraints for an autonomous mobile robot, *Robotics and Autonomous Systems* **12**:3-4 (1994) 213-221.

[2] T. Dean, J. Firby and D. Miller, Hierarchical planning involving deadlines, travel time and resources, *Computational Intelligence* **4**:4 (1988) 381-398.

[3] M. Golfarelli, D. Maio and S. Rizzi, Multi-Agent Path Planning Based on Task-Swap Negotiation. In: Proceedings 16th UK Planning and Scheduling SIG Workshop, Durham, England, 1997, pp. 69-82.

[4] A.K. Jain and R.C. Dubes, Algorithms for Clustering Data. Prentice-Hall Editors, Englewood Cliffs, NJ, 1988.

[5] D. Maio and S. Rizzi, Knowledge architecture for environment representation in autonomous agents. In: Proceedings of the 8th International Symposium on Computer and Information Sciences, Istanbul, 1993, pp. 4-11.

[6] D. Maio and S. Rizzi, Layered knowledge architecture for navigation-oriented environment representation, Technical Report CIOC-C.N.R., n. 108, 1996.

[7] T. Sandholm, An implementation of the contract net protocol based on marginal cost calculations. In: Proceedings 11th National Conference on Artificial Intelligence (AAAI-93), 1993.

[8] T. Sandholm, Necessary and sufficient contract types for optimal task allocation. In: Proceedings 14th International Joint Conference on Artificial Intelligence (IJCAI-97), Nagoya, Japan, 1997.

[9] T. Sandholm, and V. Lesser, Issues in automated negotiation and electronic commerce: extending the contract net framework. In: Proceedings 1st International Conference on Multi-Agent Systems, San Francisco, CA, 1995, pp. 328-335.

[10] R.G. Smith, The Contract Net Protocol: high-level communication and control in a distributed problem solver, *IEEE Transactions on Computers* **C-29**:12 (1980) 1104-1113.