A Mechanism for Temporal Reasoning by Collaborative Agents*

Meirav Hadad[1] and Sarit Kraus[1,2]

[1] Department of Mathematics and Computer Science
Bar Ilan University Ramat Gan 52900, Israel
[2] Institute for Advanced Computer Studies, University of Maryland
College Park, MD 20742

1 Introduction

This paper considers the problem of temporal scheduling of actions in a cooperative activity under time constraints. In order to carry out their cooperative activity the agents perform collaborative planning that includes processes that are responsible for identifying recipes, assigning actions to agents and determining the time of the actions. A recipe for an action consists of subactions which may be either *basic* actions or *complex* actions.

Basic actions are executable at will if appropriate situational and temporal conditions hold. A complex action, can be either a single-agent action or a multi-agent action. In order to perform a complex action the agents have to identify a recipe for it and it might know several recipes for an action. A recipe may include temporal constraints and precedence relations between its subactions. For the agents to have achieved a complete plan, the values of the time parameters of the actions that constitute their joint activity must be identified in such a way that all the appropriate constraints are satisfied. Determining the time of the actions in a collaborative environment that we consider is complex because of the need to coordinate actions of different agents, the partiality of the plans, the partial knowledge of other agents' activities and the environment, temporal and resource constraints. Furthermore, the temporal constraints requires interleaving of planning and execution. In this paper we present briefly algorithms for cooperative agents to determine the time of the actions that are required to perform their joint activity.

2 An Algorithm for Temporal Reasoning

Building a collaborative agent that can flexibly achieve its goals in changing environments requires a blending of Real-Time (RT) computing and Artificial Intelligence (AI) technologies. Accordingly, our single-agent system consists of an AI subsystem and an RT subsystem. The goal of the AI subsystems when the agents attempt to perform α is to identify a set of basic actions and a set

* This work is supported in part by NSF under Grant No. IIS-9907482.

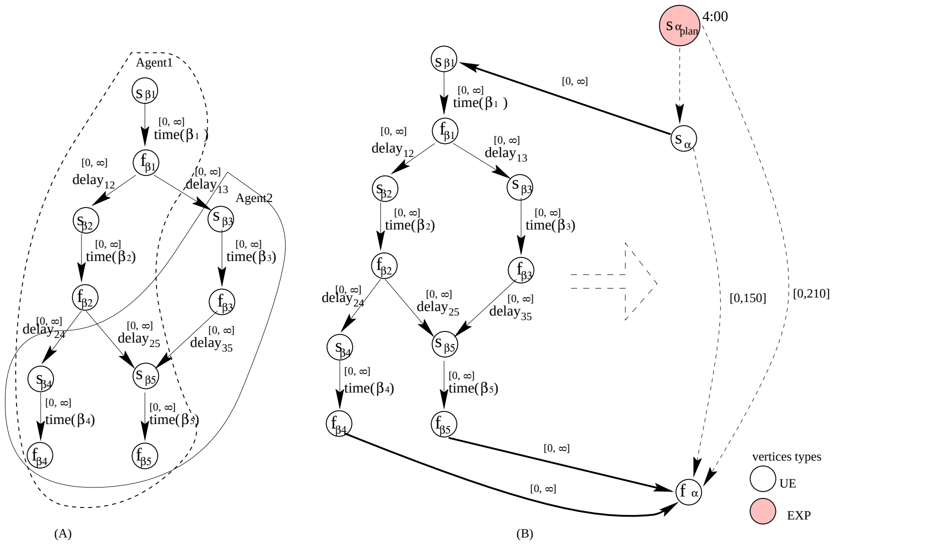


Fig. 1. (A) An example of $Gr^p_{R_\alpha}$. (B) An example of adding a precedence graph to Gr^i_α.

of the temporal constraints associated with the basic actions such that performing the basic actions under these temporal constraints constitutes performing α without conflict. As soon as it is identified, each basic action β is sent to the RT subsystem, along with its temporal requirements $\langle D_\beta, d_\beta, r_\beta, p_\beta \rangle$ where D_β is the **Duration time**, i.e., the time necessary for the agent to execute β without interruption; d_β denotes the **deadline**, i.e., the time before the β should be completed; r_β refers to the **release time**, i.e., the time at which β is ready for execution; p_β denotes the **predecessor actions**, i.e., the set $\{\beta_j|(1 \leq j \leq n)\}$ of basic actions whose execution must end before the beginning of β.

In our system, the agents collaboratively develop the plan of their joint activity. Section 2.2 presents the major constituents of the time reasoning mechanism, which is based on the mechanism for an individual agent presented in [2]. However, expansion of collaborative plans differs from the individual case [1]. When an agent acts alone, it determines how to do an action (i.e., find a recipe) and then carries out the necessary subactions. When agents work together, both the formation or selection of the recipe and the carrying out of the subactions is distributed. The group must reach a consensus on how they are going to perform the action (i.e., on the recipe they will use) and on who is going to do the subactions. When each agent (or subgroup) decides on when to perform a subaction the temporal constraints of the group members must be taken into consideration. Thus, recipe and agent selection in collaborative planning require inter-agent negotiation and communication as well as individual agent reconcili-

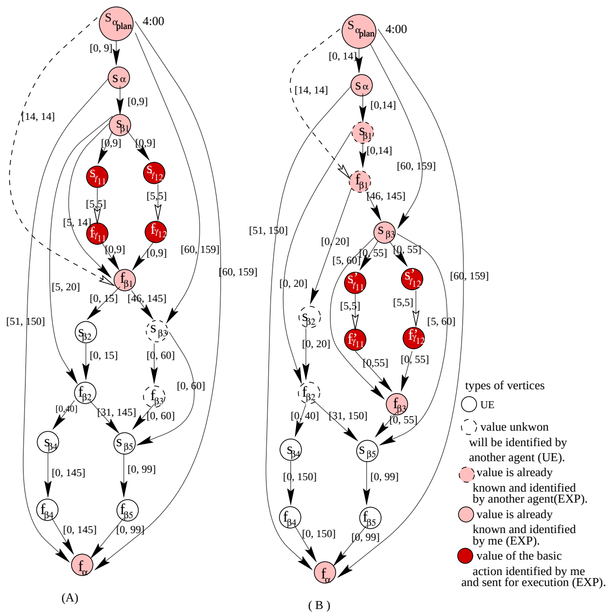


Fig. 2. An example of temporal graphs Gr^i_α which are maintained by A_1 (graph A) and A_2 (graph B) during the planning of the joint activity α.

ation. In the mechanism presented, for simplicity, we assume that each action is associated with the agent that performed it. Thus, there is no need for decision making concerning the performer of the action. We focus on the time scheduling problem and briefly describe some of its components. More details about the single-agent aspects of the system can be found in [2].

2.1 The temporal constraints graph

The main two structures that are used when the agents identify the time parameters of their joint activity α is a precedence graph and a temporal constraints graph. A precedence graph $Gr^p_{R_\alpha}$ with respect to recipe R_α represents the precedence constraints of R_α (see Figure 1(A)).

The temporal constraints graph associated with a multi-agent action α maintains information about the temporal constraints of the actions that constitute the performance of α and the precedence relations among them. Formally, a temporal constraints graph, $Gr^i_\alpha = (V, E)$, is $V = \{s_\alpha, s_{\beta_1}, \cdots, s_{\beta_n}, f_\alpha, f_{\beta_1}, \cdots, f_{\beta_n}\} \cup \{s_{\alpha_{plan}}\}$ and $\{\alpha, \beta_1, \cdots, \beta_n\}$, represents complex and basic actions that the agent intends to perform in order to execute the highest level action α (see Figure 2). The variables s_y and f_y represent the time points at which the execution of an action $y \in \{\alpha, \beta_1, \cdots, \beta_n\}$ can start and must finish, respectively. Some of the vertices may be fixed, i.e., these vertices denote a known time which cannot be modified. The vertex $s_{\alpha_{plan}}$ represents the time point that an agent A starts to plan the action α; it is a fixed vertex. Other fixed vertices may be initiated, for example, by a request from a collaborator. The activity of action y is represented by a directed edge between s_y and f_y, that is labeled by the time duration required for the execution of action y. A directed edge between the finish time point of action y, to the start time point of another action $z \in \{\alpha, \beta_1, \cdots, \beta_n\}$ denotes that the execution of z cannot start until the execution of y is completed. The interval associated with this edge indicates the possible delay between these actions. A metric constraint $a_{i,j} \leq (v_i - v_j) \leq b_{i,j}$ between two different time points $v_i, v_j \in V$ is represented by the metric edge (v_i, v_j) that is labeled $[a_{i,j}, b_{i,j}]$. One of the main purposes of the algorithm is to determine the values of s_y and f_y for each task.

In addition to the temporal information, a temporal constraints graph maintains additional information on each of the graph's actions: (a) whether the action is basic or complex; (b) whether the complex action is a multi-agent or a single agent action; (c) whether a plan for the action has been completed; (d) the agent(s) that is(are) assigned to perform the action, and (e) whether the action has been already executed by the agent(s).

This information is determined incrementally by the algorithm which also expands the graph. Each agent A_i, in the system runs the algorithm in order to build its temporal constraints graph Gr^i_α. The graphs of individual agents may be different with respect to individual actions, but similar with respect to the first level of multi-agent actions.

2.2 The algorithm in general

During the agents' planning process, each agent A_i selects an action β from its Gr^i_α. For this action, A_i is the sole performer or A_i participates in its performance and the agents agree that A_i will develop the plan for β. Then, A_i attempts to complete β's plan. Action β is selected by A_i only if the planning of all the actions which precede β have been completed. If β is a basic action that has to be performed by A_i, it is sent to A_i's RT subsystem for execution. Otherwise, A_i performs the following major activities: (1) Finds an appropriate recipe, R_β, for the complex action, β. (2) Constructs the precedence graph $Gr^p_{R_\beta}$ for β and incorporates it along with the other temporal constraints of R_β in Gr^i_α. (3) It checks the consistency of the updated temporal graph. If Gr^i_α is not consistent it searches for a different recipe for β, R'_β, and replaces the information it added in 2 with the information associated with R'_β and checks consistency again. If such R_β does not exist it backtracks. (4) Determines the new values of the vertices in Gr^i_α by using an algorithm such as the Floyd-Warshall algorithm for solving a temporal constraints problem (see [2]). (5) Obtains messages from the RT about the execution of the basic level actions and updates the timing of the appropriate actions. (6) Obtains messages from the other team members and updates the graph with respect to the information that is listed in the next section.

The development of a shared plan between the agents requires information exchange. Consider the case of two agents, A_1 and A_2, that intend to perform a joint activity α. They will exchange information in the following cases: (1) When they identify a recipe for their joint action or for any joint subaction in their plan. This exchange of messages may be about possible recipes and also may be part of their group decision making. (2) Agent A_1 may inform agent A_2 about the time values that it identified for its individual actions $\beta_1 \ldots \beta_k$ when $\beta_1 \ldots \beta_k$ *delay* the planning of action γ and γ has to be performed by A_2. We assert that $\beta_1 \ldots \beta_k$ *delay* the planning of γ if they directly precede γ. (3) When agent A_1 completes the plan of a subaction in a recipe of a joint action with A_2, it informs A_2 that the plan for this action has been completed (but does not send the details of the plan). (4) If A_1 already sent information to A_2 about some action β, but it failed to perform β, then A_1 backtracks and informs A_2 about the failure of β or about new properties of their plan that were determined as a result of its backtracking.

2.3 Example

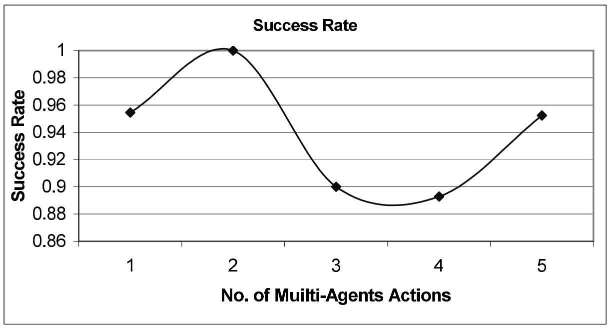
Suppose that agents A_1 and A_2 intend to perform a joint action α which is associated with the following constraints: { (finish-time $T_\alpha-$ start-time $T_\alpha \leq 150$), (start-time T_α after 4:00), (finish-time T_α before 7:30) }, where T_α represents the time interval of the execution of action α and we assume that initially the length of this interval is unknown. We also assume that both agents start the collaborative plan for α together at 4pm, then $V_{init} = \{s_{\alpha_{plan}}, s_\alpha, f_\alpha\}$ and $E_{init} = \{(s_{\alpha_{plan}}, s_\alpha, [0, \infty]), (s_\alpha, f_\alpha, [0, 150]), (s_{\alpha_{plan}}, f_\alpha, [0, 210])\}$. After the determination of the new values of the vertices, the new intervals of the edges in the example will be $E_{init} = \{(s_{\alpha_{plan}}, s_\alpha, [0, 210]), (s_\alpha, f_\alpha, [0, 150]), (s_{\alpha_{plan}}, f_\alpha, [0, 210])\}$ (see the dashed edges in Figure 1). Next, they need to agree on a recipe for α. Suppose that the agents agreed on R_α which consists of the subactions $\{\beta_1 \ldots \beta_5\}$ where β_1 and β_2 can be performed only by A_1, β_3 can be performed only by A_2 and β_4 and β_5 are multi-agent actions. In addition, the recipe is associated with the precedence relations $\{\beta_1$ before β_2; β_1 before β_3; β_2 before β_4; β_2 before β_5; β_3 before $\beta_5\}$ and the temporal constraints: {(finish-time $T_{\beta_2}-$ start-time $T_{\beta_1} \leq 20$), (start-time $T_{\beta_4}-$ finish-time $T_{\beta_2} \leq 40$), (finish-time $T_{\beta_5}-$ start-time $T_{\beta_3} \leq 60$), (start-time T_{β_3} after 5:00) }. Each agent builds the $Gr^p_{R_\alpha}=(V^p, E^p)$ and incorporates it into Gr^i_α by adding directed edges from s_α to each vertex $v_{b_1}, \ldots v_{b_m} \subseteq V^p$ with an indegree of 0 and by adding edges from each vertex $v_{e_1}, \ldots v_{e_k} \subseteq V^p$ with an outdegree of 0 to f_α. A pictorial illustration of adding this $Gr^p_{R_\alpha}$ to Gr^i_α is given in Figure 1. $Gr^p_{R_\alpha}$ is incorporated into Gr^i_α by adding the edges $(s_\alpha, s_{\beta_1}), (f_{\beta_4}, f_\alpha), (f_{\beta_5}, f_\alpha)$. After this, the agents also update the temporal constraints that are associated with R_α.

Then, each agent A_i tries to continue the development of its Gr^i_α by selecting an action to be developed. In this stage, Gr^1_α is identical to Gr^2_α but only the action β_1 may be selected since it does not depend on any other action. Since A_1 is the single agent involved with β_1, A_2 has to wait until it will receive the values of the temporal parameters of β_1 from A_1. Suppose that the recipe that A_1 selected for β_1 consists of the basic actions γ_{11} and γ_{12} and the execution time of each of them is exactly 5 minutes. Thus, A_1 should identify $\langle D_{\gamma_{11}}, r_{\gamma_{11}}, d_{\gamma_{11}}, p_{\gamma_{11}}\rangle$ and

$\langle D_{\gamma_{12}}, r_{\gamma_{12}}, d_{\gamma_{12}}, p_{\gamma_{12}} \rangle$ and send it to its RT subsystem. The decision regarding the exact time in which β_1 will be executed is determined by the RT subsystem. In this example, we assume that the RT subsystem of A_1 decides to execute γ_{11} at 4:02 and γ_{12} at 4:09. Thus A_1 will inform A_2 that it intends to terminate the execution of β_1 at 4:14. Following this announcement, A_2 can start developing β_3. Similarly, A_1 can develop β_2. A_2 will use this information to develop the plan for β_3. In addition, both of them will add the edge $(s_{\alpha_{plan}}, f_{\beta_1})$ and will label this constraint on it (i.e., the weight of this edge will be $[14, 14]$). The temporal graph Gr^i_α which is maintained by each agent in this stage of their collaborative plan is given in Figure 2. As shown in this figure each agent maintains a different graph according to its plan. Graph (A) is the graph which is built by A_1 and graph (B) is built by A_2. The identical vertices represents the subactions which appear in the recipe of their collaborative action.

2.4 Experimental Results

We developed a simulation environment to test our algorithm. In our simulations we ran the algorithm on several different recipe libraries which were created randomly. Each recipe library includes at least one possible solution to the joint activity. We tested the effects of the number of multi-agents' actions in the recipe tree of α on the success of the system in an environment comprising 2 agents. We ran 150 experiments with randomly drawn parameters from ranges with high success rates of one agent. We revealed that the number of multi-agent actions that are explored when developing the plan for α does not influence the success rate of the agents which is between $90\% - 100\%$ (see the graph below). We can conclude that for the environments that we checked, the multi-agent activity does not make the temporal reasoning problem significantly more difficult.



References

1. B. Grosz and S. Kraus. Collaborative plans for complex group action. *AIJ*, 86, 1996.
2. M. Hadad, S. Kraus, Y. Gal, and R. Lin. Time reasoning for a collaborative planning agent in a dynamic environment. *Annals of Math. and AI*, 2001. www.cs.biu.ac.il/ sarit/articles.html.