A Coherent Teamwork Model in a Dynamic Environment

M. Goyal N.Parameswaran

Department of Information Engineering School of Computer Science and Engineering The University of New South Wales Sydney 2052 Australia {madhug,paramesh}@cse.unsw.edu.au

Abstract.

A team is a unique example of collection of agents held in dynamic and functional relationships over brief periods of time by mutual belief and joint commitment. The central hypothesis in this paper is that for effective teamwork, agents should be provided with explicit time lines and an underlying model of teamwork based on a concept of team knowledge. The model also emphasizes on the team agent architecture as its core to allow explicit and coherent team behavior. The application and implementation of this model to a virtual fire-fighting domain (FFAGENTS) has revealed a promising prospect in developing team agents.

Introduction

Recently the concept of teams is evolving into a dynamic multifaceted entity; computer scientists in distributed artificial intelligence study its complicated nature. According to some philosophers and sociologists, teams are co-operative groups in that they are called into being to perform a task or tasks that cannot be attempted by an individual. The team functions effectively only when its members operate as smoothly interlocking and complementary parts of the larger whole, eschewing much independent choice in favor of predictable behavior.

In this paper, we have developed an explicit model of teamwork. Our idea of team is based on knowledge attribute i.e. agents can not form team until they have definite knowledge about forming a team. A team agent is a system the behavior of which is directed towards implementing a specific state of the world i.e. is a goal-governed system. A team event occurs when two or more agents intentionally achieve a common goal, that is, a goal with regard to which the agents depend on one another. The key here is to recognize that when an agent participates in a team activity, and hence the agent must be provided with explicit model of teamwork. In other words, a team event is multi-agents plan i.e. a plan that necessarily requires more than one agent for it to be accomplished successfully. In its fullest sense, a team plan occurs when agents have definite knowledge that they depend on one another to achieve one and the same goal. Unfortunately, in implemented multiagent systems, team activities and the underlying model of teamwork is often not represented explicitly. Instead, individual agents are often provided team plans to achieve team goals, with detailed precompiled plans for coordination and communication.

Fire World Domain

We have implemented our team ideas on a simulation of fire world FFAGENTS using a virtual research campus. The idea of simulated fire world was first given in Phoenix (Cohen et al. 1989), which uses real time, adaptive planner that manages forest fires in simulated environment. The virtual campus is implemented using C++ on Windows95/NT platform, where more than 40 agents share the world via network. Part of this world that is related to our example is shown in figure 1. FFAGENTS is a dynamic, distributed, interactive, simulated fire environment where agents are working together to solve problems, for example, rescuing victims and extinguishing fire. The fire world FFAGENTS that we have considered in this paper consists of a large number of objects (of the order of hundreds) and several agents. Agents are autonomous and heterogeneous. Objects in the fire world include walls, buildings, furniture, open areas and LPG gas tanks. Our world is different from others' (like Air Combat (Tambe 1997) and RoboCup (Kitano et al. 1995) in respect that problems posed to the agents and the changes in the environment are not only caused by the actions of other agents but also by the changes the objects themselves undergo in the world (caused by the fire). In a world such as this, no agent can have full knowledge of the whole world. Humans and animals in the fire world are modelled as autonomous and heterogeneous agents. While the animals run away from fire instinctively, the fire fighters can tackle and extinguish fire and the victims escape from fire in an intelligent fashion. An agent responds to fire at different levels. At the lower level, the agent burns like any object, such as chair. At the higher level, the agent reacts to fire by quickly performing actions, generating goals and achieving goals through plan execution. This world contains all the significant features of a dynamic environment and thus serves as a suitable domain for our team agents. Agents operating in the domain face a high level of uncertainty caused by the fire. Agents in the fire domain do not face the real time constraints as in other domains, where certain tasks have to be finished within the certain time. However, because of the hostile nature of the fire, there is strong motivation for an agent to complete a given goal as soon as possible.

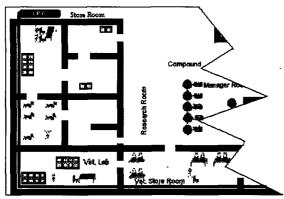


Figure 1. Part of Fire World

Explicit Model of Teamwork

The problem of modeling the activity of team of agents is a combination of two sub problems: the first is the modeling of the team itself (Tambe 97) and the second is the modelling of the team activity (Grosz and Kraus 96). In this paper, we have attempted to model both the team and its activities on a structure called time line structure (TLS). Instead of relying on *joint intentions* framework (Cohen and Levesque 91), we argue that team is an attribute of participating agents' i.e. each agent is individually aware of team, and "consciously" supports the team as an abstract entity by performing appropriate team related activities. In particular, each agent performs the following types of activities:

- Team task related activities: this relates to achieving tasks assigned to the agent by the team.
- Team definition activities: this relates to the activities that the agent has to perform in order to implement the definition of team from the individual agent's view point.
- Monitoring activities: This relates to activities the agent performs for monitoring its own as well as other agent's activities in order to dynamically control its commitments to the above type of activities.

Thus, in our team model, each agent deliberately supports the team. Every agent in team is "conscious "of its own role in the team activity.

Achieving the monitoring behaviour requires two components: To monitor any activity x, we thus need,

- 1. a model of x
- 2. a set of rules that satisfy the monitoring behaviour of the agent.

According to us, the behavior of agents in a team can be further divided into following categories:

- Team definition issues (DEF_T)
- Team activity itself (α_T)
- Attitude of team agents towards the team activity(K_T)
- Team Reactive issues(R_T)
- Team maintenance issues(MNT_T)

The team definition issues contributes to the activities agents do maintain the team definition. The team activity consists of the actual activity the agents perform to solve a team goal e.g. in our domain the team activities are put out fire, move chemicals etc. The attitude of team towards the team contributes to the communication issues and operations done on the team activity itself. It means team agents should be able to answer questions regarding the team e.g.

- Are you in a team?
- How many members are there in the team?
- What is the team task or team goal?
- What is the team state?

The operations done on the team activity includes modify, update, suspend the team activity etc. It means the agents should be able to update, modify suspend their team plan if the situation in a dynamic environment changes. The team reactive issues revolve around the survival needs of the team agent i.e. when fire is large the team will run away from the fire. The team maintenance behavior requires what the agent should do so that team does not disintegrate. For example, when a company of firefighters put out fire together, each team member is well aware of the team activity – each individual is not merely putting out fire on its own, while coordinating with others. In order to maintain the team, each agent in the team will remain within a safe distance from the other members of the team.

Team Agent Architecture

The behaviour of agents in a team is a reasonably consistent pattern based on the dynamic integration of the sum total of an agent's knowledge, beliefs, goals, intentions and plans. The agent architecture shown in figure 2 is designed for intelligent agents executing cooperative team goals in a multi-agent environment. To work cooperatively with other agents, a team agent must have a self-model (Physical and Mental State of agent) and model of other agent. In our architecture, we propose four modules for a team agent, team belief module (TBM), team plan module (TPM), team goal module (TGM), and time line and intention module (TLIM). The team belief module (TBM) consists of facts that the agent believes to be true about the world. Beliefs in general include beliefs about goals, plans, beliefs about other agents' beliefs and intentions. A maturing team moves toward becoming a social system in the sense that it is a group of people who interact with an accepted structure in order to achieve a goal. Thus a major function of team beliefs is to provide the necessary background knowledge in team plan execution, so that a flexible, efficient and

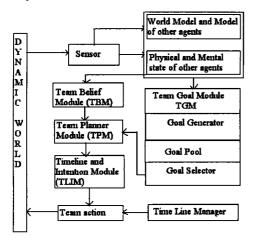


Fig. 2. Team Agent Architecture

coherent teamwork. In team goal module (TGM) module, the goal generator uses different types of rules to generate the different goals from goal pool. The team plan module (TPM) specifies how a goal can be decomposed into team activities. The set of such plans for an agent is also called the plan library. A team can execute a plan only if all its members have the same plan. The purpose of a time line and intention module (TLIM) is to schedule the team activities. All the team activities are scheduled in a particular sequence or pattern. A team agent picks up the most recent goal/activity, expands it dynamically into low level details and inserts them into time line for execution. When agents form a team, problems emerge regarding the representation and execution of actions. Thus the time line module helps to formalize complex team plans into individual team actions.

Representing Plans for Team Activity

Our team-planning module is combination of both *reactive and deliberative planning*. The dynamic nature of domain and issues of replanning forces the agents to plan reactively. Also the idea of team agents making decision requires a greater focus on deliberation than reactivity. The more focus on deliberation has led us to

draw substantially from the classical planning literature. Our team planner uses hierarchical task-decomposition technique for the team task decomposition by decomposing abstract tasks into a set of sub team actions. A team plan is a way or approach that the team agents choose and agree on to do their task and achieve their goal. It is both an abstract structure /recipe and a set of complex mental attitudes (Grosz and Kraus 1996)(Rao 1997). Thus team tasks may be abstract or primitive. We represent plans as Time Line Structures (TLS). Traditionally, time line structures are viewed as a collection of primitive actions, which are temporally ordered. Planning involves the construction of a time line, which gives detail of sequential list of actions, which guide from an initial state to final desired state. Suppose there are two agents (A and B) in a team. The time line structure for the team as well as of the agents is shown in figure 4. The team action T_1 is further divided two actions a_1 and b_1 . Similarly team action T_2 is divided into actions a_2 and b_2 . The team action T_n consists of future actions a_n and b_n , which are shown dotted because of unpredictability of the environment.

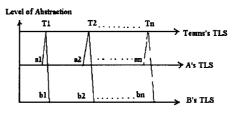


Fig. 3. Time Line Structure

However, for agents residing in a large and highly dynamic world, the time line structure enumerating all basic actions becomes too long and is difficult to reason with and maintain. Moreover, the agent often has only a partial knowledge of the world and its action execution capabilities, and it may not always be possible to fully develop time lines to their finest details. Agents operating in real-world applications must be able to notice and respond to unplanned situations. The range of possible events in the world is vast, while an agent is constrained by limited resources and limited computational speed. Because of these limitations, the agent is unable to build complete plans within arbitrary deadlines. Thus in most of real world applications, a team usually has only a partial plan at the initial stage because of dynamics and unpredictability of agent's environments. To deal with such reactive situations, we suggest incorporation of IF-THEN rules in the partial plan (Ambros-Ingerson and Steel 1988). This problem solving strategy provides a framework that allows interleaving planning and execution. Thus replanning may be triggered in response to unexpected events. Thus the planning in real world is a dynamic process. A team plan specifies a series of team activities that the agent needs to perform in order to achieve or maintain the team goal. The team activities within the team plan must eventually rest on actions of individual member's or subgroup; the collaborative team actions are usually highly abstract and consist of actions for individual members or subgroups (Figure 3). Thus a solution to a team problem can be specified in three ways:

- □ Plan-Execute Method: It involves deriving a plan, and executing the plan. This approach is used when the world is reasonably stable during planning and plans execution.
- (i Behavioral Method: It involves executing a set of rules. This approach is used when the world is changing too fast to do planning.
- E Combination Method: It involves adapting a combination of the above two methods.

When a new fire is reported, an agent retrieves an appropriate plan from the plan library and places it on the timeline. Information about the resources and sensory input are stored in the world model and is used to help the team agent to select the appropriate plan. At any time during this process, sensory data can trigger reactive actions, which will cause the team planner to modify its plan (such as moving away from the fire ahead) by incorporation of some rules. Following are some of the rules that specify the team activity at the higher level (stated in the goal generator module).

TeamSense (MediumFire) -> [TeamClearObjects(MediumFire),TeamCh oose(SubFire), TeamPutOut(Subfire)] TeamSense (SmallFire) -> [TeamChoose(Ai), AgentExec(Ai, SmallFire)] TeamSense (NoFire) -> [TeamTerminate (AllActivity, TeamGoOut]

Fig. 4. : Solution for putting out fire (Behavioral method)

The right hand side of each rule is in the form of a plan. This plan must be loaded on the TLS and executed. But each action the plan is actually a complex goal. We need to specify solution for each goal the way we did for the goal *TeamPutOut (fire)*. The plan based solution for *TeamPutOut(subfire)* for a team of two agents (each has the capability of putting out fire in one square) will be like shown in figure 5.

1.TeamPlan=[TeamPutOut(sq1,sq2),TeamPu
tOut(sq3,sq4)...]

2.TeamExecute(TeamPlan)

Fig.5. Solution for putting out fire (Plan based method)

Implementation

We have developed implementation and experimentation in fire fighting in virtual research campus to verify our ideas about the team problem solving. Here we describe the approach taken to the modeling of teams and team behavior a part of the modeling of fire fighting missions. FFAGENTS is a dynamic, distributed, interactive, simulated fire environment where agents are working together to solve problems, for example, rescuing victims and extinguishing the fire. Our experiments concentrate on the evaluation of the significant level of coordination among team members. Agents based on our approach in our domain contain 40 plans and 1000 rules, with roughly 20% dedicated to our explicit model of teamwork. In our experiment, a team of fire-fighting agents (FF-Team) consists of one company officer, and two sub teams of two fire fighters each. The company officer is the leader of the team and consequently makes decisions for tactical deployment while assisting and supervising individual members of the company. The fire fighters perform all the tasks necessary to put out fire and to control an emergency situation. These tasks may include operating the hose-line and performing search and rescue operations. When a fire is reported, a goal called "Put-Out-Fire" is generated by the goal generator. The goal generator then retrieves a corresponding plan from the plan library and passes it to the time line manager. The time line manager loads the appropriate plan on the time line. There are five kinds of actions that can be differentiated by their effects on the timeline. The team time line contains the actions for team definition (DEF_T), team maintenance (MNT_T), team reactivity (R_T), team activity (α_T) itself and team knowledge (K_T) or team attitude. Figure 6 illustrates that our current implementation provides significant insight of the activities the team does in order to achieve a particular goal.

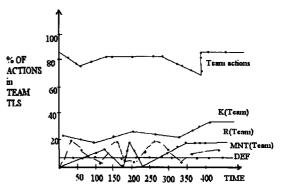


Fig. 6. Percent of team activities in team's timeline structure

Related Work

One of the most popular and contemporaneous approaches of teams is that, the joint action by a team does not depend on simultaneous and individual actions, but it is based on team goals, mutual beliefs and joint commitment (Cohen and Levesque 1991). Other popular theories in this area are based on SharedPlan (Grosz and Kraus 1996) and joint responsibility (Jennings 1995). Teamwork is becoming increasingly popular in multiagent environments, such as, real-world battlefield simulator (Tambe 1997), industrial control (Jennings 1995), and synthetic RoboCup soccer (Kitano et al. 1997). According to the theory of SharedPlan, in which the team members are required not only to be committed to the success of the team achieving the goal, but also follow the team plan when doing so. However, joint intention and SharedPlan alone are not sufficient for the successful execution of the team plan, but a team should have attributes like team maintenance, team attitude and team knowledge. In applications, teamwork model (the STEAM system) has been successfully applied in the virtual military training and RoboCup soccer game domains (Kitano et al. 1997). The agents in STEAM system seem to be more homogenous in terms of their capabilities. The hierarchy of operators /strategies is known to every agent in the system, including the opponents. There is little chance for agents to deviate from the standard, well-defined team strategy. Nick Jennings applied agent teamwork in the industrial electricity management system (Jennings 1995) using a framework called joint responsibility based on joint commitment to a joint goal, and joint commitment to a common recipe. These two types of joint commitments (a variation of joint intention) are claimed necessary because different actions are taken when the joint commitments are dropped. However joint responsibility model seems to be limited to a two level hierarchy of a joint team goal and a joint team plan, which is different from our approach.

Conclusion

Concluding our work in this paper, we explicitly present a complete teamwork model, which is used to coordinate agent activities and solve problems in unpredictable, hostile and dynamic environments. In the multiagent problem solving an explicit representation of teams is required. The agents must know what they are doing and correspondingly behave reactively and deliberatively. According to us, the team is a behaviour that an agent maintains towards some of its fellow agents. We have given a definition of team, which provides flexible, coherent and collaborative teamwork in a multi-agent dynamic world. It is believed that in a dynamic domain where agents are autonomous and heterogeneous, there is a need for timeline structures, which contain both team plans and team rules. In addition, there is also the necessity of introducing individual plans at certain level of abstraction into the team plan. We argue that because

References

1. Ambros-Ingerson, J.A., and Steel, S 1988. Integrating Planning, Execution and Monitoring. In *Proceedings of the National Conference on Artificial Intelligence*.

2. Cohen, P.R.; Greenberg, M.L.; Hart, D.M.; and Howe, A.E. 1989. Trial by fire: Understanding the design requirements for agents in complex environments. *AI Magazine*, 10(3).

3. Cohen, P.R., and Levesque, H.J. 1991.Teamwork. Nous, 35.

4. Grosz, B.J., and Kraus, S. Collaborative Plans for Complex Group Actions. *Artificial Intelligence* 86 (1996), pp269-357, 1996.

5. Jennings, N 1995. Controlling Cooperative problem solving in Industrial Multi-agent Systems using joint intentions, *Artificial Intelligence*.

6. Kitano, H.; Asada, M.; Kuniyoshi, Y.; Noda, I.; Osawa,E. 1995. Robocup: The robot world cup initiative. In Proceedings of the IJCAI-95 Workshop on Entertainment and AI/Alife.

7. Rao, A.S. 1997. A unified View of Plans as recipes. Technical Report 77. The Australian Artificial Intelligence Institute.

8. Tambe, M. 1997. Agent architectures for flexible, practical teamwork. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*.