Towards A Role-Based Framework in Agent Teams

Yu Zhang
Department of Computer Science
Trinity University
San Antonio, TX 78212
yzhang@cs.trinity.edu

Sen Cao and Richard A. Volz
Department of Computer Science
Texas A&M University
College Station, TX 77843
{sencao, volz}@cs.tamu.edu

Abstract

Teamwork has become increasingly important in many disciplines. However, existing agent teamwork systems are short of expressivity of flexible and efficient mental states underlying teamwork. This paper models and analyzes three foundation concepts towards role-based teamwork modeling: position, role and role variable. An important characteristic of our notion of role is that it includes the specific operations that an agent filling the role must perform in a specific setting as well as their ordering constraints. Taking this view on role definition will enable us to reason about the assignment of real agents to roles, to talk about the responsibility of roles, to express automated helping behaviors and to achieve a certain level of plan reusability.

Keywords

Agent modeling, Teamwork, Role, Plan

1. Introduction

Teamwork in distributed and dynamic domains has become increasingly important in many disciplines where each autonomous team member works cooperatively to solve a part of a problem in parallel [8]. Teamwork theories mainly concern the properties of joint actions and mathematically formalize the representation, evolution and reasoning of joint actions, particularly the shared mental states driving team members to act together and the interaction leading their individual actions to team efforts [2]. However, existing agent teamwork systems are short of expressivity of mental states underlying teamwork, and existing representation of the shared mental models are inefficient and further become an obstacle to support effective teamwork [4, 6, 7, 11].

To address these issues, we need some concepts to help us organize team activities. Such concepts need to meet two basic goals in order to reach flexible and efficient teamwork. First, the concepts should allow us to specify team activities in abstract level so that the knowledge about the team activities can be used by different agents. Second, the concepts should help us be able to easily decompose team activities into individual activities and delegate the individual activities to agents so that team activities can be actually executed.

The property of roles makes roles suitable to help us to realize the first goal [1, 3, 10]. According to Webster Dictionary, the concept of role is generally but loosely understood as a character assigned or assumed, a socially expected behavior pattern usually determined by an individual's status in a particular society, a part played by an actor or singer, a function or part performed especially in a particular operation, or an identifier attached to an index term to show functional relationships between terms. This understanding implies an attractive property of roles – a role stands for a functional abstraction, independent of the specific person who plays it.

In order to realize the second goal by using role, we need to define the concepts to capture the behavioral aspects of the general term of role. We ground our definitions on Biddle and Thomas' classificatory concepts of role, some of which capture the behavioral aspects of role, in their role theory [1]. We sort out those aspects and come up with three concepts, including position, role and role variable, to characterize behavioral aspects of the general term of role. While behaviors in Biddle and Thomas' classificatory concepts of role could be complex actions, we take Grosz and Kraus' view that complex actions must be rested eventually in the primitive operations of individual agents [5]. Therefore, our formalisms are based on primitive individual actions. We propose a mechanism to break complex actions into individual actions so that our formalisms accommodate complex actions.

This paper is organized as follows. Section 2 describes our understanding to the concept of role in teamwork. Sections 3, 4 and 5 formally define position, role and role variable. Section 6 accommodates complex actions to role and role variable. Section 7 concludes the paper and discusses the future work.

2. Understand the Concept of Role in Teamwork

In their role theory, Biddle and Thomas concluded that there are several distinct bases upon which a notion of roles can be defined [1].

- Roles can be defined based on partitioning concepts for behaviors. A role defined by this referent characterizes the exhibit of behaviors (i.e., the ability of behaviors). For example, a role of secretary can be defined as secretary's behaviors: answering phones, receiving faxes, filing documents, and arranging schedules. Although this definition associated roles with behaviors, it does not characterize the performance of the behaviors, i.e., when and why behaviors are performed.
- Roles also can be defined based on partitioning concepts for persons, for example, father and son.
- Roles also can be defined based on partitioning concepts for both persons and their behaviors. The distinction between this referent and the first one is, the behaviors in this referent focus on the dynamic aspects (i.e., the performance of behaviors), and those in the first referent focus on the static aspects (i.e., ability of behaviors). Given a role defined by this referent, its behaviors are combined with a specific person filling the role and following a specific procedure, which leads to concrete performance of (some or all of) the behaviors. Generally speaking, a role defined by this referent is an entity of a position, which executes specific behaviors of the position. For example, a role of a position secretary for a CEO is the one who is associated with specific actions, such as arranging the CEO's schedules by following some specific routine.

The referent of partitioning concept for persons is a non-behavioral criteria and thus not suitable for utilization in a formal role concept suitable for computation. Even though the referent of partitioning concepts for behaviors, alone, does not directly lead to concrete actions, it can characterize the behavior requirements of roles (i.e., interpreted as "job qualifications"). The referent of partitioning concepts for both persons and their behaviors is suitable to our purpose: specifying team activities conceptually and helping decompose and delegate team activities to agents. Therefore, based on the referent of partitioning concepts for behaviors, we define a position as all entities that can perform a set of primitive operations, and use this to characterize a collectively recognized category of persons who are able to exhibit a set of behaviors. Then, based on the referent of partitioning concepts for persons and their behaviors, we define a role as an entity of a position associated with a bag of temporally ordered actions. A role, as an entity of a position, must be capable of the primitive operations of the position. We note that a role is more than an entity of a position, in that it includes an (partial) ordering on a (sub)bag of actions in the position.

Biddle and Thomas also pointed out that pre-association with roles is too restrictive [1]. There is yet another level of abstraction which we wish to introduce that mirrors a concern that has arisen in the role theory literature. They suggested that some actions are not predetermined to be firmly associated with specific roles. Rather, they argue that a specific action might be performed by any one of several roles, with the determination of which role actually performs it being determined dynamically in a specific situation.

Our analogy to this is to introduce role variables. We define a role variable as an entity which is dynamically selected from a set of roles to be associated with a bag of temporally ordered actions. Generally speaking, a role variable stands for some role out of a set of roles. Depending on concrete situation, one role from the set is dynamically selected to fill (or assume) the role variable. When this happens, the bag of operations of the assuming role is dynamically expanded.

Next we formally define position, role and role variable.

3. Position

We define a concept of position to refer a collectively recognized category of entities (persons) that exhibit a set of behaviors.

Definition 1. An **entity** is an abstraction of any performer (e.g., agent or person) that is able to perform operations.

Definition 2. A **position** is a named set of all entities that are able to perform a set of primitive operations, denoted by operators. Given a set of operations O, a position based on the operations is

$$P(O)=\{e|e\in \text{Entities} \wedge \forall o\ (o\in O) \wedge Capable(e, o)\}, \quad (1)$$

where e represents an entity and $Capable(e, o)$ means that e is able to perform operation o, assuming the preconditions of o are true.

The purpose of defining position is to capture the capability requirements on agents. Every entity of a position is required to be capable to do the operators defined in the position. For example, *(POSITION sniffer (talk move movein randmove sense selectTarget)).* (2)

A position sniffer is defined by a set of operators, including talk, move, movein, randmove, sense, and selectTarget. Suppose *r1* is an entity that takes on the role sniffer; *r1* should be able to perform the above operators.

4. Role

We base our definition for a role on a position. In other words, any action (i.e., primitive operation) associated with the role must be in the operation set of the position. Let O be a bag of operations that must be performed by the same entity of a position, $COND$ be a bag of conditional operators where each action is contingent on a conjunction of conditions, and C_O be a bag of ordering constraints that impose a "temporal order" on O and the evaluation of the conditions in $COND$. Let P be a position and O_P the qualification of P. We define a role r as an abstraction of the entity of position P that satisfies the constraints.

Definition 3. A **role** is an abstraction of an entity that performs a specific bag of operations and includes temporal constraints on the order in which the operations may be performed:

$$r = (id, P, O, COND, C_O), \quad (3)$$

where id is the name of r and refers to the entity of P that must perform the operations, $O \subseteq O_P$, C_O is a set of temporal orders as (o_i, o_j) where $o_i, o_j \in O \cup COND \cup S$, and $S = \{o_s, o_e\}$. o_s is a dummy starting operation that can be performed by any entity of a position, and o_e is a dummy ending operations that can be performed by any entity of a position.

Temporal orders are transitive. That is, if there are two temporal orders (o_i, o_j) and (o_j, o_k), then (o_i, o_k) is a true temporal order too. However, we note that C_O is not a transitive closure. That means, even though (o_i, o_j) and (o_j, o_k) are in C_O, (o_i, o_k) may not be in C_O. And, cycles may exist in the set of temporal orders C_O. That is, there exist temporal orders as (o_1, o_2), (o_2, o_3), …, (o_k, o_1), where $k \geq 2$. This does not mean a temporal conflict. Rather, it means that the operators may be executed more than once in accordance with the temporal orders. The feature also holds when we talk about the temporal orders in role variables (described later in Section 5).

Operationally, we will declare role names to be associated with positions and then define the processes of team plans in terms of actions performed by the role (names). The operations associated with a role will be derived as a plan is executed. Thus, the specific operations and constraints in O and C_O are dynamically determined as the execution of a system progresses. Roles are thus dynamically determined. However, the static nature of the position underlying a role allows constraints to be imposed that assure that any agent taking on a role will be capable of performing the operations of the role.

In addition to the temporal order applied to the operations of role r, there can be temporal orders between the operations of role r and those of other roles (or role variables, discussed in Section 5). For example, it might be required that operation o associated with role r appear in order before operation o' in role r'. The above definition thus needs to be extended to deal with this situation. Let O' be the union of the sets of operations (in other roles) involved in temporal ordering constraints with operations of the role r (including evaluation of conditions and dummy starting operations). Let $C_{O'}$ be the temporal order formed to express the ordering constraints between the operations in the union of O, $COND$ and S and those in O', that is, C' is a set of temporal orders as (o_i, o_j), where $o_i \in O \cup COND \cup S$ and $o_j \in O'$, or $o_j \in O \cup COND \cup S$ and $o_i \in O'$.

The ordering constraints of role r are then extended to include C'. We use $<_r$ to denote the extended ordering constraints as $<_r = C_O \cup C_{O'}$. Therefore, the definition of a role r is extended to

$$r = (id, P, O, COND, <_r) \quad (4).$$

An important characteristic of our notion of role is that it includes the specific operations that an agent filling the role must perform in a specific setting as well as their ordering constraints.

5. Role Variable

A role variable is similar to a role except that the role variable is dynamically selected from a set of roles. So, in the definition of a role variable, we do not explicitly require it to belong to any specific position. Rather, we require a selection constraint whose satisfaction will select one role out of a set of roles to perform the operations of the role variable in accordance with the specified order. For clarity, we prefix role variables by "*?*".

Definition 4. A **role variable** is an abstraction of an entity that is dynamically selected from a set of roles to be associated with a bag of temporally ordered actions:

$$?r = (?id, O, COND, C_O, RS, SC), \quad (5)$$

where $?id$ is the name of $?r$, RS is the selection scope, i.e., a set of roles, SC is the selection constraint, and for any $r \in RS$ and if $r \in P$, $O \subseteq O_P$.

In the definition of role variable, the first four elements have the same meanings as those (with the same symbols) in the definition of role. They define the actions associated with the role variable, the conditions on the actions, and the temporal orderings on the actions. Moreover, the definition of role variable includes the selection scope *RS* and the selection constraints *SC*. The selection scope *RS* defines all possible roles that can be selected to be the role variable. The selection constraint *SC* defines the conditions that the role must satisfy for being selected to the role variable. To select a role to fill the role variable, all roles in the selection scope coordinate with each other, and the one that satisfies the selection constraint is selected. If more than one role satisfies the selection constraint, they coordinate with each other and make sure one of them is selected (we assume such selection is random). If no role satisfies the selection constraint, then the action associated with the role variable cannot be associated with any role and thus they cannot be executed. This may cause a failure in the context of a set of roles. We assume that users need to ensure that at least one role will satisfy the selection constraint when specifying the role variable, or capture the failure caused by no role satisfying the selection constraint by termination condition. We note that conditions in *COND* have exactly the same semantics as those in the definition of role for defining conditional actions instead for constraining role selections. Through the role variable, the actions associated with the role variable are dynamically associated with the role that is selected to fill the role variable.

Similar to roles, there can be temporal orders between the operations (including evaluation of conditions and dummy starting operations) of the role variable and those of other roles and role variables. Let O_E be the union of the operations of the other involved roles and role variables. A temporal order C_E is formed to express the ordering constraints between the operations in the union of O, $COND$ and S and those in O_E, that is, C_E is a set of temporal orders as (o_i, o_j), where $o_i \in O \cup COND \cup S$ and $o_j \in O_E$, or $o_j \in O \cup COND \cup S$ and $o_i \in O_E$.

It follows that the ordering constraints of role variable $?r$ are extended to include C_E. We use $<?r$ to denote the extended ordering constraints as $<?r = C_O \cup C_E$. Therefore, role variable $?r$ is defined as

$$?r = (?id, O, COND, <_{?r}, RS, SC). \quad (6)$$

6. Accommodating Complex Actions

In the above formalisms, actions associated with roles and role variables are primitive operations. However, behaviors in Biddle and Thomas' classificatory concepts of role can be either primitive operations or complex actions. Correspondingly, actions associated with roles and role variables can be complex actions too. Rather than reformalize our roles and role variables to directly include complex actions, we break complex actions into sub-actions, eventually into primitive operations, and associate the primitive operations to individuals (roles or role variables). While other researchers [9] argue complex actions may include many different aspects and could be specified in other ways, a complex action in which an individual is involved together with other individuals could appear as the following in our teamwork framework:

- Coordinated operations: the operations of a role (or a role variable) are coordinated with the operations of other roles (or role variables). Such coordinated operations could be either team operators or joint do, and modes include AND, OR, and XOR;
- Plan invocations: a role forms a sub-team (with other roles or perhaps just the role itself) to invoke a plan which consists of a process of actions.
- Role selections for role variables: the action(s) associated with a role variable might be performed by any one of a set of roles. The roles need to negotiate to decide which one is selected to fill the role variable and thus perform the actions.

Suppose $r_i = (id_i, P_{ri}, O_{ri}, COND_{ri}, <_{ri})$, $?r_k = (?id_k, P_{?rk}, O_{?rk}, COND_{?rk}, <_{?rk}, RS_{?rk}, SC_{?rk})$, and $r_j = (id_j, P_{rj}, O_{rj}, COND_{rj}, <_{rj})$. Complex actions, including coordinated operations, plan invocations and role selections, are broken eventually into primitive operations and the primitive operations dynamically associated with the involved roles (or role variables) as follows:

- role r_i (or role variable $?r_k$) involves a coordinated operation *co*. We assume there is an individual operator corresponding to each team operator and such an individual operator is associated with each involved role. For example, the individual operator "lift-table" corresponds to team operator "lift-table" and an individual operator "lift-table" is associated with r_i. For a joint do, the individual operators and their association with the involved roles are specified in the joint do statement. Role r_i (or role variable $?r_k$) needs to dynamically coordinate with each other according to its coordination mode to decide how to perform the individual operation *io*. Based on the result of coordination, if r_i needs to perform *co*, then r_i includes *io* in its O and replaces *co* with *io* in its $<_{ri}$;
- role r_i involves plan invocation *a*, which consists of a process of actions in terms of a set of roles, *VT*. The plan invocation *a* is

broken into the actions and eventually into primitive operations. Without losing generality, we can assume the actions are primitive operations. However, it is not decided which part of the actions is associated with r_i (in other words, which role(s) of *VT* are filled by r_i) until involved roles negotiate. Suppose r_i is selected to fill a role r_j in *VT*. The actions associated with r_j are dynamically included into r_i as

$$r_i \rightarrow r_j \Rightarrow r_i = (id_i, P_{ri} \cup P_{rj}, O_{ri} \cup O_{rj}, COND_{ri} \cup COND_{rj}, <_{ri} \cup <_{rj}), \quad (7)$$

where $\rightarrow$ means "is selected to fill". Moreover, after r_i has completed the actions, they should be dynamically removed from r_i. Because r_i and r_j are at two different levels of the plan hierarchy, the bags O_{ri} and O_{rj}, $COND_{ri}$ and $COND_{rj}$, and $<_{ri}$ and $<_{rj}$ do not overlap. Even though they are in the same plan definition, they are in two different plan invocations. So they there is no conflict in their unions. In this way, a plan invocation as a complex action can be eventually decomposed into individual operations and dynamically associated with the involved roles;

- role r_i involves a negotiation regarding action association of role variable $?r_k$. That is, it is not decided to which role the actions of $?r_k$ are associated until involved roles negotiate. Suppose r_i is selected to fill $?r_k$, the actions associated with $?r_k$ are dynamically included into r_i as

$$r_i \rightarrow ?r_k \Rightarrow r_i = (id_i, P_{ri} \cup P_{?rk}, O_{ri} \cup O_{?rk}, COND_{ri} \cup COND_{?rk}, <_{ri} \cup <_{?rk}). \quad (8)$$

It follows that after r_i has completed the actions, they should be dynamically removed from r_i. In this way, any negotiation on action association as a complex action can be expressed by individual operations.

We note that the primitive operations, which the complex actions eventually rest on and are associated with a role, are in the operation set of the position of the role. Also, to break complex actions to sub-actions and eventually to primitive operations, the involved individuals need to negotiate with each other. We assume that all entities can communicate with others by sending and receiving messages so that negotiations can take place.

7. Conclusions and Future Work

We have defined position, role and role variable and accommodated them to complex actions. Based on these conceptual notations, the knowledge of team processes can be specified without having to specific agents and agent variables. This will allow joint mental states to be formed and the teamwork knowledge to be reused by different teams of agents.

In the future, based on roles and role variables, we will develop mechanisms of task decomposition and task delegation, by which the knowledge of a team process is decomposed into the knowledge of a team process for individuals and then delegate it to agents. We will also develop an efficient representation of joint mental states by which agents only maintain individual processes complementary with others' individual process and a low level of overlapping called team organizations.

References

[1] Biddle, B. J. and Thomas, E. J. editors. *Role theory: Concepts and research*. John Wiley & Sons, Inc., New York, London, Sydney, 1966.

[2] Cohen, P. R. and Levesque, H. J., Teamwork. Nous, Special Issue on Cognitive Science and Artificial Intelligence, 25(4):487-512, 1991

[3] Dastani, M., Dignum, V. and Dignum, F. Role Assignment in Open Agent Societies, in Proceedings of AAMAS'03, Melbourne, Australia, 2003.

[4] Fan, X., Yen, J., and Volz, R., A Theoretical Framework on Proactive Information Exchange in Agent Teamwork, Artificial Intelligence, 169 (1): 23-97, 2005.

[5] Grosz, B. and Kraus, S., Collaborative Plans for Complex Group Actions, Artificial Intelligence, 86 (2): 269-357, 1996.

[6] Kinny, D., Ljungberg, M., Rao, A., Tidhar, G., and Werner, E., Planned Team Activity, Technical Notes 31, Australian Artificial Intelligence Institute, 1992.

[7] Stone, P, and Veloso, M., Task Decomposition and Dynamic Role Assignment for Real-Time Strategic Teamwork, Artificial Intelligence, 100 (2), 1999.

[8] Tambe, M., Towards Flexible Teamwork, Journal of artificial intelligence research, 7: 83-124, 1997.

[9] Tuomela, R., On the Structural Aspects of Collective Action and Change by Logic Programs, Theory and Decision 32 (2): 269-357, 1992.

[10] Zhu, H. and Zhou, M.C., "Role-Based Collaborations and their Kernel Mechanisms", IEEE Trans. on Systems, Man and Cybernetics, Part C, 36 (4): 578-589, 2006.

[11] Zhang, Y. and Yin, J, A Role-Based Modeling for Agent Teams, in Proceedings of the IEEE Workshop on Distributed Intelligent Systems (DIS'06), Prague, Czech Republic, June 15–16, 2006, pp. .348-353.