Multi-Agent Information Dependence

Xiaocong Fan, Rui Wang, Bingjun Sun, Shuang Sun, John Yen*
School of Information Sciences and Technology
The Pennsylvania State University
University Park, PA 16802
{zfan,rwang,ssun,jyen}@ist.psu.edu, bsun@cse.psu.edu

Abstract

Recently, there has been an increasing interest in reasoning about multi-agent information dependence, which is very important for supporting social reasoning in multi-agent cooperations. In this paper we seek to characterize the nature of multi-agent information dependence in general, and investigate ways of using information dependence knowledge in agent teamwork settings. We also describe a tool that can facilitate humans to dynamically manipulate information dependence.

1. Introduction

As a new paradigm for conceptualizing, designing, and implementing software systems, multi-agent systems have been successfully used in a variety of areas including applications for distributed situation awareness and assessment, distributed resource planning, and collaborative information processing. In these domains, intelligent agents are often required to analyze voluminous amounts of dynamic information and make decisions in a timely manner. Being restricted in information accessibility and limited in knowledge for interpreting information, it is critical for the agents to be able to effectively collaborate with each other.

Recently, there has been an increasing interest in reasoning about information dependence in multi-agent systems [11, 13, 8]. Sichman, Conte, et al. [9, 8] proposed a dependence theory, where an agent can use the information regarding others' goals, actions, resources and plans to construct dependence networks and dependence graphs that may hold among two or more agents. From the constraints (i.e., plan preconditions, preference conditions, termination conditions) specified in shared teamwork processes, the CAST [13] system can extract information requirements, which enable CAST agents to proactively deliver information to teammates without being asked. Explicitly declared information-dependency relationships among actions are used in the STEAM agent architecture [10] to derive inter-agent communication. Similarly, in the TAEMS framework [3, 4], interdependency between agents (i.e., links to non-local effects) is the driver behind coordination.

Although the aforementioned researches have shown from both theory and practice that the notion of information dependence is very important for supporting social reasoning in multi-agent cooperations, the existing approaches have been limited in several ways. For example, the dependence framework [9] focuses on action and resource dependences emerging in carrying on a certain plan. For this reason, the dependence framework does not pay much attention to the dynamics of dependence relations, assuming the models of others are fixed once established. Moreover, few of the existing approaches consider the situations where dependence knowledge may be distributed among a group of agents, or explicitly support the reasoning of indirect information needs (the notion of indirect information needs is analogous with indirect speech acts [7]). To complicate the issue further, the growth of human-centered teamwork (e.g., [1]) calls for the ability to support adjustable autonomy. Allowing human users to directly manipulate information dependence certainly offers an opportunity to foster, and investigate the nature of, mixed-initiative information flow guided by human needs. Even though recently CAST has been extended to somehow support the dynamic reasoning of information dependence [5] and indirect information dependence [12], the issue of human-manipulable information dependence has been ignored.

Hence, the objective of this research is three-fold: (a) to understand the nature of multi-agent information dependence in general, (b) to investigate ways of using information dependence knowledge, and (c) to develop a tool that

* Supported by AFOSR MURI grant No. F49620-00-1-0326.

facilitates humans to dynamically manipulate information dependence. The remainder of this paper is organized as follows. In Section 2 we discuss how to establish static and dynamic information dependence. Section 3 centers on the different uses of information dependence knowledge. Section 4 describes a tool for supporting human-manipulable information dependence, and Section 5 summarizes the paper.

2. Establishing Information Dependence

Most multi-agent dependence can be reduced to information dependence (information processing and sharing), resource dependence (resource producing and consuming), organizational or functional dependence (service delegating and providing). Multi-agent information dependence captures the epistemical asymmetry among agents in performing complex tasks or in pursuing certain desired state of affairs. Information dependence may stem from

- Domain rules: each rule establishes a causal link among information with certain patterns. For example, rule $p_1 \wedge p_2 \rightarrow r$ means information of type r depends on information of types p_1 and p_2;
- Information use/production: a piece of information has to be available before it is used. For example, prior to performing an action, an agent typically needs to check whether the precondition of the action is satisfiable. This kind of dependence can be further complicated by the division of information consumers and providers. For example, the Joint Intention theory [2] implies that all agents in a team depend on whoever first detects the termination of a joint activity;
- Information contexts: in cases where a type of information is only meaningful under certain context, the information is semantically dependent on the corresponding context. For example, $Intend(A, p, C)$ means agent A intends p hold relative to context C. Upon knowing C is invalid, A will be freed from its commitment to p.

In the following, we consider static dependence derived from inference rules and dynamically evolved dependence.

2.1. Static Dependence From Inference Rules

Information dependence can be easily extracted from *a priori* inference rules ascribed to an agent. The notion of inference tree was introduced in [12] to facilitate the reasoning about multi-level information dependence. Inference trees are similar to traditional AND/OR trees, except that in an inference tree, predicates at the same level can collectively form an information-use context. Every predicate (which denotes a type of information) in an information-use context is *weakly* dependent on the rest: a piece of information becomes meaningless when it is not used together with other information required by a context. In other words, to make sense of a piece of information, an agent has to seek ways to acquire missing information sufficient enough to interpret the current situation. Figure 4 shows an example inference tree.

However, to effectively use inference trees in dynamic, complex domains where domain knowledge and expertise are distributed among multiple agents, we need to reconsider the underpinning principles.

First, generally a team of agents may have overlapping or even disjoint sets of inference rules. Such distribution of inference knowledge necessitates information-needs anticipation and information exchange at multiple levels. To fully leverage the intelligence within a team, it is desirable to capture the information regarding "who has what inference knowledge[1]" in an inference tree. To do this, each AND node in an inference tree can be associated with a list of agents who share the corresponding inference rule. Such a list can facilitate inter-agent communication. For instance, knowing a teammate has the same inference knowledge, an agent may not consider the teammate as a needer of the inferred information unless being explicitly requested. Moreover, the agent list actually provides points of contact between information needers and information providers. When the information need of a teammate is matched with the predicate labeling the parent of an AND node, an agent can consider initiating a third-party communicative action towards some potential provider in the agent list.

Second, a team of agents may have different information acquisition capabilities (i.e., observability). In other words, agents playing a certain role may be sensitive to only specific kinds of information. Although whether an agent sensitive to certain information can actually acquire that information is situation-dependent, to have agents' observability shared at least can help information needers figure out the potential contact points. To do this, each leaf node can be associated with a list of agents who can observe the information relevant to the predicate labeling the node.

Third, in complex domains, a predicate may be derivable from several different ways (i.e., a predicate is the consequent of multiple rules). In an inference tree, this is reflected by 'OR' nodes having multiple 'AND' nodes as sons (see the *threatHigh* node in Fig. 4 as an example). In practice, rules with the same predicate as their consequent may have different degrees of usability. For instance, given
$Rule\ r_1:\ p_1(?x, ?y) \wedge p_2(?y, ?z) \rightarrow s(?x, ?z)$, and
$Rule\ r_2:\ q_1(?x, ?y) \wedge q_2(?y, ?z) \wedge q_3(?x, ?z) \rightarrow s(?x, ?z)$,
Rule r_2 may be used in 90% of times because of the high

1 Normally, this can be treated as common knowledge and prescribed in team profiles.

availability of information of type q_1, q_2 and q_3. Other things being equal, an agent may prefer to use Rule r_1 because it probably implies less inter-agent dependence since r_1 has fewer antecedents. To guarantee the success of acquiring the critical information s, an agent may prefer to use Rule r_2 if the subtree starting from r_2 contains more ways of deriving s, say, most of the OR nodes (q_i and their antecedents) have multiple branches. Thus, each AND node of an inference tree can be associated with a dynamically adjustable preference value. Such preference information is useful in circumscribing the scope of reasoning in anticipating others' information needs, and can be leveraged to govern agents' information gathering and fusing behaviors.

2.2. Dynamically Evolved Dependence

For dynamically evolved information dependence, we consider three cases: (a) human adjustable: human users adjust the existing dependence, (b) time adjustable: agents refine information dependence as time proceeds, and (c) *a posteriori*: agents learn from the actual information use.

2.2.1. Human manipulation Inference knowledge originally comes from human experts. It is thus desirable to consider ways by which humans can adjust the flaws in the existing information dependence so that afterward agents can use the updated dependence to better serve other agents and human users.

Interacting with agents on information dependence may stepwisely help people elicit the tacit knowledge that may reside solely in their minds, and transform it into explicit knowledge that can be effectively used by agents for various purposes. In Section 4 we will describe InWit–a tool that allows human users to manipulate the existing information dependence.

2.2.2. Learned from information use Learning techniques can be employed to learn information dependences that are too complex to be easily articulated as explicit knowledge. We can frame two problems.
Problem 1: information type q depends on a collection $P = \{p_1, p_2, \cdots, p_i\}$ of information types, but the exact dependences among $P \cup \{q\}$ are unknown. The goal is to learn the tacit dependences.
Problem 2: information of type q is directly derivable from a set $R = \{r_1, r_2, \cdots, r_k\}$ of rules; each rule embodies a certain degree of dependence, which is unknown. The goal is to learn the precedence order of the rules in R.

For problem 1, consider the cases where P gives a portion of situation description (e.g., p_i is directly observable from the environment or communicated from another agent), and the use of information of type q is directly connected with certain performance variables. Suppose an agent has to use (e.g., making decisions) information of type q under some timing constraints, and oftentimes not all the information described in P are available when they are needed. In other words, the agent may need to make certain assumptions (say, on the values of certain arguments of q) in a timely manner to determine a complete information of type q based on the partial description of the current situation. For each information use of q, the agent can incrementally modify the rules corresponding to q based on the evaluation of the relevant performance variables.

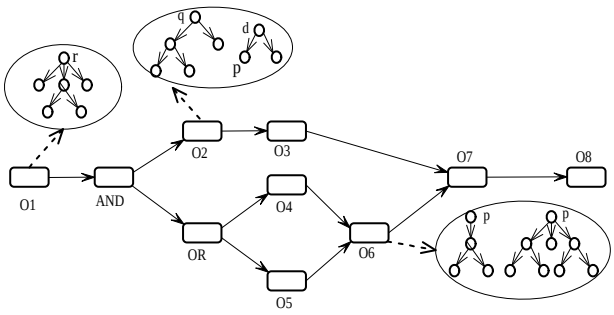

Figure 1: Dependence depends on work progress

For example, suppose initially it is known that the threat level information $Threat(?e, ?l)$ wrt. an enemy unit $?e$ may depend on the size, the moving direction and the distance of $?e$, the terrain information, etc. Assume at situation s_i, the following rule holds for all the previous episodes:
$Enemy(?e, < 10) \wedge Direction(?e, ?d) \rightarrow Threat(?e, Low)$.
Now, facing a new episode: enemy unit $e3$ with 9 members is close to a target and is approaching the target; the agent treated the threat from $e3$ as low but ended with negative performance (i.e., the actual threat is high). After this episode, the above rule is modified as: $Enemy(?e, < 10) \wedge Direction(?e, ?d) \wedge Distance(?e, Far) \rightarrow Threat(?e, Low)$.
Suppose, facing another episode: enemy unit $e9$ with 8 members is close to a target and is leaving the target, the agent treated the threat from $e9$ as low but ended with negative performance (i.e., the actual case is no threat). After this episode, the above rule is modified as: $Enemy(?e, < 10) \wedge Direction(?e, Approaching) \wedge Distance(?e, Far) \rightarrow Threat(?e, Low)$. Such a case-based learning process can thus be used to discover tacit information dependences.

Similarly, reinforcement learning can be used to learn the degrees of dependence embodied in different rules.

2.2.3. Changing as work flows We have considered the cases where information dependence may be changed as a result of learning or human manipulation. In addition, the collection of information dependence that an agent needs to consider may also changes as its activity proceeds.

Consider the abstract workflow shown in Fig. 1, where boxes represent operations or flow control-points: for an AND point, all branches from it must be done (probably

by different actors), but for an OR point, only one of its branches is required to be done. Each operation may be associated with certain constraints—some describes preconditions for executing the operation, some describes under what situation the execution must be terminated. These constraints can be represented as predicates, which in turn can be inferred from low-level information in different ways (rules). The information dependences derived from the constraints of an operation are clustered in an oval.

Intuitively, an agent no longer needs to consider dependence r associated with $O1$ after $O1$ is done (assuming all the other operations have nothing to do with r). When an agent chooses to do $O5$, those dependences associated with $O4$ should be out of concern. However, it would be worthwhile for an agent executing $O6$ to consider those dependences associated with $O2$ and $O3$ because another agent may be waiting for help (in this example, d depends on p). To complicate this issue further, each operation may be a complex subprocess in itself. Then, collecting the set of information dependences that reflects the current work progress will involve two-dimensional exploration.

One reification of this idea is the notion of information-needs graphs [5], which is used by an agent to progressively reconsider teammates' active information needs. Fig. 2 illustrates how an agent's attention regarding information dependence is partitioned by the work trace. Those dependences associated with the nodes (and the subtrees) on the left (wrt. the sibling relation) of the work trace are past attentions; those on the right of a trace node are dependences should be considered. In addition, to encourage helping behaviors among agents, those associated with the son nodes (and the subtrees) of an H-node are all considered. Since there is no sibling relation among the sons of a C-node, no other branches except the chosen one on the work trace are taken into consideration. Here, in essence, 'work progress' is leveraged as a context to temporally cut away inactive information dependences.

3. The Use of Information Dependence

It has been shown [12] that information dependence can be leveraged to fuse information at an appropriate level so that an agent can help its teammates, who only have limited cognitive capacity, with information that is in a form closest to their needs. Here, we explore two other uses of information dependence knowledge. For each use, we consider the cases where the dependence knowledge is fully shared, partly shared, and not shared among a group of agents.

3.1. Deriving Indirect Information Needs

An information need is an explicitly-represented epistemic requirement [12]. Predicate $p(\cdots)$ is an indirect information need wrt. information need $q(\cdots)$, if p plays certain role in deriving q. For the example in Fig.3, if agent A has an information need $p_0(?x, ?z)$, all the predicates labeling the other nodes in the dependence tree are indirectly needed by A. We now examine how an agent can leverage its knowledge of information dependence to anticipate other teammates' indirect information needs.

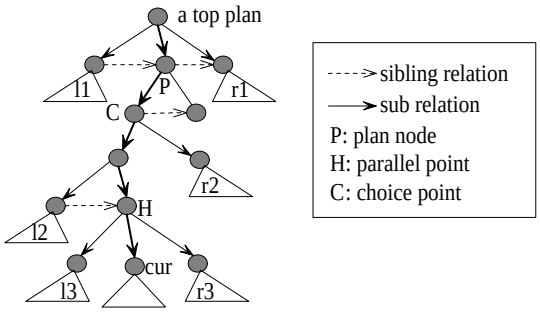

Figure 2: Attention partitioned by work trace

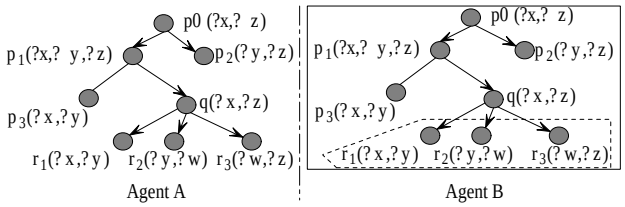

Figure 3: Case 1: the dependence tree is shared by agents A and B; Case 2: partly shared (agent B is unaware of the dotted part); Case 3: not shared (agent B is unaware of the dependence inside the box)

Case 1: Agents A and B have the same dependence knowledge regarding p_0. In this case, the collection of A's indirect information needs wrt. p_0 is the same as what B may derive from the dependence tree. However, in complex cases where a predicate (say, p_1) has different dependence relations (e.g., having multiple "AND" sons like in Fig.4), what A *really* needs and A's needs anticipated by B may not exactly match if A and B assign different preferences to the dependence relations. When communication cost is not neglectable, it is important for B to prioritize A's indirect needs in a way approximately honoring A's preference. Moreover, to restrict the scope of needs monitoring and to reduce unnecessary inter-agent communication, B also needs to consider the potential impact of the specialization of A's information needs. For example, when A's need becomes more concrete as the team activity proceeds, say, A becomes specially interest in information of form $p_0(a_5, ?z)$, B should rationally respond to such a change by refining the collection of indirect needs of A. Here, the specialization $(?x : a_5)$ serves as a *cue* for B to distinguish

A's relevant needs from those that are no longer relevant. Clearly, these two behaviors can be achieved by allowing A and B to exchange meta-information regarding A's changes of interest and preference.

Case 2: Agents A and B have overlapping dependence knowledge regarding p_0. In this case, the two agents have different perspectives on A's indirect information needs wrt. p_0. Accordingly, B may consider more than what A actually needs, sending information of no use to A; or, being unaware of the information use context that A is considering, B cannot take full advantage of the available information. For the example in Fig.3, B will not take r_1, r_2 and r_3 as A's indirect needs, not knowing they can be fused by A to derive q. Further, suppose in Fig.3, in addition to the rule of deriving p_1 from p_3 and q, A has another rule—deriving p_1 from r_1 and p_4—of which B is unknown. Due to the high unavailability of p_3, agent A has never actually used information q. Unfortunately, believing A indirectly needs q, B keeps sending information of type q to A. To overcome these limitations, one solution is to design conversation protocols that allow agents to attach information use contexts with the information being exchanged.

Case 3: Agent B has no dependence knowledge regarding p_0. This is the most limited case where agent B knows nothing about A's indirect needs wrt. p_0. The case can reduce to case 2 if B could learn A's dependence knowledge from the contexts of the information they have exchanged.

3.2. Reasoning about Incomplete Information

Here, by *incomplete information* we refer to any information whose meaning is only partially determined. Incomplete information can be represented as predicates with unbound arguments, or predicates tagged with assumptions (expectancy). For example, $hasThreat(e5, area_4, northeast, ?num)$ is a piece of incomplete information, which means the enemy unit $e5$ with unknown number of enemies is now in area 4 and moving northeast. Predicate $q(a,b)_{r_2(?y,?w)}$[2] is an incomplete information, which states that $q(a,b)$ is likely to be true, assuming some desired information of type r_2 becomes available later. Incomplete information is of special significance in multi-agent systems. For instance, in generating shared plans [6], agents need to exchange incomplete information to identify parameters for team activities.

An agent can leverage the information dependence knowledge to generate incomplete information by initiating appropriate queries to its belief base. In addition, an agent can combine multiple pieces of incomplete information if they are complementary wrt. the same dependence relation. We consider three cases for combining incomplete information (see Fig. 3). Suppose, agent A has information: $p_3(a_1,b_1), p_3(a_2,b_2), r_2(c_6,d_8)$, agent B has information: $r_1(a_3,c_0), r_1(a_2,c_6), r_3(d_8,e_2)$, and the two agents mutually know that A needs information of type p_1.

Case 1: According to the dependence tree, agent A can generate two pieces of incomplete information: $p_1(a_1,b_1,?z)$ and $p_1(a_2,b_2,?z)$. Agent B also can derive two pieces of incomplete information: $q(a_3,e_2)_{r_2(c_0,d_8)}$ and $q(a_2,e_2)_{r_2(c_6,d_8)}$. Believing q is useful in deriving p_1, B informs its incomplete information about q to A. Now, since A's information $r_2(c_6,d_8)$ confirms B's assumption associated with $q(a_2,e_2)$, A can derive $p_1(a_2,b_2,e_2)$.

Case 2: Since B knows nothing connecting r_1, r_2, and r_3 with p_1, B cannot help A now. But B can still generate incomplete information of type p_1 whenever complete information of type p_3 or q is available to B.

Case 3: In this case, B's role is largely limited; as far as A's needs–p_1–is concerned, B acts more like a broker (e.g., propagate A's needs to a third party) or a passive server (reply upon being requested by A).

4. InWit–A Tool for Manipulating Information Dependence

We have implemented a tool–InWit. Fig.4 is a screen shot of InWit, which can automatically construct information dependence trees based on the inference knowledge predefined for an agent, and allow users to indirectly adjust agents' information gathering, processing, and sharing behaviors. Through InWit, a human user can manipulate multi-agent information dependence in the following ways:

(1) adjust tree structure: the user can add (remove) an inference rule by adding (removing) the corresponding AND node to (from) the dependence tree, and adjust inference rules by adding (removing) OR nodes;

(2) adjust node properties: InWit allows a user to adjust the preference values of AND nodes, and to adjust the observability of an agent wrt. certain information;

(3) explore dependence: a list of attentions can be generated when an agent initiates a query to its knowledge base. InWit allows a user to explore the fact-level information dependence of an attention. For example, in Fig 4, the subtree related to the selected attention $(threatHigh\ E1)$ is highlighted, and the relevant facts are displayed in the Output window;

(4) remove attentions: human users may change their attentions over time. For example, in time-stress domains, a decision maker may choose to pay attention to a particular kind of events. InWit allows a user to remove superfluous attentions.

2 Suppose we have rule $r_1(?x,?y) \wedge r_2(?y,?w) \wedge r_3(?w,?z) \rightarrow q(?x,?z)$.

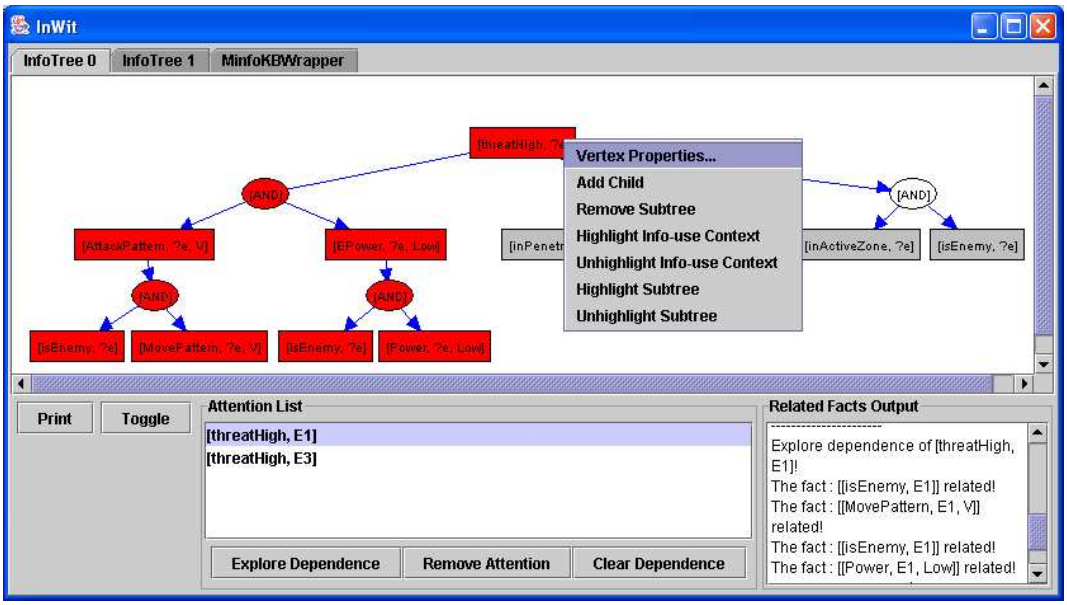

Figure 4: InWit–A Tool for Manipulating Information Dependence

5. Summary

Needs-driven information sharing is critical for alleviating the information overload problem in the Information Age. Information dependence reasoning enables agents not only to respond others' needs cautiously (sensitive to information use context, work progress, etc.), but also to commit to other's needs subtly (provide indirect or partial information). As well as characterized the nature of multi-agent information dependence and explored ways of using dependence knowledge, we implemented InWit–a tool that a human user can use to dynamically manipulate multi-agent information dependence. Our ongoing effort is to enhance InWit with other features and to integrate InWit with CAST–a multi-agent architecture that allows agents to anticipate teammates' information needs and proactively help them. The aim is to enhance CAST's capability of supporting proactive information seeking, fusing, and sharing in mixed human/agent teamwork.

References

[1] J. Bradshaw, M. Sierhuis, A. Acquisti, Y. Gawdiak, D. Prescott, R. Jeffers, N. Suri, and R. van Hoof. What we can learn about human-agent teamwork from practice. In *Workshop on Teamwork and Coalition Formation at AAMAS 02*, Bologna, Italy, 2002.

[2] P. R. Cohen and H. J. Levesque. Teamwork. *Nous*, 25(4):487–512, 1991.

[3] K. Decker. TAEMS: A Framework for Environment Centered Analysis & Design of Coordination Mechanisms. In *Foundations of Distributed Artificial Intelligence, Chapter 16*, pages 429–448. G. O'Hare and N. Jennings (eds.), Wiley Inter-Science, January 1996.

[4] K. S. Decker and V. R. Lesser. Quantitative modeling of complex computational task environments. In *Proceedings of the 11th National Conference on AI*, 1993.

[5] X. Fan, J. Yen, R. Wang, S. Sun, and R. A. Volz. Context-Centric Proactive Information Delivery. In *Proceedings of the 2004 IEEE/WIC Intelligent Agent Technology conference*, pages 31–37. IEEE Press, October 2004.

[6] B. Grosz and S. Kraus. The evolution of sharedplans. *Found. and Theories of Rational Agencies*, pages 227–262, 1999.

[7] J. Searle. Indirect speech acts. In P. Cole and J. Morgan, editors, *Syntax and semantics. III. Speech acts*, pages 59–82. NY: Academic Press, 1975.

[8] J. S. Sichman and R. Conte. Multi-agent dependence by dependence graphs. In *AAMAS 2002*, pages 483–490, 2002.

[9] J. S. Sichman, R. Conte, C. Castelfranchi, and Y. Demazeau. Social Reasoning Mechanism Based on Dependence Networks. In A. G. Cohn, editor, *Proceedings of the 11th European Conference on Artificial Intelligence*, 1994.

[10] M. Tambe. Towards flexible teamwork. *Journal of Artificial Intelligence Research*, 7:83–124, 1997.

[11] M. Williamson, K. S. Decker, and K. Sycara. Unified information and control flow in hierarchical task networks. In *Proceedings of the AAAI-96 workshop on theories of planning, action, and control*, 1996.

[12] J. Yen, X. Fan, S. Sun, T. Hanratty, and J. Dumer. Agents with shared mental models for enhancing team decision-makings. *Decision Support Systems*, page (in press), 2004.

[13] J. Yen, J. Yin, T. Ioerger, M. Miller, D. Xu, and R. Volz. Cast: Collaborative agents for simulating teamworks. In *Proceedings of IJCAI'2001*, pages 1135–1142, 2001.