On the Design of Social Diagnosis Algorithms for Multi-Agent Teams

Meir Kalech and **Gal A. Kaminka**
The MAVERICK Group
Computer Science Department
Bar Ilan University
Ramat Gan, Israel
{kalechm, galk}@cs.biu.ac.il

Abstract

Teamwork demands agreement among team-members to collaborate and coordinate effectively. When a disagreement between teammates occurs (due to failures), team-members should ideally diagnose its causes, to resolve the disagreement. Such diagnosis of social failures can be expensive in communication and computation overhead, which previous work did not address. We present a novel design space of diagnosis algorithms, distinguishing several phases in the diagnosis process, and providing alternative algorithms for each phase. We then combine these algorithms in different ways to empirically explore specific design choices in a complex domain, on thousands of failure cases. The results show that centralizing the diagnosis disambiguation process is a key factor in reducing communications, while run-time is affected mainly by the amount of reasoning about other agents. These results contrast sharply with previous work in disagreement detection, in which distributed algorithms reduce communications.

1 Introduction

One of the key requirements in teamwork is that agents come to agree on specific aspects of their joint task [Cohen and Levesque, 1991; Grosz and Kraus, 1996; Tambe, 1997]. With increasing deployment of robotic and agent teams in complex, dynamic settings, there is an increasing need to also be able respond to failures that occur in teamwork [Tambe, 1997; Parker, 1998], in particular to be able to diagnose the causes for disagreements that may occur, in order to facilitate recovery and reestablish collaboration (e.g., by negotiations [Kraus *et al.*, 1998]). This type of diagnosis is called social diagnosis, since it focuses on finding causes for failures to maintain designer-specified social relationships [Kaminka and Tambe, 2000].

For instance, suppose a team of four robotic porters carry a table, when suddenly one of the robots puts the table down on the floor, while its teammates are still holding the table up. Team-members can easily identify a disagreement, but they also need to determine its causes, e.g., that the robot believed the table reached the goal location, while its teammates did not. Given this diagnosis, the robots can negotiate to resolve the disagreement.

Unfortunately, while the problem of detection has been addressed in the literature (e.g., [Kaminka and Tambe, 2000]), social diagnosis remains an open question. Naive implementations of social diagnosis processes can require significant computation and communications overhead, which prohibits them from being effective as the number of agents is scaled up, or the number of failures to diagnose increases. Previous work did not rigorously address this concern: [Kaminka and Tambe, 2000] guarantee disagreement detection without communications, but their heuristic-based diagnosis often fails. [Dellarocas and Klein, 2000; Horling *et al.*, 1999] do not explicitly address communication overhead concerns. [Frohlich *et al.*, 1997; Roos *et al.*, 2001] assume fixed communication links, an assumption which does not hold in dynamic teams in which agents may choose their communication partners dynamically (see Section 5 for details).

We seek to examine the communication and computation overhead of social diagnosis in depth. We distinguish two phases of social diagnosis: (i) selection of the diagnosing agents; and (ii) diagnosis of the team state (by the selected agents). We provide alternative algorithms for these phases, and combine them in different ways, to present four diagnosis methods, corresponding to different design decisions. We then empirically evaluate the communications and run-time requirements of these methods in diagnosing thousands of systematically-generated failure cases, occurring in a team of behavior-based agents in a complex domain.

We draw general lessons about the design of social diagnosis algorithms from the empirical results. Specifically, the results show that centralizing the disambiguation process is a key factor in dramatically improving communications efficiency, but is not a determining factor in run-time efficiency. On the other hand, explicit reasoning about other agents is a key factor in determining run-time: Agents that reason explicitly about others incur significant computational costs, though they are sometimes able to reduce the amount of communications. These results contrast with previous work in disagreement detection, in which distributed algorithms reduce communications.

The paper is organized as follows: Section 2 motivates the research. Section 3 presents the diagnosis phases and alternative building blocks for diagnosis. Section 4 specifies diag-

nosis methods which combine the building blocks in different ways, and evaluates them empirically. Section 5 presents related work, and Section 6 concludes.

2 Motivation and Examples

Agreement (e.g., on a joint plan or goal) is key to establishment and maintenance of teamwork [Cohen and Levesque, 1991; Jennings, 1995; Grosz and Kraus, 1996; Tambe, 1997]. Unfortunately, complex, dynamic settings, sometimes lead to disagreements among team-members (e.g., due to sensing failures, or different interpretations of sensor readings) [Dellarocas and Klein, 2000]. Given the critical role agreement plays in coordinated, collaborative operation of teams, we focus on the diagnosis of disagreements in this paper.

The function of a diagnosis process is to go from fault detection (where an alarm is raised when a fault occurs), to fault identification, where the causes for the fault are discovered. In diagnosing disagreements, the idea is to go beyond simple detection that a disagreement exists, to identification of the differences in beliefs between the agents, that lead to the disagreement. Such differences in beliefs may be a result of differences in sensor readings or interpretation, in sensor malfunctions, or communication difficulties. Once differences in beliefs are known, then they can be negotiated and argued about, to resolve the disagreements [Kraus *et al.*, 1998]. Unfortunately, diagnosis of such failures can be extremely expensive both in terms of computation as well as in communications (see Section 5 for quantitative evaluation), and thus a key challenge is to minimize communications and computation.

To illustrate the problem we use an example, originally reported in [Kaminka and Tambe, 2000], from the ModSAF domain (a virtual battlefield environment with synthetic helicopter pilots). In the example a team of pilot agents is divided into two: scouts and attackers. In the beginning all teammates fly in formation looking for a specific way-point (a given position), where the scouts move forward towards the enemy, while the attackers land and wait for a signal. Kaminka and Tambe [2000] report on the following failure case (which there failure detection technique captures): There are two attackers and one scout flying in formation, when the attackers detect the way-point and land, while the scout fails to detect way-point and continues to fly.

A diagnosis system, running on at least one of the agents, must now isolate possible explanations for the disagreement, of which the real cause (agents different in their belief that the way-point was detected) is but one. Each such explanation is a diagnosis hypothesis since it reports a possible reason for the disagreement between the agents. Disambiguation between these hypotheses seems very straightforward: Each agent can generate its own hypotheses, and then exchange these hypotheses with its teammates to verify their correctness. Unfortunately, since each agent is potentially unsure of what its team-members are doing, the number of hypotheses can be quite large, which means communications will suffer greatly. Furthermore, having each agent simply report its internal beliefs to the others, while alleviating communications overhead somewhat, is also insufficient. Indeed, any approach requiring high bandwidth is not likely to scale in the number of agents [Jennings, 1995].

3 Building blocks for diagnosis

We distinguish two phases of social diagnosis: (i) selecting who will carry out the diagnosis; (ii) having the selected agents generate and disambiguate diagnosis hypotheses. To explore these phases concretely, we focus on teams of behavior-based agents. The control process of such agents is relatively simple to model, and we can therefore focus on the core communications and computational requirements of the diagnosis.

We model an agent as having a decomposition hierarchy of behaviors, where each behavior (node in the hierarchy) has preconditions (which allow its selection for execution when satisfied), and termination conditions (which terminate its execution if the behavior was selected, and the conditions are satisfied). Each behavior may have actions associated with it, which it executes once selected. It may have child behaviors, which it can select based on their matching preconditions and preference rules. At any given time, the agent is controlled by a top-to-bottom path through the hierarchy, root-to-leaf. Only one behavior can be active in each level of the hierarchy. Such a generic representation allows us to model different behavior-based controllers (e.g. [Firby, 1987; Newell, 1990]).

We assume that a team of such agents coordinates through designer-provided definition of team behaviors, which are to be selected and de-selected jointly through the use of communications or other means of synchronization. Team behaviors, typically at higher-levels of the hierarchy, serve to synchronize high-level tasks, while at lower-levels of the hierarchy agents select individual (and often different) behaviors which control their execution of their own individual role.

For instance, in the ModSAF example, the scout should fly to look for the enemy, while the attackers should wait at the way-point. All agents are executing in this case a team behavior called *wait-at-point*, in service of which the attackers are executing individual *just-wait* (land) behaviors, while the scout is executing a more complex *fly-ahead* behavior.

Disagreement between team-members is manifested by selection of different team behaviors, by different agents, at the same time. Since team behaviors are to be jointly selected (by designer specification), such a disagreement can be traced to a difference in the satisfaction of the relevant pre-conditions and termination conditions, e.g., agent A believes P, while agent B believes ¬P, causing them to select different behaviors. It is these contradictions which the diagnosis process seeks to discover. A set of contradictions in beliefs that accounts for the selection of different team behaviors by different agents is called a diagnosis.

3.1 Disambiguating Diagnosis Hypotheses

The first phase in the diagnosis process involves selection of the agent(s) that will carry out the diagnosis process. However, the algorithms used for selection may depend on the diagnosis process selected in the second phase (disambiguation), and so for clarity of presentation, we will begin by discussing the second phase. Assume for now that one or more

agents have been selected to carry out the diagnosis process. The agents must now identify the beliefs of their peers.

Perhaps the simplest algorithm for this is to have all team-members send their relevant beliefs to the diagnosing agent (the diagnosing agent can inform the team-members of the detection of a disagreement to trigger this communication). In order to prevent flooding the diagnosing agent with irrelevant information, each team-member sends only beliefs that are relevant to the diagnosis, i.e., only the beliefs that are associated with its currently selected behavior.

Upon receiving the relevant beliefs from all agents, the generation of the diagnosis proceeds simply by comparing all beliefs of team-members to find contradictions (e.g., agent A believes P, while agent B believes ¬P). Since the beliefs of the other agents are known with certainty (based on the communications), the resulting diagnosis must be the correct one. However, having all agents send their beliefs, may severely impact network load.

We thus propose a novel selective monitoring algorithm, in which the diagnosing agent controls the communications, by initiating targeted queries which are intended to minimize the amount of communications. To do this, the diagnosing agent builds hypotheses as to the possible beliefs held by each agent, and then queries the agents as necessary to disambiguate these hypotheses.

This process begins with *RESL*, a previously-published behavior recognition and belief inference algorithm [Kaminka and Tambe, 2000], which is only presented here briefly as a reminder. Under the assumption that each agent has knowledge of all the possible behaviors available to each team-member, i.e., their behavior library (an assumption commonly made in plan recognition), each observing agent creates a copy of the fully-expanded behavior hierarchy for each of its teammates. It then matches observed actions with the actions associated with each behavior. If a behavior matches, it is tagged. All tagged behaviors propagate their tags up the tree to their parents (and down to their children) such as to tag entire matching paths: These signify behavior recognition (plan recognition) hypotheses that are consistent with the observed actions of the team-member.

Once hypotheses for the selected behavior of an agent are known to the observer, it may infer the possible beliefs of the observed agent by examining the pre-conditions and the termination conditions of the selected behavior. To do this, the observer must keep track of the last known behavior(s) hypothesized to have been selected by the observed agent. As long as the hypotheses remain the same, the only general conclusion the observer can make is that the termination conditions for the selected behaviors have not been met. Thus it can infer that the observed agent currently believes the negation of the termination conditions of selected behaviors.

When the observer recognizes a transition from one behavior to another, it may conclude (for the instant in which the transition occurred) that the termination conditions of the previous behavior, and the pre-conditions of the new behavior are satisfied. In addition, again the termination conditions of the new behavior must not be satisfied; otherwise this new behavior would not have been selected. Therefore, the beliefs of the observed agent (at the moment of the transition) are: *(termination conditions of last behavior)* $\wedge$ *(pre-conditions of current behavior)* $\wedge\neg$ *(termination conditions of current behavior)*.

For instance, suppose an attacker is observed to have landed. The observer may conclude that either the *halt* or *wait-at-point* behavior has been selected. Furthermore, the observer can conclude that the termination conditions for *halt* do not hold, or that the termination conditions for *wait-at-point* do not hold. If the observed action indicates that the agent has just transitioned into the behavior associated with landing, then the preconditions of *halt* and *wait-at-point* would also be inferred as true.

Once the hypotheses are known, the agent can send targeted queries to specific agents in order to disambiguate the hypotheses. The queries are selected in a manner that minimizes the expected number of queries. Intuitively, the agent prefers to ask first about propositions whose value, when known with certainty, will approximately split the hypotheses space.

3.2 Selecting a Diagnosing Agent

Let us now turn to the preceding phase, in which the agents that will carry out the diagnosis are selected. Several techniques are available. First, a design-time selection of one of the agents is the most trivial approach. It requires a failure state to be declared in the team, such that the selected agents know to begin their task. Of course, one problem with this approach is that it requires all agents to be notified of the failure. A second technique that circumvents this need is to leave the diagnosis in the hands of those agents that have detected the failure, and allow them to proceed with the diagnosis without necessarily alerting the others unless absolutely necessary.

We present a novel third approach, in which selection of the diagnosing agent is based on its team-members' estimate of the number of queries that it will send out in order to arrive at a diagnosis, i.e., the number of queries that it will send out in the disambiguation phase of the diagnosis (previous section). The key to this approach is for each agent to essentially simulate its own reasoning in the second phase, as well as that of its teammates. Agents can then jointly select the agent with the best simulated results (i.e., the minimal number of queries).

Surprisingly, all agents can make the same selection without communicating, using a recursive modelling technique in which each agent models itself through its model of its teammates. This proceeds as follows. First, each agent uses the belief recognition algorithm to generate the hypotheses space for each team-member other than itself. To determine its own hypotheses space (as it appears to its peers), each agent uses recursive modelling, putting itself in the position of one of its teammates and running the belief recognition process described above with respect to itself. Under the assumptions that all agents utilize the same algorithm, and have access to the same observations, an agent's recursive model will yield the same results as the modelling process of its peers. At this point, each team-member can determine the agent with the minimal number of expected queries (the queries that split the space of the queries). In order to guarantee an agreement on the selected agent, each team-member has an ID number,

which is determined and known in advance. In case there are two agents with the same minimal number of expected queries, the agent with the minimal ID is selected. This entire process is carried out strictly based on team-members' observations of one another, with no communications other than an announcement of a disagreement.

For instance, suppose there are three agents A, B and C. To determine the diagnosing agent, A puts itself in B's position and considers the hypotheses B has about A and C, given A's observable actions. A also uses the belief recognition process described earlier to determine the number of hypotheses available about B's beliefs, C's beliefs, etc. It now simulates selecting queries by each agent, and selects the agent (say, C) with the minimal number of expected queries. B, and C also run the same process, and under the assumption that each agent's actions are equally observable to all, will arrive at the same conclusion.

Summary. We presented a space of social diagnosis algorithms: Each such algorithm operates in two phases, and we presented alternative techniques for each phase. For the selection of the diagnosing agent, we have the following methods: (i) rely on pre-selection by the designer; (ii) let the agents that detected the fault do the diagnosis; or (iii) choose the agent most likely to reduce communications (using the distributed recursive modelling technique described). In terms of computing the diagnosis, two choices are available: Either have all agents communicate their beliefs to the selected agents, or allow the diagnosing agents to actively query agents as to the state of their beliefs, while minimizing the number of queries as described above.

4 Evaluation and Discussion

The design alternatives define a space of diagnosis algorithms. This section evaluates four intuitive design decisions in this space, and draws lessons about the effects of specific design choices on overall computation and communication overhead.

Method 1. The first design choice corresponds to arguably the most intuitive diagnosis algorithm, in which all agents are pre-selected to carry out the diagnosis. When a failure is detected (and is made known to all agents) each agent communicates all its relevant beliefs to the others so that each and every team-member has a copy of all beliefs, and therefore can do the disambiguation itself.

Method 2. Method 1 uses redundant communications to achieve this distribution, while arguably only a single agent really needs to have the final diagnosis in order to begin a recovery process. Thus in method 2, the agents that detected the disagreement automatically take it upon themselves to carry out the diagnosis, unbeknownst to their teammates, and to each other. Because their team-mates do not know of the disagreement (or who has been selected to diagnose it), they cannot rely on their team-mates to communicate their beliefs without being queried (in phase 2). Instead, they use the querying algorithm discussed in the previous section. However, because the diagnosing agents also do not know of each other, in cases where more than one agent detects the disagreement, all such agents will query the other team-members.

Method 3. The next design choice we examine corresponds to a diagnosis algorithm, in which the designer pre-selects a neutral agent. When a failure is discovered (and is made known to all agents), all team-members communicate all their relevant beliefs to the pre-selected agent.

Method 4. A final algorithm attempts to alleviate the communications overhead. It uses the recursive modelling technique to have all team-members agree on which agent is to carry out the diagnosis (this requires the detection of the disagreement to be made known). Once the agent is selected (with no communications), it queries its teammates as needed.

Table 1 summarizes the different methods. Each method is presented in a different row. The columns correspond to the different phases of the diagnosis process. The choice of algorithm is presented in each entry, along with a marking that signifies the number of agents that execute the selected technique for the phase in question.

Method	Selection	Disambiguation
1	pre-selected (N)	N (report)
2	detectors ($K \leq N$)	K (query)
3	pre-selected (1)	1 (report)
4	estimated optimal (1)	1 (query)

Table 1: Summary of evaluated diagnosis methods

For instance row 2 should be read as follows: In method 2, the agents selected to perform the disambiguation are those who detected the disagreement. K such agents exist (where K is smaller or equal than the total number of agents in the team, N), and they each execute a minimal-queries algorithm. In contrast, row 3 indicates that a single pre-selected agent executes the diagnosis, and it relies on reports from all agents to carry out the diagnosis.

Focusing on diagnosing teams of behavior-based agents in the ModSAF domain, we performed experiments in which the different diagnosis methods were systematically tested on *thousands* of failure cases, varying the number of agents, the behaviors selected by each agent, and the roles of the agents. The experiments were executed with teams of two to ten agents. For each *n* agents there are three sets of tests: (1) one attacker and *n-1* scouts; (2) *n-1* attackers and one scout; (3) *n/2* attackers and *n/2* scouts. For each set of x attackers and y scouts, we systematically checked all possible disagreement cases for all team behaviors (a total 2372 tests for each method). In each test we recorded the number of messages sent, and runtime of the diagnosis process.

In Table 2 we present the results of a single test, where one scout and two attackers *fly in formation*, when the scout transitions to the *wait-at-point* behavior while the attackers continue to fly. The diagnosis is that the scout detected the *way-point*, while the attackers did not.

The first column in Table 2 reports the method used. The second column presents the number of messages sent reporting on beliefs, or querying about their truth (one message per belief). The third column reports the number of messages sent informing agents of the existence of failures

(we assume point-to-point communications). The next three columns summarize the runtime of each agent.

Method	Messages		Run-time(msec)		
	Belief	Failure	A1	A2	Scout
1	16	2	12	12	12
2	8	0	0	0	192
3	8	2	3	3	9
4	3	2	87	80	99

Table 2: results of diagnosing a specific failure case

For instance, the number of messages reporting on beliefs for method 3 is 8, and 2 failure messages were sent (i.e., one of the agents detected the failure and informed the others). The runtime of all the teammates for method 3 is 3 milliseconds, except for the scout, which disambiguated the beliefs in this case, and therefore took an additional 6 milliseconds (for a total of 9 milliseconds).

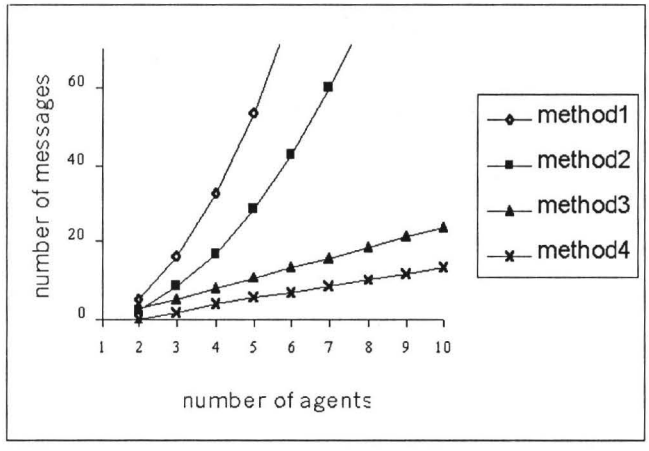

Figure 1: Average number of messages per failure case

Figures 1 and 2 summarize the results of the experiments. In both figures, the x axis shows the number of agents in the diagnosed team. Figure 1 presents the average number of belief messages for the different failure and role variations, for each team size (failure messages were ignored in the figure, since their effect is negligible). Figure 2 presents the average run-time (in milliseconds) of those same tests. The run-time of each test was taken as the maximum of any of the agents in the test.

Both figures show interesting grouping of the evaluated techniques. In Figure 1 (number of messages), methods 3 and 4 show a slow, approximately-linear growth, while methods 1 and 2 show a much faster non-linear growth. In Figure 2 (runtime), the grouping is different: Methods 1 and 3 grow significantly slower than methods 2 and 4.

The first conclusion we draw from these figures is that runtime is affected by the choice of a disambiguation method (Figure 2, ref. Table 1). Methods (here, methods 1 and 3) that rely on the agents to report their relevant beliefs do not reason about the hypothesized beliefs of others. Therefore,

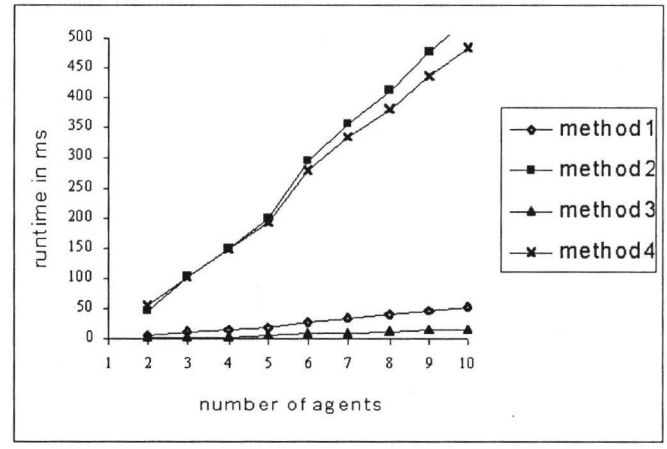

Figure 2: Average run-time per failure case

their run-time is much smaller than methods (here, methods 2 and 4) which hypothesize about the beliefs of others. However, as Figure 1 shows, the goal of reducing communications is actually achieved, as method 4 does indeed result in less communications then method 3. The question of whether the cost in run time is worth the reduction in communications is dependent on the domain.

Indeed, we draw a second conclusion from Figure 1. Despite the additional savings provided by the minimal query algorithm, the choice of a centralized diagnosing agent is the main factor in qualitatively reducing the number of messages sent, as well as in shaping the growth curve as the number of agents is scaled up. These results contrast sharply with previous work in disagreement detection, in which distributed algorithms reduce communications [Kaminka and Tambe, 2000].

5 Related Work

While diagnosis of a single-agent system is relatively well understood, and known to be computationally difficult, social diagnosis (diagnosis of social failures such as disagreements and coordination failures) remains an open area of research. In particular, to our best knowledge, an in-depth exploration of design choices in terms of communication and computation has not been done before. The most closely-related work to ours is reported in [Kaminka and Tambe, 2000]. This previous investigation provides guarantees on detection of disagreements, but only presented a heuristic approach to diagnosis, which indeed does not always succeed. The algorithms we present here succeed in the same examples where the previous heuristic approach fails. Parker [1998] reports on a behavior-based architecture which is very robust and is able to recover from failures by having robots take over tasks from failing teammates. This is done using continuous communications, but with an out explicit diagnosis process such as those described in this paper.

Dellarocas and Klein [2000] described failures handling services that use a knowledge base of generic exceptions and a decision tree of diagnoses; the diagnosis process is done by

a centralized agent traversing down the tree by asking questions about the relevant problem. Similarly, Horling [1999] uses a casual model of failures and diagnoses to detect and respond to multi-agent failures. Both investigations do not address communications overhead.

Frohlich [1997] suggested dividing a spatially distributed system into regions, each under the responsibility of a diagnosis agent. If the fault depends on two regions the agents that are responsible to those regions cooperate in making the diagnosis. This method is inappropriate for dynamic team settings, where agents cannot pre-select their communication partners. Roos [2001] expanded Frohlich's work to semantically distributed systems, where each agent looks at different aspects of the whole system. In this system each agent makes diagnosis separately and the correct diagnosis is found by exchanging the diagnoses between the agents. However, the communication links are fixed, such that each failure is diagnosed strictly by the agents that are associated with its communication link. In contrast, in teams disagreements can arise between any two agents, and thus fixed communication commitments cannot be made in design time.

Brodie [2002] tries to minimize communication costs in computer networks. He uses intelligence probes which are sent through the network in order to query remote nodes. By combining the results of different probes, failing nodes can be identified and isolated. Thus Brodie's work essentially determines the liveliness status of agents, while our work focuses on fine-grain diagnosis of causes for disagreements, in terms of contradictory beliefs held by different agents.

6 Summary and Future Work

In this paper we present a novel design space for algorithms of social diagnosis, and evaluate specific design decisions in terms of their communications and computation overheads. We presented four methods of diagnosing a team of behavior-based agents, using familiar and novel algorithms, among these an algorithm that minimizes the number of diagnosis queries by using a recursive modelling technique to select the agent which will use the smallest number of queries. We then empirically and systematically evaluated the different combinations to draw general conclusions about the design of diagnosis algorithms. A first conclusion is that centralizing the diagnosis disambiguation task is critical in reducing communications. Furthermore, techniques where agents reason explicitly about the beliefs of their peers are computationally inferior (in run-time) to techniques where agents do not reason about others. However, such computation does result in a slight reduction in communications.

All methods find only the contradictions between agent beliefs, where the beliefs are derived directly from the hypothesized behaviors. But in complex behavior-based control systems, chains of inference may lead from one belief to the next. Our system is currently not able to back chain through such inference pathways, and thus is unable to draw conclusions beyond the beliefs that immediately tied to preconditions and termination conditions. We plan to tackle this challenge in the future.

Acknowledgments

We thank Meirav Hadad for help in the submission process. As always, thanks to K. Ushi and K. Raviti.

References

[Brodie *et al.*, 2002] M. Brodie, I. Rich, and S. Ma. Intelligence probing: A cost-effective approach to fault diagnosis computer networks. *IBM Systems Journal*, 41-3, 2002.

[Cohen and Levesque, 1991] Philip R. Cohen and Hector J. Levesque. Teamwork. *Nous*, 35, 1991.

[Dellarocas and Klein, 2000] Chrysanthos Dellarocas and Mark Klein. An experimental evaluation of domain-independent fault-handling services in open multi-agent systems. pages 95–102, Boston, MA, 2000. IEEE Computer Society.

[Firby, 1987] R. James Firby. An investigation into reactive planning in complex domains. 1987.

[Frohlich *et al.*, 1997] P. Frohlich, I. Mora, W. Nejdl, and M. Schroeder. Diagnostic agents for distributed systems. *proceedings of ModelAge*, 97, January 1997.

[Grosz and Kraus, 1996] Barbara J. Grosz and Sarit Kraus. Collaborative plans for complex group actions. *JAIR*, 86:269–358, 1996.

[Horling *et al.*, 1999] Bryan Horling, Victor R. Lesser, Regis Vincent, Ana Bazzan, and Ping Xuan. Diagnosis as an integral part of multi-agent adaptability. Technical Report CMPSCI Technical Report 1999-03, University of Massachusetts/Amherst, January 1999.

[Jennings, 1995] Nicholas R. Jennings. Controlling cooperative problem solving in industrial multi-agent systems using joint intentions. 75(2):195–240, 1995.

[Kaminka and Tambe, 2000] Gal A. Kaminka and Milind Tambe. Robust multi-agent teams via socially-attentive monitoring. 12:105–147, 2000.

[Kraus *et al.*, 1998] Sarit Kraus, Sycara Katia, and Amir Evenchik. Reaching agreements through argumentation: a logical model and implementation. *Artificial Intelligence*, 104(1–2):1–69, 1998.

[Newell, 1990] Allen Newell. *Unified Theories of Cognition*. Harvard University Press, Cambridge, Massachusetts, 1990.

[Parker, 1998] Lynne E. Parker. ALLIANCE: An architecture for fault tolerant multirobot cooperation. *IEEE Transactions on Robotics and Automation*, 14(2):220–240, April 1998.

[Roos *et al.*, 2001] N. Roos, A. Teije, A. Bos, and C. Witeveen. Multi agent diagnosis with spatially distributed knowledge. *BNAIC*, 2001.

[Tambe, 1997] Milind Tambe. Towards flexible teamwork. *JAIR*, 7:83–124, 1997.