Handling Coordination Failures in Large-Scale Multi-Agent Systems

A Brief Overview of Challenges and Methods

Gal A. Kaminka

The MAVERICK Group
Computer Science Department
Bar Ilan University
Ramat Gan, Israel

Summary. Agents monitor other agents to coordinate and collaborate robustly. The goals of such monitoring include detection of coordination failures. However, as the number of monitored agents is scaled up, two key challenges arise: (i) Agents become physically and logically unconnected (unobservable) to their peers; and (ii) the number of possible coordination failures grows exponentially, with all potential interactions. This paper examines these challenges in teams of cooperating agents. We provide a brief survey of the evolution of two key approaches to handling coordination failures in large-scale teams: Restricting the number of agents that must be monitored, and using model-based rather than fault-based detection methods. We focus on a monitoring task that is of particular importance to robust teamwork: detecting disagreements among team-members.

1 Introduction

Agents in realistic, complex, domains must monitor other agents to accomplish their tasks, detect failures, coordinate, and collaborate. Indeed, the importance of agent monitoring in deployed multi-agent systems has long been recognized in theory (e.g., [2, 7, 9]), and in practice. Monitoring has been discussed in the context of industrial systems (e.g., [16]), to virtual environments for training and research (e.g., [36, 37, 30, 31]), to human-computer interaction (e.g., [27]), and multi-robot teams (e.g., [28, 6, 21]). Agent monitoring infrastructure is of particular importance in teams of cooperating agents, since the correct execution of teamwork mandates that team-members come to agree on the task that is jointly executed by the team, and manage interdependencies among team-members [2, 9].

One specific goal of monitoring in teams is detection and resolution of teamwork and coordination failures [24, 29, 38]. These may occur because of unanticipated environment states—likely in complex, dynamic

environments—or from communication, sensor, or actuator uncertainties. For instance, intermittent failures in communications may cause a failure where one agent has sent a message, while its peers have not received it.

Thus deployed multi-agent systems must include facilities for detecting, diagnosing, and resolving failures. Indeed, a number of investigations have begun to explore mechanisms for detecting failures in coordination and teamwork [23, 25, 24, 3, 29, 38] and for diagnosing such failures [22, 32, 33, 12, 17].

However, large-scale multi-agent systems—where the number of agents is the principal scale factor—pose a number of challenges to the ability of agents to monitor each other, and thus to handle failures. Two of these challenges are: (i) *Limited connectivity*, where agents become physically and logically separated, and thus less able to monitor each other; and (ii) a *combinatorial complexity* of possible failures, as the number of possible failures grows with the number of all possible interactions between failures.

This paper discusses these challenges in depth, and explores their significance in large-scale multi-agent systems. We also discuss the implications of these challenges with respect to existing approaches to failure detection. We find in the literature two approaches to failure detection. Some investigations take an approach based on fault-models, where possible faults are enumerated at design time and recognized at run-time. Other investigations take a model-based approach where agents detect failures at run-time as deviations from a model of the normative coordination in the system.

To illustrate, we focus on the example of detecting disagreements—a principal failure in multi-agent teamwork—to show the evolution of existing methods in recent years to address large-scale systems. We show how an analysis of the monitoring requirements of disagreement detection can lead to improved, reduced, bounds on the connectivity of team-members. We also discuss relevant model-based detection work, which can represent the state of multiple agents together, and can therefore be utilized for highly-scalable disagreement detection.

This chapter is organized as follows. Section 2 provides motivation for this work by showing concrete examples of limited connectivity and combinatorial failure complexity in monitoring for disagreements. Section 3 focuses on limited connectivity, and discusses a general approach in which only specific key agents must be monitored, while detection is guaranteed. Section 4 focuses on the exponential complexity of the number of possible coordination failures. Finally, Section 5 concludes.

2 Motivation and Background

Teamwork literature, addressing human and synthetic teams, has often emphasized the importance of team-members being in agreement on various fea-

tures of their state, such as goals, plans, and beliefs[1]. Teamwork theory often defines agreement as a state of mutual belief, where agents reason to infinite recursion about their beliefs and their beliefs in others' beliefs in a proposition. For instance, *SharedPlans* theory requires team-members to mutually believe in a shared recipe [9] during the planning and execution phases of the task; the *Joint Intentions* framework emphasizes mutual belief in the team goals' selection, as well as in team-members' beliefs about the goals' achievability and relevance [2, 26]. Other investigations of agent teams have emphasized agreement on team plans to be jointly executed by team-members [16], on hierarchical team operators [35], on tasks to be executed collectively [28, 5, 21], etc. Investigations of human teamwork have not only emphasized agreement on the joint task, but also agreement on features of the environment that are important to the task being carried out by the team [1].

However, the literature also recognizes that achieving and maintaining agreement can be difficult. Teamwork theory recognizes that attainment of agreement by mutual belief is undecidable [10] and must therefore be approximated in practice. Such approximations frequently involve assumptions of trustworthiness of team-members, of foolproof communications [16], of team-members being able to observe each other [14], and/or of a mutually-visible environment [8]. As is often the case with approximations, they sometimes fail in practice (e.g., due to communications failures, sensing differences due to different physical locations of agents, etc.), and therefore team-members may find themselves in disagreement with each other. Such disagreements are often catastrophic, due to the unique importance of agreement in collaboration.

It is therefore critical that teams are monitored to detect such disagreements. A monitoring agent that identifies the state of team-members can compare the state of different team-members and detect differences on state features that, by design or by selection, should have been agreed upon [24]. However, as the number of monitored agents is scaled up, two challenges arise: (i) difficulty to observe or communicate with all agents, due to latency, range, occlusion and other separation factors (Section 2.1); and (ii) an exponential number of possible coordination failures (Section 2.2).

2.1 Limited Connectivity

As the number of agents grows, agents become logically and physically distributed, and cannot maintain continuous contact with each other. This may occur due to physical separation factors, such as occlusion and limited sensor range; or it may occur due to logical separation, such as limited communication reliability, interference, latency, or bandwidth. We use the term *limited connectivity* in a general sense to describe this phenomenon. Limited connectivity thus denotes both limited ability to observe a particular agent's actions as well as limited ability to communicate with the agent.

[1] Of course, the literature also addresses other critical features of teamwork aside from agreement. But agreement is a repeating theme in recent work.

The challenge of limited connectivity is of course only of limited concern in small-scale systems. Given a few cycles, the agents can typically integrate multiple attempts at communications and sensing of the world, over time, to form a fairly coherent mental picture of what their peers are up to. However, as the number of agents grows, the ability to integrate such information over time diminishes rapidly. For instance, existing peer-to-peer (P2P) include millions of simultaneously-active nodes. Yet not one node is able to communicate directly with all of its peers at once, due to both bandwidth and processing power issues. Even spreading the efforts over time will not be sufficient, as the duration of time required is too long for any practical interest.

2.2 Combinatorial Failure Complexity

A different concern with large scale system is the number of potential coordination failures it may get into. Suppose each of N agents may be in one of k internal states. Then the number of possible joint states is k^N. In loosely-coupled systems, each agent is essentially independent of its peers, and may select between its k possible states freely. In such systems, the vast majority of joint states—if not all—are considered valid states.

However, in a coordinated multi-agent system, the selection of an internal state by an agent is conditional by the selection of its peers' internal state. In other words, agents move between joint states together. Typically, only a limited portion of these states would be valid coordinated states, from the designer's perspective. Thus most joint states may in fact be invalid from a coordination point of view.

Disagreements are a good example of this. Suppose a team of N agents agrees that their selection of internal state would be synchronous, i.e., for every selected state of one agent, all others must be in some agreed-upon internal state. For simplicity in notation, we describe this case as mutual selection of states $1 \ldots k$, i.e., all all agents select the same state. There would be $O(k)$ valid agreement joint states, and the rest of the k^n joint states would be considered invalid—coordination failure—states.

Note that the number of possible coordination failure states grows exponentially in the number of agents. Thus large-scale systems where agents coordinate may have to face an exponential number of possible faults.

3 Monitoring Graphs for Limited Connectivity

As the number of monitored team-members increases, it becomes increasingly difficult to monitor all of them (Section 2). Thus a key question is how to guarantee failure-handling results while limiting the number of agents that must be monitored.

The approach we take to this involves the construction and analysis of monitoring graphs, which represent information about which agent can monitor whom. We show that for disagreement detection, one can set conditions on the structure of the graph which, when satisfied, guarantee that detection is *complete* and *sound*. Complete detection guarantees all failures will be detected (i.e., no false negatives). Sound detection guarantees only failures will be detected (i.e., no false positives). Using the conditions we explore in this section, one can guarantee sound and complete detection of disagreements while setting conditions on the connectivity of agents.

Definition 1. *A* monitoring graph *of a team T is a directed (possibly cyclic) graph in which nodes correspond to team-members of T, and edges correspond to monitoring conditions: If an agent A is able to monitor an agent B (either visually or by communicating with it), then an edge (A, B) exists in the graph. We say that monitoring graph is connected, if its underlying undirected graph is connected.*

If the monitoring graph of a team is not connected, then there is an agent which is not monitored by any agent, and is not monitoring any agent. Obviously, a disagreement can go undetected in such a team: If the isolated agent chooses an internal state different from what has been agreed upon with its peers, it would go undetected.

It is easy to see that if the graph is connected, and each agent knows exactly the selection of its monitored peer, then sound and complete detection is possible, in a distributed fashion. Each agent A monitors at least one other agent B (or is monitored by another agent B). If A selects an internal state different from B, then at least one of them would detect the disagreement immediately. For instance, if A monitors B—and knows with certainty B's state—then simple comparison with A's selected state is all that is needed.

In the general case, however, connectivity is insufficient. Suppose an agent A has selected state P_1, and is monitoring another agent B that has selected state P_2. A disagreement exists here since agent B should have selected P_1. However, since the internal state of B may not be known to A with certainty, A may have several interpretations of B's chosen state. The set of these interpretations may contain P_1, in which case A may come to incorrectly believe that B is *not* in a state of disagreement with A.

To treat this formally, let us use the following notation when discussing agent A's hypotheses as to the state of an agent B: Suppose B's state is P (for instance, P is a plan selected by B). We denote by $M(A, B/P)$ the set of agent-monitoring hypotheses that A constructs based on communications from B, or inference from B's observable behavior (i.e., via plan recognition). In other words, $M(A, B/P)$ is the set of all A's hypotheses as to B's state, when B's state (e.g., selected plan) is P. Note that when A monitors itself, it has direct access to its own state and so $M(A, A/P) = \{P\}$.

We make the following definitions which ground our assumptions about the underlying monitoring process that implements M:

Definition 2. *Given a monitoring agent A, and a monitored agent B, we say that A's monitoring of B is* complete *if for any state P that may be selected by B, $P \in M(A, B/P)$. If A is monitoring a team of agents $B_1, \ldots, B_n$, we say that A's* team-monitoring *of the team is* complete *if A's monitoring of each of $B_1, \ldots, B_n$ is complete.*

Monitoring completeness is commonly assumed (in its individual form) in plan-recognition work, (e.g., [34, 4, 15]), and generally holds in many applications. It means that the set $M(A, B/P)$ includes the correct hypothesis P, but will typically include other matching hypotheses besides P. Using this notation, we can now formally explore disagreement detection under uncertainty in monitoring.

Centralized Disagreement Detection

In general, as discussed above, the condition of monitoring graph connectivity is necessary, but insufficient, to guarantee complete and sound detection. Indeed, in [23], Kaminka and Tambe show that if a single centralized monitoring agent monitors all others, it can guarantee either sound or complete detection of disagreements, but not both (Figure 1-a).

However, Kaminka and Tambe found that if certain *key agents* exist, then it may be possible to reduce the monitoring requirements in the system. Key agents have the property that their behavior, when selecting one of two given states, is sufficiently unambiguous, such that any agent that monitors them and has selected either one of the two states can identify with certainty whether a disagreement exists between it and the key agents. We repeat here the formal definition of key agents from [24]:

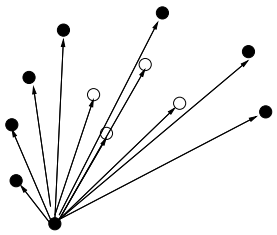
Definition 3. *Let P_1, P_2 be two agent states. Suppose an agent A is monitoring an agent B. If $M(A, B/P_1) \cap M(A, B/P_2) = \emptyset$ for any agent A, we say that B has observably-different roles in P_1 and P_2, and call B a* key agent *in $\{P_1, P_2\}$. We assume symmetry so that if two states are not observably different, then $M(A, B/P_1) \cap M(A, B/P_2) \supseteq \{P_1, P_2\}$.*

The key-agent is the basis for the conditions under which a team-member A_1 will detect a disagreement with a team-member A_2. This is done by preferring *maximally-coherent* hypotheses as to the state of the monitored agent. Maximally-coherent hypotheses are optimistic—they are hypotheses that minimize the number of disagreements between the two agents. The use of such hypotheses leads to sound disagreement detection [23, 24].

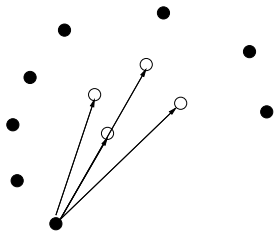
An agent A_1 (selecting state P_1) will detect a disagreement with a team-member A_2 (selecting a different state P_2) if A_2 is a key agent for the plans P_1, P_2 [24, Lemma 1]. A_1 knows that it has selected P_1. If A_2 has selected P_2, and is a key-agent in P_1 and P_2, then A_1 is guaranteed to notice that a disagreement exists between itself and A_2, since A_2 is acting observably

different than it would if it had selected P_1. A_1 can now alert its teammate, diagnose the failure, etc.

When key agents exist in a team, it is sufficient for a single agent to monitor them to guarantee sound detection in the centralized case [20]. More accurately, if the team is observably-partitioned, i.e., a key agent exists for any pair of internal states potentially selected by team-members, then it is sufficient for a single agent to monitor only the key agents, to guarantee sound detection of disagreements. However, all key agents must be monitored (Figure 1-b).



(a) Centralized monitoring, sound or complete, but not both, in [24].



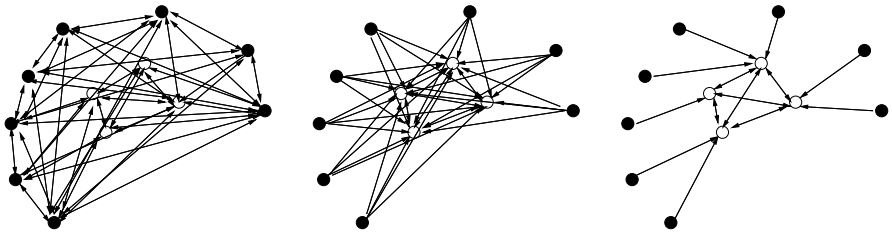
(b) Centralized monitoring, sound, in [20].

Fig. 1. Illustration of centralized monitoring graphs. Non-filled dots indicate key agents.

Distributed Disagreement Detection

We now consider the case of distributed monitoring settings, where team-members monitor each other. First, in [23] Kaminka and Tambe have shown that if at least a single key agent exists for every pair of plans (i.e., the team employs an *observably-partitioned set of team plans*), and if all team-members monitor *all* agents, then detection is not only sound, but also complete (see Figure 2-a for illustration). Later on [24, Theorem 4], the result was clarified: All agents must monitor the key agents only—all of them—and the key agents must monitor each other (Figure 2-b). Guaranteed sound and complete detection here means that at least one team-members will detect a disagreement if one occurs, and no false detections will take place.

This result is of particular interest to building practical robust teams, and fortunately the conditions for it are often easy to satisfy: Teams are very often composed such that not all agents have the same role in the same plan, and



(a) Distributed monitoring, in [23].

(b) Distributed monitoring, in [24].

(c) Distributed monitoring, in [20].

Fig. 2. Illustration of distributed monitoring graphs. Non-filled dots indicate key agents. All cases allow for sound and complete disagreement detection.

in general, roles do have observable differences between them. Often, the set $M(A, B/P)$ can be computed offline, in advance; this allows the designer to identify the key agents in a team prior to deployment. Furthermore, any agent can become a key-agent simply by communicating its state to the monitoring agent and therefore eliminating ambiguity; thus a team can use highly-focused communications to guarantee detection.

However, the requirement that *all* key-agents be monitored prohibits deployment of scaled-up applications: First, as the size of the team grows, limited connectivity becomes more common, since agents become more physically and logically distributed. Thus not all agents, and in particular key agents, are going to be visible. Second, the monitoring task itself would need to process observations of each agent. Thus reducing the number of observed agents can improve monitoring run-time in practice.

The theorem below takes a step towards addressing this issue by providing more relaxed conditions on the connected nature of the monitoring graph, in particular with respect to the connectivity of the nodes representing key agents. These conditions are: (i) every non-key agent selecting a state P monitors *a single* key agent for each possible pair of plans involving P (i.e., for each pair of plans, where one of the plans is P); and (ii) the monitoring sub-graph for all key agents for a given pair of states forms a clique (i.e., key agents are fully connected between themselves). This case is illustrated in Figure 2-c.

Theorem 1. *Let T be a team in which: (i) Each team-member A, selecting a state P_1, who is not a key agent for P_1, P_2 monitors one key agent for P_1, P_2; (ii) all key agents for a pair of states X, Z monitor all other key agents for X, Z (forming a bidirectional clique in the underlying monitoring graph); (iii) the team is observably-partitioned; and (iv) all monitoring carried out is complete, and uses maximal-coherence. Then disagreement detection in T is sound and complete.*

Proof. By induction on the number of agents in T. The full proof is provided in [19].

This theorem allows teams to overcome significant connectivity limitations, without sacrificing detection quality. The theorem translates into significant freedom for the designer or the agents in choosing whom (if any) to monitor; when a monitored agent is unobservable, an agent may choose to monitor another: Non-key agents need monitor only a single key agent, rather than all key agents (for every pair of states). The upper-bound the theorem provides is more general than may seem at first glance. First, the theorem holds for any state feature of interest—beliefs about a shared environment, goals, etc.; it is up to the designer to pick a monitoring technique that acquires the needed information for constructing the monitoring hypotheses. Second, the theorem does not depend at all on the method by which monitoring occurs, whether by communications or by observations. Thus the connectivity of a monitoring graph does not have to be maintained visually. Some or all of the edges in the monitoring graph may actually correspond to communication links between agents.

Though this theorem represents a significant advance in lowering the bound on the number of agents that must be monitored, all key agents must still monitor each other. This is a critical constraint in practice. For instance, we have reconstructed the visual monitoring graph in *thousands* of RoboCup game situations, to find that even with this new bound, sound and complete disagreement detection would have been possible without communications only in small percentage (approximately 5%) of a game. Typically, each RoboCup player can only see 2–3 key agents, this means that key agents cannot typically monitor all others. To illustrate, Figure 3 shows the monitoring graph of two teams overlayed on a screen-shot of an actual game situation. For both teams, the monitoring graph does not guarantee sound and complete disagreement detection under the known bound, despite the fact that it is connected. This empiric constraint raises the bar on the challenge to find a lower bound on the number of agents that must be monitored to guarantee detection.

4 Model-Based Disagreement Detection

There are, in general, two approaches for detecting (and later, diagnosing) failures [11]. The first is called a consistency-based approach (and sometimes, model-based). A model of the correct behavior of the system is utilized to make predictions as to the observed output of the system in question. When these predictions fail, a fault is detected. Provided that the model is sufficiently detailed, it may be used to identify the exact nature of the failure by a process of model-based diagnosis. The second approach is fault-model-based (fault-based, for short). Here, models of possible faults are matched against the

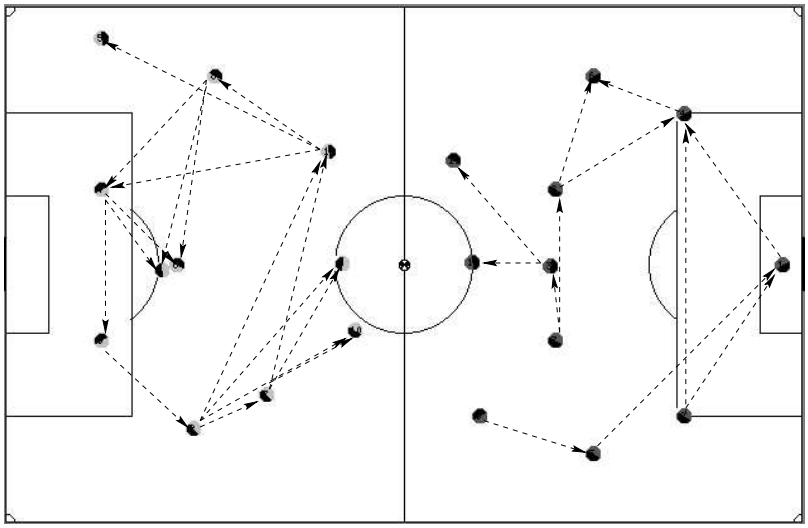


Fig. 3. Monitoring graphs in a RoboCup simulation-league game situation.

observed behavior of the system. When the observed behavior matches the models, an alarm is triggered. Often, fault-models are used together with prescribed resolution procedures, which are called into action to resolve the faults that were detected.

The same two approaches can be found in literature addressing coordination failure detection and diagnosis. On one hand, investigations such as [22, 23, 24, 20, 29] focus on using models of the correct behavior of agents to detect failures as deviations from the model, while others take a fault-based approach [25, 13, 3, 12, 38].

4.1 Detection Based on Fault-Models

We begin by examining the use of fault models to detect coordination failures. Dellarocas and Klein [25, 3] have proposed a centralized approach to detecting failures (which they refer to as *exceptions*) in coordination. Their work utilizes agent sentinels, which monitor agents to identify their state or actions, and report on it to a centralized fault detection system. The system then matches the reported information against a database of known coordination failures, for detection.

An important facet to this work is the population of the fault database. Unlike standard fault-model approaches, where fault models are closely tied to the domain and task at hand, Klein and Dellarocas propose to use general coordination fault-models. These are generated offline, before the deployment of the system, by manual analysis of domain-independent coordination models.

A different—distributed—approach is taken by Horling et al. [13, 12]. They present an integrated failure-detection and diagnosis system for a multi-agent

system in the context of an intelligent home environment. The system uses the TAEMS domain-independent multi-agent task-decomposition and modeling language to describe the ideal behavior of each agent. The agents are also supplied with additional information about the expected behavior of the environment they inhabit under different conditions, and their role within the multi-agent organization. A distributed diagnosis system, made of diagnosis agents that use fault-models, is used to identify failures in components (such as erroneous repeated requests for resources) and inefficiencies (such as over- or under-coordination). The fault-models are used in planning monitoring actions, in identifying failures responsible for multiple symptoms, and in guiding recovery actions. Multiple diagnosis agents may use communications to inform each other of their actions and diagnoses.

A key issue with fault-model approaches is their scalability, given that the number of possible faults in large-scale multi-agent systems is likely to be exponential. Models that attempt to be specific to agents (e.g., "If A does X and B does Y then that is a failure", "If A does X and C does Z then that is a failure") are not likely to scale well. On the other hand, fault models that can utilize some abstraction or capture general failure conditions may do better.

As an example, Wilkins, Lee, and Berry [38] offer an execution monitoring approach which encompasses both coordination and task-execution failures. Their work introduces a taxonomy of generic failure types, which must be adapted and specialized to the domain and task. Agents responsible for monitoring rely on communicated state reports from the monitored agents to identify failures. While experiments with the system were carried out only on relatively small multi-agent systems, the modeling of the failures shows example of how fault-models can be sufficiently non-specific so that they may be reused in larger-scale systems. For instance, the fault models included distance failures (units getting too close), which are triggered when an adversary gets closer to a friendly unit). It does not matter who the adversary or friendly units are, nor their specific location, etc.

A common theme running through all of the above works is that they mostly ignore the issue of uncertainty in monitoring, and utilize communications or direct observations to acquire knowledge as to the state of monitored agents. This is a potentially limiting factor in their use in large-scale networks, where limited connectivity will necessarily lead to uncertainty in monitoring.

4.2 Model-Based Detection

Our own work—and those of others—took a different approach to detecting failures. This consistency-based approach utilizes a model of ideal behavior (in terms of the relationships), not a model of how failure symptoms relate to possible failure diagnoses. The model-based approach has the advantages of generality and model re-use [11]. In particular, fault models, as described

above, are anticipatory; they are only able to capture failures which the designer has been able to anticipate in advance. A consistency-based approach to diagnosing failures is not limited in this respect.

We focus here on disagreement detection. In order to detect disagreements, the monitoring agent must first know which internal states are ideally to be agreed upon. Executable teamwork models such as STEAM [35] or GRATE* [16] allow the designer to specify hierarchical team plans whose execution must be synchronized across agents. To detect a disagreement, we compare the team plans selected by different agents. If they do not match, then a disagreement has occurred [24].

The seeming simplicity of the task is misleading. In the general monitoring case, there can be multiple hypotheses as to the plan selected by each individual. As a result, there can be an exponential number of hypotheses for the team as a whole. To address this, the techniques described in the previous section can guarantee detection results, as long as we select maximally-coherent hypotheses. However, this would seem to require going over the exponential number of hypotheses.

Fortunately, this is not the case. Initial work used the RESL plan-recognition algorithm to represent—implicitly—all possible hypotheses [24]. The savings here were significant, as each agent was modeled individually, and so memory use was $O(NL)$ where N is the number of agents, and L the size of all possible plans for a single agent. However, run-time was still essentially $O(L^N)$, as the algorithm still had to go through multiple hypotheses.

Recently, this result was improved, with the YOYO algorithm [20]. YOYO represents all agents in a single structure, which can only represent fully-coherent hypotheses, i.e., no disagreements. The key observation here is that if something is not representable in YOYO, then it must indicate a disagreement. Thus YOYO detects failures essentially by trying to interpret their actions as if they are not in disagreement. If there is no way to do it, then a disagreement must have occurred. YOYO is thus maximally coherent, and perfectly suited to the monitoring techniques discussed in the previous section. Its space requirements are $O(N+L)$ and runtime is $O(N+L)$. We refer the interested reader to [20] for additional details.

5 Discussion and Future Work

Multi-agent literature has often emphasized that an agent must monitor other agents in order to carry out its tasks. However, as the numbers of agents in deployed teams is scaled up, the challenges of limited connectivity and an exponential number of potential failures are raised. This paper has discussed recent approaches addressing these challenges, in the context of a critical monitoring task—detection of critical disagreements between teammates.

However, many open challenges exist in monitoring large-scale multi-agent systems. One important challenge is in reducing the load on the monitoring

agent. Durfee [7] discussed decision-theoretic and heuristic methods for reducing the amount of knowledge that agents consider in coordinating. The methods include pruning nested (recursive) models, using communications to alleviate uncertainty, using hierarchies and abstractions, etc. This work is complementary to the methods discussed above. We focus on monitoring in teams of cooperating (rather than self-interested) agents, allowing exploitation of the fact that agents are coordinating, both to limit connectivity, as well as to use model-based techniques in detection. Thus, while Durfee's work focuses on reducing computational loads in monitoring each single agent, our work focuses on reducing the number of monitored agents, and on savings possible only when monitoring teams together.

Recent work on model-based diagnosis has also begun to address limited connectivity, though indirectly, and only to a limited extent. Work by Roos et al. [32, 33] has examined the use of model-based diagnosis by agents diagnosing a distributed system. While the methods describe do not address coordination failures, they are certainly relevant in terms of discussing the type of connectivity assumptions required for the diagnosis to work. Our recent preliminary work [18] on the use of model-based diagnosis of disagreements also limits connectivity: A key focus is on using only a handful of agents to represent all others in the diagnosis process, thus limiting runtime and communication load.

Acknowledgment

This work was partially supported by BSF grant #2002401. We thank Michael Bowling, Michael Lindner, and Meir Kalech for useful discussions. As always, thanks to K. Ushi.

References

1. J. J. Burns, E. Salas, and J. A. Cannon-Bowers. Team training, mental models, and the team model trainer. In *Advancements in Integrated Delivery Technologies*, Denver, CO, 1993.
2. P. R. Cohen and H. J. Levesque. Teamwork. *Nous*, 35, 1991.
3. C. Dellarocas and M. Klein. An experimental evaluation of domain-independent fault-handling services in open multi-agent systems. In *Proceedings of the Fourth International Conference on Multiagent Systems (ICMAS-00)*, pages 95–102, Boston, MA, 2000. IEEE Computer Society.
4. M. Devaney and A. Ram. Needles in a haystack: Plan recognition in large spatial domains involving multiple agents. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, pages 942–947, Madison, WI, 1998.
5. M. B. Dias and A. T. Stentz. A free market architecture for distributed control of a multirobot system. In *6th International Conference on Intelligent Autonomous Systems (IAS-6)*, pages 115–122, July 2000.

6. M. B. Dias, R. Zlot, M. Zinck, J. P. Gonzalez, and A. Stentz. A versatile implementation of the traderbots approach for multirobot coordination. In *Proceedings of the Eigth Conference on Intelligent Autonomous Systems (IAS-8)*, 2004.
7. E. H. Durfee. Blissful ignorance: Knowing just enough to coordinate well. In *Proceedings of the First International Conference on Multiagent Systems (ICMAS-95)*, pages 406–413, 1995.
8. M. Fenster, S. Kraus, and J. S. Rosenschein. Coordination without communication: Experimental validation of focal point techniques. In *Proceedings of the First International Conference on Multiagent Systems (ICMAS-95)*, pages 102–108, California, USA, June 1995.
9. B. J. Grosz and S. Kraus. Collaborative plans for complex group actions. *Artificial Intelligence*, 86:269–358, 1996.
10. J. Y. Halpern and Y. Moses. Knowledge and common knowledge in a distributed environment. *distributed computing*, 37(3):549–587, 1990.
11. W. Hamscher, L. Console, and J. de Kleer, editors. *Readings in Model-Based Diagnosis*. Morgan Kaufmann Publishers, San Mateo, CA, 1992.
12. B. Horling, B. Benyo, and V. Lesser. Using self-diagnosis to adapt organizational structures. In *Proceedings of the Fifth International Conference on Autonomous Agents (Agents-01)*, pages 529–536, May 2001.
13. B. Horling, V. R. Lesser, R. Vincent, A. Bazzan, and P. Xuan. Diagnosis as an integral part of multi-agent adaptability. Technical Report CMPSCI Technical Report 1999-03, University of Massachusetts/Amherst, January 1999.
14. M. J. Huber and E. H. Durfee. Deciding when to commit to action during observation-based coordination. In *Proceedings of the First International Conference on Multiagent Systems (ICMAS-95)*, pages 163–170, 1995.
15. S. S. Intille and A. F. Bobick. A framework for recognizing multi-agent action from visual evidence. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, pages 518–525. AAAI Press, July 1999.
16. N. R. Jennings. Controlling cooperative problem solving in industrial multi-agent systems using joint intentions. *Artificial Intelligence*, 75(2):195–240, 1995.
17. M. Kalech and G. A. Kaminka. On the design of social diagnosis algorithms for multi-agent teams. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-03)*, 2003. socially-attentive monitoring, diagnosis, plan-recognition, belief ascription.
18. M. Kalech and G. A. Kaminka. Diagnosing a team of agents: Scaling-up. In *Proceedings of the International Workshop on Principles of Diagnosis (DX 2004)*, 2004.
19. G. A. Kaminka and M. Bowling. Towards robust teams with many agents. Technical Report CMU-CS-01-159, Carnegie Mellon University, 2001.
20. G. A. Kaminka and M. Bowling. Robust teams with many agents. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-02)*, 2002.
21. G. A. Kaminka, Y. Elmaliach, I. Frenkel, R. Glick, M. Kalech, and T. Shpigelman. Towards a comprehensive framework for teamwork in behavior-based robots. In *Proceedings of the Eigth Conference on Intelligent Autonomous Systems (IAS-8)*, 2004.
22. G. A. Kaminka and M. Tambe. What's wrong with us? Improving robustness through social diagnosis. In *Proceedings of the Fifteenth National Conference*

on Artificial Intelligence (AAAI-98), pages 97–104, Madison, WI, 1998. AAAI Press.
23. G. A. Kaminka and M. Tambe. I'm OK, You're OK, We're OK: Experiments in distributed and centralized social monitoring and diagnosis. In *Proceedings of the Third International Conference on Autonomous Agents (Agents-99)*, pages 213–220, Seattle, WA, 1999. ACM Press.
24. G. A. Kaminka and M. Tambe. Robust multi-agent teams via socially-attentive monitoring. *Journal of Artificial Intelligence Research*, 12:105–147, 2000.
25. M. Klein and C. Dellarocas. Exception handling in agent systems. In *Proceedings of the Third International Conference on Autonomous Agents (Agents-99)*. ACM Press, May 1999.
26. S. Kumar, P. R. Cohen, and H. J. Levesque. The adaptive agent architecture: Achieving fault-tolerance using persistent broker teams. In *Proceedings of the Fourth International Conference on Multiagent Systems (ICMAS-00)*, pages 159–166, Boston, MA, 2000. IEEE Computer Society.
27. N. Lesh, C. Rich, and C. L. Sidner. Using plan recognition in human-computer collaboration. In *Proceedings of the Seventh International Conference on User Modelling (UM-99)*, Banff, Canada, 1999.
28. L. E. Parker. ALLIANCE: An architecture for fault tolerant multirobot cooperation. *IEEE Transactions on Robotics and Automation*, 14(2):220–240, April 1998.
29. D. Poutakidis, L. Padgham, and M. Winikoff. Debugging multi-agent systems using design artifacts: The case of interaction protocols. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-02)*, 2002.
30. J. Rickel and W. L. Johnson. Animated agents for procedural training in virtual reality: Perception, cognition, and motor control. *Applied Artificial Intelligence*, 13:343–382, 1999.
31. J. Rickel and W. L. Johnson. Virtual humans for team training in virtual reality. In *Proceedings of the Ninth International Conference on Artificial Intelligence in Education*, pages 578–585. IOS Press, 1999.
32. N. Roos, A. t. Teije, A. Bos, and C. Witteveen. An analysis of multi-agent diagnosis. *in Proceedings of Autonomous Agents and Multi Agent Systems (AAMAS-02)*, July 2002.
33. N. Roos, A. t. Teije, and C. Witteveen. A protocol for multi-agent diagnosis with spatially distributed knowledge. *in Proceedings of Autonomous Agents and Multi Agent Systems (AAMAS-03)*, pages 655–661, July 2003.
34. M. Tambe. Tracking dynamic team activity. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96)*, August 1996.
35. M. Tambe. Towards flexible teamwork. *Journal of Artificial Intelligence Research*, 7:83–124, 1997.
36. M. Tambe, W. L. Johnson, R. Jones, F. Koss, J. E. Laird, P. S. Rosenbloom, and K. Schwamb. Intelligent agents for interactive simulation environments. *AI Magazine*, 16(1), Spring 1995.
37. M. Tambe, G. A. Kaminka, S. C. Marsella, I. Muslea, and T. Raines. Two fielded teams and two experts: A robocup challenge response from the trenches. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-99)*, volume 1, pages 276–281, August 1999.
38. D. E. Wilkins, T. Lee, and P. Berry. Interactive execution monitoring of agent teams. *Journal of Artificial Intelligence Research*, 18:217–261, March 2003.