Towards Social Comparison for Failure Detection: Extended Abstract

Gal A. Kaminka and Milind Tambe

Computer Science Department and Information Sciences Institute
University of Southern California
4676 Admiralty Way, Marina del Rey, CA 90292
(310) 822-1511 {galk, tambe}@isi.edu

Abstract

Social comparison, the process in which individuals compare their behavior and beliefs to those of other agents, is an important process in human societies. Our aim is to utilize theories of this process for synthetic agents, for the purposes of enabling social skills, team-coordination, and greater individual agent performance. Our current focus is on individual failure detection and recovery in multi-agent settings. We present a novel approach, SOCFAD, inspired by *Social Comparison Theory* from social psychology. SOCFAD includes the following key novel concepts: (a) utilizing other agents in the environment as information sources for failure detection, and (b) a detection and recovery method for previously undetectable failures using abductive inference based on other agents' beliefs[1].

Introduction

Social comparison is a process often observed in human societies, in which individuals evaluate their own beliefs, goals, and behavior by comparing themselves to other members of the group, and possibly modify their behavior or beliefs based on the results of that comparison. The process allows individuals within the group to improve their performance and coordination with other members of the group. We have begun operationalizing this process in synthetic agents, for the purpose of improving their social skills, collaboration capabilities, and individual performance. Our current focus is to utilize this process for an individual agent's behavior monitoring.

Agent behavior monitoring is a well known and difficult problem, especially in dynamic environments (e.g., Doyle et al. 1986, Reece and Tate 1994). The unpredictability of such dynamic environments causes an explosion of state space complexity, which inhibits the ability of any designer (human or machine) to enumerate in advance the correct responses for all possible states. In such environments, agents must therefore autonomously detect their own failures and attempt recovery. To this end, an agent must have information about the ideal behavior expected of it. This ideal can be compared to the agent's actual behavior to detect discrepancies indicating possible failures. Previous approaches to this problem (e.g., Doyle et al. 1986, Reece and Tate 1994, Williams and Nayak 1996) have focused on the designer supplying the agent with either a self-model for comparison (Model-Based Diagnosis), or explicitly specified execution-monitoring conditions (ideal ranges of values for percepts and actions). In the model-based approach, the agent's percepts are run through a model to generate ideal actions, which are compared to the actual actions which the agent have chosen. Any discrepancy is a sign of failure. However, if a failure occurs at the percepts themselves the model does not generate any different set of actions -- in a sense, the agent's actions are correct based on the perceptions it has. A similar process takes place in the condition monitoring approach. Thus, these approaches are susceptible to *information failures:* failures where necessary information for comparison is not available, or is faulty but within nominal ranges.

The model-based and condition monitoring approaches may also encounter difficulties in scaled-up domains. If designing a well-behaving agent in a complex, dynamic, unpredictable environment is hard, designing a self-model for such an agent, may be just as hard. In a sense, the same information (what is a correct response given the internal and external state) is to be provided twice: once in designing the agent, and once in its self-model. One facet of this problem is that execution-monitoring conditions can easily become too rigid to describe the flexible, dynamic, ranges of values that are to be expected.

We propose a novel complementary approach to failure detection and recovery, which is unique to multi-agent settings. This approach, SOCFAD (Social Comparison for FAilure Detection), is inspired by ideas from *Social Comparison Theory* (Newell 1990). The key idea in SOCFAD is that agents use other agents as sources of information on the situation and the ideal behavior. The agents compare their own behavior, beliefs, goals, and plans to those of other agents, in order to detect failures in their own behavior. The agents reason about the differences, and draw useful conclusions regarding their own behavior's correctness. This approach allows the agent to infer information from other agents' behavior, replacing its own perceptions. Also, it doesn't require the designer to provide the agent with redundant information about itself, utilizing instead other agents as sources of information.

Motivating Example

Our application domain involves developing automated pilot agents for participation in a multi-agent battlefield simulation environment, which is highly dynamic, complex and rich in detail (Tambe et al. 1995). These qualities present the agents with never-ending opportunities for failure, as anticipation of all possible internal and external states is impossible for the designer. For example, a team of three agents arrives at a specified landmark position. One of the team-members, whose role is that of a *scout*, is to continue forward towards the enemy, identifying and verifying its position. The scout's team-mates (*attackers*) are to wait for its return in the specified position. Due to unanticipated sensory failure, one of the attackers does not detect the landmark marking the waiting point. Instead of waiting behind, it continues to fly forward with the scout, following it into the battlefield, leaving the other attacker alone behind.

We have collected dozens of such failure reports over a period of a few months, despite significant development and maintenance efforts, including the use of some failure detection methods based on known approaches. However, using SOCFAD, an agent as in the example above can at least detect that something may be wrong by noticing that other agents are behaving differently.

Social Comparison: SOCFAD

SOCFAD is inspired by Social Comparison Theory (Festinger 1954), a descriptive theory from social psychology. We have begun to operationalize it for monitoring. SOCFAD's algorithm accepts inputs representing the states of the agents being compared - their beliefs, goals, behavior, etc. These are compared to the agent's own state to detect discrepancies, which would indicate possible failures. Then, a *social similarity* metric is used to reason about which discrepancies are justified, and to what degree. In this way, a level of certainty in the detected failure is produced, which is based on the expected similarity between the agents.

To operationalize SOCFAD, we therefore require: (i) a way of acquiring knowledge of the other agents (so that we have something to compare against); (ii) a procedure to compare agents' states; and (iii) a procedure for measuring the significance of any discrepancies, based on the agents' expected social similarity.

Knowledge of other agents can be communicated. However, such communication is often impractical given costs, risk in hostile territories, and unreliability in uncertain settings. Our implementation of SOCFAD relies instead on agent modeling (plan recognition) techniques that infer an agent's beliefs, goals, and plans from its observable behavior and surroundings. When the monitoring agent has access not only to its own original beliefs, goals, plans, etc., but also to those of its group members, the process of comparison can take place.

Our agents' design is based on reactive plans (operators) (Firby 1987, Newell 1990, Rao et al. 1993), which form hierarchies that control each agent. The design implements the *Joint Intention Framework* (Levesque et al. 1990). Following this framework, operators may be team operators (shared by the team) or individual (specific to one agent). Team operators achieve and maintain joint goals, and require coordination with the other members of the team as part of their application (Tambe 1996, Tambe 1997).

Different capabilities and performance result by changing the information being compared (e.g., internal beliefs and goals vs. observable behavior). It is useful to use information that captures the control processes of the agents. Operator hierarchies are therefore natural objects for modeling and comparison. The agent modeling technique we use, $RESC_{team}$, infers the operator hierarchies being executed by other agents based on their observable actions. Based on the representation of the other agents' plans by operator hierarchies, the comparison process can be implemented by comparing the chosen operators in equal depths of the hierarchies - the hierarchy of the monitoring agent, and the hierarchies for its chosen targets for comparison.

Differences with other agents are meaningful only to the extent that the other agents are *socially similar*. Other agents may not be executing plans that are relevant to the agent's goals, and may therefore be irrelevant for failure detection purposes. Worse yet, other agents may be hostile, intentionally using deception to advance their own agendas. Fortunately, team members tend to work on joint goals and sub-plans related to the one the agent should be executing, and can be assumed to be non-hostile. The comparison process we use in SOCFAD therefore considers team members only.

Team Operator Differences

In the Joint Intentions Framework, explicit team operators form the basis for teamwork, requiring mutual belief on the part of the team members as a condition for the establishment, and termination of team operators. At the team level, members are maximally socially similar, requiring that identical operators be executing. Any difference in team operators between agents in a team is therefore a certain sign of failure, regardless of its cause.

In the example above, one agent has failed to detect a key landmark position and continued execution of the "fly-flight-plan" team operator. However, its teammates correctly detected the landmark and terminated execution of that operator, switching to the "wait-at-point" team operator. Through agent modeling, the miscoordinating agent infers the operators the other agents are executing. It realizes that they could potentially be executing the "wait-at-point" operator and detects a discrepancy with its own team operator of "fly flight plan". At this point it does not know which side is correct, but can conclude with certainty that a failure has occurred within the team and the coordination among its members.

Individual Operator Differences

In service of team operators, different agents may work on different individual operators. These do not carry with them the responsibilities for mutual belief that team operators do, and so differences in individual operators are not sure signs of failure, but at best indications of the possibility. We therefore require additional information about the agents causing the difference which can help in determining whether the difference is justified or not.

Agents working towards similar goals have similar *social roles:* In our example, there were *attackers* and a *scout*. Agents with similar roles would serve as better sources of information. Related to the social role is *social status*, which may also justify differences in individual operators among team members. For instance, in the military domain agents of different ranks may follow different individual operators to guide their behavior.

We have provided our agent with the means to explicitly consider the social role and status of other agents within the team in filtering and assigning weights to the information inferred about them. For example, if the agent is an *attacker*, which is one of the roles in a team in our domain, it will assign more weight to other agents which are *attackers*.

Towards Socially-Based Recovery

From the fact that other agents are executing a different plan, the agent can conclude with some certainty that a failure has occurred, but it cannot tell which of the sides is correct. If the agent believes it is at fault, it can alter its own beliefs by adopting the (inferred) beliefs of the other agents. In particular, team operators require mutual belief in pre-conditions, and so by adopting them the agent re-synchronizes itself with the rest of the team. For example, the agent in the landmark example overcame its sensory problem by adopting its team-mates inferred belief in the landmark being reached even though its own sensors didn't support this belief. This fulfilled the preconditions of its own "wait-at-point" operator, which was selected and allowed the agent graceful recovery from the failure.

Summary and Future Work

This paper presents a novel approach to failure detection, an important problem plaguing multi-agent systems in large-scale, dynamic, complex domains. Existing approaches often face difficulty in addressing this problem in such domains. The key novelties of our approach are: (a) a new failure detection method, utilizing other agents in the environment as information sources for comparison, (b) a general heuristic for team-based comparison, and (c) a detection and repair method for (previously undetectable) information failures using abductive inference based on other agents' beliefs.

The social comparison approach to failure detection complements previous methods, being able to detect different types of failures. In general, previous approaches use the agent's own inputs to generate an ideal output which is compared to the actual output to detect problems in the process converting inputs to outputs. Thus they are limited by the quality of the agent's own perceptions. However, SOCFAD can detect such failures and correct them as demonstrated above.

Several issues are open for future work. One important issue is in techniques and biases useful for deciding which side is correct where a difference is encountered with another agent, but no information is known to support either side. A simple technique that may be used is to follow the majority, so that if a majority of agents agree with one agent, its beliefs and behavior is taken to be correct. Such a technique has clear limitations, and improvements continue to be a subject for future work.

References

1. Doyle R. J., Atkinson D. J., Doshi R. S., Generating perception requests and expectations to verify the execution of plans, in Proceedings of AAAI-86, Philadelphia, PA (1986).
2. Festinger, L. 1954. A theory of social comparison processes. Human Relations, 7, pp. 117-140.
3. Firby, J. 1987. An investigation into reactive planning in complex domains. In Proceedings of the National Conference on Artificial Intelligence (AAAI-87).
4. Levesque, H. J.; Cohen, P. R.; Nunes, J. 1990. On acting together, in Proceedings of the National Conference on Artificial Intelligence (AAAI-1990), Menlo Park, California, AAAI Press.
5. Newell A., 1990. Unified Theories of Cognition. Harvard University Press.
6. Reece, G. A.; and Tate, A. Synthesizing protection monitors from causal structure, in Proceedings of AIPS-94, Chicago, Illinois (1994).
7. Rao, A. S.; Lucas, A.; Morley, D., Selvestrel, M.; and Murray, G. 1993. Agent-oriented architecture for air-combat simulation. Technical Report: Technical Note 42, The Australian Artificial Intelligence Institute.
8. Tambe, M.; Johnson W. L.; Jones, R.; Koss, F.; Laird, J. E.; Rosenbloom, P. S.; and Schwamb, K. 1995. Intelligent Agents for interactive simulation environments. AI Magazine, 16(1) (Spring).
9. Tambe, M. 1996. Tracking Dynamic Team Activity, in Proceedings of the National Conference on Artificial Intelligence (AAAI-96), Portland, Oregon.
10. Tambe, M. 1997. Agent Architectures for Flexible, Practical Teamwork, in Proceedings of the National Conference on Artificial Intelligence, Providence, Rhode Island (To appear).
11. Williams, B. C.; and Nayak, P. P. 1996. A Model-Based Approach to Reactive Self-Configuring Systems. In Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96), Portland, Oregon.