Distributed Intrusion Detection in Partially Observable Markov Decision Processes

Doran Chakraborty
doran@utulsa.edu

Sandip Sen
sandip@utulsa.edu

Mathematical & Computer Sciences Department
University of Tulsa
Tulsa, Oklahoma, USA

1. ABSTRACT

The problem of decentralized control occurs frequently in realistic domains where agents have to cooperate to achieve a universal goal. Planning for domain-level joint strategy takes into account the uncertainty of the underlying environment in computing near-optimal joint-strategies that can handle the intrinsic domain uncertainty. However, uncertainty related to agents deviating from the recommended joint-policy is not taken into consideration. We focus on hostile domains, where the goal is to quickly identify deviations from planned behavior by any compromised agents. There is a growing need to develop techniques that enable the system to recognize and recover from such deviations. We discuss the problem from the intruder's perspective and then present a distributed intrusion detection scheme that can detect a particular type of attack.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent systems*

General Terms

Algorithms, Performance

Keywords

fault tolerance, multiagent planning

2. INTRODUCTION

The problem of decentralized control can be effectively modeled as a decentralized partially observable Markov Decision Process (DEC-POMDP) [1] where the agents in the model are the decision makers. The uncertainty related to each agent's action does not only span over the uncertainty of the underlying environment but also on the decision making of the other agents in the system. In hostile domains, malfunctioning of the agents due to internal failure or even due to enemy intrusion cannot be discounted. There is a growing need to develop multiagent systems that can address such situations and can efficiently compensate for such setbacks. We analyze the problem from the perspective of the malicious agent and come up with one possible attack that it can launch on the system. We then use decision theoretic paradigms to obtain a solution to such attacks and propose reputation update schemes that distributively identify such malicious agents. We substantiate our claims with results from the popular *multiagent tiger domain* [2].

AAMAS'07 May 14–18 2007, Honolulu, Hawai'i, USA.

3. DEC-POMDP MODEL

A DEC-POMDP is given by a tuple $< I, S, \{A_i\}, \{\Omega_i\}, O, P, R, b_0 >$ where I is the finite set of agents indexed by $1 \ldots n$, S is a finite set of states, A_i is a finite set of actions available to agent i and $A = \times_{i \in I} A_i$ is the set of joint actions where a $=< a_1, \ldots, a_n >$ denotes a joint action, Ω_i is a finite set of observations available to agent i and $\Omega = \times_{i \in I} \Omega_i$ is the set of joint observations where $o =< o_1, \ldots, o_n >$ denotes a joint observation, O is the observation function given by $O(s, a_1, \ldots, a_n, o_1, \ldots, o_n)$, the probability of observing the joint observation $(o_1, \ldots, o_n)$ when transitioning to state s after taking joint action $(a_1, \ldots, a_n)$, P is the set of Markovian state transition probabilities where $P(s, a, s')$ denotes the probability of taking action a in state s and reaching state s', R: $S \times A \rightarrow \Re$ is the common reward function, and b_0 is the initial belief state for all agents. We assume that the agent's observations are independent. Thus the observation function can be represented as $O = \times_{i \in I} O_i$ where $O_i(s, a_1, \ldots, a_n, o_i)$ is the probability that agent i observes o_i given the joint-action $< a_1, \ldots, a_n >$ resulted in state s. The decision problem spans over a finite horizon T.

4. POLICY REPRESENTATION AND RUNNING BELIEF STATE

The policy for agent i, π_i is represented by a policy tree (see Figure 1). Each node corresponds to an action and each edge corresponds to an observation that the agent makes at that time interval. We assume that there is a centralized planner that computes the policy tree for each agent. We introduce the concept of running belief state of an agent as an execution time estimate of an agent over the physical states and the observation histories of the other agents in the system. The running belief state of agent i at time interval t is given by $RB_i^t : S \times o_{-i}^{\vec{t}} \rightarrow [0, 1]$ where $o_{-i}^{\vec{t}}$ are the *t'th*

observation histories of other agents. We define Bel_i^t as the set of all such possible combinations of physical states and observation histories that have a positive probability in RB_i^t. Thus Bel_i^t can be defined as $Bel_i^t = \{b | RB_i^t(b) > 0\}$.

4.1 Running Belief State Update

The agents update RB_i^t and Bel_i^t with each execution step. The update occurs in two phases. The first exhaustive phase calculates RB_i^{t+1}, based on the observation o_i^{t+1} from Bel_i^t. We call this estimate the priori estimate, $RB_{i,t+1}^{priori}$.

$$RB_{i,t+1}^{priori}(< s', o_{-i}^{\vec{t+1}} >) = \alpha_1 \times \sum_{< s, o_{-i}^{\vec{t}} > \in Bel_i^t} RB_i^t(< s, o_{-i}^{\vec{t}} >) \times$$

$$P(s, < \pi_i(o_i^{\vec{t}}), \pi_{-i}(o_{-i}^{\vec{t}}) >, s') \times O_i(s', < \pi_i(o_i^{\vec{t}}), \pi_{-i}(o_{-i}^{\vec{t}}) >,$$

$$o_i^{t+1}) \times O_{-i}(s', < \pi_i(o_i^{\vec{t}}), \pi_{-i}(o_{-i}^{\vec{t}}) >, o_{-i}^{t+1}) \quad (1)$$

where α_1 is a normalizing constant and $s, s' \in S$. $Bel_{i,t+1}^{priori}$ is calculated accordingly. The second phase of the update happens after the agent takes its *t+1*'th step action and obtains joint reward given by the random variable R^{t+1}. The agent can now calculate the posteriori estimate $RB_{i,t+1}^{posteriori}$,

$$RB_{i,t+1}^{posteriori}(< s', o_{-i}^{\vec{t+1}} >) = \alpha_2 \times RB_{i,t+1}^{priori}(< s', o_{-i}^{\vec{t+1}} >) \times$$

$$Pr(R^{t+1} = r | R, s', < \pi_i(o_i^{\vec{t+1}}), \pi_{-i}(o_{-i}^{\vec{t+1}}) >) \quad (2)$$

where α_2 is another normalizing constant and r is the reward received at the *t+1*'th time step. The second probability term in Equation 2 returns a value of 1 or 0 depending on whether it is possible to get the reward r after taking joint action $< \pi_i(o_i^{\vec{t+1}}), \pi_{-i}(o_{-i}^{\vec{t+1}}) >$ at state s'. We then set $RB_{i,t+1}^{posteriori}$ to RB_i^{t+1} and Bel_i^{t+1} is updated accordingly. Agent i knows its own physical state and using RB_i^t and Bel_i^t, it tries to approximately estimate the state of all $j, j \in I, j \neq i$. It can be shown that Bel_i^t is always an over-estimate of the actual world state.

4.2 Anticipated set of rewards

Based on $Bel_{i,t}^{priori}$, an agent i maintains a set ψ_i^t which is a set of all possibles rewards that it can get by taking the action given by π_i at time t. The anticipated set of rewards η_i^t at time t is $\eta_i^t = \{R(s, < \pi_i(o_i^{\vec{t}}), \pi_{-i}(o_{-i}^{\vec{t}}) >) | (s, o_{-i}^{\vec{t}}) \in Bel_{i,t}^{priori}\}$. Note that the agent has to use the $Bel_{i,t}^{priori}$ estimate as the anticipated rewards are calculated before the t'th action has been taken. If an agent i receives a reward $r \notin \eta_i^t$ at time t, it concludes that some agent(s) have deviated from their committed policies.

We assume that there can be compromised agents in the system which want to have a covert harmful impact on the system. A malicious agent k can only avoid suspicion if it takes actions which are consistent with the anticipated reward estimates of other agents. The moment k takes an action that produces a joint reward $r \notin \eta_i^t$ for some i, i knows that some agent(s) must have deviated from the committed joint policy. k can choose from the set of actions available with the different observations for the particular node of policy tree it is in and thereby avoid detection. We refer to such agents as simple malicious agents. In Figure 1, if k is at branch $b1$ of π_k at time t', it chooses between the actions a_1 and a_2 pertaining to observations o_1 and o_2 respectively.

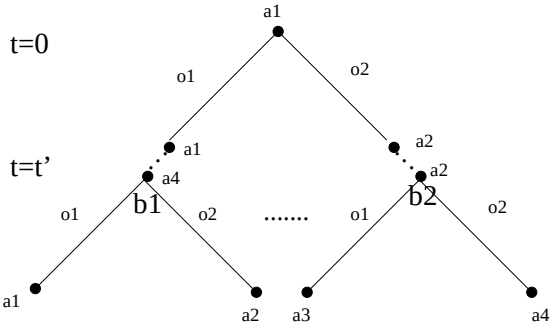

Figure 1: Policy tree.

k has no incentive to choose an action from another node at the same level of the policy tree (b_2) as it cannot guarantee that the joint reward received would be a member of η_i^t, $\forall i$, $i \neq k$. We refer to such agents as simple malicious agents. A simple malicious agent can be deterministic if it always takes the action corresponding to the least probable observation of the subtree of the policy tree it is in or stochastic if it takes any action corresponding to the observations of the subtree with equal probability.

4.3 Reputation management

To identify deviation from committed plans from compromised agents, each agent maintains updated reputation about all other agents in the system. This reputation management scheme runs in all agents in completely distributed fashion. Each agent i maintains a set $V_i = \{R_i^j\}$ where R_i^j is the reputation of the jth agent as computed by agent i, $\forall j \neq i$, and is updated in each iteration by:

$$R_i^j \leftarrow R_i^j - \kappa(R_i^j) \times$$

$$\sum_{\forall < s, o_{-i}^{\vec{t}} > \in Bel_i^{t^{max}}} (max_{o_j \in \Omega_j} O_j(s, < \pi_i(o_i^{\vec{t-1}}), \pi_{-i}(o_{-i}^{\vec{t-1}}) >, o_j)) -$$

$$O_j(s, < \pi_i(o_i^{\vec{t-1}}), \pi_{-i}(o_{-i}^{\vec{t-1}}) >, o_j^t >)) / |Bel_i^{t^{max}}| \quad (3)$$

where $Bel_i^{t^{max}} = \{b | RB_i^t(b) = max_{b' \in Bel_i^t} RB_i^t(b')\}$. $Bel_i^{t^{max}}$ is a subset of beliefs most convincing to i. Based on $Bel_i^{t^{max}}$, i tries to reason the last observational transition that each of the other agents have made. The numerator of the second expression in Equation 3 gives the difference between the probability of observing the most probable observation given *t-1*'th joint-action with the probability of observing the t'th observation given the same. Note, a simple malicious agent k would often fake observations and this inconsistency would gradually reflect in the Bel_i^t of i and result in a higher value for the numerator. This would result in a sharp drop of R_i^k. The function κ is monotonically decreasing with R_i^j and thus facilitates faster detection.

5. RESULTS

For our experiments we use a three agent version of the Tiger problem [2] and refer to the agents as Agent1, Agent2 and Agent3. The observation function for each agent is generated randomly over each pass of the run. We assume domains, where probability of one observation given a state and joint-action is significantly higher than the other observations. All results presented are averaged over ten runs.

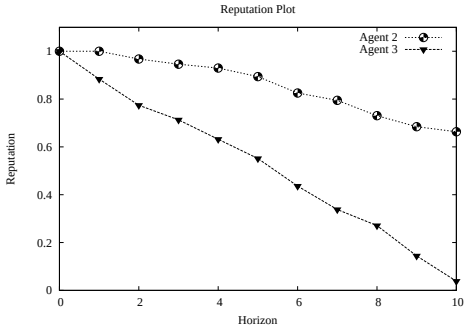

Figure 2: Agent3 deterministically takes actions corresponding to wrong observation.

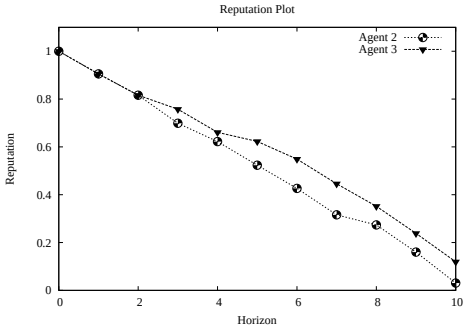

Figure 3: Both agent2 and agent3 deterministically takes actions corresponding to wrong observation.

The initial policies for each agent is generated randomly for each iteration. We do without calculating the optimal joint policy as the reputation mechanism works well for any committed joint policy. The results presented are based on the calculations made by Agent1.

In the first simulation, Agent3 is a deterministic simple malicious agent whereas Agent2 takes actions based on its true observations. Figure 2 shows the individual reputation plots of the two agents in the system. The drop in reputation for Agent2 is due to the domain effect but the much steeper drop for Agent3 is far from the normal drop due to the domain level uncertainties. In the next simulation, we changed Agent3 to a deterministic simple malicious agent. Figure 3 shows the corresponding reputation plots and we see that the reputation of both the agents show a sharp fall.

In the next set of experiments, in an attempt to avoid detection, the simple malicious agents stochastically chooses the action pertaining to the wrong observation. Again in the first case we assume that Agent3 is the only malicious agent in the system. Figure 4 is the reputation plot for the corresponding case. The steeper drop in the reputation value for Agent3 clearly identifies that Agent3 have been taking more actions pertaining to wrong observations. So Agent1 can conclude that either the agent has faulty observational sensors or it has been corrupted by external influence. In the next simulation, both Agent2 and Agent3 behave as malicious agents that stochastically take actions corresponding to the wrong observation. Figure 5 shows the corresponding reputation plot of the two agents. The sharp dip in the reputation of the two agents signify an abnormality in the expected behavior of the agents.

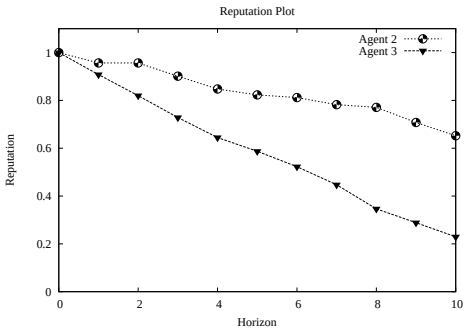

Figure 4: Agent3 stochastically takes actions corresponding to wrong observation.

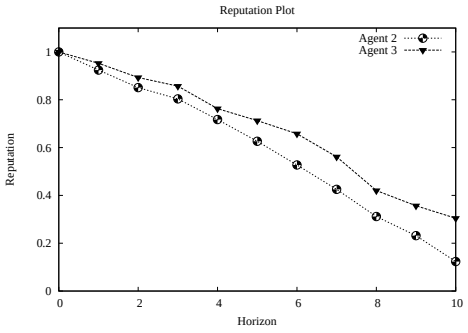

Figure 5: Both Agent2 and Agent3 stochastically take actions corresponding to wrong observation.

6. CONCLUSION AND FUTURE WORK

In this paper we dealt with the problem of detecting agents that deviate from committed joint-policy. Assuming that the agents can be compromised, we discussed one possible type of deviations from prior commitments. We presented a distributed reputation update mechanism of uniquely detecting such malicious agents. Our scheme is based on tracking of the execution time estimates of the belief state of the system and the anticipated rewards. As future work we would also like to explore more challenging scenarios dealing with malicious agents such as colluding agents that try to compromise the system.

Acknowledgment: US National Science Foundation award IIS- 0209208 partially supported this work.

7. REFERENCES

[1] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein. The complexity of decentralized control of markov decision processes. *Math. Oper. Res.*, 27(4):819–840, 2002.

[2] R. Nair, M. Tambe, M. Yokoo, D. V. Pynadath, and S. Marsella. Taming decentralized pomdps: Towards efficient policy computation for multiagent settings. In *Proceedings of the Eighteenth International Joint conference on Artificial Intelligence (IJCAI)*, 2003.