Detecting and Tracking Hostile Plans in the Hats World

Aram Galstyan and **Sinjini Mitra** and **Paul Cohen**
USC Information Sciences Institute
Center for Research on Unexpected Events (CRUE)
4676 Admiralty Way, Marina del Rey, CA 90292
{galstyan,mitra,cohen}@isi.edu.

Abstract

Plan recognition is the problem of inferring an agent's hidden state of plans, or intentions, based on the pattern of his observable actions. Among many potential applications, intention recognition can be particularly useful for intelligence analysis and homeland security related problems. While most of the existing work in plan recognition has focused on studying overt agents, problems from the intelligence domain usually have different settings, where hostile agents operate covertly in a large population of otherwise benign entities. In this paper we formulate a problem of detecting and tracking hostile intentions in such an environment – a virtual world where a large number of agents are involved in individual and collective activities. Most of the agents are benign, while a small number of them have malicious intent. We describe our initial effort for building a probabilistic framework for detecting hostile activities in this system, and provide some initial results for simple scenarios.

Introduction

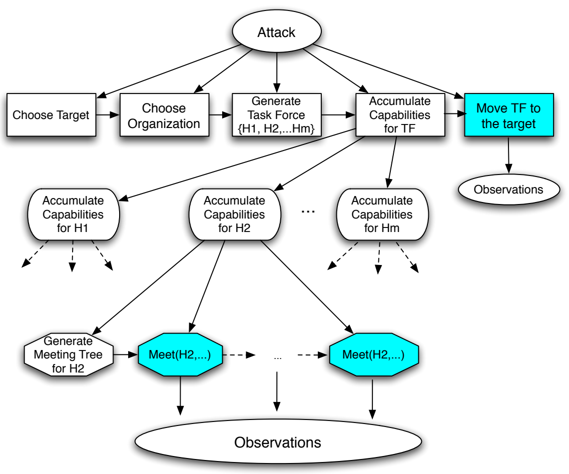
Plan recognition is the problem of inferring an agent's hidden state of plans, or intentions, based on the pattern of his observable actions. Clearly, the ability to efficiently and accurately recognize plans and/or intentions of adversarial agents can be very useful for intelligence analysis and other homeland security related problems. There has been an extensive amount of work on plan recognition in recent years. Most of the existing studies, however, consider scenarios that are not very suitable from the perspective of intelligence analysts, and do not adequately address challenges posed by plan recognition problems in the intelligence domain. For instance, most of the existing work is concerned with situations where the identities of agents engaged in the activity of interest are known in advance. While this assumption is certainly justified in some applications, it does not hold in realistic intelligence analysis scenarios where adversarial agents operate covertly, e.g., by embedding themselves in a much larger, benign population. Thus, inferring the identities of those covert agents is a very important aspect of the problem. Furthermore, majority of the existing approaches address single–agent scenarios. In the intelligence domain, however, observations often describe interactions between agents: meetings, financial transactions, and other forms of communication. Thus, taking into account this relational aspect of the data is important.

In this paper we formulate a problem of detecting and tracking hostile intentions in a virtual domain, the Hats world. Hats simulates a virtual society composed of large number of agents that are involved in individual and collective activities. Each agent carries a set of attributes, that might change dynamically through trades and/or exchanges. The overwhelming majority of the agents are benign. However, a small fraction of them are covert adversaries and intend to do harm by destroying certain landmarks. The goal of the analyst is to detect and neutralize malicious groups before the attacks, based on limited and noisy information about their activities. While the model of agent behavior in Hats is clearly oversimplified, it shares important characteristics with problems faced by intelligence analysts. Specifically, what makes the Hats problem challenging is the extremely low *signal to noise* ratio, and the huge number of hypotheses one needs to manage, which is usually the case in real–world intelligence scenarios.

The rest of this paper is organized as follows: In the next section we provide a more detailed description of the Hats domain. We then describe a Bayesian framework for tracking agent activities in a simplified Hats scenario, and illustrate this framework on two examples. We conclude the paper by the review of the relevant literature and a brief discussion about future work.

The Hats Domain

One of the hindrances for researchers who want to build tools for intelligence analysis is the lack of available intelligence data. The Hats simulator (Cohen, P.R. & Morrison, C.T. 2004; Morrison, C.T. *et al.* 2005) was proposed to fill this gap, by providing synthetic intelligence data to researchers and serving as a proxy for various intelligence analysis problems. Hats simulates a *society in a box*, where a large number of agents, called *hats*, are engaged in individual and collective activities. Agents reside on a two dimensional grid-world and travel to different locations to attend meetings. Each agent has a set of elementary attributes, called *capabilities*, which he can trade with other agents at those meetings. The overwhelming majority of the agents are benign. Some agents, however, are covert terrorists, and



Attack
Choose Target
Choose Organization
Generate Task Force {H1, H2,...Hm}
Accumulate Capabilities for TF
Move TF to the target
Observations
Accumulate Capabilities for H1
Accumulate Capabilities for H2
Accumulate Capabilities for Hm
Generate Meeting Tree for H2
Meet(H2,...)
Meet(H2,...)
Observations

Figure 1: Hierarchical representation of a plan

intend to inflict harm by destroying certain landmarks called *beacons*. Beacons are special locations on the grid and are characterized by their *vulnerabilities*. A successful attack on a beacon happens when a group of adversarial agents moves to the location of a beacon and has all the capabilities that match the beacon vulnerabilities. Beacon attacks are planned by a generative planner that chooses the target, sets up a task-force, and builds a meeting schedule for the task-force members, so that they will acquire the capabilities required for the attack.

Along with the covert adversaries, there also are hats that are known to be adversaries. Each hat belongs to one or more organizations that can be of two types, benign or adversary. Each adversarial hat belongs to at least one adversary organization, while each benign agent belongs to at least one benign organization and does not belong to any adversary organization. When a meeting is planned, the list of participants is drawn from the set of hats belonging to the same organization. Consequently, a meeting planned by an adversary organization will consist of only adversary (either known or covert) hats, whereas a meeting planned by a benign organization might contain all types of hats.

Instead of choosing a task–force that already possesses the required capabilities, the planner designs an elaborate sequence of meetings for passing capabilities among the agents. This is intended for masking the true intention of the task–force. In the Hats simulator, the actual plans are constructed as inverted trees, where each node describes a meeting between agents, and the root node corresponds to the final meeting at the beacon. More information about meeting planner in Hats can be found in (Cohen, P.R. & Morrison, C.T. 2004; Morrison, C.T. *et al.* 2005). For the purposes of this paper, however, we will consider a little simplified model for plans. In particular, we assume that the observable activities are a result of a certain hidden generative process, that can be represented hierarchically as depicted in Figure 1. Namely, the attack is planned by choosing a target and an organization that will carry out the attack. Then, a task–force composed of the members of that organization is chosen. This task–force determines the group of agents that will participate in the final meeting at the location of the beacon. Next, the planner will generate a schedule for meetings and trades, so that members of the task–force acquire capabilities matching the vulnerability of the beacon. Finally, after all the capabilities are acquired, the task–force members will move to the location of the beacon for the final meeting. We note that only highlighted nodes produce directly observable actions.

Bayesian Framework for Tracking

We now consider a simplified Hats scenario with only two organizations, one benign and one terrorist. Furthermore, we assume that each agent can hold only one capability at any given time. Let us characterize the state of the i-th agent at time n by a tuple $x_n^i = (\xi_n^i, \alpha_n^i, \theta_n^i)$. Here $\xi_n^i \in \{0, 1\}$ is a terrorist indicator variable for agent i, so that $\xi_n^i = 1$ means that agent i is a member of the terrorist organization; $\alpha_n^i \in \{0, 1, 2, \dots, M\}$ indicates the *intention* of the i-th agent to acquire a certain capability out of M possible ones ($\alpha_n^i = 0$ means the agent does not intend to get any capability at all); and θ_n^i denotes the actual capability carried by the agent i at time n, e.g., $\theta_n^i = k, k \in \{1, 2, \dots, M\}$. The joint state of the system is defined by $\mathbf{x}_n = \{x_n^1, x_n^2, ..., x_n^N\}$. Initially, all our knowledge is represented by the prior distribution over the states, $\mathbf{P}_0(\mathbf{x}_0)$. Usually, this prior distribution will have a factored representation owing to independence among the individual agents at the beginning,

$$\mathbf{P}_0(\mathbf{x}_0) = \mathbf{P}_0^1(x_0^1)\mathbf{P}_0^2(x_0^2)\dots\mathbf{P}_0^N(x_0^N), \qquad (1)$$

where $\mathbf{P}_0^i(x_0^i)$ represents our prior belief about the i–th agent.

Starting from the initial state, agents will participate in meetings and might change their internal state by acquiring capabilities at those meetings. We characterize this hidden state dynamics by a Hidden Markov Model (HMM) with observation–dependent state transitions as depicted in Figure 2. Here $\mathbf{X}_n$ and $\mathbf{Y}_n$ are the *hidden state* and *observation* (e.g., observed meetings) at time n, respectively. The influence diagram shows that the hidden state at time $n+1$ depends on the hidden state at time n, as well as the observation at time n. The reason for the latter dependence is that

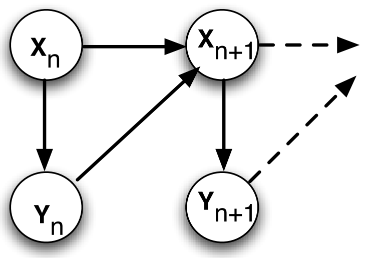


Figure 2: Schematic diagram of an HMM with states depending on observations.

whether an agent will change its state (e.g., acquire a certain capability) depends on the meeting it has participated in (e.g., observation). At the same time, the observation at time n is determined only by the corresponding hidden state $\mathbf{X}_n$.

To specify the HMM we need to define the *transition matrix* and the *emission model*. The transition matrix is given by

$$M(\mathbf{x}, \mathbf{x}'; \mathbf{y}) = \mathbf{P}(\mathbf{X}_n = \mathbf{x} | \mathbf{X}_{n-1} = \mathbf{x}', \mathbf{Y}_{n-1} = \mathbf{y}), \quad (2)$$

which is the probability that the system will be in state $\mathbf{X}_n = \mathbf{x}$ at time n, given that it was in state $\mathbf{X}_{n-1} = \mathbf{x}'$ and produced an observation $\mathbf{Y}_{n-1} = \mathbf{y}$ at time $n-1$.

The emission model specifies the probability of seeing a particular observation in a particular state:

$$\Psi(\mathbf{y}, \mathbf{x}) = \mathbf{P}(\mathbf{Y}_n = \mathbf{y} | \mathbf{X}_n = \mathbf{x}). \quad (3)$$

The Bayesian filtering problem is concerned with assessing the state of the system at time n given the history of observations up to time n, $\mathbf{y}_0^n = \{\mathbf{y}_0, \mathbf{y}_1, ..., \mathbf{y}_n\}$. This is done through estimating the so called *filtering distribution* $\mathbf{P}_n(\mathbf{x}) = \mathbf{P}(\mathbf{X}_n = \mathbf{x} | \mathbf{y}_0^n)$. In practice, it is easier to work with the un–normalized filtering distribution defined as $\Phi_n(\mathbf{x}, \mathbf{y}_0^n) = \mathbf{P}(\mathbf{X}_n = \mathbf{x}, \mathbf{Y}_0^n = \mathbf{y}_0^n)$. The normalized distribution is obtained by dividing $\Phi_n(\mathbf{x}, \mathbf{y}_0^n)$ by the data likelihood:

$$\mathbf{P}_n(\mathbf{x}) = \frac{\Phi_n(\mathbf{x}, \mathbf{y}_0^n)}{\mathbf{P}(\mathbf{y}_0^n)} = \frac{\Phi_n(\mathbf{x}, \mathbf{y}_0^n)}{\sum_{\mathbf{x}} \Phi_n(\mathbf{x}, \mathbf{y}_0^n)}. \quad (4)$$

We now derive a recursive equation for $\Phi_n(\mathbf{x}, \mathbf{y}_0^n)$ as follows. Let us write it as

$$\begin{aligned}\Phi_n(\mathbf{x}, \mathbf{y}_0^n) &= \mathbf{P}(\mathbf{Y}_n = \mathbf{y}_n | \mathbf{X}_n = \mathbf{x}, \mathbf{Y}_0^{n-1} = \mathbf{y}_0^{n-1}) \\ &\times \mathbf{P}(\mathbf{X}_n = \mathbf{x}, \mathbf{Y}_0^{n-1} = \mathbf{y}_0^{n-1}) \\ &= \Psi(\mathbf{y}_n, \mathbf{x}) \times \mathbf{P}(\mathbf{X}_n = \mathbf{x}, \mathbf{Y}_0^{n-1} = \mathbf{y}_0^{n-1}),\end{aligned} \quad (5)$$

where in the last equation we have used the fact that, conditioned on state $\mathbf{X}_n$, the observation at time n, $\mathbf{Y}_n$, does not depend on previous observations.

Let us now rewrite the second term as follows:

$$\begin{aligned}&\mathbf{P}(\mathbf{X}_n = \mathbf{x}, \mathbf{y}_0^{n-1}) = \\ &\sum_{\mathbf{x}'} \mathbf{P}(\mathbf{X}_n = \mathbf{x}, \mathbf{X}_{n-1} = \mathbf{x}', \mathbf{y}_0^{n-1}) = \\ &\sum_{\mathbf{x}'} \mathbf{P}(\mathbf{X}_n = \mathbf{x} | \mathbf{X}_{n-1} = \mathbf{x}', \mathbf{y}_0^{n-1}) \mathbf{P}(\mathbf{X}_{n-1} = \mathbf{x}', \mathbf{y}_0^{n-1}) \\ &= \sum_{\mathbf{x}'} M(\mathbf{x}, \mathbf{x}'; \mathbf{y}_{n-1}) \Phi_{n-1}(\mathbf{x}', \mathbf{y}_0^{n-1}).\end{aligned} \quad (6)$$

Substituting Equation 6 into Equation 5 we arrive at the following recursive update formula:

$$\Phi_n(\mathbf{x}, \mathbf{y}_0^n) = \Psi(\mathbf{x}, \mathbf{y}_n) \sum_{\mathbf{x}'} M(\mathbf{x}, \mathbf{x}'; \mathbf{y}_{n-1}) \Phi_{n-1}(\mathbf{x}', \mathbf{y}_0^{n-1}) \quad (7)$$

Once an emission model and state transition model are specified, Equation 7 can be used for state estimation. Below we apply this framework to a couple of simple examples.

Bayesian Guilt by Association (GBA) Model

In the first example, we neglect any capability trades, and focus on estimating group–membership of agents, based on the observation of meetings between them. Specifically, assume there are N agents indexed $i = 1, .., N$, each of which can belong to one of two organizations: one adversary, and the other benign. Thus, the state of the system is fully characterized by a binary vector $\mathbf{x} = \{\xi_n^1, \xi_n^2, ...\xi_n^N\}$. Also, let us assume that agents never change their adversary/benign status, so the time index can be dropped.

Let us now specify the emission model. For the sake of simplicity, we will assume that at any given time, a meeting contains only two agents. Thus, each observation can be represented as a pair of agent ID–s: $\mathbf{y}_n = \{i, j\}$. Further, we assume that the probability of a meeting between same agents (that is, both adversary or both benign) is $p > 0.5$, and the probability of a meeting between agents from different groups (one adversary and one benign) is $1-p$. In other words, probability of a meeting between agents i and j is p if $\xi^i = \xi^j$, and $1-p$ if $\xi^i \neq \xi^j$. This can be written as follows:

$$\begin{aligned}\Psi(\mathbf{y} = \{i, j\}, \mathbf{x} &= \{\xi_1, .., \xi^N\}) = p\delta(\xi^i, \xi^j) + (1-p) \\ &\times (1 - \delta(\xi^i, \xi^j)),\end{aligned} \quad (8)$$

where we have used the Kronecker's δ–function: $\delta(i, j) = 1$ if $i = j$, and $\delta(i, j) = 0$, $i \neq j$.

We now specify the transition matrix. Since agents do not change their organization (e.g., no recruitment), there are simply no transitions. Thus, the state transition is described by the identity matrix:

$$M(\mathbf{x}, \mathbf{x}'; \mathbf{y}) = \delta(\mathbf{x}, \mathbf{x}'). \quad (9)$$

Plugging Equation 9 into 7 yields the following simple recursion rule:

$$\Phi_n(\mathbf{x}, \mathbf{y}_0^n) = \Psi(\mathbf{y}_n, \mathbf{x}) \times \Phi_{n-1}(\mathbf{x}, \mathbf{y}_0^{n-1}) \quad (10)$$

Note that given $\Phi_n(\mathbf{x}, \mathbf{y}_0^n)$, or more precisely, its normalized form $\mathbf{P}_n(\mathbf{x} = \{\xi^1, \xi^2, ..., \xi^N\})$, one can calculate the marginal probability that a given agent is a terrorist, by summing over all the other agents. In Fig. 3 we show these individual probabilities for a system of 20 agents, 10 of which have known identities and the remaining 10 are unknown (half of them are benign and the other half are terrorist). The value of the parameter is $p = 0.8$. In this particular experiment, the algorithm correctly identifies five covert adversaries, which is reflected in high posteriors for the corresponding agents. We also note that the exact tracking performed here becomes infeasible for sufficiently large systems as the state space grows exponentially with the number of agents. Thus, one will have to resort to approximate methods such as Markov Chain Monte Carlo (MCMC) and particle filters. In particular, we will be investigating the application of Sequential Importance Sampling (SIS; (Doucet, A., de Freitas, N., & Gordon, N. Eds. 2001)), which is known to be helpful in estimating the filtering distribution in case of high-dimensional data.

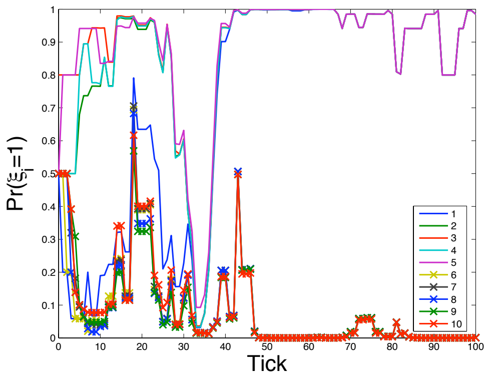


Figure 3: Marginal probabilities (y-axis) of being a terrorist vs time (shown as tics on the x-axis) for 8 agents represented by different colored lines. There is a total of 20 agents in this experiment, of which the identities of 10 are not known (half of them are benign and the other half are terrorist). The probability of a meeting between similar agents is $p = 0.8$.

Tracking Capabilities

Here we consider a little more general model of agents that are engaged in capability trades. Again, we consider a set of N agents, each belonging to one of two organizations, one terrorist and the other benign. The state of the system is defined by $\mathbf{x}_n = \{x_n^1, x_n^2, ..., x_n^N\}$. Initially, all our knowledge is represented by the prior distribution over the states, $\mathbf{P}_0(\mathbf{x}_0)$.

To trade capabilities, agents engage in meetings with other agents. For the sake of mathematical simplicity, we assume that the meetings for the capability trades are directed, in the sense that there is an *initiator* agent, and there is a *donor* agent. Thus, each observation is an ordered pair $\mathbf{y}_n = \{i \rightarrow j\}$. The initiator agent i is the one who is trying to acquire the capability, and the donor agent j is the agent from whom this capability might be acquired. We note that observing a meeting $\{i \rightarrow j\}$ does not necessarily mean that i acquired its intended capability from j: first of all, j might lack the required capability. And second, even if j possesses the capability, we will assume that there is a certain probability that the trade will fail. Thus, an agent who is after a certain capability might have to meet with more than one donor before he finally acquires the capability.

Next, we need to specify the rules according to which the meetings are generated. We assume that they are generated according to a certain probabilistic process. Namely, as in the GBA model from the previous section, we assume that the meetings are more likely to happen between agents from the same group. Furthermore, we assume that agents that intend to acquire a specific capability will tend to meet with agents that actually carry that capability. Specifically, at first the planner randomly chooses an *initiator* agent, i. Then the donor agent is chosen as follows:

1. With probability $p > 1/2$, it will belong to the same group as agent i. i.e., $\xi^j = \xi^i$.
2. If the initiator agent is not in search of a capability (e.g., either $\alpha_i = 0$, or $\theta_i = \alpha_i$), then the second agent will be chosen without any regard to his capability, e.g., each capability will be chosen with equal probability $p_0 = 1/(M+1)$. And if the initiator agent i is in search of a capability, e.g., $\theta_n^i \neq \alpha_n^i$, then with probability $p_1 > p_0$ the donor agent will be chosen among the agents that carry that capability.

Using the simple rules above, we can define the emission model $\Psi(\mathbf{y}_n = \{i \rightarrow j\}, \mathbf{x}_n)$, e.g., calculate the probability of seeing an observation $\{i \rightarrow j\}$ assuming that the system is in state $\mathbf{x}_n = \{x_n^1, x_n^2, ..., x_n^N\}$. Clearly, since the meeting involves two agents i and j, the probability of seeing such a meeting is affected by the states of i and j, e,g, $\Psi(\mathbf{y}_n = \{i \rightarrow j\}, \mathbf{x}_n) = \Psi(\mathbf{y}_n = \{i \rightarrow j\}, x_n^i, x_n^j)$. Furthermore, as we elaborated above, the probability of such a meeting depends on two independent factors: first, on the group membership, and the second, on the intentions and capabilities. Thus, we represent $\Psi(\mathbf{y}_n, x_n^i, x_n^j)$ a product of two functions:

$$\begin{aligned}\Psi(\mathbf{y}_n = \{i \rightarrow j\}, x_n^i, x_n^j) &= \\ \Psi_1(\mathbf{y}_n = \{i \rightarrow j\}, \xi_n^i, \xi_n^j) &\times \\ \Psi_2(\mathbf{y}_n = \{i \rightarrow j\}, \alpha_n^i, \theta_n^i, \theta_n^j)&. \end{aligned} \tag{11}$$

The first term of the product $\Psi_1(\mathbf{y}_n = \{i \rightarrow j\}, \xi_n^i, \xi_n^j)$ is the same as the emission model from the GBA example:

$$\begin{aligned}\Psi_1(\mathbf{y}_n = \{i \rightarrow j\}, \xi_n^i, \xi_n^j) = p\delta(\xi^i, \xi^j) + \\ (1-p)(1-\delta(\xi^i, \xi^j)).\end{aligned} \tag{12}$$

And the second term is the probability that the initiator agent i with intentions α^i and capability θ^i will meet a donor agent j with capability θ^j. According to the probabilistic rules specified above, this function can be written as follows:

$$\begin{aligned}\Psi_2(\mathbf{y}_n = \{i \rightarrow j\}, \alpha^i, \theta^i, \theta^j) &= \\ p_0 \times [\delta(\alpha^i, 0) + \delta(\alpha^i, \theta^i)]& \\ +[1 - \delta(\alpha^i, 0)][1 - \delta(\alpha^i, \theta^i)]& \\ \times[p_2\delta(\theta^j, \alpha^i) + p_0(1 - p_2)]&.\end{aligned} \tag{13}$$

For instance, the first term describes the case where the agent i has no intention of acquiring a capability (i.e, $\delta(\alpha^i, 0) = 1$), or it has already acquired the capability (i.e., $\delta(\alpha^i, \theta^i) = 1$). Then the emission function does not depend on the donor's capability θ_j, and evaluates to $\Psi_2(\mathbf{y}_n = \{i \rightarrow j\}, \alpha^i, \theta^i, \theta^j) = p_0$.

To complete the model definition, we must also specify the transition matrix. Again, for simplicity we assume that the group membership and intention to acquire a capability do not change, e.g., $\xi_n^i \equiv \xi^i$ and $\alpha_n^i \equiv \alpha^j$ are set in the beginning and never change. Thus, the only dynamic variables are θ_n^i, i.e., the actual capabilities carried by the agents. Furthermore, we note that only the initiator agent can change its state (i.e., by acquiring a capability), and the state of the donor agent remains unchanged. Thus, the transition probabilities are as follows:

$$\begin{aligned}M(\mathbf{x}_n, \mathbf{x}_{n-1}, \mathbf{y}) &\equiv \mathbf{P}(\mathbf{x}_n | \mathbf{x}_{n-1}, \mathbf{y}_{n-1} = \{i \rightarrow j\}) \\ &= \delta(x_n^j, x_{n-1}^j) \times \delta(\xi_{n-1}^i, \xi_n^i) \times \\ &\delta(\alpha_{n-1}^i, \alpha_n^i) \times T(\theta_n^i; \alpha^i, \theta_{n-1}^i, \theta_{n-1}^j),\end{aligned} \tag{14}$$

where $T(\theta_n^i; \alpha^i, \theta_{n-1}^i, \theta_{n-1}^j)$ is the probability that after the meeting $\{i \to j\}$, the agent i will have the capability θ_n^i, provided that at time $n-1$, i had the intention α^i and capability θ_{n-1}^i, and j had the capability θ_{n-1}^i. Let us determine this probability according to the following rules:

1. If the agent i has no intention of acquiring a specific capability ($\alpha^i, = 0$) or has already acquired the capability he intended ($\theta_{n-1}^i = \alpha^i$), then no trade happens, e.g., $\theta_n^i = \theta_{n-1}^i$.
2. If the agent i has not yet acquired its intended capability, i.e., $\theta_{n-1}^i \neq \alpha^i$ and $\alpha_i \neq 0$, then, if the second agent carries that capability, $\theta_{n-1}^j = \alpha^i$, then with probability γ he will acquire this capability. And if $\theta_{n-1}^j \neq \alpha^i$, (i.e., the donor agent j does not possess the required capability), then nothing changes. To simplify the notation, let us define $\tilde{\delta}(i,j) = 1 - \delta(i,j)$.

Putting all these together, we arrive at the following:

$$\begin{aligned} & T(\theta_n^i; \alpha^i, \theta_{n-1}^i, \theta_{n-1}^j) \\ &= [\delta(\alpha^i, 0) + \delta(\alpha^i, \theta_{n-1}^i)]\delta(\theta_n^i, \theta_{n-1}^i) \\ &+ \tilde{\delta}(\alpha^i, 0)\tilde{\delta}(\alpha^i, \theta_{n-1}^i)\tilde{\delta}(\alpha^i, \theta_{n-1}^j)\delta(\theta_n^i, \theta_{n-1}^i) \\ &+ \gamma\tilde{\delta}(\alpha^i, 0)\tilde{\delta}(\alpha^i, \theta_{n-1}^i)\delta(\alpha^i, \theta_{n-1}^j)\delta(\theta_n^i, \alpha^i) \\ &+ (1-\gamma)\tilde{\delta}(\alpha^i, 0)\tilde{\delta}(\alpha^i, \theta_{n-1}^i)\delta(\alpha^i, \theta_{n-1}^j)\delta(\theta_n^i, \theta_{n-1}^i). \end{aligned} \quad (15)$$

The first line on the RHS of Equation 15 corresponds to the case when the agent i either does not intend to acquire a capability, or has already acquired it. In this case i's capability does not change which is reflected in $\delta(\theta_n^i, \theta_{n-1}^i)$. The second line describes the case when i intends to acquire a capability and has not done so yet, but agent j does not carry the required capability. Again, the last term in the product ensures that the capability does not change. Further, the third and fourth lines describe the case when j actually carries the required capability – the third line describes a successful trade with probability γ, and the fourth line describes a failed trade with complementary probability $1 - \gamma$.

Equations 11–15 complete the definition of the simple capability tracking model. Using the distribution $\mathbf{P}_n(\mathbf{X}_n)$, one can get probabilistic answers to various questions of interest. For instance, to find out whether the organization is trying to acquire a set of capabilities $\{C1, C2\}$, let us define an indicator function $I(i,j)$, which is equal to 1 if the agents i and j are terrorists and have the intention to acquire capabilities $C1$ and $C2$ between them, and 0 otherwise. Using Kronecker's notation again, this function is represented as $I(i,j) = \delta(\xi^i, 1)\delta(\xi^j, 1)[\delta(\alpha^i, C1)\delta_(\alpha^j, C2) + \delta(\alpha^i, C2)\delta(\alpha^j, C21)]$. Using this indicator function and the filtering distribution $\mathbf{P}_n(\mathbf{X}_n)$, we can estimate the likelihood of different hypotheses. For instance, our belief that the organization is indeed engaged in a mission to acquire the capacities can be calculated as follows: $\mathbf{P}_n(\text{mission=true}) = 1 - \sum_{\mathbf{X}} \mathbf{P}_n(\mathbf{X}) \prod_{i,j}[1 - I(i,j)]$. Similarly, one can define a different indicator function over the actual carried capabilities to estimate the probability that the set (or a subset) of the capabilities have already been acquired. Again, we note that exact tracking becomes infeasible for sufficiently large systems because of the exponential explosion in the size of the state space. We are currently examining different approximation techniques, such as using factored representations of the belief state, that will allow us to consider a large number of agents.

Related Work

In this section we review existing probabilistic models for plan recognition. Probabilistic Hostile Agent Task Tracker (PHATT) (Geib, C.W. & Harp, S.A. 2004; Geib, C.W. & Goldman, R.P. 2005) is a hybrid symbolic–probabilistic plan recognizer, where the symbolic approach is used to filter out inconsistent hypotheses. The remaining consistent hypotheses are then evaluated probabilistically. PHATT assumes that initially the executing agent has a set of goals and chooses a set of plans to achieve these goals. The set of plans chosen determines a set of pending primitive actions. The observations of the agent's actions are taken as an execution trace and the conditional probability of each of the root goals are determined given the observed series of actions. Another hybrid symbolic–probabilistic model developed in (Avrahami-Zilberbrand, D. & Kaminka, G.A. 2006) combines a symbolic plan recognizer with a probabilistic inference framework based on *Hierarchical Hidden Markov Model* (HHMM). This approach computes the likelihood of the consistent hypotheses obtained from the symbolic recognizer, ranks them, and then determines the hypothesis with the maximum posterior probability (called the *probable current state hypothesis*). It can efficiently deal with a richer class of plan recognition challenges, such as, recognition based on duration of behaviors, and recognition of interleaved plans (where an agent interrupts a plan for another, only to return to the first one later).

Much recent work has considered using various generalizations of Hidden Markov Models for plan recognition. The Cascading Hidden Markov Model (CHMM) (Blaylock, N. & Allen, J. 2006) uses an extension of the traditional HMM to simultaneously recognize an agent's current goal schemas at various levels of a hierarchical plan. Bui et al proposed a probabilistic plan recognition framework based on Abstract Hidden Markov Models (AHMM) (Bui, Venkatesh, & West 2002). This approach allows to identify an agent's behavior in dynamic, noisy, uncertain domains and across multiple levels of abstraction. They also describe an approximate inference technique based on Rao-Blackwellized Particle Filter (RBPF) that allows an efficient, hybrid inference method for this model, that scales well with the number of levels in the plan hierarchy. The AHMM has a *memoryless* property, that is, the policies in the AHMM cannot represent a sequence of uninterrupted sub-plans and thus the decision to choose the next sub-plan is only dependent on the current state, and not on the sub-plans that were chosen in the past. This issue has been addressed by extending the AHMM model to include a memory flag (Bui, H.H. 2003).

Probabilistic State-Dependent Grammars (PSDG), suggested in (Pynadath, D.V. & Wellman, M.P. 2000) are used to represent an agent's plan generation process and are closely related to the AHMM approach. The PSDG can be described as the Probabilistic Context Free Grammar (PCFG) augmented with a state space, and a state transition probability table for each terminal symbol of the PCFG. They use an exact inference method to deal with the case where the states are fully observable. When the states are partially observable, a brute-force approach is suggested which increases the complexity of the method.

YOYO (Kaminka, G.A., Pynadath, D.V., & Tambe, M. 2002)) is an efficient probabilistic plan recognition algorithm to monitor the state of a team of agents from the members' routine conversations (hence *overhearing*) exchanged as part of their coordinated task execution. YOYO exploits knowledge about the team's social behavior to predict future observations during execution (thereby reducing monitoring uncertainty). The approach is scalable, and can handle missing observations. One of the drawbacks is that messages are assumed to be truthful, hence there is no way of detecting deception here which may be an issue in practice. Also, the method is based on the implicit assumption that the team, its members and the social structure among them are known beforehand.

As mentioned earlier also, the primary drawback of many of these methods is that they assume a known identity of the agent that one wants to track, and that there is only a moderately large number of potential plans. This is clearly not practical in realistic intelligence analysis applications where one usually deals with a huge number of agents with a large library of plans at their disposal, from which one wants to find the few that may be of interest in some sense. Such a scenario thus involves a large network of hypotheses that requires efficient hypothesis management tools with the goal of determining the relevant ones. An example of such a system is provided by the Hats world. The latter is a simple prototype for terrorist networks that are operational in the real world, and involve a huge amount of transactional data in the form of phone conversations, meetings and so on between many agents, a few of whom are expected to carry out a harmful mission. Moreover, very few plan recognition systems deal with team behavior based on coordination among the members. Therefore, none of the existing techniques are capable of providing a rigorous tracking and detection algorithm for handling plans of such complex nature as those involved in a domain like Hats, and our method is a first effort in that direction that has the potential to overcome the shortcomings of most of the current approaches .

Discussion and Future Work

In this paper we have formulated a problem of detecting and tracking hostile activities in a virtual Hats domain. Despite the over–simplified models of agent behavior, the Hats domain still attains the most important characteristics of real world intelligence analysis problems: low signal to noise ratio, and huge number of possible hypotheses one must manage. Thus, we believe that developing successful detection and tracking techniques for Hats will certainly be valuable for more realistic problems as well. The scenario considered here is a *keyhole* plan recognition, e.g., it is assumed that agents who are being planned are not aware of it, or do not pay any attention. We note, however, that in more interesting adversarial scenarios, agents might utilize deception, e.g., by demonstrating a certain behavior temporarily to cover their true intent. This type of deliberately deceptive behavior goes beyond the scope of this paper, and has not ben covered here.

We have also described our initial steps for building a probabilistic framework for detecting and tracking hostile activities in Hats. Specifically, we described a recursive equation for updating beliefs (i.e., filtering distribution) once new observations are made available. Clearly, tracking the joint filtering distribution is computationally infeasible when the number of agents is sufficiently large. One of the ways for reducing computational complexity of the tracking problem is to group variables (e.g., agents) so that variables across the groups have little or no correlation. One can then approximate the joint distribution function by a product of factored distributions over those groups (Boyen & Koller 1998; Murphy & Weiss 2001). For instance, if one knew the members of the organizations in advance, we could leave out other agents, thus significantly reducing the problem dimensionality. Moreover, grouping members of organization together has another advantage. Namely, if all the members of the organization are known in advance, then the observations about individual members can be aggregated to make inferences about the organization. In other words, if the tracking has revealed that a certain member of the organization intends to acquire a certain capability, then this indicates that the organization itself intends to get that capability. This suggests that one can try to find organizations based on transaction data, and for detecting groups of strongly connected (or highly correlated) nodes in transactional networks. Algorithms for detecting groups of strongly connected (or highly correlated) nodes in transactional networks have been proposed in (Newman & Girvan 2004; Danon *et al.* 2005; Galstyan & Cohen 2005; 2006).

Finally, we would like to note that the emission and the transition probabilities in our model consist of parameters that are, generally speaking, unknown in advance. Instead, these parameters have to be estimated using an Expectation–Maximization (EM) approach, where one starts with some initial values for each of the unknown parameters (based on prior knowledge) and continue the re-estimation process until it converges to the particular values that maximizes the likelihood function, thud yielding the Maximum Likelihood Estimate (MLE) of the model parameters. We intend to address this learning problem in our future work.

Acknowledgment

This research is supported in part by an ARO MURI grant 50363-MA-MUR.

References

Avrahami-Zilberbrand, D., and Kaminka, G.A. 2006. Hybrid symbolic-probabilistic plan recognizer: Initial steps. In *Proceedings of AAAI workshop on modeling others from observations (MOO-06).*

Blaylock, N., and Allen, J. 2006. Fast hierarchical goal schema recognition. In *Proceedings of AAAI-06*.

Boyen, X., and Koller, D. 1998. Approximate learning of dynamic models. In *Proceedings of NIPS-11, 1998.*

Bui, H.H. 2003. A general model for online probabilistic plan recognition. In *Proceedings of IJCAI'03*.

Bui, H.; Venkatesh, S.; and West, G. 2002. Policy recognition in the abstract hidden markov models. *Journal of Artificial Intelligence Research* 17:451–499.

Cohen, P.R., and Morrison, C.T. 2004. The hats simulator. In *Proceedings of the 2004 Winter Simulation Conference*, 849–856.

Danon, L.; Díaz-Guilera, A.; Duch, J.; and Arenas, A. 2005. Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment* 2005(09):P09008.

Doucet, A.; de Freitas, N.; and Gordon, N. Eds. 2001. *Sequential Monte Carol Methods in Practice*. Springer.

Galstyan, A., and Cohen, P. R. 2005. Inferring useful heuristics from the dynamics of iterative relational classifiers. In *Proceedings of IJCAI-05, 19th International Joint Conference on Artificial Intelligence*.

Galstyan, A., and Cohen, P. R. 2006. Relational classification through three–state epidemic dynamics. In *Submitted to The 9th International Conference on Information Fusion, Florence, Italy.*

Geib, C.W., and Goldman, R.P. 2005. Partial observability and probabilistic plan/goal recognition. In *Proceedings of the International workshop on modeling other agents from observations (MOO-05).*

Geib, C.W., and Harp, S.A. 2004. Empirical analysis of a probabilistic task tracking algorithm. In *Proceedings of Workshop on Agent Tracking, Autonomous Agents and MultiAgent Systems (AAMAS).*

Kaminka, G.A.; Pynadath, D.V.; and Tambe, M. 2002. Monitoring teams by overhearing: A Multi-agent plan recognition approach. *Journal of Artificial Intelligence Research* 17:83–135.

Morrison, C.T.; Cohen, P.R.; King, G.W.; Moody, J. J.; and Hannon, A. 2005. Simulating terrorist threat with the hats simulator. In *Proceedings of the First International Conference on Intelligence Analysis*.

Murphy, K., and Weiss, Y. 2001. The factored frontier algorithm for approximate inference in DBNs. In *UAI–01*.

Newman, M. E. J., and Girvan, M. 2004. Finding and evaluating community structure in networks. *Phys. Rev. E* 69:026113.

Pynadath, D.V., and Wellman, M.P. 2000. Probabilistic state-dependent grammars for plan recognition. In *Proceedings of Sixteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI'00).*