A Generative Game-Theoretic Framework for Adversarial Plan Recognition

Nicolas Le Guillarme, Abdel-Illah Mouaddib, Xavier Lerouvreur, Sylvain Gatepaille

A Generative Game-Theoretic Framework for Adversarial Plan Recognition

Nicolas Le Guillarme and **Abdel-Illah Mouaddib**
GREYC (UMR 6072)
University of Caen Basse-Normandie
Caen, France
{nicolas.leguillarme,abdel-illah.mouaddib}@unicaen.fr

Xavier Lerouvreur
Signal Processing, Algorithms & Performance
Airbus Defence and Space
Toulouse, France
xavier.lerouvreur@astrium.eads.net

Sylvain Gatepaille
Advanced Information Processing
Airbus Defence and Space
Val-de-Reuil, France
sylvain.gatepaille@cassidian.com

Abstract

Adversarial reasoning is of the first importance for defence and security applications since it allows to (1) better anticipate future threats, and (2) be proactive in deploying effective responses. In this paper, we address the two subtasks of adversarial reasoning, namely adversarial plan recognition and strategy formulation, from a generative, game-theoretic perspective. First, a set of possible future situations is computed using a contextual action model. This projected situation serves as a basis for building a set of Markov games modeling the planning strategies of both the defender and his adversary. Finally, a library of critical plans for the attacker and a library of best responses for the defender are generated automatically by computing a Nash equilibrium in each game. The adversarial plan recognition task therefore consists of inferring a probability distribution over the set of possible plans of the adversary, while the strategy formulation problem reduces to the selection of the most appropriate response. Initial results on a urban warfare scenario suggest that our framework can be useful to model complex strategic interactions inherent to plan recognition in adversarial situations.

Introduction

Defence and security activities, such as military operations or cybersecurity to name a few, are adversarial by nature. In such situations, where a decision-maker (”the defender”) is threatened by one or several intelligent opponents with conflicting goals (”the attackers”), a key enabler for the success of operations is the ability for the defender to ”read the opponent’s mind” (Kott and McEneaney 2006). Adversarial reasoning generally consists of inferring the intentions of an adversary from the available evidence so as to make possible the prediction of his future actions, which then enables the planning of an effective response. Adversarial reasoning can therefore be divided into two main subtasks, namely *adversarial plan recognition*, or the identification of an adversary’s plans and goals on the basis of observations of his actions, and *strategy formulation*, where the defender has to formulate his own plan of action taking into account possible counteractions of the adversary.

Compared to other types of plan recognition, namely intended plan recognition, where the observed agent is cooperative, and keyhole plan recognition, where the observed agent is not aware or indifferent to the recognition process (Geib and Goldman 2001), adversarial plan recognition is characterized by the fact that the observer (defender) and the observed agent (attacker) evolve in the same environment and that the observed agent is at least not cooperative, if not actively hostile to the inference of his plans. These characteristics of adversarial plan recognition result in two types of adversarial interactions between the defender and the attacker. First, since both agents evolve in the same environment and pursue conflicting goals, each agent will have to reason about the planning strategy of his opponent to plan his actions, keeping in mind that his adversary will do the same. Secondly, should the observed agent be actively hostile to the recognition process, he will plan his actions accordingly so as to minimize the probability for his plans to be recognized (e.g. using concealment and/or deceptive actions), while the plan recognition process will have to reason about the adversary’s planning strategy to predict his action.

Game-theory is a natural and popular formalism to address problems involving interactions between agents ranging from total cooperation to full hostility (Burkov and Chaib-draa 2008). In (Lisỳ et al. 2012), the problem of adversarial plan recognition is defined as an imperfect information two-player zero-sum game in extensive form between an actor and an observer. From a game-theoretic perspective, the solution is the strategies for both players that optimize the tradeoff between plan utility and plan detectability. The drawbacks of this method are that a plan library must be provided explicitly, the model does not handle simultaneous decisions, and the observer can only perform activity recognition actions. In (Braynov 2006), the interaction between adversarial planning and adversarial plan recognition is modeled as a two-player zero-sum game over an attack graph representing the possible plans of the attacker. This approach addresses both types of strategic interactions between the observer and the observed agent. However, it relies on a plan recognition algorithm as a plugin. Closest to the present paper is (Chen et al. 2007) where a set of optimal courses of action (COAs) for the enemy and friendly forces is generated using a decentralized Markov game. The state space is defined as the set of all the possible COAs for enemy, friendly, and neutral forces. The effectiveness of the approach has been demonstrated through combat simulation, but the authors are not explicit about the way their method can be used for goal/plan recognition.

In this work, we propose a generative game-theoretic framework to tackle the problem of adversarial plan recognition in a fully observable, multi-agent setting where actions have stochastic effects. The adversarial plan recognition task therefore consists of inferring a probability distribution over the set of possible plans of an agent whose behavior results from a finite two-player zero-sum stochastic game, where the players are respectively the attacker/observed agent and the defender/observer. The problem of strategy formulation then reduces to the selection of the most likely best response. Stochastic games are used to model the strategic interactions between the defender and the attacker planning strategies due to the common environment and their conflicting goals. The use of deception and concealment by the attacker is out of the scope of this paper and is left for future work. Since generative plan recognition techniques require the definition of an action model to encode the possible behaviors of the observed agent, we define a contextual action model based on the COI model of intentional action (Steinberg 2007) which enables the automatic assessment of all opportunities for action available to each player in the current situation.

The rest of this paper is organized as follows. In Section 2, we provide a brief background on generative plan recognition and stochastic game theory. Section 3 contains an extensive description of PRESAGE, our generative game-theoretic framework for adversarial plan recognition. Section 4 applies this to a urban warfare scenario and presents preliminary experimental results. In Section 5 we discuss the current progress of our work and the planned extensions.

Background

Generative Plan Recognition

Most of previous research in plan recognition relies on an a priori, handcrafted plan library specifying all the possible plans of the observed agent (Avrahami-Zilberbrand and Kaminka 2007; Geib and Goldman 2009; Lisỳ et al. 2012). This library is generally assumed to be exhaustive, which is quite impractical in real-world applications. Recently, a new paradigm known as *generative plan recognition* (also called *model-based* plan recognition or plan recognition *by inverse planning*) has been proposed by Baker et al. (Baker, Saxe, and Tenenbaum 2009) and further studied in a series of work by Ramírez and Geffner (Ramırez and Geffner 2009; 2010; 2011). In plan recognition by inverse planning, observed agents are assumed to be rational, i.e. they will plan optimally to achieve their goals. The first advantage of generative methods is that the need for an explicit declaration of the possible agent behaviors as a plan library is replaced by an implicit encoding, using an agent action model and a set of possible goals, thus making generative methods far more flexible and less dependent on the availability of expert knowledge. The second advantage is that the set of optimal plans can be generated automatically using state-of-the-art planners, including classical planners (Ramırez and Geffner 2009; 2010), Markov Decision Processes (MDPs) (Baker, Saxe, and Tenenbaum 2009), and partially-observable MDPs (Ramırez and Geffner 2011).

Stochastic Games

Definition Stochastic games - also called Markov games - have been introduced by Lloyd Shapley (Shapley 1953) as an extension of MDPs to the multi-agent case. In a stochastic game, the players perform joint actions that determine both the new state of the environment, according to transition probabilities, and the reward obtained by each agent. Formally, a stochastic game is defined as a tuple $\langle Ag, S, \mathbf{A}, \{R^i, i = 1, ..., |Ag|\}, T\rangle$ where

- $Ag = \{1, ..., |Ag|\}$ is a finite set of players.
- S is a finite set of states.
- $A_i = \{a^i_1, ..., a^i_{|A_i|}\}$ is a finite set of actions available to player i.
- $\mathbf{A} \equiv \times_{i\in Ag} A_i$ is the set of joint actions
- $R^i : S \times \mathbf{A} \to \mathbb{R}$ is the payoff function of player i.
- $T : S \times \mathbf{A} \times S \to [0,1]$ is the transition function.

The game is played in discrete time steps. In each time step t, the players choose their actions simultaneously and the joint action $\mathbf{a} = \{a^1, ..., a^{|Ag|}\}$ is obtained. Each agent i receives a reward $R^i(s_t, \mathbf{a})$ depending on the current state of the environment and the joint action, and the players are transferred to the next state s_{t+1} according to the transition function T. A policy $\pi_i : S \to \Delta A_i$ for agent i defines for each state of the game a local, mixed strategy, i.e. a probability distribution over the set of available actions.

Solution concepts Given a joint strategy $\pi = \langle \pi_i, \pi_{-i}\rangle$ where π_{-i} is the joint strategy of all players except player i, the expected utility of π for player i is defined in each $s \in S$ as the expected value of the utility function of normal form games (Burkov and Chaib-draa 2008)

$$u_i^\pi(s) = E_{\mathbf{a}\in\mathbf{A}}[R^i(s, \mathbf{a})] \tag{1}$$

The state utilities $U_i^\pi(s)$ for player i associated with joint policy π can be quantified using the same performance criteria than for MDPs, i.e. the long-term expected value over i's reward. For instance, using the γ-discounted criterion

$$\begin{aligned} U_i^\pi(s) &= E\,[\textstyle\sum_{t=0}^{\infty} \gamma^t u_i^\pi(s_t) | s_0 = s] \\ &= u_i^\pi(s) + \gamma \textstyle\sum_{\mathbf{a}\in\mathbf{A}} \sum_{s'\in S} T(s, \mathbf{a}, s') \pi^{\mathbf{a}}(s) U_i^\pi(s'), \end{aligned} \tag{2}$$

where $\pi^{\mathbf{a}}(s)$ is the probability of joint action $\mathbf{a}$ in s given joint policy π, s_0 is the initial state of the game and $\gamma \in [0,1]$ is the discount factor. For other performance criterion, see (Garcia and Rachelson 2008).

The concept that is most commonly used as a solution of non-cooperative stochastic games is the one of Nash equilibrium, i.e. a combination of strategies where each player selects the best response to the other players' strategies, and no player can improve its utility by unilaterally deviating from this strategy. Formally, a joint strategy $\pi^* = \langle \pi_i^*, \pi_{-i}^*\rangle$ is a Nash equilibrium if

$$\forall s \in S,\ \forall i \in Ag,\ \forall \pi_i \in \Pi_i, \qquad U_i^{\langle \pi_i^*, \pi_{-i}^*\rangle}(s) \geq U_i^{\langle \pi_i, \pi_{-i}^*\rangle}(s), \tag{3}$$

where Π_i is the set of policies available to agent i. The existence of at least one Nash equilibrium for 2-player stochastic games has been demonstrated by Shapley (Shapley 1953).

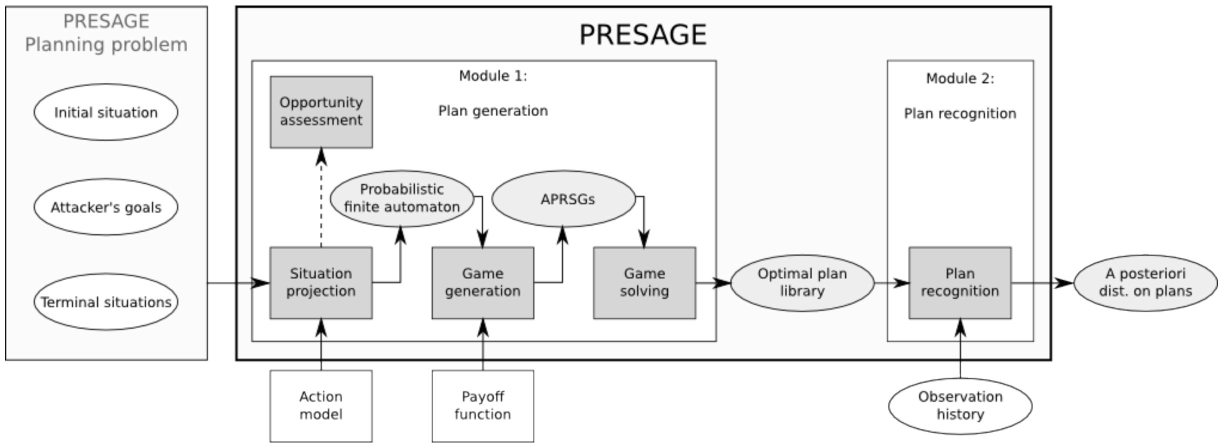

Figure 1: PRESAGE general architecture.

Plan Recognition using Stochastic Games

In this section, we describe PRESAGE (Plan REcognition using StochAstic GamEs), a generative game-theoretic framework for adversarial plan recognition. This framework models the strategic interactions between the decision-making strategies of an attacker (observed agent) and a defender (observer) as Markov games. As any other generative method, our framework requires the definition of an action model. We propose an adaptation of the COI model of action (Steinberg 2007) to the domain of automated planning so that the opportunities (equivalent to MDPs' actions) available to each player can be automatically assessed from the situation and used to generate valid plans. Finally, we design a probabilistic plan recognition method whose aim is to create and maintain a belief state over the set of optimal strategies of the attacker from observations of his behavior.

General Architecture

Figure 1 is a representation of PRESAGE functional architecture. PRESAGE is divided into two independent modules. The first module (Module 1) is dedicated to the automatic generation of the optimal plan library. Starting from the definition of a planning problem P, including the initial situation and the set of possible goals of the attacker, the plan generation module executes a three-step procedure to build the library of possible plans for the attacker:

1. **Situation projection:** first, the set of possible future situations is generated and modeled as a *probabilistic finite automaton* Σ (Stoelinga 2002), whose transitions are labeled by joint opportunities of the attacker and the defender. Opportunities for action available to each actor are assessed in each state of Σ using a generic opportunity assessment engine.
2. **Game generation:** for each possible goal g of the attacker, we build an Adversarial Plan Recognition Stochastic Game (APRSG) Γ_g by considering each state of Σ where g is achieved as a terminal state, and by representing each remaining state as a two-player zero-sum static game using a predefined payoff function.
3. **Game solving:** each APRSG is solved independently of the others using off-the-shelf solving algorithms so as to obtain an optimal plan library Π_P^* containing one optimal strategy for each possible goal of the attacker.

The second module (Module 2) encapsulates a plan recognition engine which, given a plan library and an observation history of both actors' actions, returns an a posteriori distribution over the set of possible plans of the attacker.

Assessing Opportunities for Action

Our action model is an adaptation of the COI model of threat to the planning domain. This model was first proposed in (Steinberg 2005; Little and Rogova 2006) and has been further extended in (Steinberg 2007) to represent any intentional action. It considers the *capability*, *opportunity*, and *intent* of an agent to carry out actions on some targets to be the necessary and sufficient factors to characterize these actions:

Capability the possession of the resources and/or skills required to perform an action.

Opportunity "the right context" to perform an action, i.e. the presence of an operating environment in which potential targets are susceptible to be acted upon.

Intent the plans and goals an agent wants to achieve.

These action components are tightly interrelated. For instance, the intent of an agent can evolve in light of his capabilities and current opportunities, e.g. by changing his target. Inversely, the agent's intent can motivate the acquisition of new capabilities/opportunities, etc. Since our goal is to infer the intent of an agent from observations of his actions as restrained by its capabilities and opportunities, intent is implicit in our model and we postulate that it is constrained by the rationality principle. We also assume that we have full and certain knowledge of the capabilities of each agent. We therefore focus on the opportunity assessment task which consists of determining which action(s) can be performed on which target(s) in the current situation and estimating the expected outcome of these actions. To understand the concept of opportunity assessment, let us consider the example depicted on Figure 2.

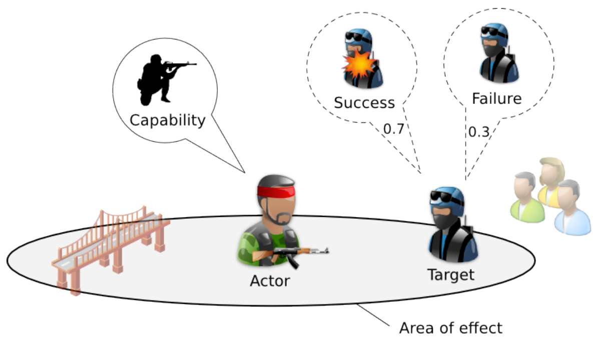

Figure 2: An example of opportunity assessment.

In this situation, the actor owns a resource *assault rifle* which provides him with the capability to perform action *shooting*. However, if the agent wants this action to have an effect on a target t, he must first acquire the opportunity to act upon t. Given a situation s, we will consider that an agent has the opportunity to perform action a on a target t if:

- the agent has the capability to perform action a in s
- the agent has access to target t in s,
- target t is vulnerable to action a in s.

We propose a simple, yet expressive opportunity model that takes into account the three aforementioned conditions and associates probabilities to the possible consequences of an action, namely success or failure. Enhanced action definitions may include more than two outcomes.

Definition 1. *An opportunity o^i for actor i is a tuple $\langle a,t,p\rangle$, where a is an action, t is the target of the action, and p is the probability of success of action a when performed on t.*

Given a situation s, the goal of the opportunity assessment engine is to compute for each actor i, the set $O_i(s)$ of opportunities available to i in s. In our model, each capability (feasible action) is associated with one area of effect (AoE).

Definition 2. *The area of effect associated with a capability a is a tuple $AoE(a) = \langle \mathscr{P}, P_a(success)\rangle$ where $\mathscr{P}$ is a set of preconditions necessary for a target t to be considered accessible (e.g. lying within a spatial area), and $P_a(success)$ is the probability of success of action a.*

Definition 3. *Assuming that actor i has the capability to perform action a, there exists an opportunity o^i for i to perform action a on a target t iff*

1. *t satisfies the accessibility conditions listed in $\mathscr{P}$,*
2. *the probability of success $p > 0$.*

p is determined by both the probability of success of the action $P_a(success)$ and the vulnerability of the target t to action a when t satisfies the accessibility conditions (written $t \in AoE(a)$).

$$\begin{aligned} p &= P(success(a), t \in AoE(a), vulnerable(t,a)) \\ & P_a(success)P(vulnerable(t,a)|success(a), t \in AoE(a)) \end{aligned} \tag{4}$$

The vulnerability of a target to an action may be seen as its ability to thwart some possible effects of this action. For instance, considering the two possible effects of the *shooting* action to be *target killed* (success) and *target missed* (failure), we can assume that a target wearing a bulletproof vest will be less vulnerable to this action, i.e. the probability to obtain effect *target killed* on this particular target will be smaller than with an unprotected target.

Definition 4. *The vulnerability of a target t to an action a is the conditional probability $V_t(a)$, where $V_t(a) = 0$ means that t is not vulnerable to action a and $V_t(a) = 1$ represents the highest level of vulnerability.*

The overall probability of success p is therefore given by

$$p = P_a(success) \times V_t(a) \tag{5}$$

Example 1. *In the situation depicted in Figure 2, the only opportunity for the attacker is to attack the vulnerable ($V_{blueforce}(shooting) = 1.0$) blue force with a probability of success $P_{shooting}(success) = 0.7$; the civilian group cannot be acted upon because it does not lie within the (spatial) AoE, and the attacker does not possess the capability bomb-bridge required to act upon the bridge. Therefore, we have $O_i(s) = \{\langle shooting, blueforce, 0.7\rangle\}$.*

PRESAGE Planning Problem

A PRESAGE planning problem is defined as a tuple $P = \langle Ag, I, G, \mathscr{T}\rangle$, where Ag is a finite set of actors, I is the initial situation, G is the set of goals of the attacker, and $\mathscr{T}$ is a set of terminal situations. Elements of G are not necessarily elements of $\mathscr{T}$, thus offering the possibility to account for an adversary trying to achieve several goals sequentially.

Situation Projection

The goal of the situation projection engine is, given the definition of a PRESAGE planning problem, to generate a probabilistic finite automaton Σ which aims at formally representing the possible future situations and their dynamics. Σ is defined as a tuple $\langle s_0, S, S^{\mathscr{T}}, \mathbf{O}, T\rangle$. The definitions for the different parameters are compiled in Table 1.

Par.	Description	Expression
S	A finite set of states	$S = \{s_0, ..., s_{\lvert S\rvert}\}$
s_0	The initial state	$s_0 \in S$
$S^{\mathscr{T}}$	A finite set of terminal states	$S^{\mathscr{T}} \subset S$
O_i	A mapping assigning to each $s \in S \setminus S^{\mathscr{T}}$ the set of opportunities for player i in s	$O_i(s) = \{o^i_1, ..., o^i_n\}$ with $o^i_j = (a^i_j, t^i_j, p^i_j)$
O(s)	The set of possible joint opportunities in state $s \in S \setminus S^{\mathscr{T}}$	$\mathbf{O(s)} \equiv \times_{i \in Ag} O_i(s)$
SO	The set of all possible joint opportunity profiles	$SO = \{(s, \mathbf{o}) \mid s \in S \setminus S^{\mathscr{T}}, \mathbf{o} \in \mathbf{O(s)}\}$
T	A transition function	$T : SO \times S \rightarrow [0,1]$

Table 1: Definitions for the parameters of Σ

Given a PRESAGE planning problem P and an opportunity assessment engine `OA` with $\forall s \in S \setminus S^{\mathscr{T}}, i \in Ag$,

$\texttt{OA}(s,i) \rightarrow O_i(s)$, we build Σ using the situation projection algorithm shown in Algorithm 1. The algorithm first checks the terminal conditions (lines 5-6). If the current state s is not terminal, opportunities available to each player in s are assessed using the opportunity assessment engine so as to build the set of possible joint opportunities $\mathbf{O}(\mathbf{s})$ (line 8-10). Line 11 loops over all possible joint opportunities $\mathbf{o}$ in $\mathbf{O}(\mathbf{s})$. Given a state s and a joint opportunity $\mathbf{o}$, the `findSuccessors` function (whose algorithm is not detailed here due to a lack of space) computes the set of successor states S' obtained when executing joint action $\mathbf{o}$ in s, as well as the transition probability $P(s'|s,\mathbf{o})$ for each $s' \in S'$ (line 12). Each successor state s' in S', if not already processed, is added to S and the corresponding transition probability is used to update the transition function T (lines 13-16). The recursive situation projection procedure is applied to each state until no new state can be added to S (line 17).

Algorithm 1: Situation projection algorithm

```
   Data: $P = \langle Ag, I, G, \mathscr{T} \rangle$, OA
   Result: $\Sigma$
1  begin
2      $s_0 \leftarrow I$, $S \leftarrow \{s_0\}$, $S^{\mathscr{T}} \leftarrow \{\}$, $T \leftarrow \{\}$
3      project($P$, $s_0$, $S$, $S^{\mathscr{T}}$, $T$)
4  procedure project($P$, $s$, $S$, $S^{\mathscr{T}}$, $T$)
5      if $s \in \mathscr{T}$ then
6          $S^{\mathscr{T}} \leftarrow S^{\mathscr{T}} \cup \{s\}$
7      else
8          foreach $i \in Ag$ do
9              $O_i(s) \leftarrow$ OA($s,i$)
10         $\mathbf{O}(\mathbf{s}) \leftarrow \times_{i \in Ag} O_i(s)$
11         foreach $\mathbf{o} \in \mathbf{O}(\mathbf{s})$ do
12             $S' \leftarrow$ findSuccessors($s,\mathbf{o}$)
13             foreach $s' \in S'$ such as $P(s'|s,\mathbf{o}) \neq 0$ do
14                 $T(s,\mathbf{o},s') \leftarrow P(s'|s,\mathbf{o})$
15                 if $s' \notin S$ then
16                     $S \leftarrow S \cup \{s'\}$
17                     project($Ag$, $P$, $s'$, $S$, $S^{\mathscr{T}}$, $T$)
```

Stochastic Games Generation and Solving

The projected situation Σ is used as a basis for building one Adversarial Plan Recognition Stochastic Game (APRSG) planner per possible goal of the attacker. An APRSG for goal $g \in G$ is a finite two-player zero-sum stochastic game which is formally defined as a tuple $\Gamma_g = \langle \Sigma_g, \{R_g^i, i = 1, ..., |Ag|\}\rangle$ where $\Sigma_g = \langle Ag, S_g, S_g^{\mathscr{T}}, \mathbf{O_g}, T_g \rangle$ and $R_g^i : SO_g \rightarrow \mathbb{R}$ is the payoff function of player i. The only addition to the classical definition of stochastic games (as discussed above) is the set $S_g^{\mathscr{T}} \subset S_g$ of terminal states for goal g.

Players — An APRSG is a two-player game between an attacker (player 1, the observed agent), which tries to maximize his payoff, and a defender (player 2, the observer), which aims at minimizing player 1's payoff. Therefore we have $Ag = \{1,2\}$.

State space and transition function — The game generation procedure is shown in Algorithm 2. First, S_g and $S_g^{\mathscr{T}}$ are initialized as copies of their respective equivalent set in Σ. Then, each state s in S_g corresponding to a situation where goal g is achieved is marked as a terminal state of game Γ_g (lines 3-5). The `pruneDisconnectedStates` function (line 6) acts by removing all the edges originating from terminal states ($\forall s \in S_g^{\mathscr{T}}, \forall \mathbf{o} \in \mathbf{O_g}(\mathbf{s}), \forall s' \in S_g, T(s,\mathbf{o},s') \leftarrow 0$). States in S_g and $S_g^{\mathscr{T}}$, which are no more connected to the initial state s_0 due to the fact that goal states are now considered as terminal, are pruned using depth-first-search algorithm. The state space S_g of game Γ_g is therefore at most as large as the initial state space S. Finally, T_g and $\mathbf{O_g}(\mathbf{s})$ are respectively defined as the restriction of T and $\mathbf{O}(\mathbf{s})$ to the set $S_g \setminus S_g^{\mathscr{T}}$ (lines 7-8). This algorithm is applied for each attacker's goal $g \in G$. A simple example illustrating this game generation procedure is depicted in Figure 3.

Algorithm 2: Game generation algorithm

```
   Data: $\Sigma$, $g$, $\{R_g^i, i = 1, ..., |Ag|\}$
   Result: $\Gamma_g$
1  begin
2      $S_g \leftarrow copy(S)$, $S_g^{\mathscr{T}} \leftarrow copy(S^{\mathscr{T}})$
3      foreach $s \in S_g$ do
4          if $g$ is achieved in $s$ then
5              $S_g^{\mathscr{T}} \leftarrow S_g^{\mathscr{T}} \cup \{s\}$
6      pruneDisconnectedStates($s_0, S_g, S_g^{\mathscr{T}}, T$)
7      $T_g \leftarrow T_{|S_g \setminus S_g^{\mathscr{T}}}$
8      $\mathbf{O_g}(\mathbf{s}) \leftarrow \mathbf{O}_{|S_g \setminus S_g^{\mathscr{T}}}(\mathbf{s})$
9      $\Sigma_g \leftarrow \langle Ag, S_g, S_g^{\mathscr{T}}, \mathbf{O_g}, T_g \rangle$
10     $\Gamma_g \leftarrow \langle \Sigma_g, \{R_g^i, i = 1, ..., |Ag|\}\rangle$
```

Payoffs and optimal strategy — The payoffs for player i playing game Γ_g are given by the application-dependent payoff function R_g^i which associates to each joint opportunity and in every state of the game, a gain in terms of utility for player i. Each state of an APRSG can be seen locally as a two-player zero-sum static game. Consequently, the payoff function must satisfy the following constraint: $\forall s \in S_g, \forall \mathbf{o} \in \mathbf{O_g}(\mathbf{s}),\ \sum_{i \in Ag} R_g^i(s,\mathbf{o}) = 0$. A two-player zero-sum static game can be solved using the Minimax algorithm (Neumann 1928). Assuming that player 1 wants to maximize its payoff while player 2 wants to minimize player 1's payoff, the optimal strategy for player 1 (resp. player 2) is called the *maximin* (resp. *minimax*) strategy, and an equilibrium is reached if *maximin* = *minimax*. Such an equilibrium always exists in mixed strategy. Let $\mathbf{R_g}(\mathbf{s})$ be a $m \times n$ matrix where m (resp. n) is the number of opportunities available to player 1 (resp. player 2) in s, and $\mathbf{R_g}(\mathbf{s})_{i,j} = R_g^1(s, o_i^1, o_j^2)$, the mixed maximin strategy $x^* = (x_1^*, ..., x_m^*)$ in s is obtained by solving the following linear program (Nisan et al. 2007):

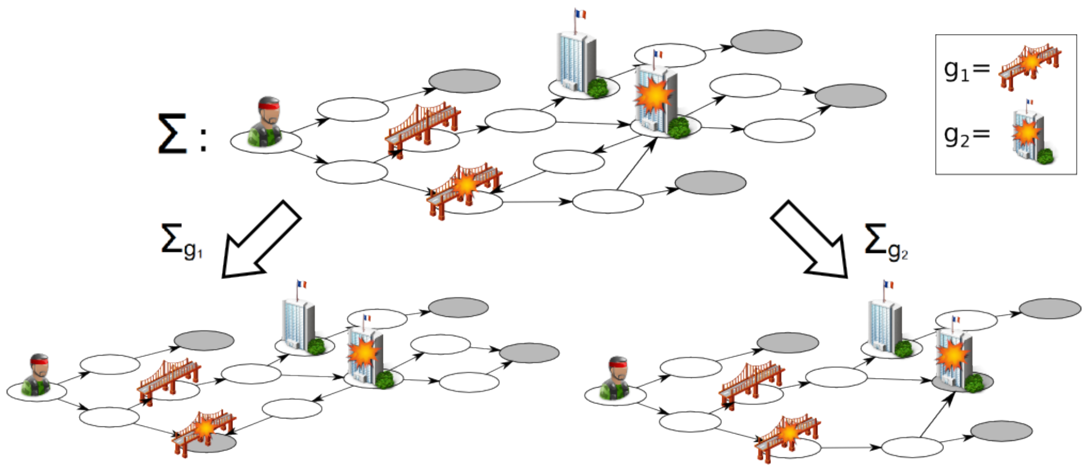

Figure 3: Illustration of the game generation procedure. In this example, the attacker has two possible goals: bombing the bridge (g_1) and/or attacking the embassy (g_2). Terminal states are represented as grey ellipses and do not necessarily corresponds to goal states. By defining terminal situations distinct from goal states, we allow an attacker to achieve a sequence of goals. From the projected situation Σ resulting from the situation projection procedure, the game generation algorithm derives two Markov games Σ_{g_1} and Σ_{g_2} associated to goal g_1 and g_2 respectively, by converting their respective goal states into terminal states and pruning unreachable successor states. The attacker can achieve the sequence of goal (*bombBridge, attackEmbassy*) by executing the optimal strategy π^*_{1,g_1}, then switch to strategy π^*_{1,g_2} once he reach the first goal state.

$$\begin{array}{rl} \textit{maximize} & v \\ s.t & \sum_i x_i\, \mathbf{R_g}(\mathbf{s})_{i,j} \geqslant v, \;\; \textit{for}\; 1 \leqslant j \leqslant n \\ \textit{and} & \sum_i x_i = 1. \end{array}$$

The mixed minimax strategy $y^* = (y^*_1, \ldots, y^*_n)$ in state s is obtained by solving the analogous linear program:

$$\begin{array}{rl} \textit{minimize} & v \\ s.t & \sum_i y_i\, \mathbf{R_g}(\mathbf{s})^T_{i,j} \leqslant v, \;\; \textit{for}\; 1 \leqslant j \leqslant m \\ \textit{and} & \sum_i y_i = 1. \end{array}$$

An APRSG Γ_g is played in discrete time steps. In each time step t, each $i \in Ag$ chooses an opportunity o^i_t from his set of available opportunities $O_i(s_t)$, where s_t is the current state of the game. Decisions are simultaneous and the joint opportunity $\mathbf{o_t} \in \mathbf{O_g}(\mathbf{s_t})$ is obtained. Each player i received a reward $R^i_g(s_t, \mathbf{o_t})$ and the players are transferred to the next state s_{t+1}. A policy $\pi_{i,g} : SO_g \rightarrow [0,1]$ for agent i defines for each state of Γ_g a mixed strategy, i.e. a probability distribution over the set of available opportunities.

We use Shapley's algorithm (Shapley 1953), an extension of the Value-Iteration algorithm to the case of stochastic games, to find one Nash equilibrium $\pi^*_g = (\pi^*_{1,g}, \pi^*_{2,g})$ in each Γ_g. Shapley's algorithm iteratively builds a normal form game $M(s)$ for each state $s \in S_g$ by using a value function which accounts for both the immediate utility of choosing an action in s and the long-term utility of playing the equilibrium starting from the next state. The optimal joint strategy $\pi^*(s)$ is then obtained by calculating the maximin and minimax strategies for the two-player zero-sum static game $M(s)$. Once a Nash equilibrium π^*_g is known for each Γ_g, we can finally build the set $\Pi^*_P = \{\pi^*_{1,\ g} | \forall g \in G\}$ of optimal plans for the attacker and the set $br(\Pi^*_P) = \{\pi^*_{2,\ g} | \forall g \in G\}$ of best responses for the defender given problem P.

Plan Recognition

The aim of the plan recognition module is to infer the current intent (goal and plan) of the attacker from observations of his actions. The past behaviors of both the attacker and the defender are represented as an observation history H_h which keeps record of the last h states visited by the players and of the joint opportunities that were played in these states: $H_h(t) = \{(s_{t-(h-1)}, \mathbf{o_{t-(h-1)}}), \ldots, (s_t, \mathbf{o_t})\}$.

According to the rationality principle underlying the generative plan recognition paradigm, if the attacker intends to reach the desired end-state $g \in G$, then he will follow the optimal policy $\pi^*_g \in \Pi^*_P$ to achieve g. Our plan recognition procedure acts by creating and updating at each time step t a belief state defined over the set Π^*_P of optimal plans for the attacker. This belief state is represented by a belief function $b_t : \Pi^*_P \longrightarrow [0,1]$ with $b_t(\pi^*_g) = P\left(\pi_t = \pi^*_g | H_h(t), s_t\right)$, where π_t is the policy of the attacker at time t. Hence $b_t(\pi^*_g)$ represents the belief of the defender that, at time t, the attacker is following policy $\pi^*_g \in \Pi^*_P$ given the observation history $H_h(t)$.

Proposition 1. *Let o_t be the opportunity played by the attacker in state s_t. Given history $H_h(t)$, the belief that the attacker is following policy π^*_g at t is given by:*

$$b_t(\pi^*_g) \propto P(o_t | \pi^*_g, s_t) \times \prod_{i=1}^{h-1} P(o_{t-i} | \pi^*_g, s_{t-i}) T_g(s_{t-i}, \mathbf{o_{t-i}}, s_{t-i+1}),$$

with the constraint that b_t is a probability distribution over Π^*_P.

Proof. From Bayes' rule:

$$P\left(\pi_g^*|H_h(t),s_t\right)=P\left(H_h(t)|\pi_g^*,s_t\right)P\left(\pi_g^*,s_t\right)/P\left(H_h(t),s_t\right),$$

Let $H_n = H_{h-n}(t)$ (i.e. the last $h-n$ observations at t), and $\tau = t-(h-1)$. The expression for $P\left(H_h(t)|\pi_g^*,s_t\right)$ results from the following recursive decomposition

$$P\left(H_n|\pi_g^*,s_{\tau+n}\right)=P\left(o_{\tau+n}|\pi_g^*,s_{\tau+n}\right) T_g\left(s_{\tau+n},\mathbf{o}_{\tau+\mathbf{n}},s_{\tau+(n+1)}\right)P\left(H_{n+1}|\pi_g^*,s_{\tau+(n+1)}\right),$$

with $H_{n+1}=H_n\setminus\{(s_{\tau+n},\mathbf{o}_{\tau+\mathbf{n}})\}$. Therefore, starting from state $s_{t-(h-1)}$ ($n=0$), we obtain

$$P\left(H_h(t)|\pi_g^*,s_t\right)=P(o_t|\pi_g^*,s_t)\times\prod_{i=1}^{h-1}P(o_{t-i}|\pi_g^*,s_{t-i}) T_g(s_{t-i},\mathbf{o}_{\mathbf{t-i}},s_{t-i+1}).$$

□

The use of a finite length history allows us to handle the fact that the adversary may change his intent during the execution of the plan recognition procedure. By keeping only "up-to-date" information, we prevent old observations inconsistent with the new intent from deteriorating the solution of the inference process. Another important remark is that the plan recognition algorithm introduced in Proposition 1 does not require the plan library to contain optimal strategies, but can be used to recognize any type of plan, with the constraint however that these plans are represented as MDP policies.

Experimental results

Scenario description

We perform our experiments on the scenario depicted in Figure 4. In this scenario, the defender is the Blue force (bf) with capabilities "move to", "attack red force", and "do nothing", and the attacker is the Red force (rf) with capabilities "move to", "attack embassy", "attack civilians", "attack blue force" and "do nothing". Each capability is associated with an area of effect specifying the type of target which can be acted upon, the action range, and the probability of success of the action. The set of possible goals of the attacker is $G=\{g_i = destroyed(target_i)|i=1...6\}$. Of course, the true goal of the attacker is hidden to the defender. The mission of the defender is to determine which target has been chosen and to protect this target by eliminating the attacker. In this particular set of experiments, we assume that the attacker has only one single goal (hence he will not try to achieve several goals sequentially) and therefore the set of terminal situations is $\mathscr{T}=G\cup\{destroyed(rf),destroyed(bf)\}$. We also define a simple application-dependent payoff function $R_g^i: S\rightarrow\mathbb{R}$ which associates immediate rewards to each state of a game given a goal $g\in G$:

$$R_g^i(s)=\begin{cases} = & 200, & \text{if } s\models g,\\ = & 50, & \text{if } s\models destroyed(bf),\\ = & -100, & \text{if } s\models destroyed(rf),\\ = & \alpha d(rf,target)-\beta d(rf,bf), & \text{otherwise,}\end{cases}$$

with $d(rf,target)$ the distance between the attacker and his target, $d(rf,bf)$ the distance between the attacker and the defender, and $\alpha+\beta=1$.

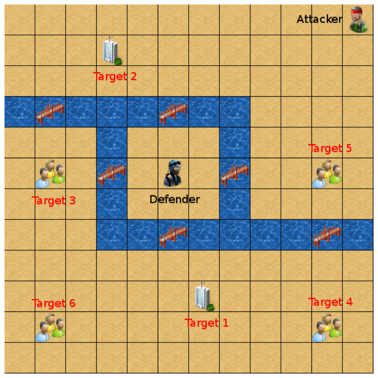

Figure 4: Initial situation of our urban warfare scenario.

Experiments We use the scenario defined in the previous section to evaluate the ability of our generative game theoretic framework to recognize the current plan of a rational adversary. We also verify that the recognition of the attacker's plan is rapid enough for the defender to select and implements the appropriate response before the attacker reaches his goal. With this in mind, we define two possible strategy selection policies for the defender. Let $b_t(\pi_g^*)$ be the belief state of the defender at time t:

WAIT : the defender waits until $\exists g\in G$ such as $b_t(\pi_g^*)=1$, then executes the best response $br(\pi_g^*)$.

MAX : the defender executes the best response $br(\pi_g^*)$ with $g = argmax_{g'\in G}\ b_t(\pi_{g'}^*)$, and chooses randomly between goals with the same maximum likelihood.

Figure 5 shows the belief state $b_t(\pi_g^*)$ for $g = destroyed(target_6)$ as a function of time. As we can see from this plot, the true goal of the attacker is correctly recognized by our plan recognition algorithm at $t=16$. This is not a surprise: since the attacker is assumed to be strictly rational, we are guaranteed to recognize his real goal in finite time (in the worst case when the attacker executes the final action leading to the achievement of the goal), unless a terminal situation such as $destroyed(bf)$ or $destroyed(rf)$ is achieved before the true goal has been recognized. The real question is therefore to know if our plan recognition engine is able to recognize the real goal of the attacker before it is fulfilled.

Policy	Goal 1	Goal 2	Goal 3	Goal 4	Goal 5	Goal 6
MAX (%)	98.9 ± 1.0	75.5 ± 2.6	100	80.4 ± 3.8	89.2 ± 3.8	100
WAIT (%)	100	42.1 ± 4.5	100	17.6 ± 2.7	100 ± 0	100

Table 2: Mean percentage of instances during which the real goal of the attacker was recognized before it was achieved.

Table 2 contains the results of the evaluation of the ability of our plan recognition method to infer the real goal of the adversary before he achieves this goal. For instance, when the true goal of the adversary is goal 1 and the strategy selection policy of the defender is **MAX**, our approach is able to recognize the goal of the attacker before the destruction of the target in 98.9% of cases (mean over 100 runs of 100

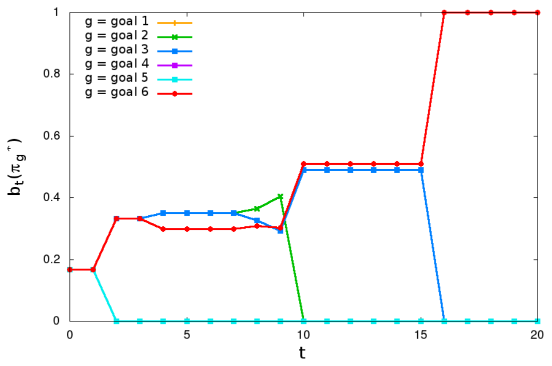

Figure 5: Evolution of the defender's belief about the real goal of the attacker.

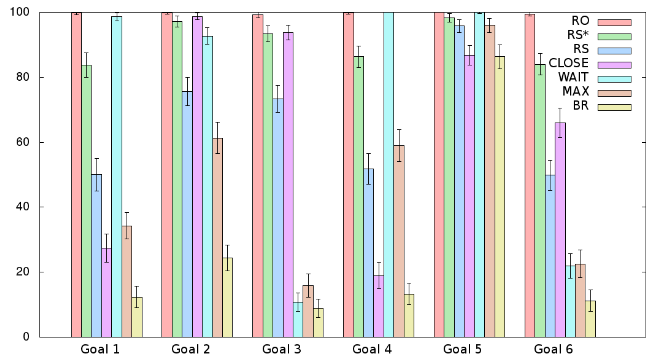

Figure 6: Mean attacker's percentage of success when using **MAX** and **WAIT**, compared to **BR** and other baselines.

scenario instances). Our method performs better on average when associated to the **MAX** selection policy, since it seems to lead the attacker in being less unclear about the goal he is chasing. This is particularly true for ambiguous goals such as goal 2 (whose optimal policy is very similar to those of goals 3 and 6) and goal 4 (similar to goal 5 and 1).

From Table 2, we saw that our plan recognition method is able to infer the real goal of an adversary before the achievement of this goal in most cases. But does this inference occurs soon enough for the defender to implement the appropriate response and defend the target? To evaluate this point, we define several strategy formulation policies which will serve as baseline values for comparison with the **WAIT** and **MAX** policies defined above:

RO : the defender selects an opportunity randomly in each state of the game (uniform distribution).

RS* : the defender selects a goal g randomly in each state of the game and executes the best response $br(\pi_g^*)$.

RS : the defender selects a goal g randomly at the beginning of the game and executes the best response $br(\pi_g^*)$.

CLOSE : the defender selects the goal g which is closest from the attacker in each state of the game and executes the best response $br(\pi_g^*)$.

BR : the defender always executes the best response $br(\pi_g^*)$ where g is the true goal of the attacker.

The quality of each policy is evaluated by averaging the number of scenario instances during which the attacker successfully achieved his goal over a total of 100 runs of 100 instances for each goal. From the results depicted in Figure 6, we can see that the **MAX** strategy formulation policy, which relies on our plan recognition engine, performs better on average than every other baseline (except **BR** of course). In a few cases however (goal 3 and 6), it may be preferable for the defender to wait to be certain about the real goal of the attacker before acting. But the disappointing results of the **WAIT** policy on the other 4 goals tend to confirm this famous quote from Prussian general and military theorist Carl Von Clausewitz

The greatest enemy of a good plan is the dream of a perfect plan (Clausewitz 1832).

Conclusion and Future Work

In this paper, we presented PRESAGE, a generative game theoretic framework for adversarial plan recognition in stochastic multi-agent environments. This framework generates a library of optimal plans for a rational attacker from the definition of a planning problem. We proposed an algorithm for situation projection relying on a simple yet expressive contextual action model. We showed how a set of Markov games can be generated automatically from this projected situation to model the planning processes of both the defender and the attacker. By computing one Nash equilibrium for each one of these games, we built a library of optimal plans for the attacker and a set of best responses for the defender. This library was later exploited by a probabilistic plan recognition algorithm to infer the current plan of the adversary from observations of his behavior. Finally, we demonstrated the capability of our adversarial plan recognition to assist a decision-maker in selecting the most appropriate response to a given threat.

An interesting direction for future works would be to relax the attacker's rationality assumption and to adapt our plan recognition algorithm to the case of an adversary with bounded rationality. We also plan to extend our framework in order to deal with an adversary which would be actively hostile to the inference of his plans. This would require the ability to detect deceptive actions performed by the attacker. We would also have to relax the assumption of full observability of the opponent's actions, since he may use concealment. Currently, our APRSGs are defined as two-player zero-sum stochastic games and therefore, they can only model the strategic interactions between one defender and one attacker. Our intuition is that our framework can be generalized quite directly to the 1 vs N and N vs N cases.

Acknowledgments

We thank Bruno Zanuttini, assistant professor with habilitation at the University of Caen Basse-Normandie, for helpful comments and corrections.

References

Avrahami-Zilberbrand, D., and Kaminka, G. A. 2007. Incorporating observer biases in keyhole plan recognition (efficiently!). In *AAAI*, volume 7, 944–949.

Baker, C. L.; Saxe, R.; and Tenenbaum, J. B. 2009. Action understanding as inverse planning. *Cognition* 113(3):329–349.

Braynov, S. 2006. Adversarial planning and plan recognition: Two sides of the same coin. In *Secure Knowledge Management Workshop*.

Burkov, A., and Chaib-draa, B. 2008. Stochastic games. In *Markov Decision Processes in Artificial Intelligence*, volume 1. Wiley Online Library. 229–276.

Chen, G.; Shen, D.; Kwan, C.; Jr., J. B. C.; Kruger, M.; and Blasch, E. 2007. Game theoretic approach to threat prediction and situation awareness. *Journal of Advances in Information Fusion* 2(1):35–48.

Clausewitz, C. v. 1832. *On War*. Ed./trans. Michael Howard and Peter Paret. Princeton University Press, 1976, revised 1984.

Garcia, F., and Rachelson, E. 2008. MDPs: models and methods. In *Markov Decision Processes in Artificial Intelligence*, volume 1. Wiley Online Library. 1–38.

Geib, C. W., and Goldman, R. P. 2001. Plan recognition in intrusion detection systems. In *Proceedings of DARPA Information Survivability Conference &amp; Exposition II, 2001. DISCEX'01.*, volume 1, 46–55. IEEE.

Geib, C. W., and Goldman, R. P. 2009. A probabilistic plan recognition algorithm based on plan tree grammars. *Artificial Intelligence* 173(11):1101–1132.

Kott, A., and McEneaney, W. M. 2006. *Adversarial reasoning: computational approaches to reading the opponents mind*. CRC Press.

Lisỳ, V.; Píbil, R.; Stiborek, J.; Bosanskỳ, B.; and Pechoucek, M. 2012. Game-theoretic approach to adversarial plan recognition. In *ECAI*, 546–551.

Little, E. G., and Rogova, G. L. 2006. An ontological analysis of threat and vulnerability. In *9th International Conference on Information Fusion*, 1–8. IEEE.

Neumann, J. v. 1928. Zur theorie der gesellschaftsspiele. *Mathematische Annalen* 100(1):295–320.

Nisan, N.; Roughgarden, T.; Tardos, E.; and Vazirani, V. V. 2007. *Algorithmic game theory*, volume 1. Cambridge University Press Cambridge.

Ramırez, M., and Geffner, H. 2009. Plan recognition as planning. In *Proceedings of the 21st international joint conference on Artifical intelligence. Morgan Kaufmann Publishers Inc*, 1778–1783.

Ramırez, M., and Geffner, H. 2010. Probabilistic plan recognition using off-the-shelf classical planners. In *Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence (AAAI 2010)*.

Ramırez, M., and Geffner, H. 2011. Goal recognition over pomdps: Inferring the intention of a pomdp agent. In *IJCAI*, 2009–2014.

Shapley, L. S. 1953. Stochastic games. *Proceedings of the National Academy of Sciences of the United States of America* 39(10):1095.

Steinberg, A. N. 2005. An approach to threat assessment. In *8th International Conference on Information Fusion*, 1–8. IEEE.

Steinberg, A. N. 2007. Predictive modeling of interacting agents. In *10th International Conference on Information Fusion*, 1–6. IEEE.

Stoelinga, M. 2002. An introduction to probabilistic automata. *Bulletin of the EATCS* 78(176-198):2.