Constructing adversarial models for threat/enemy intent prediction and inferencing

Eugene Santos Jr. and Allesandro Negri
Intelligent Distributed Information Systems Laboratory
University of Connecticut
Storrs, CT 06269-3155
eugene@engr.uconn.edu

ABSTRACT

We examine an adversary model that captures goals, intentions, biases, beliefs, and perceptions based on a dynamic cognitive architecture that evolves over time. The model manages the uncertainty surrounding the adversary using probabilistic networks. In particular, we consider the challenges of constructing such adversaries and provide solutions towards more effective and efficient engineering of such adversaries. We present the AII Template Generator tool which enables the rapid deployment of adversary models as well as on-demand construction of new adversary components.

Keywords: adversarial intent inferencing, decision making, knowledge acquisition, adversarial modeling, model building

1. INTRODUCTION

Modern elements of war gaming and mission planning and execution ultimately require a computational model of dynamic adversary behaviors. This is especially critical to future operations with regards to predictive analysis and predictive battlespace awareness (PBA) which “…involves studying an adversary to understand what he’ll do, how he’ll do it, what his capacity to inflict harm will be, and the environment in which he is operating — in short, knowing the scene of the crime before the crime is committed” [3]. In particular, determining an adversary’s courses of actions (COAs) involves predicting and evaluating their likely goals, objectives, actions, and desired end states. This is in order to provide the identification and prioritization of the full of set of friendly courses of actions to meet the adversarial threat [11].

As noted in [35]: “In the current world environment, the rapidly changing dynamics of organizational adversaries are increasing the difficulty for Military Analysts and Planners to accurately predict potential actions. As an integral part of the planning process, we need to assess our planning strategies against the range of potential adversarial actions. This dynamic world environment has established a necessity to develop tools to assist in establishing hypotheses for future adversary actions.” Thus, without an automated or even systematic computational approach to adversarial decision modeling, the ability to accurately capture and predict (let alone even update) adversary intentions is fundamentally limited by the speed and effectiveness of the human decision-maker, hence, presenting a multitude of challenges with respect to the necessary speed and complexity of current and future military operations. For example, in contemporary war gaming, one challenge is to: “[…] develop automated techniques that allow the exploration of as many courses of action as possible. Manual play limits the number of games as well as concepts and courses of action that can be addressed. Because of time and resource constraints, war games tend to address a specific scenario. The resulting strategy tends to be tailored to that scenario. It may not be robust as it does not take into account possible alternative opponent strategies and their implications. […] Games must include not only attrition warfare but effects based, network centric, and other additional styles of warfare.” [23] (emphasis added).

In this paper, we present our work in adversary intent inferencing sponsored through the auspices of the Air Force Research Laboratories, Information Directorate and the Information Institute. Our adversary model captures the adversary’s goals, intentions, biases, beliefs, and perceptions based on a dynamic cognitive architecture that evolves over time and manages the uncertainty surrounding the adversary using probabilistic networks. Our model has been successfully deployed within multiple war gaming environments to explore the effects of dynamic adversarial behaviour
and emergent adversary COAs on potential blue COAs [35, 23]. In particular, we examine the challenges of constructing such adversaries and provide solutions towards more effective and efficient engineering of such adversaries. We present the AII Template Generator tool which enables the rapid deployment of adversary models as well as on-demand construction of new adversary components. We conclude this paper with a discussion of future concepts and directions for adversary intent inferencing.

2. BACKGROUND: TERMS AND CONCEPTS

In this section, we provide some background on terms and concepts we will be using throughout this paper. For additional details and discussion see [4, 5, 11, 26, 29, 31, 32, 33].

**Intent Inferencing.** Intent inference involves deducing an entity’s goals based on observations of that entity’s actions [15, 25]. Such deduction involves the construction of one or more behavioral models that have been optimized to the entity’s behavior patterns. Data/knowledge representing observations of an entity, the entity’s actions, or the entity’s environment (collectively called observables) are collected and delivered to the model(s), which match the observables against patterns of behavior and derive inferred intent from those patterns. These inferences can then be passed to an application for generation of advice, definition of future information requirements, proactive aiding, or a host of other benefits. For this paper, we are interested in predicting future adversarial actions, explanation of their behaviors, generation of red courses of actions, and “what-if” analyses to determine the effectiveness of blue courses of actions which is all central to effects-based operations (EBO) [13, 18].

Intent inference can be dissected into three informational components [25, 27, 29, 31, 14]: The first, *interests and focus*, captures at a high level the direction of the entity’s attention. The second, *actions and preferences*, describes the activities that can be used to carry out the goals that currently hold the entity’s attention, with a focus on how the entity tends to carry them out. The third, *knowledge and reasoning*, provides insight into the deeper motivations behind the goals upon which the entity is focused and illuminates connections between goals. In other words, the first component captures what the entity is doing, the second captures how the entity might do it, and the third infers why the entity is doing it.

Applying the principles of modeling the what/how/why of entity intent, we see that our approach naturally integrates into the major themes of EBO. While the field of individual intent inference has historically focused on better improving the human-system interface, we contend that there is a natural isomorphism between our own prior work in the field of user and team intent inference [8, 10, 11, 27, 29, 30, 31] and the domain of adversary intent inference [5, 11]. While the operational world surrounding an intent inference application would be very different, the inner mechanisms of intent inference map directly between domains as we have demonstrated in [23, 25, 35].

While observables in the user intent domain stem from data collected from human use of systems, observables in the adversary intent domain take the form of tactical information derived from intelligence databases, observations of the tactical environment, and input from online human experts. In place of window events, keystrokes, and mouse movements, our system in the adversary intent domain uses information about adversary location, movements, and activities to drive its inference. In place of computer state, analyses of information queries, and the content of user dialogue with team members, our system bases inferences on facts about the local terrain, regional weather, and the salient political climate.

Likewise, tactical goals will replace computer operational goals in the results of our intent inference. This would result in identification of an adversary force’s objectives and, given models of tactical reasoning, could recommend appropriate reactions. Furthermore, this would indicate expected activity by the adversary and explain the reasons behind that activity. Finally, we can produce alerts of attempted subterfuge or uncover missteps on the part of the adversary.

**Uncertainty and Bayesian Knowledge-Bases.** We must capture the uncertainties inherent in the adversarial model as well as the uncertainties found in the observables. In probabilistic reasoning, random variables (abbreviated, r.v.) are used to represent events and/or objects in the world. By making various instantiations to these r.v.s, we can model the current state of the world probabilistically. Thus, this will involve computing joint probabilities of the given r.v.s.
Unfortunately, the task is nearly impossible without additional information concerning relationships between the r.v.s. In the worst case, we would need the probabilities of every instantiation combination which is combinatorially explosive.

On the other hand, consider the chain rule as follows:


Bayesian networks [20] take this process further by making the important observation that certain r.v. pairs may become uncorrelated once information concerning some other r.v.(s) is known. More precisely, we may have the following independence condition:

\[ P(A | C_1, ..., C_n, U) = P(A | C_1, ..., C_n) \]

for some collection of r.v.s U. Intuitively, we can interpret this as saying that A is determined by C1, ..., Cn regardless of U.

Combined with the chain rule, these conditional independencies allow us to replace the terms in the chain rule with the smaller conditionals. Thus, instead of explicitly keeping the joint probabilities, all we need are smaller conditional probability tables which we can then use to compute the joint probabilities. In this manner, a unique joint probability distribution is defined over the events (r.v.s) of interest. As such, we can then answer probabilistic questions of the form “What is \( P(A_1 = a_1, ..., A_m = a_m \mid B_1 = b_1, ..., B_n = b_n) \)” by taking the sum and product of appropriate conditional probabilities.

In Bayesian networks, these conditional dependencies are represented as a directed acyclic graph of r.v. relationships. Directed arcs between r.v.s represent direct conditional dependencies. When all the parents of a given r.v. A are instantiated, that r.v. is said to be conditionally independent of the remaining r.v.s which are not descendants of A given it's parents. (For more details on this, see d-separation in [20].) Although Bayesian networks have been successfully used to prototype numerous intelligent systems including adversarial intent inference [25, 35] and the causal analysis tool [17, 21] for EBO, there are limitations to constructing such networks as we have discussed above. In this paper, we recommend another uncertainty model called Bayesian Knowledge-Bases (BKBs).

Bayesian Knowledge-Bases (BKBs) are a generalization of Bayesian networks. BKBs have been extensively studied both theoretically [32, 33] and for use in knowledge engineering [24] in a wide variety of domains such as space shuttle engine diagnosis [2, 24], medical information processing [19], and freshwater aquarium maintenance [32]. BKBs provide a highly flexible and intuitive representation following a basic “if-then” structure in conjunction with probability theory. Furthermore, BKBs were designed keeping in mind typical domain incompleteness to retain semantic consistency as well as soundness of inference in the absence of complete knowledge. Bayesian networks, on the other hand, typically assume a complete probability distribution is available from the start. Also, BKBs have been shown to capture knowledge at a finer level of detail as well as knowledge that would be cyclical (hence disallowed) in BNs.

As described in [33], probabilistic models exhibiting significant local structure are common. In such models, explicit representation of that structure as done in BKBs, is advantageous, as the resulting representation is much more compact than the full table representation of the conditional probability tables (CPTs) in a BN. For example, consider the following setting: X, a binary variable, is known to be true if any of the variables \( Y_i \) is true for i from 1 to n, and X is false with probability p otherwise. The global structure here is that X depends on all the \( Y_i \) and in a BN one might represent this with a set of arcs \((Y_i, X)\) for i from 1 to n. The representation of the distribution in the “standard” form of a CPT would require \( O(2^n) \) entries. However, the (partially) given distribution also exhibits “local” structure, as when \( Y_i \) is known to be true for some i, X no longer depends on the value of \( Y_j \) for j not equal to i. The size of the representation of the conditional probabilities in terms of rules is only \( O(n) \). Although work has been done on representing local structure using other methods, such as local decision trees and default tables [34, 7], rules have significant advantages in size of the representation, as well as their better explainability.
For example, contrasting rules with decision trees as a representation of local structure, every decision tree is compactly representable as a set of rules, while the reverse is not necessarily true - the decision tree may be exponentially larger than the set of rules [1]. Although rule-based systems for representing an exact distribution exist (e.g. [22]), these systems are a (compact) notational variant of Bayesian networks, and are thus less flexible than BKBs, as they do not allow for incompleteness or cyclicity.

3. OVERVIEW OF ADVERSARY INTENT INFERENCING ARCHITECTURE

In this section, we provide a brief overview of our basic adversary intent architecture. In [25], we decomposed the architecture into the what/how/why model in order to provide a natural and intuitive organization of both the adversarial decision-making process and central knowledge-base. The components of our adversary intent inferencing model, and the interactions between these components, are shown in Figure 1. The three core components that comprise our architecture and functions are as follows:

1. **Goals/Foci**: Probabilistically prioritized short- and long-term goals list, representing adversary intents, objectives or foci
2. **Rationale**: A probabilistic network, representing the influences of the adversary’s beliefs, both about themselves and about us, on their goals and on certain high level actions associated with those goals
3. **Actions**: A probabilistic network, representing the detailed relationships between adversary goals and the actions they are likely to perform to realize those goals

The goal component captures what the adversary is doing, the action component captures how the adversary might do it, and the rationale component infers why the individual is doing it. Due to the inherent uncertainty involved in adversary course of action prediction, we used Bayesian networks as the main knowledge representation for the rationale and action networks [25]. Each random variable (RV) involved in the Bayesian networks is classified into one of four classes: axioms, beliefs, goals, and actions. Each RV class is described below:

a) **Adversary axioms (X)** – represents the underlying beliefs of the adversary about themselves (vs. beliefs about our forces). This can range from an adversary’s beliefs about his or her own capabilities to modeling a fanatic’s belief of invulnerability. Axioms typically serve as inputs or explanations to the other RVs such as adversary goals.

b) **Adversary beliefs (B)** – represents the adversary’s beliefs regarding our forces (e.g., an adversary may believe that the United States is on a crusade against them or that the United States is not carpet-bombing territory).

c) **Adversary goals (G)** – represents the goals or desired end-states of the adversary. These goals are defined as either short-term or long-term in a goals list. Further we partition goals into two types: abstract and concrete. Abstract goals have subgoals and are not immediately satisfiable by actions (e.g., preserving launchers, damage US world opinion, defeating US foreign policy). Concrete goals are satisfiable by actions.

d) **Adversary actions (A)** – represents the actions of the adversary that can typically be observed by friendly forces.

These four random variable types are arranged in the two networks: rationale network and action network. The rationale network contains all of the Belief (B), Axiom (X), and Goal (G) variables, as well as any Action (A) variables which have goals as inputs. This network is used to infer what short and long term goals the adversary may have. Once the goals are determined, the action network is used to reason on what the most likely actions will be that the adversary may carry out. The action net contains the entire set of Action (A) variables and any concrete Goal (G) variables. Figure 2 depicts a rationale network and an action network.

The AII process (as shown in Figure 1) works iteratively as follows:

1) Observables regarding the adversary such as actions and beliefs are set as evidence in both rationale and action networks (depicted as red nodes in figure). Also, feedback from analyst is set as evidence.

2) Current short- and long-term enemy foci from the foci lists are also set as evidence in both networks (depicted as green nodes).

3) The rationale network is then used to infer new goals which are set as evidence for the action network.

4) The action network is now used to predict adversarial actions.

5) The analyst is presented with the inferred goals and predicted actions.

6) The analyst provides feedback in terms of corrected goals and actions if desired.
7) The goals list is updated based on newly inferred goals and current strength of existing goals. If goals exceed a given threshold value, they are added to the list. If goals fall below a set threshold, they are removed. If goals in the short-term list persist beyond a given time threshold, they become long-term goals.

8) Go to step 1.

The inference process on both the rationale and action networks is based on belief updating [25]. In essence, given a target random variable R and evidence set E, belief updating computes P(R|E) assuming that random variables have two states (true/false) for simplicity of discussion.

As we can see in the above process, the adversary model is capable of adapting to changes in the adversaries goals and intentions over time as reflected in the enemy foci lists. Also note that there are feedback and explanation paths within the adversary intent inference (AII) model. Feedback from a human analyst, although unlikely to be totally certain, can be extremely valuable to the AII model, correcting and extending its intent inferencing logic. Explanation capabilities are essential in order for intelligence analysts, using AII, to understand why the AII model has reached particular inferences. The analysts must be able to inspect the reasoning paths used by AII so that they can develop a level of confidence in the output of the AII model.
4. HOW TO CONSTRUCT ADVERSARIES

In this section, we focus on how to organize the adversarial model and consider the structural relationships central to the rationale and action networks.

Semantic Structure. As we mentioned above, the four random variable types are arranged in the two networks: rationale network and action network. The rationale network contains all of the Belief (B), Axiom (X), and Goal (G) variables, as well as any Action (A) variables which have goals as inputs. This network is used to infer what short and long term goals the adversary may have. Once the goals are determined, the action network is used to reason on what the most likely actions will be that the adversary may carry out. The action net contains the entire set of Action (A) variables and any concrete Goal (G) variables. Of course, the question is “How do we systematically build such adversaries while minimizing the bottleneck-effect of knowledge acquisition?”

In [25, 35], we proposed and utilized a fundamental structure of how the networks should be organized and built. Here, we expand the semantic structure in order to further account for the impact of blue actions on the adversary’s actions. As a rule, belief variables are independent and serve as inputs to Axioms, Goals, or Actions. Belief variables can be categorized into two basic types: strategic beliefs and tactical beliefs. Strategic beliefs include philosophy, strategic goals, and general characteristics/behaviors of blue from the adversary’s point of view (such as those described in the earlier examples). Tactical beliefs represent actionable blue events such as physical repositioning of assets, specific kinetic attacks, etc. While it seems reasonable to construct a dependency structure among the belief variables especially tactical beliefs that represent sequences or hierarchies of blue actions, this increases the complexity of the networks. Since blue actions are typically known with certainty, the belief variables can be set as evidence which has the effect of rendering the beliefs independent. Still, if the added complexity is acceptable, partial evidence on blue activity can reflect the “fog of war” in which the tactical beliefs can be probabilistically inter-related and thus provides a natural sensitivity/what-if analysis of blue actions with respect to the adversary. In this case, we would amend the semantic structure that organizes the beliefs into (multiple) hierarchies of beliefs such that Belief variables can serve as input to other Belief variables, strategic beliefs serve as input to Goals, and tactical beliefs can serve as input to Goals and Actions. We also allow beliefs to occur in either the rationale or action network.

Next, Axioms have strategic Beliefs as inputs and serve as inputs to Goals and other Axioms. Goals have Axioms and Beliefs as inputs and serve as inputs to Actions or other Goals. Actions have Goals and tactical Beliefs as inputs and can only be inputs to other Actions. Basically, the structure follows an intuitive hierarchical pre- and post-condition organization. Hence, there is a natural dominance relationship between related variable types, say Axioms that are descendants of other Axioms. This dominance reflects the fact that one variable may be “more general”, “more abstract”, “aggregates”, “pre-conditions”, etc. with respect to a descendant variable.

In order to maintain the appropriate division of variables among the rationale and action networks, Goal variables are partitioned into abstract goals and concrete goals. Abstract goals are goals composed of additional Goal variables. Concrete goals are goals that are immediately actionable, i.e., satisfied by one or more Action variables. As such, abstract goals can only appear in the rationale network and are critical to providing the proper explanations for adversary rationale. Concrete goals must appear in both networks and serves as the causal “glue” between the networks.

Finally, there are three basic rules that maintains the semantic integrity of the AII:

1. All axioms, beliefs, goals and actions occur in at least one of the adversary rationale or action networks.
2. Given a concrete goal G, if A is an action node with input G, then G, A, and the inputs of A must occur in both networks with the same connection structure.
3. All Bayesian networks must be directed acyclic graphs (DAGs).

Rule 2 is particularly important in that it permits the propagation of reasoning between the two networks. Such propagation occurs in both directions: rationale to action to rationale, reflecting the predictive and explanatory processes in the AII.

Causal Structuring. While the rules above for structuring the relationships between the major classes of variables ensures a proper semantic organization throughout the AII model, there still remains basic engineering issues in
constructing the specific relationships especially with regards to causal structuring. Consider scenarios where you have collections of potentially mutually exclusive events. For example,

- **Case 1:** Simple unit movement – Ground Unit A can move in one of 8 possible directions \{N, NE, E, SE, S, SW, W, NW\}. If the domain consists of a fixed octagonal-grid, then the movement direction (not moving is not considered here) must be mutually exclusive. Often, a single random variable (rv) is used in this case to indicate direction. However, if fidelity of movement needs to be considered, then one can have a N NE movement. As such, eight rvs corresponding to the eight directions of movement for a given unit are used where each rv has a true/false value. Hence, N NE is represented by two true assignments. This case is not recommended unless absolutely necessary given the problems of a potential N S setting with the rvs. [Note – there is a solution to handling this latter problem by generalizing our mutual exclusion mechanism as described later. In essence, to prevent a simultaneous N S setting, one can introduce a new rv that causes such a N S setting to have a 0 probability.]

- **Case 2:** Complimentary actions – Air Unit A attacking SAM Site S and Air Unit B attacking SAM Site S. If only one Air Unit can be engaging a SAM Site at any given time, then these two events must be mutually exclusive. A typical approach would be to have a single rv that has states \{Air Unit A attacking SAM Site S, Air Unit B attacking SAM Site S, no attack against SAM Site S\}. Let’s complicate the problem by considering that Air Unit A can also attack Bunker Q. If the events are mutually exclusive, we must make sure that if the rv controlling attacks on SAM Site S must make sure that it is a parent of the rv to stop Air Unit A from attacking Bunker Q if Air Unit A must also attack SAM Site S.

The largest knowledge acquisition difficulty with the examples above when one uses single rvs to deal with mutual exclusion is the problem of capturing the appropriate interactions between all rvs such as in Case 2. Such interactions can lead to serious looping and spaghetti-like constructs which ultimately leads to a failure in capturing the correct information. Furthermore, we incur a significant space and time complexity because of the explosion in the size of the conditional probability tables when a single rv is used to capture mutual exclusion since that rv has to be the parent of any rv that needs information regarding one of its states. For example, if some rv wants to know Air Unit A’s current attack, it must look at the rv which has at least 3 states.

A simple solution to mutual exclusion in our model is the following: Assume all rvs are true/false only. Let A1, A2, ..., An be rvs that need to be mutually exclusive. Construct a new rv B that is also true/false with parents A1, ..., An. Now, set the conditional probability table for B as follows: \(P(B=\text{true} | \text{only one } A_i \text{ is true}) = 1.0, 0.0 \text{ otherwise.} \) \(P(B=\text{false} | ...) = 1.0 - P(B=\text{true} | ...)\), of course. [Note – we assume here that one of the Ai’s must be true. This can be easily modified if it is necessary to model when all Ai’s are false as a valid state.] In order for the model to function properly, make B of type Axiom. If mutual exclusion is desired between A1, ..., An, then set B=true as evidence. This will guarantee that only one of the rvs will be set to true since all other scenarios will result in a 0 probability. By separating out the states, this will reduce the explosion and spaghetti structure involved in the single rv method. Construction should also be more methodical.

We also recommend the migration to Bayesian Knowledge-Bases as the knowledge representation [32, 33]. BKBs do not require a complete instantiation of all conditional probabilities since BKBs are rule-based in nature. Consider a BKB rule: If A = true and B = true then C = false with probability p. As such, \(p = P(C=\text{false} | A=\text{true}, B=\text{true})\). We do not have to specify other combinations of A, B, and C unless we wish to or the information is relevant. Also, we can capture: If B=false and D=true and E=false, then C=true with probability p’ can also be handled in a BKB together with the last rule. Finally, in situations where we find that we have a rule: If C = true, then A=false with probability p” We can also capture this in a BKB. Thus, a BKB allows for incomplete information (avoids explosion from not having to specify all conditional probabilities), content-specific rules, and limited forms of cyclicity. As presented in BKBs, there exists a simple set of rules for building BKB rules that guarantees the BKB is consistent. The recommended mutual exclusion mechanisms above can be directly used in BKBs but without the \(2^n\) explosion since we would only need to encode the probabilities with a non-zero value. Thus, only need \(n\) rules. As a starting recommendation, try incompleteness and context-specific rules before trying cyclicity.
5. TEMPLATES FOR BUILDING AIIS

In this section, we describe a tool we have developed to assist in building the AII. In particular, the tool provides the functionality for building the various random variables, action network, and rationale network for the AII. Furthermore, it provides the capability for generating templates and instantiating them for the AII that should greatly ease the construction process. With such a library of templates, this will allow the AII’s to be modified “on-demand.” For example, assume the current AII is modeling an air commander adversary and his/her assets and capabilities. If a new asset is discovered such as an additional air base, a new SAM site, etc., this capability can be directly incorporated into the AII model by identifying the relevant template that defines these capabilities in general and appropriately instantiating them with regards to the specific characteristics and behavioral probabilities of the new capability, etc.

The Adversary Intent Inference model editor, also known as the AII Template Generator (ATG), is a tool for quickly constructing probabilistic networks specific to AII. This tool is being built because the creation of a set of Bayesian Network files required by the Adversary Intent Inference system is very tedious and time consuming. Prior to ATG, there have been compatibility problems between different applications that help build BNs and the rule/constraint handling that is central to the AII described earlier. The ATG allows for compatibility between different file types, such as from different Bayesian Network editors, by providing an import and export feature to translate existing BNs between applications. The ATG also provides validation of current BNs against the AII semantic structure rules. The BNs used in AII are very specific as well as structurally constrained by the types of random variables used. Hence error handling, prior to ATG, was left mostly to the creator of the files and to their respective intuition. Consequently, BN files created for the use in the AII take longer to build and are more likely to contain errors. The main feature of the ATG is the ability to create BN nodes with their respective random variables based on template information, which is user specified and can be used globally between different AII systems/projects. This allows for quick generation of random variables, nodes, and default probabilities for the nodes in the BNs.

The ATG allows the easy implementation of the same BN across different AII applications/systems. The ATG contains import and export features which actually converts other formats into its own XML format. The import features lets the application import, into an existing project, files from a number of BN application formats. Because there are three different specific files loaded into the Adversary Intent Inference system: the AII model (*.am), the rationale Bayesian Network (*.gra), and the action Bayesian Network (*.gra), the importing of an existing network will be in a similar format, with the advantage of generating the random variables automatically. Similarly the export feature will generate from an existing project the three files required by the Adversary Intent Inference system.

Template models offer quick and easy creation of BNs and random variables that comply with their respective constraints. Templates should be used for repetitive tasks that will need nodes of the same type of random variables as inputs (parents), or will need nodes with similar types of input, but with pre-set probability values. Template models are stored in the application globally; therefore, more than one project can then use the same template(s). Templates are organized into groups. The purpose of a template is to generate a specific model with pre-set probabilities in which random variables are bound to template variables that are of a specific type (Axiom | Goal | Action | Belief). Figure 3 shows the template dialog and the results after applying the template onto an empty Bayesian Network.

Figure 4 graphically depicts the Template stored in ATG, where a specific parent node (Go_IraquiMassForces of type Goal) is used as a parent onto a variable child node (Ac_IraquiForcesMassing), and the variable child node can be any node of type Action. Specifying the probabilities for the child node such as AtAreaA, AtAreaB, and AtAreaC will let the template tool generate the three nodes of type Action (Ac_IraquiForcesMassingAtAreaA, Ac_IraquiForcesMassingAtAreaB, and Ac_IraquiForcesMassingAtAreaC) with their respective parent node and pre-set probabilities after applying the template, thus generating quickly the set of nodes from the template used.

There are two ways that templates can be used in the ATG application:

1. Templates can be applied to existing or newly created models (nodes). This adds and maintains consistency within the model by only allowing the random variables of specific type to be bounded within the model, conforming to the template specification.
2. Templates and template groups can be dragged & dropped onto the BNs. This option is more complex, but allows us to systematically and consistently create new nodes. Only the templates supported by the rule for the type of BN are enabled, so that models that do not comply with the rule are not created. Next, the existing random variables are listed, but new random variables can be added as well, so that binding can be applied onto each of the templates’ inputs (the nodes parents) and outputs (the generated node). The systematic binding will allow as many different combinations for outputs, creating many different instances of the model (node) with distinct output values. Every node is an output, where there are different outputs/nodes with the same parents and probabilities, but there cannot exist any duplicates of one node with different parents. See Figure 5.
The ATG is a complementary tool to the AII system. The ATG allows quick creation of a set of BNs to use in the AII system. By allowing import and export of BN file formats ATG can be used as a “bridge” between different applications. Also, through constraint checking and validation, the development of these BNs to the specification of the AII rules should make debugging and troubleshooting much simpler. The use of templates in the development of these networks can reduce the creation/development time and helps avoid some repetitive tasks that can be done in a systematic manner. A library of such templates can also be exploited computationally to construct/update AII components on-the-fly during execution of the AII.

The next step for the ATG is to improve load efficiency on projects, allow undo procedures, improve rule and constraint checking of BN to increase accuracy, extend import and export capabilities, and implement a graphical view of the BNs by displaying a directed graph of the parent-to-child node relationships. We are also migrating towards a Bayesian Knowledge-Base representation for ATG.

6. ABOUT DECEPTION

As we described earlier, our model of the adversary’s biases and fundamental beliefs are modeled with the variables in the adversary Axioms. Again, this is different from those in adversary Beliefs which represent the adversary’s belief about us – blue forces. To recap, the current enemy foci/goals and the observables regarding the adversary are set as evidence in order to predict future adversary actions, determine adversary goals and changes in foci, etc. In essence, the collection of evidence represents the adversary’s perceptions. Clearly, the perceptions which are set as evidence (observables) heavily impacts the predicted adversary goals and actions and serves as the basis for explaining the predictions. However, those perceptions which have not been set as evidence because they have yet to be observed play a very important factor in adversary analysis. The explanation of a predicted goal or action may rest on assumptions regarding the unobserved perceptions. This leads to a rich series of what-if analyses that can be readily conducted to better support the model’s predictions as well as direct blue forces to gather more “targeted” information in order to support or refute a given prediction.

By considering adversary perceptions within the AII, this naturally provides a mechanism for handling deception (both red and blue). The objective of a deception can range from masking the true intent/goal of red or blue by hiding actions (say, camouflage or diverting attention from a given action say in urban operations [16]) to influencing and/or changing red or blue goals and desired end-states through misdirection or misinformation [28]. Combined with the what-if analysis and explanatory capabilities of AII, we can analyze the effects of deceptions as follows: For blue deceptions, we can naturally analyze their impacts by setting various combinations of adversary perceptions as observables corresponding to the target deceptions. The predictions and explanations generated can then be used to evaluate the effectiveness of the blue deception especially if the goal of the deception was to alter red’s actions or goals. Similarly, to detect/determine red deceptions, various observables can be applied and sensitivity analysis can be conducted on the resulting predictions to determine potential red COAs. From the sensitivity analysis, this can identify to blue what additional reconnaissance information should be obtained in order to pinpoint the red deception and red’s true goals. As we can see, the AII cognitive architecture provides a natural interface to take into account deception.
Acknowledgements. This work was supported in part by Air Force Research Labs, Information Directorate, Grant No. F30602-01-1-0595. Special thanks to Robert Hillman and Jim Hanna for insights and encouragements on this effort. Also, thanks to Frank Vetesi (LM ATL) for his practical comments on constructing adversaries. Finally, I would also like to especially thank John Graniero (AFRL, Information Institute), Don Monk (AFRL, Human Effectiveness), and Scott Brown (USAF/ESC) for their tremendous support for this research project.

REFERENCES


