Cyber Attribution: An Argumentation-Based Approach

Paulo Shakarian, Gerardo I. Simari, Geoffrey Moores, and Simon Parsons

Abstract Attributing a cyber-operation through the use of multiple pieces of technical evidence (i.e., malware reverse-engineering and source tracking) and conventional intelligence sources (i.e., human or signals intelligence) is a difficult problem not only due to the effort required to obtain evidence, but the ease with which an adversary can plant false evidence. In this paper, we introduce a formal reasoning system called the InCA (Intelligent Cyber Attribution) framework that is designed to aid an analyst in the attribution of a cyber-operation even when the available information is conflicting and/or uncertain. Our approach combines argumentation-based reasoning, logic programming, and probabilistic models to not only attribute an operation but also explain to the analyst why the system reaches its conclusions.

1 Introduction

An important issue in cyber-warfare is the puzzle of determining who was responsible for a given cyber-operation — be it an incident of attack, reconnaissance, or

P. Shakarian
Arizona Sate University, Tempe AZ, USA
e-mail: pshakari@asu.edu

G. I. Simari
Department of Computer Science, University of Oxford, United Kingdom
e-mail: gerardo.simari@cs.ox.ac.uk

G. Moores
Department of Electrical Engineering and Computer Science
U.S. Military Academy, West Point, NY, USA
e-mail: geoffrey.moores@usma.edu

S. Parsons
Dept. of Computer Science, University of Liverpool, Liverpool, United Kingdom
e-mail: s.d.parsons@liverpool.ac.uk

information theft. This is known as the “attribution problem” [18]. The difficulty of this problem stems not only from the amount of effort required to find forensic clues but also the ease with which an attacker can plant false clues to mislead security personnel. Further, while techniques such as forensics and reverse-engineering [3], source tracking [25], honeypots [23], and sinkholing [1] are commonly employed to find evidence that can lead to attribution, it is unclear how this evidence is to be combined and reasoned about. In a military setting, such evidence is augmented with normal intelligence collection, such as human intelligence (HUMINT), signals intelligence (SIGINT) and other means — this adds additional complications to the task of attributing a given operation. Essentially, cyber-attribution is a highly-technical intelligence analysis problem where an analyst must consider a variety of sources, each with its associated level of confidence, to provide a decision maker (e.g., a military commander) insight into who conducted a given operation.

As it is well known that people’s ability to conduct intelligence analysis is limited [9], and due to the highly technical nature of many cyber evidence-gathering techniques, an automated reasoning system would be best suited for the task. Such a system must be able to accomplish several goals, among which we distinguish the following main capabilities:

1. Reason about evidence in a formal, principled manner, i.e., relying on strong mathematical foundations.
2. Consider evidence for cyber attribution associated with some level of probabilistic uncertainty.
3. Consider logical rules that allow for the system to draw conclusions based on certain pieces of evidence and iteratively apply such rules.
4. Consider pieces of information that may not be compatible with each other, decide which information is most relevant, and express why.
5. Attribute a given cyber-operation based on the above-described features and provide the analyst with the ability to understand how the system arrived at that conclusion.

In this paper we present the InCA (Intelligent Cyber Attribution) framework, which meets all of the above qualities. Our approach relies on several techniques from the artificial intelligence community, including argumentation, logic programming, and probabilistic reasoning. We first outline the underlying mathematical framework and provide a running example based on real-world cases of cyber-attribution (cf. Section 2); then, in Sections 3 and 4, we formally present InCA and attribution queries, respectively. Finally, we discuss conclusions and future work in Section 6.

2 Two Kinds of Models

Our approach relies on *two separate models of the world*. The first, called the ***environmental model*** (EM) is used to describe the background knowledge and is proba-

EM	**AM**
"Malware X was compiled on a system using the English language."	"Malware X was compiled on a system English-speaking country Y."
"Malware W and malware X were created in a similar coding sytle."	"Malware W and malware X are related."
"Country Y and country Z are currently at war."	"Country Y has a motive to launch a cyber-attack against country Z."
"Country Y has a significant investment in math-science-engineering (MSE) education."	"Country Y has the capability to conduct a cyber-attack."

Fig. 1 Example observations — EM vs. AM.

bilistic in nature. The second one, called the ***analytical model*** (AM) is used to analyze competing hypotheses that can account for a given phenomenon (in this case, a cyber-operation). The EM *must be consistent* — this simply means that there must exist a probability distribution over the possible states of the world that satisfies all of the constraints in the model, as well as the axioms of probability theory. On the contrary, the AM will allow for contradictory information as the system must have the capability to reason about competing explanations for a given cyber-operation. In general, the EM contains knowledge such as evidence, intelligence reporting, or knowledge about actors, software, and systems. The AM, on the other hand, contains ideas the analyst concludes based on the information in the EM. Figure 1 gives some examples of the types of information in the two models. Note that an analyst (or automated system) could assign a probability to statements in the EM column whereas statements in the AM column can be true or false depending on a certain combination (or several possible combinations) of statements from the EM. We now formally describe these two models as well as a technique for *annotating* knowledge in the AM with information from the EM — these annotations specify the conditions under which the various statements in the AM can potentially be true.

Before describing the two models in detail, we first introduce the language used to describe them. Variable and constant symbols represent items such as computer systems, types of cyber operations, actors (e.g., nation states, hacking groups), and other technical and/or intelligence information. The set of all variable symbols is denoted with $\mathbf{V}$, and the set of all constants is denoted with $\mathbf{C}$. For our framework, we shall require two subsets of $\mathbf{C}$, $\mathbf{C}_{act}$ and $\mathbf{C}_{ops}$, that specify the actors that could conduct cyber-operations and the operations themselves, respectively. In the examples in this paper, we will use capital letters to represent variables (e.g., X,Y,Z). The constants in $\mathbf{C}_{act}$ and $\mathbf{C}_{ops}$ that we use in the running example are specified in the following example.

Example 1. The following (fictitious) actors and cyber-operations will be used in our examples:

$$\mathbf{C}_{act} = \{baja, krasnovia, mojave\} \tag{1}$$
$$\mathbf{C}_{ops} = \{worm123\} \tag{2}$$

$\mathbf{P}_{EM}$:	$origIP(M,X)$	Malware M originated from an IP address belonging to actor X.
	$malwInOp(M,O)$	Malware M was used in cyber-operation O.
	$mwHint(M,X)$	Malware M contained a hint that it was created by actor X.
	$compilLang(M,C)$	Malware M was compiled in a system that used language C.
	$nativLang(X,C)$	Language C is the native language of actor X.
	$inLgConf(X,X')$	Actors X and X' are in a larger conflict with each other.
	$mseTT(X,N)$	There are at least N number of top-tier math-science-engineering universities in country X.
	$infGovSys(X,M)$	Systems belonging to actor X were infected with malware M.
	$cybCapAge(X,N)$	Actor X has had a cyber-warfare capability for N years or less.
	$govCybLab(X)$	Actor X has a government cyber-security lab.
$\mathbf{P}_{AM}$:	$condOp(X,O)$	Actor X conducted cyber-operation O.
	$evidOf(X,O)$	There is evidence that actor X conducted cyber-operation O.
	$motiv(X,X')$	Actor X had a motive to launch a cyber-attack against actor X'.
	$isCap(X,O)$	Actor X is capable of conducting cyber-operation O.
	$tgt(X,O)$	Actor X was the target of cyber-operation O.
	$hasMseInvest(X)$	Actor X has a significant investment in math-science-engineering education.
	$expCw(X)$	Actor X has experience in conducting cyber-operations.

Fig. 2 Predicate definitions for the environment and analytical models in the running example.

■

The next component in the model is a set of predicate symbols. These constructs can accept zero or more variables or constants as arguments, and map to either *true* or *false*. *Note that the EM and AM use separate sets of predicate symbols* — however, they can share variables and constants. The sets of predicates for the EM and AM are denoted with $\mathbf{P}_{EM}, \mathbf{P}_{AM}$, respectively. In InCA, we require $\mathbf{P}_{AM}$ to include the binary predicate $condOp(X,Y)$, where X is an actor and Y is a cyber-operation. Intuitively, this means that actor X conducted operation Y. For instance, $condOp(baja, worm123)$ is true if *baja* was responsible for cyber-operation *worm123*. A sample set of predicate symbols for the analysis of a cyber attack between two states over contention of a particular industry is shown in Figure 2; these will be used in examples throughout the paper.

A construct formed with a predicate and constants as arguments is known as a *ground atom* (we shall often deal with ground atoms). The sets of all ground atoms for EM and AM are denoted with $\mathbf{G}_{EM}$ and $\mathbf{G}_{AM}$, respectively.

Example 2. The following are examples of ground atoms over the predicates given in Figure 2.

$$\begin{aligned}\mathbf{G}_{EM}: \ & \mathit{origIP}(\mathit{mw123sam1}, \mathit{krasnovia}),\\ & \mathit{mwHint}(\mathit{mw123sam1}, \mathit{krasnovia}),\\ & \mathit{inLgConf}(\mathit{krasnovia}, \mathit{baja}),\\ & \mathit{mseTT}(\mathit{krasnovia}, 2).\end{aligned}$$

$$\begin{aligned}\mathbf{G}_{AM}: \ & \mathit{evidOf}(\mathit{mojave}, \mathit{worm123}),\\ & \mathit{motiv}(\mathit{baja}, \mathit{krasnovia}),\\ & \mathit{expCw}(\mathit{baja}),\\ & \mathit{tgt}(\mathit{krasnovia}, \mathit{worm123}).\end{aligned}$$

■

For a given set of ground atoms, a *world* is a subset of the atoms that are considered to be true (ground atoms not in the world are false). Hence, there are $2^{|\mathbf{G}_{EM}|}$ possible worlds in the EM and $2^{|\mathbf{G}_{AM}|}$ worlds in the AM, denoted with $\mathcal{W}_{EM}$ and $\mathcal{W}_{AM}$, respectively.

Clearly, even a moderate number of ground atoms can yield an enormous number of worlds to explore. One way to reduce the number of worlds is to include *integrity constraints*, which allow us to eliminate certain worlds from consideration — they simply are not possible in the setting being modeled. Our principle integrity constraint will be of the form:

$$\mathsf{oneOf}(\mathscr{A}')$$

where $\mathscr{A}'$ is a subset of ground atoms. Intuitively, this says that any world where more than one of the atoms from set $\mathscr{A}'$ appear is invalid. Let $\mathbf{IC}_{EM}$ and $\mathbf{IC}_{AM}$ be the sets of integrity constraints for the EM and AM, respectively, and the sets of worlds that conform to these constraints be $\mathcal{W}_{EM}(\mathbf{IC}_{EM}), \mathcal{W}_{AM}(\mathbf{IC}_{AM})$, respectively.

Atoms can also be combined into formulas using standard logical connectives: conjunction (*and*), disjunction (*or*), and negation (*not*). These are written using the symbols $\wedge, \vee, \neg$, respectively. We say a world (w) *satisfies* a formula (f), written $w \models f$, based on the following inductive definition:

- if f is a single atom, then $w \models f$ iff $f \in w$;
- if $f = \neg f'$ then $w \models f$ iff $w \not\models f'$;
- if $f = f' \wedge f''$ then $w \models f$ iff $w \models f'$ and $w \models f''$; and
- if $f = f' \vee f''$ then $w \models f$ iff $w \models f'$ or $w \models f''$.

We use the notation $formula_{EM}, formula_{AM}$ to denote the set of all possible (ground) formulas in the EM and AM, respectively. Also, note that we use the notation $\top, \bot$ to represent tautologies (formulas that are true in all worlds) and contradictions (formulas that are false in all worlds), respectively.

2.1 Environmental Model

In this section we describe the first of the two models, namely the EM or environmental model. This model is largely based on the probabilistic logic of [14], which we now briefly review.

First, we define a *probabilistic formula* that consists of a formula f over atoms from $\mathbf{G}_{EM}$, a real number p in the interval $[0,1]$, and an error tolerance $\varepsilon \in [0, \min(p, 1-p)]$. A probabilistic formula is written as: $f : p \pm \varepsilon$. Intuitively, this statement is interpreted as "formula f is true with probability between $p-\varepsilon$ and $p+\varepsilon$" — note that we make no statement about the probability distribution over this interval. The uncertainty regarding the probability values stems from the fact that certain assumptions (such as probabilistic independence) may not be suitable in the environment being modeled.

Example 3. To continue our running example, consider the following set Π_{EM}:

$$
\begin{aligned}
f_1 &= govCybLab(baja) : 0.8 \pm 0.1\\
f_2 &= cybCapAge(baja, 5) : 0.2 \pm 0.1\\
f_3 &= mseTT(baja, 2) : 0.8 \pm 0.1\\
f_4 &= mwHint(mw123sam1, mojave) \wedge compilLang(worm123, english) : 0.7 \pm 0.2\\
f_5 &= malwInOp(mw123sam1, worm123)\\
&\quad \wedge malwareRel(mw123sam1, mw123sam2)\\
&\quad \wedge mwHint(mw123sam2, mojave) : 0.6 \pm 0.1\\
f_6 &= inLgConf(baja, krasnovia) \vee \neg cooper(baja, krasnovia) : 0.9 \pm 0.1\\
f_7 &= origIP(mw123sam1, baja) : 1 \pm 0
\end{aligned}
$$

Throughout other examples in the rest of the paper, we will make use of the subset $\Pi'_{EM} = \{f_1, f_2, f_3\}$. ■

We now consider a probability distribution Pr over the set $\mathcal{W}_{EM}(\mathbf{IC}_{EM})$. We say that Pr *satisfies* probabilistic formula $f : p \pm \varepsilon$ iff the following holds: $p - \varepsilon \leq \sum_{w \in \mathcal{W}_{EM}(\mathbf{IC}_{EM})} Pr(w) \leq p + \varepsilon$. A set Π_{EM} of probabilistic formulas is called a *knowledge base*. We say that a probability distribution over $\mathcal{W}_{EM}(\mathbf{IC}_{EM})$ *satisfies* Π_{EM} if and only if it satisfies all probabilistic formulas in Π_{EM}.

It is possible to create probabilistic knowledge bases for which there is no satisfying probability distribution. The following is a simple example of this:

$$
\begin{aligned}
&condOp(krasnovia, worm123) \vee condOp(baja, worm123) : 0.4 \pm 0;\\
&condOp(krasnovia, worm123) \wedge condOp(baja, worm123) : 0.6 \pm 0.1.
\end{aligned}
$$

Formulas and knowledge bases of this sort are *inconsistent*. In this paper, we assume that information is properly extracted from a set of historic data and hence consistent; (recall that inconsistent information can only be handled in the AM, not the EM). A consistent knowledge base could also be obtained as a result of curation by experts, such that all inconsistencies were removed — see [10, 17] for algorithms for learning rules of this type.

The main kind of query that we require for the probabilistic model is the *maximum entailment* problem: given a knowledge base Π_{EM} and a (non-probabilistic) formula q, identify p, ε such that all valid probability distributions Pr that satisfy Π_{EM} also satisfy $q : p \pm \varepsilon$, and there does not exist p', ε' s.t. $[p-\varepsilon, p+\varepsilon] \supset [p'-\varepsilon', p'+\varepsilon']$, where all probability distributions Pr that satisfy Π_{EM} also satisfy $q : p' \pm \varepsilon'$. That is, given q, can we determine the probability (with maximum tolerance) of statement q given the information in Π_{EM}? The approach adopted in [14] to solve this problem works as follows. First, we must solve the linear program defined next.

Definition 1 ($\mathsf{EM\text{-}LP\text{-}MIN}$). Given a knowledge base Π_{EM} and a formula q:

- create a variable x_i for each $w_i \in \mathscr{W}_{EM}(\mathsf{IC}_{EM})$;
- for each $f_j : p_j \pm \varepsilon_j \in \Pi_{EM}$, create constraint:

$$p_j - \varepsilon_j \leq \sum_{w_i \in \mathscr{W}_{EM}(\mathsf{IC}_{EM}) \text{ s.t. } w_i \models f_j} x_i \leq p_j + \varepsilon_j;$$

- finally, we also have a constraint:

$$\sum_{w_i \in \mathscr{W}_{EM}(\mathsf{IC}_{EM})} x_i = 1.$$

The objective is to minimize the function:

$$\sum_{w_i \in \mathscr{W}_{EM}(\mathsf{IC}_{EM}) \text{ s.t. } w_i \models q} x_i.$$

We use the notation $\mathsf{EP\text{-}LP\text{-}MIN}(\Pi_{EM}, q)$ to refer to the value of the objective function in the solution to the $\mathsf{EM\text{-}LP\text{-}MIN}$ constraints.

Let ℓ be the result of the process described in Definition 1. The next step is to solve the linear program a second time, but instead maximizing the objective function (we shall refer to this as $\mathsf{EM\text{-}LP\text{-}MAX}$) — let u be the result of this operation. In [14], it is shown that $\varepsilon = \frac{u-\ell}{2}$ and $p = \ell + \varepsilon$ is the solution to the maximum entailment problem. We note that although the above linear program has an exponential number of variables in the worst case (i.e., no integrity constraints), the presence of constraints has the potential to greatly reduce this space. Further, there are also good heuristics (cf. [10, 21]) that have been shown to provide highly accurate approximations with a reduced-size linear program.

Example 4. Consider KB Π'_{EM} from Example 3 and a set of ground atoms restricted to those that appear in that program. Hence, we have:

$$\begin{aligned}
w_1 &= \{govCybLab(baja), cybCapAge(baja,5), mseTT(baja,2)\}\\
w_2 &= \{govCybLab(baja), cybCapAge(baja,5)\}\\
w_3 &= \{govCybLab(baja), mseTT(baja,2)\}\\
w_4 &= \{cybCapAge(baja,5), mseTT(baja,2)\}\\
w_5 &= \{cybCapAge(baja,5)\}\\
w_6 &= \{govCybLab(baja)\}\\
w_7 &= \{mseTT(baja,2)\}\\
w_8 &= \emptyset
\end{aligned}$$

and suppose we wish to compute the probability for formula:

$$q = govCybLab(baja) \vee mseTT(baja,2).$$

For each formula in Π_{EM} we have a constraint, and for each world above we have a variable. An objective function is created based on the worlds that satisfy the query formula (here, worlds w_1—w_4, w_6, w_7). Hence, $\mathsf{EP\text{-}LP\text{-}MIN}(\Pi'_{EM}, q)$ can be written as follows:

$$\begin{array}{rcl}
\max & x_1 + x_2 + x_3 + x_4 + x_6 + x_7 & w.r.t.:\\
0.7 \leq & x_1 + x_2 + x_3 + x_6 & \leq 0.9\\
0.1 \leq & x_1 + x_2 + x_4 + x_5 & \leq 0.3\\
0.8 \leq & x_1 + x_3 + x_4 + x_7 & \leq 1\\
& x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 & = 1
\end{array}$$

We can now solve $\mathsf{EP\text{-}LP\text{-}MAX}(\Pi'_{EM}, q)$ and $\mathsf{EP\text{-}LP\text{-}MIN}(\Pi'_{EM}, q)$ to get solution 0.9 ± 0.1. ∎

2.2 Analytical Model

For the analytical model (AM), we choose a structured argumentation framework [15] due to several characteristics that make such frameworks highly applicable to cyber-warfare domains. Unlike the EM, which describes probabilistic information about the state of the real world, the AM must allow for competing ideas — it *must be able to represent contradictory information*. The algorithmic approach allows for the creation of *arguments* based on the AM that may "compete" with each other to describe who conducted a given cyber-operation. In this competition — known as a *dialectical process* — one argument may defeat another based on a *comparison criterion* that determines the prevailing argument. Resulting from this process, the InCA framework will determine arguments that are *warranted* (those that are not *defeated* by other arguments) thereby providing a suitable explanation for a given cyber-operation.

The transparency provided by the system can allow analysts to identify potentially incorrect input information and fine-tune the models or, alternatively, collect more information. In short, argumentation-based reasoning has been studied as a natural way to manage a set of inconsistent information — it is the way humans settle disputes. As we will see, another desirable characteristic of (structured) argumentation frameworks is that, once a conclusion is reached, we are left with an explanation of how we arrived at it and information about why a given argument is warranted; this is very important information for analysts to have. In this section, we recall some preliminaries of the underlying argumentation framework used, and then introduce the analytical model (AM).

Defeasible Logic Programming with Presumptions

DELP with Presumptions (PreDeLP) [13] is a formalism combining Logic Programming with Defeasible Argumentation. We now briefly recall the basics of PreDeLP; we refer the reader to [7, 13] for the complete presentation. The formalism contains several different constructs: facts, presumptions, strict rules, and defeasible rules. Facts are statements about the analysis that can always be considered to be true, while presumptions are statements that may or may not be true. Strict rules specify logical consequences of a set of facts or presumptions (similar to an implication, though not the same) that must always occur, while defeasible rules specify logical consequences that may be assumed to be true when no contradicting information is present. These constructs are used in the construction of *arguments*, and are part of a PreDeLP program, which is a set of facts, strict rules, presumptions, and defeasible rules. Formally, we use the notation $\Pi_{AM} = (\Theta, \Omega, \Phi, \Delta)$ to denote a PreDeLP program, where Ω is the set of strict rules, Θ is the set of facts, Δ is the set of defeasible rules, and Φ is the set of presumptions. In Figure 3, we provide an example Π_{AM}. We now describe each of these constructs in detail.

Facts (Θ) are ground literals representing atomic information or its negation, using strong negation "$\neg$". Note that all of the literals in our framework must be formed with a predicate from the set $\mathbf{P}_{AM}$. Note that information in this form cannot be contradicted.

Strict Rules (Ω) represent non-defeasible cause-and-effect information that resembles a material implication (though the semantics is different since the contrapositive does not hold) and are of the form $L_0 \leftarrow L_1, \ldots, L_n$, where L_0 is a ground literal and $\{L_i\}_{i>0}$ is a set of ground literals.

Presumptions (Φ) are ground literals of the same form as facts, except that they are not taken as being true but rather defeasible, which means that they can be contradicted. Presumptions are denoted in the same manner as facts, except that the symbol $-\!\!\prec$ is added. While any literal can be used as a presumption in InCA, we specifically require all literals created with the predicate *condOp* to be defeasible.

Defeasible Rules (Δ) represent tentative knowledge that can be used if nothing can be posed against it. Just as presumptions are the defeasible counterpart of facts, defeasible rules are the defeasible counterpart of strict rules. They are of the form $L_0 \prec L_1, \dots, L_n$, where L_0 is a ground literal and $\{L_i\}_{i>0}$ is a set of ground literals. Note that with both strict and defeasible rules, *strong negation* is allowed in the head of rules, and hence may be used to represent contradictory knowledge.

We note that strict rules and facts are necessary constructs as they may not be true in all environmental conditions. We shall discuss this further in the next section with the introduction of an annotation function.

Even though the above constructs are ground, we allow for schematic versions with variables that are used to represent sets of ground rules. We denote variables with strings starting with an uppercase letter; Figure 4 shows a non-ground example.

When a cyber-operation occurs, InCA must derive arguments as to who could have potentially conducted the action. Informally, an argument for a particular actor x conducting cyber-operation y is a consistent subset of the analytical model that entails the atom $condOp(x, y)$. If the argument contains only strict rules and facts, then it is *factual*. If it contains presumptions or defeasible rules, then it *defeasibly derives* that actor x conducted operation y.

Derivation follows the same mechanism of Logic Programming [12]. Since rule heads can contain strong negation, it is possible to defeasibly derive contradictory literals from a program. For the treatment of contradictory knowledge, PreDeLP incorporates a defeasible argumentation formalism that allows the identification of the pieces of knowledge that are in conflict, and through the previously mentioned *dialectical process* decides which information prevails as warranted.

This dialectical process involves the construction and evaluation of arguments that either support or interfere with a given query, building a *dialectical tree* in the process. Formally, we have:

Definition 2 (Argument). An *argument* $\langle \mathscr{A}, L \rangle$ for a literal L is a pair of the literal and a (possibly empty) set of the EM ($\mathscr{A} \subseteq \Pi_{AM}$) that provides a minimal proof for L meeting the requirements: (1.) L is defeasibly derived from $\mathscr{A}$, (2.) $\Omega \cup \Theta \cup \mathscr{A}$ is not contradictory, and (3.) $\mathscr{A}$ is a minimal subset of $\Delta \cup \Phi$ satisfying 1 and 2, denoted $\langle \mathscr{A}, L \rangle$.

Literal L is called the *conclusion* supported by the argument, and $\mathscr{A}$ is the *support* of the argument. An argument $\langle \mathscr{B}, L \rangle$ is a *subargument* of $\langle \mathscr{A}, L' \rangle$ iff $\mathscr{B} \subseteq \mathscr{A}$. An argument $\langle \mathscr{A}, L \rangle$ is *presumptive* iff $\mathscr{A} \cap \Phi$ is not empty. We will also use $\Omega(\mathscr{A}) = \mathscr{A} \cap \Omega$, $\Theta(\mathscr{A}) = \mathscr{A} \cap \Theta$, $\Delta(\mathscr{A}) = \mathscr{A} \cap \Delta$, and $\Phi(\mathscr{A}) = \mathscr{A} \cap \Phi$.

Note that our definition differs slightly from that of [22] where DeLP is introduced, as we include strict rules and facts as part of the argument. The reason for this will become clear in Section 3. Arguments for our scenario are shown in the following example.

Example 5. Figure 5 shows example arguments based on the knowledge base from Figure 3. Note that the following relationship exists:

$\langle \mathscr{A}_5, isCap(baja, worm123) \rangle$ is a sub-argument of
$\langle \mathscr{A}_2, condOp(baja, worm123) \rangle$ and
$\langle \mathscr{A}_3, condOp(baja, worm123) \rangle$. ■

$$
\begin{aligned}
\Theta :\ & \theta_{1a} &&= \mathit{evidOf}(\mathit{baja}, \mathit{worm123})\\
& \theta_{1b} &&= \mathit{evidOf}(\mathit{mojave}, \mathit{worm123})\\
& \theta_{2} &&= \mathit{motiv}(\mathit{baja}, \mathit{krasnovia})
\end{aligned}
$$

$$
\begin{aligned}
\Omega :\ & \omega_{1a} &&= \neg \mathit{condOp}(\mathit{baja}, \mathit{worm123}) \leftarrow \mathit{condOp}(\mathit{mojave}, \mathit{worm123})\\
& \omega_{1b} &&= \neg \mathit{condOp}(\mathit{mojave}, \mathit{worm123}) \leftarrow \mathit{condOp}(\mathit{baja}, \mathit{worm123})\\
& \omega_{2a} &&= \mathit{condOp}(\mathit{baja}, \mathit{worm123}) \leftarrow\\
& && \qquad \mathit{evidOf}(\mathit{baja}, \mathit{worm123}),\ \mathit{isCap}(\mathit{baja}, \mathit{worm123}),\\
& && \qquad \mathit{motiv}(\mathit{baja}, \mathit{krasnovia}),\ \mathit{tgt}(\mathit{krasnovia}, \mathit{worm123})\\
& \omega_{2b} &&= \mathit{condOp}(\mathit{mojave}, \mathit{worm123}) \leftarrow\\
& && \qquad \mathit{evidOf}(\mathit{mojave}, \mathit{worm123}),\ \mathit{isCap}(\mathit{mojave}, \mathit{worm123}),\\
& && \qquad \mathit{motiv}(\mathit{mojave}, \mathit{krasnovia}),\ \mathit{tgt}(\mathit{krasnovia}, \mathit{worm123})
\end{aligned}
$$

$$
\begin{aligned}
\Phi :\ & \phi_{1} &&= \mathit{hasMseInvest}(\mathit{baja}) \mathrel{-\!\!\prec}\\
& \phi_{2} &&= \mathit{tgt}(\mathit{krasnovia}, \mathit{worm123}) \mathrel{-\!\!\prec}\\
& \phi_{3} &&= \neg \mathit{expCw}(\mathit{baja}) \mathrel{-\!\!\prec}
\end{aligned}
$$

$$
\begin{aligned}
\Delta :\ & \delta_{1a} &&= \mathit{condOp}(\mathit{baja}, \mathit{worm123}) \mathrel{-\!\!\prec} \mathit{evidOf}(\mathit{baja}, \mathit{worm123})\\
& \delta_{1b} &&= \mathit{condOp}(\mathit{mojave}, \mathit{worm123}) \mathrel{-\!\!\prec} \mathit{evidOf}(\mathit{mojave}, \mathit{worm123})\\
& \delta_{2} &&= \mathit{condOp}(\mathit{baja}, \mathit{worm123}) \mathrel{-\!\!\prec} \mathit{isCap}(\mathit{baja}, \mathit{worm123})\\
& \delta_{3} &&= \mathit{condOp}(\mathit{baja}, \mathit{worm123}) \mathrel{-\!\!\prec} \mathit{motiv}(\mathit{baja}, \mathit{krasnovia}),\ \mathit{tgt}(\mathit{krasnovia}, \mathit{worm123})\\
& \delta_{4} &&= \mathit{isCap}(\mathit{baja}, \mathit{worm123}) \mathrel{-\!\!\prec} \mathit{hasMseInvest}(\mathit{baja})\\
& \delta_{5a} &&= \neg \mathit{isCap}(\mathit{baja}, \mathit{worm123}) \mathrel{-\!\!\prec} \neg \mathit{expCw}(\mathit{baja})\\
& \delta_{5b} &&= \neg \mathit{isCap}(\mathit{mojave}, \mathit{worm123}) \mathrel{-\!\!\prec} \neg \mathit{expCw}(\mathit{mojave})
\end{aligned}
$$

Fig. 3 A ground argumentation framework.

$$
\begin{aligned}
\Theta :\ & \theta_{1} &&= \mathit{evidOf}(\mathit{baja}, \mathit{worm123})\\
& \theta_{2} &&= \mathit{motiv}(\mathit{baja}, \mathit{krasnovia})
\end{aligned}
$$

$$
\begin{aligned}
\Omega :\ & \omega_{1} &&= \neg \mathit{condOp}(X, O) \leftarrow \mathit{condOp}(X', O),\ X \neq X'\\
& \omega_{2} &&= \mathit{condOp}(X, O) \leftarrow \mathit{evidOf}(X, O),\ \mathit{isCap}(X, O), \mathit{motiv}(X, X'),\ \mathit{tgt}(X', O), X \neq X'
\end{aligned}
$$

$$
\begin{aligned}
\Phi :\ & \phi_{1} &&= \mathit{hasMseInvest}(\mathit{baja}) \mathrel{-\!\!\prec}\\
& \phi_{2} &&= \mathit{tgt}(\mathit{krasnovia}, \mathit{worm123}) \mathrel{-\!\!\prec}\\
& \phi_{3} &&= \neg \mathit{expCw}(\mathit{baja}) \mathrel{-\!\!\prec}
\end{aligned}
$$

$$
\begin{aligned}
\Delta :\ & \delta_{1} &&= \mathit{condOp}(X, O) \mathrel{-\!\!\prec} \mathit{evidOf}(X, O)\\
& \delta_{2} &&= \mathit{condOp}(X, O) \mathrel{-\!\!\prec} \mathit{isCap}(X, O)\\
& \delta_{3} &&= \mathit{condOp}(X, O) \mathrel{-\!\!\prec} \mathit{motiv}(X, X'), \mathit{tgt}(X', O)\\
& \delta_{4} &&= \mathit{isCap}(X, O) \mathrel{-\!\!\prec} \mathit{hasMseInvest}(X)\\
& \delta_{5} &&= \neg \mathit{isCap}(X, O) \mathrel{-\!\!\prec} \neg \mathit{expCw}(X)
\end{aligned}
$$

Fig. 4 A non-ground argumentation framework.

Given argument $\langle \mathscr{A}_1, L_1 \rangle$, counter-arguments are arguments that contradict it. Argument $\langle \mathscr{A}_2, L_2 \rangle$ *counterargues* or *attacks* $\langle \mathscr{A}_1, L_1 \rangle$ literal L' iff there exists a sub-argument $\langle \mathscr{A}, L'' \rangle$ of $\langle \mathscr{A}_1, L_1 \rangle$ s.t. set $\Omega(\mathscr{A}_1) \cup \Omega(\mathscr{A}_2) \cup \Theta(\mathscr{A}_1) \cup \Theta(\mathscr{A}_2) \cup \{L_2, L''\}$ is contradictory.

$\langle \mathscr{A}_1, \mathit{condOp}(\mathit{baja}, \mathit{worm123})\rangle$	$\mathscr{A}_1 = \{\theta_{1a}, \delta_{1a}\}$
$\langle \mathscr{A}_2, \mathit{condOp}(\mathit{baja}, \mathit{worm123})\rangle$	$\mathscr{A}_2 = \{\phi_1, \phi_2, \delta_4, \omega_{2a}, \theta_{1a}, \theta_2\}$
$\langle \mathscr{A}_3, \mathit{condOp}(\mathit{baja}, \mathit{worm123})\rangle$	$\mathscr{A}_3 = \{\phi_1, \delta_2, \delta_4\}$
$\langle \mathscr{A}_4, \mathit{condOp}(\mathit{baja}, \mathit{worm123})\rangle$	$\mathscr{A}_4 = \{\phi_2, \delta_3, \theta_2\}$
$\langle \mathscr{A}_5, \mathit{isCap}(\mathit{baja}, \mathit{worm123})\rangle$	$\mathscr{A}_5 = \{\phi_1, \delta_4\}$
$\langle \mathscr{A}_6, \neg\mathit{condOp}(\mathit{baja}, \mathit{worm123})\rangle$	$\mathscr{A}_6 = \{\delta_{1b}, \theta_{1b}, \omega_{1a}\}$
$\langle \mathscr{A}_7, \neg\mathit{isCap}(\mathit{baja}, \mathit{worm123})\rangle$	$\mathscr{A}_7 = \{\phi_3, \delta_{5a}\}$

Fig. 5 Example ground arguments from Figure 3.

Example 6. Consider the arguments from Example 5. The following are some of the attack relationships between them: $\mathscr{A}_1$, $\mathscr{A}_2$, $\mathscr{A}_3$, and $\mathscr{A}_4$ all attack $\mathscr{A}_6$; $\mathscr{A}_5$ attacks $\mathscr{A}_7$; and $\mathscr{A}_7$ attacks $\mathscr{A}_2$. ∎

A *proper defeater* of an argument $\langle A, L\rangle$ is a counter-argument that — by some criterion — is considered to be better than $\langle A, L\rangle$; if the two are incomparable according to this criterion, the counterargument is said to be a *blocking* defeater. An important characteristic of PreDeLP is that the argument comparison criterion is modular, and thus the most appropriate criterion for the domain that is being represented can be selected; the default criterion used in classical defeasible logic programming (from which PreDeLP is derived) is *generalized specificity* [24], though an extension of this criterion is required for arguments using presumptions [13]. We briefly recall this criterion next — the first definition is for generalized specificity, which is subsequently used in the definition of presumption-enabled specificity.

Definition 3. Let $\Pi_{AM} = (\Theta, \Omega, \Phi, \Delta)$ be a PreDeLP program and let $\mathscr{F}$ be the set of all literals that have a defeasible derivation from Π_{AM}. An argument $\langle \mathscr{A}_1, L_1\rangle$ is *preferred to* $\langle \mathscr{A}_2, L_2\rangle$, denoted with $\mathscr{A}_1 \succ_{PS} \mathscr{A}_2$ iff the two following conditions hold:

1. For all $H \subseteq \mathscr{F}$, $\Omega(\mathscr{A}_1) \cup \Omega(\mathscr{A}_2) \cup H$ is non-contradictory: if there is a derivation for L_1 from $\Omega(\mathscr{A}_2) \cup \Omega(\mathscr{A}_1) \cup \Delta(\mathscr{A}_1) \cup H$, and there is no derivation for L_1 from $\Omega(\mathscr{A}_1) \cup \Omega(\mathscr{A}_2) \cup H$, then there is a derivation for L_2 from $\Omega(\mathscr{A}_1) \cup \Omega(\mathscr{A}_2) \cup \Delta(\mathscr{A}_2) \cup H$.
2. There is at least one set $H' \subseteq \mathscr{F}$, $\Omega(\mathscr{A}_1) \cup \Omega(\mathscr{A}_2) \cup H'$ is non-contradictory, such that there is a derivation for L_2 from $\Omega(\mathscr{A}_1) \cup \Omega(\mathscr{A}_2) \cup H' \cup \Delta(\mathscr{A}_2)$, there is no derivation for L_2 from $\Omega(\mathscr{A}_1) \cup \Omega(\mathscr{A}_2) \cup H'$, and there is no derivation for L_1 from $\Omega(\mathscr{A}_1) \cup \Omega(\mathscr{A}_2) \cup H' \cup \Delta(\mathscr{A}_1)$.

Intuitively, the principle of specificity says that, in the presence of two conflicting lines of argument about a proposition, the one that uses more of the available information is more convincing. A classic example involves a bird, Tweety, and arguments stating that it both flies (because it is a bird) and doesn't fly (because it is a penguin). The latter argument uses more information about Tweety — it is more specific — and is thus the stronger of the two.

Definition 4 ([13]). Let $\Pi_{AM} = (\Theta, \Omega, \Phi, \Delta)$ be a PreDeLP program. An argument $\langle \mathscr{A}_1, L_1 \rangle$ is *preferred to* $\langle \mathscr{A}_2, L_2 \rangle$, denoted with $\mathscr{A}_1 \succ \mathscr{A}_2$ iff any of the following conditions hold:

1. $\langle \mathscr{A}_1, L_1 \rangle$ and $\langle \mathscr{A}_2, L_2 \rangle$ are both factual arguments and $\langle \mathscr{A}_1, L_1 \rangle \succ_{PS} \langle \mathscr{A}_2, L_2 \rangle$.
2. $\langle \mathscr{A}_1, L_1 \rangle$ is a factual argument and $\langle \mathscr{A}_2, L_2 \rangle$ is a presumptive argument.
3. $\langle \mathscr{A}_1, L_1 \rangle$ and $\langle \mathscr{A}_2, L_2 \rangle$ are presumptive arguments, and

   a. $\neg(\Phi(\mathscr{A}_1) \subseteq \Phi(\mathscr{A}_2))$, or
   b. $\Phi(\mathscr{A}_1) = \Phi(\mathscr{A}_2)$ and $\langle \mathscr{A}_1, L_1 \rangle \succ_{PS} \langle \mathscr{A}_2, L_2 \rangle$.

Generally, if $\mathscr{A}, \mathscr{B}$ are arguments with rules X and Y, resp., and $X \subset Y$, then $\mathscr{A}$ is stronger than $\mathscr{B}$. This also holds when $\mathscr{A}$ and $\mathscr{B}$ use presumptions P_1 and P_2, resp., and $P_1 \subset P_2$.

Example 7. The following are relationships between arguments from Example 5, based on Definitions 3 and 4:

$\mathscr{A}_1$ and $\mathscr{A}_6$ are incomparable (blocking defeaters);
$\mathscr{A}_6 \succ \mathscr{A}_2$, and thus $\mathscr{A}_6$ defeats $\mathscr{A}_2$;
$\mathscr{A}_6 \succ \mathscr{A}_3$, and thus $\mathscr{A}_6$ defeats $\mathscr{A}_3$;
$\mathscr{A}_6 \succ \mathscr{A}_4$, and thus $\mathscr{A}_6$ defeats $\mathscr{A}_4$;
$\mathscr{A}_5$ and $\mathscr{A}_7$ are incomparable (blocking defeaters). ■

A sequence of arguments called an *argumentation line* thus arises from this attack relation, where each argument defeats its predecessor. To avoid undesirable sequences, that may represent circular or fallacious argumentation lines, in DELP an *argumentation line* is *acceptable* if it satisfies certain constraints (see [7]). A literal L is *warranted* if there exists a non-defeated argument $\mathscr{A}$ supporting L.

Clearly, there can be more than one defeater for a particular argument $\langle \mathscr{A}, L \rangle$. Therefore, many acceptable argumentation lines could arise from $\langle \mathscr{A}, L \rangle$, leading to a tree structure. The tree is built from the set of all argumentation lines rooted in the initial argument. In a dialectical tree, every node (except the root) represents a defeater of its parent, and leaves correspond to undefeated arguments. Each path from the root to a leaf corresponds to a different acceptable argumentation line. A dialectical tree provides a structure for considering all the possible acceptable argumentation lines that can be generated for deciding whether an argument is defeated. We call this tree *dialectical* because it represents an exhaustive dialectical analysis (in the sense of providing reasons for and against a position) for the argument in its root. For argument $\langle \mathscr{A}, L \rangle$, we denote its dialectical tree with $\mathscr{T}(\langle \mathscr{A}, L \rangle)$.

Given a literal L and an argument $\langle \mathscr{A}, L \rangle$, in order to decide whether or not a literal L is warranted, every node in the dialectical tree $\mathscr{T}(\langle \mathscr{A}, L \rangle)$ is recursively marked as "D" (*defeated*) or "U" (*undefeated*), obtaining a marked dialectical tree $\mathscr{T}^*(\langle \mathscr{A}, L \rangle)$ where:

- All leaves in $\mathscr{T}^*(\langle \mathscr{A}, L \rangle)$ are marked as "U"s, and

- Let $\langle \mathscr{B}, q\rangle$ be an inner node of $\mathscr{T}^*(\langle \mathscr{A}, L\rangle)$. Then, $\langle \mathscr{B}, q\rangle$ will be marked as "U" iff every child of $\langle \mathscr{B}, q\rangle$ is marked as "D". Node $\langle \mathscr{B}, q\rangle$ will be marked as "D" iff it has at least a child marked as "U".

Given argument $\langle \mathscr{A}, L\rangle$ over Π_{AM}, if the root of $\mathscr{T}^*(\langle \mathscr{A}, L\rangle)$ is marked "U", then $\mathscr{T}^*(\langle \mathscr{A}, h\rangle)$ *warrants* L and that L is *warranted* from Π_{AM}. (Warranted arguments correspond to those in the grounded extension of a Dung argumentation system [5].)

We can then extend the idea of a dialectical tree to a *dialectical forest*. For a given literal L, a dialectical forest $\mathscr{F}(L)$ consists of the set of dialectical trees for all arguments for L. We shall denote a marked dialectical forest, the set of all marked dialectical trees for arguments for L, as $\mathscr{F}^*(L)$. Hence, for a literal L, we say it is *warranted* if there is at least one argument for that literal in the dialectical forest $\mathscr{F}^*(L)$ that is labeled "U", *not warranted* if there is at least one argument for literal $\neg L$ in the forest $\mathscr{F}^*(\neg L)$ that is labeled "U", and *undecided* otherwise.

3 The InCA Framework

Having defined our environmental and analytical models (Π_{EM}, Π_{AM} respectively), we now define how the two relate, which allows us to complete the definition of our InCA framework.

The key intuition here is that given a Π_{AM}, every element of $\Omega \cup \Theta \cup \Delta \cup \Phi$ might only hold in certain worlds in the set $\mathscr{W}_{EM}$ — that is, worlds specified by the environment model. As formulas over the environmental atoms in set $\mathbf{G}_{EM}$ specify subsets of $\mathscr{W}_{EM}$ (i.e., the worlds that satisfy them), we can use these formulas to identify the conditions under which a component of $\Omega \cup \Theta \cup \Delta \cup \Phi$ *can be* true. Recall that we use the notation $formula_{EM}$ to denote the set of all possible formulas over $\mathbf{G}_{EM}$. Therefore, it makes sense to associate elements of $\Omega \cup \Theta \cup \Delta \cup \Phi$ with a formula from $formula_{EM}$. In doing so, we can in turn compute the probabilities of subsets of $\Omega \cup \Theta \cup \Delta \cup \Phi$ using the information contained in Π_{EM}, which we shall describe shortly. We first introduce the notion of *annotation function*, which associates elements of $\Omega \cup \Theta \cup \Delta \cup \Phi$ with elements of $formula_{EM}$.

We also note that, by using the annotation function (see Figure 6), we may have certain statements that appear as both facts and presumptions (likewise for strict and defeasible rules). However, these constructs would have different annotations, and thus be applicable in different worlds. Suppose we added the following presumptions to our running example:

$\phi_3 = evidOf(X, O) \mathrel{-\!\!\prec}$, and
$\phi_4 = motiv(X, X') \mathrel{-\!\!\prec}$.

Note that these presumptions are constructed using the same formulas as facts θ_1, θ_2. Suppose we extend *af* as follows:

$$\begin{aligned} af(\phi_3) &= malwInOp(M, O) \wedge malwareRel(M, M') \wedge mwHint(M', X) \\ af(\phi_4) &= inLgConf(Y, X') \wedge cooper(X, Y) \end{aligned}$$

$$\begin{aligned}
af(\theta_1) = \;& origIP(worm123, baja) \vee \big(malwInOp(worm123, o) \wedge \big(mwHint(worm123, baja) \vee \\
& (compilLang(worm123, c) \wedge nativLang(baja, c))\big)\big) \\
af(\theta_2) = \;& inLgConf(baja, krasnovia) \\
af(\omega_1) = \;& \mathsf{True} \\
af(\omega_2) = \;& \mathsf{True} \\
af(\phi_1) = \;& mseTT(baja, 2) \vee govCybLab(baja) \\
af(\phi_2) = \;& malwInOp(worm123, o') \wedge infGovSys(krasnovia, worm123) \\
af(\phi_3) = \;& cybCapAge(baja, 5) \\
af(\delta_1) = \;& \mathsf{True} \\
af(\delta_2) = \;& \mathsf{True} \\
af(\delta_3) = \;& \mathsf{True} \\
af(\delta_4) = \;& \mathsf{True} \\
af(\delta_5) = \;& \mathsf{True}
\end{aligned}$$

Fig. 6 Example annotation function.

So, for instance, unlike θ_1, ϕ_3 can potentially be true in any world of the form:

$$\{malwInOp(M, O), malwareRel(M, M'), mwHint(M', X)\}$$

while θ_1 cannot be considered in any those worlds.

With the annotation function, we now have all the components to formally define an InCA framework.

Definition 5 (InCA Framework). Given environmental model Π_{EM}, analytical model Π_{AM}, and annotation function af, $\mathscr{I} = (\Pi_{EM}, \Pi_{AM}, af)$ is an **InCA framework**.

Given the setup described above, we consider a *world-based* approach — the defeat relationship among arguments will depend on the current state of the world (based on the EM). Hence, we now define the status of an argument with respect to a given world.

Definition 6 (Validity). Given InCA framework $\mathscr{I} = (\Pi_{EM}, \Pi_{AM}, af)$, argument $\langle \mathscr{A}, L\rangle$ is valid w.r.t. world $w \in \mathscr{W}_{EM}$ iff $\forall c \in \mathscr{A}, w \models af(c)$.

In other words, an argument is valid with respect to w if the rules, facts, and presumptions in that argument are present in w — the argument can then be built from information that is available in that world. In this paper, we extend the notion of validity to argumentation lines, dialectical trees, and dialectical forests in the expected way (an argumentation line is valid w.r.t. w iff all arguments that comprise that line are valid w.r.t. w).

Example 8. Consider worlds $w_1, \ldots, w_8$ from Example 4 along with the argument $\langle \mathscr{A}_5, isCap(baja, worm123)\rangle$ from Example 5. This argument is valid in worlds w_1—w_4, w_6, and w_7. ■

We now extend the idea of a dialectical tree w.r.t. worlds — so, for a given world $w \in \mathscr{W}_{EM}$, the dialectical (resp., marked dialectical) tree induced by w is denoted by

$\mathscr{T}_w\langle\mathscr{A},L\rangle$ (resp., $\mathscr{T}^*_w\langle\mathscr{A},L\rangle$). We require that all arguments and defeaters in these trees to be valid with respect to w. Likewise, we extend the notion of dialectical forests in the same manner (denoted with $\mathscr{F}_w(L)$ and $\mathscr{F}^*_w(L)$, respectively). Based on these concepts we introduce the notion of *warranting scenario*.

Definition 7 (Warranting Scenario). Let $\mathscr{I} = (\Pi_{EM}, \Pi_{AM}, af)$ be an InCA framework and L be a ground literal over $\mathbf{G}_{AM}$; a world $w \in \mathscr{W}_{EM}$ is said to be a *warranting scenario* for L (denoted $w \vdash_{\mathsf{war}} L$) iff there is a dialectical forest $\mathscr{F}^*_w(L)$ in which L is warranted and $\mathscr{F}^*_w(L)$ is valid w.r.t w.

Example 9. Following from Example 8, argument $\langle\mathscr{A}_5, isCap(baja, worm123)\rangle$ is warranted in worlds w_3, w_6, and w_7. ■

Hence, the set of worlds in the EM where a literal L in the AM *must* be true is exactly the set of warranting scenarios — these are the "necessary" worlds, denoted:

$$nec(L) = \{w \in \mathscr{W}_{EM} \mid (w \vdash_{\mathsf{war}} L)\}.$$

Now, the set of worlds in the EM where AM literal L *can* be true is the following — these are the "possible" worlds, denoted:

$$poss(L) = \{w \in \mathscr{W}_{EM} \mid w \not\vdash_{\mathsf{war}} \neg L\}.$$

The following example illustrates these concepts.

Example 10. Following from Example 8:

$nec(isCap(baja, worm123)) = \{w_3, w_6, w_7\}$ and
$poss(isCap(baja, worm123)) = \{w_1, w_2, w_3, w_4, w_6, w_7\}$. ■

Hence, for a given InCA framework $\mathscr{I}$, if we are given a probability distribution Pr over the worlds in the EM, then we can compute an upper and lower bound on the probability of literal L (denoted $\mathbf{P}_{L,\mathsf{Pr},\mathscr{I}}$) as follows:

$$\ell_{L,\mathsf{Pr},\mathscr{I}} = \sum_{w \in nec(L)} \mathsf{Pr}(w),$$

$$u_{L,\mathsf{Pr},\mathscr{I}} = \sum_{w \in poss(L)} \mathsf{Pr}(w),$$

and

$$\ell_{L,\mathsf{Pr},\mathscr{I}} \leq \mathbf{P}_{L,\mathsf{Pr},\mathscr{I}} \leq u_{L,\mathsf{Pr},\mathscr{I}}.$$

Now let us consider the computation of probability bounds on a literal when we are given a knowledge base Π_{EM} in the environmental model, which is specified in $\mathscr{I}$, instead of a probability distribution over all worlds. For a given world $w \in \mathscr{W}_{EM}$, let $for(w) = \left(\bigwedge_{a \in w} a\right) \wedge \left(\bigwedge_{a \notin w} \neg a\right)$ — that is, a formula that is satisfied only by world w. Now we can determine the upper and lower bounds on the probability of a literal w.r.t. Π_{EM} (denoted $\mathbf{P}_{L,\mathscr{I}}$) as follows:

$$\ell_{L,\mathscr{I}} = \textsf{EP-LP-MIN}\left(\Pi_{EM}, \bigvee_{w \in nec(L)} for(w)\right),$$

$$u_{L,\mathscr{I}} = \textsf{EP-LP-MAX}\left(\Pi_{EM}, \bigvee_{w \in poss(L)} for(w)\right),$$

and

$$\ell_{L,\mathscr{I}} \leq \mathbf{P}_{L,\mathscr{I}} \leq u_{L,\mathscr{I}}.$$

Hence, we have:

$$\mathbf{P}_{L,\mathscr{I}} = \left(\ell_{L,\mathscr{I}} + \frac{u_{L,\mathscr{I}} - \ell_{L,\mathscr{I}}}{2}\right) \pm \frac{u_{L,\mathscr{I}} - \ell_{L,\mathscr{I}}}{2}.$$

Example 11. Following from Example 8, argument $\langle \mathscr{A}_5, isCap(baja, worm123)\rangle$, we can compute $\mathbf{P}_{isCap(baja,worm123),\mathscr{I}}$ (where $\mathscr{I} = (\Pi'_{EM}, \Pi_{AM}, af)$). Note that for the upper bound, the linear program we need to set up is as in Example 4. For the lower bound, the objective function changes to: $\min x_3 + x_6 + x_7$. From these linear constraints, we obtain: $\mathbf{P}_{isCap(baja,worm123),\mathscr{I}} = 0.75 \pm 0.25$. ■

4 Attribution Queries

We now have the necessary elements required to formally define the kind of queries that correspond to the attribution problems studied in this paper.

Definition 8. Let $\mathscr{I} = (\Pi_{EM}, \Pi_{AM}, af)$ be an InCA framework, $\mathscr{S} \subseteq \mathbf{C}_{act}$ (the set of "suspects"), $\mathscr{O} \in \mathbf{C}_{ops}$ (the "operation"), and $\mathrm{e} \subseteq \mathbf{G}_{EM}$ (the "evidence"). An actor $\mathsf{A} \in \mathscr{S}$ is said to be a *most probable suspect* iff there does not exist $\mathsf{A}' \in \mathscr{S}$ such that $\mathbf{P}_{condOp(\mathsf{A}',\mathscr{O}),\mathscr{I}'} > \mathbf{P}_{condOp(\mathsf{A},\mathscr{O}),\mathscr{I}'}$ where $\mathscr{I}' = (\Pi_{EM} \cup \Pi_{\mathrm{e}}, \Pi_{AM}, af')$ with Π_{e} defined as $\bigcup_{c \in \mathrm{e}} \{c : 1 \pm 0\}$.

Given the above definition, we refer to $Q = (\mathscr{I}, \mathscr{S}, \mathscr{O}, \mathrm{e})$ as an *attribution query*, and A as an *answer* to Q. We note that in the above definition, the items of evidence are added to the environmental model with a probability of 1 ± 0. While in general this may be the case, there are often instances in analysis of a cyber-operation where the evidence may be true with some degree of uncertainty. Allowing for probabilistic evidence is a simple extension to Definition 8 that does not cause any changes to the results of this paper.

To understand how uncertain evidence can be present in a cyber-security scenario, consider the following. In Symantec's initial analysis of the Stuxnet worm, they found the routine designed to attack the S7-417 logic controller was incomplete, and hence would not function [6]. However, industrial control system expert Ralph Langner claimed that the incomplete code would run provided a missing data block is generated, which he thought was possible [11]. In this case, though the code was incomplete, there was clearly uncertainty regarding its usability. This situation provides a real-world example of the need to compare arguments — in this case, in the worlds where both arguments are valid, Langner's argument would likely defeat Symantec's by generalized specificity (the outcome, of course, will depend on the exact formalization of the two). Note that Langner was later vindicated by the discovery of an older sample, Stuxnet 0.5, which generated the data block.[1]

InCA also allows for a variety of relevant scenarios to the attribution problem. For instance, we can easily allow for the modeling of non-state actors by extending the available constants — for example, traditional groups such as Hezbollah, which has previously wielded its cyber-warfare capabilities in operations against Israel [18]. Likewise, the InCA can also be used to model cooperation among different actors in performing an attack, including the relationship between non-state actors and nation-states, such as the potential connection between Iran and militants stealing UAV feeds in Iraq, or the much-hypothesized relationship between hacktivist youth groups and the Russian government [18]. Another aspect that can be modeled is deception where, for instance, an actor may leave false clues in a piece of malware to lead an analyst to believe a third party conducted the operation. Such a deception scenario can be easily created by adding additional rules in the AM that allow for the creation of such counter-arguments. Another type of deception that could occur include attacks being launched from a system not in the responsible party's area, but under their control (e.g., see [1]). Again, modeling who controls a given system can be easily accomplished in our framework, and doing so would simply entail extending an argumentation line. Further, campaigns of cyber-operations can also be modeled, as well as relationships among malware and/or attacks (as detailed in [2]).

As with all of these abilities, InCA provides the analyst the means to model a complex situation in cyber-warfare but saves him from carrying out the reasoning associated with such a situation. Additionally, InCA results are constructive, so an analyst can "trace-back" results to better understand how the system arrived at a given conclusion.

5 Open Questions

In this section we review some major areas of research to address to move InCA toward a deployed system.

[1] http://www.symantec.com/connect/blogs/stuxnet-05-disrupting-uranium-processing-natanz

5.1 Rule Learning

The InCA framework depends on logical rules and statements as part of the input, though there are existing bodies of work we can leverage (decision tree rule learning, inductive logic programming, etc.) there are some specific challenges with regard to InCA that we must account for, specifically:

- Quickly learning probabilistic rules from data received as an input stream
- Learning of the annotation function
- Identification of the diagnosticity of new additions to the knowledgebase
- Learning rules that combine multiple, disparate sources (i.e. malware analysis and PCAP files, for instance)

5.2 Belief Revision

Even though we allow for inconsistencies in the AM portion of the model, inconsistency can arise even with a consistent EM. In a companion paper, [19] we introduce the following notion of consistency.

Definition 9. InCA program $\mathscr{I} = (\Pi_{EM}, \Pi_{AM}, af)$, with $\Pi_{AM} = \langle \Theta, \Omega, \Phi, \Delta \rangle$, is *Type II consistent* iff: given any probability distribution Pr that satisfies Π_{EM}, if there exists a world $w \in \mathscr{W}_{EM}$ such that $\bigcup_{x \in \Theta \cup \Omega \,|\, w \models af(x)} \{x\}$ is inconsistent, then we have $\mathsf{Pr}(w) = 0$.

Thus, any EM world in which the set of associated facts and strict rules are inconsistent (we refer to this as “classical consistency”) must always be assigned a zero probability. The intuition is as follows: any subset of facts and strict rules are thought to be true under certain circumstances —- these circumstances are determined through the annotation function and can be expressed as sets of EM worlds. Suppose there is a world where two contradictory facts can both be considered to be true (based on the annotation function). If this occurs, then there must not exist a probability distribution that satisfies the program Π_{EM} that assigns such a world a non-zero probability, as this world leads to an inconsistency.

While we have studied this theoretically [19], several important challenges remain: How do different belief revision methods affect the results of attribution queries? In particular, can we develop tractable algorithms for belief revision in the InCA framework? Further, finding efficient methods for re-computing attribution queries following a belief revision operation is a related concern for future work.

5.3 Temporal Reasoning

Cyber-security data often has an inherent temporal component (in particular, PCAP files, system logs, and traditional intelligence). One way to represent this type of information in InCA is by replacing the EM with a probabilistic temporal logic (i.e. [8, 4, 17, 20]). However, even though this would be a relatively straightforward adjustment to the framework, it leads to several interesting questions, specifically:

- Can we identify hacking groups responsible for a series of incidents over a period of time (a cyber campaign)?
- Can we identify the group responsible for a campaign if it is not known a priori?
- Can we differentiate between multiple campaigns conducted by multiple culprits in time-series data?

5.4 Abductive Inference Queries

We may often have a case where more than one culprit is attributed to the same cyber-attack with nearly the same probabilities. In this case, can we identify certain evidence that, if found, can lead us to better differentiate among the potential culprits? In the intelligence community, this is often referred as identifying *intelligence gaps*. We can also frame this as an abductive inference problem [16]. This type of problems leads to several interesting challenges:

- Can we identify all pieces of diagnostic evidence that would satisfy an important intelligence gap?
- Can we identify diagnostic evidence under constraints (i.e., taking into account limitations on the type of evidence that can be collected)?
- In the case where a culprit is attributed with a high probability, can we identify evidence that can falsify the finding?

6 Conclusions

In this paper we introduced InCA, a new framework that allows the modeling of various cyber-warfare/cyber-security scenarios in order to help answer the attribution question by means of a combination of probabilistic modeling and argumentative reasoning. This is the first framework, to our knowledge, that addresses the attribution problem while allowing for multiple pieces of evidence from different sources, including traditional (non-cyber) forms of intelligence such as human intelligence. Further, our framework is the first to extend Defeasible Logic Programming with probabilistic information. Currently, we are implementing InCA along with the associated algorithms and heuristics to answer these queries.

Acknowledgments

This work was supported by UK EPSRC grant EP/J008346/1 – "PrOQAW", ERC grant 246858 – "DIADEM", by NSF grant #1117761, by the National Security Agency under the Science of Security Lablet grant (SoSL), Army Research Office project 2GDATXR042, and DARPA project R.0004972.001.

References

1. Shadows in the Cloud: Investigating Cyber Espionage 2.0. Tech. rep., Information Warfare Monitor and Shadowserver Foundation (2010)
2. APT1: Exposing one of China's cyber espionage units. Mandiant (tech. report) (2013)
3. Altheide, C.: Digital Forensics with Open Source Tools. Syngress (2011)
4. Dekhtyar, A., Dekhtyar, M.I., Subrahmanian, V.S.: Temporal probabilistic logic programs. In: ICLP 1999, pp. 109–123. The MIT Press, Cambridge, MA, USA (1999)
5. Dung, P.M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and *n*-person games. Artif. Intell. **77**, pp. 321–357 (1995)
6. Falliere, N., Murchu, L.O., Chien, E.: W32.Stuxnet Dossier Version 1.4. Symantec Corporation (2011)
7. García, A.J., Simari, G.R.: Defeasible logic programming: An argumentative approach. TPLP **4**(1-2), 95–138 (2004)
8. Hansson, H., Jonsson, B.: A logic for reasoning about time and probability. Formal Aspects of Computing **6**, 512–535 (1994)
9. Heuer, R.J.: Psychology of Intelligence Analysis. Center for the Study of Intelligence
10. Khuller, S., Martinez, M.V., Nau, D.S., Sliva, A., Simari, G.I., Subrahmanian, V.S.: Computing most probable worlds of action probabilistic logic programs: scalable estimation for $10^{30,000}$ worlds. AMAI **51(2–4)**, 295–331 (2007)
11. Langner, R.: Matching Langner Stuxnet analysis and Symantic dossier update. Langner Communications GmbH (2011)
12. Lloyd, J.W.: Foundations of Logic Programming, 2nd Edition. Springer (1987)
13. Martinez, M.V., García, A.J., Simari, G.R.: On the use of presumptions in structured defeasible reasoning. In: Proc. of COMMA, pp. 185–196 (2012)
14. Nilsson, N.J.: Probabilistic logic. Artif. Intell. **28**(1), 71–87 (1986)
15. Rahwan, I., Simari, G.R.: Argumentation in Artificial Intelligence. Springer (2009)
16. Reggia, J.A., Peng, Y.: Abductive inference models for diagnostic problem-solving. Springer-Verlag New York, Inc., New York, NY, USA (1990)
17. Shakarian, P., Parker, A., Simari, G.I., Subrahmanian, V.S.: Annotated probabilistic temporal logic. TOCL **12**(2), 14 (2011)
18. Shakarian, P., Shakarian, J., Ruef, A.: Introduction to Cyber-Warfare: A Multidisciplinary Approach. Syngress (2013)
19. Shakarian, P., Simari, G.I., Falappa, M.A.: Belief revision in structured probabilistic argumentation. In: Proceedings of FoIKS, pp. 324–343 (2014)
20. Shakarian, P., Simari, G.I., Subrahmanian, V.S.: Annotated probabilistic temporal logic: Approximate fixpoint implementation. ACM Trans. Comput. Log. **13**(2), 13 (2012)
21. Simari, G.I., Martinez, M.V., Sliva, A., Subrahmanian, V.S.: Focused most probable world computations in probabilistic logic programs. AMAI **64**(2-3), 113–143 (2012)
22. Simari, G.R., Loui, R.P.: A mathematical treatment of defeasible reasoning and its implementation. Artif. Intell. **53**(2-3), 125–157 (1992)
23. Spitzner, L.: Honeypots: Catching the Insider Threat. In: Proc. of ACSAC 2003, pp. 170–179. IEEE Computer Society (2003)

24. Stolzenburg, F., García, A., Chesñevar, C.I., Simari, G.R.: Computing Generalized Specificity. Journal of Non-Classical Logics **13**(1), 87–113 (2003)
25. Thonnard, O., Mees, W., Dacier, M.: On a multicriteria clustering approach for attack attribution. SIGKDD Explorations **12**(1), 11–20 (2010)