An Answer Set Programming Framework for Reasoning about Agents' Beliefs and Truthfulness of Statements

Marcello Balduccini[1] , **Michael Gelfond**[2] , **Enrico Pontelli**[3] , **Tran Cao Son**[3]
[1]Saint Joseph's University
[2]Texas Tech University
[3]New Mexico State University
mbalducc@sju.edu, michael.gelfond@ttu.edu, {epontell, tson}@cs.nmsu.edu

Abstract

The paper proposes a framework for capturing how an agent's beliefs evolve over time in response to observations and for answering the question of whether statements made by a third party can be believed. The basic components of the framework are a formalism for reasoning about actions, changes, and observations and a formalism for default reasoning. The paper describes a concrete implementation that leverages answer set programming for determining the evolution of an agent's "belief state", based on observations, knowledge about the effects of actions, and a theory about how these influence an agent's beliefs. The beliefs are then used to assess whether statements made by a third party can be accepted as truthful. The paper investigates an application of the proposed framework in the detection of man-in-the-middle attacks targeting computers and cyber-physical systems. Finally, we briefly discuss related work and possible extensions.

1 Introduction

Artificial Intelligence research, in the area of knowledge-based and intelligent agents, has been progressing at a rapid pace, making it possible to develop agents that can assist and/or replace humans in several tasks. Organizations use web-bots for interacting with clients in various capacities, e.g., in providing information about the company or making offers. This trend continues to be fueled by the ubiquitous use of online resources.

Unfortunately, not every business on the Internet is as honest as one would hope. Stories about businesses that cheat people of goods or services, or falsely advertise their services are not uncommon. Indeed, lying and misrepresentations are more widespread in online negotiations than offline ones (Rowe 2006). Dishonesty is also present in the news media, leading to the dissemination of untruthful stories (e.g., the 'Phuc Dat Bich' story[1] reported by BBC and Sky News). Another example is the case of a hotel reviewer with the alias *"AmishBoy"* who gave high ratings to different hotels in the same day (Minnich et al. 2015)—thus suggesting that his statements may be untrustworthy.

The concern about misinformation, disinformation, or dishonesty is also reflected in the development of resources that allow people to rate companies (e.g., expedia.com, yelp.com, *Angie's List*) or defend the reputation of a company or an entity (e.g., reputation.com) or check for the accuracy of some information disseminated by a person or a new outlet (e.g., factcheck.org). All these aim at providing users with tools for checking the trustworthiness of a piece of information.

The above development highlights the problem of *understanding whether statements made by an agent can be accepted as truthful with respect to a context composed of observations,knowledge about default behavior, and corresponding beliefs.* This problem is not new to human. It is indeed as old as the existence of human on the earth. Yet, the amount of information that a human is receiving everyday has gone up significantly since the earth becomes flat, with the introduction of the Internet, and the advances in mobile technologies. This requires, ideally, tools that automatically filter out untrustworthy information for an user. The development of such a tool, on the other hand, requires the understanding of the reasoning process that helps a human to conclude that a piece of information receiving from a third party is true or false.

In this paper, we are interested in reasoning about the "belief state" of an agent and what that tells us about the truthfulness of statements made by a third party. We assume that we can observe the world as well as other agents' actions. The basis for our judgments will be composed of our observations, performed on a linear time line, along with commonsense knowledge about the world and about how observations and knowledge influence an agent's beliefs. We assume that observations are *true* at the time they are made, and will stay true until additional pieces of information prove otherwise. The approach is designed to reflect the fact that an agent's beliefs are subjective – they might not correspond to the ground truth and could change over time. This is because judgments are often made in the presence of incomplete information. This makes reasoning about beliefs and truthfulness of statements *non-monotonic*. Note that, in this paper, judgments about a statement are made *independently* of whether or not we trust the agent from whom the statement originated. As such, our study in this paper differs from the extensive literature about trust models between agents (see, e.g., (Artz and Gil 2007; Sabater and Sierra 2005)). We will elaborate further on this issue in the related work section.

[1] http://gizmodo.com/phuc-dat-bich-is-a-massive-phucking-faker-1744588099

Consider the following motivating example.

Example 1. *When we first met at an event (time t_0), John stated that he comes from a poor family. Later, we learn that John attends a private college, renowned for its expensive tuition (time t_1). Later yet, we learn that John attends the college thanks to a scholarship awarded due to his financial hardship (time t_2).*

This story spans over three time instants: t_0, t_1, and t_2. During the story, we learn (through observations) some facts, which affect our beliefs about the world as follows:

- *Time t_0: John makes the statement that his family is poor (property poor is true). No observations are available to us and thus we do not know if John's family is indeed poor or not. We have no reasons to conclude that John's statement is not truthful.*
- *Time t_1: We observe that John attends an expensive college (property in_college is true).* Normally, *we believe that someone attending an expensive college is from a rich family (default d_1). Hence, our belief at this point is that John is rich. This prevents us from accepting John's statement as truthful. We indicate that the default d_1 is the reason to draw such conclusion.*
- *Time t_2: We observe that John has a need-based scholarship (property has_scholarship is true). It is our belief that a student's need-based scholarship is* usually *derived from the family's financial situation (default d_2). Because both defaults are applicable and in conflict with each other, we believe neither that John's family is rich, nor that it is poor. This allows us to withdraw our previous conclusion that John's statement could not be accepted as truthful.*
- *The situation might be different if we had a preference between our defaults. For instance, if we were inclined to favor d_2 over d_1, then we would believe that John's family is poor. This may strengthen our conclusion that John's statement can be accepted as truthful.*

The main contributions of this paper can be summarized as follows:

1. The formalization of an abstract model to represent and reason about the evolution of an agent's beliefs over time and to draw conclusions about whether third-party statements can be accepted as truthful;
2. A concrete realization of the model using Answer Set Programming; and
3. A demonstration of the proposed framework on an important problem in the area of cyber security.

We illustrate the framework using examples and discuss possible extensions that need to be considered.

2 Beliefs and Truthfulness of Statements

In this section, we propose a general framework for representing, and reasoning about, the beliefs held by an agent. The framework also establishes a link between such beliefs and whether the agent may accept as truthful statements made by a third party. The framework can be instantiated using specific paradigms for reasoning about actions and change and for non-monotonic reasoning. We build upon the following foundations:

- An *action signature* $\langle F, B, A\rangle$ is given, where F are symbols for fluents (properties of the world whose truth value may change over time), B are symbols for an agent's beliefs and A are symbols for actions. The components of the signatures are pairwise disjoint.[2] The notions of fluent literals and of set Σ of states of the domain are introduced as usual.
- It is possible to observe the properties of the world and the occurrences of the actions over time (e.g., we observe that John buys a car, John is a student, etc.). Let H^n be the set of all observations up to step n and all recorded action occurrences up to step $n-1$.
- We have adequate knowledge about actions and their effects (e.g., the action of buying a car requires that the agent has money and its execution will result in the agent owning a car). This knowledge is represented by an action theory Act in a suitable logic that allows reasoning about actions' effects and consequent changes to the world. Let $\Phi_{Act} : \Sigma \times A \to 2^{\Sigma}$ denote a suitable transition function describing the evolution of the dynamic system of interest. $\hat{\Phi}_{Act}$ is the usual extension of Φ_{Act} to action sequences with $\hat{\Phi}_{Act}(\sigma, []) = \sigma$ and $\hat{\Phi}_{Act}(\sigma, [a;\beta]) = \bigcup_{\sigma' \in \Phi_{Act}(a,\sigma)} \hat{\Phi}_{Act}(\sigma', \beta)$ where $\sigma \in \Sigma$ and β is an action sequence and a is executable in σ.
- We have commonsense knowledge about how an agent of interest forms its beliefs (e.g., an agent may believe that a person attending an expensive school normally comes from a rich family, a person obtaining need-based scholarship usually comes from a poor family). This knowledge is represented by a default theory with preferences Def, that enables reasoning about the state of the agent's beliefs. Let us denote with $\models_D$ the entailment relation defined over the default theory framework defining Def.

The set of observations $O_w \subseteq H^n$ in Example 1 includes the observations such as

- 'John attends an expensive college' at time point t_1, and
- 'John receives a need-based scholarship' at time point t_2.

In this particular example we do not have any action occurrences.[3] Our default theory Def consists of defaults d_1 and d_2 stated earlier, which allow us to draw conclusions regarding our agent's beliefs.

Let us consider a theory $T = (H^n, Act, Def)$ and the associated Φ_{Act} and $\models_D$. First, we are interested in defining the beliefs held by the agent over time. And, then, we are interested in using this information for answering the question of whether a statement $b[t]$, where $b \in B$ and t is the time step at which the statement was made, can be accepted as truthful. For this, we define the entailment relation $\models$ between T and $b[t]$. Intuitively, the process entails the steps below:

[2] This requirement is imposed to simplify some aspects of the presentation, but is not essential.

[3] Some frameworks for reasoning about actions and change may require the introduction of a NOP action for modeling the example.

- Compute $\hat{\Phi}_{Act}(H^n, \Sigma_0)$, the set of states that are reachable from the initial state given the history H^n, where Σ_0 denotes the initial state specified by *Act* and H^n, which can be incomplete. For compactness, below we abbreviate $\hat{\Phi}_{Act}(H^n, \Sigma_0)$ with $W[n]$;
- Determine the agent's beliefs by finding which propositions are derivable from *Def* and $W[t]$ (using $\models_D$). We note that $W[t]$ can be a set of states and the different notions of entailment such as *skeptical* and *credulous* entailment could be applied here;
- Determine if the statement $b[t]$ is consistent with respect to the agent's beliefs.

The entailment relation between T and $b[t]$ can thus be defined by:

$$T \models b[t] \Leftrightarrow \langle W[t], Def \rangle \models_D b \tag{1}$$

Note that this definition also allows one to identify elements of H^t that, when obtained, will result in the confirmation or denial of $T \models b[t]$. As such, a system that obeys (1) can also be used by users who are interested in what they need to do in order to believe in a statement about b at the time step t, given their beliefs about the behavior of the observed agents.

To develop a concrete system for reasoning about truthfulness of agents using (1), specific formalizations of *Act* and *Def* need to be developed. There is a large body of research related to these two areas, and deciding which one to use depends on the system developer. Well-known formalisms for reasoning about actions and change, such as action languages (Gelfond and Lifschitz 1998), situation calculus (Reiter 2001), etc., can be employed for *Act* and Φ_{Act}. Approaches to default reasoning with preferences, such as those proposed in (Brewka and Eiter 1999; Brewka and Eiter 2000; Delgrande, Schaub, and Tompits 2003; Gelfond and Son 1998)), can be used for *Def* (and $\models_D$). In addition, let us note that, in the literature, $\models_D$ can represent *skeptical* or *credulous* reasoners; and the model does not specify how observations are collected. Deciding which type of reasoning is suitable or how to collect observations is an important issue, but it is application-dependent and beyond the scope of this paper.

3 Reasoning about Beliefs and Truthfulness Using Answer Set Programming

We will now present a concrete system for reasoning about truthfulness of agents using *Answer Set Programming (ASP)* (Marek and Truszczyński 1999; Gelfond and Lifschitz 1991). We select ASP as the host language since there exist several approaches to reasoning about actions and changes as well as default reasoning using ASP. This enables the seamless integration of the two steps involved in (1) into a single system. Several efficient ASP solvers are available, that enable the use of the proposed system in practical applications. In particular, we will use answer sets (and projections of answer sets over specific time points) to capture $W[t]$, and use established encodings of action theories and defaults in ASP.

3.1 Background: Answer Set Programming

A logic program Π is a set of rules of the form

$$a_0 \leftarrow a_1, \ldots, a_m, \mathit{not}\ a_{m+1}, \ldots, \mathit{not}\ a_n \tag{2}$$

or

$$\bot \leftarrow a_1, \ldots, a_m, \mathit{not}\ a_{m+1}, \ldots, \mathit{not}\ a_n \tag{3}$$

where $0 \leq m \leq n$, each a_i is an atom of a first-order language $\mathscr{P}$, $\bot$ is a special symbol denoting the truth value *false*, and *not* is a connective called *negation-as-failure*. A negation as failure literal (or naf-literal) is of the form *not a* where a is an atom. For a rule of the form (2)-(3), the left and right hand side of the rule are called the *head* and the *body*, respectively. A rule of the form (3) is also called a constraint.

Given a logic program Π, we will assume that each rule in Π is replaced by the set of its ground instances so that all atoms in Π are ground. Consider a set of ground atoms X. The body of a rule of the form (2) or (3) is satisfied by X if $\{a_{m+1}, \ldots, a_n\} \cap X = \emptyset$ and $\{a_1, \ldots, a_m\} \subseteq X$. A rule of the form (2) is satisfied by X if either its body is not satisfied by X or $a_0 \in X$. A rule of the form (3) is satisfied by X if its body is not satisfied by X. An atom a is supported by X if a is the head of some rule of the form (2) whose body is satisfied by X.

For a set of ground atoms S and a program Π, the reduct of Π with respect to S, denoted by Π^S, is the program obtained from the set of all ground instances of Π by deleting

1. each rule that has a naf-literal *not a* in its body with $a \in S$, and
2. all naf-literals in the bodies of the remaining clauses.

S is an *answer set (or a stable model)* of Π if it satisfies the following conditions.

1. If Π does not contain any naf-literal (i.e. $m = n$ in every rule of Π) then S is the smallest set of atoms that satisfies all the rules in Π.
2. If the program Π does contain some naf-literal ($m < n$ in some rule of Π), then S is an answer set of Π if S is the answer set of Π^S. (Note that Π^S does not contain naf-literals, its answer set is defined in the first item.)

A program Π is said to be *consistent* if it has an answer set. Otherwise, it is inconsistent.

Several extensions (e.g., *choice atoms*, *aggregates*, etc.) have been introduced to simplify the use of ASP. We will use and explain them whenever it is needed. Answer sets can be computed using answer set solvers. Extended syntactic notations have been proposed to facilitate the encoding of commonly used patterns (e.g., aggregates). In this paper, we use the syntax of ASP used in CLASP[4] or DLV[5].

3.2 Reasoning with No Observations about Action Occurrences

Let us start with a simpler case—where we do not worry about action occurrences. We represents the components of the theory by means of keywords *stm* (for *statement*), *obs*

[4] https://github.com/potassco

[5] http://www.dlvsystem.com/

(for *observed*), *rule*, $default$, and $prefer$. Fluent literals are defined as usual. Similarly, symbols for beliefs yield the notion of belief literal. By literals we mean fluent literals or belief literals. $\bar{l}$ denotes the complement of literal l. Agents can make statements about belief literals. Statements are given in the form $stm(b,s)$ where s is a non-negative integer (representing a time step) and b is a belief literal. For example, the statement 'John said that he is poor' can be stated as $stm(poor,0)$.

Definition 1. *A* knowledge base (KB) *over an action signature $\langle F,B,A\rangle$, is a tuple $\langle O,RD\rangle$ where:*

- *O is a set of observations, each of the form*

$$obs(l,s) \tag{4}$$

where l is a fluent literal and s is a non-negative integer;

- *RD is a collection of expressions of the form*

$$rule(r,b,body) \tag{5}$$

$$default(d,b,body) \tag{6}$$

$$prefer(d,d',body) \tag{7}$$

where r, d and d' are constants that do not occur in the signature, b is a belief literal and body is a set of literals.

Statements (5), (6), and (7) are referred to as a *rule*, a *default*, and a *preference* between defaults, respectively.

Intuitively, r and d are used to name rules and defaults. We say that a set of observations O is *consistent* if, for each time step s and fluent literal l, $\{obs(l,s),obs(\neg l,s)\} \not\subseteq O$. Following the conventional approach from the literature, we make the assumptions that the preference relation in RD is an *acyclic preference* and that it is a transitive and irreflexive relation, i.e., there exists no sequence of preference statements $prefer(d_i,d_{i+1},body_i)$, $i = 1,\ldots,k$, in RD such that $d_1 = d_{k+1}$ and $\bigcup_{i=1}^{k} body_i$ is consistent.

A set of rules RD is *consistent* if, for any consistent set of fluent literals X, the logic program $\{b \leftarrow body \mid rule(r,b,body) \in RD\} \cup \{l \leftarrow \mid l \in X\}$ is consistent.

Definition 2. *A $KB = \langle O,RD\rangle$ is* consistent *if O and RD are consistent.*

Our goal is to identify the agent's beliefs and to use them to determine whether a statement $stm(p,s)$ is true at a time step t given KB. We will develop a program $\Pi(KB)$ to answer this question. We assume a finite horizon of steps, denoted by $s(0),\ldots,s(k)$. We use $h(L,S)$ to encode that fluent literal L is true at step S. We use $believes(B,S)$ to state that the agent believes B at step S. Furthermore, to simplify the use of $\Pi(KB)$ with current ASP solvers, we encode a rule of the form (5) using **(a)** the ASP atom $rule(r,head,body\_id_r)$ and **(b)** the set of ASP atoms $\{mbr(l,body\_id_r) \mid l \in body_r\}$, where *body_id* is an identifier for the body of the rule and *mbr* stands for '*member_of*'; similar encodings are used for (6) and (7). This method of encoding is similar to what is used in answer set planning and allows for the separation between the encoding of the problem and the rules for reasoning with it. The main challenge to be addressed lies in the interaction between observations and defaults. We apply the following principles:

(O)ptimistic: The most recent observation reflects the true state of the world;

(SK)eptical: When conflicting conclusions can be drawn then do not believe in any.

The first part of $\Pi(KB)$ contains the set of facts encoding O and RD as described above. Their semantics is captured by the following rules.[6]

The first set of rules is used to reason about observations of the form (4):

$$nr(L,U,S) \leftarrow s(U),neg(L,L_1),obs(L_1,V),s(S), U<V,V<S. \tag{8}$$

$$h(L,S) \leftarrow obs(L,S_1),s(S),S_1 \leq S, not\ nr(L,S_1,S). \tag{9}$$

These rules encode the principle **(O)**. Rule (8) states that $nr(L,U,S)$ is true if there exists a more recent observation of $\neg L$. Rule (9) indicates that an observation stays true if every conflicting observation is 'older'.

For reasoning about statements of the form (5), we have[7]:

$$\begin{aligned} h\_body(M,S) \leftarrow\ & s(S), \\ & N = \#count\{L : mbr(L,M)\}, \\ & N_h = \#count\{L : mbr(L,M), h(L,S)\}, \\ & N_b = \#count\{L : mbr(L,M), believes(L,S)\}, \\ & N_h + N_b == N. \end{aligned} \tag{10}$$

$$believes(H,S) \leftarrow s(S),rule(R,H,M), h\_body(M,S) \tag{11}$$

(10) defines an auxiliary predicate *h_body* where for each set M of literals and time step S, $h\_body(M,S)$ is true when all literals in M are true or believed at S. The '==' symbol denotes identity at the level of ground terms.[8] Checking whether the body is satisfied is compactly achieved by leveraging the #*count* aggregate of the ASP Core 2 language[9], which calculates the cardinality of a set. For instance, the expression $\#count\{L : mbr(L,M),h(L,S)\}$ finds the number of literals that are in the body of the rule ($mbr(L,M)$) and hold at time step S ($h(L,S)$). The combination of aggregates present in (11) checks that the number of literals that hold or are believed equals the size of the body of the rule.[10] Intuitively, (11) states that, if the body of a *rule* statement is satisfied, then its head must be true. Recall that, according to the syntax given earlier, the head is a belief – hence the expression $believes(H,S)$ in (11).

[6] In the following, The predicate *neg* encodes the complement of a literal, i.e, $neg(f,\neg f)$ and $neg(\neg f,f)$ are facts of the program.

[7] We would like to thank the anonymous reviewer for the suggestion on how to encode this rule for better performance.

[8] The variables involved are replaced by a constant during the grounding process and the identity is checked during the solving phase.

[9] Details can be found in (ASP Standardization Working Group 2013)

[10] Recall that fluent symbols and belief symbols are disjoint.

For reasoning about defaults of the form (6) we have:

$$app(D,H,S) \leftarrow s(S), default(D,H,M), h\_body(M,S) \quad (12)$$

$$defeated(D,H,S) \leftarrow s(S), app(D,H,S), neg(H,H_1), h(H_1,S). \quad (13)$$

$$defeated(D,H,S) \leftarrow s(S), app(D,H,S), neg(H,H_1), believes(H_1,S). \quad (14)$$

$$defeated(D,H,S) \leftarrow s(S), app(D,H,S), neg(H,H_1), app(D_1,H_1,S), prefer(D_1,D,S). \quad (15)$$

$$ab(D,H,S) \leftarrow s(S), app(D,H,S), neg(H,H_1), app(D_1,H_1,S), not\ defeated(D_1,H_1,S). \quad (16)$$

$$believes(H,S) \leftarrow app(D,H,S), not\ ab(D,H,S), not\ defeated(D,H,S). \quad (17)$$

The rule for *app* defines when a default is *applicable*, i.e., when its body is satisfied. The interaction between defaults and rules in the knowledge base are dealt with using rules for *ab* and *defeated*. The first rule for *defeated* dismisses the applicability of a default if the complement of its conclusion is already established. The second rule for *defeated* expresses that a default is defeated if there is a preferred default with a conflicting conclusion that is applicable. The rule for *ab* enforces the principle **(SK)**. It states that a default d should be blocked if there is another default with the conflicting conclusion (h' in $neg(h,h')$), which is not defeated. Finally, the last rule enforces the application of any default that is applicable and not otherwise blocked.

For reasoning about preferences of the form (7), we employ the rules:

$$prefer(D_1,D_2,S) \leftarrow s(S), prefer(D_1,D_2,M), h\_body(M,S). \quad (18)$$

This rule defines when a preference among two defaults can be applied.

In summary, for a $KB = \langle O, RD\rangle$, $\Pi(KB)$ consists of rules (9)–(18) and the set of facts encoding *RD* and *O*.

Proposition 1. *For each consistent KB, there exists no answer set of $\Pi(KB)$ that contains $believes(l,t)$ and $believes(\bar{l},t)$ for some literal l and time step t.*

Proof. The proof of this proposition relies on the fact that $believes(l,t)$ is in an answer set A if the logic program rule (11) (or (17)) is satisfied by A. If a rule (11) is satisfied by A then a KB contains a rule $rule(r,l,body)$ and $body$ is satisfied by A. Because of the consistency of KB, we know that there exists no rule $rule(r',\bar{l},body')$ such that $body'$ is satisfied by A. For, otherwise, KB is inconsistent. This will imply that for every default $default(d,\bar{l},body')$, if the default is applicable, then the body of the rule (14) is satisfied and therefore $defeated(d)$ belongs to A. Consequently, the body of (17) is not satisfied for the default d, and hence, $believes(\bar{l},t)$ cannot belong to A.

Similar argument can be used to argue for the case that two defaults, one supporting l and another supporting $\bar{l}$, given that KB is consistent can never be applied at the same time to create both $believes(l,t)$ and $believes(\bar{l},t)$ in an answer set. □

Example 2. *The story in Example 1 can be represented by the $KB_1 = \langle O_1, RD_1\rangle$ where*

$$O_1 = \left\{ \begin{array}{l} obs(in\_college,1). \\ obs(has\_scholarship,2). \end{array} \right\},$$

$$RD_1 = \left\{ \begin{array}{l} default(d_1,\neg poor,[in\_college]). \\ default(d_2,poor,[has\_scholarship]). \end{array} \right\}$$

The statement in Example 1 can be represented by $stm(poor,0)$. Consider $\Pi(KB_1)$ with $k = 0,1,2$:

- *$k = 0$: since there is no observation, no answer set of $\Pi(KB_1)$ contains any atom of the form $believes(poor,0)$;*
- *$k = 1$: because of $obs(in\_college,1)$, $h(in\_college,1)$ is true, d_1 is applicable but d_2 is not. So, every answer set of $\Pi(KB_1)$ contains $h(in\_college,1)$ and $believes(\neg poor,1)$.*
- *$k = 2$: $h(has\_scholarship,2)$, $h(in\_college,1)$, and $h(in\_college,2)$ are true and d_1 is applicable at 1 and 2; d_2 is not applicable at 1 but is applicable at 2. Hence, every answer set of $\Pi(KB_1)$ contains $believes(\neg poor,1)$. However, neither $believes(poor,2)$ nor $believes(\neg poor,2)$ belongs to every answer sets of $\Pi(KB_1)$.*

Given a $KB = \langle O, RD\rangle$, an integer k, and a $stm(b,s)$, we are interested in determining the truthfulness of the statement at time steps $s \le t \le k$.

Definition 3. *Let $KB = \langle O, RD\rangle$ be a knowledge base. Let $stm(b,s)$ be a statement about a belief literal b at the time step s. For each time step t such that $t \ge s$, we say that*

- *$KB \models +stm(b,s)@t$ (or $stm(b,s)$ is believed to be* true *w.r.t. KB at t) if $believes(b,t) \in A$ for every answer set A of $\Pi(KB)$;*
- *$KB \models -stm(b,s)@t$ (or $stm(b,s)$ is believed to be* false *w.r.t. KB at t) if $believes(\bar{b},t) \in A$ for every answer set A of $\Pi(KB)$;*
- *$KB \not\models \pm stm(b,s)@t$ (or $stm(b,s)$ is* undecided *w.r.t. KB at t) if $KB \not\models +stm(b,s)@t$ and $KB \not\models -stm(b,s)@t$.*

Intuitively, $KB \models +stm(b,s)@t$ (resp. $KB \models -stm(b,s)@t$) says that at the time step t, at the same time step or later than the time step that the statement about b is made, the statement is believed to be true (resp. believed to be false). $KB \not\models \pm stm(b,s)@t$ states that there is no information that supports or denies the statement at the time step t. Observe that b, t, and s are constants representing the question whether or not $stm(b,s)$ is believed to be true at step t. As such, requiring $believes(b,t)$ to be in every answer set of the program does not mean that $believes(b,t)$ belongs to every answer set for every t such that $s \le t \le k$.

Observe that this entailment can be computed in ASP by the following rules:

$$\mathtt{t}(H,T) \leftarrow stm(H,S), s(T), T \geq S, believes(H,T). \quad (19)$$

$$\mathtt{f}(H,T) \leftarrow stm(H,S), s(T), T \geq S, neg(H,H_1), believes(H_1,T). \quad (20)$$

$$\mathtt{u}(H,T) \leftarrow stm(H,S), s(T), T \geq S, neg(H,H_1), not\ believes(H,T), not\ believes(H_1,T). \quad (21)$$

Let $\Pi_Q = \Pi(KB) \cup \{(19)-(21)\}$. It can be shown that $KB \models +stm(b,s)@t$, $KB \models -stm(b,s)@t$, and $KB \models \pm stm(b,s)@t$ correspond to $\Pi_Q \models \mathtt{t}(b,t)$, $\Pi_Q \models \mathtt{f}(b,t)$, and $\Pi_Q \models \mathtt{u}(b,t)$, respectively. Hence, we can compute skeptical entailment for the truthfulness of a statement using two calls to an ASP-solver.

Proposition 2. *For* $\mathtt{x} \in \{\mathtt{t},\mathtt{b},\mathtt{u}\}$*, if* $\Pi_Q \cup \{\leftarrow \mathtt{x}(b,t).\}$ *does not have an answer set and* $\Pi_Q \cup \{\leftarrow\ not\ \mathtt{x}(b,t)\}$ *has an answer set then* $\Pi_Q \models \mathtt{x}(b,t)$.

Proof. Observe that if $\Pi_Q \cup \{\leftarrow \mathtt{x}(b,t).\}$ does not have an answer set, this implies that either Π_Q has no answer set or every answer set of Π_Q contains $\mathtt{x}(b,t)$. The second condition implies that Π_Q has at least one answer set. This implies that $\Pi_Q \models \mathtt{x}(b,t)$. □

The above proposition allows us to compute the entailment $KB \models +/-/\pm stm(b,s)@t$ by making two calls to the answer set solver. We can show that $KB_1 \not\models \pm stm(poor,0)@0$; $KB_1 \models -stm(poor,0)@1$; and $KB_1 \not\models \pm stm(poor,0)@2$.

For the sake of our discussion, let us consider $KB_2 = \langle O_1, R_1, D_1 \cup \{prefer(d_2,d_1,[])\}\rangle$. It is easy to see that every answer set of $\Pi(KB_2)$ contains $defeated(d_1,\neg poor,2)$. As such, we have that $KB_2 \not\models \pm stm(poor,0)@0$; $KB_2 \models -stm(poor,0)@1$; and $KB_2 \models +stm(poor,0)@2$.

3.3 Reasoning With Observations about Action Occurrences

In this section, we shift our focus to the set A of actions from the action signature $\langle F,B,A\rangle$. Following common practice in action languages (Gelfond and Lifschitz 1998), we assume that each action is associated with a set of literals, called its *effects*, with each effect potentially predicated on a different set of literals, called *preconditions*. A third set of literals, the *executability conditions*, states when the action can be executed vs impossible. This information is encoded in statements of the form: $exec(a,body)$ and $causes(a,p,body)$ where $a \in A$ is an action, p is a fluent literal, and $body$ is a set of fluent literals. The first statement gives the executability conditions of action a, while the second states that, when $body$ holds, p is one effect of a. We represent an *action occurrence observation* (or *action occurrence*) by a statement of the form $occ(a,s)$ where $a \in A$ and s is a time step. Recall that the notion of knowledge base already supports providing action occurrence observations, as H^n is a set of observations and action occurrences. The semantics of the above statements is encoded by suitable rules added to $\Pi(KB)$. Note that the body of *exec* and *causes* statements is encoded by means of the *mbr* predicate, as we did for rules and defaults in the previous section. We add to $\Pi(KB)$ the following rules:

$$obs(P,S) \leftarrow s(S), occ(A,S), exec(A,M), mbr(P,M). \quad (22)$$

$$obs(L,S+1) \leftarrow s(S), occ(A,S), causes(A,L,M), N = \#count\{X : mbr(X,M)\}, \#count\{X : mbr(X,M), h(X,S)\} == N. \quad (23)$$

Rule (22) indicates that the observation of an action occurrence allows us to infer that its executability conditions must be satisfied—we capture this by generating new observations about the time step in which the action has occurred. In rule (23), we take the approach of encoding the effects of an action occurrence as observations; the rule allows us to observe the consequences of an action from knowledge of its occurrence.

Definition 3 is extended to knowledge bases with action occurrences in a trivial way.

Example 3. *Assume that buying a house near John's university, in Example 1, requires an amount of money that only wealthy people can afford. The effect and precondition of the action (denoted by Act) can be represented by* $cause(buy\_house, has\_house, [])$ *and* $exec(buy\_house, [\neg poor])$*. Let us assume that we observe John buying a house at time step* t_2*. Let* KB_3 *be* KB_1 *extended with Act and the observation* $occ(buy\_house, 2)$*. Given this specification, we expect that the system tells us that the statement of John being poor made at step 0 is false at step 2. Indeed, this is the result sanctioned by* $\Pi(KB_3)$*, since rule* (22) *indicates that* $\neg poor$ *is observed at time 2 and, thus, the default* d_2 *is defeated even when it is applicable (Rule* (11)*), i.e.,* $KB_3 \models -stm(\neg poor, 0)@2$ *holds.*

4 Applications

As a potential application of our framework, consider the following scenario. Bob is walking in a good area of the city when a person suddenly appears, brandishing a knife and displaying a menacing demeanor. At some point during the confrontation, Bob shoots and kills the person. A court needs to decide whether Bob acted in self-defense or not. Simplifying the scenario, we will assume that the only relevant law says that one acted in self-defense if they believed to be in imminent danger.

To make the example more interesting, suppose the police found that the knife the person was holding was fake. Thus, we can identify two possible scenarios:

1. When Bob shot the person, he had not yet noticed that the knife was fake. We could expect to conclude that at that time Bob believed to be in imminent danger.

2. If instead Bob shot the person after observing that the knife was fake, we could expect to conclude that, at the time of the shooting, Bob did not believe to be in imminent danger.

Assume that the police are able to tell when Bob shot the person and when he observed that the knife was fake, for example thanks to a reliable witness who saw the events from Bob's same perspective[11]. We will demonstrate the use of our framework to reason about the evolution of Bob's beliefs and determine if he acted in self-defense.

A possible formalization of the knowledge base from this example is the following set RD^s of statements:

Listing 1: Set RD^s of statements for the self-defense example

```
causes(appears(X),is(X,present),[]).
default(d1s,inImminentDanger,[has(att,k),
  is(att,present),is(att,menc)])
default(d2s,¬inImminentDanger,[has(att,k),
  is(att,present),is(att,menc),is(k,fake)])
prefer(d2s,d1s, [ ]).
default(d3s,¬inImminentDanger,
  [at(a1), is(a1,good)])
prefer(d1s,d3s, [ ]).
```

The representation uses fluents $has(p,o)$ (p has/holds object o), $is(o,a)$ (object o has attribute a), and $at(l)$ (Bob is at location l). The objects are k (the knife), a_1 (the good area of the city), and att (the attacker). Additionally, we attributes constants *present* (for attribute "physically present"), $fake$ (attribute of the knife), *menc* (menacing-looking), and *good* (attribute of the area of the city). The actions are $appears(p)$ (person p appears) and $shoots(p)$ (Bob shoots p). The effect of the former is described by the *causes* statement; the latter action has no effects within the scope of the example. The intuitive meaning of the defaults is:

- d_1^s: If there is someone who is menacing-looking and holding a knife, one will normally believe to be in imminent danger
- d_2^s: In the above case, if the knife is fake, one will not normally believe to be in imminent danger
- d_3^s: One does not normally believe to be in imminent danger in a good area of the city

As can be seen from the $prefer$ statements, d_2^s is the most preferred default and d_3^s the least preferred. The initial part of the story is captured by:

$$O^s_{init} = \left\{ \begin{array}{ll} obs(at(a_1),0). & obs(is(a_1,good),0). \\ obs(\neg is(att,present),0). & occ(appears(att),0). \\ obs(is(att,menc),1). & obs(has(att,k),1). \end{array} \right.$$

Let us consider what our framework yields for $KB^s_{init} = \langle O^s_{init}, RD^s \rangle$. One can check that the answer set of $\Pi(KB^s_{init})$ contains $h(at(a_1),0)$ and $h(is(a_1,good),0)$. The only applicable default is d_3^s, yielding $believes(\neg inImminentDanger,0)$, i.e. Bob does not believe to be in imminent danger at time step 0. The two possible evolutions of the story are formalized as follows:

- Scenario 1:

$$O^s_1 = \{ occ(shoots(bob,att),1). \}$$

$$KB^s_1 = \langle O^S_{init} \cup O^s_1, RD^s \rangle$$

The time step of interest is 1. Note that $h(is(att,menc),1)$ and $h(has(att,k),1)$ follow trivially from the observations. Defaults d_1^s and d_3^s are thus applicable, while d_2^s is not. Because d_1^s is preferred to d_3^s, the answer set of $\Pi(KB^s_1)$ contains $believes(inImminentDanger,1)$, meaning that, at that time Bob pulled the trigger, he believed to be in imminent danger at the time of the shooting. It is worth noting how the framework is able to capture the evolution of Bob's beliefs over time.

- Scenario 2:

$$O^s_2 = \left\{ \begin{array}{l} obs(is(k,fake),1). \\ occ(shoots(bob,att),1). \end{array} \right\}$$

$$KB^s_2 = \langle O^S_{init} \cup O^s_2, RD^s \rangle$$

At time step 1, Bob has observed that the knife is fake and, thus, $h(is(k,fake),1)$ holds. Hence, all defaults are applicable. Because d_2^s is preferred to d_1^s, the answer set of $\Pi(KB^s_2)$ contains $believes(\neg inImminentDanger,1)$. That is, Bob did not initially believe to be in imminent danger and did not believe so at the time of the shooting either.

For another demonstration of our framework, consider the problem of detecting *Man-in-the-Middle* (MITM) attacks targeting computer and cyber-physical systems. In a MITM attack, the attacker secretly places itself as an intermediary between two communicating parties, relaying the information between them. By intercepting the communications, the attacker may steal valuable information, or even alter the information exchanged between the parties and fool them into performing unintended or undesirable actions. For example, a MITM attack was used in Stuxnet,[12] a sophisticated malicious software (malware) that targeted certain models of industrial Programmable Logic Controllers (PLCs). Stuxnet is remarkable in that it is reported to have been successful in impacting industrial systems involved in Iran's nuclear enrichment program. Its success has major implications on the security of industrial systems world-wide.

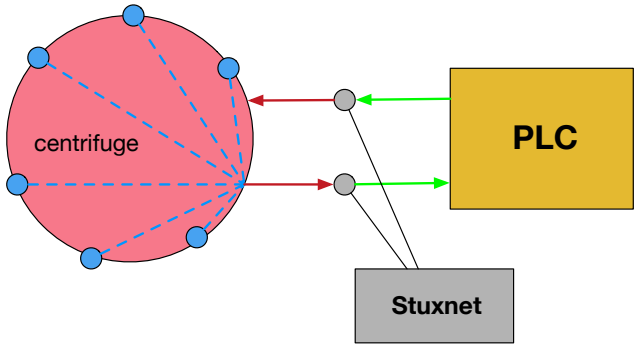

Figure 1: MITM attack carried out by Stuxnet: safe commands sent by the PLC (green) are replaced by dangerous ones (red); alarm readings from the centrifuge's sensors (red) are replaced by safe ones

The behavior of Stuxnet's MITM component is outlined in Figure 1. Its component operates by intercepting the com-

[11] In an actual case, other considerations may play a role, e.g. the reliability of the witness or whether the perspective was really the same. These complications are outside the scope of our example.

[12] https://en.wikipedia.org/wiki/Stuxnet

mands sent by a PLC to a connected centrifuge. The malware first increased the speed of the centrifuge above normal levels for a short amount of time, and later slowed it down below normal levels for a longer period of time. It is believed that the resulting stress caused components of the centrifuge to expand and eventually destroy it. Under normal conditions, sensors installed in the centrifuge would have alerted the PLC—and its users—about the abnormal conditions, giving them a chance to shut down the system before damage occurred. However, as part of the MITM attack, Stuxnet also intercepted the sensor readings from the centrifuge, and sent to the PLC fake readings based on previous recordings that indicated that the system was operating normally (this is known as a *replay attack*).

A seriously concerning feature of MITM attacks is their ability to take control of the involved parties' inputs and outputs, making the attack virtually undetectable to the parties. In this section, we show that detection of a MITM attack is indeed possible if the detection task is reduced to that of reasoning about the truthfulness of a communication partner, as long as one has access to some external knowledge that can be used to evaluate the partner's statements.

To demonstrate this, we consider a simplified MITM attack scenario along the lines of Stuxnet's MITM component. For simplicity of presentation, we do not include in the scenario occurrences of actions, but it is not difficult to see that our approach extends in a natural way when actions occur.

Consider the case of a motor, M, that can be *on* or *off*. A sensor mounted on the motor tells whether the motor is overheating. A controller, C, is programmed to turn off the motor if it is found to be overheating.

Suppose now that the system is the target of a MITM attack. An attacker, A, manages to place itself between C and M, intercepting the communications between them. A intercepts the output of M's sensor, discarding the sensor reading and always providing C with a negative reading independently of the actual state of M, i.e, the attacker could prevent C from turning off an overheating motor, eventually causing it to be damaged. How can such an attack be detected?

The solution leverages a technique for reasoning about cyber-physical systems and their interaction with the physical environment discussed in (Nedelcu and Balduccini 2015). Suppose that a thermostat, T is located near the motor, which generates an audible alert every 20 minutes if the temperature of the room is above 90°F, in order to ensure that the operators take more frequent breaks. Based on world knowledge, during cold weather an observer will normally believe the temperature in the room to be below 90°F.

The thermostat is not related to the functioning of the motor and, thus, it is conceivable that A will not attempt to alter its activities. However, given the proximity of the motor, an alert from the thermostat when the room is not hot can be taken as an indication that M is indeed overheating (and that M's sensor reading may have been tampered with). Let us see how one can draw this conclusion using our framework.

The relevant information can be formalized as shown below. The signature is defined so that *alert* and *winter* are fluents, while all others are beliefs.

Listing 2: Relevant Information

```
default(d1m, hot_room, [alert]).
default(d2m, ¬hot_room, [cold_weather]).
prefer(d2m,d1m, [ ]).
rule(r1m,cold_weather,[winter]).
default(d3m,overheat,[alert, ¬hot_room]).
```

Default d_1^m says that, as a rule, an observer hearing an alert from T will believe the room to be hot. Default d_2^m states that the observer will not believe the room to be hot during cold weather. The third statement expresses a preference for d_2^m when both d_1^m and d_2^m are applicable. This captures the intuition that, during cold weather, one will have a tendency to assume that the room is not hot even if an alert is heard. Rule r_1^m states that, during the winter, one will believe the weather to be cold[13]. The final statement says that, typically, if an alert is generated by T when the room is not hot, then the observer will believe that M is overheating. The statement is encoded as a default to increase elaboration tolerance, making it possible, for example, to take into account faults in the thermostat or the presence of other heat sources.

Let us now suppose that we would like to evaluate the truthfulness of M's sensor reading, and suppose that the sensor reports that there is no overheating. The corresponding statement is: $stm(\neg overheat, 0)$. Next, an alert is generated by T: $obs(alert, 1)$. Let KB_m be the corresponding knowledge base. Clearly, default d_2^m is not applicable. Default d_1^m leads the reasoner to conclude $believes(hot\_room, 1)$, that is, that the temperature in the room is above 90°F. Thus, the reasoner has no reason to doubt the sensor reading: $KB_m \models +stm(\neg overheat, 0)@1$.

Next, the system is informed that it is winter. The updated knowledge base, KB'_m, extends KB_m by the statement $obs(winter, 2)$. It is not difficult to check that the answer set of $\Pi(KB'_m)$ contains $believes(cold\_weather, 2)$ because of the application of rule r_1^m. Both defaults d_1^m and d_2^m are now applicable, but because of the preference over them, only the latter is applied, yielding the conclusion $believes(\neg hot\_room, 2)$. Our framework now derives: $KB'_m \models -stm(\neg overheat, 0)@2$. That is, the sensor reading from M is deemed not truthful, which indicates that the system may be under a MITM attack.

5 Related Work

Within the scope of the ASP and logic programming community, this work is, to the best of our knowledge, novel. As we have mentioned earlier, there is an extensive literature—within and outside the ASP community—for reasoning about actions and change, reasoning about defaults, and diagnostic reasoning about observations. Besides action languages used in this paper, event calculus (e.g., (Kowalski and Sergot 1986; Denecker, Missiaen, and Bruynooghe 1992)) or situation calculus (e.g., (McCarthy and Hayes 1969; Reiter 2001)) are formalisms for reasoning about actions and change and could be used instead of action lan-

[13] We encode this statement in a non-defeasible way for the purpose of illustrating the *rule* construct. In a more realistic scenario, one may want to use a default instead.

guages; and, these formalisms have also been used in diagnosis (e.g., (Arias et al. 2019; McIllraith 1997)). All of these works are somewhat related to the proposed system. However, they are separate formalisms/systems that can be used as components of our system. In this sense, our work is related to (Balduccini and Gelfond 2003b; Balduccini and Gelfond 2003a) and the ones mentioned earlier, such as (Brewka and Eiter 1999; Brewka and Eiter 2000; Delgrande, Schaub, and Tompits 2003; Gelfond and Son 1998) as they are using ASP in the implementation of a certain formalism. Also of note is that the proposed framework bears similarity to frameworks developed to support *diagnosis* as both rely on observations to draw conclusions but our framework does not focus on explaining what goes wrong and does not assume completeness of the knowledge of the agent (e.g., in the form of a complete model).

Formalisms like ASP with preferences (e.g., (Brewka 2005; Brewka et al. 2015)) can provide a foundation for the implementation of the proposed framework—e.g., by facilitating the preferences between defaults.

The AI community has explored the issue of computational trust and reputation in several works. It should be noted that the term "trust" is associated with many orthogonal directions of research. For instance, some researchers intend trust as a measure of the probability that an agent will complete a given task (Player and Griffiths 2019). Others evaluate have studied how trust can be measured in terms of opinions based on, e.g., user ratings or feedback (Drawel, Bentahar, and Qu 2020). Others still have developed techniques for reasoning while taking into account information about trust (of agents, of statements, etc.) (Tang et al. 2012).

Views of trust closer to ours can be found, e.g., in (Sabater and Sierra 2005) for a survey. The survey focuses predominantly on trust models observed in multi-agent scenarios, taking explicitly into account the observations concerning interactions among agents. The survey provides classification of the models according to different dimensions: conceptual model (cognitive vs. game theoretical), information sources (direct interactions, direct observations, witness information, sociological information, prejudice), visibility (subjective vs. global), granularity (context dependent vs. non-context dependent), model type (trust vs. reputation), type of information exchanged (boolean vs. continuous), and agent behavior's assumptions (honest, biased but not lying, lying). Within such classification, our model focuses on trust—but could easily accommodate other interesting forms of reputation)—based on cognitive aspects, it builds on direct observations, but can accommodate sociological biases and prejudice through defaults, it captures subjective visibility, it is context dependent and relies on boolean information.

The survey in (Artz and Gil 2007) places a greater emphasis on surveying models of trust as models to predict attitude towards future interactions with an agent—with less emphasis on assessing the trustworthiness of a current statement.

Finally, this paper substantially expands and improves upon our prior work (Son et al. 2016a; Son et al. 2016b). Specifically, in this paper: we develop a clearer characterization of the reasoning task of interest and of the role played by the agents involved; we provide a more precise treatment of, and distinction between, fluents and beliefs; we develop a characterization of the evolution of beliefs as a stand-alone, central reasoning task; we redefine the task of reasoning about the truthfulness of statements; we develop a more rigorous abstract framework, including a clear description of the various forms of entailment involved and of their roles; we include proofs of the main claims; and we include new and expanded applications and use cases to highlight the new findings.

6 Conclusions and Future Work

In this paper, we proposed a general framework for reasoning about the truthfulness of statements made by an agent. We showed how the framework can be implemented using ASP using well-known methodologies for reasoning about actions and change and for default reasoning with preferences. The framework does not assume complete knowledge about the agent being observed and the reasoning process builds on observations about the state of the world and occurrences of actions. We explored the use of the framework in simple scenarios derived from man-in-the-middle attacks and placed the proposed framework in the context of other related work.

The proposed work can be extended in several directions. First, the default theory considered in this paper is static in the sense that its rules and defaults encode knowledge about the world at a single time point. It is not difficult to imagine that rules and defaults can related knowledge at different time points (e.g., the knowledge about the behavior of some machine within a time interval). Second, we note that our implementation is rather impartial to statements by others (the (**SK**) principle). An alternative view of this is to believe in every statement unless it is proven false. Third, it is reasonable to assume that there might be several sources of information that a user can employ to collect observations which include the agents that are observed. The reliability of each source might influence our decision on whether or not a statement could be believed as true. We believe that a combination of a model of trust together with attitude of agents as described in (Sabater and Sierra 2005; Artz and Gil 2007) and our framework might be useful here. Our system can be easily extended to deal with the first extension as it currently deals with time stamped literals already. The second extension might require the change in Def. 3. The third extension might involve a significant amount of modification. We leave all these extensions for future work.

Acknowledgments

This research is partially supported by NSF grants 1757207, 1812619, 1812628, and 1914635. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the sponsoring organizations, agencies, or the U.S. government.

References

Arias, J.; Carro, M.; Chen, Z.; and Gupta, G. 2019. Modeling and Reasoning in Event Calculus Using Goal-Directed Constraint Answer Set Programming. In *LOPSTR, 2019*.

Artz, D., and Gil, Y. 2007. A Survey of Trust in Computer Science and the Semantic Web. *Journal of Web Semantics* 5:58–71.

ASP Standardization Working Group. 2013. ASP-Core-2 input language format, version 2.01c. Available at https://www.mat.unical.it/aspcomp2013/ASPStandardization.

Balduccini, M., and Gelfond, M. 2003a. Diagnostic Reasoning with A-Prolog. *Theory and Practice of Logic Programming* 3(4,5):425–461.

Balduccini, M., and Gelfond, M. 2003b. Logic Programs with Consistency-Restoring Rules. In *International Symposium on Logical Formalization of Commonsense Reasoning*, AAAI Spring Symposium Series, 9–18.

Brewka, G., and Eiter, T. 1999. Preferred answer sets for extended logic programs. *Artificial Intelligence* 109:297–356.

Brewka, G., and Eiter, T. 2000. Prioritizing default logic. In *Intellectics and Computational Logic*, volume 19 of *Applied Logic Series*. Kluwer. 27–45.

Brewka, G.; Delgrande, J.; Romero, J.; and Schaub, T. 2015. asprin: Customizing answer set preferences without a headache. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25 - 30, 2015, Austin, Texas, USA*.

Brewka, G. 2005. Preferences in answer set programming. In *Current Topics in Artificial Intelligence, 11th Conference of the Spanish Association for Artificial Intelligence, CAEPIA 2005, Santiago de Compostela, Spain, November 16-18, 2005, Revised Selected Papers*, 1–10.

Delgrande, J.; Schaub, T.; and Tompits, H. 2003. A framework for compiling preferences in logic programs. *Theory and Practice of Logic Programming* 3(2):129–187.

Denecker, M.; Missiaen, L.; and Bruynooghe, M. 1992. Temporal reasoning with abductive event calculus. In *Proceedings of ECAI92*.

Drawel, N.; Bentahar, J.; and Qu, H. 2020. Computationally Grounded Quantitative Trust with Time. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS 2020)*, 1837–1839.

Gelfond, M., and Lifschitz, V. 1991. Classical negation in logic programs and disjunctive databases. *New Generation Computing* 365–387.

Gelfond, M., and Lifschitz, V. 1998. Action Languages. *Electronic Transactions on Artificial Intelligence* 3(6).

Gelfond, M., and Son, T. C. 1998. Prioritized default theory. In *Selected Papers from the Workshop on Logic Programming and Knowledge Representation 1997*, 164–223. Springer Verlag, LNAI 1471.

Kowalski, R., and Sergot, M. 1986. A logic-based calculus of events. *New Generation Computing* 4:67–95.

Marek, V., and Truszczyński, M. 1999. Stable models and an alternative logic programming paradigm. In *The Logic Programming Paradigm: a 25-year Perspective*, 375–398.

McCarthy, J., and Hayes, P. 1969. Some philosophical problems from the standpoint of artificial intelligence. In Meltzer, B., and Michie, D., eds., *Machine Intelligence*, volume 4. Edinburgh: Edinburgh University Press. 463–502.

McIllraith, S. 1997. *Towards a formal account of diagnostic problem solving*. Ph.D. Dissertation, Toronto.

Minnich, A. J.; Chavoshi, N.; Mueen, A.; Luan, S.; and Faloutsos, M. 2015. Trueview: Harnessing the power of multiple review sites. In Gangemi, A.; Leonardi, S.; and Panconesi, A., eds., *Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, May 18-22, 2015*, 787–797. ACM.

Nedelcu, A., and Balduccini, M. 2015. An Approach and Tool for Reasoning about Situated Cyber-Physical Systems. In *The First Workshop on Action Languages, Process Modeling, and Policy Reasoning (ALPP 2015)*.

Player, C., and Griffiths, N. 2019. Addressing Class Imbalance in Trust and Stereotype Assessment. In *Proceedings of the 21st International Workshop on Trust in Agent Societies (TRUST 2019)*.

Reiter, R. 2001. *KNOWLEDGE IN ACTION: Logical Foundations for Specifying and Implementing Dynamical Systems*. The MIT Press.

Rowe, N. 2006. Deception in Electronic Goods and Services. In *Encyclopedia of E-Commerce, E-Government, and Mobile Commerce*. The Idea Group.

Sabater, J., and Sierra, C. 2005. Review on Computational Trust and Reputation Models. *Artificial Intelligence Review* 24:33–60.

Son, T. C.; Pontelli, E.; Gelfond, M.; and Balduccini, M. 2016a. Reasoning about Truthfulness of Agents Using Answer Set Programming. In *15th International Conference on Principles of Knowledge Representation and Reasoning*.

Son, T. C.; Pontelli, E.; Gelfond, M.; and Balduccini, M. 2016b. An Answer Set Programming Framework for Reasoning about Truthfulness of Statements by Agents. In *32nd International Conference on Logic Programming (ICLP16)*.

Tang, Y.; Cai, K.; McBurney, P.; Sklar, E.; and Parsons, S. 2012. Using Argumentation to Reason about Trust and Belief. *Journal of Logic and Computation* 22(5):979–1018.