Reasoning about Truthfulness of Agents Using Answer Set Programming

Tran Cao Son and **Enrico Pontelli**
Computer Science Department
New Mexico State University

Marcello Balduccini
Computer Science Department
Drexel University

Abstract

We propose a declarative framework for representing and reasoning about truthfulness of agents using answer set programming. We show how statements by agents can be evaluated against a set of observations over time equipped with our knowledge about the actions of the agents and the normal behavior of agents. We illustrate the framework using examples and discuss possible extensions that need to be considered.

Introduction

The recent advancements of AI enable the development of agents that can replace human in many tasks. Increasingly, organizations are using web-bots for interacting with clients in various capacities, e.g., in providing information about the company or making offers. It is reasonable to believe that this trend will be continued as long as the Internet exists. Unfortunately, not every business on the Internet is honest as one would hope. Stories about businesses that cheat people of goods or services or wrongly advertise their services are not uncommon. This leads to the development of businesses that allow people to rate companies (e.g., expedia.com or Angie's List) or defend the reputation of a company or an entity (e.g., reputation.com).

In this paper we are interested in reasoning about the truthfulness of agents. We start with an optimistic assumption that agents are truthful unless otherwise proven. We will judge agents by what they do or what we observe rather than by what they say. Our observations are made at different time instances along a linear time line. We assume that whatever observed is true at the time it is observed and will stay true until additional information indicates otherwise. Furthermore, we will need to judge agents even when we do not have complete information about them. This means that reasoning about the truthfulness of agents is a non-monotonic task that is often done under incomplete information. To illustrate these issues, let us consider the following situations.

1. John said that *he does not have money*.

   We observe that John does not attend college.

   Given the above observation, we would likely conclude that John is being honest in his statement, i.e., we trust that *John does not have money* which does not allow him to attend a college.

2. We observe that John has a Ferrari.

   A common person would normally be unable to afford a luxury car. At this point, we will likely withdraw our conclusion about John having no money and classify John as a liar.

3. We learn that John inherited the Ferrari from his parents.

   Inheritance is a possible way to acquire something and thus we might need to reconsider our position about John being a liar. Yet, if someone inherits a Ferrari from his parents then commonsense dictates that the person comes from an affluent household and thus should have money (enough to attend a college). All this reasoning implies that we should not trust John even after this new piece of information.

4. We learn that John is a gambler.

   A gambler is usually penny-less. Luxury items he may own are likely obtained by gambling rather than purchased. Hence, we might be inclined to trust that John does indeed have no money.

The above sequence of observations and the associated reasoning process are rather simple but they illustrate clearly the way humans reason about the truthfulness of statements by others and make judgements about each other. The fact that John does not attend college is observed and is used to deduce that John does not have (enough) money to attend college. This makes John's statement truthful. However, additional observations such as John has a Ferrari or inherits the Ferrari from his family would cause us to withdraw our conclusion about John being poor, i.e., John's statement is untruthful. Yet, an additional observation that John is a gambler will force us to withdraw our latest conclusion about John and conclude again that his statement is truthful. Understandably, this process can continue forever and our opinion about John's statement might or might not change after each round of new observations.

In this paper, we propose a framework for representing and reasoning about the truthfulness of agents using Answer Set Programming, a knowledge representation language useful for commonsense reasoning in presence of incomplete information (Baral 2003).

Background: Answer Set Programming

In this section, we will review the basic definitions of Answer Set Programming (ASP) (Baral 2003), i.e., the language of logic programs under the answer set semantics (Gelfond and Lifschitz 1990). A logic program Π is a set of rules of the form

$$c_1 \mid \ldots \mid c_k \leftarrow a_1, \ldots, a_m, \ not\ a_{m+1}, \ldots, \ not\ a_n \quad (1)$$

where $0 \leq m \leq n$, $0 \leq k$, each a_i or c_j is a literal of a propositional language[1] and not represents *negation-as-failure*. A negation as failure literal (or naf-literal) is of the form $not\ a$ where a is a literal. For a rule of the form (1), the left and right hand sides of the rule are called the *head* and the *body*, respectively. Both the head and the body can be empty. When the head is empty, the rule is called a *constraint*. When the body is empty, the rule is called a *fact*.

For a rule r of form (1), $H(r)$ and $B(r)$ denote the left and right hand side of $\leftarrow$, respectively; $head(r)$ denotes $\{c_1, \ldots, c_k\}$; and $pos(r)$ and $neg(r)$ denote $\{a_1, \ldots, a_m\}$ and $\{a_{m+1}, \ldots, a_n\}$. For a program Π, $lit(\Pi)$ denotes the set of literals occurring in Π.

Consider a set of ground literals X. X is *consistent* if there exists no atom a such that both a and $\neg a$ belong to X. The body of a rule r of the form (1) is *satisfied* by X if $neg(r) \cap X = \emptyset$ and $pos(r) \subseteq X$. A rule of form (1) with nonempty head is satisfied by X if either its body is not satisfied by X or $head(r) \cap X \neq \emptyset$. A constraint is *satisfied* by X if its body is not satisfied by X.

For a consistent set of ground literals S and a program Π, the *reduct* of Π w.r.t. S, denoted by Π^S, is the program obtained from the set of all ground instances of Π by deleting (**i**) each rule that has a naf-literal *not* a in its body with $a \in S$, and (**ii**) all naf-literals in the bodies of the remaining rules.

S is an *answer set (or a stable model)* of Π (Gelfond and Lifschitz 1990) if it satisfies the following conditions: (**i**) If Π does not contain any naf-literal (i.e. $m = n$ in every rule of Π) then S is a minimal consistent set of literals that satisfies all the rules in Π; and (**ii**) If Π does contain some naf-literal ($m < n$ in some rules 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; hence its answer set is defined in the first item. A program Π is said to be *consistent* if it has a consistent answer set. Otherwise, it is inconsistent.

An Illustration

In this section, we encode the reasoning process that is used in the introduction to evaluate John's statements. We assume a propositional language that encodes properties of interest such as has_money (John has money), $in\_college$ (John is in college), etc. with obvious meaning[2]. In addition, we will use special predicates of the form $statement(x, t)$ to denote that "the statement x is made at time step t," $observed(x, t)$ to denote that "we observe x is true at the time step t," and $h(x, t)$ to indicate that "x is (believed to be) true at the time step t." For simplicity, the representation of time is based on discrete steps with negligible duration.

John's statement that he does not have money can be encoded by the following formula[3].

$$statement(\neg has\_money, 0). \quad (2)$$

Our observation that John does not attend colleges is encoded as follows.

$$observed(\neg in\_college, 0). \quad (3)$$

The common knowledge that attending college normally requires money (i.e., normally, a person does not attend college then he/she does not have money) can be encoded as follows.

$$h(\neg has\_money, T) \leftarrow h(\neg in\_college, T), \ not\ ab_1(T). \quad (4)$$

In the above rule, $ab_1(T)$ is used to characterize possible exceptions to the rule. For example, unqualified individuals or people who are not interested in a college degree are exceptions to this rule.

Let π_1 be a program consisting of the commonsense rule (4), the observation (3), and the rules

$$h(l, N) \leftarrow observed(l, T), N \geq T, \ not\ ab_{ob}(l, T, N). \quad (5)$$

$$\neg ab_{ob}(l, T, T) \leftarrow observed(l, T) \quad (6)$$

$$ab_{ob}(l, T, N) \leftarrow h(\bar{l}, N) \quad (7)$$

that we will refer to as rules for reasoning about observations. $\bar{l}$ denotes the complement of l, i.e., $\overline{a} = \neg a$ and $\overline{\neg a} = a$ for an atom a. Rule (5) says that an observation about l (a literal) is true from the time it is observed unless otherwise stated. Rule (6) indicates that an observation must be true at the time it is made. Finally, (7) says that newer information will override older one. It is easy to see that the program π_1 has a unique answer set S_1 consisting of (3) and

$$\left\{ \begin{array}{l} h(\neg in\_college, 0), h(\neg has\_money, 0), \\ \neg ab_{ob}(has\_money, 0, 0) \end{array} \right\}$$

Furthermore, $h(\neg has\_money, 0)$ indicates that John does not have money (at step 0), which means that the literal occurs in the statement (2) is "satisfied" by S_1.

Let us consider the second observation. We observe that John has a Ferrari. This is expressed as follows.

$$observed(has\_ferrari, 1). \quad (8)$$

A person with a Ferrari usually has money. This is encoded as follows[4].

$$h(has\_money, T) \leftarrow h(has\_ferrari, T), \ not\ ab_2(T). \quad (9)$$

[1] Rules with variables are viewed as a shorthand for the set of their ground instances.

[2] For simplicity, we will not attach the subject (John) to the literals as our discussion focuses on him. The programs and discussions can easily be extended to consider several subjects simultaneously.

[3] We abuse notation slightly and use $\neg$ in conjunction with reified properties.

[4] For simplicity of the presentation, we use a fairly simple representation. A more realistic formalization would need to include the fact that to has a Ferrari one needs to buy it and the action of buying a Ferrari requires a lot of money. This will ultimately allow us to conclude that John has money.

It is easy to see that rules (5)–(6) will allow us to conclude that $h(has\_money, 1)$ is true in every answer set of the program π_2 that consists of π_1 and (8)-(9). In fact, it can be shown that the unique answer set S_2 of π_2 is the union of S_1, (8), and the two literals

$$\left\{ \begin{array}{l} h(has\_money, 1), h(has\_ferrari, 1), \\ \neg ab_{ob}(\neg has\_money, 0, 1) \end{array} \right\}$$

which indicates that, at step 1, we believe that John does have money, i.e., John's statement is not truthful.

Let us consider the next observation that John inherited the Ferrari.

$$observed(has\_ferrari\_from\_parent, 2). \quad (10)$$

Our commonsense knowledge stating that if a person inherits a Ferrari then that person would come from an affluent family and thus would have money is encoded—in a simplified manner—as follows.

$$h(has\_money, T) \leftarrow h(has\_ferrari\_from\_parent, T), \; not\; ab_3(T). \quad (11)$$

Let π_3 be the program that consists of π_2 and (10)–(11). Again, we can easily check that this program has a unique answer set S_3 that consists of S_2, (10), and the two literals

$$\left\{ \begin{array}{l} h(has\_money, 2), h(has\_ferrari\_from\_parent, 2) \\ \neg ab_{ob}(\neg has\_money, 0, 2) \end{array} \right\}$$

which indicates that John has money and thus is not truthful.

Let us now consider the last observation that John is a gambler. This is expressed by the fact

$$observed(is\_gambler, 3). \quad (12)$$

and our knowledge about gamblers is expressed by

$$h(\neg has\_money, T) \leftarrow h(is\_gambler, T),\; not\; ab_4(T). \quad (13)$$

A gambler would constitute an exception to the rule (9) as well as to the rule (11). This means that we also need the following rules

$$ab_2(T) \leftarrow h(is\_gambler, T). \quad (14)$$

$$ab_3(T) \leftarrow h(is\_gambler, T). \quad (15)$$

Let π_4 be the program consisting of π_3 and the rules (12)—(15). We can easily check that it has one answer set S_4 that consists of S_3, (12), and the literals

$$\left\{ \begin{array}{l} ab_2(3), ab_3(3), h(is\_gambler, 3), h(\neg has\_money, 3) \\ ab_{ob}(has\_money, 1, 3), ab_{ob}(has\_money, 2, 3) \end{array} \right\}$$

This answer set allows us to conclude that, at step 3, we believe that John's initial statement is indeed true.

Our Framework

The previous section provides a glimpse on how answer set programming can be used in reasoning about the truthfulness of agents. We will now present a framework for representing and reasoning about the truthfulness of (statements made by) agents. We assume that

- agents' actions and their effects are fully observable, and actions have preconditions (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);
- we can observe over time the occurrence of the agents' actions and the properties of the world (e.g., observe that John buys a car; John is a gambler);
- we have a repertoire of commonsense knowledge about normal behaviors (e.g., a person buying a luxury car normally has money; a gambler usually has no money).

This leads us to define a knowledge base about an agent as follows.

Definition 1 *A* knowledge base *(about an agent) over a propositional language $\mathcal{L}$ is a triple $\langle \Pi_A, \Pi_K, \Pi_O \rangle$ where*

- *Π_A is a program encoding the agents' actions and their effects by defining the truth value of literals of the form $h(l, t)$ where l is a (reified) literal of the language $\mathcal{L}$ and t denotes a time step. That is, Π_A defines the set of agent actions, specifies their direct effects and their executability conditions, and includes rules for reasoning about the indirect effects;*
- *Π_K is a program encoding commonsense knowledge that contains rules defining $h(l, t)$; and*
- *Π_O is a collection of atoms of the form $observed(l, t)$ or $occurred(a, t)$ where l is a literal of $\mathcal{L}$, a is action defined in Π_A, and t is a time step.*

Observe that there is extensive literature on reasoning about actions and change and commonsense reasoning using logic programming (e.g., (Baral and Gelfond 1994; Gelfond and Lifschitz 1992; Son et al. 2006)). As such, the two components Π_A and Π_K can be adapted from previous research for the purpose of this paper in a straightforward manner. For instance, Π_A can follow the proposal given in the seminal paper (Gelfond and Lifschitz 1992); Π_K can contain the rules (9), (11), (13), and (14)–(15), etc. discussed in the previous section.

Next, we define the semantics of knowledge bases. For this purpose, we introduce rules that allow for the reasoning about observations. Rules (5)–(7) allow for reasoning about observations of the form $observed(l, t)$. The following rules are for observations of the form $occurred(a, t)$. For each action a with executability condition $h(p_1, t), \ldots, h(p_n, t)$, we include the set of rules

$$h(p_i, t) \leftarrow occurred(a, t),\; not\; ab_a(t).\; (i = 1, \ldots, n) \quad (16)$$

This rule says that, if the action a was observed to occur, then its preconditions must have been satisfied, unless it is an exception. We will denote the set of rules (5)—(7) and (16) as $\Pi_R(KB)$. We define

Definition 2 *Given a knowledge base $KB = \langle \Pi_A, \Pi_K, \Pi_O \rangle$, we say that S is an answer set of KB if and only if it is an answer set of the program $\Pi_A \cup \Pi_K \cup \Pi_O \cup \Pi_R(KB)$ under the semantics given in the background section.*

Next, we define the notion of a statement.

Definition 3 *A* statement *about l of a language $\mathcal{L}$ at the time step t is an expression of the form $statement(l, t)$.*

Intuitively, the expression represents a statement by an external source that l holds at the time step t. The next definition allows for the evaluation of the truthfulness of a statement against a knowledge base.

Definition 4 *Let $KB = \langle \Pi_A, \Pi_K, \Pi_O \rangle$ be a knowledge base over $\mathcal{L}$. We say that*

- *$statement(l, t)$ is* true *w.r.t. KB, denoted $KB \models +statement(l, t)$, if for every answer set S of KB, $h(l, t) \in S$.*
- *$statement(l, t)$ is* false *w.r.t. KB, denoted $KB \models -statement(l, t)$, if for every answer set S of KB, $h(\bar{l}, t) \in S$.*
- *$statement(l, t)$ is* unknown *w.r.t. KB, denoted by $KB \not\models \pm statement(l, t)$, if $KB \not\models +statement(l, t)$ and $KB \not\models -statement(l, t)$.*

The above definition allows for reasoning about statements against a knowledge base. We consider statements that are true, false, and unknown as *truthful*, *dishonest*, and *undecided* respectively.

Example 1 *Let $KB^1_{John} = \langle \Pi^1_A, \Pi^1_K, \Pi^1_O \rangle$ where*

- *$\Pi^1_A = \emptyset$*
- *Π^1_K is the set of rules* (4)*,* (9)*,* (11)*,* (13)*,* (14)–(15)
- *Π^1_O is the set of facts* (3)*,* (8)*,* (10)*, and* (12)*.*

We can easily check that

- *$KB^1_{John} \models statement(\neg has\_money, 0)$*
- *$KB^1_{John} \models -statement(\neg has\_money, 1)$*
- *$KB^1_{John} \models -statement(\neg has\_money, 2)$*
- *$KB^1_{John} \models statement(\neg has\_money, 3)$*

This reflects correctly our sentiment regarding John's statement about him not having money given the sequence of observations about John.

Next, we change the example slightly to account for observations about action occurrences.

Example 2 *Consider the action of buying an expensive car like a Ferrari (buy_car). We know that a pre-condition of this action is money, i.e., an agent executing this action must have money. Let us know consider the $KB^2_{John} = \langle \Pi^2_A, \Pi^2_K, \Pi^2_O \rangle$ where*

- *Π^2_A consists of the encoding of the action buy_car (e.g., following the representation described in (Gelfond and Lifschitz 1992)).*
- *Π^2_K is the collection of rules* (4)*,* (9)*,* (11)*,* (13)*, and* (14)–(15)
- *Π^2_O is the set of facts* (3)*,* (10)*, and* (12)*, and the new observation, shown below, instead of* (8)*.*

$$occurred(buy\_car, 1)$$

Notice that Π_R for KB^2_{John} contains the following rule

$$h(has\_money, T) \leftarrow occured(buy\_car, T), \quad not\ ab_{buy\_car}(T). \tag{17}$$

that is an instance of (16)*. It is easy to see that*

- *$KB^2_{John} \models statement(\neg has\_money, 0)$*
- *$KB^2_{John} \models -statement(\neg has\_money, 1)$*
- *$KB^2_{John} \models -statement(\neg has\_money, 2)$*
- *$KB^2_{John} \models statement(\neg has\_money, 3)$*

i.e., our conclusions on the statement do not change.

Conclusions and Discussion

We describe a framework based on answer set programming for reasoning about the truthfulness of agents. We show that the task can be achieved by combining the three components, our knowledge about the agent's actions, our commonsense knowledge about the world, and our observations about the agents. To the best of our knowledge, no such formalization exists.

In the literature, dishonesty and lying have been investigated from different perspective. The focus has been on defining what is a lie (Mahon 2008) or a bullshit (Frankfurt 2005) in a static environment. Our framework takes into consideration changes in the environment that can be the result of the agents' own actions and/or our knowledge about normal behaviors. So far, we only attempted to evaluate simple statements, i.e., statements about literals. One of our immediate desires is to extend the framework to allow the reasoning about the truthfulness of statements about rules, e.g., "John said that he does not attend college because he does not have money."

References

Baral, C., and Gelfond, M. 1994. Logic programming and knowledge representation. *Journal of Logic Programming* 19,20:73–148.

Baral, C. 2003. *Knowledge Representation, reasoning, and declarative problem solving with Answer sets*. Cambridge University Press, Cambridge, MA.

Frankfurt, H. G. 2005. *On Bullshit*. Princeton Univ. Press.

Gelfond, M., and Lifschitz, V. 1990. Logic programs with classical negation. In Warren, D., and Szeredi, P., eds., *Logic Programming: Proceedings of the Seventh International Conference*, 579–597.

Gelfond, M., and Lifschitz, V. 1992. Representing actions in extended logic programs. In *Joint International Conference and Symposium on Logic Programming.*, 559–573.

Mahon, J. E. 2008. Two definitions of lying. *Journal of Applied Philosophy* 22(2):211–230.

Son, T. C.; Baral, C.; Tran, N.; and McIlraith, S. 2006. Domain-Dependent Knowledge in Answer Set Planning. *ACM Transactions on Computational Logic* 7(4).