Ideological conflict - notes on the development of a commonsense knowledge base about ideologies

Yana Todorova
Texas Tech University
Computer Science
Box 43104
Lubbock, TX 79409-3104, USA
yana.todorova@ttu.edu

Abstract

In this paper, we discuss the applicability of the knowledge representation language A-$Prolog$ for the design and implementation of a commonsense knowledge base about ideologies. We also present a formalization of a simple motivating story, which involves an ideological conflict between countries. The emphasis is on the development and implementation of a commonsense knowledge base needed for the axiomatization of the domain **ideological conflict**. *We use A-$Prolog$ (a language of logic programs under the answer set semantics), to model ideologies and to detect conflicts between them. The notion of ideological conflict presented in this paper is a special case of a more general notion of war of ideologies, which is an important topic for the intelligence community.*

1. Introduction and Motivation

The current political situation around the world is rapidly evolving. The intelligence organizations of all the countries are trying to understand the global consequences of different actions of their nations. It is important to be able to create automated tools to help those organizations. These tools should contain large knowledge bases about political sciences and commonsense knowledge. Therefore, to represent knowledge regarding the political situation is important for the global intelligence and security.

For instance, the Advanced Research and Development Activity (ARDA) is the US intelligence community (IC) center for conducting advanced research and development related to information technology (IT) (see [IC03]). ARDA is a support system for the intelligence analysts and its goal is to improve the reliability of the conclusions of decision makers. One of the programs, in which we are participating, is the Advanced Questions Answering for Intelligence Program (AQUAINT). It is pursuing advanced research for scenario-based, advanced question answering in which, multiple, inter-related questions are asked in a particular topic area by a skilled, professional information analyst, who is attempting to respond to larger, more complex information needs or requirements.

Part of our efforts consists of building models of relevant domains. There are already results in this direction. For instance, the formalization of the travel domain [Gel06]. In that work, the author shows the axiomatization of a journey (a movement of a group of objects from one place to another). He outlines a language M for defining knowledge modules and for assembling them into a knowledge base.

In this work, we model a different domain, namely the ideological conflict. It is important for the better understanding of the effects of actions of the countries in the real world. The emphasis of our work is on the development and implementation of a general commonsense knowledge base needed for the axiomatization of the domain **ideological conflict**. For simplicity, we will be mainly interested in the relationship between groups of countries, based on their political ideology at different stages of their history (other type of ideology is the religious one).

Each alliance is formed by several countries, which share the same ideology. In this paper, we view an ideology as a set of ideas central to a society and an ideological conflict as a discord between collections of countries with different ideologies. The following example will be used to illustrate the proposed formalization of knowledge.

Example 1 (Consider the following story:) *In our hypothetical world, depicted in Figure 1, there are only ten countries: US, $Poland$, $Bulgaria$, $Korea$, $Cuba$, $Russia$, $China$, $Iran$, $Iraq$ and $Syria$. We divide them into three ideologically-driven groups, based on our knowledge about the current global political situation: Capitalism (called*

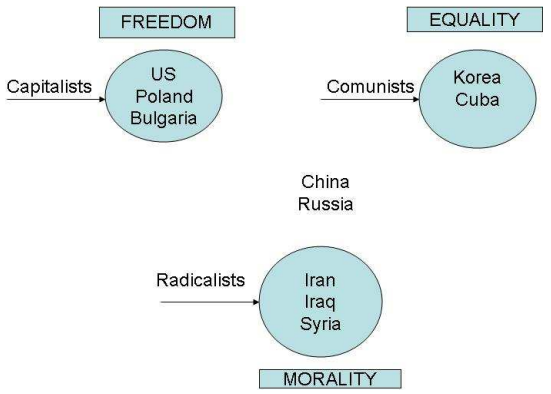

Figure 1. At the beginning of the story: Countries divided based on their ideologies

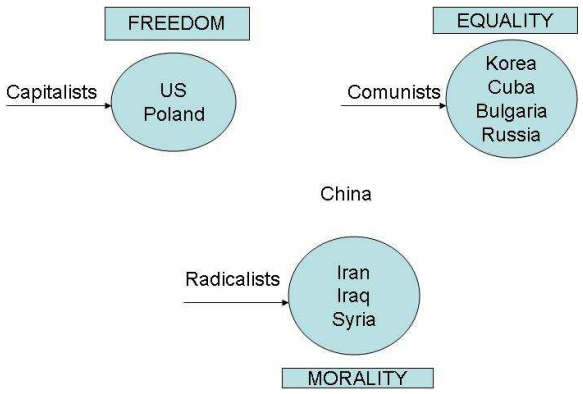

Figure 2. At the end of the story: Countries divided based on their ideologies

freedom alliance), Comunism (or equality alliance) and Muslim Radicalism (named morality alliance). Assume this unreal situation: at the beginning of the story US, $Poland$ and $Bulgaria$ belong to the first group. On the other hand, $Korea$ and $Cuba$ belong to the second group. The last one is composed by $Iran$, $Iraq$ and $Syria$. $Russia$ and $China$ do not belong to any of those coalitions.

For simplicity assume that normally, there is an ideological conflict between the alliances $freedom$ and $equality$. Imagine that later, after a political change, Bulgaria becomes a comunist country; Russia also joins the equality alliance; and the US remains in the freedom alliance. Consider that an intelligence analyst would like to answer the following simple questions about three of the countries:
1. What is the alliance to which Bulgaria belonged before the change?
2. What is the alliance to which Bulgaria belongs after the change?
3. Is there an ideological conflict at the end of the story between US and Russia?

The expected answers to the first and second questions are $freedom$ and $equality$, respectively. Since the new ideology of $Russia$ is different from the ideology of US, the answer of the third question is yes: $Russia$ and US are in an ideological conflict at the end of the story (see figure 2).

To automate this reasoning, we need a language capable of representing the above story as well as expressing defaults, causal relations and other types of commonsense knowledge. Therefore, we use A-$Prolog$ (ASP) - a language of logic programs with two negations and disjunction under the answer set semantics [GL91]. Among the important properties of ASP are its simplicity, expressiveness and the ability to reason with incomplete information. ASP has also a theoretical support and already developed reasoning systems (see for instance [SS00]).

We review the syntax and semantics of ASP in section 2. For more details about the use of ASP, as a knowledge representation language, one may look at [Bar03]. In addition, we use the action language AL [GB00], which can be thought of as formal model of the part of the natural language that is used for describing the behavior of a dynamic domain. We select AL, because it increases the programmer's confidence that the formalization of the domain is correct. A theory in an action language normally consists of an action description (knowledge about effects of actions) and a history description (observations of an agent) [Gel02]. We give a brief review of AL in section 3. For examples in AL, the reader may refer to [GB00].

The rest of the paper is organized as follows: in section 4, we formalize the domain using AL and in section 5, we show the translation to ASP for reasoning. The codification of the particular story and results are shown in section 6. Some of the related work is presented in section 7. Finally, the conclusions and the future work are listed in section 8.

2. Syntax and Semantics of A-Prolog

In this section, we will review the syntax and semantics of *A-Prolog* (ASP), as described in [GBS04]. Before, we recall some basic definitions from [Bar03].

Definition 1 *A term is inductively defined as follows:*
(1) A veriable is a term.
(2) A constant is a term.
(3) If f is an n-ary function symbol and $t_1, \ldots, t_n$ are terms then $f(t_1, \ldots, t_n)$ is a term.

Definition 2 *A term is said to be ground, if no variable occurs in it.*

Definition 3 *An atom is of the form $p(t_1, \ldots, t_n)$, where p is a predicate symbol and each t_i is a term. If each of the t_is is ground, then the atom is said to be ground.*

Definition 4 *A literal is either an atom or an atom preceded by the symbol $\neg$. A literal is referred to as ground if the atom in it is ground.*

Definition 5 *An ASP knowledge base consists of rules of the form:*

$$l_0 \leftarrow l_1, \ldots, l_m, \textbf{ not } l_{m+1}, \ldots, \textbf{ not } l_n$$

where each of the l_is is a literal (atom or its classical negation) and **not** *is the $negation\ as\ failure$ connective.*

The classical negation states that something is false, while the negation as failure means that there is no reason to believe in something.

The answer set semantics of a logic program Π assigns to Π a collection of $answer\ sets$. These are consistent sets of ground literals corresponding to beliefs, which can be built by a rational reasoner on the basis of the rules of Π. The reasoner is guided by the following informal principles:

- It should satisfy the rules of the program in this way: If one believes in the body of a rule, one must believe in its head.
- It should follow the $rationality\ principle$, which states that, one shall not believe anything he is not forced to believe.

The formal definition of answer sets is given first for programs without default negation. Let Π be such a program and let S be a $consistent$ set of ground literals (that is, no atom and its negation appear in such set). The set S is $closed$ under Π if, for every rule of Π, $l_0 \in S$ whenever for every $1 \leq i \leq m,\ l_i \in S$ and for every $m+1 \leq j \leq n,\ l_j \notin S$.

Definition 6 (Answer Sets-part one) *A set S is an $answer\ set$ for Π if S is minimal among the sets closed under Π.*

Now, let us review the second part of the definition of answer sets. For any program Π and consistent set S of ground literals, the $reduct$ Π^S of Π relative to S is the set of rules:

$$l_0 \leftarrow l_1, \ldots, l_m$$

for all rules in Π such that $l_{m+1}, \ldots, l_n \notin S$. Therefore, Π^S is a program without default negation.

Definition 7 (Answer Sets-part two) *S is an answer set for Π if S is an answer set for Π^S.*

Definition 8 (Entailment) *A program Π entails a literal l ($\Pi \models l$) if l belongs to all answer sets of Π. The Π's answer to a query l is* **yes** *if $\Pi \models l$,* **no** *if $\Pi \models \bar{l}$, and* **unknown** *otherwise, where $\bar{l}$ is the negation of l.*

3. Action language AL

The action language AL is divided in three parts: $action\ description\ language$, $history\ description\ language$, and $query\ language$. Let review the action description part AL_d first, as described in [GB00].

The signature, Σ, of AL_d consists of the set F of fluents (statements whose truth depends on time) and the set A of $elementary\ actions$. A set $\{a_1, \ldots, a_n\}$ of elementary actions is called a $compound$ action (where the elementary actions are performed simultaneously). An action description of $AL_d(\Sigma)$ is a collection of propositions of the form:
1. $causes(a_e, l_0, [l_1, ..., l_n])$,
2. $caused(l_0, [l_1, ..., l_n]), and$
3. $impossible\_if(a, [l_1, ..., l_n])$.
where a_e and a are elementary and arbitrary actions respectively and $l_0, \ldots, l_n$ are fluent literals (fluents and their negations) from the signature Σ. An action description A of AL_d defines a transition diagram describing effects of actions on the possible states of the domain.

We now review the language AL_h, which specifies the history of the domain. The past is described by the set Γ of axioms (referred to as $observations$):
1. $happened(a, k)$.
2. $observed(l, k)$.
A set of axioms defines the collection of paths in a transition diagram. A pair $\langle A, \Gamma \rangle$, where A is an action description and Γ is a set of observations, is called a $domain\ description$.

Finally, the query language AL_q includes the following queries:
1. $holds\_at(l, t)$.
2. $currently(l)$.
3. $holds\_after(l, [a_n, \ldots, a_1], t)$.
There exists a close relationship between AL and logic programming under the answer set semantics, which allows reformulation of the knowledge in ASP [BM03].

4. Building the Knowledge Base

We start with the building of the knowledge base, which is used to describe the ideological conflict domain. This domain can be represented by a transition diagram, whose states are sets of fluents and whose arcs are labeled by actions.

Our domain has countries, alliances and ideologies. A country may perform an action of switching to a new ideology and an action of changing from its current alliance

to another one. The first action is possible only if the country has different ideology than the one it desires to switch to. The second action is possible only if the country belongs to a different alliance than the one it desires to join.

An ideological conflict may occur between two alliances with different ideologies. Let us construct an action description A of the domain and use it as a starting point to model an ideological conflict in ASP. We start with a description of the signature.

The fluent $belong(C, A)$ means that a country C belongs to an alliance A. The value of this fluent may be changed with a dynamic causal law, which says that if a country changes to an alliance, it will belong to that alliance. In addition, a static law guarantees that a country belongs to only one alliance at the same time. The fluent $has(A, I)$ stands for: an alliance A has an ideology I. It is defined by a static causal law, which says that an alliance can have only one ideology at the same time. The predicate $in\_conf(A1, A2)$ says that two alliances are in conflict. There is a law for the symmetry of this relationship. Finally, the fluent $conflict(C1, C2)$ means that country $C1$ and country $C2$ are in conflict. It is changed by the static causal law, which says that if two alliances are in conflict, then the countries that belong to these alliances, will be in conflict, respectively. Below is the action description.

Types:

```
country(C).
ideology(I).
alliance(A).
```

Fluents:

```
fluent(belong(C,A)).
fluent(has(A,I)).
fluent(conflict(C1,C2)).
```

Actions:

```
action(change(C,A)).
```

Causal Laws:

```
impossible change(C,A) if belong(C,A).

change(C,A) causes belong(C,A).

caused -belong(C,A1) if belong(C,A2),
                        A1 != A2.

caused -has(A,I1) if has(A,I2),
                     I1 != I2.

caused conflict(C1,C2) if in_conf(A1,A2),
                          belong(C1,A1),
                          belong(C2,A2),
                          A1 != A2.

caused in_conf(A1,A2) if in_conf(A2,A1),
                         A1 != A2.
```

Initially:

```
normally in_conf(equality,freedom).
```

In the following two sections, we represent the general knowledge about ideologies, effects of actions and situation in our hypothetical world (section 5), as well as the particular story (section 6).

5. ASP representation of the Knowledge Base

This is the first part of our Knowledge Base, namely, the representation of general knowledge of ideologies and conflicts.

The following is a representation of the domain description in AL. We use the syntax of the ASP's inference engine Smodels [SS00]. In order to save place, we use $h(F, T)$ instead of $holds(F, T)$ and $o(A, T)$ instead of $occurs(A, T)$. The first one is used to say that a fluent F holds (or is true) at a given point of time T. The second one means that an action A occurs at some moment T.

```
#const n=1.
time(0..n).
#domain country(C;C1;C2).
#domain ideology(I;I1;I2).
#domain alliance(A;A1;A2).
#domain time(T).
#domain fluent(FL;FL1;FL2).

% country C belong to alliance A
fluent(belong(C,A)).

% alliance A has ideology I
fluent(has(A,I)).

% country C1 is in
% conflict with country C2
fluent(conflict(C1,C2)).

% country C changes to
% a new alliance A
action(change(C,A)).

% symmetry

in_conf(A1,A2):- in_conf(A2,A1),
 A1!=A2.

-in_conf(A1,A2):- -in_conf(A2,A1),
```

```
                A1!=A2.

% impossible to change to an alliance,
% if already belongs to it

:-o(change(C,A),T),h(belong(C,A),T).

% if a country changes to a
% new alliance,it will belong to it

h(belong(C,A),T+1):-o(change(C,A),T).

% a country can belong
% only to one alliance,
% at the same time

-h(belong(C,A1),T):-
           h(belong(C,A2),T),
           A1!=A2.

% an alliance can have
% only one ideology,
% at the same time

-h(has(A,I1),T):-
           h(has(A,I2),T),
           I1!=I2.

% two countries are in conflict,
% if they belong to alliances
% in conflict

h(conflict(C1,C2),T):-
                in_conf(A1,A2),
                h(belong(C1,A1),T),
                h(belong(C2,A2),T),
                C1!=C2,
                A1!=A2.
% DEFAULT: Normally equality and freedom
% alliances are in conflict, if there
% is no evidence of the opposite

in_conf(equality,freedom) :-
       not -in_conf(equality,freedom).
```

Now we state the domain independent part, which may be used with other domains. We define the inertia rule (*normally actions do not affect fluents*) in this part.

```
% INERTIA RULE

h(FL,T+1) :- T<n, h(FL,T),
             not -h(FL,T+1).
-h(FL,T+1) :- T<n, -h(FL,T),
              not h(FL,T+1).
```

6. Formalizing the Story

This is the second part of our knowledge base, i.e., the representation of the particular story. The general knowledge base, created previously, can be used in different stories containing information about ideological conflicts. To illustrate this, let us consider again our main story from section 1 and present a logical representation of it.

```
country(us).
country(bulgaria).
country(russia).

alliance(freedom).
alliance(equality).
alliance(none).

ideology(capitalism).
ideology(comunism).
ideology(other).

% HISTORY OF THE DOMAIN

h(belong(bulgaria,freedom),0).
h(belong(russia,none),0).
h(belong(us,freedom),0).

h(has(freedom,capitalism),0).
h(has(equality,comunism),0).
h(has(none,other),0).

o(change(bulgaria,equality),0).
o(change(russia,equality),0).

% the equality and the freedom
% alliance are not in conflict
% with the none alliance of the
% non-allied countries

-in_conf(equality,none).
-in_conf(freedom,none).
```

The program is implemented in smodels and it runs with the following command:

```
lparse --true-negation conflict.sm |
smodels 0| mkatoms
```

It is useful to download the output formatting program $mkatoms$ from:

```
http://krlab.cs.ttu.edu/~marcy/mkatoms
```

As expected, in the output of this program, we have that, before the change, Bulgaria is in the $freedom$ alliance; after the change Bulgaria is in the $equality$ alliance; and at the end of the story Russia and US are in conflict.

```
h(belong(bulgaria,freedom),0)
h(belong(bulgaria,equality),1)
h(conflict(russia,us),1)
```

7. Related work

A number of formal logical languages were used in the past to formalize similar domains. For instance, in [Mue04], the author uses descrete event calculus to understand real stories involving kidnapping and terrorist attacks. Mueller also develops a commonsense knowledge base, but he uses different techniques from the ones presented in this work.

In addition, other domains have been formalized in ASP, such as the travel domain (see [Gel06]). Although different in first glance, this domain could be related to the ideological conflict domain in cases, where travelling between countries is involved.

Moreover, there is a recent call to build micro-theories using nonmonotonic reasoning. In fact, building micro-theories, learning how to expand them and how to combine them in larger modules, is one of the most interesting challenges we are facing now.

Recently, a group of scientists have been working on an *Influence Net* modeling for analyzing the causal relations of complex situations [RS98]. They use the combination of two established methods of decision analysis: Bayesian inference net analysis originally employed by the mathematical community; and influence diagramming techniques originally employed by operations researchers. In that work, the term conflict includes situations of economic instability, ideological or cultural contrasts, as well as the more traditional political and diplomatic security concerns. The authors state that if these crisis situations are left untended tend towards armed conflict situations that affect the global stability.

8. Conclusions and future work

In this work we automated a particular example by constructing a Knowledge Base consisting of general rules and rules of a particular story. This is a simple version of what is going to be a formalization of ideological conflicts. We illustrated our methodology of using ASP and AL for formalizing commonsense knowledge in a different domain from what have been done before.

Our formalization differs from what we have already in the domain topic. We believe that an ideological conflict may lead to a war of ideology between countries, where one country is trying to impose a new ideology on the other country, which is a global situation that need to be solved.

Currently, we are working on the expansion of the general knowledge base. Our future work is to extend our commonsense knowledge base with knowledge necessary to reason about more countries and more complex stories. The ideal goal is to have one general knowledge base, containing common-senses and expert knowledge about various domains.

9. Acknowledgments

The author would like to thank Michael Gelfond and Marcello Balduccini for their useful discussions on the subject of this paper. She is also grateful to ARDA for supporting this research (ARDA / DTO contract).

References

[Bar03] C. Baral. *Knowledge Representation, reasoning and declarative problem solving with Answer Sets*. Cambridge University Press, 2003.

[BM03] M. Balduccini and M.Gelfond. Diagnostic reasoning with a-prolog. *Theory and Practice of Logic Programming*, 2003.

[GB00] M. Gelfond and C. Baral. Reasoning agents in dynamic domains. *Logic Based Artificial Intelligence*, 2000.

[GBS04] M. Gelfond, C. Baral, and Richard Scherl. Using answer set programming to answer complex queries. *In Workshop on Pragmatics of Question Answering at HLT-NAAC2004*, 2004.

[Gel02] M. Gelfond. Representing knowledge in a-prolog. *Computational Logic: Logic Programming and Beyond, Essays in Honour of Robert A. Kowalski*, 2408, Part II:413–451, 2002.

[Gel06] M. Gelfond. Going places - notes on a modular development of knowledge about travel. *In AAAI Spring 2006 Symposium*, 2006.

[GL88] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. *5th Conference on Logic Programming*, pages 1070–1080, 1988.

[GL91] M. Gelfond and V. Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9:365–385, 1991.

[IC03] IC. Advanced research and development activity. http://www.ic-arda.org/, 2003.

[Mue04] E. Mueller. Understanding script-based stories using commonsense reasoning. *Cognitive Systems Research*, 5(4):307–340, 2004.

[RS98] J. Rosen and W. Smith. Influencing global situations. *Air and Space Power Chronicles of the Chronicles Online Journal*, 1998.

[SS00] P. Simons and T. Syrjnen. Smodels-an implementation of the stable model semantics for logic programs. http://www.tcs.hut.fi/Software/smodels/, 2000.