Satisfiability for First-order Logic as a Non-Modal Deontic Logic

Robert Kowalski
Imperial College London, United Kingdom
rak@doc.ic.ac.uk
17 August 2017

Abstract. In modal deontic logics, the focus is on inferring logical consequences, for example inferring whether an obligation $O \text{mail}$, to mail a letter, logically implies $O [\text{mail} \vee \text{burn}]$, an obligation to mail or burn the letter. Here I present an alternative approach in which obligations are sentences (such as $\text{mail}$) in first-order logic (FOL), and the focus is on satisfying those sentences by making them true in some best model of the world. To facilitate this task and to make it manageable, candidate models are defined by a logic program (LP) extended by means of candidate action assumptions (A). The resulting combination of FOL, LP and A is a variant of abductive logic programming (ALP).

1 Goal satisfaction in FOL

In the abductive logic programming (ALP) approach to reasoning about obligations of [8, 10], candidate assumptions (A), representing actions and other “abducibles”, and logic programs (LP), representing an agent’s beliefs, are combined with first-order logic (FOL), representing an agent’s goals. However, this characterisation of ALP is potentially misleading, because it fails to identify the primary role of goals, and the supporting role of beliefs and assumptions in helping to make goals true. Here is a more abstract characterisation, which is formulated entirely in terms of FOL, and does not mention A or LP at all:

A goal satisfaction problem is a tuple $\langle G, M_0, W \rangle$ where:

$G$ is a set of sentences in FOL, representing goals.

$M_0$ is a classical FOL model-theoretic structure, representing a partial history of the world.

$W$ is a set of classical FOL model-theoretic structures, representing alternative extensions of $M_0$.

$M \in W$ satisfies a goal satisfaction problem $\langle G, M_0, W \rangle$ if and only if $G$ is true in $M$.

We will see that the models $M_0$ and $W$ can be defined constructively, by using a logic program $P$ to define $M_0$, and by using a set of candidate actions $A$ to specify alternative ways of extending $M_0$. We will also see that it is possible to satisfy goals without generating complete models, by using backward reasoning.

There can be many models that satisfy the same goal satisfaction problem, and some models may be better than others. Usual formulations of ALP ignore this fact, and are neutral with respect to the choice between different models satisfying the same goals. In contrast, in deontic logic, an agent is obliged to satisfy its goals in the best way possible.

To capture this normative character of deontic logic in FOL/ALP, we introduce a partial ordering between the models in $W$. 
A normative goal satisfaction problem is a tuple \( \langle G, M_0, W, < \rangle \) where:

\[< \text{ is a strict partial ordering over } W, \]
where \( M < M' \) represents \( M' \) being better than \( M \).

\( M \in W \) satisfies a normative goal satisfaction problem \( \langle G, M_0, W, < \rangle \) if and only if

\( M \) satisfies the goal satisfaction problem \( \langle G, M_0, W \rangle \) and
there does not exist \( M' \in W \) such that \( M' \) satisfies \( \langle G, M_0, W \rangle \) and \( M < M' \).

There is an obvious relationship with the possible world semantics of modal logics: \( W \) is like a frame of possible worlds. The extension of \( M_0 \) to \( M \) is like the accessibility relation between possible worlds. The partial ordering \( < \) is like the preference relation in preference-based deontic logics, such as those of [5] and [14].

However, whereas preference-based deontic logics build the preference relation into the semantics, here the partial ordering is external to the logic, which is simply FOL. Moreover, while in modal logics, actions and events are normally represented by labels on the accessibility relation, here they are “reified”, as part of the record of a partial history of the world.

Deontic logic, the logic of obligation, contrasts with alethic logic, the logic of necessity. In alethic logic, a necessary truth cannot be falsified. But in deontic logic, an obligation is a normative ideal. If an obligation is violated, it does not cause the end of the world, but it results in a world that is less than ideal.

The focus in modal logics on deriving logical consequences makes it difficult to deal with violations, conflicting norms, and contrary-to-duty obligations. In contrast, the focus in FOL/ALP on goal satisfaction turns these problems into pragmatic choices between alternative models. Moreover, it makes it possible for an agent, aspiring towards the normative deal, to fall short, but nonetheless succeed in generating a best model possible with the limited resources available.

2 Logic programs as constructive definitions of models.

These definitions of goal satisfaction imply that, for an agent to satisfy its goals \( G \), whether in some best way or not, the agent needs to search the space of alternative world histories \( W \), to find some model \( M \in W \) of \( G \). For this to be feasible in practice, the models \( M \in W \) must be constructible, and the space \( W \) itself must be searchable. This is where ALP comes in. Logic programs in ALP provide a constructive representation of models, and an efficient way to guarantee truth in a model without necessarily generating the model in its totality.

In our variant of ALP, the given model \( M_0 \) is defined by a logic program \( P \), and the space of candidate models \( M \in W \) is defined by logic programs \( P \cup \Delta \), where \( \Delta \subseteq A \) and \( A \) is a set of ground atomic sentences representing candidate assumptions.

In ordinary abduction, the goals \( G \) represent observations, and \( A \) represents candidate hypotheses that can be used to explain \( G \). In default reasoning, \( A \) represents assumptions that conditions are normal, and \( G \) represents constraints that ensure conditions are not assumed to be normal if they are exceptional. In deontic applications, \( G \) represents obligations and prohibitions, and \( A \) represents candidate actions that can be used to satisfy \( G \).

In the simplest case, a logic program is a set of definite clauses of the form \( \text{conclusion} \leftarrow \text{condition}_1 \land \ldots \land \text{condition}_n \), where \( \text{conclusion} \) and each \( \text{condition}_i \) is an atomic formula. All variables are universally quantified with scope the entire clause. Every such set of definite clauses \( P \) has a unique minimal model \( M \) [3]. The minimal model \( M \) can be viewed as the intended model of \( P \), and \( P \) can be regarded as a constructive definition of \( M \).

In LP, it is convenient to represent models \( M \) as Herbrand interpretations, which are sets of atomic sentences representing all the atomic sentences that are true in \( M \). A Herbrand interpretation \( M \) of a logic program \( P \) is then a minimal model of \( P \) if
and only if $M \subseteq M'$ for any other Herbrand model $M'$ of $P$. Minimal models can be constructed by forward reasoning: using modus ponens to exhaustively derive atomic sentences that are instances of the conclusions of clauses from atomic sentences that are instances of the conditions of clauses.

Forward reasoning can also be used to satisfy a goal: guessing a set of candidate assumptions $\Delta \subseteq A$, adding them to $P$, generating the minimal model $M$ of $P \cup \Delta$, and then checking whether $M$ is also a model of $G$. However, in the general case this kind of reasoning is not computationally feasible. It is not feasible to generate candidate $\Delta$ blindly and independently of $G$; nor is it feasible to generate models in their totality.

In contrast, backward reasoning using SLD resolution [9] is computationally feasible. SLD resolution reasons backwards from $G$, reducing goals that match the conclusion of a clause in $P$ to subgoals that are the instantiated conditions of the clause. It continues this process of goal-reduction, until all subgoals are solved directly either by atomic sentences in $P$ or by assumptions in $A$. In this way, backward reasoning generates only assumptions $\Delta$ that are relevant to satisfying $G$. Moreover, it does so by generating only the assumptions $\Delta$, which together with $P$ determine the minimal model of $P \cup \Delta$.

In the following three, simple examples, the logic program $P$ is either an empty set of clauses $\{}$ or a singleton set containing the clause $X = X$. Backward reasoning generates only those assumptions/actions in $A$ or those instances of $X = X$ that are needed to satisfy the goal. Moreover, in the first two examples, the preference relation $\{}$ is empty. So all models are equally good.

3 Map colouring as goal satisfaction

The classic map colouring problem illustrates the use of FOL/ALP for goal satisfaction. However, it can also be formulated in deontic terms:

- It is obligatory that every country is painted with a colour.
- It is forbidden to paint the same country two different colours.
- It is forbidden to paint two adjacent countries the same colour.

The problem can be stated as a goal $G$ in FOL:

\[
\{ \forall X [\text{country}(X) \rightarrow \exists C [\text{colour}(C) \land \text{paint}(X, C)]], \\
\forall X C1 C2 [\text{paint}(X, C1) \land \text{paint}(X, C2) \rightarrow C1 = C2], \\
\forall X Y C \neg [\text{adjacent}(X, Y) \land \text{paint}(X, C) \land \text{paint}(Y, C)] \}
\]

The map can be represented by an initial model $M_0$, defined by a logic program $P$ that specifies the countries, the colours and the adjacency relation. For a simple problem with two adjacent countries and two colours, this is given by the following program $P$, which also includes a definition of the identity relation:

\[
\{ \text{country}(iz), \text{country}(oz), \text{adjacent}(iz, oz), \text{colour}(red), \text{colour}(blue), \ X = X \}
\]

Because the program $P$ is so simple, the only difference between $P$ and its minimal model $M_0$ is that $P$ contains a general clause $X = X$, whereas $M_0$ contains all variable free instances $iz = iz$, $oz = oz$, $red = red$, $blue = blue$ of the clause.

The alternative ways of colouring the two countries are given by the set of candidate actions $A$:

\[
\{ \text{paint} (iz, red), \text{paint} (iz, blue), \text{paint} (oz, red), \text{paint} (oz, blue) \}
\]
There are exactly two models that satisfy the goals G; and, given no restrictions on the preference relation, both are equal good:

\[
M_1 = M_0 \cup \{ \text{paint (iz, red), paint (oz, blue)} \} \quad \text{and} \\
M_2 = M_0 \cup \{ \text{paint (iz, blue), paint (oz, red)} \}
\]

The map colouring problem can also be expressed in modal deontic logic, formalising the English statement of the problem. It would then be possible to infer the following logical consequence:

\[
\begin{align*}
O [\text{paint (iz, red) \& paint (oz, blue)}] & \vee \\
O [\text{paint (iz, blue) \& paint (oz, red)}]
\end{align*}
\]

which describes all solutions of the problem. It is not obvious how to infer a single solution, which would be more useful in practice.

4 Ross’s Paradox

SDL (Standard Deontic Logic) is commonly used as a basis for comparison between different deontic logics. It is a propositional logic with a modal operator O representing obligation. Among the many problems of SDL, which also affects many other deontic logics, is Ross’s Paradox [12].

It is obligatory that the letter is mailed.
If the letter is mailed, then the letter is mailed or the letter is burned.

i.e. 
O mail, mail \rightarrow mail \vee burn.

In SDL it is a logical consequence of O mail that O [mail \vee burn].
As McNamara [11] puts it, “it seems rather odd to say that an obligation to mail the letter entails an obligation that can be fulfilled by burning the letter (something presumably forbidden), and one that would appear to be violated by not burning it if I don’t mail the letter”.

The “Paradox” can be understood as another example of the inadequacy of logical consequence for dealing with problems of satisfying obligations. Here is a formulation of the paradox as a goal satisfaction problem (G, M₀, W, <):

\[
\begin{align*}
G &= \{ mail, \neg burn \} \\
M_0 &= \{ \} \\
W &= \{ M_0 \cup \Delta \mid \Delta \subseteq A \} \quad \text{where} \quad A = \{ \text{mail, burn} \} \\
&= \{ \{ \}, \{ \text{mail} \}, \{ \text{burn} \}, \{ \text{mail, burn} \} \} \\
< &= \{ \}
\end{align*}
\]

P = { } and M₀ is the minimal model of P.

M = {mail} is the only minimal model that satisfies G. But mail \vee burn is true in M. So satisfying G, entails satisfying mail \vee burn. But, contrary to suggestions that may be associated with the fact that O [mail \vee burn] is a logical consequence of O mail, satisfying the goal mail \vee burn does not satisfy the goal mail.

Viewed in this way, Ross’s Paradox is not a paradox at all, but rather, as Fox [4] also argues, a confusion between satisfying an obligation and implying that one obligation is a logical consequence of another.

5 Chisholm’s Paradox
Ross’s Paradox and the map colouring problem do not need preferences between alternative models. However, the need for preferences arises with Chisholm’s Paradox [2]:

\begin{itemize}
  \item It \textit{ought} to be that Jones goes to assist his neighbours.
  \item It \textit{ought} to be that, if Jones goes, then he tells them he is coming.
  \item If Jones doesn't go, then he \textit{ought not} tell them he is coming.
\end{itemize}

Jones doesn't go.

\textit{i.e.}

\begin{align*}
  &\text{O } \text{go} \\
  &\text{O } (\text{go } \rightarrow \text{tell}) \\
  &\neg\text{go } \rightarrow \text{O } \neg\text{tell} \\
  &\neg\text{go}
\end{align*}

Much of the discussion [1] in the deontic logic literature concerns the problems that arise with alternative representations of the conditional obligations in the second and third sentences. I will not repeat this discussion here, but will present the example as a normative goal satisfaction problem \( \langle G, M_0, W, < \rangle \):

\begin{align*}
  G &= \{ \text{go } \rightarrow \text{tell}, \ \neg\text{go } \rightarrow \neg\text{tell} \} \\
  M_0 &= \{ \} \\
  W &= \{ M_0 \cup \Delta \mid \Delta \subseteq A \} \text{ where } A = \{ \text{go, tell} \} \\
  &= \{ \{ \}, \{ \text{go} \}, \{ \text{tell} \}, \{ \text{go, tell} \} \} \\
  M < M' \text{ if } \text{go} \not\in M \text{ and } \text{go} \in M'.
\end{align*}

Here the “obligation” for Jones to \textit{go} is represented by the preference for models in which \textit{go} is true over models in which \textit{go} is false. There are two models \( M_1 = \{ \} \) and \( M_2 = \{ \text{go, tell} \} \) that make \( G \) true. But \( M_2 \) is better than \( M_1 \). So only \( M_2 \) satisfies \( \langle G, M_0, W, < \rangle \). This means that Jones must \textit{go} and \textit{tell}.

Now suppose that Jones doesn’t go. We can represent this simply by removing \textit{go} from the candidate actions \( A \), updating the problem to \( \langle G, M_0, W', < \rangle \) where \( W' = \{ \{ \}, \{ \text{tell} \} \} \). The only (and best) candidate model that now satisfies the updated problem is the less than ideal model \( M_1 = \{ \} \), which means that Jones must \textit{not} \textit{tell}, as is intuitively correct.

In a more realistic representation with explicit time, when we discover that Jones doesn’t go, we would update \( M_0 \) to a new world history in which \textit{going} is no longer an option. Notice that, in any case, whether we update \( M_0, W \) or both, the solution of the updated problem changes, even though the goals remain the same. This gives the satisfiability problem for FOL a non-monotonic character, even though logical consequence in FOL is monotonic.

\section{6 LPS (Logical Production System)}

The logic and computer language LPS [6, 7], is a scaled-down implementation of ALP, intended for practical applications of goal satisfaction. Goals in LPS have the simplified form of rules \textit{antecedent } \rightarrow \textit{consequent}, which are a logical reconstruction of condition-action rules in production systems [13]. All variables in the \textit{antecedent} of a rule are universally quantified with scope the entire rule, and all variables in the \textit{consequent} but not in the \textit{antecedent} are existentially quantified with scope the \textit{consequent} of the rule. Variables include time variables, and all times in the \textit{consequent} are later than or equal to all times in the \textit{antecedent}.

Computation in LPS observes external events and executes actions, generating a sequence \( M_0 \subseteq \ldots M_{i+1} \subseteq M_i \subseteq \ldots \) of partial histories, which in the limit determines a model \( M = M_0 \cup \ldots M_{i+1} \cup M_i \cup \ldots \) which makes all the goals true. Each history \( M_i \) is obtained from the previous history \( M_{i+1} \) by updating \( M_{i+1} \) with the set of all
external events and actions that take place between $M_{i-1}$ and $M_i$. These updates are performed destructively, like change of state in the real world, and like destructive updates in imperative computer languages.

Computation in LPS gives rules both a logical and imperative interpretation. Whenever any instance of an antecedent of a rule becomes true in some history $M_i$, forward reasoning treats the corresponding instance of the consequent of the rule as a command to perform actions, to make that instance of the consequent true in some future history $M_j, j \geq i+1$.

Clauses conclusion $\leftarrow$ conditions in LP also have both a logical and imperative interpretation. In addition to their purely logical interpretation, they have an imperative interpretation as procedures, which use backward reasoning to decompose a goal of determining whether a conclusion is true or of making a conclusion true to the subgoal of determining or making the conditions true. Conditions representing actions are made true by adding them to future histories generated by clauses written later. There is also an in-built preference for models that satisfy goals as soon as possible.

An online implementation of LPS is accessible from http://lps.doc.ic.ac.uk/. The LPS examples notebook in the examples menu contains executable links to the map colouring problem, and to several other examples having a deontic interpretation. The first steps notebook presents an example in which two agents have conflicting goals. Agent bob wants the light on whenever he is in a room, and agent dad wants the light off whenever there is a room in which a light is on. It may be natural to think of bob’s goal as a personal goal, and dad’s goal as an obligation. But LPS makes no distinction between the two kinds of goals.

7 Conclusions

In this short paper, I have formulated deontic reasoning in ALP as goal satisfaction in FOL, with the A and LP components of ALP used to define the space of candidate models. I have argued that backward reasoning with LP overcomes the need to generate complete models, and makes it possible to avoid generating candidate actions blindly without relevance to the goals that need to be satisfied. I have also argued that, for philosophical applications, the focus on goal satisfaction in ALP is more useful than the focus on logical consequence in modal deontic logics.

References


