Alan Smaill
24 Feb 2006

KRE Second Assessed Practical

This practical involves implementing a decision procedure for a simple modal logic.

1 The Logic

The logic in question is given by the following axioms and rules of inference.

The syntax uses the notion of sequent, *ie* the statements at any point in the derivation take the form a *sequent*

$$\texttt{Set\_of\_Formulas} \Longrightarrow \texttt{Set\_of\_Formulas}$$

indicating that *some* formula on the right follows from *all* of the assumptions of the left.

Derivations take the form of a numbered list of such assertions, labeled with a justification. The justification is one of:

- That the assertion is an axiom.
- That the assertion follows from previous assertions by an inference rule.

There is also a modal operator, $\Box$, built in to the logic (so this is a *modal logic*). $\Box \mathtt{P}$ is interpreted as "P is necessarily true".

There are two rules for each connective, one when it appears on the right, and one when it appears on the left of the sequent (in the conclusion).

Axioms and Rules of Inference

(axiom) $\mathtt{G}, \mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{G}, \mathtt{G}_1, \ldots, \mathtt{G}_m$ (Axiom)

(andI) $$\frac{\mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{F}, \mathtt{G}_1, \ldots, \mathtt{G}_m \qquad \mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{G}, \mathtt{G}_1, \ldots, \mathtt{G}_m}{\mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{F} \wedge \mathtt{G}, \mathtt{G}_1, \ldots, \mathtt{G}_m}$$

(andE) $$\frac{\mathtt{F}, \mathtt{G}, \mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{C}}{\mathtt{F} \wedge \mathtt{G}, \mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{C}}$$

(impI) $$\frac{\mathtt{F}, \mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{G}, \mathtt{G}_1, \ldots, \mathtt{G}_m}{\mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{F} \to \mathtt{G}, \mathtt{G}_1, \ldots, \mathtt{G}_m}$$

(impE) $$\frac{\mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{F}, \mathtt{C}_1, \ldots, \mathtt{C}_m \qquad \mathtt{G}, \mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{C}_1, \ldots, \mathtt{C}_m}{\mathtt{F} \to \mathtt{G}, \mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{C}_1, \ldots, \mathtt{C}_m}$$

(orI) $$\frac{\mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{F}, \mathtt{G}, \mathtt{G}_1, \ldots, \mathtt{G}_m}{\mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{F} \vee \mathtt{G}, \mathtt{G}_1, \ldots, \mathtt{G}_m}$$

(orE) $$\frac{\mathtt{F}, \mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{C}_1, \ldots, \mathtt{C}_m \qquad \mathtt{G}, \mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{C}_1, \ldots, \mathtt{C}_m}{\mathtt{F} \vee \mathtt{G}, \mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{C}_1, \ldots, \mathtt{C}_m}$$

$$\text{(notI)}\ \frac{\mathtt{F}, \mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{C}_1, \ldots, \mathtt{C}_m}{\mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \neg\mathtt{F}, \mathtt{C}_1, \ldots, \mathtt{C}_m}$$

$$\text{(notE)}\ \frac{\mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{F}, \mathtt{G}_1, \ldots, \mathtt{G}_m}{\neg\mathtt{F}, \mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{G}_1, \ldots, \mathtt{G}_m}$$

$$\text{(boxI)}\ \frac{\mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{G}, \mathtt{P}_1, \ldots, \mathtt{P}_m}{\Box\mathtt{F}_1, \ldots \Box\mathtt{F}_n, \mathtt{G}_1 \ldots \mathtt{G}_m \Longrightarrow \Box\mathtt{G}, \Box\mathtt{P}_1, \ldots \Box\mathtt{P}_m, \mathtt{H}_1, \ldots, \mathtt{H}_k}$$

$$\text{(boxE)}\ \frac{\mathtt{F}, \mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{H}_1, \ldots, \mathtt{H}_k}{\Box\mathtt{F}, \mathtt{F}_1, \ldots, \mathtt{F}_n \Longrightarrow \mathtt{H}_1, \ldots, \mathtt{H}_k}$$

An implementation of these rules via Prolog predicates can be found on the `practicals` section of the course web page, together with some utilities for manipulating an appropriate syntax.

2 Searching for derivations

The first task is to write a decision procedure for this logic, using the full search space generated by the inference rules above used backwards. A *decision procedure* for this problem is an algorithm which always terminates, and whose result correctly indicates whether or not a given sequent has a derivation in this logic.

In each case, the procedure when successful should return a tree of rule names, where the rules are those used in building the derivation. The syntax for this is:

Derivation ::= **axiom** | Rule **then** Derivation list.

For example, $[\,] \Longrightarrow \mathtt{p} \to (\mathtt{q} \to (\mathtt{p} \wedge \mathtt{q}))$ has a derivation described by

```
impI then [impI then [andI then [axiom,axiom]]].
```

You should use the procedures supplied to you as a starting point; there are several possibilities for the search strategy to use here; Ideally, you should consider what might be a good *ordering* of the rules to test for applicability, so as to prefer smaller derivations to larger ones (as measure by the number of sequents appearing in the derivation).

Credit will be given for a clearly described algorithm, even in the absence of fully complete code.

Examples

Translate the following examples into modal logic notation, and generate derivations for them, where they are derivable. The starting point should be a sequent with the given statements on the left, and the query on the right of the sequent. You should probably use *short* predicate and constant names in your translation.

1. If it's Saturday. then if it's cold, there is a football match. If it's Saturday, then it's cold. It's Saturday.

   Is there a football match?

2. It's necessarily not Saturday. Necessarily, if it's sunny, then it's Saturday, and it's hot.

   Is it necessarily necessarily the case that it's not sunny?

3. It's necessarily not both hot and snowing.
   If it's necessarily hot, then it's May.
   If it's necessarily the case that if it's winter, then it's winter, then it's hot.
   It's necessarily the case that if it's hot, then it's snowing.

   Is it May?

Justification of the algorithm

Does your procedure constitute a decision procedure for the existence of a derivation for a given query in this logic? Justify your answer carefully.

3 Submission

Use the `submit` procedure to submit a file containing your program, test results, and justification of the algorithm.

The deadline for submissions is Wednesday 15th March.