

1 PTTP- Prolog Technology Theorem Proving

As exemplified by PTTP (“Prolog Technology Theorem Prover”) [?, ?], Prolog can be viewed as an “almost complete” theorem prover, which has to be extended by only a few ingredients in order to handle the non-Horn case. By this technique the benefits of optimizing Prolog compilers are accessible to theorem proving. First we will briefly review the standard approach, and then we will describe the necessary modifications to obtain restart model elimination.

The PTTP-approach transforms a given clause set into a Prolog program. The transformed Prolog program must execute the clauses according to some complete proof procedure. *Model elimination* turns out to be particularly useful for this, since it is, like Prolog, an input proof procedure. In particular, the transformation from the input clauses to Prolog works as follows:

- An input clause such as

$$C \leftarrow A \wedge B$$

is transformed into a Prolog clause

$$(1) \quad c :- a, b.$$

Additionally, since in the model elimination calculus every literal in a clause can equally well serve as an entry point into the clause, all contrapositives are needed. In this case these are

$$(2) \quad \text{not_a} :- \text{not_c}, b.$$

$$(3) \quad \text{not_b} :- a, \text{not_c}.$$

This example also shows how negation is treated, namely by making it part of the predicate name.

- Prolog’s unsound unification has to be replaced by a sound unification algorithm. This can either be done by directly building-in sound unification into the Prolog implementation, or by reprogramming sound unification in Prolog and calling this code instead of Prolog’s unsound unification.
- A complete search strategy is needed. Usually depth bounded iterative deepening is used. The strategy can be compiled into the prolog program by additional parameters, being used as “current depth” and “limit depth”. The cost of an extension step can be uniformly 1 (depth bounded search), or can be proportional to the length of the input clause (inference bounded search).
- The model elimination reduction operation has to be implemented. This can be realized by memorizing the subgoals solved so far (the A-literals) as a list in an additional argument, and by Prolog code that checks a goal for a complementary member of that list. Of course, this check has to be carried out with sound unification.

The Prolog clause (1) from above then looks like

$$(1') \quad c(\text{Anc}) :- a([-a|\text{Anc}]), b([-b|\text{Anc}]).$$

where `Anc` is a Prolog list which contains the ancestor literals (called A-literals in [?]); code for reduction steps then looks like

```
(Red-C)      c(Anc) :- member(c,Anc).
(Red-notC)  not_c(Anc) :- member(-c,Anc).
```

Modal Logic

Modal logic is concerned with the investigation of modalities in logics. Modalities are e.g. *necessity*, *possibility*, *provability* or *uncertainty*. In classical propositional logics, propositions like *married(tom,mary)* are *true* or *false*; in real life however, the truthvalue of the above proposition obviously can change in time. In a different context the truthvalue may be different in different worlds: assume that Tom is dreaming about being married with Mary; hence in the world of his dreams the proposition may be true, while in real-life it may be false. In still another context, the truthvalue can depend whether it is considered under legal aspects: it is possible that Tom and Mary are legally married, while the catholic church is considering them to be singles.

Modal logics have been studied extensively already during the first half of the 20th century by various logicians. The main breakthrough, however, was the establishment of a formal semantics of modal logic given by Kripke.

Propositional logics can be extended by modalities to describe *believe*, *knowledge* or *temporal aspects*. Hence it is very appropriate to be used with knowledge representation systems. Recently modal logics are applied in verification contexts and as a means to describe the semantics of description logics.

As an example take the following puzzle:

Assume 3 children (works for n as well), who are perfect reasoners, are always truthful, and always given an answer, if they know one. The children are playing outside, and may get muddy foreheads. Any child can see if the other children have muddy foreheads, but can't see his or her own forehead. At any point, an adult says to them: at least one of you has mud on your forehead.

The adult now says: Do any of you know if you have mud on your forehead. No one answers. The adult repeats the question, and again no one answers. The 3rd time the adult asks the question, one or more of the children answer.

How many of the children have muddy foreheads?

This puzzle can be solved by constructing a Kripke structure, which is a set of states together with links which express the accessibility of states. There are 3 children, each can be muddy or not muddy and, hence, we have 2^3 possible states. States can be represented by triple of booleans, where a 1 in the n th position means that the n th child is muddy, and a 0 in the n th position that the n th child is not muddy.

The Kripke structure can be defined as follows: consider, e.g. state (111). Since child 1 does not know whether or not he or she is muddy, as far as child 1 is concerned, he or she could be in state (011). For child 2 the state (011), however, is not accessible, since child 2 knows that child 1 is muddy.

A structure which is constructed as depicted above, can be used to solve the puzzle, by drawing the structure as it results after each speech action of the adult.

Syntax

We assume the syntax of classical propositional logic as used in the chapters above. Additionally we have the following two rules:

If A is a formula

$$\begin{array}{l} \diamond A \text{ and} \\ \square A \end{array}$$

are formulae.

The symbols \diamond and \square stand traditionally for *possibility* and *necessity*; in the context of temporal logic they stand for *always* and *eventually*, so that $\diamond A$ stands for *A is eventually true* and $\square A$ for *A is always true*.

Kripke Semantics

Definition 1 A Kripke frame is a pair $\langle W, R \rangle$, where W is a non-empty set (of possible worlds) and R a binary relation on W . We write wRw' iff $(w, w') \in R$ and we say that world w' is accessible from world w , or that w' is reachable from w , or that w' is a successor of w .

A Kripke model is a triple $\langle W, R, V \rangle$, with W and R as above and V is a mapping $\mathcal{P} \mapsto 2^W$, where \mathcal{P} is the set of propositional variables. $V(p)$ is intended to be the set of worlds at which p is true under the valuation V .

Given a model $\langle W, R, V \rangle$ and a world $w \in W$, we define the satisfaction relation \models by:

$$\begin{array}{ll} w \models p & \text{iff } w \in V(p) \\ w \models \neg A & \text{iff } w \not\models A \\ w \models A \wedge B & \text{iff } w \models A \text{ and } w \models B \\ w \models A \vee B & \text{iff } w \models A \text{ or } w \models B \\ w \models A \rightarrow B & \text{iff } w \not\models A \text{ or } w \models B \\ w \models \square A & \text{iff for all } v \in W, (w, v) \notin R \text{ or } v \models A \\ w \models \diamond A & \text{iff there exists some } v \in W, \text{ with } (w, v) \in R \text{ and } v \models A \end{array}$$

We say that w satisfies A iff $w \models A$ (without mentioning the valuation V). A formula A is called satisfiable in a model $\langle W, R, V \rangle$, iff there exists some $w \in W$, such that $w \models A$. A formula A is called satisfiable in a frame $\langle W, R \rangle$, iff there exists some valuation V and some world $w \in W$, such that $w \models A$.

A formula A is called valid in a model $\langle W, R, V \rangle$, written as $\langle W, R, V \rangle \models A$ iff it is true at every world in W . A formula A is valid in a frame $\langle W, R \rangle$, written as $\langle W, R \rangle \models A$ iff it is valid in all models $\langle W, R, V \rangle$.

Lemma 1 The operators \diamond and \square are dual, i.e. for all formulae A and all frames $\langle W, R \rangle$, the equivalence $\diamond A \leftrightarrow \neg \square \neg A$ holds.

Axiomatics

The simplest modal logic is called K and is given by the following axioms:

- All classical tautologies (and substitutions thereof)
- Modal Axioms: All formulae of the form $\Box(X \rightarrow Y) \rightarrow (\Box X \rightarrow \Box Y)$

and the inference rules

- Modus Ponens Rule: Conclude Y from X and $X \rightarrow Y$
- Necessitation Rule: Conclude $\Box X$ from X

A K derivation of X from a set S of formulae is a sequence of formulae, ending with X , each of it is an axiom of K , a member of S or follows from earlier terms by application of an inference rule. A K proof of X is a K derivation of X from \emptyset .

As an example take the K proof of $(\Box P \wedge \Box Q) \rightarrow \Box(P \wedge Q)$:

Tautology:	$P \rightarrow (Q \rightarrow (P \wedge Q))$
Necessitation:	$\Box(P \rightarrow (Q \rightarrow (P \wedge Q)))$
Modal axiom:	$\Box(P \rightarrow (Q \rightarrow (P \wedge Q))) \rightarrow (\Box P \rightarrow \Box(Q \rightarrow (P \wedge Q)))$
Modus ponens:	$\Box P \rightarrow \Box(Q \rightarrow (P \wedge Q))$
Modal axiom:	$\Box(Q \rightarrow (P \wedge Q)) \rightarrow (\Box Q \rightarrow \Box(P \wedge Q))$
Classical arg:	$\Box P \rightarrow (\Box Q \rightarrow \Box(P \wedge Q))$
Classical arg:	$(\Box P \wedge \Box Q) \rightarrow \Box(P \wedge Q)$

There is a similar proof of the converse of this implication; hence it follows that in K we have

$$\Box(P \wedge Q) \leftrightarrow (\Box P \wedge \Box Q)$$

Note that distributivity over disjunction does not hold! (Why?)

Extensions of K

Starting from the modal logic K one can add additional axioms, yielding different logics. We list the following basic axioms:

$$K : \Box(X \rightarrow Y) \rightarrow (\Box X \rightarrow \Box Y)$$

$$T : \Box A \rightarrow A$$

$$D : \Box A \rightarrow \Diamond A$$

$$4 : \Box A \rightarrow \Box \Box A$$

$$5 : \Diamond A \rightarrow \Box \Diamond A$$

$$B : A \rightarrow \Box \Diamond A$$

Traditionally, if one adds axioms A_1, \dots, A_n to the logic K one calls the resulting logic $KA_1 \dots A_n$. Sometimes, however the logic is so well known, that it is referred to under another name; e.g. $KT4$, is called $S4$.

These logics can as well be characterised by certain classes of frames, because it is known that particular axioms correspond to particular restrictions on the reachability relation R of the frame. If $\langle W, R \rangle$ is a frame, then a certain axiom will be valid on $\langle W, R \rangle$, if and only if R meets a certain restriction. Some restrictions are expressible by first-order logical formulae where the binary predicate $R(x, y)$ represents the reachability relation:

- T : Reflexive $\quad \forall w : R(w, w)$
- D : Serial $\quad \forall w \exists w' : R(w, w')$
- 4 : Transitive $\quad \forall s, t, u : (R(s, t) \wedge R(t, u)) \rightarrow R(s, u)$
- 5 : Euclidean $\quad \forall s, t, u : (R(s, t) \wedge R(s, u)) \rightarrow R(t, u)$
- B : Symmetric $\quad \forall w, w' : R(w, w') \rightarrow R(w', w)$

Multimodal Logics – An example

If modal logics is to be used for expressing the knowledge of various agents, one needs operators which are parametrised: we introduce box operators $[a]$, which are parametrised by arbitrary terms a ; the same holds for $\langle a \rangle$, the parametrised diamond operator.

A king, wishing to know which of his three advisers is the wisest, paints a white spot on each of their foreheads, tells them the spots are black or white and that there is at least one white spot, and asks them to tell him the color of their own spots. After a time the first wise man says, “I do not know whether I have a white spot.” The second, hearing this, also says he does not know. The third (truly!) wise man then responds, “My spot must be white.”

The following is a formalisation using the the abbreviation $\Box F$, which stands for arbitrary nested parametrised \Box -operators. If we had only 2 wise man A and B , $\Box F$ would stand for $[A]F \wedge [B]F \wedge [A][B]F \wedge [B][A]F \wedge [A][A]F \dots$. Hence in general $\Box F$ stands for “ F is generally known”.

- $B \neq A \wedge C \neq A \wedge C \neq B$ the three wise are different
- $\Box(w(A) \vee w(B) \vee w(C))$ it is generally known, that one of them has a white spot
- $\Box(\forall x, y : x \neq y \rightarrow (\neg w(x) \rightarrow [y]\neg w(x)))$ it is generally known, that if someone has no white spot, the others know it
- $[C][B]\neg[A]w(A)$ C knows, that B knows, that A does not know the colour of his spot
- $[C]\neg[B]w(B)$ C knows, that B does not know the colour of his spot

The theorem to be proven is “ C knows that he has a white spot”:

$$[C]w(c)$$

If we had a theorem prover for modal logics we could apply it to the above specified problem in order to obtain a proof and hence an explanation for the theorem.

In the rest of this section we will introduce two methods for the definition of such a theorem prover: A direct tableau calculus and a translation method in to first order classical predicate logic.

Modal Logic Tableaux

In classical propositional logics we introduced a tableau calculus (cf Definition) for a logic L as a finitely branching tree whose nodes are formulas from L .

Given a set Φ of formulae from L , a tableau for Φ was constructed by a (possibly infinite) sequence of applications of a tableau rule schema:

$$\rho \frac{\psi}{D_1 \mid D_2 \mid \cdots \mid D_n}$$

where the premise ψ as well as the denominators D_1, \dots, D_n are sets of formulae; ρ is the name of the rule.

We introduce K -tableau with the help of the following rules:

$$(\wedge) \frac{X; P \wedge Q}{X; P; Q} \quad (\vee) \frac{X; \neg(P \wedge Q)}{X; \neg P; \mid X; \neg Q}$$

$$(\perp) \frac{X; P; \neg P}{\perp} \quad (\neg) \frac{X; \neg\neg P}{X; P}$$

$$(\theta) \frac{X; Y}{X} \quad (K) \frac{\Box X; \neg\Box P}{X; \neg P}$$

A tableau for a set X of formulae is a finite tree with root X whose nodes carry finite formulae sets. The rules for extending a tableau are given by:

- choose a leaf node n with label Y , where n is not an end node, and choose a rule ρ , which is applicable to n ;
- if ρ has k denominators then create k successor nodes for n , with successor i carrying an appropriate instance of denominator D_i ;
- if a successor s carries a set Z and Z has already appeared on the branch from the root to s then s is an end node.

A branch is called closed if its end node is carrying \perp , otherwise it is open.

As in the classical case, a formula A is a theorem in modal logic K , iff there is a closed K -tableau for the set $\neg A$.

As an example take the formula $\Box(p \rightarrow q) \rightarrow (\Box p \rightarrow \Box q)$: its negation is certainly unsatisfiable, because the formula is an instance of our previously given K -axiom.

Some remarks are in order:

- The θ -rule, called the thinning rule, is necessary in order to construct the premisses for the application of the K -rule.
- Both can be combined by a new rule

$$(K\theta) \frac{Y; \Box X; \neg \Box P}{X; \neg P}$$

- In order to get tableau calculi for the other mentioned calculi, like T or $K4$ one has to introduce additional rules:

$$(T) \frac{X; \neg \Box P}{X; \Box P; P}$$

which obviously reflects reflexivity of the reachability relation, or

$$(K4) \frac{\Box X; \neg \Box P}{X; \Box X; \neg P}$$

reflecting transitivity.

Translation Method

There are several methods aiming at a translation of propositional modal logics into first order predicate logics.

The idea is, to transform the semantic conditions for the reachability into logical formulae: One rule for the definition of the semantic was:

$$w \models \Box A \quad \text{iff for all } v \in W, (w, v) \in R \text{ or } v \models A$$

This can be compiled into a formula by substituting the modal formula $\Box P$ by the first order formula $\forall y R(x, y) \rightarrow P'(y)$. Hence we can eliminate all modal operators by introducing the first order translations. The result of such a translation is a classical first order formula, which can be processed by the methods we have seen before.

For a modal formula F we define its translation F^* :

- $P^* = P(x)$, if P is a propositional constant
- $(\neg F)^* = \neg(F^*)$
- $(F \wedge G)^* = (F^* \wedge G^*)$

- $(\Box F)^* = (\forall y(R(x, y) \rightarrow F^*(x/y)))$, where y is a new variable not occurring in F^* and $F^*(x/y)$ is the result of replacing all free occurrences of x in F^* by y .

As a result we have

Theorem 1 *F is a valid modal formula in K iff F^* is a valid first order formula.*

Together with the observation that validity in modal logic K (like in many others) is decidable, we hence have a sublogic of first order classical predicate logic which is decidable! Modal logic can be seen as a fragment of 2-variable first-order logic FO^2 .

Temporal Logics

The two modalities \Box and \Diamond cannot be used to distinguish between past and future. For this we need a multi-modal logic with the following \Box -operators

- $[F]A$: A holds always in the future
- $[P]A$: A holds always in the past
- $[A]A$: A holds always

and the corresponding \Diamond -operators:

- $\langle F \rangle A$: A holds somewhere in the future
- $\langle P \rangle A$: A holds somewhere in the past
- $\langle A \rangle A$: A holds somewhere

The semantics is then given as before, by giving constraints for the three reachability relations or by giving appropriate axioms, e.g.

- $[F]A \rightarrow [F][F]A$: Transitivity; an analog axiom holds for the two other \Box -operators.
- $A \rightarrow [F]\langle P \rangle A$: if we go from a time point t in the future t' , we can go back in the past to the time point where A was true.
- $[A]A \leftrightarrow [F]A \wedge [P]A$: connection of past with future.

In addition there are many other aspects of temporal logics. E.g. one can distinguish between left- and rightlinear structures or between dense and discrete time structures.

Layout

[Format document for screen](#)