Mapping Inferences: Constraint Propagation and Diamond Satisfaction

Gennari, R.

Citation for published version (APA):

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: http://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.
Chapter 7

Modal Logics

7.1 Introduction

7.1.1 Motivations

Modal logic [BdRV01] was originally conceived as the logic of necessity and possibility. Indeed, for many years modal logic was viewed as an extension of propositional logic by the addition of the modal operators $\Diamond$ (possibly) and $\Box$ (necessarily). To some extent this picture is still valid and useful. For instance, in the branch of modal logic that is known as epistemic logic, the modal language is used to reason about the knowledge of an agent. Under this reading, $\Box \phi$ stands for “the agent knows that $\phi$”.

But over the past decade or so, the picture has changed, or rather, broadened considerably: modal logic has developed into a powerful discipline on the interface of computer science and mathematics that deals with restricted logical languages for talking about various kinds of relational structures (see [Are00]). Let us elaborate on this. First, relational structures (that is, sets equipped with relations on them) are to be found just about everywhere. For example, in computer science, we use labelled transition systems (LTSs) to model program executions, but an LTS is just a set (the states) together with a collection of binary relations (the transition relations) that model the behaviour of programs [HR00]. Second, modal languages are restricted languages because they talk about relational structures in a special way: modal formulas are evaluated locally, at a particular state, and only the states that are linked to the current state through a relation may be explored. Because of such restrictions, many modal logics end up being fragments of first-order logic. Moreover, they often end up being decidable fragments of first-order logic. The decidability of many important modal systems stems from the step-by-step way that modal formulas are evaluated. More generally, the latter helps to explain why many modal logics enjoy the so-called tree model property: if a formula has a model, it has a model that looks like a tree. The tree
model property has become a key tool in establishing decidability and complexity results for modal languages [Grä99, Var97]. And as we shall see below, a very strong form of the tree model property can be used to devise practical algorithms for modal languages — but this is running ahead of things.

We have to address another issue first: what do constraints and constraint propagation have to do with modal logic? Sitting, as it does, between first-order and propositional logic, two natural strategies suggest themselves for reasoning with modal logic:

- constrain first-order methods so that they become decision procedures for modal logics,
- boost propositional reasoning methods so that they fit modal languages.

In this part of the thesis, we follow both strategies. More specifically, in Chapter 8, we follow the first strategy when we exploit the stepwise way of evaluating modal formulas, and use it to devise a new translation from the modal language into a highly constrained fragment of first-order logic. We provide ample experimental evidence to show that this translation into a fragment of first-order logic yields significant improvements in processing times.

Then, in Chapter 9, we follow the other strategy: boosting propositional methods to make them work for modal languages. Various computational problems have been solved by reformulating them as propositional satisfiability (SAT) problems. Even problems for higher complexity classes than SAT can be efficiently solved by reformulating them as (a sequence of) SAT problems. This is what we do: to solve the modal satisfiability problem we reformulate it as a sequence of SAT problems, and each of those SAT problems is then reformulated as a constraint satisfaction problem — see also Subsection 2.3.2.

### 7.1.2 Outline and Structure

The present chapter introduces the non-expert reader to the basics of modal logics. In Section 7.2, we briefly touch on modal languages, the basic modal logics and their semantics, as needed for the comprehension of the remaining two chapters of this thesis. We only assume from the reader a working knowledge of propositional logic, so to speak. Section 7.3 treats the standard relational translation from modal to first-order languages; as for the latter languages, it is sufficient to know them as extensions of propositional languages through variables and quantifiers, relation and function symbols.
7.2 Background

7.2.1 Modal Languages

Formulas of a unimodal language are built up from proposition letters $p$, using the propositional operators $\neg$, $\lor$, $\land$ and the modal operators $\diamondsuit$ and $\Box$. Formally, let $P$ be a set of proposition letters, that we usually denote as $p$ or $q$, or possibly these with indices. So, consider all sets $B$ of finite strings of modal operators, proposition letters and operators, that enjoy the following properties:

1. $P \subseteq B$;
2. if $\phi$ belongs to $B$, then so does $\neg\phi$;
3. if $\phi$ and $\psi$ belong to $B$, then so do $\psi \land \phi$ and $\psi \lor \phi$;
4. if $\phi$ belongs to $B$, then so does $\Box \phi$.

Then the unimodal language $\mathcal{ML}(P)$ is the smallest (with respect to subset inclusion) of such sets. We denote formulas of $\mathcal{ML}(P)$ by means of Greek alphabet letters, usually $\phi$ and $\psi$.

The dual of the box operator $\Box$, namely the diamond operator $\diamondsuit$, is introduced as an abbreviation: i.e., $\diamondsuit \phi$ stands for $\neg \Box \neg \phi$, for every $\phi$ in $\mathcal{ML}(P)$.

Let Index be some index set. Formulas of the multimodal language denoted by $\mathcal{MML}(\text{Index}, P)$ are built up, as in the unimodal case, from proposition letters, by using $\lor$, $\land$ as above, and modal operators $\Box_i$, for $i \in \text{Index}$.

Since the extension to the multimodal language is often straightforward, we usually state definitions and results for the unimodal language, and only sketch the corresponding ones for the multimodal case.

7.2.2 Modal Models

Models for $\mathcal{ML}(P)$ are structures of the form $\mathcal{M} = \langle M, R, V \rangle$, where:

- $W$ is a non-empty domain,
- $R$ is a binary relation on $W$,
- $V$ is a function from $P$ into the power set $\wp(W)$.

The elements of $W$ are often referred to as states or worlds. They are supposed to represent the states/worlds in which a proposition $p$ holds true.

In fact, truth is defined relative to a state in a model, following the classical interpretation of propositional operators. The important case is given by formulas with modal operators. Formally, consider a model $\mathcal{M}$ and a world $w \in W$. Then we write $\mathcal{M}, w \models \phi$, and read it as $\mathcal{M}$ satisfies $\phi$ at $w$, iff the following is true:
1. in case $\phi$ is $p \in P$, $w \in V(p)$ holds; 
2. in case $\phi$ is $\neg \psi$, $\mathcal{M}, w \models \psi$ holds; 
3. in case $\phi$ is $\psi_1 \lor \psi_2$, $\mathcal{M}, w \models \psi_1$ or $\mathcal{M}, w \models \psi_2$ hold; 
4. in case $\phi$ is $\psi_1 \land \psi_2$, both $\mathcal{M}, w \models \psi_1$ and $\mathcal{M}, w \models \psi_2$ hold; 
5. in case $\phi$ is $\Box \psi$, for any $v \in W$, we have that either $Rwv$ does not hold or $\mathcal{M}, v \models \phi$ does.

Formulas of the form $\Diamond \psi$ are interpreted dually: 

$\mathcal{M}, w \models \Diamond \psi$ iff there exists $v \in M$ such that $\mathcal{M}, v \models \phi$ and $Rwv$.

Models for $\mathcal{M} \langle \text{Index}, P \rangle$ are structures of the form 

$\langle W, \{R_i : i \in \text{Index}\}, V \rangle$,

on which modal operators $\Box_i$, for $i \in \text{Index}$, are interpreted using the associated binary relation $R_i$, marked by the same index $i \in \text{Index}$.

A natural generalisation of the above definition is that of satisfiability of $\phi$ in a model $\mathcal{M}$ for the language of $\phi$: the modal formula $\phi$ is satisfiable in the model $\mathcal{M}$, or the model satisfies $\phi$, if $\phi$ holds true at some world of $\mathcal{M}$. The formula $\phi$ is satisfiable if there exists a model, for the language of $\phi$, that satisfies $\phi$.

The notion of unsatisfiability is then derived in the obvious manner.

### 7.2.3 Basic Modal Logics

At this point, we have seen both modal languages (see Subsection 7.2.1) and structures for interpreting those languages, namely models (see Subsection 7.2.2). The next step consists in defining logics in those languages, and see if and how each is the perfect counterpart of some class of models: i.e., if soundness or completeness holds for the logic with respect to a certain class of models. In the remainder of this subsection, we introduce the non-expert reader to basic modal logics, and state their soundness and completeness with respect to the class of all models; we refer those interested in a complete introduction to modal logics to [BdRV01]. In what follows, we only assume some basic knowledge of logics like propositional logic: i.e., the notion of tautology, axiom and inference rule.

We start with the basic unimodal logic, as in the following definition.

**Definition 7.2.1.** Given a unimodal language $\mathcal{M} := \mathcal{M}(P)$, the basic modal logic $K$ in $\mathcal{M}$ has the following set of axioms, where $\psi$, $\phi$ and $\theta$ range over all the modal formulas in $\mathcal{M}$:

(P1). $\phi \rightarrow (\psi \rightarrow \phi)$;
7.2. Background

(P2). \((\phi \rightarrow \psi) \rightarrow (\neg \psi \rightarrow \neg \phi)\);

(P3). \((\phi \rightarrow (\psi \rightarrow \theta)) \rightarrow ((\psi \rightarrow \phi) \rightarrow (\psi \rightarrow \theta))\);

(K1). \(\Box (\phi \rightarrow \psi) \rightarrow (\Box \phi \rightarrow \Box \psi)\).

The inference rules of \(K\) are as follows:

(MP). \(\phi, \phi \rightarrow \psi \vdash \psi\);

(NEC). \(\phi \vdash \Box \phi\).

An \(\mathcal{ML}\) formula \(\phi\) is a \(K\) theorem if it is either an axiom of the above form, or is obtained by applying one of the rules MP or NEC to \(K\) theorems.

If \(\phi\) is a \(K\) theorem, then we write \(\vdash K \phi\) or simply \(\vdash \phi\).

The definition for \(K(\text{Index})\) is analogous to the above one, the only difference being in the language: all the \(K\) axioms and rules come with indices.

As stated at the opening of the present part, we are interested in automated theorem proving. In this setting, a strategy to prove that a formula is a theorem of a logic appeals to the related semantics: the prover has to derive that the negation of the formula is unsatisfiable. Hence, if the logic is sound and complete with respect to its semantics, this strategy allows us to test whether a formula is or is not a theorem of a logic. As far as basic modal logics are concerned, soundness and completeness hold as follows.

**Theorem 7.2.2.**

i. A unimodal formula is a theorem of \(K\) iff its negation is unsatisfiable.

ii. A multimodal formula is a theorem of \(K(\text{Index})\) iff its negation is unsatisfiable.

For a proof of Theorem 7.2.2, the reader is invited to consult [BdRV01].

7.2.4 Examples

Now that we have introduced some basic formal machinery for modal logic, let us return to the informal discussion in Section 7.1, and provide some examples to complement it with.

Recall that in epistemic logic, \(\Box \phi\) is read as "the agent knows that \(\phi\)" and for that reason one often writes \(K\phi\) in stead of \(\Box \phi\). Given that we are talking about knowledge in epistemic logic (as opposed to, say, belief or rumour), it seems natural to view all instances of \(\Box \phi \rightarrow \phi\) as true: if the agent really knows that \(\phi\), then \(\phi\) must hold. On the other hand (assuming that the agent is not omniscient) we would regard \(\phi \rightarrow K\phi\) as false.
We pointed out, in Section 7.1, that labelled transition systems are an especially important kind of relational structures, and, hence, of modal models. They are typically described using temporal logic, where one has modal operators \([G]\) and \([H]\). The intended interpretation of a formula \([G]\phi\) is "\(\phi\) is always going to be the case", and the intended interpretation of \([H]\phi\) is "\(\phi\) has always been the case". We can express many interesting assertions involving time with this language; for instance, we could use \(\text{request}_i \rightarrow \neg[G]\text{ignored}_i\), to say that, whenever resource \(i\) is requested, it is eventually granted.

Researchers developing formalisms for reasoning about graphs have sometimes come up with notational variants of modal logic. For example, computational linguists use Attribute-Value Matrices (AVMs) for describing feature structures (directed acyclic graphs that encode linguistic information). Here is a fairly typical AVM:

\[
\begin{bmatrix}
\text{AGREEMENT} \\
\text{CASE}
\end{bmatrix}
\begin{bmatrix}
\text{PERSON} & 1st \\
\text{NUMBER} & \text{plural}
\end{bmatrix}
\begin{bmatrix}
dative
\end{bmatrix}
\]

But this is just a two dimensional notation for the following modal formula:

\[
\langle\text{AGREEMENT}\rangle \langle\text{PERSON}\rangle 1st \land \langle\text{NUMBER}\rangle \text{plural} \land \langle\text{CASE}\rangle \text{dative}.
\]

Similarly, researchers in artificial intelligence needing a notation for describing and reasoning about ontologies developed description logic. For example, the concept of "being a free-lance musician" is true of any individual who is a musician and is employed by someone who organizes a birthday party. In description logic we can define the latter concept as follows:

\[
\text{musician} \sqcap \exists\text{employer.}\text{organizer}.
\]

But this is simply the following modal formula lightly disguised:

\[
\text{musician} \land \langle\text{employer}\rangle \text{organizer}.
\]

The links between modal logic on the one hand, and feature and description logic on the other, are far more interesting than these rather simple examples might suggest; see [BdRV01] for details and references on these connections.

### 7.3 The Standard Translation

As we pointed out in the introduction to the present chapter, modal languages can often be mapped into fragments of suitable first-order languages. To make this annotation precise, we need some basic definitions.

The begin with, the vocabulary of the first-order language \(\mathcal{FO}(P)\) has a unary predicate symbol \(P\) for each proposition letter \(p\) in \(P\), and a single binary relation symbol \(R\). Instead of a single binary relation symbol, the vocabulary of the first-order language \(\mathcal{FO}(\text{Index}, P)\) has a binary relation symbols \(R_i\) for each \(i \in \text{Index}\).
7.3. The Standard Translation

**Definition 7.3.1.** [vB83] The standard relational translation $ST(\phi)$ of unimodal formulas into first-order formulas of $\mathcal{FO}(P)$ is defined as below. In what follows, let $x$ and $y$ be distinct individual variables:

\[ ST_x(p) := P(x), \]
\[ ST_x(\neg \phi) := \neg ST_x(\phi), \]
\[ ST_x(\phi \land \psi) := ST_x(\phi) \land ST_x(\psi), \]
\[ ST_x(\phi \lor \psi) := ST_x(\phi) \lor ST_x(\psi), \]
\[ ST_x(\Box \phi) := \exists y (Rxy \land ST_y(\phi)), \]
\[ ST_x(\Diamond \phi) := \forall y (\neg Rxy \lor ST_y(\phi)). \]

The translation $ST$ is defined to be $ST_x$ for a generic individual variable $x$.

The above is easily extended to a translation taking multimodal formulas into $\mathcal{FO}(\text{Index}, P)$, by means of the relation symbol $R_i$ instead of $R$ in the translation of the operators $\Diamond_i$ and $\Box_i$, for $i \in \text{Index}$.

**Note 7.3.2.** In (7.1) and (7.2), $P$ is the unary predicate symbol corresponding to the proposition letter $p$. Observe how (7.3) and (7.4) reflect the truth definitions of the modal operators.

As a consequence of Note 7.3.2, models for the unimodal language $\mathcal{ML}(P)$ and the multimodal language $\mathcal{MMML}(\text{Index}, P)$ can be recast as structures for the corresponding first-order languages $\mathcal{FO}(P)$ and $\mathcal{FO}(\text{Index}, P)$, respectively. To interpret the unary predicate symbols, we look up the values of the corresponding proposition letters in the valuation.

**Example 7.3.3.**

- The unimodal formula $\Box(\neg p \lor \Diamond p)$ translates into the first-order formula $\forall y (\neg Rxy \lor (\neg Py \lor \exists z (Ryz \land Pz)))$.

- The multimodal formula $\Box_i(\neg p \lor \Diamond_{k<p} p)$ translates into the first-order formula $\forall y (\neg R_i xy \lor (\neg Py \lor \exists z (R_ky z \land Pz)))$.

**Theorem 7.3.4 ([vB83]).** A modal formula is satisfiable iff its standard relational translation is.

The above result effectively embeds the modal languages considered here into first-order languages, and paves the way to deciding modal satisfiability by first-order means, as explained in Subsection 8.2.2.
7.4 Conclusions

The standard translation from modal to first-order languages, devised in [vB83], is at the base of correspondence theory: in this setting, the translation is conceived as a first step in the study of the expressivity of modal languages for describing both models and the relational structures models are based on, namely frames. Thus the standard translation is devised to preserve satisfiability, as quoted above, and also satisfiability at every world of certain relational structures, that is validity with respect to classes of frames.

As we prove in Chapter 8, our translation preserves satisfiability, but not validity. Our aim is to make use of automated theorem provers to decide whether a modal formula is a theorem of a certain modal logic or not. For this task, preserving satisfiability is sufficient, due to the soundness and completeness of the logics we are interested in, see Subsection 7.2.3. Given this, the goal of our translation becomes to convey information that boosts automated theorem proving. Therefore, our translation aims at preserving the structure of the original modal formula, as much as possible, and encoding semantic properties of basic modal logics that are computationally relevant, loosely speaking. Chapter 8 explains how this is achieved.