Tableau-based decision procedures for hybrid and description logics

Bolzano, September, 2010

Thomas Bolander
DTU Informatics
Technical University of Denmark

A soft start: Modal logic

Propositional modal logic (in negation normal form, NNF) has the following syntax, where p is a **propositional symbol**:

$$\phi, \psi \quad ::= \quad p \mid \neg p \mid \phi \wedge \psi \mid \phi \vee \psi \mid \Diamond \phi \mid \Box \phi$$

A **Kripke model** (or simply **model**) is a tuple $\mathcal{M} = (W, R, V)$ where W is a set of **worlds**, R is a binary **accessibility relation** on W, and V maps propositional symbols into subsets of W. The underlying graph (W, R) is called a **frame**. The relation $\mathcal{M}, w \models \phi$ is defined inductively, where $w \in W$.

$$\begin{array}{rll} \mathcal{M}, w \models p & \text{iff} & w \in V(p) \\ \mathcal{M}, w \models \neg\phi & \text{iff} & \text{not } \mathcal{M}, w \models \phi \\ \mathcal{M}, w \models \phi \wedge \psi & \text{iff} & \mathcal{M}, w \models \phi \text{ and } \mathcal{M}, w \models \psi \\ \mathcal{M}, w \models \Box\phi & \text{iff} & \text{for all } v \in W, \text{ if } (w, v) \in R \text{ then } \mathcal{M}, v \models \phi \end{array}$$

$\Diamond$ is dual of $\Box$. A formula ϕ is **valid** if $\mathcal{M}, w \models \phi$ for *all* $\mathcal{M}, w$. It is **satisfiable** if $\mathcal{M}, w \models \phi$ for *some* $\mathcal{M}, w$, that is, if $\neg\phi$ is *not* valid.

Applications of modal logic in computer science

Model logic has numerous applications in computer science. This is due to the many possible interpretations of the model operators ($\Diamond$ and $\Box$):

- **Temporal**: $\Box\phi$ means "in the next step, ϕ" or "always in the future, ϕ". Application examples: specification & verification.
- **Epistemic**: $\Box\phi$ means "it is known that ϕ" (by some agent). Application example: multi-agent systems (MAS).
- **Temporal** + **Epistemic**.
- **Ontological**: $\mathit{Student} \leftrightarrow \mathit{Person} \wedge \Diamond_{\mathit{enrolled}} \mathit{Course}$. This gives **description logic**. Application examples: ontological databases, semantic web.

A major advantage over using first-order predicate logic is that modal logics tend to be decidable.

Decision procedures for modal logic

The main decision problem for a modal logic is satisfiability of formulas (or, dually, validity).

Types of decision methods suitable for implementation: **tableau procedures**, **resolution procedures** and **automata-based procedures**.

Here we consider **tableau procedures**.

We focus only on **decidability**, not **complexity**.

A tableau calculus for modal logic

Given: a formula ϕ of modal logic, that we want to check whether is **satisfiable**.

We build a **completion tree** (or **tableau branch**) T, a tree labelled with sets of subformulas of ϕ (this is the description logic style of tableau).

Initialisation: A single node x_0 with label $\mathcal{L}_T(x_0) = \{\phi\}$.

Then we iteratively apply the following **expansion rules** (or **tableau rules**)—with the stipulation that no rule is applied twice to the same premises:

- $\wedge$**-rule:** If $\phi \wedge \psi \in \mathcal{L}_T(x)$ then set $\mathcal{L}_T(x) = \mathcal{L}_T(x) \cup \{\phi, \psi\}$.
- $\vee$**-rule:** If $\phi \vee \psi \in \mathcal{L}_T(x)$ then set $\mathcal{L}_T(x) = \mathcal{L}_T(x) \cup \{\phi\}$ **or** set $\mathcal{L}_T(x) = \mathcal{L}_T(x) \cup \{\psi\}$.
- $\Diamond$**-rule:** If $\Diamond\phi \in \mathcal{L}_T(x)$ then create child node y of x with $\mathcal{L}_T(y) = \{\phi\}$.
- $\Box$**-rule:** If $\Box\phi \in \mathcal{L}(x)$ and y is a child of x then set $\mathcal{L}_T(y) = \mathcal{L}_T(y) \cup \{\phi\}$.

A tableau calculus for modal logic

Recall the rules:

$\wedge$-rule: If $\phi \wedge \psi \in \mathcal{L}_T(x)$ then set $\mathcal{L}_T(x) = \mathcal{L}_T(x) \cup \{\phi, \psi\}$.

$\vee$-rule: If $\phi \vee \psi \in \mathcal{L}_T(x)$ then set $\mathcal{L}_T(x) = \mathcal{L}_T(x) \cup \{\phi\}$ **or** set $\mathcal{L}_T(x) = \mathcal{L}_T(x) \cup \{\psi\}$.

$\Diamond$-rule: If $\Diamond\phi \in \mathcal{L}_T(x)$ then create child node y of x with $\mathcal{L}_T(y) = \{\phi\}$.

$\Box$-rule: If $\Box\phi \in \mathcal{L}_T(x)$ and y is a child of x then set $\mathcal{L}_T(y) = \mathcal{L}_T(y) \cup \{\phi\}$.

A completion tree T is called **saturated** (or **fully expanded**) if no more rule applies. It is called **closed** if $p, \neg p \in \mathcal{L}_T(x)$ for some p, x—otherwise it is called **open** (or **clash free**).

Tableau algorithm: Construct all possible **saturated** completion trees of the input-formula ϕ (note the nondeterminism of the $\vee$-rule). If all **close**, declare ϕ unsatisfiable, otherwise, if at least one is open, declare it satisfiable.

Soundness, completeness and termination of the calculus

Termination: If y is a child of x in a completion tree T then $\max\{|\phi| \mid \phi \in \mathcal{L}_T(y)\} < \max\{|\phi| \mid \phi \in \mathcal{L}_T(x)\}$. (+ König's Lemma).

Soundness (if algorithm says "ϕ is satisfiable" then it is): We can immediately read of a model from any open, saturated completion tree.

Completeness (if ϕ is satisfiable, algorithm will say so): The non-deterministic choices can be made so that every rule preserves the property that the completion tree can be extended into a model of ϕ.

Extending modal logic

Many applications require more than the basic modal logic. There are (at least) two ways to extend it (make it more expressive):

- **Extending the language**: nominals, multiple modalities, modalities of arbitrary arity, inverse modalities, graded modalities (number restrictions), transitive closure operator, fixed-point operators, etc.
- **Imposing frame conditions (on one or all modalities)**: global modality (everywhere), difference modality (everywhere else), transitive modalities, reflexive modalities, irreflexive modalities, asymmetric modalities, etc.

Sound and complete tableau calculi for most of these extensions are easy to formulate. In particular for the **frame conditions**, as it is proven in [Blackburn, 2000] that the tableau calculus for hybrid logic extended with any class of **frame defining axioms** is automatically sound and complete with respect to the class of frames defined by those axioms. This covers e.g. all frame conditions mentioned above.

However, in most cases termination is lost...

Modal logic with the global modality

One of the simplest extensions that produce **non-termination** is modal logic extended with the global modality A with semantics defined by:

$$\mathcal{M}, w \models A\phi \quad \text{iff} \quad \text{for all } v \in W, \ \mathcal{M}, v \models \phi$$

This logic is an extension of the description logic $\mathcal{ALC}$ (in disguise).

The global modality can be handled by adding the following rule to our previous calculus:

> **A-rule:** If $A\phi \in \mathcal{L}_T(x)$ and y is any node in the completion tree then set $\mathcal{L}_T(y) = \mathcal{L}_T(y) \cup \{\phi\}$.

Termination is lost. How can it be regained?

Blocking

Termination is usually regained by imposing a **blocking condition** (also called **loop-check condition**) to avoid repeating worlds. The following works for modal logic with the global modality.

A node x in a completion tree T is called **directly blocked** if it has an ancestor y with $\mathcal{L}_T(y) = \mathcal{L}_T(x)$. A node is called **blocked** if it is directly blocked or if it has an ancestor that is directly blocked.

When constructing completion trees we now impose the following condition:

Blocking condition: No expansion rule is allowed to be applied to a blocked node.

This type of blocking is called **equality blocking** and was first introduced in [Halpern & Moses, 1992] (decision procedures for epistemic logics).

Soundness, completeness and termination with blocking

Termination: Since all labels are sets of subformulas of the input-formula, there can only be finitely many distinct labels, and thus only finitely many unblocked nodes on any branch.

Soundness (if algorithm says "ϕ is satisfiable" then it is): We can build a model from an open, saturated branch by **redirecting** edges leading to directly blocked nodes to their corresponding blocking nodes.

Completeness (if algorithm says "ϕ is **not** satisfiable" then it isn't): Reduces to the unblocked case. Blocking just makes it harder for a completion tree to close, so if the algorithm answers "no" with blocking in place, it will also answer "no" without it.

Extensions of the result

The equality blocking method doesn't only work for the global modality. It works for all of the following extensions of modal logic and their combinations: **global modality**, **transitive modalities**, **nominals and satisfaction operators** (that is, **hybrid logic**), **multiple modalities**, **inverse modalities**, **modalities of arbitrary arity** [Bolander & Blackburn, 2007].

Further extensions

In a number of recent paper [2008–2010], Kaminski and Smolka have shown how to furthermore extend the terminating tableau calculus with: **graded modalities**; **difference modality**; **role hierarchies**; **reflexive modalities**. This gives an extension of the description logic $\mathcal{SHOQ}$.

In description logic, the most expressive logic for which a terminating tableau calculus has been given is $\mathcal{SROIQ}$ [Horrocks, Kutz & Sattler, 2006]. It includes: **complex role inclusions**; **reflexive, antisymmetric, and irreflexive modalities**; **disjoint modalities**; **global modality**; **inverse modalities**; **graded modalities**; and even more.

$\mathcal{SROIQ}$ is the logical bases of the Web Ontology Language ver. 1.1 (OWL 1.1).

Note, however: Most of the results on this slide need more than just equality blocking.

A new approach to soundness and termination

The non-trivial part in all the above results is to prove **soundness** (in the case of satisfiability checking), that is, how to construct a model from an open, saturated completion tree.

It is particularly tricky when we impose frame conditions: consider e.g. the case of **I4** (model logic on **transitive**, **irreflexive** frames). Redirecting edges from blocked to blocking nodes will destroy irreflexivity!

The case of **I4** can be solved: redirect as usual, then unravel, then take transitive closure. But it can get even worse with more complicated frame conditions, and for many of these, no terminating tableau calculi are yet known.

I therefore propose a different approach to termination and soundness of tableau calculi, which I dub the **reduction method**. It is based on an idea introduced in [Fitting, 1983].

Blocking techniques from a general perspective

Assume the following are given:

- `C`: a sound and complete non-terminating tableau calculus wrt. some class of models **M**.
- `C'`: the result of adding some blocking mechanism to `C` that ensures termination (`C'` is the **restricted calculus**).

How do we prove that `C'` is still sound and complete wrt. **M**?

Completeness is trivial (if a completion tree closes with blocking in place, it will also close without it).

Soundness is the (potentially) hard part. We need to prove that the following implication holds for all formulas ϕ:

there exists a saturated, open completion tree in `C'`
with input-formula ϕ

$\Downarrow$

ϕ is satisfiable in **M**.

Two approaches to soundness and termination

As mentioned, we need to prove:

there exists a saturated, open completion tree in C'
with input-formula ϕ

$\Downarrow$

ϕ is satisfiable in **M**.

Two different possible approaches:

- **Direct method**. Show how a model of ϕ can be built directly from the open, saturated branch in the calculus C'. This requires a completely new soundness proof for C', independent of the soundness proof for C.
- **Reduction method**. Prove that the open, saturated branch in the restricted calculus C' can be extended into an open, saturated branch in the unrestricted calculus C. Soundness of C' is thus reduced to soundness of C.

In all results mentioned above, the direct method has been used. We will now explore the reduction method...

Blocking revisited

We define blocking of tableaus in the following generalised way.

Let $\mathtt{C}$ be a tableau calculus. A **blocking operator** for $\mathtt{C}$ is a mapping $\sim$ that to each completion tree $\mathcal{T}$ of $\mathtt{C}$ assigns an equivalence relation $\sim_{\mathcal{T}}$ on the nodes of $\mathcal{T}$.

Let $\mathtt{C}$ be a tableau calculus, and $\sim$ a blocking operator for it. A node x in a completion tree $\mathcal{T}$ of $\mathtt{C}$ is **directly blocked** wrt. $\sim$ if it has an ancestor y with $y \sim_{\mathcal{T}} x$. A node is **blocked** if it is directly blocked or if it has an ancestor that is directly blocked.

Any **blocking operator** on $\mathtt{C}$ now induces a **blocking condition** on completion trees: no expansion rule is allowed to be applied to a blocked node.

Example. If we define $\sim$ by $x \sim_{\mathcal{T}} y \Leftrightarrow \mathcal{L}_{\mathcal{T}}(x) = \mathcal{L}_{\mathcal{T}}(y)$, then the induced blocking condition is simply equality blocking as considered above.

Subtrees and acceptable blockings

Let T be a completion tree. A **possible subtree** of x in T is any subtree of x in a completion tree extending $T' = T - \{y \mid y \text{ is a descendant of } x \text{ in } T\}$.

A blocking operator $\sim$ for a tableau calculus $\mathtt{C}$ is said to be **acceptable** if the following holds:

- For all completion trees T of $\mathtt{C}$, and all nodes x, y in T, if $x \sim_T y$ then x and y have the same set of possible completion subtrees in T up to isomorphism (renaming of nodes).
- For all completion trees T, the number of equivalence classes in $\sim_T$ is bounded by some function in the root formula of T.

If $\mathtt{C}$ is a tableau calculus and $\sim$ is a blocking operator for it, we use $\mathtt{C}_\sim$ to denote the calculus obtained by adding the blocking condition induced by $\sim$ to the calculus $\mathtt{C}$.

Theorem. Let $\mathtt{C}$ be a tableau calculus, and let $\sim$ be an acceptable blocking operator for it. Then $\mathtt{C}_\sim$ is terminating as well as sound and complete wrt. the same class of models as $\mathtt{C}$.

Application examples of the reduction method

Consider the following blocking operator:

$$x \sim_T y \Leftrightarrow \mathcal{L}_T(x) = \mathcal{L}_T(y) \wedge$$
$$\cup\{\mathcal{L}_T(x') \mid x' \text{ is ancestor of } x \text{ in } T\} = \cup\{\mathcal{L}_T(y') \mid y' \text{ is ancestor of } y \text{ in } T\}.$$

This operator is **acceptable** e.g. for tableau calculi for irreflexivity + transitivity (**I4**).

More generally, define n-**to-**m **transitivity** to be defined by the following formula: $\Diamond^n p \rightarrow \Diamond^m p$. Then the following operator is **acceptable** for tableau calculi for irreflexivity + n-to-1 transitivity:

$$x \sim_T y \Leftrightarrow \mathcal{L}_T(x) = \mathcal{L}_T(y) \ \wedge \bigwedge_{0<i<n}$$
$$\cup\{\mathcal{L}_T(x') \mid x' \text{ is ancestor of } x \text{ in } T \text{ and } (\text{dist}_T(x', x) - 1) \bmod (n-1) = i \ \} =$$
$$\cup\{\mathcal{L}_T(y') \mid y' \text{ is ancestor of } y \text{ in } T \text{ and } (\text{dist}_T(y', y) - 1) \bmod (n-1) = i \ \}.$$

A small modification of the blocking operator will make it work also for n-to-m transitivity for any $n > m$. This solves an open problem in temporal logic concerning the decidability of modal logic extended with **weak transitivity**: $\Diamond^3 p \rightarrow \Diamond^2 p$.

Further perspectives

Recall the "automatic sound and completeness" result of Blackburn concerning tableau calculi for axiom definable frames. Combining this with the reduction method has the perspective of providing general results on terminating tableau calculi for a wide range of frame classes.

Conclusion and future work

- We have presented blocking techniques in tableau calculi for modal/hybrid/description logics.
- For the richer logics, the soundness proof usually gets tricky (how to build a model from an open, saturated branch).
- To (partly) solve this problem, we have proposed a new approach: the reduction method.
- The reduction method is particularly useful for logics/modalities with frame conditions, and has promising perspectives in hybrid logic in connection with the result of Blackburn.