Logic and **L**ogical **P**hilosophy
Volume 8 (2000), 99–114

Luis Fariñas del Cerro
Olivier Gasquet

MINIMAL STRUCTURES FOR MODAL TABLEAUX: Some Examples

Abstract. In this paper we present some examples of decision procedures based on tableau calculus for some mono- and multimodal logics having a semantics involving properties that are not easily representable in tree-like structures (like e.g. density, confluence and persistence). We show how to handle them in our framework by generalizing usual tableaux (which are trees) to richer structures: rooted directed acyclic graphs.

1. Introduction and background

In [5], a general completeness proof based in tableaux has been presented for a large family of modal logics. In the line of this previous work, we presented in [6] strategies for some of these logics, and obtain thus decision procedures for logics based on transitivity plus density or confluence as characteristic axioms. The main argument of these results is, as for the usual one concerning S4, that the tableau rules have the subformula property and that the number of subformulas of a given formula is finite.

These previous works lead us to split the usual modal tableau rules into two distinct sets beside the usual classical and diamond rules:

1. propagation rules
2. structural rules.

The former are formulated as "if in some node of such pattern there is such formula, then propagate such formula (the same or another one)", while the latter are "if there is such pattern then add some new node(s) and edge(s)". They respectively correspond to two different families of axioms (relational properties):

- Propagation rules correspond to axioms T and 4 (properties of reflexivity and transitivity);
- Structural rules correspond to axioms D, De, C and Per (respectively properties of seriality, density, confluence and persistence).

In the present paper and in order to give a panorama of our approach, we summarize results for the monomodal case. Then we extend this approach to a multimodal logic with a persistence axiom.

We assume that the reader is familiar with modal logic, Kripke semantics and tableau methods for modal logics as presented e.g. in [7].

1.1. Modal logics and relational properties

A modal logic can be specified syntactically or semantically. We recall what the links between these presentations are. Monomodal language have $\Box$ and $\Diamond$ as additionnal connectives, multimodal ones have $\Box_{\mathtt{K}}$, $\Box_{\mathtt{T}}$, ... and $\Diamond_{\mathtt{K}}$, $\Diamond_{\mathtt{T}}$, ... as additionnal connectives. As usual, $\Diamond$ abbreviates $\neg\Box\neg$ and as well for $\Diamond_{\mathtt{K}}$ and $\Diamond_{\mathtt{T}}$.

The monomodal logics we consider are all obtained by extending the basic modal logic K by one or several of the well-known axioms T, 4, D, De (axiom of density: $\Diamond p \rightarrow \Diamond\Diamond p$) and C (axiom of confluence: $\Diamond\Box p \rightarrow \Box\Diamond p$). Thus KD4.C denotes the modal logic obtained by adding the axioms D, C and 4 to the basic system K.

Among multimodal logic, we will only investigate the system $\mathrm{K}(\mathtt{K},\mathtt{T})+Per$, i.e. the logic obtained with axioms K for $\Box_{\mathtt{K}}$ and $\Box_{\mathtt{T}}$, and the *interaction* axiom $\Box_{\mathtt{K}}\Box_{\mathtt{T}}p \rightarrow \Box_{\mathtt{T}}\Box_{\mathtt{K}}p$ of persistence. The choice of using $\mathtt{K}$ and $\mathtt{T}$ as indexes comes from the epistemic-temporal interpretation of the axiom of Persistence: If one knows that tomorrow A will holds($\Box_{\mathtt{K}}\Box_{\mathtt{T}}A$), then tomorrow one will know that A holds ($\Box_{\mathtt{T}}\Box_{\mathtt{K}}A$), i.e. the knowledge about future persists.

With each of these axioms can be associated a relational property of the accessibility relation of the Kripke models:

Axiom	Property	Notation
$\mathrm{T} = \Box p \rightarrow p$	reflexivity	*Ref*
$4 = \Box p \rightarrow \Box\Box p$	transitivity	*Tr*

Group 1: Properties handled by propagation rules

Axiom	Property	Notation
$\mathrm{D} = \Box p \rightarrow \Diamond p$	seriality	*Ser*
$\mathrm{De} = \Diamond p \rightarrow \Diamond\Diamond p$	density	*Dens*
$\mathrm{C} = \Diamond\Box p \rightarrow \Box\Diamond p$	confluence	*Conf*
$\mathrm{Per} = \Box_{\mathtt{K}}\Box_{\mathtt{T}} p \rightarrow \Box_{\mathtt{T}}\Box_{\mathtt{K}} p$	persistence	*Per*

Group 2: Properties handled by structural rules

As a consequence of Sahlqvist's theorem [13], a system based on K plus any combination of these axioms is characterized by the Kripke models whose accessibility relation satisfies the corresponding properties. Thus, KD4 is characterized by Kripke models where the accessibility relation is both serial and transitive. As well K(K,T).*Per* is characterized by the class of Kripke models $(W, R_{\mathtt{K}}, R_{\mathtt{T}})$ with $R_{\mathtt{T}} \circ R_{\mathtt{K}} \subseteq R_{\mathtt{K}} \circ R_{\mathtt{T}}$ where $\circ$ denotes the composition of relations: $(x, y) \in R \circ S$ iff $\exists z \colon (x, z) \in R$ and $(z, y) \in S$.

1.2. Preliminaries and notations

The tableau method we are going to present is based on RDAG (rooted directed acyclic graphs) having additional properties; let ρ be the set of these additional properties, we define:

DEFINITION 1. A labelled ρ-RDAG is a triple $(\mathcal{N}, \Sigma, \Phi)$ where:

- $(\mathcal{N}, \Sigma)$, where $\mathcal{N}$ is a set of nodes and Σ is a set of edges, is a directed acyclic graph (DAG for short), i.e. a directed graph that contains no cycle, with a distinguished node called the *root* that can access every other node in the transitive closure of Σ,
- $(\mathcal{N}, \Sigma)$ satisfies all the properties of ρ,
- Φ is a function that associates additional information with each of the nodes: if x is a node, $\Phi(x)$ is a set of formulas.
- In the case of multimodal logics, Σ is partitionned into $\Sigma_{\mathtt{K}}$ and $\Sigma_{\mathtt{T}}$.

By abuse of notation and for the sake of notational economy, we will make no distinction between the nodes and their associated sets of formulas; thus

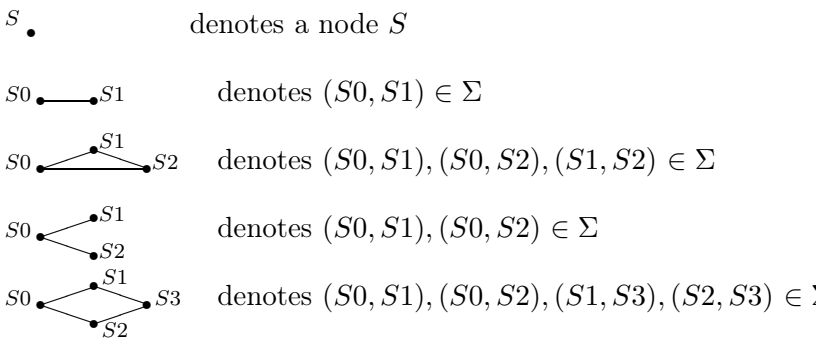
we will write $A \in x$ instead of $A \in \Phi(x)$. Also by abuse of notation, we will sometimes denote a ρ-RDAG $(\mathcal{N}, \Sigma)$ by the binary relation Σ. Thus we will make no distinction between labelled structures and structures.

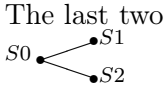
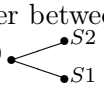
This notion also extend to graphs:

DEFINITION 2. An rgraph is a graph that has a root, and a ρ-rgraph is a rgraph that satisfies all properties of ρ.

As usual, $\Sigma(x)$ will denote the set of nodes accessible from x by Σ: $\Sigma(x) = \{y \in \mathcal{N} \colon (x, y) \in \Sigma\}$. Also, Σ^n will denote the pairs (x, y) such that there is a path of length n between x and y. The diagonal relation: $\{(x, x) \colon x \in \mathcal{N}\}$ will be denoted by I and also by Σ^0. Also, given a binary relation Σ, we denote by Σ^+ its transitive closure.

For the sake of clarity, we will use diagrammatic representation for RDAG. The figure below gives the intended meaning of those diagrammatic representations in which the edges are implicitely left-to-right directed[1]:



The last two diagrams do not involve any order between $S1$ and $S2$, e.g.  can be represented as well by 

Also, the same graphical conventions hold for multimodal logics where edges are labelled with either `K` or `T`.

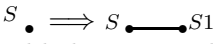
1.3. Rewriting RDAG

Usually, tableaux calculi consist in rewriting a structure by using some appropriate set of rewriting rules (or simply rules). But before presenting our

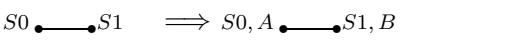
[1] Note that RDAG's are of course antisymmetrical.

rules, we propose some visual conventions. The rules we will use will all be of one of the following forms (the intended meaning is given below the rule); as usual, S, A denotes $S \cup \{A\}$:

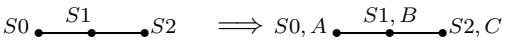
rewrite the node S into the node $S \cup \{A\}$, i.e. add the formula A to the node S,



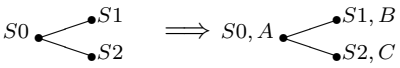
add the new node $S1$ to the successors of the node S,



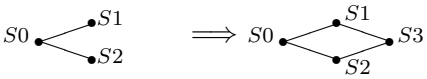
add the formula A to the node $S0$ and B to $S1$,



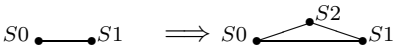
add the formula A to $S0$, B to $S1$ and C to $S2$,



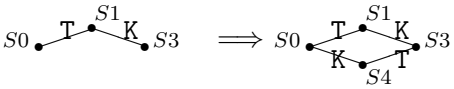
add the formula A to $S0$, B to $S1$ and C to $S2$.



add the new node $S3$ as a common successor of the node $S1$ and $S2$,

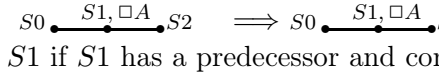


add the new node $S2$ between $S0$ and $S1$,



with its obvious reading: if there is a T-K path between $S0$ and $S3$ then add the new K-T path between and create the intermediary new node $S4$.

This presentation allows to implicitly take into account constraints on the applicability of rules: e.g. a rule such as



reads "add $\Box A$ to any successor of $S1$ if $S1$ has a predecessor and contains $\Box A$".

1.4. Rules and naive tableaux

Here are the rules we need:

- Classical and $\Diamond$ rules:
  - Rule $\bot$: $A, \underset{\bullet}{\neg A}, S \Longrightarrow A, \underset{\bullet}{\neg A}, \bot, S$
  - Rule $\neg$: $\underset{\bullet}{\neg\neg A}, S \Longrightarrow \underset{\bullet}{\neg\neg A}, A, S$
  - Rule $\wedge$: $\underset{\bullet}{A \wedge B}, S \Longrightarrow \underset{\bullet}{A \wedge B}, A, B, S$
  - Rule $\vee$: $\underset{\bullet}{\neg(A \wedge B)}, S \Longrightarrow \underset{\bullet}{\neg(A \wedge B)}, C, S$
    where C is one among $\neg A$ and $\neg B$

    Usually, treatment of disjunction is presented by so-called *tableau branching*: both possibilities are computed in parallel (width-first computation), while in our presentation, we use a depth-first computation. These presentations are in fact equivalent, in practice only depth-first implementations are realized.
  - Rule $\Diamond$: $\underset{\bullet}{\Diamond A}, S \Longrightarrow \Diamond A, S \bullet\!\!-\!\!\bullet A$

    And its multimodal versions:
  - Rule $\Diamond_{\mathtt{K}}$: $\underset{\bullet}{\Diamond_{\mathtt{K}} A}, S \Longrightarrow \Diamond_{\mathtt{K}} A, S \overset{\mathtt{K}}{\bullet\!\!-\!\!\bullet} A$
  - Rule $\Diamond_{\mathtt{T}}$: $\underset{\bullet}{\Diamond_{\mathtt{T}} A}, S \Longrightarrow \Diamond_{\mathtt{T}} A, S \overset{\mathtt{T}}{\bullet\!\!-\!\!\bullet} A$

- Propagation rules:
  - Rule K: $\Box A, S \bullet\!\!-\!\!\bullet S1 \Longrightarrow \Box A, S \bullet\!\!-\!\!\bullet A, S1$
  - Rule 4: $S, \Box A \bullet\!\!-\!\!\bullet S1 \Longrightarrow S, \Box A \bullet\!\!-\!\!\bullet S1, \Box A$

    And multimodal versions of Rule K:
  - Rule $\Box_{\mathtt{K}}$: $\Box_{\mathtt{K}} A, S \overset{\mathtt{K}}{\bullet\!\!-\!\!\bullet} S1 \Longrightarrow \Box_{\mathtt{K}} A, S \overset{\mathtt{K}}{\bullet\!\!-\!\!\bullet} A, S1$
  - Rule $\Box_{\mathtt{T}}$: $\Box_{\mathtt{T}} A, S \overset{\mathtt{T}}{\bullet\!\!-\!\!\bullet} S1 \Longrightarrow \Box_{\mathtt{T}} A, S \overset{\mathtt{T}}{\bullet\!\!-\!\!\bullet} A, S1$

- Structural rules:
  - Rule D: $\underset{\bullet}{S} \Longrightarrow S \bullet\!\!-\!\!\bullet \emptyset$
  - Rule C: $S0 \bullet\!\!-\!\!\bullet S1, S2 \Longrightarrow S0 \bullet\!\!-\!\!\bullet S1 \bullet\!\!-\!\!\bullet \emptyset \bullet\!\!-\!\!\bullet S2$
  - Rule $C*$: $S \bullet\!\!-\!\!\bullet S1 \Longrightarrow S \overset{S1}{\bullet\!\!-\!\!\bullet\!\!-\!\!\bullet} \emptyset$
  - Rule De: $S0 \bullet\!\!-\!\!\bullet S1 \Longrightarrow S0 \overset{\emptyset}{\bullet} S1$
  - Rule Per: $S0 \overset{\mathtt{T}}{\bullet} \overset{S1}{\bullet} \overset{\mathtt{K}}{\bullet} S2 \Longrightarrow S0 \overset{\mathtt{T}}{\bullet} \overset{S1}{\bullet} \overset{\mathtt{K}}{\bullet} S2$, $\bullet \overset{\mathtt{K}}{\bullet} \overset{\emptyset}{\bullet} \overset{\mathtt{T}}{\bullet}$

Of course, it may be discussed why not to consider rule $\Diamond$ as a structural rule. This decision is quite arbitrary, we consider that structural rules are those describing an existential property of the accessibility relation, while rule $\Diamond$ correspond to the expression *in the object language* of the existence object.

In order to define a tableau calculus for a logical system, we must associate a set of rules with it. All the tableaux calculi we are going to define contain:

- Classical rules
- Rule(s) $\Diamond$ (or $\Diamond_{\mathtt{K}}$ and $\Diamond_{\mathtt{T}}$ in the multimodal case)
- Rule K (or $\Box_{\mathtt{K}}$ and $\Box_{\mathtt{T}}$)
  (All these rules being common to all tableaux calculi, we will henceforth omit them)
- Some or none structural and propagation rules.

A tableau calculus for a system denoted by a set $(\rho_1 \cup \rho_2)$ of properties is obtained by taking (in addition to classical, $\Diamond$ and K rules) the rules corresponding to properties of $(\rho_1 \cup \rho_2)$; this correspondance is given in the figure below, where properties of group 1 are handled by propagation rules while those of group 2 are handled by structural rules.

	Properties	Rules	
Group 1	*Ref* *Tr*	T 4	Propagation Rules
Group 2	*Ser* *Dens* *Conf* *Per*	D De C C* Per	Structural Rules

Tableau rules

DEFINITION 3. We define what we call *naive tableaux*, i.e. tableaux computed with no strategy. Thus naive tableaux may lead to non terminating computation. We will see in the next section that terminating strategies do exist in the case we are investigating.

A naive $(\rho_1 \cup \rho_2)$-tableau for a formula A is the limit of a sequence Υ_0, $\ldots$, Υ_i, Υ_{i+1}, $\ldots$ where:

- Υ_0 is an RDAG consisting of only one node whose associated set of formulas is $\{A\}$,
- Υ_{i+1} is obtained from Υ_i by applying either a classical rule, or the $\Diamond$ rule, or the rule K, or a rule of $(\rho_1 \cup \rho_2)$
- and in which every applicable rule has been applied.

DEFINITION 4. A tableau is closed if some node in it contains $\bot$; it is open otherwise. A formula is $\rho_1 \cup \rho_2$-closed iff all its $(\rho_1 \cup \rho_2)$-tableaux are closed.[2]

The following result has been stated and proved in [5]:

THEOREM 1. *The tableaux calculi thus defined are sound and complete.*[3]

As they are defined, naive tableaux may run infinitely. As an example, a $\{Ser, Tr\}$-tableau for $\Box\Diamond p$ runs infinitely because of rules (4) and $(\Diamond)$ that apply infinitely. In this case, of course, the nodes generated all contain the same formulas and thus the tableau loops. But this loop detection concerns the strategy.

2. The kernel approach

The basic idea once completeness has been obtained to get decision procedure is to find for each logic a so-called family of "kernels": a kernel is simply a finite structure able to simulate the infinite tableaux obtained with a naive algorithm that would just implement tableaux as presented above. In this sense, it is well-known that kernels for S4 (i.e. KT4) are finite trees. We will also recall (from [6]) that kernels for KD4.C are finite sequences of finite lattices, and we will show a similar result for KD4.De.

With respect to [6], nothing new will be added in the monomodal case, but this will help understanding the approach.

In what follows we will use some conventions and notations that are presented here:

1. Given a tableau $Y = (N, \Sigma, \Phi)$ and structural rule (S) (among (D), $(\Diamond)$, (De), (C),...) we will denote by N^{S} the set of nodes created by applying this rule at some iteration, and dually, Σ^{S} the set of edges created by applying this rule.

[2] Due to the rule $\vee$, a formula may have several distinct tableaux.

[3] In fact the whole theorem — that is proved in [5] — takes into account many other axioms and properties.

2. In order to lighten the notations, we decided to use bold-face symbols for those which concern kernels, while naive tableau will be denoted by non-bold symbols (thus Y denotes a naive tableau, while $\boldsymbol{Y}$ denotes a kernel).

2.1. The monomodal case

In this section, we consider two subsections, one corresponding to the well-known monomodal cases KD4, KD4.C and KD4.D (Since the results of this subsection appeared in previous publications ([5], [6]), the corresponding proofs are not given), the other corresponding to a multimodal logic with two modal operators and an interaction axiom that allows the permutation of the modal operators.

2.1.1. Kernels for KD4

The set of rules that will be used is: all classical rules, and rules (D), ($\Diamond$), (K) and (4). Kernels for KD4 are simply naive tableaux provided with a strategy that allows to conclude that some tableau is open after only finitely many steps; this proves that kernels for KD4 are finite trees. Let A be the starting formula, we get the following non-deterministic (w.r.t. the order of application of the rules) algorithm that computes a sequence $(\boldsymbol{Y})_i$ of RDAG:

Starting from $\boldsymbol{Y}_0 = (\boldsymbol{N}_0, \boldsymbol{\Sigma}_0, \boldsymbol{\Phi}_0)$ where $\boldsymbol{N}_0$ only consists of one node $\boldsymbol{r}$ (the root), $\boldsymbol{\Sigma}_0$ is empty and $\boldsymbol{\Phi}_0$ associate the formula A with $\boldsymbol{r}$. Then compute $\boldsymbol{Y}_{i+1} = (\boldsymbol{N}_{i+1}, \boldsymbol{\Sigma}_{i+1}, \boldsymbol{\Phi}_{i+1})$ from $\boldsymbol{Y}_i = (\boldsymbol{N}_i, \boldsymbol{\Sigma}_i, \boldsymbol{\Phi}_i)$ by applying successively each of the following steps:

1. Loop step: consider all nodes $x \in \boldsymbol{N}_i$ such that $\exists y \in \boldsymbol{N}_i$ and y is an ancestor of x (i.e. $(y, x) \in \boldsymbol{\Sigma}_i^+$) and $\boldsymbol{\Phi}_i(x) \subseteq \boldsymbol{\Phi}_i(y)$ and set them as *marked*; (nodes that are "contained" in one already present in the tree needs not to be further developed)
2. Classical step: apply classical rules (($\bot$), ($\neg$), ($\vee$), ($\wedge$)) on all nodes as much as possible (also known as classical saturation);
3. Structural step: apply rules (D) and ($\Diamond$) on each *non marked* node where they have not been applied yet[4];
4. Propagation step: apply rules (K) and (4) as much as possible.

[4] As usual, rule (D) must be applied only once on each node while rule ($\Diamond$) must be applied once for each formula $\Diamond B$ of each node.

The above algorithm must be applied until for some i, either $\boldsymbol{Y}_i$ is closed or $\boldsymbol{Y}_{i+1} = \boldsymbol{Y}_i$ (i.e. there are marked nodes on each branch)[5].

THEOREM 2. *The strategy given above is sound, complete and terminating for* KD4.

2.1.2. Kernels for KD4.C

In this subsection, we give a terminating non-deterministic tableau calculus for the system KD4.C, that can be straightforwardly modified in order to apply to K4.C and KT4.C.

For this we define a strategy to be applied on naive tableaux as defined previously; the set of rules that is needed is: all classical rules, and rules (D), ($\Diamond$), (K), (4) and (C) (rule (C*) is superfluous since it is subsumed by the rule (D)).

This strategy mainly consists of the following:

1. Compute a KD4-kernel (using only classical rules, and rules (D), ($\Diamond$), (K) and (4)). This provides a finite tree (either closed or looping on each branch).
2. Create a successor common to each loop node (we will call this node the anti-root) and propagate formulas (rules (K) and (4)) into it. Then go back to step 1, with the anti-root as the starting node (and as such, as the new root).

Stop the computation when: the tableau closes at any step, or if looping anti-roots are successively generated.

The algorithm runs as follows: Starting from $\boldsymbol{Y}_0 = (\boldsymbol{N}_0, \boldsymbol{\Sigma}_0, \boldsymbol{\Phi}_0)$ where $\boldsymbol{N}_0$ only consists of only one node $\boldsymbol{r}^0$ (the root), $\boldsymbol{\Sigma}_0$ is empty and $\boldsymbol{\Phi}_0$ associate the formula A with $\boldsymbol{r}^0$.

1. Compute $\boldsymbol{Y}_{i+1} = (\boldsymbol{N}_{i+1}, \boldsymbol{\Sigma}_{i+1}, \boldsymbol{\Phi}_{i+1})$ from $\boldsymbol{Y}_i = (\boldsymbol{N}_i, \boldsymbol{\Sigma}_i, \boldsymbol{\Phi}_i)$ by applying the strategy for KD4 only (i.e. by using only classical rules, and rules (D), ($\Diamond$), (K) and (4)). In $\boldsymbol{Y}_{i+1}$, each branch loops (i.e. each of its leaves is marked), or else, it is closed.

   Let us denote by $Loop_{i+1}$ the set of nodes of $\boldsymbol{N}_{i+1}$ that are marked.

[5] Since there are no backwards rules, rules (K) and (4) must be applied only in order to propagate formulas in new nodes introduced at step 3.

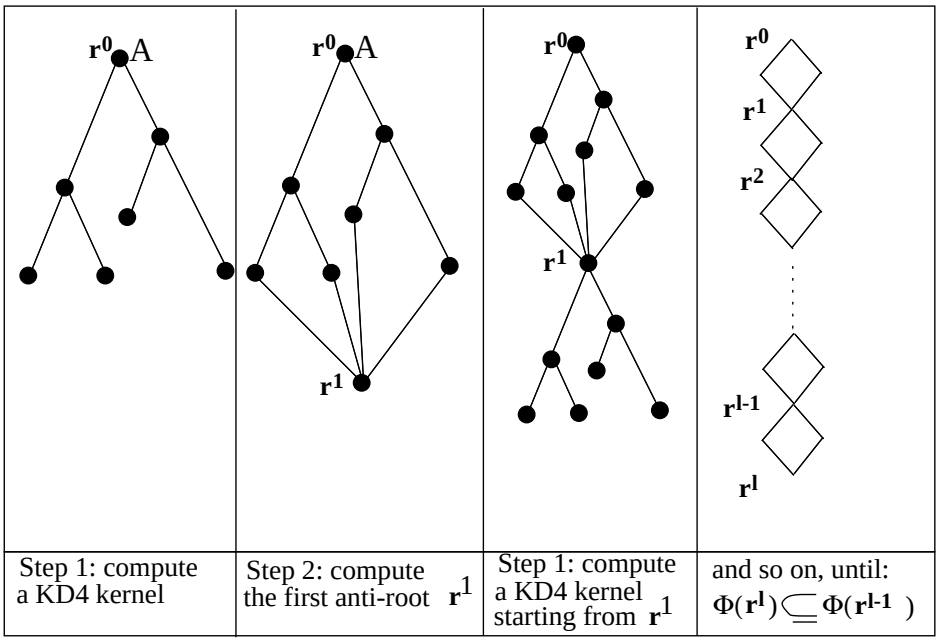


Figure 1. The strategy for KD4.C

2. Compute $\boldsymbol{Y}_{i+2} = (\boldsymbol{N}_{i+2}, \boldsymbol{\Sigma}_{i+2}, \boldsymbol{\Phi}_{i+2})$ from $\boldsymbol{Y}_{i+1} = (\boldsymbol{N}_{i+1}, \boldsymbol{\Sigma}_{i+1}, \boldsymbol{\Phi}_{i+1})$ by:
   - $\boldsymbol{N}_{i+2} = \boldsymbol{N}_{i+1} \cup \{\boldsymbol{r}^{i+2}\}$, where $\boldsymbol{r}^{i+2}$ is a new node,
   - $\boldsymbol{\Sigma}_{i+2} = \boldsymbol{\Sigma}_{i+1} \cup \{(x, \boldsymbol{r}^{i+2})\colon x \in \mathit{Loop}_{i+1}\}$
   - $\boldsymbol{\Phi}_{i+2}(\boldsymbol{r}^{i+2}) = \bigcup_{x \in \mathit{Loop}_{i+1}} (\boldsymbol{\Phi}_{i+1}(x))^{\Box}$; where as usual, $S^{\Box}$ denotes the set $\{A, \Box A\colon \Box A \in S\}$

The above algorithm must be applied until for some i and some l, either $\boldsymbol{Y}_i$ is closed or $\boldsymbol{\Phi}_i(\boldsymbol{r}^l) \subseteq \boldsymbol{\Phi}_i(\boldsymbol{r}^{l-1})$, where $\boldsymbol{r}^l$ denotes the last anti-root generated. This strategy is graphically represented in figure 1.

THEOREM 3. *The strategy given above is sound, complete and terminating for* KD4.C.

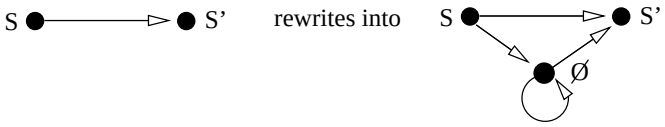
2.1.3. Kernels for KD4.De

In this subsection, we give a terminating tableau calculus for the system KD4.De, that can be straightforwardly modified in order to apply to K4.De (Note that the system KT4.De is the same as KT4 i.e. S4).

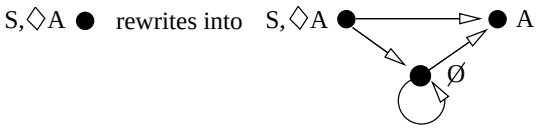
The algorithm consists in using a modified rule for the treatment of density, together with the loop-test of KD4.

Consider the new following rules:

Rule (D^{De}):



Rule ($\Diamond^{De}$):



The set of rules that will be used is: all classical rules, and rules K, 4, $\Diamond^{De}$, and D^{De}.

We will prove below that this set of rules is both complete and sound. I.e. that in order to handle density (together with transitivity) one just need to consider models with only one intermediary world having a reflexive edge.

The algorithm we propose is same as that for KD4, but using rules ($\Diamond^{De}$), and (D^{De}) instead of rules ($\Diamond$) and (D). Also, a node in this algorithm is not considered as its own ancestor (otherwise the loop step would apply immediately!).

It must be applied until for some i, either $\boldsymbol{Y}_i$ is closed or $\boldsymbol{Y}_{i+1} = \boldsymbol{Y}_i$ (i.e. there are marked nodes on each branch).

THEOREM 4. *The strategy given above is sound, complete and terminating for* KD4.De.

2.2. The multimodal logic K(K,T).Per

The kernel is obtained as follows: the set of rules that will be used is: all classical rules, and ($\Diamond_{\mathtt{K}}$), ($\Diamond_{\mathtt{T}}$),($\Box_{\mathtt{T}}$) and ($\Box_{\mathtt{K}}$). The structural rule (*Per*) will be handled in a completion step of our strategy that will consist in applying it at all possible place at once. Kernels for K(K,T).*Per* are simply naive tableaux provided with a strategy that allows to conclude that some tableau is open after only finitely many steps. Let A be the starting formula, we get

the following non-deterministic (w.r.t. the order of application of the rules) algorithm that computes a sequence $(\boldsymbol{Y})_i$ of RDAG:

Starting from $\boldsymbol{Y}_0 = (\boldsymbol{N}_0, \boldsymbol{\Sigma}_0^{\mathtt{K}}, \boldsymbol{\Sigma}_0^{\mathtt{T}}, \boldsymbol{\Phi}_0)$ where $\boldsymbol{N}_0$ only consists of one node $\boldsymbol{r}$ (the root), both $\boldsymbol{\Sigma}_0^{\mathtt{K}}$ and $\boldsymbol{\Sigma}_0^{\mathtt{T}}$ are empty and $\boldsymbol{\Phi}_0$ associate the formula A with $\boldsymbol{r}$. Then compute $\boldsymbol{Y}_{i+1} = (\boldsymbol{N}_{i+1}, \boldsymbol{\Sigma}_{i+1}^{\mathtt{K}}, \boldsymbol{\Sigma}_{i+1}^{\mathtt{T}}, \boldsymbol{\Phi}_{i+1})$ from $\boldsymbol{Y}_i = (\boldsymbol{N}_i, \boldsymbol{\Sigma}_i^{\mathtt{K}}, \boldsymbol{\Sigma}_i^{\mathtt{T}}, \boldsymbol{\Phi}_i)$ by applying successively each of the following steps:

1. Classical step: apply classical rules ($(\bot)$, $(\neg)$, $(\vee)$, $(\wedge)$) on all nodes as much as possible (also known as classical saturation);
2. Structural step: apply rules $(\Diamond_{\mathtt{K}})$ and $(\Diamond_{\mathtt{T}})$ on each node where they have not been applied yet[6];
3. Propagation step: apply rules $(\Box_{\mathtt{T}})$ and $(\Box_{\mathtt{K}})$ as much as possible.
4. Completion step: Detailed below

The above algorithm must be applied until for some i, either $\boldsymbol{Y}_i$ is closed or $\boldsymbol{Y}_{i+1} = \boldsymbol{Y}_i$ (i.e. no new node are created)[7].

Completion. Here we precise how works the completion step, and prove that it is effective in the sense that:

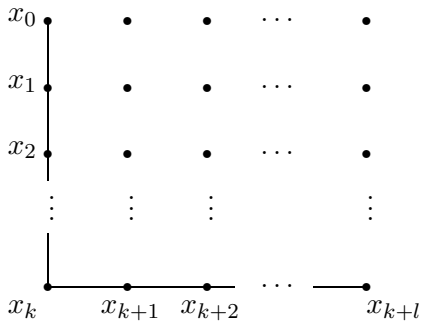
- It terminates
- After it has been performed at step i, $\boldsymbol{\Sigma}_{i+1}^{\mathtt{T}} \circ \boldsymbol{\Sigma}_{i+1}^{\mathtt{K}} \subseteq \boldsymbol{\Sigma}_{i+1}^{\mathtt{K}} \circ \boldsymbol{\Sigma}_{i+1}^{\mathtt{T}}$

DEFINITION 5 (Maximal T-K paths). Let $\boldsymbol{Y}_i = (\boldsymbol{N}_i, \boldsymbol{\Sigma}_i^{\mathtt{K}}, \boldsymbol{\Sigma}_i^{\mathtt{T}}, \boldsymbol{\Phi}_i)$ be the kernel at step i. Let $\mathtt{Path}_i$ be the set of **maximal T-K paths** of any length in $\boldsymbol{Y}_i$, i.e.:
$\mathtt{Path}_i = \{(x_0, \ldots, x_k, x_{k+1}, \ldots, x_{k+l}) / (x_j, x_{j+1}) \in \boldsymbol{\Sigma}_i^{\mathtt{T}}$ for $0 \leq j \leq k-1$, and $(x_j, x_{j+1}) \in \boldsymbol{\Sigma}_i^{\mathtt{K}}$ for $k \leq j \leq k+l-1\}$, moreover these paths are maximal: given a path $\mathtt{pth} = (x_0, \ldots, x_k, x_{k+1}, \ldots, x_{k+l})$ of $\mathtt{Path}_i$, no path of $\mathtt{Path}_i$ contains $\mathtt{pth}$, i.e. $\forall x$: neither $(x, x_0) \in \boldsymbol{\Sigma}_i^{\mathtt{T}}$, nor $(x_{k+l}, x) \in \boldsymbol{\Sigma}_i^{\mathtt{K}}$.

Let $\boldsymbol{Y}_i = (\boldsymbol{N}_i, \boldsymbol{\Sigma}_i^{\mathtt{K}}, \boldsymbol{\Sigma}_i^{\mathtt{T}}, \boldsymbol{\Phi}_i)$ be the kernel at step i. Let $\mathtt{Path}_i$ be as defined in the above definition, it is clear that in order to complete any path of $\mathtt{Path}_i$, we need $k.l$ new nodes (see figure below) in order to ensure that $\boldsymbol{\Sigma}_{i+1}^{\mathtt{T}} \circ \boldsymbol{\Sigma}_{i+1}^{\mathtt{K}} \subseteq \boldsymbol{\Sigma}_{i+1}^{\mathtt{K}} \circ \boldsymbol{\Sigma}_{i+1}^{\mathtt{T}}$.

[6] As usual, rules $(\Diamond_{\mathtt{K}})$ and $(\Diamond_{\mathtt{T}})$ must be applied once for each formula $\Diamond_{\mathtt{K}} B$ and $\Diamond_{\mathtt{T}} B$ of each node.

[7] Since there are no backwards rules, rules $(\Box_{\mathtt{T}})$ and $(\Box_{\mathtt{K}})$ must be applied only in order to propagate formulas in new nodes introduced at step 3.

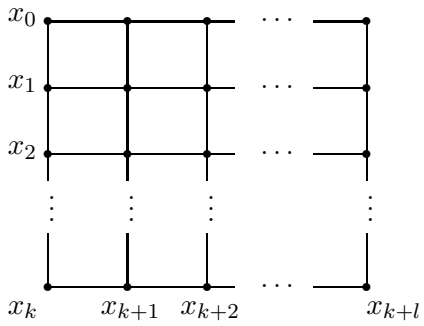


In this picture edges of $\boldsymbol{\Sigma}_i^{\mathtt{T}}$ are vertical, while those of $\boldsymbol{\Sigma}_i^{\mathtt{K}}$ are horizontal. Let us denote the new point at intersection of line u and row v by $x_{u,v}$.

DEFINITION 6 (Completion). Given a path $\mathtt{pth} = (x_0, ..., x_k, x_{k+1}, ..., x_{k+l})$ of $\mathtt{Path}_i$, let

$$\begin{aligned}
\mathtt{NewNodes}_{i+1}(\mathtt{pth}) &= \{x_{u,v} \colon 0 \leq u \leq k-1,\ k+1 \leq v \leq k+l\} \\
&\quad \text{a set of } k.l \text{ new nodes,} \\
\mathtt{NewEdges}^{\mathtt{T}}_{i+1}(\mathtt{pth}) &= \{(x_{u,v}, x_{u+1,v}) \colon 0 \leq u \leq k-2,\ k+1 \leq v \leq k+l\} \\
&\quad \cup \{(x_{k-1,v}, x_v) \colon k+1 \leq v \leq k+l\} \\
\mathtt{NewEdges}^{\mathtt{K}}_{i+1}(\mathtt{pth}) &= \{(x_{u,v}, x_{u,v+1}) \colon 0 \leq u \leq k-1,\ k+1 \leq v \leq k+l-1\} \\
&\quad \cup \{(x_u, x_{u,k+1}) \colon 0 \leq u \leq k-1\}
\end{aligned}$$

Graphically :



Then applying completion to $\boldsymbol{Y}_i = (\boldsymbol{N}_i, \boldsymbol{\Sigma}_i^{\mathtt{K}}, \boldsymbol{\Sigma}_i^{\mathtt{T}}, \boldsymbol{\Phi}_i)$ gives $\boldsymbol{Y}_{i+1} = (\boldsymbol{N}_{i+1}, \boldsymbol{\Sigma}_{i+1}^{\mathtt{K}}, \boldsymbol{\Sigma}_{i+1}^{\mathtt{T}}, \boldsymbol{\Phi}_{i+1})$ defined by:

- $\boldsymbol{N}_{i+1} = \boldsymbol{N}_i \cup (\bigcup_{\mathtt{pth} \in \mathtt{Path}_i} \mathtt{NewNodes}_{i+1}(\mathtt{pth}))$

- $\boldsymbol{\Sigma}^{\mathtt{K}}_{i+1} = \boldsymbol{\Sigma}^{\mathtt{K}}_{i} \cup (\bigcup_{\mathtt{pth}\in\mathtt{Path}_i} \mathtt{NewEdges}^{\mathtt{K}}_{i+1}(\mathtt{pth}))$
- $\boldsymbol{\Sigma}^{\mathtt{T}}_{i+1} = \boldsymbol{\Sigma}^{\mathtt{T}}_{i} \cup (\bigcup_{\mathtt{pth}\in\mathtt{Path}_i} \mathtt{NewEdges}^{\mathtt{T}}_{i+1}(\mathtt{pth}))$
- $\boldsymbol{\Phi}_{i+1} = \boldsymbol{\Phi}_{i} \cup (\bigcup_{\mathtt{pth}\in\mathtt{Path}_i} \{(x,\emptyset)\colon x \in \mathtt{NewNodes}_{i+1}(\mathtt{pth})\})$

It is straightforward to check that $\boldsymbol{\Sigma}^{\mathtt{T}}_{i+1} \circ \boldsymbol{\Sigma}^{\mathtt{K}}_{i+1} \subseteq \boldsymbol{\Sigma}^{\mathtt{K}}_{i+1} \circ \boldsymbol{\Sigma}^{\mathtt{T}}_{i+1}$. Thus completion is effective in the sense given below (terminating, and completing). It simply compiles several successive applications of the rule (Per) at once.

THEOREM 5. *The strategy given above is sound for* K(K,T).*Per*.

PROOF. Straightforward since the resulting algorithm is a restriction (on the order of application of the rules) of the naive one which is sound. Hence, if there is an open naive tableau for A, there is a fortiori an open kernel for A. □

THEOREM 6. *The strategy given above is complete for* K(K,T).*Per*.

PROOF. If there is an open kernel for A, then there is an open naive tableau for A: the fact that for some i, $\boldsymbol{Y}_{i+1} = \boldsymbol{Y}_i$ (i.e. no new node are created) proves that if the computation goes on, no closure can occur. □

Now we come to the termination argument which is standard:

LEMMA 1. *The strategy given above is terminating.*

PROOF. Since completion step is effective, each step does indeed terminate. Now, given a path pth, it is easy to show that the average modal degree of nodes introduced in $\mathtt{NewNodes}_{i+1}(\mathtt{pth})$ (i.e. the modal degree of each nodes divided by the number of nodes) is strictly less than that of nodes of pth since propagation rules strictly decrease the modal degree of formulas. Hence, at some iteration, only empty nodes that will remain empty will be generated. □

References

[1] Laurent Catach, *Les logiques multi-modales*, Ph.D. thesis, Université Paris VI, France, 1989.

[2] G. De Giacomo, F. Massacci. "Tableau and algorithm for propositional dynamic logic with converse". In M. A. McRobbie, J. K. Slaney editor, *Proc. CADE-13, LNAI 1104, Springer, 1996.*

[3] S. Demri, E. Orłowska. "Every finitely reductible logic has the finite model property with respect to the class of ◇-formulae". Bulletin of the Section of Logic, to appear in Studia Logica.

[4] H. C. M. de Swart. "Gentzen-type systems for C, K and several extensions of C and K; constructive completeness proofs and effective decision procedure for these systems". Logique et Analyse, 1980.

[5] Marcos A. Castilho, Luis Fariñas del Cerro, Olivier Gasquet, and Andreas Herzig. "Modal tableaux with propagation rules and structural rules". Fundamenta Informaticæ 32(3/4):281–297, 1997.

[6] Luis Fariñas del Cerro, and Olivier Gasquet. "Tableaux based decision procedures for modal logics of confluence and density". Fundamenta Informaticæ 41(1):1–17, 2000.

[7] M. Fitting. *Proof Methods for Modal and Intuitionistic Logics*. D. Reidel, Dordrecht, 1983.

[8] M. Fitting. "Basic modal logic". In D. Gabbay et al., editor, *Handbook of Logic in Artificial Intelligence and Logic Programming: Logical Foundations*, vol. 1, Oxford University Press, 1993.

[9] S. Kripke. "A completeness theorem in modal logic". Journal of Symbolic Logic 24, 1959.

[10] S. Kripke. "Semantical analysis of modal logic I: Normal modal propositional calculi". Zeitschrift für Mathematische Logik und Grundlagen der Mathematik 9, 1963.

[11] F. Massacci. "Strongly analytic tableaux for normal modal logics". In Alan Bundy editor, *Proc. CADE-12, LNAI 814, Springer, 1994.*

[12] E. Orłowska. "Relational formalization of non-classical logics". In Brink, Kahl, Schmidt editor, *Relational Methods in Computer Science*, pp. 91–105, Springer-Verlag, 1997.

[13] H. Sahlqvist. "Completeness and correspondence in the first and second order semantics for modal logics". In S. Kanger editor, *Proc. 3rd Scandinavian Logic Symposium 1973*, Studies in Logic 82 (1975), North-Holland, 1973

LUIS FARIÑAS DEL CERRO and OLIVIER GASQUET
IRIT-UPS
118 route de Narbonne
Toulouse Cedex 04, France
{farinas,gasquet}@irit.fr