

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 $S \cup \{A\}$:

$$S \bullet \implies S, A \bullet$$

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

$$S \bullet \implies S \bullet \text{---} S1$$

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

$$S0 \bullet \text{---} S1 \implies S0, A \bullet \text{---} S1, B$$

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

$$S0 \bullet \text{---} S1 \text{---} S2 \implies S0, A \bullet \text{---} S1, B \text{---} S2, C$$

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

$$S0 \begin{array}{l} \bullet S1 \\ \diagdown \quad \diagup \\ \bullet S2 \end{array} \implies S0, A \begin{array}{l} \bullet S1, B \\ \diagdown \quad \diagup \\ \bullet S2, C \end{array}$$

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

$$S0 \begin{array}{l} \bullet S1 \\ \diagdown \quad \diagup \\ \bullet S2 \end{array} \implies S0 \begin{array}{l} \bullet S1 \\ \diagdown \quad \diagup \\ \bullet S2 \end{array} \begin{array}{l} \bullet S3 \\ \diagdown \quad \diagup \\ \bullet S2 \end{array}$$

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

$$S0 \bullet \text{---} S1 \implies S0 \begin{array}{l} \bullet S2 \\ \diagdown \quad \diagup \\ \bullet S1 \end{array}$$

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

$$S0 \begin{array}{l} \bullet T \\ \diagdown \quad \diagup \\ \bullet S1 \end{array} \begin{array}{l} \bullet K \\ \diagdown \quad \diagup \\ \bullet S3 \end{array} \implies S0 \begin{array}{l} \bullet T \\ \diagdown \quad \diagup \\ \bullet S1 \end{array} \begin{array}{l} \bullet K \\ \diagdown \quad \diagup \\ \bullet S3 \end{array} \begin{array}{l} \bullet S4 \\ \diagdown \quad \diagup \\ \bullet S1 \end{array} \begin{array}{l} \bullet T \\ \diagdown \quad \diagup \\ \bullet S3 \end{array}$$

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

$$S0 \bullet \text{---} S1, \Box A \bullet S2 \implies S0 \bullet \text{---} S1, \Box A \bullet S2, \Box A$$

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 \perp : $A, \neg A, S \implies A, \neg A, \perp, S$
- Rule \neg : $\neg \neg A, S \implies \neg \neg A, A, S$
- Rule \wedge : $A \wedge B, S \implies A \wedge B, A, B, S$
- Rule \vee : $\neg(A \wedge B), S \implies \neg(A \wedge B), C, S$
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 : $\diamond A, S \implies \diamond A, S, \text{---}A$

And its multimodal versions:

- Rule \diamond_K : $\diamond_K A, S \implies \diamond_K A, S, \text{---}K.A$
- Rule \diamond_T : $\diamond_T A, S \implies \diamond_T A, S, \text{---}T.A$

- Propagation rules:

- Rule K : $\Box A, S, \text{---}S1 \implies \Box A, S, \text{---}A, S1$
- Rule 4: $S, \Box A, \text{---}S1 \implies S, \Box A, \text{---}S1, \Box A$

And multimodal versions of Rule K :

- Rule \Box_K : $\Box_K A, S, \text{---}K.S1 \implies \Box_K A, S, \text{---}K.A, S1$
- Rule \Box_T : $\Box_T A, S, \text{---}T.S1 \implies \Box_T A, S, \text{---}T.A, S1$

- Structural rules:

- Rule D : $S \bullet \implies S, \text{---}\emptyset$
- Rule C : $S0 \begin{array}{l} \nearrow S1 \\ \searrow S2 \end{array} \implies S0 \begin{array}{l} \nearrow S1 \\ \searrow S2 \end{array} \rightarrow \emptyset$
- Rule C^* : $S \bullet \text{---}S1 \implies S, \text{---}S1, \text{---}\emptyset$
- Rule De : $S0 \text{---}S1 \implies S0 \begin{array}{l} \nearrow \emptyset \\ \searrow S1 \end{array}$
- Rule Per : $S0 \begin{array}{l} \nearrow T \\ \searrow K \end{array} \text{---}S1 \text{---}S2 \implies S0 \begin{array}{l} \nearrow T \\ \searrow K \end{array} \begin{array}{l} \nearrow S1 \\ \searrow T \end{array} \text{---}\emptyset$



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_K and \diamond_T in the multimodal case)
- Rule K (or \Box_K and \Box_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	<i>Ref</i> <i>Tr</i>	T 4	Propagation Rules
Group 2	<i>Ser</i> <i>Dens</i> <i>Conf</i> <i>Per</i>	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 $\mathcal{T}_0, \dots, \mathcal{T}_i, \mathcal{T}_{i+1}, \dots$ where:



- \mathcal{T}_0 is an RDAG consisting of only one node whose associated set of formulas is $\{A\}$,
- \mathcal{T}_{i+1} is obtained from \mathcal{T}_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 \perp ; 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 \mathbf{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 $(\mathbf{Y})_i$ of RDAG:

Starting from $\mathbf{Y}_0 = (\mathbf{N}_0, \Sigma_0, \Phi_0)$ where \mathbf{N}_0 only consists of one node r (the root), Σ_0 is empty and Φ_0 associate the formula A with r . Then compute $\mathbf{Y}_{i+1} = (\mathbf{N}_{i+1}, \Sigma_{i+1}, \Phi_{i+1})$ from $\mathbf{Y}_i = (\mathbf{N}_i, \Sigma_i, \Phi_i)$ by applying successively each of the following steps:

1. Loop step: consider all nodes $x \in \mathbf{N}_i$ such that $\exists y \in \mathbf{N}_i$ and y is an ancestor of x (i.e. $(y, x) \in \Sigma_i^+$) and $\Phi_i(x) \subseteq \Phi_i(y)$ 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 ((\perp) , (\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 yet⁴;
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 \mathbf{Y}_i is closed or $\mathbf{Y}_{i+1} = \mathbf{Y}_i$ (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 $\mathbf{Y}_0 = (\mathbf{N}_0, \mathbf{\Sigma}_0, \mathbf{\Phi}_0)$ where \mathbf{N}_0 only consists of only one node r^0 (the root), $\mathbf{\Sigma}_0$ is empty and $\mathbf{\Phi}_0$ associate the formula A with r^0 .

1. Compute $\mathbf{Y}_{i+1} = (\mathbf{N}_{i+1}, \mathbf{\Sigma}_{i+1}, \mathbf{\Phi}_{i+1})$ from $\mathbf{Y}_i = (\mathbf{N}_i, \mathbf{\Sigma}_i, \mathbf{\Phi}_i)$ by applying the strategy for KD4 only (i.e. by using only classical rules, and rules (D), (\diamond), (K) and (4)). In \mathbf{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 \mathbf{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.

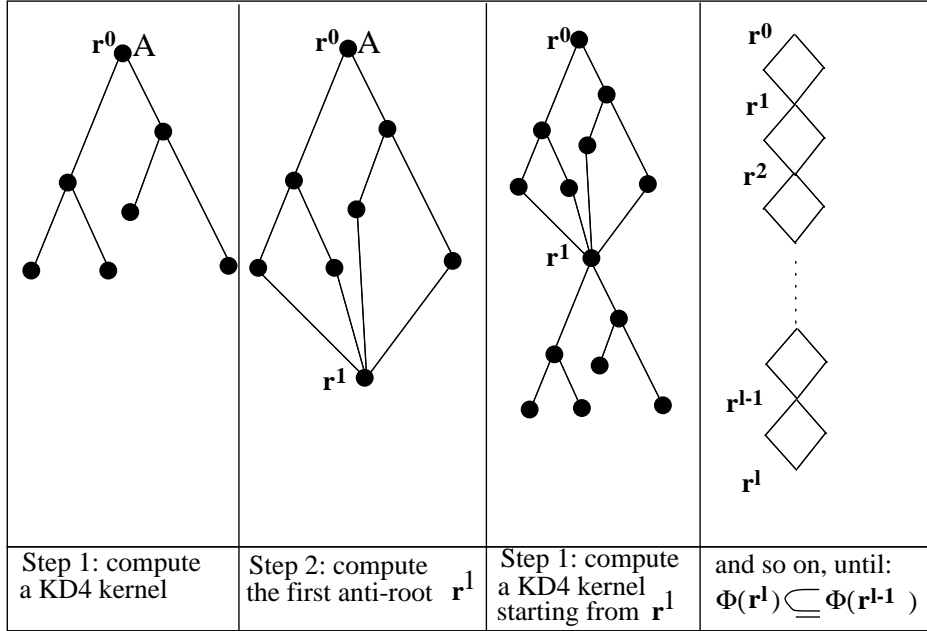


Figure 1. The strategy for KD4.C

2. Compute $Y_{i+2} = (N_{i+2}, \Sigma_{i+2}, \Phi_{i+2})$ from $Y_{i+1} = (N_{i+1}, \Sigma_{i+1}, \Phi_{i+1})$ by:

- $N_{i+2} = N_{i+1} \cup \{r^{i+2}\}$, where r^{i+2} is a new node,
- $\Sigma_{i+2} = \Sigma_{i+1} \cup \{(x, r^{i+2}) : x \in Loop_{i+1}\}$
- $\Phi_{i+2}(r^{i+2}) = \bigcup_{x \in Loop_{i+1}} (\Phi_{i+1}(x))^{\square}$; where as usual, S^{\square} denotes the set $\{A, \square A : \square A \in S\}$

The above algorithm must be applied until for some i and some l , either Y_i is closed or $\Phi_i(r^l) \subseteq \Phi_i(r^{l-1})$, where 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.*

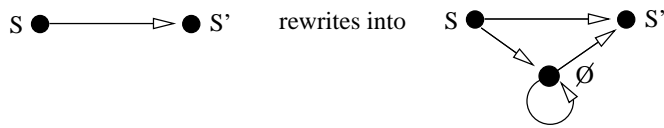
2.1.3. Kernels for KD4.De

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).

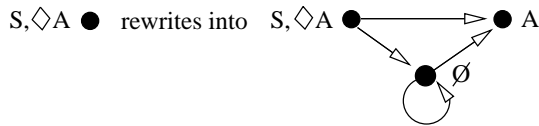
The algorithm consists in using a modified rule for the treatment of density, together with the loop-test of KD4.

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 \mathbf{Y}_i is closed or $\mathbf{Y}_{i+1} = \mathbf{Y}_i$ (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_K), (\diamond_T), (\square_T) and (\square_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 $(\mathbf{Y})_i$ of RDAG:

Starting from $\mathbf{Y}_0 = (\mathbf{N}_0, \Sigma_0^K, \Sigma_0^T, \Phi_0)$ where \mathbf{N}_0 only consists of one node \mathbf{r} (the root), both Σ_0^K and Σ_0^T are empty and Φ_0 associate the formula A with \mathbf{r} . Then compute $\mathbf{Y}_{i+1} = (\mathbf{N}_{i+1}, \Sigma_{i+1}^K, \Sigma_{i+1}^T, \Phi_{i+1})$ from $\mathbf{Y}_i = (\mathbf{N}_i, \Sigma_i^K, \Sigma_i^T, \Phi_i)$ by applying successively each of the following steps:

1. Classical step: apply classical rules $((\perp), (\neg), (\vee), (\wedge))$ on all nodes as much as possible (also known as classical saturation);
2. Structural step: apply rules (\diamond_K) and (\diamond_T) on each node where they have not been applied yet⁶;
3. Propagation step: apply rules (\square_T) and (\square_K) as much as possible.
4. Completion step: Detailed below

The above algorithm must be applied until for some i , either \mathbf{Y}_i is closed or $\mathbf{Y}_{i+1} = \mathbf{Y}_i$ (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 , $\Sigma_{i+1}^T \circ \Sigma_{i+1}^K \subseteq \Sigma_{i+1}^K \circ \Sigma_{i+1}^T$

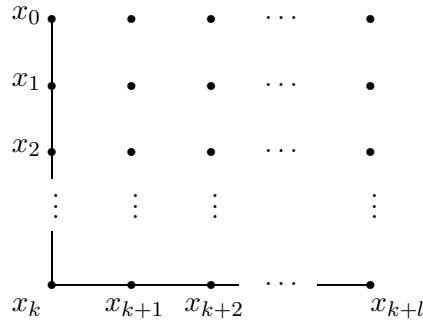
DEFINITION 5 (Maximal T-K paths). Let $\mathbf{Y}_i = (\mathbf{N}_i, \Sigma_i^K, \Sigma_i^T, \Phi_i)$ be the kernel at step i . Let Path_i be the set of **maximal T-K paths** of any length in \mathbf{Y}_i , i.e.:

$\text{Path}_i = \{(x_0, \dots, x_k, x_{k+1}, \dots, x_{k+l}) / (x_j, x_{j+1}) \in \Sigma_i^T \text{ for } 0 \leq j \leq k-1, \text{ and } (x_j, x_{j+1}) \in \Sigma_i^K \text{ for } k \leq j \leq k+l-1\}$, moreover these paths are maximal: given a path $\text{pth} = (x_0, \dots, x_k, x_{k+1}, \dots, x_{k+l})$ of Path_i , no path of Path_i contains pth , i.e. $\forall x$: neither $(x, x_0) \in \Sigma_i^T$, nor $(x_{k+l}, x) \in \Sigma_i^K$.

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

⁶ As usual, rules (\diamond_K) and (\diamond_T) must be applied once for each formula $\diamond_K B$ and $\diamond_T B$ of each node.

⁷ Since there are no backwards rules, rules (\square_T) and (\square_K) must be applied only in order to propagate formulas in new nodes introduced at step 3.



In this picture edges of Σ_i^T are vertical, while those of Σ_i^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 $\text{pth} = (x_0, \dots, x_k, x_{k+1}, \dots, x_{k+l})$ of Path_i , let

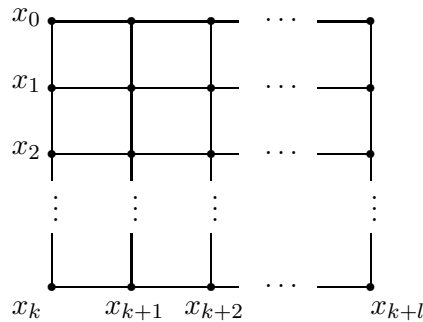
$$\text{NewNodes}_{i+1}(\text{pth}) = \{x_{u,v} : 0 \leq u \leq k-1, k+1 \leq v \leq k+l\}$$

a set of $k \cdot l$ new nodes,

$$\text{NewEdges}_{i+1}^T(\text{pth}) = \{(x_{u,v}, x_{u+1,v}) : 0 \leq u \leq k-2, k+1 \leq v \leq k+l\} \\ \cup \{(x_{k-1,v}, x_v) : k+1 \leq v \leq k+l\}$$

$$\text{NewEdges}_{i+1}^K(\text{pth}) = \{(x_{u,v}, x_{u,v+1}) : 0 \leq u \leq k-1, k+1 \leq v \leq k+l-1\} \\ \cup \{(x_u, x_{u,k+1}) : 0 \leq u \leq k-1\}$$

Graphically :



Then applying completion to $Y_i = (N_i, \Sigma_i^K, \Sigma_i^T, \Phi_i)$ gives $Y_{i+1} = (N_{i+1}, \Sigma_{i+1}^K, \Sigma_{i+1}^T, \Phi_{i+1})$ defined by:

- $N_{i+1} = N_i \cup (\bigcup_{\text{pth} \in \text{Path}_i} \text{NewNodes}_{i+1}(\text{pth}))$



- $\Sigma_{i+1}^K = \Sigma_i^K \cup (\bigcup_{\text{pth} \in \text{Path}_i} \text{NewEdges}_{i+1}^K(\text{pth}))$
- $\Sigma_{i+1}^T = \Sigma_i^T \cup (\bigcup_{\text{pth} \in \text{Path}_i} \text{NewEdges}_{i+1}^T(\text{pth}))$
- $\Phi_{i+1} = \Phi_i \cup (\bigcup_{\text{pth} \in \text{Path}_i} \{(x, \emptyset) : x \in \text{NewNodes}_{i+1}(\text{pth})\})$

It is straightforward to check that $\Sigma_{i+1}^T \circ \Sigma_{i+1}^K \subseteq \Sigma_{i+1}^K \circ \Sigma_{i+1}^T$. 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 . \square

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 , $\mathbf{Y}_{i+1} = \mathbf{Y}_i$ (i.e. no new node are created) proves that if the computation goes on, no closure can occur. \square

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 $\text{NewNodes}_{i+1}(\text{pth})$ (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. \square

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 reducible logic has the finite model property with respect to the class of \diamond -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`