Proc. Estonian Acad. Sci. Phys. Math., 1998, **47**, 3, 174–185
https://doi.org/10.3176/phys.math.1998.3.03

PREFIXED TABLEAUS FOR THREE-VALUED MODAL PROPOSITIONAL LOGICS

Jūratė SAKALAUSKAITĖ

Institute of Mathematics and Informatics, Akademijos 4, 2600 Vilnius, Lithuania; e-mail: jurates@ktl.mii.lt

Received 19 January 1998

Abstract. Prefixed tableau systems for some of the three-valued modal propositional logics are given. The soundness and completeness theorems for these tableau systems with respect to Kripke-type semantics are proved.

Key words: tableau systems, three-valued logic, modal logic, completeness.

1. INTRODUCTION

Finitely-valued logics have numerous applications to computer science in such areas as hardware verification, nonmonotonic reasoning, theory of logic programming. This has aroused interest in the investigation of automated theorem proving for finitely-valued logics.

In this paper tableau-based theorem proving for some of the three-valued modal propositional logics is discussed. We consider three-valued counterparts (two for each logic) of well-known two-valued modal logics K, $K4$, T, B, $S4$, $S5$ (see, e.g. [1]) denoting them by K_i, $K4_i$, T_i, B_i, $S4_i$, $S5_i, i \in \{2,3\}$. A notion of Kripke frame for these logics is as in the two-valued case, but now truth values of formulas at possible worlds are from the set {true, false, undefined}. These counterparts (except for logics $K4$ and B) are considered by Takano in [2], where the cut free sequent calculi for these logics are constructed.

In this paper the formal proof procedures called the prefixed tableau systems are introduced for the logics K_i, $K4_i$, T_i, B_i, $S4_i$, $S5_i$, $i \in \{2,3\}$ and soundness and completeness theorems are proved for these systems. The importance of the prefixed tableaus is that they provide us with the algorithm to determine whether a formula is valid.

Prefixed tableau systems for the two-valued modal case were elaborated by Fitting in [1]. These are refutation procedures, i.e. we attempt to obtain a falsifying model of possible worlds for a given formula. In a prefixed tableau each formula has a prefix. Prefixes of formulas represent the names of possible worlds and the accessibility relation between the worlds is reflected by syntactic features of these names.

The paper is organized as follows. In Section 2 we introduce the syntax and semantics of the three-valued modal logics. In Section 3 the prefixed tableau systems for these logics are presented and soundness of the systems is proved. In Section 4 we prove the completeness for these systems. Section 5 contains some concluding remarks.

2. SYNTAX AND SEMANTICS

We put $T = \{1,\ 2,\ 3\}$ and shall use T as the set of *truth values*. Intuitively, the truth values 1, 2, 3 stand for "true", "undefined", and "false", respectively. We let λ, μ, ν, ... denote truth values.

Formulas are constructed from propositional variables by means of propositional connectives and the necessity operator $\Box$; we assume that for each propositional connective F the arity $\alpha(F) \geq 0$ and the truth function $f_F : T^{\alpha(F)} \to T$ are predetermined.

Definition 2.1. *A* valued formula *is any pair of a formula and a truth value.*

Definition 2.2. *A* (three-valued) Kripke frame *is the triplet* $(W,\ R,\ v)$, *where* W *is a nonempty set* (*set of worlds*), R *is a binary relation on* W, *and* v *is a mapping which assigns a truth value from* T *to each pair of a propositional variable and an element of* W.

Definition 2.3. *Suppose that* $(W,\ R,\ v)$ *is a Kripke frame and* $i \in \{2,3\}$. *We call the triplet* $(W,\ R,\ v^i)$ *a* (three-valued) Kripke structure of type i (*generated from* $(W,\ R,\ v)$) *if* v^i *is the mapping which assigns a truth value to each pair of a formula and an element of* W *and is defined by recursion as follows*:

$v^i(p,s) = v(p,s)$, *where* p *is a propositional variable*;

$$v^i(F(A_1,\ldots,A_{\alpha(F)}),s) = f_F(v^i(A_1,s),\ldots,v^i(A_{\alpha(F)},s));$$

$$v^2(\Box A,s) = \begin{cases} 1, & \text{if } sRt \text{ implies } v^2(A,t) = 1 \text{ for every } t \in W; \\ 2, & \text{if } sRt \text{ and } v^2(A,t) = 2 \text{ for some } t \in W; \\ 3, & \text{otherwise.} \end{cases}$$

$$v^3(\Box A,s) = \begin{cases} 1, & \text{if } sRt \text{ implies } v^3(A,t) = 1 \text{ for every } t \in W; \\ 3, & \text{otherwise.} \end{cases}$$

Let L be a modal logic. Models of L are defined as follows.

Definition 2.4. Models of K_i *are nothing but the Kripke structures of type* i, *whereas a* model of T_i, $K4_i$, B_i, $S4_i$ *is a Kripke structure* (W, R, v^i) *of type* i *such that* R *is reflexive, transitive, reflexive and symmetric, reflexive and transitive, respectively. A model of* $S5_i$ *is a Kripke structure* (W, R, v^i), *where* R *is reflexive, symmetric and transitive.*

Let L_i be one of the logics we are considering.

Definition 2.5. *A valued formula* (A, μ) *is* L_i-satisfiable *if there exist a model* (W, R, v^i) *of* L_i *and a world* $s \in W$ *such that* $v^i(A, s) = \mu$. *A valued formula* (A, μ) *is* L_i-valid *if for each model* (W, R, v^i) *of* L_i *and for each world* $s \in W$, $v^i(A, s) = \mu$.

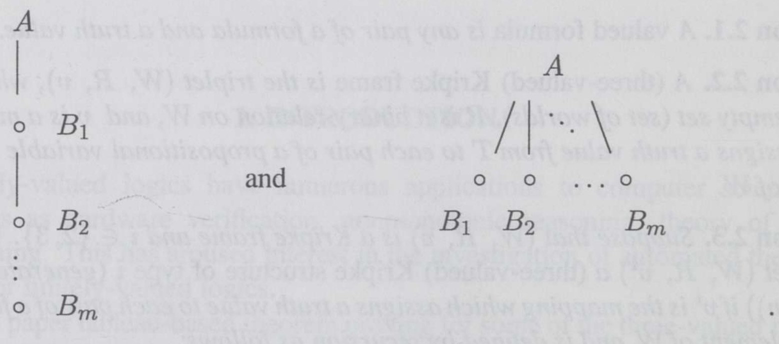
Note that a valued formula (A, μ) is L_i-valid iff valued formulas (A, μ_1), (A, μ_2), where $\mu_1, \mu_2 \in T - \{\mu\}$, are L_i-unsatisfiable.

3. PREFIXED TABLEAUS

We shall consider informally the notions of trees, branches, nodes, etc. The symbols

$$\frac{A}{B_1 \bigcirc B_2 \bigcirc \ldots \bigcirc B_m} \quad \text{and} \quad \frac{A}{B_1 + B_2 + \ldots + B_m}$$

will respectively denote trees of the following form



Those symbols will be abbreviated by the following expressions, respectively:

$$\frac{A}{\bigcirc\{B_i : i \leq m\}}, \qquad \frac{A}{+\{B_i : i \leq m\}}.$$

Definition 3.1. *A* prefix *is a finite sequence of positive integers. A* prefixed valued formula $\sigma(A, \lambda)$ *is a prefix* σ *followed by a valued formula* (A, λ).

We shall systematically use σ, σ', etc. for prefixes throughout this paper. The prefixes will be interpreted as naming worlds in some model. $\sigma(A, \lambda)$ means that under this model A is forced to have the value λ in the world σ names.

We define tableau rules as follows. These rules are divided into two parts: the rules for formulas of the form $F(A_1, \ldots, A_m)$, where F is a propositional connective, and the rules for modalized formulas. We begin with formulas of the form $F(A_1, \ldots, A_m)$ and present the rules for these formulas following [3]. (In fact, these rules are obtained from the definition of the rules for the formulas of the form $F(A_1, \ldots, A_m)$ from [3] by omitting the first condition.)

For each prefixed valued formula $\sigma(F(A_1, \ldots, A_m), \lambda)$, where F is an m-ary propositional connective, we define the rule as follows:

$$\frac{\sigma\big(F(A_1, \ldots, A_m), \lambda\big)}{\begin{array}{c}+\{\sigma(A_{i_1}, \lambda_{j_1}) \bigcirc \ldots \bigcirc \sigma(A_{i_t}, \lambda_{j_t}) : \quad \lambda_{j_1}, \ldots, \lambda_{j_t} \leq 3,\ t \leq m \\ \text{and the propositional condition } H_\lambda(F; \lambda_{j_1}, \ldots, \lambda_{j_t}) \text{ holds}\}\end{array}},$$

where $H_\lambda(F; \lambda_{j_1}, \ldots, \lambda_{j_t})$ means that

(1) if f represents the connective F, then

$$f(\nu_1, \ldots, \nu_{i_1}, \ldots, \nu_{i_2}, \ldots, \nu_{i_t}, \ldots, \nu_m) = \lambda$$

for all values of the function f, where $\nu_{i_k} = \lambda_{j_k}$ and other ν's are arbitrary; and

(2) no $t' < t$ satisfies (1).

Before presenting rules for modalized formulas we give some more terminology. The following two definitions are borrowed from [1].

Definition 3.2. *We say a prefix σ is* used *on a tableau branch if σZ occurs on the branch for some valued formula Z. We say a prefix σ is* unrestricted *on a tableau branch if σ is not an initial segment (proper or otherwise) of any prefix used on the branch.*

Let σ be an arbitrary prefix.

Definition 3.3. *We say the relation of* "accessibility from" *on prefixes satisfies*:

(1) *the* general *condition if σ, n is accessible from σ for every integer n;*

(2) *the* reverse *condition if σ is accessible from σ, n for every integer n;*

(3) *the* identity *condition if σ is accessible from σ;*

(4) *the* transitivity *condition if the sequence σ, σ' is accessible from σ for every non-empty sequence σ'.*

For the various logics we are considering, the conditions which the accessibility relation on prefixes satisfies are given in the following chart.

Logic	Conditions
K_i	General
$K4_i$	General, transitivity
T_i	General, identity
B_i	General, identity, reverse
$S4_i$	General, identity, transitivity
$S5_i$	No special conditions, any prefix is accessible from any other

Definition 3.4. *A prefixed formula $\sigma(A, \lambda)$ which occurs over the line of a rule is called a* premise *of the rule. Let $B_1 + \ldots + B_j$ be the expression below the line in a tableau rule. We say that $B_i, 1 \leq i \leq j$, is a* consequence *of this rule.*

Now we present the tableau rules for modalized formulas. Each of these rules has conditions that depend on a branch which is supposed to be extended by this rule. So, let θ be a branch such that θ contains the premise of the rule and θ will be extended by this rule.

Let L_i be a logic we are considering, i.e. $i \in \{2, 3\}$, $L \in \{K, K4, T, B, S4, S5\}$.

For a logic L_2 these rules are defined as follows:

R1 $\dfrac{\sigma(\Box A, 1)}{\sigma'(A, 1)}$, where σ' has been used on θ and is L_2-accessible from σ.

R2 $\dfrac{\sigma(\Box A, 2)}{\sigma, n(A, 2)}$, where σ, n is an unrestricted prefix on θ.

R3 $\dfrac{\sigma(\Box A, 3)}{\sigma'(A, 1) + \sigma'(A, 3)}$, where σ' has been used on θ and is L_2-accessible from σ.

R4 $\dfrac{\sigma(\Box A, 3)}{\sigma, n(A, 3)}$, where σ, n is an unrestricted prefix on θ.

For a logic L_3 the tableau rules for modalized formulas are defined as follows:

R$'$1 $\dfrac{\sigma(\Box A, 1)}{\sigma'(A, 1)}$, where σ' has been used on θ and is L_3-accessible from σ.

R$'$2 $\dfrac{\sigma(\Box A, 3)}{\sigma, n(A, 2) + \sigma, n(A, 3)}$, where σ, n is an unrestricted prefix on θ.

A tableau is a tree, with each node labelled with a prefixed formula.

Definition 3.5. *A L_i-*tableau *for a formula (A, λ) is any tree whose first node is the formula $1(A, \lambda)$ and the next nodes are determined by the following procedure: if a branch of the tree contains a prefixed formula σZ and a tableau rule with the premise σZ for the logic L_i is defined, then this branch can be extended by adding new nodes through the application of this rule to σZ (following the convention that formulas separated by "$\bigcirc$" go into the same branch and sets of formulas separated by "+" go into different branches).*

Definition 3.6. *A L_i-tableau branch is* closed *if it contains both $\sigma(A, \lambda)$ and $\sigma(A, \mu)$, $\lambda \neq \mu$, or the branch contains some nonatomic prefixed formula $\sigma(A, \lambda)$ and there exists no defined tableau rule with the premise $\sigma(A, \lambda)$ for the logic L_i. A tableau is closed if each branch of it is closed.*

Example. Consider the logic K_2. Let the propositional binary connective $\supset$ be defined as follows:

$\supset$	1	3	2
1	1	3	2
3	1	1	1
2	1	2	2

Then the tableau rules for $\supset$ are as follows:

$$\frac{\sigma(A \supset B, 1)}{\sigma(B,1) + \sigma(A,3)} (\supset_1)$$

$$\frac{\sigma(A \supset B, 3)}{\sigma(A,1) \circ \sigma(B,3)} (\supset_3)$$

$$\frac{\sigma(A \supset B, 2)}{\sigma(A,1) \circ \sigma(B,2) + \sigma(A,2) \circ \sigma(B,3) + \sigma(A,2) \circ \sigma(B,2)} (\supset_2).$$

We construct a K_2-tableau for the formula $(\Box(A \supset B) \supset (\Box A \supset \Box B), 3)$ as follows:

1. $1(\Box(A \supset B) \supset (\Box A \supset \Box B), 3)$
2. $1(\Box(A \supset B), 1)$ from 1. by $\supset_3$
3. $1(\Box A \supset \Box B, 3)$ from 1. by $\supset_3$
4. $1(\Box A, 1)$ from 3. by $\supset_3$
5. $1(\Box B, 3)$ from 3. by $\supset_3$
6. $1, 1(B, 3)$ from 5. by R4
7. $1, 1(A \supset B, 1)$ from 2. by R1
8. $1, 1(A, 1)$ from 4. by R1

9a. $1, 1\ (B,\ 1)$ from 7. by $\supset_1$ 9b. $1, 1(A,\ 3)$ from 7. by $\supset_1$

× ×

The tableau is closed.

The definition of satisfiability below shows that we treat a tableau as the disjunction of its branches and a branch as the conjunction of its nodes. Let L be a logic we are considering. For short, we refer to the accessibility notion on prefixes that is appropriate for L as L-accessibility.

Let L_i be a logic we are considering, i.e. $i \in \{2, 3\}$, $L \in \{K, K4, T, B, S4, S5\}$. Let S be a set of prefixed formulas and let $\mathcal{M} = (W, R, v^i)$ be an L_i-model.

Definition 3.7. *By an L_i-*interpretation *of S in the model $\mathcal{M}$ we mean a mapping I from the set of prefixes that occur in S to W such that if a prefix τ is L_i-accessible from a prefix σ, then $I(\sigma)RI(\tau)$. S is L_i-*satisfiable under the L_i-interpretation I *if, for each $\sigma(A, \lambda) \in S$, $v^i(A, I(\sigma)) = \lambda$. S is L_i-*satisfiable *if S is L_i-satisfiable under some L_i-interpretation.*

Loosely, a set of prefixed formulas is L_i-satisfiable if it partially describes some model.

Definition 3.8. *A tableau is* L_i-satisfiable *if some branch of it is L_i-satisfiable. A branch is* L_i-satisfiable *if the set of prefixed formulas on it is L_i-satisfiable.*

Note. This use of the term L_i-satisfiable should not be confused with L_i-satisfiable for valued formulas (Definition 2.5 above).

Let L_i be a logic as above.

Lemma 3.9. *Suppose $\mathcal{T}$ is a prefixed tableau that is L_i-satisfiable. Let $\mathcal{T}'$ be the tableau that results from a single L_i-tableau rule being applied to $\mathcal{T}$. Then $\mathcal{T}'$ is also L_i-satisfiable.*

Proof. As the proof of Lemma 3.1 in Chapter 8 in [1]. □

Corollary 3.10 (soundness). *If there exists a closed L_i-tableau for (A, λ), then the valued formula (A, λ) is L_i-unsatisfiable.*

Proof. Suppose that there exists a closed tableau for (A, λ), but there are an L_i-model (W, R, v^i) and a world $s \in W$ such that $v^i(A, s) = \lambda$. Define an L_i-interpretation I by setting $I(1) = s$. It follows that the starting L_i-tableau $\{1(A, \lambda)\}$ is L_i-satisfiable. Then, by Lemma 3.9, every subsequent L_i-tableau is L_i-satisfiable. But an L_i-satisfiable tableau cannot be closed, which contradicts the assumption. □

4. COMPLETENESS

In this section the completeness theorem is proved for every logic L we are considering. The proof of this theorem follows the proof of completeness for prefixed tableaus for two-valued modal logics presented in [1].

We describe a systematic procedure for constructing tableaus for each logic L_i, $i \in \{2, 3\}$, $L \in \{K, K4, T, B, S4, S5\}$. This procedure must produce a closed tableau if one exists and if there is no information necessary to construct a model.

Certain formulas in a tableau must be considered over and over in any reasonable proof procedure. Namely, if $\sigma(\Box A, 1)$ occurs on a branch, we want to make sure that $\sigma'(A, 1)$ has been introduced for each prefix σ' accessible from σ which is used on the branch. As time goes on, there may be more and more such prefixes. This means that we want to work with $\sigma(\Box A, 1)$ many times. In our systematic procedure we proceed as follows: Whenever a branch θ_1 is obtained from a branch θ by adding a formula $\sigma' Z$, where σ' is an unrestricted prefix on θ, we apply the tableau rule to each formula of the form $\sigma(\Box A, 1)$ on θ_1 such that σ' is L_i-accessible from σ, i.e. we add $\sigma'(A, 1)$ to the branch θ_1 (the actions just described are formulated as the instruction $\mathcal{I}(\theta_1, L_i)$ defined below in the definition of the systematic procedure). Similarly we work with formulas of the form $\sigma(\Box A, 3)$ in case of a logic L_2.

Let (A, λ) be a valued formula. The systematic L_i-procedure for (A, λ) has different stages and is defined as follows.

Stage 1. Begin by placing $1(A, \lambda)$ at the root. This concludes stage 1. Suppose that n stages of the construction have been completed. If the tableau we have constructed is closed, then stop. Likewise, stop if every occurrence of a prefixed formula is finished. Otherwise go on to

Stage n + 1. Choose an occurrence of a prefixed formula that has not been finished as high up in the tree as possible on the leftmost branch, say it is $\sigma(B, \mu)$. If B is atomic, simply declare the occurrence *finished*. This ends stage $n + 1$. Otherwise we extend the tableau as follows: for each open branch θ through the occurrence of $\sigma(B, \mu)$:

(1) let B be of the form $F(A_1, \dots, A_m)$. For a logic L_i, $i \in \{2, 3\}$, let the tableau rule for $\sigma(B, \mu)$ have k consequences (see Definition 3.4 above). Then split the end of θ into k branches and add these consequences to the ends of these branches, respectively;

(2) let (B, μ) be of the form $(\Box A, 1)$. For a logic L_i, $i \in \{2, 3\}$, let σ_j, $1 \leq j \leq n$, be all the prefixes which have been used on θ and are L_i-accessible from σ. Then extend θ by adding nodes $\sigma_j(A, 1), 1 \leq j \leq n$;

(3) let (B, μ) be of the form $(\Box A, 2)$:

(a) for a logic L_2 let k be the smallest integer such that σ, k is unrestricted on θ. Then add $\sigma, k(A, 2)$ to the end of θ.
Let θ_1 be an open branch obtained from θ as just described. Then perform the instruction $\mathcal{I}(\theta_1, L_2)$ defined as follows: let $\sigma_i(\Box A_i, 1)$, $1 \leq i \leq n$, be all the formulas of the form $\sigma'(\Box A, 1)$ on θ_1 such that σ, k is L_2-accessible from σ_i, $1 \leq i \leq n$. Then extend θ_1 by adding nodes $\sigma, k(A_i, 1), 1 \leq i \leq n$. This is the end of the instruction $\mathcal{I}(\theta_1, L_2)$.
Let θ_2 be an open branch obtained from θ_1 after performing the instruction $\mathcal{I}(\theta_1, L_2)$. Then perform the instruction $\mathcal{I}_1(\theta_2, L_2)$ defined as follows: let $\sigma_i(\Box A_i, 3)$, $1 \leq i \leq m$, be all the formulas of the form $\sigma'(\Box A, 3)$ on θ_2 such that σ, k is L_2-accessible from σ_i, $1 \leq i \leq m$. Then extend θ_2, applying the rule R3 to formulas $\sigma_i(\Box A_i, 3)$, $1 \leq i \leq m$, step by step as follows: step 0: put $\theta^0 = \theta_2$; step i, $1 \leq i \leq m$: let θ^{i-1} be an open branch obtained by step $i - 1$; then split the end of θ^{i-1} and add $\sigma, k(A_i, 1)$ to the end of one fork and add $\sigma, k(A_i, 3)$ to the end of the other. This is the end of the instruction $\mathcal{I}_1(\theta_2, L_2)$.

(b) for a logic L_3 the branch θ which contains $(\Box A, 2)$ is closed (see the definition of the closed branch). So this case is excluded.

(4) let (B, μ) be of the form $(\Box A, 3)$:

(a) for a logic L_2 let k be the smallest integer such that σ, k is unrestricted on θ. Then add $\sigma, k(A, 3)$ to the end of θ.

Let θ_1 be an open branch obtained from θ as just described. Then perform the instruction $\mathcal{I}(\theta_1, L_2)$ defined above.

Let θ_2 be an open branch obtained from θ_1 after performing the instruction $\mathcal{I}(\theta_1, L_2)$. Then perform the instruction $\mathcal{I}_1(\theta_2, L_2)$ defined above.

Let θ_3 be an open branch obtained from θ_2 after performing the instruction $\mathcal{I}_1(\theta_2, L_2)$. Let σ_i, $1 \leq i \leq m$, be all the prefixes which have been used on θ_3 and are L_2-accessible from σ. Then extend θ_3, repeating the application of the rule R3 to the formula $\sigma(\Box A, 3)$, step by step as follows: step 0: put $\theta^0 = \theta_3$; step i, $1 \leq i \leq m$: let θ^{i-1} be an open branch obtained by step $i-1$; then split the end of θ^{i-1} and add $\sigma_i(A, 1)$ to the end of one fork and add $\sigma_i(A, 3)$ to the end of the other.

(b) for a logic L_3 let k be the smallest integer such that σ, k is unrestricted on θ. Then split the end of the branch θ and add $\sigma, k(A, 2)$ to the end of one fork and $\sigma, k(A, 3)$ to the end of the other. Let θ_1 be an open branch obtained from θ as just described. Then perform the instructions $\mathcal{I}(\theta_1, L_3)$ defined above.

Having done this for each open branch θ through the particular occurrence of $\sigma(B, \mu)$ being considered, declare that the occurrence of $\sigma(B, \mu)$ is *finished*. This completes stage $n + 1$.

If the construction of a tableau never terminates, then an infinite branch must always be involved as follows from famous König's lemma.

Definition 4.1. *A tree is* finitely generated *if each node has only a finite number of immediate successors.*

Lemma 4.2. (König's lemma). *An infinite finitely generated tree has at least one infinite branch.*

Proof. See [1]. □

Let L_i be a logic we are considering, i.e. $i \in \{2, 3\}$, $L \in \{K, K4, T, B, S4, S5\}$.

Definition 4.3. *Let S be a set of prefixed formulas. We say that S is L_i-downward saturated if:*

(1) *for no atomic formula A, prefix σ, and truth values $\lambda, \mu, \lambda \neq \mu$ do we have $\sigma(A, \lambda) \in S$ and $\sigma(A, \mu) \in S$;*

(2) *for each formula $\sigma(A, \lambda)$, such that there does not exist any defined tableau rule for L_i with a premise $\sigma(A, \lambda)$, we have $\sigma(A, \lambda) \notin S$;*

(3) *if $\sigma(F(A_1, \ldots, A_m), \lambda) \in S$ and there exists a defined tableau rule with a premise $\sigma(F(A_1, \ldots, A_m), \lambda)$, then some consequence of that rule belongs to S;*

(4) *if* $\sigma(\Box A, 1) \in S$, *then, for each prefix* σ' *that occurs in* S *and is* L_i*-accessible from* $\sigma, \sigma'(A,1) \in S$;

(5) *for a logic* L_2: *if* $\sigma(\Box A, 2) \in S$, *then, for some* σ' *that is* L_2*-accessible from* $\sigma, \sigma'(A,2) \in S$;

(6) *for a logic* L_2 : *if* $\sigma(\Box A, 3) \in S$, *then*

(a) *for each prefix* σ' *that occurs in* S *and is* L_2*-accessible from* σ, $\sigma'(A,1) \in S$ *or* $\sigma'(A,3) \in S$; *and*

(b) *for some prefix* σ' *that is* L_2*-accessible from* $\sigma, \sigma'(A,3) \in S$;

(7) *for a logic* L_3: *if* $\sigma(\Box A, 3) \in S$ *and* A *is of the form* $\Box B$, *then, for some prefix* σ' *that is* L_3*-accessible from* $\sigma, \sigma'(A,3) \in S$; *if* $\sigma(\Box A, 3) \in S$ *and* A *is not of the form* $\Box B$, *then, for some prefix* σ' *that is* L_3*-accessible from* $\sigma, \sigma'(A,2) \in S$ *or* $\sigma'(A,3) \in S$.

Lemma 4.4. *If* S *is* L_i*-downward saturated, then* S *is* L_i*-satisfiable* .

Proof. Suppose S is L_i-downward saturated. Construct a model as follows.

Let W be the set of prefixes that occurs in S. For $\sigma, \sigma' \in W$, let $\sigma R \sigma'$ provided the prefix σ' is L_i-accessible from σ.

Define a valuation v as follows: for $\sigma \in W$ and atomic A

$$v(A,\sigma) = \begin{cases} \lambda, & \text{if} \quad \sigma(A,\lambda) \in S, \\ 1, & \text{if} \quad \sigma(A,\mu) \notin S \text{ for each } \mu \in T. \end{cases}$$

This determines a frame (W, R, v) and a structure (W, R, v^i). It is easy to check that the relation R satisfies the conditions for the logic L_i.

Let $\sigma(B,\lambda)$ be a prefixed valued formula. By induction on the complexity of B it can be verifed:

$$\text{if} \quad \sigma(B,\lambda) \in S, \quad \text{then} \quad v^i(B,\sigma) = \lambda.$$

Finally, S is obviously L_i-satisfiable in the L_i-model just constructed under the L_i-interpretation I that is simply the identity map on prefixes of S. □

Let L_i be the logic we are considering, i.e. $i \in \{2,3\}$, $L \in \{K, K4, T, B, S4, S5\}$.

Theorem 4.5. *If a valued formula* (A,λ) *is* L_i*-unsatisfiable, then a systematic* L_i*-procedure for the formula* (A,λ) *produces a closed* L_i*-tableau.*

Proof. We show the contrapositive. Suppose that there exists a systematic L_i-procedure for (A,λ) which produces a L_i-tableau $\mathcal{T}$ which is not closed. If the procedure ever terminates, there is an open branch in $\mathcal{T}$, say θ. Likewise, if the procedure does not terminate, then, by König's Lemma 4.2, there still is an open branch, say θ, again. It is easy to check that the set of formulas on the branch θ will be a L_i-downward saturated set. Then, by Lemma 4.4, this set of formulas is L_i-satisfiable. But $1(A, \lambda)$ is on θ, hence in the L_i-model with the valuation v^i, defined in the proof of Lemma 4.4, $v^i(A, 1) = \lambda$, i.e. the formula (A, λ) is L_i-satisfiable. Contradiction. □

As we mentioned above, a valued formula (A, λ) is L_i-valid iff both formulas (A, λ_1), (A, λ_2), where $\lambda_1, \lambda_2 \in T - \{\lambda\}$, are L_i-unsatisfiable. This yields the following

Corollary 4.6 (completeness). *If a valued formula (A, λ) is L_i-valid, then there exist a closed L_i-tableau for the formula (A, λ_1) and a closed L_i-tableau for the formula (A, λ_2), where $\lambda_1, \lambda_2 \in T - \{\lambda\}$.*

Corollary 4.7. *If there exists a closed L_i-tableau for the formula (A, λ), then a systematic L_i-procedure for (A, λ) produces a closed tableau.*

From these corollaries we get the following algorithm to determine if a valued formula (A, λ) is L_i-valid: carry on systematic L_i-procedures for formulas (A, λ_1), (A, λ_2), where $\lambda_1, \lambda_2 \in T - \{\lambda\}$.

5. CONCLUDING REMARKS

In this paper we presented prefixed tableau systems for some three-valued propositional modal logics. We proved soundness and completeness theorems for these systems. The completeness proof involves the argument of a systematic procedure to construct tableaus. As shown in Section 4, this procedure provides us with the algorithm to determine if a valued formula is valid.

As in case of two-valued modal logics (see [1]), the notions of logical consequence for valued formulas can be introduced and soundness and completeness theorems for them can be proved.

REFERENCES

1. Fitting, M. *Proof Methods for Modal and Intuitionistic Logics*. D. Reidel Publishing Company, Dordrecht, 1983.
2. Takano, M. Cut-free systems for three-valued modal logics. *Notre Dame J. Form. Log.*, 1992, **33**, 3, 359–368.
3. Carnielli, W. A. Systematization of finite many-valued logics through the method of tableaux. *J. Symb. Log.*, 1987, **52**, 2, 473–493.

PREFIKSIGA TABELID KOLMEVALENTSETE PROPOSITSIONAALSETE MODAALLOOGIKATE TARVIS

Jūratė SAKALAUSKAITĖ

On esitatud mõningatele kolmevalentsetele propositsionaalsetele modaalloogikatele sobivad prefiksiga tabelite süsteemid. Prefiksiga tabelite süsteemide mittevasturääkivuse ja täielikkuse teoreem on tõestatud Kripke tüüpi semantikate suhtes.