A New Space Bound for the Modal Logics K4, KD4 and S4 *

Linh Anh Nguyen

Institute of Informatics, Warsaw University, ul. Banacha 2, 02-097 Warsaw
nguyen@melkor.mimuw.edu.pl

Abstract. We propose so called clausal tableau systems for the common modal logics **K4**, **KD4** and **S4**. Basing on these systems, we give more efficient decision procedures than those hitherto known for the considered logics. In particular space requirements for our logics are reduced from the previously established bound $O(n^2. \log n)$ to $O(n. \log n)$.

1 Introduction

It is well known that complexity of provability for the modal logics **K4**, **KD4** and **S4** is PSPACE-complete [5]. Recently, some authors have analyzed space requirements for modal logics [4, 10, 1, 8]. In [4] Hudelmaier translates formulae to clausal form and proposes a contraction-free sequent calculus, which is defined only for clauses and has a decreasing measure for rules, for **S4**. Using the measure he has shown that provability for **S4** is decidable in $O(n^2. \log n)$-space. Basing on labeled sequent systems, Viganò in [10] and Basin, Matthews and Viganò in [1] have given decision procedures for the logics **K4** and **KD4** using $O(n^2. \log n)$-space.

In this paper we propose so called clausal tableau systems for the common modal logics **K4**, **KD4** and **S4**. Following Hudelmaier, our systems are defined only for clauses. This simplifies proofs of completeness and gives a good method to estimate space bounds for modal logics. We give algorithms that for a **L**-satisfiable set X of clauses, where **L** is **K4** or **KD4** or **S4**, construct a **L**-model for X that has a *frame diameter* of order $O(n)$. Analyzing the algorithms allows us to establish a lower space bound, $O(n. \log n)$, for the considered logics.

This work is based on the work of Hudelmaier [4]. What makes our space bounds lower than the one of Hudelmaier is that we deal with constructing models and with saturation. The idea is that if we consider the saturating operation to be atomic then the search tree will have a depth of order $O(n)$ instead of $O(n^2)$. Due to the lack of space, proofs have been omitted or considerably shortened; see [8] for details.

* In M. Kutylowski, L. Pacholski, T. Wierzbicki (Eds.): Mathematical Foundations of Computer Science 1999, 24th International Symposium, MFCS'99, Szklarska Poreba, Poland, September 6-10, 1999, Proceedings. LNCS 1672, pages 321–331, Springer, 1999.

2 Preliminaries

2.1 Syntax and Semantics Definition for Modal Logics

A modal formula, hereafter simply called a *formula*, is any sequence of these symbols obtained from the following rules: any primitive proposition p_i is a formula, and if ϕ and ψ are formulae then so are $(\neg\phi)$, $(\phi \vee \psi)$, and $(\Box\phi)$.

We use letters like p and q to denote primitive propositions. We call formulae of the forms p or $\neg p$ *classical literals* and use letters like a, b, c to denote them. We call formulae of the forms a, $\Box a$, or $\neg\Box a$ *atoms* and use letters like A, B, C to denote them. A *simple clause* is an atom or a disjunction of atoms. We write $[A_1, \ldots, A_k]$ to denote the simple clause $A_1 \vee \ldots \vee A_k$. If ϕ is a simple clause then $\Box^s\phi$, where $s \geq 1$, is a *clause*. We use Greek letters like ϕ, ψ, ζ to denote formulae and clauses.

A *Kripke frame* is a triple $\langle W, \tau, R\rangle$, where W is a nonempty set (of possible worlds), $\tau \in W$ is the actual world, and R is a binary relation on W. If $(w, w^{'}) \in R$ then we say that the world $w^{'}$ is reachable from the world w. A *Kripke model* (resp. *model graph*) is a tuple $\langle W, \tau, R, h\rangle$ (resp. $\langle W, \tau, R, H\rangle$), where $\langle W, \tau, R\rangle$ is a Kripke frame, and h (resp. H) is a mapping from worlds to sets of primitive propositions (resp. formulae); that is, $h(w)$ (resp. $H(w)$) is the set of primitive propositions (resp. formulae) which are "true" at the world w. We sometimes treat model graphs as models with h being H restricted to the set of primitive propositions.

A world w in a model graph M is said to be *inconsistent* if there is a primitive proposition p such that both p and $\neg p$ belong to $H(w)$. A model graph is *consistent* if it contains no inconsistent world.

Given some Kripke model $M = \langle W, \tau, R, h\rangle$, and some $w \in W$, we write $M, w \models p$ iff $p \in h(w)$, and say that p is true at w in M. This satisfaction relation $\models$ is then extended to more complex formulae as follows:

$M, w \models p$	iff	$p \in h(w)$;
$M, w \models \neg\phi$	iff	$M, w \not\models \phi$;
$M, w \models \phi \vee \psi$	iff	$M, w \models \phi$ or $M, w \models \psi$;
$M, w \models \Box\phi$	iff	for all $v \in W$ such that $R(w, v)$, $M, v \models \phi$.

We say that M *satisfies* ϕ *at* w iff $M, w \models \phi$. We say that M *satisfies* ϕ, or ϕ *is satisfied in* M, iff $M, \tau \models \phi$.

In this paper we will consider the common modal logics **K4**, **KD4** and **S4**. These logics require the accessibility relation to be transitive. Additionally, **S4** requires the accessibility relation to be reflexive, and **KD4** requires the accessibility relation to satisfy the formula $\forall x\ \exists y\ R(x, y)$. For **L** being one of the considered logics, we call these restrictions ***L**-frame restrictions*.

We call a model M a ***L**-model* if the accessibility relation of M satisfies all **L**-frame restrictions. We say that ϕ is **L**-satisfiable if there exists a **L**-model of ϕ.

For **L** being **K4** or **S4**, and for a binary relation $R^{'}$, we write $Ext_L(R^{'})$ to denote the least extension of $R^{'}$ that satisfies all **L**-frame restrictions. It is clear that this operator is well defined.

2.2 Modal Clauses

We write $\phi_1; \phi_2; \ldots; \phi_k$ to denote the set $\{\phi_1, \phi_2, \ldots, \phi_k\}$. From now on, we use letters like X, Y, Z to denote sets of formulae or clauses. We write $X; Y$ to denote $X \bigcup Y$. For $X = \{\phi_1, \phi_2, \ldots, \phi_k\}$, we write $\Box^s X$ to denote $\Box^s \phi_1; \Box^s \phi_2; \ldots; \Box^s \phi_k$.

We define modal-depth of a formula, denoted by *mdepth*, as follows: $mdepth(a) = 0$, $mdepth(\Box\phi) = mdepth(\phi) + 1$, $mdepth(\neg\phi) = mdepth(\phi)$, $mdepth(\phi \vee \psi) = mdepth(\phi; \psi) = max(mdepth(\phi), mdepth(\psi))$. For a clause $\phi = \Box^s \psi$, where ψ is a simple clause, we define *restrictive* length of ϕ to be the length of the clause $\Box\psi$, and denote it by $rlength(\phi)$. We define restrictive length of a set of clauses to be the sum of restrictive lengths of its elements. Restrictive length can be understood as length in the common sense if sequences of $\Box^s$ are considered to be primitive connectives.

We call two sets of formulae X and Y *equisatisfiable* in a logic **L** iff (X is **L**-satisfiable iff Y is **L**-satisfiable).

Lemma 1. *Let p be a primitive proposition which only occurs at the indicated positions. Then the following pairs of sets of formulae are equisatisfiable in any normal modal logic:*

$$\begin{array}{ll}
X; \Box^s[\phi, \neg\neg\psi] & \text{and } X; \Box^s[\phi, \psi] \\
X; \Box^s[\phi, \psi \vee \zeta] & \text{and } X; \Box^s[\phi, \psi, \zeta] \\
X; \Box^s[\phi, \neg(\psi \vee \zeta)] & \text{and } X; \Box^s[\phi, \neg p]; \Box^s[p, \neg\psi]; \Box^s[p, \neg\zeta] \\
X; \Box^s[\phi, \Box\psi] & \text{and } X; \Box^s[\phi, \Box p]; \Box^{s+1}[p, \neg\psi] \\
X; \Box^s[\phi, \neg\Box\psi] & \text{and } X; \Box^s[\phi, \neg\Box p]; \Box^{s+1}[p, \neg\psi] \\
X; \Box^s[\phi, \psi] & \text{and } X; \Box^s[\phi, p]; \Box^s[\neg p, \psi]
\end{array}$$

This lemma is well known (cf. [6]) and using it we can translate any formula ϕ to a set X of clauses such that:

- X is a set of clauses of the form $\Box^s[a_1, \ldots, a_k]$, where $s \geq 0$, $k \geq 1$, or $\Box^s[a, B]$, where $s \geq 0$ and B is an atom of the form $\Box b$ or $\neg\Box b$;
- ϕ and X are equisatisfiable in any normal modal logic;
- the modal-depth of X is equal to the modal-depth of ϕ and the restrictive length of X is linearly bounded by the length of ϕ (note that X can have quadratic length).

During translation we treat sequences of $\Box^s$ as primitive connectives, which allows us to do the task and encode X in $O(n. \log n)$-space.

2.3 Syntax of Clausal Modal Tableau Systems

Our formulation of tableau systems is based on the work of Hintikka [3], Rautenberg [9] and Goré [2], and is defined only for sets of clauses. Given a formula, we can translate it to an equisatisfiable set of clauses to be usable by our systems.

A *tableau rule* R consists of a numerator N above the line and a (finite) list of denominators $D_1, D_2, \ldots, D_k$ (below the line) separated by vertical bars.

$$\frac{N}{D_1 \mid D_2 \mid \ldots \mid D_k}$$

The numerator is a finite set of clauses and so is each denominator. As we shall see later, each rule is read downwards as "if the numerator is **L**-satisfiable, then so is one of the denominators".

A *tableau system* (or *calculus*) $\mathcal{C}\mathbf{L}$ is a finite set of tableau rules. A $\mathcal{C}\boldsymbol{L}$*-tableau* is a tree with nodes carrying sets of clauses such that if x is a node carrying X and y_1, y_2, $\ldots y_k$ are all child nodes of x that carrying Y_1, Y_2, $\ldots Y_k$, then there exists a $\mathcal{C}\mathbf{L}$-rule R such that R has k denominators and X is an instance of the numerator of R and Y_i, $1 \leq i \leq k$, is the corresponding instance of the denominator number i of R.

A branch in a tableau is *closed* if it ends with $\perp$. A tableau is *closed* if all of its branches are closed. A tableau is *open* if it is not closed. A set X of clauses is said to be $\mathcal{C}\mathbf{L}$-*consistent* if every $\mathcal{C}\mathbf{L}$-tableau for X is open. If there is a closed $\mathcal{C}\mathbf{L}$-tableau for X then we say that X is $\mathcal{C}\mathbf{L}$-*inconsistent*.

A tableau system $\mathcal{C}\mathbf{L}$ is said to be *sound* if for any set X of clauses, if X is **L**-satisfiable then X is $\mathcal{C}\mathbf{L}$-consistent. A tableau system $\mathcal{C}\mathbf{L}$ is said to be *complete* if for any set X of clauses, if X is $\mathcal{C}\mathbf{L}$-consistent then X is **L**-satisfiable.

3 Clausal Tableau Systems for K4, KD4 and S4

Tables 1 and 2 represent clausal tableau rules and calculi for the modal logics **K4**, **KD4** and **S4**. The connective $\boxdot$ has the following semantics: $M, w \vDash \boxdot\phi$ iff $M, w \vDash (\Box\phi; \phi)$; i.e. $\boxdot\phi$ is a shortened form of $\Box\phi; \phi$. When dealing with the logic **S4** we assume that the language does not contain the connective $\boxdot$. The calculi $\mathcal{C}\mathbf{K4}$ and $\mathcal{C}\mathbf{KD4}$ are built in a similar way as $\mathcal{C}\mathbf{S4}$; the rules $(K4^{'})$, (K_r), $(K4_a)$, $(K4_b)$ and $(K4_c)$ are similar respectively to $(K4)$, (T_r), $(S4_a)$, $(S4_b)$ and $(S4_c)$. Note that in **S4** we have $\Box^s\phi \equiv \Box\phi$. The proofs of soundness of the calculi are straightforward.

3.1 Completeness of $\mathcal{C}$S4

Definition 1. *Let X be a $\mathcal{C}\boldsymbol{S4}$-consistent set of clauses. Let $Y = X$ and repeat the following steps until Y does not change:*

- *If Y is an instance of the numerator of the rule $(\vee)$ then one of the corresponding instances of denominators of $(\vee)$, lets say Z, must be $\mathcal{C}\boldsymbol{S4}$-consistent. Set $Y = Z$.*
- *Let R be $(S4_a)$ or $(S4_b)$. If Y is an instance of the numerator of R and the corresponding instance Z of the first denominator of R is $\mathcal{C}\boldsymbol{S4}$-consistent, then set $Y = Z$.*

It is easily seen that this process always terminates. We call Y a first kind $\mathcal{C}$**S4**-saturation *of X.*

Definition 2. *Let X be a $\mathcal{C}\boldsymbol{S4}$-consistent set of clauses. Let $Y = X$ and repeat the following steps until Y does not change:*

$$(\vee)\ \frac{X; [A_1, \ldots, A_k]}{X; A_1 \mid \ldots \mid X; A_k} \qquad\qquad (\bot)\ \frac{X; a; \neg a}{\bot}$$

$$(T_r)\ \frac{X; \Box^s \phi}{X; \Box^s \phi; \phi} \qquad \text{where } \phi = [a_1, \ldots, a_k]$$

$$(K4)\ \frac{X; \Box Y; \neg \Box a}{\Box Y; \neg a} \qquad\qquad (S4_a)\ \frac{X; \Box^s [a, \Box b]}{X; \Box b \mid X; \Box^s [a, \Box b]; a}$$

$$(S4_b)\ \frac{X; \Box Y; \Box^s [a, \neg \Box b]}{X; \Box Y; \Box a \mid \Box Y; \Box^s [a, \neg \Box b]; \neg b} \qquad (S4_c)\ \frac{X; \Box Y; \Box^s \neg \Box a}{\Box Y; \Box^s \neg \Box a; \neg a}$$

$$(K4^{'})\ \frac{X; \Box Y; \Box^2 Z; \neg \Box a}{\boxdot Y; \Box Z; \neg a} \qquad\qquad (KD4)\ \frac{X; \Box Y; \Box^2 Z; \boxdot U}{\boxdot Y; \Box Z; \boxdot U}$$

where Y is a set of simple clauses

$$(K_r)\ \frac{X; \boxdot \phi}{X; \boxdot \phi; \phi} \qquad \text{where } \phi = [a_1, \ldots, a_k]$$

$$(K4_a)\ \frac{X; \boxdot [a, \Box b]}{X; \Box b \mid X; \boxdot [a, \Box b]; a}$$

$$(K4_b)\ \frac{X; \boxdot Y; \Box Z; \Box^2 U; \boxdot [a, \neg \Box b]}{X; \boxdot Y; \Box Z; \Box^2 U; \boxdot a \mid \boxdot Y; \boxdot Z; \Box U; \boxdot [a, \neg \Box b]; \neg b}$$

$$(K4_c)\ \frac{X; \boxdot Y; \Box Z; \Box^2 U; \boxdot \neg \Box a}{\boxdot Y; \boxdot Z; \Box U; \boxdot \neg \Box a; \neg a}$$

where Z is a set of simple clauses

Table 1. Clausal tableau rules for **K4**, **KD4** and **S4**

$\mathcal{C}$**L**	Rules
$\mathcal{C}$**K4**	$(\vee), (\bot), (K_r), (K4_a), (K4_b), (K4_c), (K4^{'})$
$\mathcal{C}$**KD4**	$(\vee), (\bot), (K_r), (K4_a), (K4_b), (K4_c), (K4^{'}), (KD4)$
$\mathcal{C}$**S4**	$(\vee), (\bot), (T_r), (S4_a), (S4_b), (S4_c), (K4)$

Table 2. Clausal tableau calculi for **K4**, **KD4** and **S4**

– *Let R be $(\vee)$ or (T_r). If Y is an instance of the numerator of R then one of the corresponding instances of denominators of R, lets say Z, must be $\mathcal{C}\boldsymbol{S4}$-consistent. Set $Y = Z$.*
– *If Y is an instance of the numerator of the rule $(S4_a)$ then set Y to be the corresponding instance of the second denominator of $(S4_a)$.*

It is easily seen that this process always terminates. We call Y a second kind $\mathcal{C}\mathbf{S4}$-saturation *of X.*

In the following algorithm, and in Algorithm 3 as well, we assume that we have oracles to compute the steps 2b and 4.

Algorithm 1 *Let X be a $\mathcal{C}\boldsymbol{S4}$-consistent set of clauses. We construct a consistent $\boldsymbol{S4}$-model graph $M = \langle W, \tau, R, H \rangle$ that satisfies X as follows:*

1. *Set $W = \{\tau\}$, $H_0(\tau) = X$, $R_0 = R_1 = \emptyset$, and mark τ as unsolved.*
2. *(a) Take an unsolved world w from W.*
   (b) Let $H_1(w)$ be a first kind $\mathcal{C}\boldsymbol{S4}$-saturation of $H_0(w)$.
   (c) For every atom $\neg\Box a$ from $H_1(w)$:
   – *Create a new world w_a, add it to W and mark it as unsolved.*
   – *Set $R_0 = R_0 \bigcup \{(w, w_a)\}$.*
   – *Set $H_0(w_a) = \{\neg a\} \bigcup \{\Box Y | \Box Y \in H_1(w)\}$.*
   (d) For every clause $\phi \in H_1(w)$ such that $\phi = \Box^s \psi$, where $s \geq 1$ and $(\psi = [a, \neg\Box b]$ or $\psi = \neg\Box b)$:
   – *Let $Y = \{\neg b\} \bigcup \{\Box Z \mid \Box Z \in H_1(w)\}$.*
   – *If there exists a world $w_\phi \in W$ such that $w_\phi = w$ or $R_0^*(w_\phi, w)$, and $Y \subseteq H_1(w_\phi)$, then set $R_1 = R_1 \bigcup \{(w, w_\phi)\}$;*
   – *else*
     - *Create a new world w_ϕ, add it to W and mark it as unsolved.*
     - *Set $R_0 = R_0 \bigcup \{(w, w_\phi)\}$, and $H_0(w_\phi) = Y$.*
   (e) Mark w as resolved.
3. *While there are unsolved worlds, repeat the step 2.*
4. *For every $w \in W$, let $H(w)$ be a second kind $\mathcal{C}\boldsymbol{S4}$-saturation of $H_1(w)$.*
5. *Set $R = Ext_{S4}(R_0 \bigcup R_1)$.*

In the step 2d, we use R_0^* to denote the transitive closure of R_0. The step 2c corresponds to the rule $(K4)$, the step 2d to the rules $(S4_b)$ and $(S4_c)$. It is easily seen that the number of possible contents of nodes is finitely bounded. It follows that this algorithm always terminates.

Lemma 2. *Let M be the model graph constructed by Algorithm 1. Then M is a consistent $\boldsymbol{S4}$-model graph and for any $w \in W$ and $\phi \in H(w)$, $M, w \vDash \phi$.*

Proof. It is clear that M is consistent and R satisfies all **S4**-frame restrictions. Suppose that $\phi \in H(w)$, we show that $M, w \vDash \phi$.

It is easily seen that ϕ cannot be of the form $[A_1, \ldots, A_k]$ with $k > 1$.

Case $\phi = \neg\Box a$: We have $R_0(w, w_a) \wedge (\neg a) \in H_0(w_a)$. Hence $M, w \vDash \phi$.

Case $\phi = \Box^s[a_1, \ldots, a_k]$, where $s \geq 1$: We claim that for any $u \in W$ such that $\phi \in H_1(u)$ the following assertions hold:

$$\begin{array}{l} M, u \vDash [a_1, \ldots, a_k] \\ \forall v\ R_0(u, v) \to \phi \in H_1(v) \\ \forall v\ R_1(u, v) \to \phi \in H_1(v) \\ \forall v\ R(u, v) \to (M, v \vDash [a_1, \ldots, a_k]) \end{array}$$

It follows that $\forall u \in W\ \phi \in H_1(u) \to (M, u \vDash \phi)$. Since $\phi \in H(w)$, it is easily seen that $\phi \in H_1(w)$. Therefore $M, w \vDash \phi$.

Case $\phi = \Box^s[a, \Box b]$, where $s \geq 1$: We claim that for any $u \in W$ such that $\phi \in H_0(u)$ the following assertions hold:

$$\begin{array}{l} \Box b \in H(u) \vee a \in H(u) \\ \forall v\ R_0(u, v) \to \phi \in H_0(v) \vee \Box b \in H(v) \\ \forall v\ R_1(u, v) \to \phi \in H_0(v) \vee \Box b \in H(v) \end{array}$$

It follows that $\forall u \in W\ \phi \in H_0(u) \to (M, u \vDash \phi)$. Since $\phi \in H(w)$, it is easily seen that $\phi \in H_0(w)$. Therefore $M, w \vDash \phi$.

Case $\phi = \Box^s[a, \neg\Box b]$, where $s \geq 1$: We claim that for any $u \in W$ such that $\phi \in H_1(u)$ the following assertions hold:

$$\begin{array}{l} \exists v\ R(u, v) \wedge (\neg b) \in H(v) \\ \forall v\ R_0(u, v) \to \phi \in H_1(v) \vee \Box a \in H_1(v) \\ \forall v\ R_1(u, v) \to \phi \in H_1(v) \end{array}$$

It follows that $\forall u \in W\ \phi \in H_1(u) \to M, u \vDash \phi$. Since $\phi \in H(w)$, it is easily seen that $\phi \in H_1(w)$. Therefore $M, w \vDash \phi$.

Case $\phi = \Box^s \neg\Box a$, where $s \geq 1$, is similar to the preceding case.

Corollary 1. *Let X be a $\mathcal{C}$**S4**-consistent set of clauses. Let M be the model graph constructed by Algorithm 1 for the input X. Then M is a* ***S4**-model of X.*

Proof. By Lemma 2, M is a consistent **S4**-model graph and $M \vDash H(\tau)$. Since $H(\tau)$ is a second kind $\mathcal{C}$**S4**-saturation of $H_1(\tau)$, which is a first kind $\mathcal{C}$**S4**-saturation of X, we conclude that $M \vDash X$.

We arrive at

Theorem 2. *The calculus $\mathcal{C}$**S4** is sound and complete.*

3.2 Completeness of $\mathcal{C}$K4 and $\mathcal{C}$KD4

In this subsection we use **L** to denote **K4** or **KD4**, and we write $\mathcal{C}$**L** to denote the corresponding calculus. We define the first and the second kind of $\mathcal{C}$**L**-saturation in the same way as for $\mathcal{C}$**S4** as in Definitions 1 and 2, with $(S4_a)$, $(S4_b)$ and (T_r) replaced by $(K4_a)$, $(K4_b)$ and (K_r), respectively. We will use $CreateANewWorldFrom(w, x)$ to denote the following:

– Let x be a new world, add it to W and mark it as unsolved.
– Set $R_0 = R_0 \bigcup \{(w, x)\}$.

Algorithm 3 *Let X be a $\mathcal{C}\boldsymbol{L}$-consistent set of clauses not containing the connective $\boxdot$. We construct a consistent $\mathcal{C}\boldsymbol{L}$-model graph $M = \langle W, \tau, R, H\rangle$ that satisfies X as follows:*

1. *Set $W = \{\tau\}$, $H_0(\tau) = X$, $R_0 = R_1 = \emptyset$, and mark τ as unsolved.*
2. (a) *Take an unsolved world w from W.*
   (b) *Let $H_1(w)$ be a first kind $\mathcal{C}\boldsymbol{L}$-saturation of $H_0(w)$.*
   (c) *For every atom $\neg\Box a$ from $H_1(w)$:*
      – *CreateANewWorldFrom(w, w_a).*
      – *Set $H_0(w_a) = \{\neg a\} \bigcup \{\Box Z \mid \Box^2 Z \in H_1(w)\} \bigcup$*
        $\bigcup \{\boxdot Y \mid \Box Y \in H(w)$ and Y is a set of simple clauses$\}$
   (d) *If the connective $\Box$ occurs in $H_1(w)$ then:*
      For every clause $\phi \in H_1(w)$ such that $\phi = \boxdot\psi$, where $s \geq 1$ and ($\psi = [a, \neg\Box b]$ or $\psi = \neg\Box b$):
      – *CreateANewWorldFrom(w, w_ϕ).*
      – *Set $H_0(w_\phi) = \{\neg b\} \bigcup \{\Box U \mid \Box^2 U \in H_1(w)\} \bigcup$*
        $\{\boxdot Y \mid \boxdot Y \in H_1(w) \vee \Box Y \in H_1(w)$
        and Y is a set of simple clauses$\}$.
   (e) *If the connective $\Box$ does not occur in $H_1(w)$ then:*
      For every clause $\phi \in H_1(w)$ such that $\phi = \boxdot\psi$, where $s \geq 1$ and ($\psi = [a, \neg\Box b]$ or $\psi = \neg\Box b$):
      – *Let $Y = \{\neg b\} \bigcup \{\boxdot Z \mid \boxdot Z \in H_1(w)\}$.*
      – *If there exists a world $w_\phi \in W$ such that $w_\phi = w$ or $R_0^*(w_\phi, w)$, and $Y \subseteq H_1(w_\phi)$, then set $R_1 = R_1 \bigcup \{(w, w_\phi)\}$;*
      – *else*
        • *CreateANewWorldFrom(w, w_ϕ).*
        • *Set $H_0(w_\phi) = Y$.*
   (f) *If $\boldsymbol{L}$ is* ***KD4*** *and there is no u such that $R_0(w, u)$ or $R_1(w, u)$ then*
      – *If the connective $\Box$ occurs in $H_1(w)$ then*
        • *CreateANewWorldFrom(w, w').*
        • *Set $H_0(w') = \{\Box Z \mid \Box^2 Z \in H_1(w)\} \bigcup$*
          $\{\boxdot Y \mid \boxdot Y \in H_1(w) \vee \Box Y \in H_1(w)$
          and Y is a set of simple clauses$\}$.
      – *else set $R_0 = R_0 \bigcup \{(w, w)\}$.*
   (g) *Mark w as resolved.*
3. *While there are unsolved worlds, repeat the step 2.*
4. *For every $w \in W$, let $H(w)$ be a second kind $\mathcal{C}\boldsymbol{L}$-saturation of $H_1(w)$.*
5. *Set $R = Ext_{K4}(R_0 \bigcup R_1)$.*

The step 2c corresponds to the rule $(K4')$, the steps 2d and 2e to the rules $(K4_b)$ and $(K4_c)$, and the step 2f to the rule $(KD4)$. Let $l = rlength(X)$ and $s = mdepth(X)$. On any path of the tree R_0, there is at most one world created at the step 2c (with $w = \tau$), and at most l worlds created at the steps 2d or 2f, with depths greater than s. Moreover, the number of possible contents of nodes is finitely bounded. It follows that this algorithm always terminates.

Lemma 3. *Let X be a $\mathcal{C}\boldsymbol{L}$-consistent set of clauses not containing the connective $\boxdot$. Let M be the model graph constructed by Algorithm 3 for X. Then M is a $\boldsymbol{L}$-model of X.*

The proof of this lemma is similar to the proof of Lemma 2 and Corollary 1, so we omit it. We arrive at

Theorem 4. *The calculi $\mathcal{C}\boldsymbol{K4}$ and $\mathcal{C}\boldsymbol{KD4}$ are sound and complete.*

4 Space Bounds for the Logics

Lemma 4. *Let X be a $\mathcal{C}\boldsymbol{S4}$-consistent set of clauses, with $rlength(X) = n$. Let R_0 be the relation computed by Algorithm 1 for X. Then R_0 is a tree with a depth bounded by n.*

Proof. Suppose that the depth of R_0 is greater than n. It is easily seen that there exist two different worlds w, u such that: $R_0^*(\tau, w) \wedge R_0^*(w, u)$, $H_1(w) = \{\neg b\} \bigcup Y$, $H_1(u) = \{\neg b\} \bigcup Z$, for some b, Y, Z such that $Y \neq Z$ and Y and Z contain only clauses of the form $\Box^s \phi$, where $s \geq 1$. Since $Y \neq Z$, either the rule $(S4_a)$ or $(S4_b)$ must be applied on the path from w to u. Observe that if $\Box^s \phi \in H_1(w)$, where $s \geq 1$, then:

- If ϕ is of the form $[a_1, \ldots, a_k]$ then $\Box^s \phi \in H_1(u)$.
- If ϕ is of the form $[a, \Box b]$ then either $\Box^s \phi$ or $\Box b$ belongs to $H_1(u)$.
- If ϕ is of the form $[a, \neg\Box b]$ then either $\Box^s \phi$ or $\Box a$ belongs to $H_1(u)$.

This contradicts the fact that $H_1(w)$ is a first kind $\mathcal{C}$**S4**-saturation of $H_0(w)$.

Consider the nondeterministic algorithm obtained from Algorithm 1 by deleting the steps 4 and 5, ignoring computing W and R_1, and replacing the step 2b by:

"Nondeterministically choose a candidate for first kind $\mathcal{C}$**S4**-saturation of $H_0(w)$ and assign it to $H_1(w)$. If all candidates for second kind $\mathcal{C}$**S4**-saturation of $H_1(w)$ are inconsistent in the sense that they contain both p and $\neg p$ for some p, or if the depth of w in the tree R_0 is greater than n, then reject the computation."

It is easily seen that for any set X of clauses, X is **S4**-satisfiable iff the new algorithm has an unrejected computation for X. We can simulate the mentioned algorithm by a deterministic one, denoted by A_1, by backtracking. During computation of A_1 for X we only need to keep information about the current path from the root τ to the current world.

We will treat sequences of $\Box^s$ as primitive connectives. Let n be the restrictive length of X. For each world on the current path, we need $O(\log n)$-space to keep information about "and" branching from its parent to it, i.e. to keep the position either of the atom $\neg\Box a$, in the case of the step 2c, or of the clause ϕ, in the case of the step 2d, in the sequential display of X. For each formula ϕ of the form $\Box^s[a, \Box b]$ (resp. $\Box^s[a, \neg\Box b]$), where $s \geq 1$, we need $O(\log n)$-space to keep the depth at which it is replaced by $\Box b$ (resp. $\Box a$) as a result of applying the rule $(S4_a)$ (resp. $(S4_b)$), i.e. to keep information about "or" branching caused by the formula. Note that formulae of the form $[A_1, \ldots, A_k]$, where $k > 1$, can occur only in the root τ.

We do not keep the sets $H_0(w)$ and $H_1(w)$, except for w being the current node or the root node. For any node w on the current path, from the content of the root node and the information about and-or branching (on whole of the path) we can reconstruct the sets $H_0(w)$ and $H_1(w)$ using $O(n. \log n)$-space. For this, it suffices to show that for any worlds w and u on the current path such that $R_0(w, u)$, having $H_1(w)$ and the information about and-or branching we can construct $H_0(u)$ and $H_1(u)$ using $O(n. \log n)$-space. It is clear that having $H_1(w)$ and the information about "and" branching from w to u we can construct $H_0(u)$ using $O(n. \log n)$-space. Let k be the depth of u in the tree R_0. We have $1 \leq k \leq n$. The set $H_1(u)$ is obtained from $H_0(u)$ by replacing every clause of the form $\Box^s[a, \Box b]$ (resp. $\Box^s[a, \neg\Box b]$) for which there is information that it is replaced by $\Box b$ (resp. $\Box a$) at the depth k by $\Box b$ (resp. $\Box a$). It is clear that the task can be done in $O(n. \log n)$-space.

For the two special worlds, the root and the current, we can use $O(n. \log n)$-space to keep information about them, and to test consistency of candidates for second kind $\mathcal{C}$**S4**-saturation of the current world. Since every path has a depth bounded by n, we conclude that the algorithm A_1 can be computed in $O(n. \log n)$-space.

Theorem 5. *The logic **S4** is decidable in $O(n. \log n)$-space.*

Proof. Just recall that using $O(n. \log n)$-space we can translate any formula Φ to a **S4**-equisatisfiable set X of clauses such that the restrictive length of X is linearly bounded by the length of Φ.

Lemma 5. *Let **L** be **K4** or **KD4**. Let X be a $\mathcal{C}$**L**-consistent set of clauses, with rlength$(X) = l$ and mdepth$(X) = s$. Let R_0 be the relation computed by Algorithm 3 for X. Then R_0 is a tree with a depth bounded by $2l + s$.*

The proof of this lemma is similar to the proof of Lemma 4. Reasoning similarly as for $\mathcal{C}$**S4**, we arrive at

Theorem 6. *The logics **K4** and **KD4** are decidable in $O(n. \log n)$-space.*

5 Conclusion

We have presented clausal tableau systems for the modal logics **K4**, **KD4** and **S4**, and have shown that these logics are decidable in $O(n. \log n)$-space. This space bound is lower than those hitherto known. The method used in this paper is applicable for other modal logics, including **K**, **KD**, **T**, **KB**, **KDB** and **B**; see [7] for details.

Acknowledgement

The author wishes to express his thanks to prof. Andrzej Szałas and the anonymous reviewers for comments.

References

1. D. Basin, S. Matthews, and L. Viganò. *Complexity Bounds for Propositional Modal Logics.* Technical Report, Institut für Informatik, Albert-Ludwigs-Universität Freiburg, Germany. To appear. Available at http://www.informatik.uni-freiburg.de/˜luca, 1999.
2. R. Goré. Tableau methods for modal and temporal logics. In D'Agostino Gabbay Hähnle Posegga, editor, *Handbook of Tableau Methods*, pages 297–396. Kluwer Academic Publishers, 1999.
3. K.J.J. Hintikka. Form and content in quantification theory. *Acta Philosophica Fennica*, 8:3–55, 1955.
4. J. Hudelmaier. A contraction-free sequent calculus for s4. In *H. Wansing, editor, Proof Theory of Modal Logic*, pages 3–15. Kluwer, Dordrecht, 1996.
5. R. Ladner. The computational complexity of provability in systems of modal propositional logic. *SIAM Journal of Computing*, 6:467–480, 1977.
6. G.E. Mints. Gentzen-type systems and resolution rules. In *P.Martin-Löf, G. Mints (eds.): COLOG-88, LNCS 417*, pages 198–231. Springer, 1988.
7. L.A. Nguyen. *Clausal Tableau Systems and Space Bounds for Propositional Modal Logics.* Technical Report, Institute of Informatics, Warsaw University, 1999. To appear.
8. L.A. Nguyen. *Results on Modal Reasoning with Applications to Modal Deductive Databases.* PhD thesis, Warsaw University, forthcoming.
9. W. Rautenberg. Modal tableau calculi and interpolation. *Journal of Philosophical Logic*, 12:403–423, 1983.
10. L. Viganò. *A framework for non-classical logics.* PhD thesis, Universität des Saarlandes, Saabrücken, Germany, 1997.