Contents

Clausal Tableau Systems and Space Bounds for the Modal Logics K, D, T, KB, KDB and B

Linh Anh Nguyen
Institute of Informatics, Warsaw University
Banacha 2, 02-097, Warsaw, Poland
(nguyen@melkor.mimuw.edu.pl)

Abstract

We propose so called clausal tableau systems for the common modal logics K, KD, T, KB, KDB and B. There is a measure such that for each tableau rule of these systems the measure of all its denominators is smaller than the measure of its numerator. Basing on these systems, we give a decision procedure for the logics, which uses $O(n^2)$-space for the logics T, KB, KDB and B, and $O(n.\log n)$-space for the logics K and KD. We also show that the problem of checking satisfiability in T, KB, KDB, or B for formulae with finitely bounded modal-depth is decidable in $O(n.\log n)$-space. We are the first who explicitly establish space requirements for the logics KB, KDB and B.

1 Introduction

Recently, some authors have analyzed space requirements for modal logics. In [5] Hudelmaier translates formulae to clausal form and proposes contraction-free sequent calculi for the logics K, T and $S4$ that are defined only for sets of clauses and have decreasing measures for rules. Using the measures he has shown that provability for $S4$ is decidable in $O(n^2.\log n)$-space, for K and T in $O(n.\log n)$-space. Using labeled sequent systems, Basin, Matthews and Viganò have given decision procedures for the logics K, KD and T that use $O(n.\log n)$-space [1], and Viganò has given decision procedures for the logics $K4$, $KD4$ and $S4$ that use $O(n^2.\log n)$-space [11]. In [9] we develop so called clausal tableau systems for the logics $K4$, $KD4$ and $S4$, and reduce space requirements for these logics to $O(n.\log n)$.

In this paper we propose clausal tableau systems for the common modal logics K, KD, T, KB, KDB and B. Following Hudelmaier, our systems are defined only for sets of clauses. This simplifies proofs of completeness and gives a good method to estimate space bounds for modal logics. There is a measure such that for each tableau rule of our systems the measure of all its denominators is smaller than the measure of its numerator. This is just a nice property of our systems, which allows depth-first search without loop checking. To estimate space bounds for the modal logics we do not use the measure. Basing on our systems, we give a decision procedure for the logics, which uses $O(n^2)$-space for the logics T, KB, KDB and B, and $O(n.\log n)$-space for the logics K and KD. We also show that the problem of checking satisfiability in T, KB, KDB, or B for formulae with finitely bounded modal-depth is decidable in $O(n.\log n)$-space. We are the first who explicitly estimates space requirements for the logics KB, KDB and B.

Our formulation of tableau systems is based on the work of Goré [3]. Our method for proving completeness of tableau systems is slightly different to his. Instead of constructing a *saturated* L-model graph for a $\mathcal{CL}$-consistent set X of clauses we construct a L-model graph (that needs not to be saturated) and prove by induction that the model graph satisfies all its formulae at the corresponding worlds.

What makes our method good in estimating space bounds for modal logics is that we deal with constructing model graphs and with saturation. By saturating contents of nodes before generating their successors we shorten the depth of the search tree. In Section 5, we will discuss advantages of our methods.

2 Preliminaries

2.1 Syntax and Semantics Definition for Modal Logics

A modal formula, hereafter simply called a *formula*, is any sequence 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 p and q to denote primitive propositions, and Greek letters ϕ, ψ, ζ to denote formulae. We denote the set of primitive propositions by $\mathcal{P}$, and the set of formulae by $\mathcal{F}$.

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 called the accessibility relation. If $R(w, u)$ holds then we say that the world u is reachable from w.

A *Kripke model* is a tuple $\langle W, \tau, R, h\rangle$, where $\langle W, \tau, R\rangle$ is a Kripke frame, $h : W \to P(\mathcal{P})$, and $h(w)$ is the set of primitive propositions which are "true" at the world w.

A *model graph* is a tuple $\langle W, \tau, R, H\rangle$, where $\langle W, \tau, R\rangle$ is a Kripke frame, $H : W \to P(\mathcal{F} \bigcup \{\bot\})$, and $H(w)$ is the set of formulae which should be "true" at the world w. The symbol $\bot$ stands for "false". A world w in a model graph M is said to be *inconsistent* if it contains $\bot$. A model graph is *consistent* if it contains no inconsistent worlds. We sometimes treat model graphs as models with H being restricted to the set of primitive propositions.

Given some Kripke model $M = \langle W, \tau, R, h\rangle$, some world $w \in W$, the satisfaction relation $\models$ is defined as follows:

$$
\begin{array}{lll}
M, w \models p & \text{iff} & p \in h(w); \\
M, w \models \neg\phi & \text{iff} & M, w \not\models \phi; \\
M, w \models \phi \vee \psi & \text{iff} & M, w \models \phi \text{ or } M, w \models \psi; \\
M, w \models \Box\phi & \text{iff} & \text{for all } v \in W \text{ such that } R(w, v), M, v \models \phi.
\end{array}
$$

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 $M, \tau \models \phi$.

We define the length of a formula ϕ to be the number of connectives and primitive propositions in ϕ (counting occurrences). We define the modal-depth of a formula, denoted by *mdepth*, as follows:

$$
\begin{array}{l}
mdepth(p) = 0 \\
mdepth(\neg\phi) = mdepth(\phi) \\
mdepth(\phi \vee \psi) = max(mdepth(\phi), mdepth(\psi)) \\
mdepth(\Box\phi) = mdepth(\phi) + 1
\end{array}
$$

2.2 Modal Logic Correspondences

The simplest normal modal logic (called K) is axiomatized by the standard axioms for the classical propositional logic, the *modus ponens* inference rule, the *K-axiom* schema $\Box(\phi \to \psi) \to (\Box\phi \to \Box\psi)$,

Axiom	Schemata	First-Order Formula
D	$\Box\Phi \to \Diamond\Phi$	$\forall x\ \exists y\ R(x,y)$
T	$\Box\Phi \to \Phi$	$\forall x\ R(x,x)$
B	$\Phi \to \Box\Diamond\Phi$	$\forall x,y\ R(x,y) \to R(y,x)$

Table 1: Axioms and corresponding first-order conditions on R

Logic	Axioms	Frame Restriction
K	K	no restriction
KD	KD	serial
T	KT	reflexive
KB	KB	symmetric
KDB	KDB	serial and symmetric
B	KTB	reflexive and symmetric

Table 2: Modal logics and frame restriction

plus the additional *necessitation rule*

$$\frac{\vdash \phi}{\vdash \Box\phi}$$

It can be shown that a modal logic formula is provable in this axiomatization iff it is satisfied in every Kripke model (i.e. without any special R-properties) [7]. It is known that certain axiom schemata added to this axiomatization are mirrored by certain properties of the accessibility relation (see also [6], [2]).

Many of such correspondences are definable as formulae of first-order logic where the binary predicate $R(x,y)$ represents the accessibility relation, as shown in Table 1. Different modal logics are distinguished by their respective additional axiom schemata. We refer to properties of the accessibility relation of a modal logic L as *L-frame axioms* or *L-frame restrictions*. Some of the most popular modal logics together with their axiom schemata are listed in Table 2.

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.

Given two modal logics L and L', we say that L' is a *normal extension* of L, and write $L \leq L'$, if all L-frame restrictions are also L'-frame restrictions. By *normal modal logics* we call modal logics that are normal extensions of the logic K.

Let L be one of the modal logics listed in Table 2. For a binary relation R', we use $Ext_L(R')$ to denote the least extension of R' that satisfies all L-frame axioms, excluding the axiom D. This operator is well defined for such logics.

2.3 Modal Clauses

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 classical literals, or a formula of one of the forms $(a \vee \Box b)$, $(a \vee \neg\Box b)$. If ϕ is a simple clause then we call $\Box\phi$ a $\Box$-*simple clause*, and $\Box^s\phi$, where $s \geq 0$, a *clause*. The length of a clause is the length of the formula it stands for.

For a clause $\phi = \Box^s\psi$, where ψ is a simple clause, we define the *restrictive length* of ϕ to be the length of the clause $\Box\psi$, and denote it by $rlength(\phi)$. We define the 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. It plays an important role in estimating space bounds for the logics considered in this paper.

We write $[\phi_1, \ldots, \phi_k]$ to denote the disjunction $\phi_1 \vee \ldots \vee \phi_k$, and 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 the sum of X and Y, and write $\Box^s X$ to denote the set $\{\Box^s\phi \mid \phi \in X\}$. A set of formulae is sometimes considered as the conjunction of its formulae, in particular when we are talking about length, modal depth, or satisfiability.

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 (Mints) *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:*

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

This lemma is well known (cf. [8]) and using it we can translate any formula ϕ to a set X of clauses such that:

- 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).

Let n be the length of ϕ. X can be coded in $O(n.\log n)$-space; each sequence $\Box^s$ takes $O(\log n)$-space, each other connective $O(1)$-space, and each primitive proposition $O(\log n)$-space. It should be clear that the translation can be done using $O(n.\log n)$-space.

2.4 Syntax of Clausal Modal Tableau Systems

Our formulation of tableau systems is based on the work of Goré [3], 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, ..., 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". The terms "numerator" and "denominator" are borrowed from Goré [3]. The numerator of each tableau rule contains one or more distinguished clauses called the *principal clauses*.

A *tableau system* (or *calculus*) $\mathcal{C}L$ is a finite set of tableau rules. A $\mathcal{C}L$-*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 carry Y_1, Y_2, $\ldots Y_k$, then there exists a $\mathcal{C}L$-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.

Note that in our definition, tableaux can be infinite. We say that a tableau system $\mathcal{C}L$ has the *finite tableau property* if every $\mathcal{C}L$-tableau is finite.

Let X and Y be sets of clauses, and $\mathcal{R}$ be a set of tableau rules. We say that Y can be obtained from X by applying the rules from $\mathcal{R}$ if there exists a tableau for X that uses only rules from $\mathcal{R}$, and Y is its node.

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 $\mathcal{C}L$-*consistent* if every $\mathcal{C}L$-tableau for X is open. If there is a closed $\mathcal{C}L$-tableau for X then we say that X is $\mathcal{C}L$-*inconsistent*.

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

Let R be one of the rules of $\mathcal{C}L$. We say that R is sound wrt. L if for any instance R' of R, if the numerator of R' is L-satisfiable then so is one of the denominators of R'.

3 Clausal Tableau Systems for the Modal Logics *K*, *KD*, *T*, *KB*, *KDB* and *B*

Tables 3 and 4 represent clausal tableau rules and calculi for the modal logics K, KD, T, KB, KDB and B.

The connective $\blacksquare$ used in Table 3 has the same semantics as $\Box$, i.e. $M, w \vDash \blacksquare\phi$ iff $M, w \vDash \Box\phi$. We write $\blacksquare^s\phi$ to denote $\blacksquare\Box^{s-1}\phi$. Syntactically, clauses of the form $\blacksquare^s\phi$, where $s \geq 1$, are blocked from being principal clauses. The connective $\blacksquare$ has the same role as the connective $\bigcirc$ introduced by Hudelmaier in [5].

Following Goré [3], we categorize each rule as either a *static rule* or as a *transitional rule*. The intuition behind this sorting is that in the static rules, the numerator and denominator represent the same world (in the same model), whereas in the transitional rules, the numerator and denominator represent different worlds (in the same model).

For intuition of the rules, as an example, let us consider the rule (T). This is a static rule. If $\Box^s\phi$ is present at a world w then so are $\Box^{s-1}\phi$, $\ldots$, $\Box\phi$, ϕ. The rule takes advantage of a one-step treatment of iterated modality (i.e. $\Box^s$). The intermediate clauses $\Box^{s-1}\phi$, $\ldots$, $\Box\phi$ are not necessary as they do not contain as much information as $\Box^s\phi$, and cannot be directly used to introduce $\perp$ as ϕ. Hence these clauses are not present in the denominator, and we restrict ϕ to be a simple clause. Once the clause $\Box^s\phi$ has been used at a world w to introduce ϕ, it cannot bring anything else to w in order to introduce $\perp$, hence we block using it at w by replacing it by $\blacksquare^s\phi$. Clauses of the form $\blacksquare^s\phi$ are unblocked in applications of transitional rules. In the calculus $\mathcal{C}B$, clauses of the form $\Box^s[a, \Box b]$

In this table, let $s \geq 1$.

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

(K) $\dfrac{X; \blacksquare Y; \Box Z; \neg\Box a}{Y; Z; \neg a}$ $\qquad$ (KD) $\dfrac{X; \blacksquare Y; \Box Z}{Y; Z}$

(T) $\dfrac{X; \Box^s \phi}{X; \blacksquare^s \phi; \phi}$ where ϕ is a simple clause

(T') $\dfrac{X; \Box^s \phi}{X; \blacksquare^s \phi; \phi}$ where ϕ is a simple clause not of the form $[\mathrm{a}, \Box\mathrm{b}]$

(KB) $\dfrac{X; \Box^{2s} \phi; \neg\Box a}{X; \blacksquare^{2s} \phi; \phi; \neg\Box a}$ $\qquad$ (KDB) $\dfrac{X; \Box^{2s} \phi}{X; \blacksquare^{2s} \phi; \phi}$

where ϕ is either a simple clause or a $\Box$-simple clause

(KB_b) $\dfrac{X; \Box[a, \Box b]}{X; \blacksquare\phi; \Box a \mid X; \blacksquare\phi; b}$ $\qquad$ (KTB) $\dfrac{X; \Box^s [a, \Box b]}{X; \blacksquare^s \phi; \Box a | X; \blacksquare^s \phi; \Box b | X; \blacksquare^s \phi; a; b}$

where $\phi = [\mathrm{a}, \Box\mathrm{b}]$

Table 3: Clausal tableau rules

are solved by the rule (KTB), so we use (T') instead of (T). For specific intuition of other rules, we refer readers to the proof of Lemma 2 given below.

3.1 Soundness

Lemma 2 *All rules in Table 3 are sound wrt. any normal modal logic extending the axioms listed in the rule's name.*

Proof. Fix a rule R from Table 3. Let L be a normal modal logic extending the axioms listed in the name of R. Let R' be an instance of R. We show that if the numerator of R' is satisfiable in L then so is one of the denominators of R'.

This assertion obviously holds for $(\vee)$ and $(\perp)$.

For (K), let M be a Kripke model and let $M, w \vDash X; \blacksquare Y; \Box Z; \neg\Box a$. There exists a world u reachable from w such that $M, u \vDash \neg a$. It is easily seen that $M, u \vDash Y; Z; \neg a$.

For (KD), let M be a KD-model and let $M, w \vDash X; \blacksquare Y; \Box Z$. Let u be a world reachable from w. We have $M, u \vDash Y; Z$.

For (T) and (T'), let M be a T-model and let $M, w \vDash X; \Box^s \phi$. It should be clear that $M, w \vDash X; \blacksquare^s \phi; \phi$.

For (KB), let M be a KB-model and let $M, w \vDash X; \Box^{2s} \phi; \neg\Box a$. There exists a world u reachable from w because $\neg\Box a$ is satisfied at w. We have $M, u \vDash \Box^{2s-1} \phi$, hence $M, w \vDash \Box^{2s-2} \phi$ by symmetry. Repeat the argument until we obtain $M, w \vDash \phi$. Therefore $M, w \vDash X; \blacksquare^{2s} \phi; \phi; \neg\Box a$.

$\mathcal{C}L$	Static Rules	Transitional Rules
$\mathcal{C}K$	$(\vee), (\perp)$	(K)
$\mathcal{C}KD$	$(\vee), (\perp)$	$(K), (KD)$
$\mathcal{C}T$	$(\vee), (\perp), (T)$	(K)
$\mathcal{C}KB$	$(\vee), (\perp), (KB), (KB_b)$	(K)
$\mathcal{C}KDB$	$(\vee), (\perp), (KDB), (KB_b)$	$(K), (KD)$
$\mathcal{C}B$	$(\vee), (\perp), (KTB), (T')$	(K)

Table 4: Clausal tableau calculi for propositional modal logics

Similarly, the assertion holds for (KDB).

For (KB_b), let M be a KB-model and let $M, w \models X; \Box[a, \Box b]$. Suppose that $M, w \not\models \Box a$, we show that $M, w \models b$. There exists a world u reachable from w such that $M, u \models \neg a$. Thus $M, u \models \Box b$, hence $M, w \models b$.

For (KTB), let M be a B-model and let $M, w \models X; \Box^s[a, \Box b]$. We have $M, w \models \Box[a, \Box b]; [a, \Box b]$. Reasoning similarly as for (KB_b) we derive $M, w \models [\Box a, b]$. Therefore one of the following conditions must hold: $(M, w \models \Box a)$, $(M, w \models \Box b)$, $(M, w \models a; b)$. $\Box$

Corollary 3 *The calculi listed in Table 4 are sound.*

3.2 Estimating Depths of Tableaux

In this section, we define a measure function f such that for any rule given in Table 3 the measure of all its denominators is smaller than the measure of its numerator. It follows that our tableau systems have the finite tableau property. This property is helpful in proving completeness of our systems. If we replace all $\blacksquare$ in our systems by $\Box$, then the newly obtained systems will not have this property. However, one can show that they are still sound and complete.

The measure function f is defined as follows:

$$\begin{array}{l} f(a) = 0, \quad f(\neg\Box a) = 1 \\ f([A_1, \ldots, A_k]) = 1 + max\{f(A_i) \mid 1 \leq i \leq k\}, \quad \text{where } k > 1 \\ f(\Box^s \phi) = (f(\phi) + 2).s + f(\phi), \text{where } s \geq 1 \text{ and } \phi \text{ is a simple clause} \\ f(\blacksquare\phi) = f(\phi) + 1 \\ f(X; Y) = f(X) + f(Y) \end{array}$$

Observe that if R is one of the rules listed in Table 3, X is an instance of the numerator of R, and Y is one of the corresponding instances of the denominators of R, then $f(Y) < f(X)$. This leads to the following conclusion:

Lemma 4 *Let X be a set of clauses, with $rlength(X) = n$ and $mdepth(X) = s$. Then every $\mathcal{C}L$-tableau for X, where $\mathcal{C}L$ is one of the calculi $\mathcal{C}K$, $\mathcal{C}KD$, $\mathcal{C}T$, $\mathcal{C}KB$, $\mathcal{C}KDB$, and $\mathcal{C}B$, has depth of order $O((s+1).n)$.*

Proof. It suffices to show that $f(X)$ has order $O((s+1).n)$. Observe that if ϕ is a simple clause then $f(\phi) \leq 3$. If ϕ is a clause then $f(\phi) \leq (3+2).s + 3 < 5(s+1)$. Therefore $f(X) < 5(s+1).n$. $\Box$

3.3 Completeness

From now on we use L to denote one of the logics K, KD, T, KB, KDB, and B; and we use $\mathcal{C}L$ to denote the corresponding calculus. To show completeness of the calculi we give an algorithm to construct a L-model graph for X for a given $\mathcal{C}L$-consistent set X of clauses not containing the connective $\blacksquare$. In the algorithm, saturation is exploited. We saturate a node by applying static $\mathcal{C}L$-rules to it until no changes occur. When a node has been saturated, its successors will be created according to transitional rules. Using saturation we will not have to update contents of the nodes we have processed. The method has been previously used by other authors, e.g. Hintikka [4], Rautenberg [10] and Goré [3].

We use $\boxtimes$ to denote either $\Box$ or $\blacksquare$, for instance we write $\boxtimes^s\phi \in X$ to mean $(\Box^s\phi \in X)$ or $(\blacksquare^s\phi \in X)$. We define the relation $\in_\boxtimes$ as follows:

$$\phi \in_\boxtimes X \equiv \begin{cases} \phi \in X & \text{if } \phi \text{ is not of the form } \boxtimes\psi \\ \boxtimes\psi \in X & \text{if } \phi = \Box\psi \text{ or } \phi = \blacksquare\psi \end{cases}$$

For example, $\Box\phi \in_\boxtimes \{\blacksquare\phi\}$. We write $\phi \in_\vee X$ to mean that if ϕ is of the form $[A_1, \ldots, A_k]$, with $k > 1$, then $A_i \in_\boxtimes X$ for some $1 \leq i \leq k$, else $\phi \in_\boxtimes X$.

Definition 5 *We say that a set X of clauses is $\mathcal{C}L$-*saturated *if no static $\mathcal{C}L$-rule is applicable for X.*

Definition 6 *Let X be a $\mathcal{C}L$-consistent set of clauses. We say that a set Y of clauses is a $\mathcal{C}L$-*saturation *of X if:*

- *Y can be obtained from X by applying static $\mathcal{C}L$-rules, and*
- *Y is $\mathcal{C}L$-consistent and $\mathcal{C}L$-saturated.*

Lemma 7 *For any $\mathcal{C}L$-consistent set X of clauses, there exists a $\mathcal{C}L$-saturation of X.*

Proof. We apply static $\mathcal{C}L$-rules to X until no changes occur. When applying a rule to X we always choose a $\mathcal{C}L$-consistent denominator. The process always terminates because every $\mathcal{C}L$-tableau for X is finite. The resulting set is a $\mathcal{C}L$-saturation of X. $\Box$

Here we give intuition for "redundants" defined below. It requires referring to Algorithm 10 given in page 10. Given a $\mathcal{C}L$-consistent set X of clauses, the algorithm constructs a L-model graph $M = \langle W, \tau, R, H \rangle$ for X. The skeleton of M is the tree R_0. The algorithm always goes downward along R_0; it never recomputes contents of the nodes it has passed. If $KB \leq L$ then for any nodes w and u such that $R_0(w, u)$, we have $R(u, w)$. This implies that if $KB \leq L$, $R_0(w, u)$, and $\boxtimes c \in H(u)$, then c should be present in $H(w)$. Hence, if $KB \leq L$ and $R_0(w, u)$, then being at u (and having $H(w)$ fixed) the strategy is to reduce clauses of the form $\boxtimes c$ in u. This leads to the following formulation of redundants.

Definition 8 *Let X be a $\mathcal{C}L$-consistent set of clauses, and Y be a $\mathcal{C}L$-saturation of X. We call an atom $\boxtimes c \in Y$ a* redundant *with respect to L and X if $L = KB$ or $L = KDB$ and:*

- *X contains no clauses of the form $\boxtimes^{2s+1}c$, where $s \geq 0$*
- *For any formula $\Box^{2s}[a, \Box c] \in X$, where $s \geq 0$, it holds that $a \in Y$*

- *For any formula* $\Box^{2s+1}[c, \Box b] \in X$*, where* $s \geq 0$*, it holds that* $b \in Y$*;*

or $L = B$ *and:*

- X *contains no clauses of the form* $\boxtimes^s c$*, where* $s \geq 1$
- *For any formula* $\Box^s[a, \Box c] \in X$*, where* $s \geq 0$*, it holds that* $a \in Y$
- *For any formula* $\Box^s[c, \Box b] \in X$*, where* $s \geq 1$*, it holds that* $b \in Y$*.*

We denote the set obtained from Y *by deleting all redundants wrt.* L *and* X *by* $RemoveRedundants_L(Y, X)$*.*

(Note that if X is a $\mathcal{C}L$-consistent set of clauses, Y is a $\mathcal{C}L$-saturation of X, and $Z = RemoveRedundants_L(Y, X)$, then Z is also a $\mathcal{C}L$-saturation of X.)

We give below an auxiliary lemma concerning properties of $\mathcal{C}L$-saturations.

Lemma 9 *Let* X *be a* $\mathcal{C}L$*-consistent set of clauses not containing the connective* $\blacksquare$*. Let* Y *be a* $\mathcal{C}L$*-saturation of* X*, and let* $Z = RemoveRedundants_L(Y, X)$*. Then the following assertions hold:*

1. Z *is* $\mathcal{C}L$*-consistent.*
2. *If* $\phi \in X$ *then* $\phi \in_\vee Z$*.*
3. *If* $L = T$ *and* $\boxtimes^s \phi \in Z$*, where* $s \geq 1$ *and* ϕ *is a simple clause, then* $\phi \in_\vee Z$*.*
4. *If* $L = B$ *and* $\boxtimes^s \phi \in Z$*, where* $s \geq 1$ *and* ϕ *is a simple clause of one of the forms* $[a_1, \ldots, a_k]$*,* $[a, \neg\Box b]$*, then* $\phi \in_\vee Z$*.*
5. *If* $L = KB$ *or* $L = KDB$*, and* $\boxtimes^{2s} \phi \in Z$*, where* $s \geq 1$ *and* ϕ *is a simple or* $\Box$*-simple clause, then* $\phi \in_\vee Z$*.*
6. *If* $\phi \in Z$ *and* $\phi = \boxtimes^s \psi$*, where* $s \geq 1$ *and* ψ *is a simple clause such that if* $s = 1$ *then* ψ *is not a classical literal, then there exists* $t \geq 0$ *such that* $\Box^{s+2t}\psi \in X$*.*
7. *If* $L = KB$ *or* $L = KDB$*, and* $\boxtimes[a, \Box b] \in Z$*, then* $\boxtimes a \in Z$ *or* $b \in Z$*.*
8. *If* $L = B$ *and* $\boxtimes^s[a, \Box b] \in Z$*, where* $s \geq 1$*, then (*$\boxtimes a \in Z$ *and* $a \in Z$*) or (*$\boxtimes b \in Z$ *and* $b \in Z$*) or (*$a \in Z$ *and* $b \in Z$*).*

Proof. The proofs of 1 and 2 are straightforward.

3) Suppose that $L = T$. The set Z cannot contain clauses of the form $\Box^s \phi$ with $s \geq 1$. Suppose that $\blacksquare^s \phi \in Z$, where $s \geq 1$ and ϕ is a simple clause. It follows that $\Box^s \phi \in X$, and ϕ is introduced during saturation of X. If ϕ is of the form $[A_1, \ldots, A_k]$, then $A_i \in_\boxtimes Z$ for some $1 \leq i \leq k$, else $\phi \in Z$. Thus $\phi \in_\vee Z$.

4) The proof is similar to the proof of 3.

5) Suppose that $L = KB$ or $L = KDB$. The set Z cannot contain clauses of the form $\Box^{2s}\phi$ with $s \geq 1$. Suppose that $\blacksquare^{2s}\phi \in Z$, where $s \geq 1$ and ϕ is a simple or $\Box$-simple clause. It follows that $\Box^{2s}\phi \in X$, and ϕ is introduced during saturation of X. Consider the following cases:

- If ϕ is of the form $\Box a$, then $\phi \in Z$, because $\Box a$ is not a redundant wrt. L and X.

- If ϕ is of the form $\Box\psi$, where ψ is not a classical literal, then $\phi \in_{\boxtimes} Z$.
- If ϕ is a simple clause not of the form $[a, \Box b]$, then $\phi \in_{\vee} Z$.
- For the last case, suppose that ϕ is of the form $[a, \Box b]$. If $a \notin Z$ then $\Box b \in Y$, and since $\Box b$ is not a redundant wrt. L and X, it follows that $\Box b \in Z$.

For all of these cases we can conclude that $\phi \in_{\vee} Z$.

6) Suppose that the conditions of the assertion hold. If $\phi = \Box^s\psi$ then $\phi \in X$, and $\Box^{s+2t}\psi \in X$ for $t = 0$. Suppose that $\phi = \blacksquare^s\psi$. It follows that $L \in \{T, KB, KDB, B\}$. The assertion clearly holds for $L = KB$ or $L = KDB$. If $L = T$ or $L = B$ then $\Box^s\psi \in X$, and we choose $t = 0$.

7) Suppose that $L = KB$ or $L = KDB$, $\boxtimes[a, \Box b] \in Z$, and $b \notin Z$. We show that $\boxtimes a \in Z$. We have $b \notin Y$ and $\Box^{2s+1}[a, \Box b] \in X$ for some $s \geq 0$. Hence either the rule (KB) or (KDB), and then the rule (KB_b) must be used in the saturation, and $\Box a \in Y$. Since $b \notin Y$, $\Box a$ is not a redundant wrt. L and X. Hence $\Box a \in Z$.

8) Suppose that $L = B$ and $\boxtimes^s[a, \Box b] \in Z$. Suppose that $b \notin Z$, we show that $\boxtimes a \in Z$ and $a \in Z$. We have $b \notin Y$ and $\Box^s[a, \Box b] \in X$. The rule (KTB) must be used in the saturation, and $\Box a$ is introduced. Since $b \notin Y$, the clauses $\Box a$ and $\blacksquare a$ are not redundants wrt. L and X. Hence $\blacksquare a \in Z$ and $a \in Z$. Analogously, if $a \notin Z$ then $\boxtimes b \in Z$ and $b \in Z$. We conclude that the assertion holds. $\Box$

In the following algorithm we assume that we have an oracle to compute the step 2b. For how this algorithm can be implemented, see Section 4.

Algorithm 10 *Let X be a $\mathcal{CL}$-consistent set of clauses not containing the connective $\blacksquare$. We construct a consistent L-model graph $M = \langle W, \tau, R, H\rangle$ that satisfies X as follows:*

1. *Let $W = \{\tau\}$, $H_0(\tau) = X$, $R_0 = \emptyset$, and mark τ as unsolved.*
2. (a) *Take an unsolved world w from W.*
   (b) *Let Y be a $\mathcal{CL}$-saturation of $H_0(w)$.*
   (c) *Set $H(w) = RemoveRedundants_L(Y, H_0(w))$.*
   (d) *For every atom $\neg\Box a$ from $H(w)$:*
      - *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 $H_0(w_a) = \{\neg a\} \bigcup \{Y \mid \boxtimes Y \in H(w)\}$.*

   (e) *If $L = KD$ or $L = KDB$, and there is no atom of the form $\neg\Box a$ in $H(w)$ then*
      - *If $H(w)$ is not empty*
         - *Create a new world w', add it to W and mark it as unsolved.*
         - *Set $R_0 = R_0 \bigcup \{(w, w')\}$.*
         - *Set $H_0(w') = \{Y \mid \boxtimes Y \in H(w)\}$.*
      - *else set $R_0 = R_0 \bigcup \{(w, w)\}$.*

   (f) *Mark w as resolved.*
3. *While there are unsolved worlds, repeat the step 2.*
4. *Set $R = Ext_L(R_0)$.*

The step 2e takes place when $L = KD$ or $L = KDB$, and it connects w to at least one node. The step 2d corresponds to the rule (K), the step 2e to the rule (KD).

Note that the relation R_0 in the above algorithm, without self-referencing edges, forms a tree. For any nodes w and u such that $R_0(w,u)$, we have $mdepth(H(u)) < mdepth(H(w))$. Therefore the depth of the tree R_0 is bounded by the modal-depth of the input X. A consequence of this is that Algorithm 10 always terminates.

Lemma 11 *Let X be a $\mathcal{C}L$-consistent set of clauses not containing the connective $\blacksquare$, and let $M = \langle W, \tau, R, H\rangle$ be the model graph constructed by Algorithm 10 for X. Then M is a consistent L-model graph satisfying X.*

Proof. Since X is $\mathcal{C}L$-consistent, by Lemma 9:1, M is a consistent L-model graph. By Lemma 9:2, for any clause $\phi \in X$ we have $\phi \in_\vee H(\tau)$. To prove that M satisfies X it is sufficient to show that for any $w \in W$ and $\phi \in H(w)$, $M, w \vDash \phi$. Denote this assertion by $(\star)$. It clearly holds for the case when ϕ is of the form $\neg\Box a$. We observe that for any $w \in W$, $H(w)$ does not contain clauses of the form $[A_1, \ldots, A_k]$, with $k > 1$.

We now show that $(\star)$ holds for the case $\phi = \boxtimes c$, where c is a classical literal.

The case $L \in \{K, KD, T\}$ is trivial.

Case $L = KB$ or $L = KDB$: Suppose that $\boxtimes c \in H(w)$. We have $\forall u \in W\ R_0(w,u) \to c \in H(u)$. It remains to show that $\forall u\ R_0(u,w) \to c \in H(u)$. Suppose that $R_0(u,w)$ holds. Since $\boxtimes c \in H(w)$, $\boxtimes c$ is not a redundant wrt. $H_0(w)$ and L. Hence one of the following conditions must hold:

- $\boxtimes^{2s+2} c \in H(u)$ for some $s \geq 0$
- $\boxtimes^{2s+1}[a, \Box c] \in H(u)$ for some $a \notin H(w)$ and $s \geq 0$
- $\boxtimes^{2s+2}[c, \Box b] \in H(u)$ for some $b \notin H(w)$ and $s \geq 0$

If the first condition holds, we have $c \in H(u)$, by Lemma 9:5. Suppose that the second condition holds and $c \notin H(u)$. By Lemma 9:5, we have $\boxtimes[a, \Box c] \in H(u)$. Hence $\boxtimes a \in H(u)$, by Lemma 9:7. This contradicts the fact that $a \notin H(w)$. Now assume that the third conditions holds and $c \notin H(u)$. By Lemma 9:5, we have $[c, \Box b] \in_\vee H(u)$, which implies $\boxtimes b \in H(u)$. This contradicts the fact that $b \notin H(w)$. Hence $c \in H(u)$.

Case $L = B$: Suppose that $\boxtimes c \in H(w)$. We have $\forall u \in W\ R_0(w,u) \to c \in H(u)$. By Lemma 9:4, we have $c \in H(w)$. It remains to show that $\forall u\ R_0(u,w) \to c \in H(u)$. Suppose that $R_0(u,w)$ holds. Since $\boxtimes c \in H(w)$, $\boxtimes c$ is not a redundant wrt. $H_0(w)$ and L. Hence one of the following conditions must hold:

- $\boxtimes^s c \in H(u)$ for some $s \geq 2$
- $\boxtimes^s [a, \Box c] \in H(u)$ for some $a \notin H(w)$ and $s \geq 1$
- $\boxtimes^s [c, \Box b] \in H(u)$ for some $b \notin H(w)$ and $s \geq 2$

If the first condition holds, we have $c \in H(u)$, by Lemma 9:4. Suppose that the second condition holds and $c \notin H(u)$. By Lemma 9:8, we have $\boxtimes a \in H(u)$, which contradicts the fact that $a \notin H(w)$. Now assume that the third condition holds and $c \notin H(u)$. By Lemma 9:8, we have $\boxtimes b \in H(u)$, which contradicts the fact that $b \notin H(w)$. Hence $c \in H(u)$.

It remains to show that ($\star$) holds for the case $\phi = \boxtimes^s\psi$, where $s \geq 1$ and ψ is a simple clause such that if $s = 1$ then ψ is not a classical literal.

Case $L = K$ or $L = KD$: If $\boxtimes^s\psi \in H(w)$ and $R(w,u)$ hold, then $\Box^{s-1}\psi \in_\vee H(u)$. By induction on s, it follows that if $\phi \in H(w)$ then $M, w \vDash \phi$.

Case $L = T$: We show by induction on i that for any $u \in W$, if there exists $s \geq i$ such that $\boxtimes^s\psi \in H(u)$ then $M, u \vDash \Box^i\psi$. Case $i = 0$: Suppose that $\boxtimes^s\psi \in H(u)$. By Lemma 9:3, we have $\psi \in_\vee H(u)$, hence $M, u \vDash \psi$. Now assume that the assertion holds for all $0 \leq j < i$, and $\boxtimes^s\psi \in H(u)$ for some $s \geq i$. We show that $M, u \vDash \Box^i\psi$. By induction, we have $M, u \vDash \Box^{i-1}\psi$. For any v such that $R_0(u,v)$, we have $\boxtimes^{s-1}\psi \in H(v)$, which by induction implies $M, v \vDash \Box^{i-1}\psi$. Thus $M, u \vDash \Box^i\psi$. As a consequence, if $\phi \in H(w)$ then $M, w \vDash \phi$.

Case $L = KB$ or $L = KDB$: We show by induction on i that for any $u \in W$, if there exists $k \geq 0$ such that $\boxtimes^{i+2k}\psi \in H(u)$ then $M, u \vDash \Box^i\psi$. Case $i = 0$: Suppose that $\boxtimes^{2k}\psi \in H(u)$. By Lemma 9:5, we have $\psi \in_\vee H(u)$, hence $M, u \vDash \psi$. Now assume that the assertion holds for all $0 \leq j < i$, and $\boxtimes^{i+2k}\psi \in H(u)$ for some $k \geq 0$. We show that $M, u \vDash \Box^i\psi$. For any v such that $R_0(u,v)$, we have $\boxtimes^{i+2k-1}\psi \in H(v)$, which by induction implies $M, v \vDash \Box^{i-1}\psi$. For any v such that $R_0(v,u)$, by Lemma 9:6, we have $\boxtimes^{i+2k+2t+1}\psi \in H(v)$ for some $t \geq 0$, which by induction implies $M, v \vDash \Box^{i-1}\psi$. Thus $M, u \vDash \Box^i\psi$. As a consequence, if $\phi \in H(w)$ then $M, w \vDash \phi$.

Case $L = B$: We show by induction on i that for any $u \in W$, if there exists $s \geq i$ such that $\boxtimes^s\psi \in H(u)$ then $M, u \vDash \Box^i\psi$. Case $i = 0$: Suppose that $\boxtimes^s\psi \in H(u)$. If ψ is of the form $[a, \Box b]$ then by Lemma 9:8 we have $a \in H(u)$ or $\boxtimes b \in H(u)$, hence $M, u \vDash \psi$. If ψ is not of the form $[a, \Box b]$ then by Lemma 9:4 we have $\psi \in_\vee H(u)$, which implies $M, u \vDash \psi$. Now assume that the assertion holds for all $0 \leq j < i$, and $\boxtimes^s\psi \in H(u)$ for some $s \geq i$. We show that $M, u \vDash \Box^i\psi$. By induction, we have $M, u \vDash \Box^{i-1}\psi$. For any v such that $R_0(u,v)$, we have $\boxtimes^{s-1}\psi \in H(v)$, which by induction implies $M, v \vDash \Box^{i-1}\psi$. For any v such that $R_0(v,u)$, by Lemma 9:6, we have $\boxtimes^{s+2t+1}\psi \in H(v)$ for some $t \geq 0$, which by induction implies $M, v \vDash \Box^{i-1}\psi$. Thus $M, u \vDash \Box^i\psi$. As a consequence, if $\phi \in H(w)$ then $M, w \vDash \phi$. $\Box$

The following theorem immediately follows from Corollary 3 and Lemma 11

Theorem 12 *The calculi $\mathcal{C}K$, $\mathcal{C}KD$, $\mathcal{C}T$, $\mathcal{C}KB$, $\mathcal{C}KDB$ and $\mathcal{C}B$ are sound and complete.*

4 Space Bounds for the Logics

In this section, we estimate space bounds for the considered modal logics. Restrictive lengths of sets of clauses will play an important role in estimation.

We need the following auxiliary lemma:

Lemma 13 *Let X be an input to Algorithm 10, with restrictive length n and modal-depth s, $s \leq n$. Then for any node w of the resulting model graph, the restrictive length of $H(w)$ is of order $O(n)$, and $H(w)$ can be coded in $O(n.\log n)$-space.*

Proof. We define the set $sc(X)$, which contains significant subclauses of X, to be the smallest set satisfying the following. Let $\phi = \Box^s\psi$, where ψ is a simple clause and $s \geq 0$, and let $\phi \in X$. Then the followings clauses belong to $sc(X)$: ϕ, $\blacksquare^s\psi$, $\Box\psi$, $\blacksquare\psi$, ψ, and

- $a_1, \ldots, a_k$, when $\psi = [a_1, \ldots, a_k]$;

- $\Box a$, $\blacksquare a$, a, $\Box b$, $\blacksquare b$, b, when $\psi = [a, \Box b]$;
- a, $\neg\Box b$, $\neg b$, when $\psi = [a, \neg\Box b]$;
- $\neg b$, when $\psi = \neg\Box b$.

Observe that the restrictive length of $sc(X)$ is of order $O(n)$. For any node w of the resulting model graph we have $H(w) \subseteq sc(X)$, hence the restrictive length of $H(w)$ is of order $O(n)$. We use $O(\log n)$-space to encode each sequence $\boxtimes^k$, $O(1)$-space for each other connective, and $O(\log n)$-space for each primitive proposition. Therefore $H(w)$ can be coded in $O(n.\log n)$-space. $\Box$

Definition 14 *Let X and Y be sets of clauses. We say that Y is a candidate for $\mathcal{C}L$-saturation of X if Y can be obtained from X by applying static $\mathcal{C}L$-rules, and Y is $\mathcal{C}L$-saturated (i.e. we reject the condition demanding Y to be $\mathcal{C}L$-consistent).*

We translate Algorithm 10 to a nondeterministic one, to overcome the assumption about the oracle, by ignoring the computation of W and R, and replacing the step 2b by:

"Nondeterministically generate a candidate for $\mathcal{C}L$-saturation of $H_0(w)$, and assign it to $H(w)$. If $H(w)$ is inconsistent in the sense that it contains $\perp$, then reject the computation."

Observe that for any set X of clauses not containing the connective $\blacksquare$, X is L-satisfiable iff the new algorithm has an unrejected computation for X.

We then simulate the obtained algorithm by a deterministic one, denoted by $\mathcal{A}$, by backtracking. During computation of $\mathcal{A}$ we only need to keep information about the current path from the root τ to the current node. We record all and-or branching on the current path, and keep the contents of the root node and the current node.

Let n and s be the restrictive length and the modal-depth of X, respectively, and assume that $s \leq n$. By Lemma 13, the contents of the root node and the current node can be coded using $O(n.\log n)$-space.

We do not keep contents of other nodes on the current path as they can be reconstructed from the content of the root node and the information about and-or branching on the path. Let u be a node on the current path. Suppose that w is a predecessor of u, i.e. $R_0(w, u)$, and $H(w)$ has been reconstructed. The set $H(u)$ is reconstructed as follows. Using the information about and-branching from w to u, we compute the set $H_0(u)$, as defined in the steps 2d and 2e of Algorithm 10. We then repeatedly apply static $\mathcal{C}L$-rules to $H_0(u)$ accordingly to the information about or-branching at u until no more changes occur. The resulting set is $H(u)$. Note that the order in which we apply static $\mathcal{C}L$-rules to $H_0(u)$ is not important; in any order we obtain the same set. The task of computing $H(u)$ from $H(w)$ requires $O(n.\log n)$-space (mainly to store $H(u)$). Hence the content of any node on the current path can be reconstructed in $O(n.\log n)$-space.

When and-branching from one node to another in the search tree, we use $O(\log n)$-space to record the position of the clause $\neg\Box a$ (see the step 2d of Algorithm 10) in the *sequent*[1] X, treating sequences of $\boxtimes^k$ as primitive connectives. The length of every path is bounded by s, hence we need $O(s.\log n)$-space to keep all information about and-branching on the current path.

Case $L = K$ or $L = KD$: Each formula of the form $\Box^k[A_1, \ldots, A_h]$, where $k \geq 0$ and $h > 1$, causes or-branching at the depth k by choosing some A_i, $1 \leq i \leq h$. For each formula of that form, we need $O(\log n)$-space to keep the index i. Therefore we need $O(n.\log n)$-space to keep all information about

[1]We treat X as a sequent in order to encode it.

or-branching on the current path. Summing up, when $L = K$ or $L = KD$, the algorithm $\mathcal{A}$ runs in $O(n.\log n)$-space.

Case $L \in \{T, KB, KDB, B\}$: We claim that for any $w \in W$ all information about or-branching at w can be coded in $O(n)$-space. We reserve two bits for every sequence of the form $\Box^k$, and for every other symbol occurring in $H_0(w)$. Thus we have an array of length $O(n)$ of pieces of two bits. To keep information about or-branching caused by a clause of the form $\Box^k[a, \Box b]$, $k \geq 0$, we write into the piece reserved for the sequence $\Box^k$ (or for the comma if $k = 0$) the position (1, 2, or 3) of the chosen denominator. To keep information about or-branching caused by a clause of the form $[A_1, \ldots, A_h]$, but not of the form $[a, \Box b]$, we set the piece reserved for the chosen disjunct to 1, and the other pieces to 0. Since the length of every path is bounded by s, we conclude that it needs $O((s+1).n)$-space to keep all information about or-branching on the current path. Summing up, when $L \in \{T, KB, KDB, B\}$, the algorithm $\mathcal{A}$ runs in $O(n.(s + \log n))$-space (this amount is computed as $O(n.\log n) + O(s.\log n) + O((s+1).n)$, with $s \leq n$).

Theorem 15 *The logics K and KD are decidable in $O(n.\log n)$-space, and the logics T, KB, KDB, and B are decidable in $O(n^2)$-space. The problem of checking satisfiability in T, KB, KDB, or B for formulae with finitely bounded modal-depth is decidable in $O(n.\log n)$-space. (Here n is the length of an input formula.)*

The space bound $O(n.\log n)$ for the logic K was first established by Hudelmaier [5], and for the logic KD by Basin, Matthews, and Viganò [1].

Proof. It suffices to recall that for any formula ϕ with length n and modal-depth s (here we have $s < n$), we can use $O(n.\log n)$-space to translate ϕ to an equisatisfiable, in any normal modal logic, set X of clauses such that the restrictive length of X is linearly bounded by n, and the modal-depth of X is equal to s. $\Box$

5 What is Behind Our Method ?

First, why have we defined calculi only for sets of clauses? The answer is that due to simple forms of clauses it is easy to analyze behavior of clauses in deduction. Until the moment, according to Góre [3], there were no tableau systems with implicit accessibility for the logic KB and KDB. The tableau system with implicit accessibility for the logic B by Rautenberg [10] contains cut rules. Using clauses we have been successful to construct tableau systems for the logics KB, KDB and B, without cut rules.

We have used a well-known method to prove completeness of our tableau systems. The method has been previously used in the work of Hintikka [4], Rautenberg [10] and Goré [3]. To show that a $\mathcal{C}L$-consistent set X of clauses is $\mathcal{C}L$-satisfiable, we construct a L-model graph for X. We complete building a node by saturating its content (i.e. by applying static $\mathcal{C}L$-rules to it until no changes occur), and then generating its successors accordingly to transitional rules. One of the aims of saturation is that we will not have to update contents of the nodes we have processed. We do not saturate model graphs as we do not require them to satisfy the formula $R(w, u) \wedge (\Box\psi \in H(w)) \rightarrow \psi \in H(u)$. A model graph M constructed by our method, however, satisfies the assertion $\phi \in H(w) \rightarrow M, w \vDash \phi$. To prove this assertion we have used induction.

Our method is good in estimating space bounds for modal logics. Saturating content of a node means grouping applications of static rules to that node together. By grouping applications of rules we *shorten the depth of the search tree*. Our static rules given in Table 3 do not block each other, and if two static rules are applicable then it does not matter which rule is applied first; this means that static rules of our systems are permutable. Hence *we do not have to keep the order of applications of static rules to a node*. Another gain is that *there is an effective method to code a saturation* (we use only $O(n)$-space to code $O(n)$ or-branching at a node).

6 Comparison with Related Works

While the work of Hudelmaier [5] and the works of Basin, Matthews and Viganò [1, 11] base on sequent systems, our work bases on tableau systems. We use "semantical" proofs to show soundness and completeness of our systems, whereas Hudelmaier and Basin et al used proof-theoretic proofs for their systems. At this aspect, which method is preferable is a matter of taste.

We have followed Hudelmaier in translating formulae to clauses, and defining deduction systems only for sets of clauses. The connective $\blacksquare$ has the same role as the connective $\bigcirc$ introduced by Hudelmaier in [5]. Our systems $\mathcal{C}K$ and $\mathcal{C}T$ are very similar to the ones of his. The differences lie in proofs of completeness and in analysis of space requirements for the logics.

We are the first who explicitly establish space requirements for the logics KB, KDB and B. The space bound $O(n.\log n)$ for the logic K was first established by Hudelmaier [5], and later also proved by Basin et al [1]. The space bound $O(n.\log n)$ for the logic KD is due to Basin et al [1].

In [5] Hudelmaier gave the space bound $O(n.\log n)$ for the logic K and T. However, he has accepted some wrong assumption. Namely, he claimed that when a formula is translated to clausal form, linearly many connectives are added to the output sequent. The correct statement should be that when a formula of length n is translated to clausal form, the number of connectives in the output sequent is of order $O(n^2)$. To see this, consider formulae $\phi_n = \Box^n(\Box p_1 \vee \Box p_2 \vee \ldots \vee \Box p_n)$. By Mints translation, ϕ_n is first translated to $(\Box^n[\Box p_1, q_1];\ \Box^n[\neg q_1, \Box p_2, q_2];\ \ldots;\ \Box^n[\neg q_{n-2}, \Box p_{n-1}, q_{n-1}];\ \Box^n[\neg q_{n-1}, \Box p_n])$, and then to $(\Box^n[q_1, \Box p_1];\ \Box^n[\neg q_1, q_2, r_2];\ \Box^n[\neg r_2, \Box p_2];\ \ldots;\ \Box^n[\neg q_{n-2}, q_{n-1}, r_{n-1}];\ \Box^n[\neg r_{n-1}, \Box p_{n-1}];\ \Box^n[\neg q_{n-1}, \Box p_n])$. Denote the later sequent by ψ_n. Now note that while the number of connectives in ϕ_n is of order $O(n)$, the number of connectives in ψ_n is greater than n^2 (note that the number of connectives in $\Box^n$ is n). Therefore "the maximal length of a deduction" of a sequent in $C_1(T)$ (see [5]) is of order $O(n^2)$, and space bound obtained by the Hudelmaier's method in [5] for the logic T should be $O(n^2.\log n)$. For the logic K, the space bound $O(n.\log n)$ for the algorithm given by Hudelmaier is true, however, the proof needs a correction. A possible solution is as follows:

"Define $d(s) = rlength(s) + mdepth(s)$. When a formula ϕ is translated to clausal form, to a sequent X, $d(X)$ is linearly bounded by the length of ϕ. The measure d decreases under backwards application of the rule $CK\Box_0$. Thus every deduction of a sequent in $C_0(K)$ has length of order $O(n)$."

In this work, we have introduced *restrictive length* of a set of clauses in order to overcome the problem.

In [1] Basin et al gave the space bound $O(n.\log n)$ for the logic T. It is better than our space bound $O(n^2)$ for T.

7 Conclusions

We have presented clausal tableau systems for the common modal logics K, KD, T, KB, KDB and B. The tableau systems have a measure such that for each rule of the systems the measure of all its denominators is smaller than the measure of its numerator. This property allows depth-first search without loop checking. We have given a decision procedure for the considered logics, which uses $O(n^2)$-space for the logics T, KB, KDB and B, and $O(n.\log n)$-space for the logics K and KD. We are the first who explicitly estimates space requirements for the logics KB, KDB and B. Our space bounds for the logics K and KD are equal to the ones previously established by Hudelmaier [5] and Basin et al [1]. We have also shown that the problem of checking satisfiability in T, KB, KDB, or B for formulae with finitely bounded modal-depth is decidable in $O(n.\log n)$-space.

Our method is applicable to construct tableau systems and estimate space requirements for other modal logics. It often gives good space bounds. See [9] for the case of the modal logics $K4$, $KD4$, $S4$.

Acknowledgements

The author wishes to express his thanks to Andrzej Szałas and the anonymous reviewers from Journal of Logic and Computation for many helpful comments.

References

[1] D. Basin, S. Matthews, and L. Viganò. A new method for bounding the complexity of modal logics. In *G. Gottlob, A. Leitsch, and D. Mundici, editors, Proceedings of the 5th Kurt Gödel Colloquium on Computational Logic and Proof Theory (KGC'97), LNCS 1289*, pages 89–102. Springer, 1997.

[2] B.F. Chellas. *Modal Logic: An Introduction.* Cambridge Univ. Press, 1980.

[3] R. Goré. Tableau methods for modal and temporal logics. In D'Agostino, Gabbay, Hänle, and Posegga, editors, *Handbook of Tableau Methods*, pages 297–396. Kluwer Academic Publishers, 1999.

[4] K.J.J. Hintikka. Form and content in quantification theory. *Acta Philosophica Fennica*, 8:3–55, 1955.

[5] J. Hudelmaier. Improved decision procedures for the modal logics K, T, S4. In *H. Kleine Büning, editor, Proceedings of CSL'95, LNCS 1092*, pages 320–334. Springer, 1996.

[6] G.E. Hughes and M.J. Cresswell. *An Introduction to Modal Logic.* Methuen, 1968.

[7] S.A. Kripke. Semantical analysis of modal logic, i. normal modal propositional calculus. *Z. Math. Logic Grundlag*, 9:67–96, 1963.

[8] 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.

[9] L.A. Nguyen. A new space bound for the modal logics K4, KD4 and S4. In *M. Kutylowski and L. Pacholski, editors, Proceedings of MFCS'99.* Springer, 1999.

[10] W. Rautenberg. Modal tableau calculi and interpolation. *Journal of Philosophical Logic*, 12:403–423, 1983.

[11] L. Viganò. *A framework for non-classical logics.* PhD thesis, Universität des Saarlandes, Saabrücken, Germany, 1997.