Complexity of Logics with Bounded Modalities

Robin Hirsch, Evan Tzanis

August 17, 2009

Abstract

For $n \in \mathbb{N}$, we define the following modal operators: $\Box^{=n}, \Box^{\leq n}, \Box^{\geq n}, \Box^{*}, \blacksquare$ with intended meanings 'at every point n steps away', 'at every point within n steps', 'at every point at least n steps away', 'at every point from now on', and 'at every point', respectively. $\mathcal{L}(\Box^{=n}, \blacksquare : n \in \mathbb{N})$ is the modal language of modal propositional formulas built using $\blacksquare$ and any $\Box^{=n}$, for $n \in \mathbb{N}$, for example. Modal logics using these operators can be very concise, because the superscripts of the modal operators are binary encoded. We consider the complexity of the satisfiability problems over arbitrary Kripke frames and over linear discrete frames. We consider a number of variants of **PDL** where binary encoded program iterators $^{=n}, ^{\leq n}$ are allowed, as well as program operators $\cup, \cap, ;, ^{*}$ for union, intersection, composition and reflexive transitive closure, and consider the complexity of the satisfiability problem. We have obtained the following results.

Language (Range)	Complexity
Arbitrary Frames	
$\mathcal{L}(\Box^{=n}, \Box^{\leq n} : n \in \mathbb{N})$	**EXPSPACE**-complete
$\mathcal{L}(\Box^{=n}, \blacksquare : n \in \mathbb{N}) - \mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \blacksquare : n \in \mathbb{N})$	**2-EXPTIME**-complete
$\mathcal{L}(\Box^{=n}, \Box^{*} : n \in \mathbb{N}) - \mathcal{L}(\Box^{=n}, \Box^{\leq n}, \Box^{\geq n}, \Box^{*} : n \in \mathbb{N})$	**2-EXPTIME**-complete
Frame $(\mathbb{N}, <)$	
$\mathcal{L}(\Box^{=n}, \Box^{*} : n \in \mathbb{N}) - \mathcal{L}(\Box^{=n}, \Box^{\leq n}, \Box^{\geq n}, \Box^{*} : n \in \mathbb{N})$	**EXPSPACE**-complete
Dynamic Logics	
$\mathcal{P}(\cap, \cup, ^{\leq n} : n \in \mathbb{N})$	**EXPSPACE**-complete
$\mathcal{P}^1(^{=n}, ^{\leq n} : n \in \mathbb{N}) - \mathcal{P}(\cap, \cup, ;, ^{=n}, ^{\leq n} : n \in \mathbb{N})$	**EXPSPACE**-complete
$\mathcal{P}(\cup, ^{=n}, ^{*} : n \in \mathbb{N}) - \mathcal{P}(\cup, ;, ^{=n}, ^{\leq n}, ^{\geq n}, ^{*} : n \in \mathbb{N})$	**2-EXPTIME**-complete
$\mathcal{P}(\cap, \cup, ;, ^{*}) - \mathcal{P}(\cap, \cup, ;, ^{=n}, ^{\leq n}, ^{\geq n}, ^{*} : n \in \mathbb{N})$	**2-EXPTIME**-complete

Keywords: Modal Logic, Complexity, Satisfiability Problem, Alternating Time Automata.

1 Introduction

The topic of this article is the complexity of various modal logics with 'bounded modalities'. These bounded modalities can express, for example, that a property will continue to hold at all points within n steps. Since we are interested in the complexity of such logics, it is convenient to represent this bound n as a binary number. This can make our formulas very concise, so we might expect a computational cost incurred by the use of these binary encoded bounds. This turns out to be the case, the logics we consider have fairly high complexities, and we will provide a detailed analysis.

For each modal logic dealt with here we can define a Kripke-like semantics and consider the complexity of the satisfiability problem for the logic over arbitrary frames. We will also consider the case of discrete, linear (non-branching) semantics.

The use of binary encoded bounds on modalities is not new. In this section we will briefly review the syntax and semantics of a number of well-known modal and branching logics which adopt this kind of binary encoded bounds, and consider the complexity of the satisfiability problem for each of them. In the following section we will introduce new modal logics which use bounded unary modalities, making them in this respect more simple than some of the previously considered logics. We will establish the complexity of their satisfaction problems.

Figure 1 summarises the main known languages and their complexities, but may need some illumination. First we explain the syntax of these logics. In each case, except for the final two, a

Name	Atoms	Unary C.	Binary Connectives	Complexity	Citation
			State Based Semantics		
Modal	Prop	$\Box$		**PSPACE-C**	[L77]
Modal Universal	Prop	$\Box, \blacksquare$		**EXPTIME-C**	[Hem96]
CTL	Prop	$\mathbf{EX}, \mathbf{AX}$	$\mathbf{EU}, \mathbf{AU}$	**EXPTIME-C**	[EMH82]
RCTL$^{\leq}$	Prop	$\mathbf{EX}, \mathbf{AX}$	$\mathbf{EU}, \mathbf{AU}, \mathbf{EU}^{\leq n}, \mathbf{AU}^{\leq n}$	**EXPTIME-C**	[EMSS92]
RCTL$^{\geq}$	Prop	$\mathbf{EX}, \mathbf{AX}$	$\mathbf{EU}, \mathbf{AU}, \mathbf{EU}^{\geq n}, \mathbf{AU}^{\geq n}$ $\mathbf{EU}^{\leq n}, \mathbf{AU}^{\leq n}$	**EXPTIME-C**	[EMSS92]
RCTL$^{=}$	Prop	$\mathbf{EX}, \mathbf{AX}$	$\mathbf{EU}, \mathbf{AU}, \mathbf{EU}^{=n}, \mathbf{AU}^{=n}$	**2-EXPTIME-C**	[EMSS92]
			Linear Path Based Semantics		
PLTL	Prop	$\mathbf{X}$	$\mathbf{U}$	**PSPACE-C**	[SC82]
TPTL	Prop, $\tau_1 \leq \tau_2$, $\tau_1 \equiv_d \tau_2$	$x.$ $\mathbf{X}$	$\mathbf{U}$	**EXPSPACE-C**	[AH94]
			State and Path Based Semantics		
CTL^{+}	Prop	$\mathbf{E}, \mathbf{X}$	$\mathbf{U}$	**2-EXPTIME-C**	[JL03]
CTL*	Prop	$\mathbf{E}, \mathbf{X}$	$\mathbf{U}$	**2-EXPTIME-C**	[VS85, EMJ86]

Figure 1: Summary of various modal, branching logics, their semantics and the complexity of their satisfaction problems.

formula is defined recursively by

$$\phi := \text{atom}|\neg\phi|(\phi_1 \wedge \phi_2)|Un(\phi)|Bin(\phi_1, \phi_2)$$

where 'atom' is the set of atoms, which always includes a countable set Prop of propositions, Un is any of the listed unary connectives and Bin is any of the listed binary connectives. The final cases, CTL^{+} and CTL* are two-sorted logics, with state formulas ϕ and path formulas ψ, defined by:

CTL^{+}	CTL*
$\phi = \mathsf{Prop}\|\neg\phi\|(\phi_1 \wedge \phi_2)\|\mathbf{E}\psi$	$\phi = \mathsf{Prop}\|\neg\phi\|(\phi_1 \wedge \phi_2)\|\mathbf{E}\psi$
$\psi = \mathsf{Prop}\|\neg\psi\|(\psi_1 \wedge \psi_2)\|\mathbf{X}\phi\|\mathbf{U}(\phi_1, \phi_2)$	$\psi = \phi\|\neg\psi\|(\psi_1 \wedge \psi_2)\|\mathbf{X}\phi\|\mathbf{U}(\phi_1, \phi_2)$

We use standard abbreviations: $\mathbf{F}\phi = \mathbf{U}(\phi, \top)$, and $\mathbf{G}\phi = \neg\mathbf{F}\neg\phi$, .

For **TPTL** there is more than one kind of atom. This language has *terms*

$$\tau = c|x + c$$

where x is a variable and c is a binary string. An atom is either a proposition $p \in \mathsf{Prop}$, an inequality $\tau_1 \leq \tau_2$ or a congruence $\tau_1 \equiv_d \tau_2$ where d is a non-zero binary string. The intended meaning of $\tau_1 \equiv_d \tau_2$ is that τ_1 is congruent to τ_2 modulo d.

The binary strings c, d represent natural numbers. The superscripts of $\mathbf{EU}^{\geq n}, \mathbf{EU}^{\leq n}$ and $\mathbf{EU}^{=n}$ are any natural numbers written in binary. This means that the formulas of these languages can be very short. Thus, for example $|\mathbf{EU}^{=n}(\phi_1, \phi_2)| = 1 + \lceil \log_2 n \rceil + |\phi_1| + |\phi_2|$ i.e. we charge 1 for the modality $\mathbf{EU}^{=}$, $\lceil \log_2 n \rceil$ for the binary representation of the bound n plus the lengths of formulas ϕ_1 and ϕ_2. Note that throughout the paper we do not count brackets and commas when calculating the length of a formula. We will use standard abbreviations $\Diamond\phi = \neg\Box\neg\phi$, $\blacklozenge\phi = \neg\blacksquare\neg\phi$, etc. For the semantics, state based formulas are evaluated at points w in Kripke structures $\mathcal{S} = (W, R, V)$. Here W is a set of worlds, $R \subseteq W \times W$ is an accessibility relation and $V : \mathbf{Props} \to \wp(W)$ is a valuation assigning a set $V(p)$ of worlds to each proposition p. We write $\overline{w}$ to denote a countably infinite sequence from W, and it is implicit that $\overline{w} = (w_0, w_1, \ldots) \in {}^{\omega}W$. If $(w_i, w_{i+1}) \in R$, for all $i < \omega$, then we call $\overline{w}$ and *R-path* and we may write $\rho(\overline{w})$. To evaluate a formula,

$$\begin{array}{lll}
\mathcal{S}, w \models p & \iff & w \in V(p) \\
\mathcal{S}, w \models \neg\phi & \iff & \mathcal{S}, w \not\models \phi \\
\mathcal{S}, w \models (\phi_1 \wedge \phi_2) & \iff & \mathcal{S}, w \models \phi_1 \& \mathcal{S}, w \models \phi_2 \\
\mathcal{S}, w \models \Box\phi & \iff & \forall w' \in W((w, w') \in R \to \mathcal{S}, w' \models \phi) \\
\mathcal{S}, w \models \blacksquare\phi & \iff & \forall w' \in W\ \mathcal{S}, w' \models \phi \\
\mathcal{S}, w \models \mathbf{EX}(\phi) & \iff & \exists \overline{w}\ [\rho(\overline{w}) \wedge (w = w_0) \wedge \mathcal{S}, w_1 \models \phi] \\
\mathcal{S}, w \models \mathbf{EU}(\phi_1, \phi_2) & \iff & \exists n \in \mathbb{N},\ \exists \overline{w}\ [\rho(\overline{w}) \wedge (w = w_0) \wedge \mathcal{S}, w_n \models \phi_2 \wedge \\
 & & \bigwedge_{i<n} \mathcal{S}, w_i \models \phi_1] \\
\mathcal{S}, w \models \mathbf{AU}(\phi_1, \phi_2) & \iff & \forall \overline{w}[(\rho(\overline{w}) \wedge w_0 = w) \to \\
 & & \exists n \in \mathbb{N},\ \mathcal{S}, w_n \models \phi_2 \wedge \bigwedge_{j<n} \mathcal{S}, w_j \models \phi_1] \\
\mathcal{S}, w \models \mathbf{EU}^{\leq n}(\phi_1, \phi_2) & \iff & \exists k \in \mathbb{N},\ k \leq n,\ \exists \overline{w}\ [\rho(\overline{w}) \wedge (w_0 = w) \\
 & & \wedge \mathcal{S}, w_k \models \phi_2 \wedge \bigwedge_{i<k} \mathcal{S}, w_i \models \phi_1] \\
\mathcal{S}, w \models \mathbf{AU}^{\leq n}(\phi_1, \phi_2) & \iff & \forall \overline{w}[(\rho(\overline{w}) \wedge w_0 = w) \to \\
 & & \exists k \in \mathbb{N},\ k \leq n,\ \mathcal{S}, w_n \models \phi_2 \wedge \bigwedge_{i<k} \mathcal{S}, w_i \models \phi_1] \\
\mathcal{S}, w \models \mathbf{EU}^{=n}(\phi_1, \phi_2) & \iff & \exists \overline{w}\ [\rho(\overline{w}) \wedge (w_0 = w) \wedge \\
 & & \mathcal{S}, w_n \models \phi_2 \wedge \bigwedge_{i<n} \mathcal{S}, w_i \models \phi_1] \\
\mathcal{S}, w \models \mathbf{AU}^{=n}(\phi_1, \phi_2) & \iff & \forall \overline{w}[(\rho(\overline{w}) \wedge w_0 = w) \to \\
 & & \mathcal{S}, w_n \models \phi_2 \wedge \bigwedge_{i<n} \mathcal{S}, w_i \models \phi_1] \\
\mathcal{S}, w \models \mathbf{EU}^{\geq n}(\phi_1, \phi_2) & \iff & \exists k \in \mathbb{N},\ k \geq n,\ \exists \overline{w}\ [\rho(\overline{w}) \wedge (w_0 = w) \\
 & & \wedge \mathcal{S}, w_k \models \phi_2 \wedge \bigwedge_{i<k} \mathcal{S}, w_i \models \phi_1] \\
\mathcal{S}, w \models \mathbf{AU}^{\geq n}(\phi_1, \phi_2) & \iff & \forall \overline{w}[(\rho(\overline{w}) \wedge w_0 = w) \to \\
 & & \exists k \in \mathbb{N},\ k \geq n,\ \mathcal{S}, w_n \models \phi_2 \wedge \bigwedge_{i<k} \mathcal{S}, w_i \models \phi_1]
\end{array}$$

Note that **EU** is treated as a single binary connective and $\mathbf{EU}(\phi_1, \phi_2)$ means "there is a branch and along that branch ϕ_1 is true until ϕ_2". We abbreviate $\mathbf{AX}\phi = \neg\mathbf{EX}\neg\phi$.

With path based semantics, we can still use a Kripke structure $S = (W, R, V)$ but we evaluate a formula at an R-path $\overline{w}$. For $i \in \mathbb{N}$ we write $\overline{w}^i$ for the R-path $(w_i, w_{i+1}, \ldots)$. **PLTL** formulas are evaluated by

$$\begin{array}{rll}
\mathcal{S}, \overline{w} \models p & \text{iff} & w_0 \in V(p) \\
\mathcal{S}, \overline{w} \models \neg\varphi & \text{iff} & \mathcal{S}, \overline{w} \not\models \varphi \\
\mathcal{S}, \overline{w} \models \varphi \wedge \psi & \text{iff} & \mathcal{S}, \overline{w} \models \varphi \text{ and } \mathcal{S}, \overline{w} \models \psi \\
\mathcal{S}, \overline{w} \models \mathbf{X}\varphi & \text{iff} & \mathcal{S}, \overline{w}^1 \models \varphi \\
\mathcal{S}, \overline{w} \models \phi\mathbf{U}\psi & \text{iff} & \exists j(\mathcal{S}, \overline{w}^j \models \psi \text{ and } \forall k\ (k < j \to (\mathcal{S}, \overline{w}^k \models \phi)))
\end{array}$$

The semantics of **TPTL** are slightly more complicated because we have variables and terms. **TPTL** is interpreted over linear, discrete models: $(\mathbb{N}, <, V, T)$ where $\mathbb{N}$ is the natural numbers, $<$ is the accessibility relation, V is a valuation, $V : \mathsf{Prop} \to \wp(\mathbb{N})$, and T is an assignment that maps constants to themselves (recall that constants are binary encodings of natural numbers) and variables to natural numbers, clearly T can then be extended to all terms. Given a model $\mathcal{S} = (\mathbb{N}, <, V)$, an assignment T and an index $i \in \mathbb{N}$ we evaluate formulas as follows:

$$\begin{array}{rll}
\mathcal{S}, T, i \models p & \text{iff} & i \in V(p) \\
\mathcal{S}, T, i \models \pi_1 \leq \pi_2 & \text{iff} & T(\pi_1) \leq T(\pi_2) \\
\mathcal{S}, T, i \models \pi_1 \equiv_d \pi_2 & \text{iff} & T(\pi_1) = T(\pi_2) \pmod{d} \\
\mathcal{S}, T, i \models \neg\varphi & \text{iff} & \mathcal{S}, T, i \not\models \varphi
\end{array}$$

$$\begin{array}{rcl}
\mathcal{S}, T, i \models \varphi \wedge \psi & \text{iff} & \mathcal{S}, T, i \models \varphi \text{ and } \mathcal{S}, T, i \models \psi \\
\mathcal{S}, T, i \models \mathbf{X}\varphi & \text{iff} & \mathcal{S}, T, i+1 \models \varphi \\
\mathcal{S}, T, i \models \phi \mathbf{U} \psi & \text{iff} & \exists j \ (\mathcal{S}, T, j \models \psi \text{ and } \forall k, \ (k < j \rightarrow (\mathcal{S}, T, k \models \phi))) \\
\mathcal{S}, T, i \models x.\phi & \text{iff} & \mathcal{S}, T[x := i], i \models \phi
\end{array}$$

where $T[x := i]$ denotes the assignment that agrees with the assignment T on all variables except x, and maps x to $i \in \mathbb{N}$.

Finally, CTL$^+$ and CTL* have semantics for states and paths, so the semantics is based on mutual recursion. Let $\mathcal{S} = (W, R, V)$ be a Kripke structure, let $w \in W$ and let $\overline{w}$ be an R-path. Let ϕ_1, ϕ_2 be state formulas and let ψ be a path formula. Then $\mathcal{S}, w \models \mathbf{E}\psi$ iff there is a R-path $\overline{w} = (w_0, w_1, \ldots)$ such that $w_0 = w$ and $\mathcal{S}, \overline{w} \models \psi$. And $\mathcal{S}, \overline{w} \models \mathbf{U}(\phi_1, \phi_2)$ iff there is $k \in \mathbb{N}$ such that $\mathcal{S}, w_k \models \phi_2$ and for all j with $j < k$ we have $\mathcal{S}, w_j \models \phi_1$.

It is clear from figure 1 that the more expressive the modal logic is, the higher the complexity of the satisfaction problem for the logic is, but also a logic with the same expressive power as another but with a more concise notation tends to have a higher complexity. It seems that there are several different factors which might potentially be contributing to the higher complexity. There is the use of the universal modality, the use of path based semantics and path quantifiers, there is the use of the Until connective, which is strictly more expressive than $\Box$ and there is the use of a binary encoding for the restricted Until connectives $\mathbf{U}^{=n}$ and $\mathbf{U}^{\leq n}$. It would be helpful to separate out these issues in order to analyse the source of the higher complexity. The use of binary encodings for restricting modalities and the effect of this on the complexity of the satisfaction problem is the main focus of this paper. To this end, we concentrate on state based Kripke semantics and we use only unary modalities for the logics we define.

The outline of this article is as follows. We start by defining the syntax of some new state based modal logics which use a binary encoding for superscripts to $\Box$. In section 3 we prove some easily derived upper bounds regarding the complexity of their satisfiability problems. In section 4.1 we give the definition and summarise the complexity results for Alternating Turing Machines and in section 4.2 we use our modal languages to describe the computations of Alternating Turing Machines thereby establishing lower complexity bounds for our logics. Complexity results for the modal logics introduced in this paper are proved in section 4.3. In section 5 we focus on linear and discrete structures and we prove that the satisfiability problem of a language between $\mathcal{L}(\Box^{=n}, \Box^* : n \in \mathbb{N})$ and $\mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^* : n \in \mathbb{N})$ are **EXPSPACE**-complete. In section 6 we establish the complexity of the satisfiability problem for variants of **PDL** where binary encoded iterations of programs are permitted.

2 Binary Encoded Modal Logics

In this section we introduce 'binary encoded bounded modalities' and three modal logics logics which adopt them.

Definition 1 (Languages) *Given a sequence $(*_i : i \in I)$ where each $*_i$ is a unary modal operator (e.g. $\Box, \Diamond$) we let $\mathcal{L}(*_i : i \in I)$ be the language defined by*

$$\phi = p \mid \neg\phi \mid (\phi_1 \wedge \phi_2) \mid *_i(\phi) : i \in I$$

where p is an arbitrary propositional letter.

For $n \in \mathbb{N}$, when we write $\Box^{\leq n}$ our convention is that n is written using the binary notation, so that $|\Box^{\leq n}\phi| = 1 + \lceil \log_2 n \rceil + |\phi|$. The languages we consider in this paper use the following unary modal operators: $\{\Box, \Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^*, \blacksquare : n \in \mathbb{N}\}$. We use standard abbreviations $\vee, \rightarrow, \Diamond$ etc. defined by $(\phi \vee \psi) = \neg(\neg\phi \wedge \neg\psi)$, $(\phi \rightarrow \psi) = (\neg\phi \vee \psi)$, $\Diamond\phi = \neg\Box\neg\phi$, $\Diamond^{\leq n}\phi = \neg\Box^{\leq n}\neg\phi$, $\Diamond^{=n}\phi = \neg\Box^{=n}\neg\phi$, $\Diamond^{\geq n}\phi = \neg\Box^{\geq n}\neg\phi$, $\Diamond^*\phi = \neg\Box^*\neg\phi$ and $\blacklozenge\phi = \neg \blacksquare\neg\phi$.

Definition 2 (Models) *A* structure $\mathcal{S}$ *for any of our languages is a Kripke structure* $\mathcal{S} = (W, R, V)$ *such that* W *is a non-empty set of possible worlds,* $V : Props \to \wp(W)$ *assigns a set of worlds to each proposition and* R *is a binary relation on* W*. For* $n \in \mathbb{N}$ *we define the binary relation* R^n *over* W *inductively. Let* xR^0y *hold if and only if* $x = y$*. For* $n > 0$ *let* xR^ny *hold if and only if there is* $z \in W$ *such that* $xR^{n-1}z$ *and* zRy*. The binary relations* $R^*, R^{\geq n}$ *and* $R^{\leq n}$ *are defined to be* $R^* = \bigcup_{n<\omega} R^n$*,* $R^{\geq k} = \bigcup_{k \leq n<\omega} R^n$ *and* $R^{\leq k} = \bigcup_{n \leq k} R^n$*. Formulas of these logics can be evaluated as follows:*

$$\begin{array}{rcl}
\mathcal{S}, x \models p & \textit{iff} & x \in V(p) \\
\mathcal{S}, x \models \neg\varphi & \textit{iff} & \mathcal{S}, x \not\models \varphi \\
\mathcal{S}, x \models \varphi \wedge \psi & \textit{iff} & \mathcal{S}, x \models \varphi \textit{ and } \mathcal{S}, x \models \psi \\
\mathcal{S}, x \models \Box\phi & \textit{iff} & \forall v \in W\ (xRy \to \mathcal{S}, y \models \phi) \\
\mathcal{S}, x \models \Box^{\leq n}\phi & \textit{iff} & \forall y \in W, (xR^{\leq n}y \to \mathcal{S}, y \models \phi) \\
\mathcal{S}, x \models \Box^{\geq n}\phi & \textit{iff} & \forall y \in W, (xR^{\geq n}y \to \mathcal{S}, y \models \phi) \\
\mathcal{S}, x \models \Box^{=n}\phi & \textit{iff} & \forall y \in W\ (xR^{n}y \to \mathcal{S}, y \models \phi) \\
\mathcal{S}, x \models \Box^{*}\phi & \textit{iff} & \forall y \in W,\ (xR^{*}y \to \mathcal{S}, y \models \phi) \\
\mathcal{S}, x \models \blacksquare\phi & \textit{iff} & \forall y \in W,\ \mathcal{S}, y \models \phi
\end{array}$$

If $\mathcal{S}$ *is a structure,* ϕ *is a formula and* $\mathcal{S}, x \models \phi$ *then* $(\mathcal{S}, x)$ *is a* model *of* ϕ*. If* $\mathcal{S}, x \models \phi$ *for all* $x \in W$ *and all structures* $\mathcal{S}$ *then we say that* ϕ *is* valid*. Dually, if there is some* x *in some model* $\mathcal{S}$ *such that* $\mathcal{S}, x \models \phi$ *then* ϕ *is* satisfiable*. Given any two formulas* ϕ, ψ *we write* $\phi \equiv \psi$ *if* $(\phi \leftrightarrow \psi)$ *is valid.*

The table of complexities in the abstract summarises the complexity of the satisfaction problem for these languages. The results for arbitrary Kripke frames are all established in theorems 8 and 18 and may usefully be compared to those in figure 1. The slightly unusual result is the **EXPSPACE**-complete complexity of $\mathcal{L}(\Box^{=n}, \Box^{\leq n} : n \in \mathbb{N})$-SAT which does not correspond to any of the known results in figure 1.

3 Upper Bounds

Next, we quote some well known complexity results for the satisfiability problem for some elementary modal logics. We use these results to prove natural upper bounds to the satisfiability problems for sublanguages of $\mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^*, \blacksquare : n \in \mathbb{N})$.

Proposition 3

1. *The satisfiability problem for* $\mathcal{L}(\Box)$ *is* **PSPACE**-*complete [L77].*
2. *The satisfiability problem for* $\mathcal{L}(\Box, \Box^*)$ *is* **EXPTIME**-*complete [HM92, theorem 5.1].*
3. *The satisfiability problem for* $\mathcal{L}(\Box, \blacksquare)$ *is* **EXPTIME**-*hard (by theorem 16, or the proof of [Hem96, theorem 5.1]).*

The following theorem seems to be very well known, but we had difficulty in finding a published proof.

Theorem 4 *The satisfiability problem for* $\mathcal{L}(\Box, \blacksquare)$ *is in* **EXPTIME**.

PROOF:

Let ϕ a formula of $\mathcal{L}(\Box, \blacksquare)$. For $\bigotimes \in \{\blacksquare, \blacklozenge, \Box, \Diamond\}$ we have $\bigotimes\blacksquare\psi \equiv \blacksquare\psi$ and $\bigotimes\blacklozenge\psi \equiv \blacklozenge\psi$. Hence, in linear time, we can replace ϕ by an equivalent formula ϕ', built from propositions using negations, disjunctions, conjunctions, $\Box, \Diamond$'s or $\blacksquare, \blacklozenge$s, where negations only occur immediately above propositions and there are no modal operators above any occurrence of $\blacksquare$ or $\blacklozenge$.

We check the satisfiability of ϕ' by filtration over $\{\blacksquare, \blacklozenge\}$-free subformulas of ϕ'. Let X be the set of subformulas of ϕ' not involving $\blacksquare$ or $\blacklozenge$ and let Y by the closure of X under single negations. A maximally consistent set (MCS) is a subset S of Y such that

- exactly one of ψ and $\neg\psi$ belongs to S, for each $\psi \in X$,
- if the conjunction $\alpha \wedge \beta$ is in S then so are α and β, and
- if the disjunction $\alpha \vee \beta$ is in S then so is either α or β.

If S, T are MCSs let $(S, T) \in R$ iff for all formulas $\Box\theta \in S$ we have $\theta \in T$.

In exponential time, we can replace ϕ' by an equivalent ϕ", a disjunction of conjunctions of formulas of the form $\blacksquare\alpha, \blacklozenge\beta$ or γ, where α, β, γ are subformulas of ϕ' not involving $\blacksquare$ or $\blacklozenge$. Conjunctions $\blacksquare(\alpha_1) \wedge \blacksquare(\alpha_2)$ can be replaced by the equivalent $\blacksquare(\alpha_1 \wedge \alpha_2)$. To check the consistency of ϕ" it suffice to check the consistency of each conjunctive clause, and the number of clauses is bound by an exponential function. To check the consistency of one conjunctive clause $\blacksquare\alpha \wedge \bigwedge_{i<k} \blacklozenge\beta_i \wedge \gamma$, it suffices to check that for each $i < k$, $\blacksquare\alpha \wedge \blacklozenge\beta_i$ is consistent and $\blacksquare\alpha \wedge \gamma$ is consistent (if each of these formulas is satisfiable then a model of the original clause can then by found as the disjoint union of their models) and the number of such conjuncts is again bound above by an exponential function. To check if $\blacksquare\alpha \wedge \blacklozenge\beta_i$ is consistent in exponential time, let Z be the set of all MCSs that include α and check whether β_i belongs to an MCS in Z. The case $\blacksquare \wedge \gamma$ is entirely similar.

$\Box$

Corollary 5 *The satisfiability problem for $\mathcal{L}(\Box, \blacksquare)$ is* **EXPTIME**-*complete.*

PROOF:

By proposition 3 and theorem 4. $\Box$

Definition 6 *The translation* $\mathbf{g}$ *maps formulas of $\mathcal{L}(\Box^{=n}, \Box^{\leq n}, \Box^{\geq n}, \Box^*, \blacksquare : n \in \mathbb{N})$ to formulas of $\mathcal{L}(\Box, \Box^*, \blacksquare)$. It is defined by $\mathbf{g}(p) = p$, $\mathbf{g}(\neg\phi) = \neg\mathbf{g}(\phi)$, $\mathbf{g}(\phi_1 \wedge \phi_2) = \mathbf{g}(\phi_1) \wedge \mathbf{g}(\phi_2)$, $\mathbf{g}(\Box^*\phi) = \Box^*\mathbf{g}(\phi)$, $\mathbf{g}(\blacksquare\phi) = \blacksquare\mathbf{g}(\phi)$ and:*

$$\begin{aligned}
\mathbf{g}(\Box^{=n}\phi) &= \overbrace{\Box\Box\ldots\Box}^{n}\,\mathbf{g}(\phi)\\
\mathbf{g}(\Box^{\geq n}\phi) &= \overbrace{\Box\Box\ldots\Box}^{n}\,\Box^*\mathbf{g}(\phi)\\
\mathbf{g}(\Box^{\leq n}\phi) &= \bigwedge_{i\leq n} \mathbf{g}(\Box^j\phi)
\end{aligned}$$

The use of superscripts in $\Box^{=n}, \Box^{\leq n}$ was already introduced, as an abbreviation, in Ladner's seminal work [L77] where he proved the **PSPACE**-completeness of the satisfiability problem for $\mathcal{L}(\Box)$ (i.e. the basic modal logic **K**). However, in Ladner's work there is no binary encoding of the superscripts. Our use of a binary encoding for these superscripts in the current paper makes our formulas more concise and pushes the complexity up.

Since the superscripts of $\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}$ are encoded in binary, the last three cases can produce exponentially longer formulas, but not worse.

Lemma 7 *If $\phi \in \mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^*, \blacksquare : n \in \mathbb{N})$ then $\phi \equiv \mathbf{g}(\phi)$ and $|\mathbf{g}(\phi)| \leq 2^{|\phi|^2}$.*

PROOF:

The equivalence $\phi \equiv \mathbf{g}(\phi)$ derives directly from the definition of our semantics and the definition of $\mathbf{g}$. Let $\phi \in \mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^*, \blacksquare : n \in \mathbb{N})$ and let k be the number of occurrences of $\Box^{\leq n}$ (any n) in ϕ, clearly $k < |\phi|$. We prove by induction over k that $|\mathbf{g}(\phi)| \leq 2^{(k+1)\cdot|\phi|}$. If $k = 0$ there are no occurrences of $\Box^{\leq n}$ in ϕ. When calculating $\mathbf{g}(\phi)$ we replace each occurrence of $\Box^{=n}$ (which has length $1 + \lceil \log_2 n \rceil$) by $\overbrace{\Box\Box\ldots\Box}^{n}$ (which has length n) and each occurrence of $\Box^{\geq n}$ by $\overbrace{\Box\Box\ldots\Box}^{n}\Box^*$ (which has length $n+1$). It follows that $|\mathbf{g}(\phi)| \leq 2^{|\phi|}$ when $k = 0$, proving the base case. Now let $k > 0$ and suppose for $j < k$ that if ψ has only j occurrences of operators $\Box^{\leq n}$ then $\mathbf{g}(\psi)$ has length at most $2^{(j+1)\cdot|\psi|}$. Consider a formula $\psi = \Box^{\leq n}\theta$ (some $n \in \mathbb{N}$) where θ has only $k-1$ occurrences of operators $\Box^{\leq n}$. We have $\mathbf{g}(\Box^{\leq n}\theta) = \bigwedge_{i \leq n} \mathbf{g}(\Box^{=i}\theta)$. By the induction hypothesis, $|\mathbf{g}(\Box^{\leq n}\theta)| \leq n \times (|\mathbf{g}(\Box^{=n}\theta|)) \leq n \times (2^{k\cdot|\theta|})$. But $\log_2 n \leq |\psi|$ so $n \leq 2^{|\psi|}$, hence $|\mathbf{g}(\psi)| = |\mathbf{g}(\Box^{\leq n}\theta)| \leq 2^{|\psi|} \cdot 2^{k\cdot|\psi|} \leq 2^{(k+1)\cdot|\psi|}$, proving the induction step.

For an arbitrary formula $\psi \in \mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^*, \blacksquare : n \in \mathbb{N})$, the number of occurrences of operators $\Box^{\leq n}$ in ψ is less than $|\psi|$, hence $|\mathbf{g}(\psi)| \leq 2^{|\psi|\cdot|\psi|}$, as required.
$\Box$

Theorem 8

1. *The satisfiability problem for* $\mathcal{L}(\Box^{=n}, \Box^{\leq n} : n \in \mathbb{N})$ *is in* **EXPSPACE**.
2. *The satisfiability problem for* $\mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \blacksquare : n \in \mathbb{N})$ *is in* **2-EXPTIME**.
3. *The satisfiability problem for* $L(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^* : n \in \mathbb{N})$ *is in* **2-EXPTIME**.

PROOF:

1. To determine whether $\phi \in \mathcal{L}(\Box^{=n}, \Box^{\leq n} : n \in \mathbb{N})$ is satisfiable we first compute, in exponential time, the translation $\mathbf{g}(\phi) \in \mathcal{L}(\Box)$. By lemma 7, $|\mathbf{g}(\phi)| \leq 2^{|\phi|^2}$. By proposition 3 there is a polynomial p such that the satisfiability of $\mathbf{g}(\phi)$ can be solved in space $p(|\mathbf{g}(\phi)|) \leq p(2^{|\phi|^2})$. Hence the satisfiability problem for $\mathcal{L}(\Box^{=n}, \Box^{\leq n} : n \in \mathbb{N})$ can be solved in **EXPSPACE**.
2. For $\phi \in \mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \blacksquare : n \in \mathbb{N})$ we know that $\mathbf{g}(\phi) \in \mathcal{L}(\Box, \blacksquare)$ and we can solve the satisfiability of $\mathbf{g}(\phi)$ in time $2^{p(|\mathbf{g}(\phi)|)} \leq 2^{p(2^{|\phi|^2})}$, for some polynomial p, by proposition 3. Thus the satisfiability problem of $\mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \blacksquare : n \in \mathbb{N})$ is in **2-EXPTIME**.
3. Similarly, for $\phi \in \mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n} : n \in \mathbb{N})$ we can solve $\mathbf{g}(\phi) \in \mathcal{L}(\Box, \Box^*)$ in exponential time, in terms of $|\mathbf{g}(\phi)|$, i.e. double exponential time in terms of $|\phi|$.
$\Box$

4 Lower Complexity Bounds and Alternating Turing Machines

4.1 Basic definitions and results

In this section we introduce the needed notation and definitions regarding Alternating Turing Machines of [CKS81], in order to prove tight lower complexity bounds for our logics. Our basic notation and definitions mostly come from [LL05]. An alternating Turing Machine $\mathcal{M}$ is of the form $\mathcal{M} = (Q, \Sigma, q_0, q_{acc}, q_r, \delta)$, where Q is the set of states, Σ is the alphabet which contains a blank symbol, and q_0, $q_{acc}, q_r \in Q$ the starting, the accepting and the rejecting state respectively.

Q is partitioned into $Q = Q_\exists \cup Q_\forall \cup \{q_{acc}, q_r\}$. The transition δ is a quintic relation:

$$\delta \subseteq Q \times \Sigma \times Q \times \Sigma \times \{\mathbf{L}, \mathbf{R}\}$$

We also write $(q', b, m) \in \delta(q, a)$ to denote $(q, a, q', b, x) \in \delta$ for $q, q' \in Q$, $a, b \in \Sigma$ and $x \in \{\mathbf{L}, \mathbf{R}\}$.

A configuration (w, i, q) of a Turing Machine consists of a word $w \in \Sigma^*$, an integer $i < |w|$ and a state $q \in Q$. It represents the state of the Turing Machine when w is on the tape (followed by blanks), the tape head is in position i and the machine is in state q. We write w_j for the jth character of w.

For each configuration (w, i, q) and each instruction $(p, b, \mathbf{L}) \in \delta(q, w_i)$ we can obtain a *successor configuration* $(w[i/b], i-1, p)$ where $w[i/b]$ is obtained from w by replacing w_i by b and for each $(p, b, \mathbf{R}) \in \delta(q, w_i)$ we obtain another successor configuration $(w[i/b], i+1, p)$. No other configurations are successor configurations to (w, i, q). An initial configuration has the form $(w, 0, q_0)$ for some $w \in \Sigma^*$, where q_0 is the start state.

A finite computation path of an **ATM** $\mathcal{M}$ on a word w is a finite sequence of configurations $c_0, c_1, \ldots, c_n$ such that $c_0 = (w, 0, q_0)$, if $c_n = (w', j, q)$ then $q \in \{q_{acc}, q_r\}$, and for $i < n$ c_{i+1} is a successor configuration to c_i and $c_i = (w', j, q) \to q \notin \{q_{acc}, q_r\}$. An infinite computation path is an infinite sequence of configurations $c_0, c_1, \ldots$ with the same properties as finite computation paths, except that none of the states occurring in the configurations belongs to $\{q_{acc}, q_r\}$. All the **ATM**s considered in this paper have the property that all computation paths are finite. Hence, we define the acceptance for an **ATM** with finite computation paths, the general case was given in [CKS81].

Definition 9 *Let $\mathcal{M}$ be an* **ATM** *with finite computation paths. The acceptance of a configuration (w, i, q) depends on the sort of the state q and:*

- *If the state q is q_{acc} then the configuration is accepting (so if $q = q_r$ then the configuration is not accepting).*
- *If $q \in Q_\exists$, then the configuration is accepting iff at least one of its successors is accepting.*
- *If $q \in Q_\forall$, then the configuration is accepting iff all of its successors are accepting.*

Since all computation paths are finite this definition is well founded. A Turing Machine accepts input w if the configuration $(w, 0, q_0)$ is accepting. We write $\mathcal{L}(\mathcal{M})$ for the language accepted by $\mathcal{M}$, that is, $\mathcal{L}(\mathcal{M}) = \{w \in \Sigma^* |\ \ \mathcal{M}\ \ \text{accepts}\ \ w\}$. We write **APTIME** for the class of problems that can be solved by an alternating Turing Machine in polynomial time, etc. The following proposition was proved in [CKS81, corollary 3.6].

Proposition 10

$$\begin{aligned} \mathbf{APTIME} &= \mathbf{PSPACE} \\ \mathbf{APSPACE} &= \mathbf{EXPTIME} \\ \mathbf{AEXPTIME} &= \mathbf{EXPSPACE} \end{aligned}$$

Definition 11 *Given an* **ATM** *$\mathcal{M}$ with finite computation paths, an* acceptance tree *$\mathcal{T} = (T, F, \lambda)$ consists of a set T of nodes, a binary relation F over T defining a tree with root $r \in T$, and a function λ mapping nodes to acceptance configurations, such that:*

- *For the root node $r \in T$ we have $\lambda(r) = (w, 0, q_0)$, for some $w \in \Sigma^*$.*
- *If $(\tau, \tau') \in F$ then $\lambda(\tau')$ is a successor configuration to $\lambda(\tau)$.*
- *For all $\tau \in T$, it is not the case that $\lambda(\tau) = (w, i, q_r)$.*
- *For all $\tau \in T$, if $\lambda(\tau) = (w, i, q)$ where $q \in Q_\exists$ then there is $\tau' \in \mathcal{T}$ with $(\tau, \tau') \in F$.*

- *For all $\tau \in T$, if $\lambda(\tau) = (w, i, q)$ where $q \in Q_\forall$ and (w', i', q') is any successor configuration to (w, i, q) then there is $\tau' \in T$ with $(\tau, \tau') \in F$ and $\lambda(\tau') = (w', i', q')$.*

Lemma 12 ([LL05]) *Let $\mathcal{M}$ be an* **ATM** *with only finite computation paths. Then there exists an acceptance tree $\mathcal{T}$ of $\mathcal{M}$ on w iff $\mathcal{M}$ accepts w.*

4.2 Reducing Alternating Turing Machines to Modal Languages

In this section we define a polynomial time reduction of the inputs to an exponential space **ATM** $\mathcal{M}$ to formulas of $\mathcal{L}(\Box^{=n}, \boxtimes : n \in \mathbb{N})$, for various modalities $\boxtimes \in \{\blacksquare, \Box^*, \Box^{\leq n} : n \in \mathbb{N}\}$. This will allow us to prove hardness results for various modal languages $\mathcal{L}(\Box^{=n}, \boxtimes : n \in \mathbb{N})$. For example, we will deduce that the satisfiability problem for $\mathcal{L}(\Box^{=n}, \blacksquare : n \in \mathbb{N})$ is **2-EXPTIME**-hard. For the case where $\mathcal{M}$ is known to run in **EXPTIME** we will be able to prove that $\mathcal{L}(\Box^{=n}, \Box^{\leq n} : n \in \mathbb{N})$-SAT is **EXPSPACE**-hard. Further restrictions on the computation of $\mathcal{M}$ yield lower complexity bounds on a number of other modal logics.

Let $\mathcal{M} = (Q, \Sigma, q_0, q_{acc}, q_r, \delta)$ be an an exponential space bounded **ATM** and let p be a polynomial such that the space used by $\mathcal{M}$ on any input of size n is bound by $2^{p(n)}$. Let $w = w_0, \ldots, w_{n-1} \in \Sigma^*$ be an input for $\mathcal{M}$. In the following, all configurations will have length $m = 2^{p(n)}$ (configurations may include blank symbols). We will define a formula $\phi_{\mathcal{M},w}(\boxtimes, m)$, where $\boxtimes \in \{\blacksquare, \Box^*, \Box^{\leq n} : n \in \mathbb{N}\}$ and m is the configuration size. We will substitute different operators for $\boxtimes$, depending on the particular complexity result we wish to obtain.

The models of the formula $\phi_{\mathcal{M},w}(\boxtimes, m)$ (if any) will represent acceptance trees. Each world in such a model will represent a cell in one of the accepting configurations. The position of the cell within the configuration will be determined by the truth of the space propositions s_i ($i < p(n)$ at that world. If the position of the cell is strictly less than $m - 1$ then the only accessible world will represent the next cell in the same configuration. If the position of the cell is $m - 1$ then the successor worlds will represent the initial cells in successor configurations.

We will use the following propositions telling us everything about the **ATM**.

$$Q \cup \Sigma \cup \{s_i : i < p(n)\} \cup \{p_{first}, p_{seen}\}$$

The intended interpretations are:

- $q \in Q$ means that the tape head is at the current cell and $\mathcal{M}$ is in state q. We will use formula $h = \bigvee_{q \in Q} q$, which means that the tape head is on the current cell.
- $a \in \Sigma$ means that a is the symbol in the current cell.
- $s_{p(n)-1}, \ldots, s_0$ represent a counter in binary code for counting the $m = 2^{p(n)}$ tape cells of a configuration. The counter value will be 0 in the leftmost and $m - 1$ in the rightmost tape cell, for instance.
- p_{first} should be true at each cell of the initial configuration, but need not be true elsewhere.
- p_{seen} should be true at each cell of any configuration to the right of the tape head position, but need not be true to the left or at the tape head position.

We need formulas expressing particular space counter values. Let $k < m$ and let k_i be the ith bit of a binary representation of k, for $i < p(n)$, so $k = \sum_{i<p(n)} k_i \times 2^i$.

$$\chi_{\mathbf{S}=k} = \bigwedge_{i<p(n),\ k_i=1} s_i \wedge \bigwedge_{i<p(n),\ k_i=0} \neg s_i$$

Observe that $\chi_{\mathbf{S}=0}$, $\chi_{\mathbf{S}=m-1}$ express that the space counters represent the leftmost and rightmost cells of a configuration, respectively. Let

$$\chi_{\mathbf{S}\leq n} = \bigvee_{k \leq n} \chi_{\mathbf{S}=k}$$

Note that the length of $\chi_{\mathbf{S}\leq n}$ is bound by a polynomial $(O(p(n)^2))$ in terms of $|w| = n$. Let

$$\mathbf{inc}(s,i) = (\neg s_i \wedge \bigwedge_{j<i} s_j) \to (\Box(s_i \wedge \bigwedge_{j<i} \neg s_j) \wedge \bigwedge_{i<j<p(n)} ((s_j \to \Box s_j) \wedge (\neg s_j \to \Box \neg s_j))$$

For $i < p(n)$, $\mathbf{inc}(s,i)$ says that if s_i is true but for $j < i$, s_j is false then at any successor the last $i+1$ space counters flip, but the other space counters are unchanged. Let

$$\mathbf{inc}(s) = \bigwedge_{i<p(n)} \mathbf{inc}(s,i) \ \wedge \ (\bigwedge_{i<p(n)} s_i \to \Box \bigwedge_{i<p(n)} \neg s_i) \tag{1}$$

$\mathbf{inc}(s)$ says that the space counters at a successor cell represent one more (modulo m) than the space counters at the current cell.

Now we define the formula $\phi_{\mathcal{M},w}(\boxtimes, m) \in \mathcal{L}(\Box^{=n}, \boxtimes : n \in \mathbb{N})$:

$$\begin{aligned}\phi_{\mathcal{M},w}(\boxtimes, m) = \ & \phi_{start}(w) \wedge \\ & \boxtimes(\phi_{first} \wedge \phi_{seen} \wedge \mathbf{inc}(s) \wedge \phi_{head} \wedge \phi_{conf} \wedge \phi_{acc} \wedge \phi_\delta \wedge \phi_\forall \wedge \phi_\exists)\end{aligned} \tag{2}$$

where

$$\phi_{start}(w) \ = \ \chi_{\mathbf{S}=0} \wedge q_0 \wedge p_{first} \wedge w_0 \wedge \Box(w_1 \wedge \Box(w_2 \wedge \ldots \wedge \Box(w_{n-1} \wedge \Box \boxtimes (p_{first} \to \mathsf{blank}) \ldots)$$

$$\begin{aligned}
\phi_{first} &= (p_{first} \to \Box(p_{first} \vee \chi_{\mathbf{S}=0})) \\
\phi_{seen} &= (h \vee p_{seen}) \to \Box(\chi_{\mathbf{S}=0} \vee p_{seen}) \\
\phi_{conf} &= (\bigvee_{a\in\Sigma} a) \ \wedge \bigwedge_{a,b\in\Sigma, b\neq a} \neg(a \wedge b) \wedge \bigwedge_{q,q'\in Q, q\neq q'} \neg(q \wedge q') \\
\phi_{head} &= \neg(p_{seen} \wedge h) \\
\phi_{acc} &= \neg q_r \\
\phi_\delta &= \bigwedge_{a\in\Sigma} [((\neg h \wedge a) \to \Box^{=m} a) \ \wedge \\
&\qquad \bigwedge_{q\in Q,\ a\in\Sigma} ((q \wedge a) \to \Box^{=m-1}(\bigvee_{(q',a',L)\in\delta(q,a)} (q' \wedge \Box a') \vee \bigvee_{(q',a',R)\in\delta(q,a)} \Box(a' \wedge \Box q'))] \\
\phi_\forall &= \bigwedge_{q\in Q_\forall,\ a\in\Sigma} ((q \wedge a) \to \bigwedge_{(q',a',L)\in\delta(q,a)} \Diamond^{=m-1}(q' \wedge \Diamond a') \wedge \bigwedge_{(q',a',R)\in\delta(q,a)} \Diamond^{=m}(a' \wedge \Diamond q')) \\
\phi_\exists &= \bigwedge_{q\in Q_\exists,\ a\in\Sigma} ((q \wedge a) \to \bigvee_{(q',a',L)\in\delta(q,a)} \Diamond^{=m-1}(q' \wedge \Diamond a') \vee \bigvee_{(q',a',R)\in\delta(q,a)} \Diamond^{=m}(a' \wedge \Diamond q'))
\end{aligned}$$

The first part ϕ_{start} says that we start with space counter zero and the first n cells on any path starting from the root defines the non-blank part of the starting configuration $(w, 0, q_0)$. The formula ϕ_{first} ensures that p_{first} holds throughout the initial configuration, so ϕ_{start} in conjunction with ϕ_{first} ensures that after the nth cell there are only blanks on the tape in the initial configuration. The formula ϕ_{seen} ensures that if h is true at a cell of a configuration then p_{seen} is true at each cell to the right within that configuration. In conjunction with ϕ_{head} it ensures that the head of the Turing Machine cannot be in more than one cell of a configuration. $\mathbf{inc}(s)$ ensures that the space counters correctly represent the position of a cell. It expresses that the value of the space counter increases by one (modulo m) when you move to a successor. ϕ_{conf} expresses the fact that at each world there is exactly one tape symbol and at most one state. ϕ_{acc} makes sure that no computation path ends in the rejecting state. ϕ_δ requires that any successor configuration can be obtained from the current configuration by some δ-transition. $\phi_\forall$ ensures that all successor configurations to a universal configuration can be found. And finally, $\phi_\exists$ says that some successor configuration to any existential configuration can be found.

Theorem 13 *Let $\mathcal{M}$ be an exponential space* **ATM**, *then $\phi_{\mathcal{M},w}(\blacksquare, m)$ is satisfiable iff $w \in \mathcal{L}(\mathcal{M})$.*

PROOF:

For the right to left implication suppose $w \in \mathcal{L}(\mathcal{M})$ and let $\mathcal{T} = (T, F, \lambda)$ be an acceptance computation tree. Let $W = T \times m$ and

$$R = \{((\tau, i), (\tau, i+1)) : \tau \in \mathcal{T},\ i < m-1\} \cup \{((\tau, m-1), (\tau', 0)) : (\tau, \tau') \in F\}$$

Let the valuation V be defined by

$$\begin{array}{rcl} V(p_{first}) & = & \{(\tau_0, i) : i < m\} \text{ where } \tau_0 \text{ is the initial configuration} \\ V(p_{seen}) & = & \{(\tau, j) : \tau = (w, i, q) \in T,\ j > i\} \\ V(q) & = & \{(\tau, i) : \tau \in T,\ \lambda(\tau) = (w, i, q) \text{ some } w\} \\ V(a) & = & \{(\tau, j) : \lambda(\tau) = (w, i, q),\ w_j = a\} \\ V(s_i) & = & \{(\tau, j) : \tau \in \mathcal{T}, \text{ bit } i \text{ of binary code for } j \text{ is } 1\} \end{array}$$

Now it is just a routine exercise to check that $(W, R, V) \models \phi_{\mathcal{M},w}(\blacksquare, m)$, so $\phi_{\mathcal{M},w}(\blacksquare, m)$ is satisfiable.

For the left to right implication, suppose $\mathcal{S}, r_0 \models \phi_{\mathcal{M},w}(\blacksquare, m)$ for some structure $\mathcal{S} = (W, R, V)$ and some $r_0 \in W$. Since $\mathcal{S}, r_0 \models \blacksquare\mathbf{inc}(s)$, for all $y \in W$ we can define the position

$$pos(y) = \sum_{i < p(n), \mathcal{S}, y \models s_i} 2^i$$

Since $\mathcal{S} \models \blacksquare\mathbf{inc}(s)$, for any $x, y \in W$ if xRy then either $pos(x) = m-1,\ pos(y) = 0$ or $pos(y) = pos(x) + 1$. Define $T \subseteq W^m$ by

$$T = \left\{ \begin{array}{lcl} (x_0, x_1, \ldots, x_{m-1}) & : & pos(x_i) = i,\ x_i R x_{i+1} \text{ for } i < m-1 \\ & & \wedge \exists i < m,\ \mathcal{M}, x_i \models h \end{array} \right\} \quad (3)$$

We write $\overline{x}$ for $(x_0, \ldots, x_{m-1})$. Define $F \subseteq T \times T$ by letting $(\overline{x}, \overline{y}) \in F$ iff $x_{m-1} R y_0$. For each $\overline{x} \in T$ and each $i < m$, since $\mathcal{S} \models \blacksquare\phi_{conf}$ there is a unique symbol $a \in \Sigma$ such that $\mathcal{S}, x_i \models a$ and there is at most one state symbol $q \in Q$ such that $\mathcal{S}, x_i \models q$. Since $\mathcal{S} \models \blacksquare\phi_{head}$ and by definition of T, there is exactly one $i < m$ such that $\mathcal{S}, x_i \models h$. It follows that each $\overline{y} \in T$ defines a unique configuration (w, i, q) where $\mathcal{S}, y_j \models w_j$, for $j < m$, and $\mathcal{S}, y_i \models q$. We denote this configuration as $conf(\overline{y})$.

We claim that $\mathcal{T} = (T, F, conf)$ is an acceptance tree for $\mathcal{M}, w$. For any $\overline{x} \in T$ with $x_0 = r_0$, since $\mathcal{S}, r_0 \models \phi_{start}(w) \wedge \blacksquare\phi_{first}$ we have $conf(\overline{x}) = (w, 0, q_0)$, the initial configuration. Since $(W, R, V) \models \blacksquare\phi_\delta$, if $(\overline{x}, \overline{y}) \in F$ then $conf(\overline{y})$ is a successor configuration to $conf(\overline{x})$. Since $(W, R, V) \models \blacksquare\phi_\exists$, if $\overline{x} \in T$, $conf(\overline{x}) = (w, i, q)$ and $q \in Q_\exists$ then there is $\overline{y} \in T$ such that $(\overline{x}, \overline{y}) \in F$. Finally, since $(W, R, V) \models \blacksquare\phi_\forall$, if $\overline{x} \in T$ $conf(\overline{x}) = (w, i, q)$ where $q \in Q_\forall$, and γ is any successor configuration to $conf(\overline{x})$ then there is $\overline{y} \in T$ such that $(\overline{x}, \overline{y}) \in F$ and $conf(\overline{y}) = \gamma$.

Hence (T, F) is an acceptance tree. By lemma 12, $\mathcal{M}$ accepts w. $\Box$

THEOREM 14 *$\phi_{\mathcal{M},w}(\blacksquare, m)$ is satisfiable if and only if $\phi_{\mathcal{M},w}(\Box^*, m)$ is satisfiable.*

PROOF:

The formula $\blacksquare\psi \to \Box^*\psi$ is valid, so any model of $\phi_{\mathcal{M},w}(\blacksquare, m)$ is also a model of $\phi_{(\mathcal{M},w)}(\Box^*, m)$, proving the left to right implication. Conversely, suppose $(W, R, V), r_0 \models \phi_{\mathcal{M},w}(\Box^*, m)$. Let $W_0 \subseteq W$ consist of all those worlds which can be reached from r_0 by a chain of R steps and let R_0, V_0 be obtained from R, V by restricting to this set. Since the deletion of inaccessible worlds cannot affect the truth of $\mathcal{L}(\Box^{=n}, \Box^{\leq n}, \Box^* : n \in \mathbb{N})$ formulas, we have $(W_0, R_0, V_0), r_0 \models \phi_{\mathcal{M},w}(\Box^*, m)$. Since all worlds in W_0 are now accessible from r_0, we have $(W_0, R_0, V_0), r_0 \models \Box^*\psi \to \blacksquare\psi$ for all $\psi \in \mathcal{L}(\Box^{=n}, \Box^{\leq n} : n \in \mathbb{N})$, hence $(W_0, R_0, V_0), r_0 \models \phi_{\mathcal{M},w}(\blacksquare, m)$. This proves the right to left implication. $\Box$

Now we consider the case where the run-time of $\mathcal{M}$ is bounded by $m = 2^{p(n)}$ for some polynomial p (we lose no generality by taking m to be a bound on both the space and the time used by $\mathcal{M}$).

Theorem 15 *Let $\mathcal{M}$ be an exponential time* **ATM** *and let $w \in \Sigma^*$. $\phi_{\mathcal{M},w}(\Box^{\leq m^2}, m)$ is satisfiable iff $w \in \mathcal{L}(\mathcal{M})$.*

PROOF:

By theorem 13 we know that $\phi_{\mathcal{M},w}(\blacksquare, m)$ is satisfiable iff $w \in \mathcal{L}(\mathcal{M})$. Any model for $\phi_{\mathcal{M},w}(\blacksquare, m)$ is also a model for $\phi_{\mathcal{M},w}(\Box^{\leq m}, m)$, since $\blacksquare\psi \rightarrow \Box^{\leq k}\psi$ is valid, for any $k \in \mathbb{N}$, so if $\phi_{\mathcal{M},w}(\blacksquare, m)$ is satisfiable then so is $\phi_{\mathcal{M},w}(\Box^{\leq m^2}, m)$. Conversely, let $\mathcal{S}, r_0 \models \phi_{\mathcal{M},w}(\Box^{\leq m^2}, m)$ be any model of $\phi_{\mathcal{M},w}(\Box^{\leq m^2}, m)$, say $\mathcal{S} = (W, R, V)$. As before, since $\phi_{\mathcal{M},w}(\Box^{\leq m^2}, m)$ does not involve $\blacksquare$, we can assume that for each $x \in W$ there is an R-path from r_0 to x. Because the run time of $\mathcal{M}$ is bounded by $m = 2^{p(n)}$, the maximum possible length of a computation path for w is m. Hence the maximum length of an R-path from r_0 is at most m^2 (m configurations, each of length m). Hence $\mathcal{S}, r_0 \models \Box^{\leq m^2}\psi \rightarrow \blacksquare\psi$, for all $\psi \in \mathcal{L}(\Box^{=n}, \Box^{\leq n} : n \in \mathbb{N})$, and therefore $\mathcal{S}, r_0 \models \phi_{\mathcal{M},w}(\blacksquare, m)$. We have now shown that $\phi_{\mathcal{M},w}(\Box^{\leq m^2}, m)$ is satisfiable iff $\phi_{\mathcal{M},w}(\blacksquare, m)$ is satisfiable. This proves the theorem. $\Box$

We continue by supposing that $\mathcal{M}$ runs using only polynomial space.

Theorem 16 *$\mathcal{L}(\blacksquare, \Box)$ is* **EXPTIME**-*hard.*

PROOF:

By proposition 10, **APSPACE=EXPTIME**. Let $\mathcal{M}$ be a polynomial space **ATM**, let q be a polynomial and suppose a computation of $\mathcal{M}$ on an input of size n uses at most $q(n)$ space. We reduce the language accepted by $\mathcal{M}$ to the satisfiability problem for $\mathcal{L}(\blacksquare, \Box)$. Let $w \in \Sigma^*$ have size n. We map w to $\mathbf{g}(\phi_{\mathcal{M},w}(\blacksquare, q(n))) \in \mathcal{L}(\blacksquare, \Box)$. Consult the definition of $\phi_{\mathcal{M},w}(\blacksquare, q(n))$ in (2). Each occurrence of $\Box^{=k}$ occurring in $\phi_{\mathcal{M},w}(\blacksquare, q(n))$ has superscript $k \leq q(n)$. Hence the translation $\mathbf{g}$ given in definition 6 runs in polynomial time. By theorem 13, $\phi_{\mathcal{M},w}(\blacksquare, q(n))$ is satisfiable iff $w \in \mathcal{L}(\mathcal{M})$. By lemma 7, this holds iff $\mathbf{g}(\phi_{\mathcal{M},w}(\blacksquare, q(n)))$ is satisfiable. Hence the reduction is correct and can be computed in polynomial time. $\Box$

If $\mathcal{M}$ is known to run in **PTIME** (say time $p(n)$) we can modify the reduction of theorem 16 so as to eliminate all occurrences of $\blacksquare$ and produce a formula in $\mathcal{L}(\Box)$.

Theorem 17 *Let $\mathcal{M}$ be an polynomial time* **ATM***, let p be a polynomial bound on the run-time of $\mathcal{M}$, and let $w \in \Sigma^*$. Let $m = p(|w|)$ — this is a bound on both the time and space of any $\mathcal{M}$-computation. Then $\mathbf{g}(\phi_{\mathcal{M},w}(\Box^{\leq m^2}, m)) \in \mathcal{L}(\Box)$ can be computed in polynomial time and it is satisfiable iff $w \in \mathcal{L}(\mathcal{M})$.*

PROOF:

Similar. $\Box$

This shows that the satisfiability problem for $\mathcal{L}(\Box)$ is **PSPACE**-hard. Looking back to figure 1, we have now gone in a complete circle by reproving Ladner's result that the satisfiability problem for the basic modal logic **K** is **PSPACE**-complete.

4.3 Complexity Results for Branching Modal Logics

Theorem 18

1. *If* $\mathcal{L}(\Box^{=n}, \blacksquare : n \in \mathbb{N}) \subseteq \mathcal{L} \subseteq \mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \blacksquare : n \in \mathbb{N})$ *then* $\mathcal{L}$*-SAT is* **2-EXPTIME**-*complete.*

2. *If* $\mathcal{L}(\Box^{=n}, \Box^{*} : n \in \mathbb{N}) \subseteq \mathcal{L} \subseteq \mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^{*} : n \in \mathbb{N})$ *then* $\mathcal{L}$*-SAT is* **2-EXPTIME**-*complete.*

3. $\mathcal{L}(\Box^{=n}, \Box^{\leq n} : n \in \mathbb{N})$*-SAT is* **EXPSPACE**-*complete.*

PROOF:

The upper complexity bounds for all cases are established in theorem 8. We now prove the lower bounds.

1. To prove that $\mathcal{L}(\Box^{=n}, \blacksquare : n \in \mathbb{N})$-SAT is **2-EXPTIME**-hard, let A be an arbitrary **2-EXPTIME** decision problem. By proposition 10, A can be solved by an **ATM** $\mathcal{M}$ using exponential space. Let $w \in \Sigma^*$ be an instance of A. In polynomial time we can compute $\phi_{\mathcal{M},w}(\blacksquare, m)$, where m is an exponential bound on the space used by $\mathcal{M}$ on input w. By theorem 13, the map $w \mapsto \phi_{\mathcal{M},w}(\blacksquare, m)$ is a polynomial time reduction of A to $\mathcal{L}(\Box^{=n}, \blacksquare : n \in \mathbb{N})$-SAT. Hence this problem is **2-EXPTIME**-hard.
2. By theorem 14, $\phi_{\mathcal{M},w}(\Box^*, m)$ is satisfiable iff $\phi_{\mathcal{M},w}(\blacksquare, m)$ is satisfiable. Hence the map $w \mapsto \phi_{\mathcal{M},w}(\Box^*, m)$ is a polynomial time reduction, so $\mathcal{L}(\Box^{=n}, \Box^* : n \in \mathbb{N})$-SAT is **2-EXPTIME**-hard.
3. If A is a decision problem in **EXPSPACE** then by proposition 10, there is an exponential time **ATM** $\mathcal{M}$ that solves A. By theorem 15 the map $w \mapsto \phi_{\mathcal{M},w}(\Box^{\leq m^2}, m)$ is a polynomial time reduction of A to $\mathcal{L}(\Box^{=n}, \Box^{\leq n} : n \in \mathbb{N})$-SAT, so this problem is **EXPSPACE**-hard.

$\Box$

5 Linear Discrete Time

So far we have been considering unrestricted Kripke semantics. We now turn our attention to linear discrete frames. We show that the satisfiability problem for languages between $\mathcal{L}(\Box^{=n}, \Box^* : n \in \mathbb{N})$ and $\mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^* : n \in \mathbb{N})$ on the linear frame $(\mathbb{N}, <)$ are **EXPSPACE**-complete. To prove that, we use the same approach as we did in the last section but work this time with Deterministic Turing Machines (**DTM**s). Note that **DTM**s can be considered as a special case of Alternating Turing Machines (**ATM**s): a deterministic Turing Machine is an **ATM** where all its states are existential and the transition function δ is single valued. The semantics of $\mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^* : n \in \mathbb{N})$ on $(\mathbb{N}, <)$ follow:

Definition 19 *Formulas of* $\mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^* : n \in \mathbb{N})$ *are interpreted with respect to linear discrete models:* $(\mathbb{N}, <, V)$*. Given* $\mathcal{S} = (\mathbb{N}, <, V)$ *and an index* $i \in \mathbb{N}$ *we evaluate formulas as follows:*

$$\begin{array}{rcl}
\mathcal{S}, i \models p & \textit{iff} & i \in V(p) \\
\mathcal{S}, i \models \neg\varphi & \textit{iff} & \mathcal{S}, i \not\models \varphi \\
\mathcal{S}, i \models \varphi \wedge \psi & \textit{iff} & \mathcal{S}, i \models \varphi \textit{ and } \mathcal{S}, i \models \psi \\
\mathcal{S}, i \models \Box^{=n}\phi & \textit{iff} & \mathcal{S}, i + n \models \phi \\
\mathcal{S}, i \models \Box^{\geq n}\phi & \textit{iff} & \forall k,\ i + n \leq k \ \rightarrow\ (\mathcal{S}, k \models \phi)
\end{array}$$

$$\begin{array}{rcl}\mathcal{S}, i \models \Box^{\leq n}\phi & \textit{iff} & \forall k,\ i \leq k \leq i+n \ \rightarrow\ (\mathcal{S}, k \models \phi)\\ \mathcal{S}, i \models \Box^{*}\phi & \textit{iff} & \forall k \geq i\ (\mathcal{S}, k \models \phi)\end{array}$$

Definition 20 *The translation* $\mathbf{f}$ *maps formulas of* $\mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^{*} : n \in \mathbb{N})$ *to formulas of* **TPTL** *and is defined by* $\mathbf{f}(p) = p$, $\mathbf{f}(\neg\phi) = \neg\mathbf{f}(\phi)$, $\mathbf{f}(\phi_1 \wedge \phi_2) = \mathbf{f}(\phi_1) \wedge \mathbf{f}(\phi_2)$ *and*

$$\begin{array}{rcl}\mathbf{f}(\Box^{=n}\phi) & = & x.\mathbf{G}(y.(y = x+n \rightarrow \mathbf{f}(\phi)))\\ \mathbf{f}(\Box^{\geq n}\phi) & = & x.\mathbf{G}(y.(y \geq x+n \rightarrow \mathbf{f}(\phi)))\\ \mathbf{f}(\Box^{\leq n}\phi) & = & x.\mathbf{G}(y.(y \leq x+n \rightarrow \mathbf{f}(\phi)))\\ \mathbf{f}(\Box^{*}\phi) & = & \mathbf{G}\mathbf{f}(\phi)\end{array}$$

where x, y *are new variables, not occurring in* $\mathbf{f}(\phi)$.

Proposition 21 ([AH94]) *The satisfiability problem for formulas of* **TPTL** *formulas over* $(\mathbb{N}, <)$ *is* **EXPSPACE**-*complete.*

Lemma 22 *Let* $\phi \in \mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^{*} : n \in \mathbb{N})$,

1. $|\mathbf{f}(\phi)| \leq 8 \cdot |\phi|$.
2. $(\mathbb{N}, <) \models \mathbf{f}(\phi) \leftrightarrow \phi$.

PROOF:

Substrings $\Box^{\geq n}, \Box^{\leq n}, \Box^{=n}$ of ϕ of length $1 + \lceil \log_2 n \rceil$ are replaced by at most $8 + \lceil \log_2 n \rceil$ characters in $\mathbf{f}(\phi)$, so at most 7 additional characters for each of these operators. We count any variable $x.$ as single character and we charge 3 for $\mathbf{G} = \neg\mathbf{F}\neg$. The equivalence of ϕ and $\mathbf{f}(\phi)$ over $(\mathbb{N}, <)$ is direct from the definition of the semantics. $\Box$

Theorem 23 *The satisfiability problem for formulas of* $\mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^{*} : n \in \mathbb{N})$ *can be solved in* **EXPSPACE**.

PROOF:

To determine whether $\phi \in \mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^{*} : n \in \mathbb{N})$ is satisfiable in $(\mathbb{N}, <)$ we first compute, in polynomial time, the translation $\mathbf{f}(\phi) \in$ **TPTL**. By lemma 22, $|\mathbf{f}(\phi)| \leq 8 \cdot |\phi|$. By proposition 21 there is a polynomial p such that the satisfiability of $\mathbf{f}(\phi)$ can be solved in space $2^{p(|\mathbf{f}(\phi)|)} \leq 2^{p(8 \cdot |\phi|)}$. Hence the satisfiability problem for $\mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^{*} : n \in \mathbb{N})$ over $(\mathbb{N}, <)$ can be solved in **EXPSPACE**. $\Box$

For the lower bound, we take an exponential space **DTM** $\mathcal{M}$ and an input w to the machine. In order to avoid prematurely terminating runs, it is convenient to assume that the transition function $\delta : Q \times \Sigma \rightarrow Q \times \Sigma \times \{\mathbf{L}, \mathbf{R}\}$ of $\mathcal{M}$ is always defined (if q is a final state and $a \in \Sigma$ then we may let $\delta(q, a) = (q, a, \mathbf{L})$ except at a marker $\triangleright$ for the leftmost cell where $\delta(q, \triangleright) = (q, \triangleright, \mathbf{R})$, so $\mathcal{M}$ stays in this final state forever). Note that although runs of $\mathcal{M}$ are infinite, computation paths are still finite, according to the definition we gave just before definition 9, because on any infinite run, $\mathcal{M}$ enters a final state after finitely many steps. In polynomial time we compute the formula $\phi_{\mathcal{M},w}(\Box^{*}, m)$ of (2).

Theorem 24 $\phi_{\mathcal{M},w}(\Box^{*}, m)$ *is satisfiable over* $(\mathbb{N}, <)$ *iff* $w \in \mathcal{L}(\mathcal{M})$.

PROOF:

If $\phi_{\mathcal{M},w}(\Box^*, m)$ is satisfiable in $(\mathbb{N}, <)$ then of course it is satisfiable. So by theorems 13 and 14, $\mathcal{M}$ has an accepting run on input w (in fact, since $\mathcal{M}$ is deterministic, there is a unique run of $\mathcal{M}$ on w and $\mathcal{M}$ accepts w).

Conversely, suppose $\mathcal{M}$ accepts w. By theorems 13 and 14 there is a structure $\mathcal{S} = (W, R, V)$ and a world $v \in W$ such that $\mathcal{S}, v \models \phi_{\mathcal{M},w}(\Box^*, m)$, though this model may not be linear. As in the proof of theorem 13 we can define the position $pos(x)$ of each world $w \in \mathcal{S}$, we can define $T \subseteq W^m$ using (3) and we can make T into the base of an acceptance tree for $\mathcal{M}$. Since $\mathcal{M}$ is deterministic, the conjuncts $\phi_\forall$ and $\phi_\exists$ of $\phi_{\mathcal{M},w}(\Box^*, m)$ demand at most only a single successor configuration for each configuration. We can now restrict $\mathcal{S}$ to $\mathcal{S}^-$, say, by starting with m worlds representing the initial configuration and then appending only a single successor configuration to each configuration, and $\phi_{\mathcal{M},w}(\Box^*, m)$ will still be true in $\mathcal{S}^-$. This restricted structure $\mathcal{S}^-$ is clearly isomorphic to a structure $(\mathbb{N}, <, V)$.

Hence $\mathcal{M}$ accepts w iff $\phi_{\mathcal{M},w}(\Box^*, m)$ is satisfiable iff $\phi_{\mathcal{M},w}(\Box^*, m)$ is satisfiable over $(\mathbb{N}, <)$. $\Box$

Theorem 25 *The satisfiability problems for any language between $\mathcal{L}(\Box^{=n}, \Box^* : n \in \mathbb{N})$ and $\mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^* : n \in \mathbb{N})$ over $(\mathbb{N}, <)$ is* **EXPSPACE***-complete.*

PROOF:

We have already seen in theorem 23 that satisfiability of $\mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^* : n \in \mathbb{N})$ over $(\mathbb{N}, <)$ is in **EXPSPACE**. We prove that satisfiability of $\mathcal{L}(\Box^{=n}, \Box^* : n \in \mathbb{N})$ over $(\mathbb{N}, <)$ is **EXPSPACE**-hard. Let A be an arbitrary **EXPSPACE** decision problem, that is, A can be solved by a **DTM** $\mathcal{M}$ using exponential space. Let w be an instance of A, coded into the alphabet of $\mathcal{M}$. In polynomial time we can compute $\phi_{\mathcal{M},w}(\Box^*, m)$. By theorem 24, the map $w \mapsto \phi_{\mathcal{M},w}(\Box^*, m)$ is a polynomial time reduction of A to the satisfiability problem for $\mathcal{L}(\Box^{=n}, \Box^* : n \in \mathbb{N})$ over $(\mathbb{N}, <)$. Hence this problem is **EXPSPACE**-hard. $\Box$

6 PDL with Bounded Modalities

In this section we introduce bounded iteration operators in the context of Propositional Dynamic Logic (**PDL**) [FL79]. There are two main differences between these dynamic logics and the modal logics we have previously considered. Firstly, dynamic logics usually include more than one modality. Note that the satisfiability problem for the multi-modal generalisation of K remains **PSPACE**-complete [HM92], so the extra modalities by themselves do not account for the increase in complexity. The other differences is that dynamic logics include operators that produce new modalities from old ones. In the context of dynamic logics, the modalities are usually called programs.

Definition 26 (PDL) *Let* Prop *be a countable set of propositions and let A be a set of atomic program names. Formulas ϕ and programs π of* **PDL** *are defined by the following syntax rules:*

$$\begin{array}{rcl} \phi & := & p \ (p \in \mathsf{Prop}) \mid \neg\phi \mid \phi \wedge \psi \mid [\pi]\phi \\ \pi & := & a \ (a \in A) \mid \pi_1 \cup \pi_2 \mid \pi_1; \pi_2 \mid \pi^* \end{array}$$

where A is a countably infinite set of atomic programs. We write $\langle\pi\rangle\phi$ as an abbreviation of $\neg[\pi]\neg\phi$. We write π^+ for $\pi; \pi^*$. Formulas of **IPDL** *are defined similarly, but the definition of a program becomes*

$$\pi := a \ (a \in A) \mid \pi_1 \cup \pi_2 \mid \pi_1 \cap \pi_2 \mid \pi_1; \pi_2 \mid \pi^*$$

Formulas of $\mathbf{PDL}_{/*}$, $\mathbf{IPDL}_{/*}$ *are obtained from* **PDL**, **IPDL** *(respectively) by omitting π^* from the definition of program.*

Operators	**Complexity of Satisfiability Problem**	**Reference**
$\mathcal{P}(\cup, \cap, ;)$	**PSPACE**-complete	[F01]
$\mathcal{P}(\cup, ;, ^*)$	**EXPTIME**-complete	[FL79, PR79]
$\mathcal{P}(\cup, \cap, ;, ^*)$	**2-EXPTIME**-complete	[LL05, D84]

Figure 2: Complexity of Propositional Dynamic Logics with bounded modalities

Formulas of these languages may be evaluated at a point of a multimodal Kripke structure $\mathcal{S} = (W, \langle R_a : a \in A\rangle, V)$, where each R_a is a binary relation over W (corresponding to the program $a \in A$) and the valuation V maps propositions to subsets of W. For each $a \in A$ let $a^{\mathcal{S}} = R_a$ and let $(\pi_1 \cup \pi_2)^{\mathcal{S}}, (\pi_1 \cap \pi_2)^{\mathcal{S}}, (\pi_1; \pi_2)^{\mathcal{S}}, (\pi^*)^{\mathcal{S}}$ be defined to be $(\pi_1)^{\mathcal{S}} \cup (\pi_2)^{\mathcal{S}}, (\pi_1)^{\mathcal{S}} \cap (\pi_2)^{\mathcal{S}}, (\pi_1)^{\mathcal{S}}|(\pi_2)^{\mathcal{S}}, (\pi^{\mathcal{S}})^*$, respectively, where $|$ denotes composition of binary relations and * denotes the reflexive, transitive closure of a binary relation. Formulas are evaluated at a world $w \in W$ of a structure $\mathcal{S} = (W, \langle R_a : a \in A\rangle)$ by,

$$\begin{aligned}
\mathcal{S}, w \models p &\iff w \in V(p)\\
\mathcal{S}, w \models \neg\phi &\iff w \not\models \phi\\
\mathcal{S}, w \models \phi_1 \wedge \phi_2 &\iff \mathcal{S}, w \models \phi_1 \text{ and } \mathcal{S}, w \models \phi_2\\
\mathcal{S}, w \models [\pi]\phi &\iff \forall v\ ((w, v) \in \pi^{\mathcal{S}} \to \mathcal{S}, v \models \phi)
\end{aligned}$$

A formula ϕ is satisfiable *if there is a multimodal Kripke model such that ϕ is true at some world in the model.*

Proposition 27

1. *The satisfiability problem for* $\mathbf{IPDL}_{/*}$ *is* **PSPACE**-*complete ([F01]).*
2. *The satisfiability problem for* **PDL** *is* **EXPTIME**-*complete ([FL79, PR79]).*
3. *The satisfiability problem for* **IPDL** *is* **2-EXPTIME**-*complete ([LL05, D84]).*

We now augment these dynamic logics by adding programs $\pi^{=n}, \pi^{\leq n}, \pi^{\geq n} : n \in \mathbb{N}$. As before, the superscripts are encoded in binary, so $|\pi^{=n}| = 1 + \lceil \log_2 n \rceil$. For the semantics, we let $(\pi^{=n})^{\mathcal{S}} = \overbrace{\pi^{\mathcal{S}}|\pi^{\mathcal{S}}|\ldots|\pi^{\mathcal{S}}}^{n}$, $(\pi^{\leq n})^{\mathcal{S}} = \bigcup_{i \leq n} (\pi^{=i})^{\mathcal{S}}$ and $(\pi^{\geq n})^{\mathcal{S}} = \bigcup_{n \leq i < \omega} ((\pi^{=i})^{\mathcal{S}})$. Let $Un \subseteq \{^*, ^{=n}, ^{\leq n}, ^{\geq n}\}$ and $Bin \subseteq \{\cup, \cap, ;\}$. We write $\mathcal{P}(Un \cup Bin)$ for the dynamic logic whose programs are defined by

$$\pi := a\ (a \in A) \mid \pi^u\ (u \in Un) \mid B(\pi_1, \pi_2)\ (B \in Bin)$$

where A is a countably infinite set of generator programs. We write $\mathcal{P}^k(Un \cup Bin)$ for the case where the set A of programs has at most k elements (so $\mathcal{P}^\omega(Un \cup Bin) = \mathcal{P}(Un \cup Bin)$). Thus **PDL** $= \mathcal{P}(^*, \cup, ;)$ and **IPDL**$= \mathcal{P}(^*, \cup, \cap, ;)$, etc. Figure 2 summarises known complexity results for dynamic logics and the table in the abstract summarises our new results, we prove the new results in theorems 30, 33, 35, 38 and corollary 34.

Definition 28 *The translation* $\mathbf{t}$ *maps programs of* $\mathcal{P}(\cup, \cap, ;, ^{=n}, ^{\geq n}, ^{\leq n}, ^* : n \in \mathbb{N})$ *to programs of* $\mathcal{P}(\cup, \cap, ;, ^*) =$ **IPDL**. *Given a program π of $\mathcal{P}(\cup, \cap, ;, ^{=n}, ^{\geq n}, ^{\leq n}, ^* : n \in \mathbb{N})$ we obtain $\mathbf{t}(\pi)$ by replacing each occurrence of $\pi^{=n}$ by $\overbrace{\pi; \pi; \ldots; \pi}^{n}$, each occurrence of $\pi^{\geq n}$ by $\overbrace{\pi; \pi; \ldots; \pi}^{n}; \pi^*$ and each occurrence of $\pi^{\leq n}$ by $\bigcup_{i \leq n} \pi^{=i}$. We can apply this translation $\mathbf{t}$ to formulas of $\mathcal{P}(\cup, \cap, ;, ^{=n}, ^{\geq n}, ^{\leq n}, ^* : n \in \mathbb{N})$ by replacing each program π occurring in a formula by $\mathbf{t}(\pi)$.*

Lemma 29 *Let $\phi \in \mathcal{P}(\cup, \cap, ;, ^{=n}, ^{\geq n}, ^{\leq n}, ^* : n \in \mathbb{N})$. We have $|\mathbf{t}(\phi)| \leq 2^{|\phi|^2}$ and $\mathbf{t}(\phi) \equiv \phi$. Let $X \subseteq \{\cup, \cap, ;, ^*\}$. If $\phi \in \mathcal{P}(X \cup \{^{=n}, ^{\geq n}, ^{\leq n} : n \in \mathbb{N}\})$ then $\mathbf{t}(\phi) \in \mathcal{P}(X)$.*

PROOF:

Similar to the proof of lemma 7 $\square$

THEOREM 30

- *The satisfiability problem for* $\mathcal{P}(\cap, \cup, ;, {}^{=n}, {}^{\leq n} : n \in \mathbb{N})$ *is in* **EXPSPACE**.
- *The satisfiability problem for* $\mathcal{P}(\cup, ;, {}^{=n}, {}^{\geq n}, {}^{\leq n}, {}^{*} : n \in \mathbb{N})$ *is in* **2-EXPTIME**.

PROOF:

For a formula $\phi \in \mathcal{P}(\cap, \cup, ;, {}^{=n}, {}^{\leq n} : n \in \mathbb{N})$ we know that $\mathbf{t}(\phi) \in \mathcal{P}(\cap, \cup, ;) =$ $\mathbf{IPDL}_{/*}$, and we can solve the satisfiability of $\mathbf{t}(\phi)$ in space $p(|\mathbf{t}(\phi)|) \leq p(2^{|\phi|^2})$, for some polynomial p, by proposition 27 and lemma 29. Hence the satisfiability problem for $\mathcal{P}(\cap, \cup, ;, {}^{=n}, {}^{\geq n}, {}^{\leq n})$ is in **EXPSPACE**.

Similarly, if $\phi \in \mathcal{P}(\cap, \cup, ;, {}^{=n}, {}^{\geq n}, {}^{\leq n}, {}^{*} : n \in \mathbb{N})$ we know that $t(\phi) \in \mathcal{P}(\cup, ;, {}^{*}) =$ **PDL**, and we can solve the satisfiability of $t(\phi)$ in time $2^{p(|t(\phi)|)} = 2^{p(2^{|\phi|^2})}$, i.e. in double-exponential time.

$\square$

DEFINITION 31 *Let a be the single generator for* $\mathcal{P}^1({}^{=n}, {}^{\geq n}, {}^{\leq n}, {}^{*} : n \in \mathbb{N})$. *Let* $\phi \in \mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^{*} : n \in \mathbb{N})$. *The formula* $h(\phi) \in \mathcal{P}^1({}^{=n}, {}^{\geq n}, {}^{\leq n}, {}^{*} : n \in \mathbb{N})$ *is obtained from* ϕ *by replacing each occurrence of* $\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^{*}$ *by* $[a^{=n}], [a^{\geq n}], [a^{\leq n}], [a^{*}]$*, respectively.*

LEMMA 32

1. *h can be computed in linear time.*
2. *A formula* $\phi \in \mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^{*} : n \in \mathbb{N})$ *is satisfiable iff* $h(\phi)$ *is satisfiable.*
3. *Hence h is a polynomial time reduction of* $\mathcal{L}(\Box^{=n}, \Box^{\geq n}, \Box^{\leq n}, \Box^{*} : n \in \mathbb{N})$ *to* $\mathcal{P}^1({}^{=n}, {}^{\geq n}, {}^{\leq n}, {}^{*} : n \in \mathbb{N})$.

The proof is obvious and we omit it.

THEOREM 33 *The satisfiability problem for* $\mathcal{P}^1({}^{=n}, {}^{\leq n} : n \in \mathbb{N})$ *is* **EXPSPACE**-*hard.*

PROOF:

By theorem 18(3) and lemma 32. $\square$

COROLLARY 34 *Let* $k \geq 1$ *and let* $\{{}^{=n}, {}^{\leq n} : n \in \mathbb{N}\} \subseteq X \subseteq \{\cap, \cup, ;, {}^{=n}, {}^{\geq n}, {}^{\leq n} : n \in \mathbb{N}\}$. *The satisfiability problem for* $\mathcal{P}^k(X)$ *is* **EXPSPACE**-*complete.*

PROOF:

By theorems 30 and 33. $\square$

THEOREM 35 *The satisfiability problem for any language between* $\mathcal{P}(\cup, {}^{=n}, {}^{*} : n \in \mathbb{N})$ *and* $\mathcal{P}(\cup, ;, {}^{=n}, {}^{\geq n}, {}^{\leq n}, {}^{*} : n \in \mathbb{N})$ *is* **2-EXPTIME**-*complete.*

PROOF:

The satisfiability problem for $\mathcal{P}(\cup, ;, {}^{=n}, {}^{\geq n}, {}^{\leq n}, {}^{*} : n \in \mathbb{N})$ is in **2-EXPTIME**, by theorem 30. By theorem 18 and lemma 32, the satisfiability problem for $\mathcal{P}(\cup, {}^{=n}, {}^{*} : n \in \mathbb{N})$ is **2-EXPTIME**-hard. $\square$

THEOREM 36

1. *There is a polynomial time reduction from the satisfiability problem for $\mathcal{P}(\cap, \cup, ;, {}^{\geq n}, {}^{\leq n}, {}^{=n}, {}^{*} : n \in \mathbb{N})$ to that of $\mathcal{P}(\cap, \cup, ;, {}^{*})$.*

2. *There is a polynomial time reduction from the satisfiability problem for $\mathcal{P}(\cap, \cup, ;, {}^{\leq n}, {}^{=n} : n \in \mathbb{N})$ to that of $\mathcal{P}(\cap, \cup, ;, {}^{\leq n} : n \in \mathbb{N})$.*

PROOF:

1. For the first part, let $\phi \in \mathcal{P}(\cap, \cup, ;, {}^{\geq n}, {}^{\leq n}, {}^{=n}, {}^{*} : n \in \mathbb{N})$. We will define the reduction by defining a formula

$$\phi' = \phi_1 \wedge [\Delta^*]\theta_1 \quad \in \mathcal{P}(\cap, \cup, ;, {}^{*}) \tag{4}$$

where Δ is the union of all the atomic programs occurring in ϕ. Since the deletion of worlds not reachable by Δ^* in any structure does not affect the truth of a formula, we will restrict our attention to generated structures. Let η be a new program. θ_1 will include a conjunct $([\eta]\bot)$, so η will be interpreted as the empty program, in any model of $[\Delta^*]\theta_1$. Let $e = \eta^*$. This program e will be interpreted as the identity over the domain of any model of θ_1.

Suppose $\pi^{\diamond n}$ occurs in ϕ, for some program π, some $\diamond \in \{\geq, \leq, =\}$ and some $n \in \mathbb{N}$. We can assume that n is even, as we can replace $\pi^{\geq n}$ by $\pi; \pi^{\geq n-1}$, $\pi^{\leq n}$ by $e \cup \pi; \pi^{\leq n-1}$ and $\pi^{=n}$ by $\pi; \pi^{=n-1}$ if n is odd. For each n and π such that $\pi^{\diamond n}$ occurs in ϕ, for some $\diamond$, let $k = \lceil \log_2 \frac{n}{2} \rceil$. We introduce propositional space counters $r, s_i : i < k$ and programs $\rho, \overline{\rho}, \sigma_i, \overline{\sigma}_i : i < k$, unique to n, π. Loosely speaking, the propositions $s_i : i < k$ (together with corresponding programs $\sigma_i, \overline{\sigma}_i : i < k$) suffice to count the distance from one point to another, modulo $\frac{n}{2}$, but we need the other proposition r (and corresponding programs $\rho, \overline{\rho}$) to dispense with the 'modulo $\frac{n}{2}$' and measure the true distance from one point to another, so long as the distance does not exceed n.

In more detail, the propositional space counters define the *position* $pos_s^{\mathcal{S}}(v)$ of any world v in a structure $\mathcal{S}$, by $pos_s^{\mathcal{S}}(v) = \sum\{2^i : i < k, \mathcal{S}, v \models s_i\}$. Let $m < 2^k$ and define

$$\begin{aligned}\chi_{s=m} &= \bigwedge_{\mathsf{bit}(i,m)=1} s_i \wedge \bigwedge_{\mathsf{bit}(i,m)=0} \neg s_i \\ \chi_{s<m} &= \bigvee_{\mathsf{bit}(i,m)=1} [\neg s_i \wedge \bigwedge_{i<j<k,\ \mathsf{bit}(j,m)=0} \neg s_i]\end{aligned}$$

Clearly, $\mathcal{S}, v \models \chi_{s=m} \iff pos_s^{\mathcal{S}}(v) = m$ and $\mathcal{S}, v \models \chi_{s<m} \iff pos_s^{\mathcal{S}}(v) < m$.
θ_1 will include, under the scope of $[\Delta^*]$, a conjunct $\mathbf{inc}(s)$, defined to be

$$\begin{array}{lcl}(\neg\chi_{s=\frac{n}{2}-1} & \rightarrow & \bigwedge_{i<k}[(\neg s_i \wedge \bigwedge_{j<i} s_i) \rightarrow ([\pi](s_i \wedge \bigwedge_{j<i} \neg s_i) \wedge \bigwedge_{i<j<k}(s_j \rightarrow [\pi]s_j \wedge \neg s_j \rightarrow [\pi]\neg s_j))] \\ & \wedge & \\ \chi_{s=\frac{n}{2}-1} & \rightarrow & [\pi]\chi_{s=0})\end{array}$$

Observe that if $\mathcal{S}, v \models \mathbf{inc}(s)$, $pos_s^{\mathcal{S}}(v) < \frac{n}{2}$ and u is any successor of v then $pos_s^{\mathcal{S}}(u) = pos_s^{\mathcal{S}}(v) + 1 (\mathsf{mod}\ \frac{n}{2})$.
θ_1 will include under $[\Delta^*]$ a further conjunct $Alt_s =$

$$(r \rightarrow [\pi]((\neg\chi_{s=0} \wedge r) \vee (\chi_{s=0} \wedge \neg r)) \wedge (\neg r \rightarrow [\pi]((\neg\chi_{s=0} \wedge \neg r) \vee (\chi_{s=0} \wedge r))))$$

If $\mathcal{S}, w \models \chi_{s<\frac{n}{2}} \wedge Copy_s \wedge Alt_s$ and $\overline{w}$ is any w-path of π transitions, then the position $pos_s^{\mathcal{S}}(w_i)$ must increment by one modulo $\frac{n}{2}$ every time you increment i, the proposition r must stay constant (either it must stay true, or it must stay

false) every time you increment the position, except when the position goes from $\frac{n}{2}-1$ to 0 and in this case the truth of r is flipped. We call a sequence of $\frac{n}{2}$ worlds $v_0, v_1, \ldots v_{\frac{n}{2}-1}$ a *block* if $(v_i, v_{i+1}) \in \pi^{\mathcal{S}}$ (all $i < \frac{n}{2}-1$) and either $\mathcal{S}, v_i \models r$ (all $i < \frac{n}{2}-1$) or $\mathcal{S}, v_i \models \neg r$ (all $i < \frac{n}{2}-1$). In the former case we may call it an r-block and in the latter a $\neg r$-block.
Let $\sigma_i' = \sigma \cap e$, $\rho' = \rho \cap e$, $\overline{\sigma}_i' = \overline{\sigma} \cap e$ and $\overline{\rho}' = \overline{\rho} \cap e$, for $i < k$. The reflexive programs $\sigma_i, \rho, \overline{\sigma}_i, \overline{\rho} : i < k$ are intended to mirror the corresponding propositions $s_i, r : i < k$. θ_1 will also include under $[\Delta^*]$ a final conjunct:

$$Copy_s = (\bigwedge_{i<k}(s_i \rightarrow \langle\sigma_i'\rangle\top \wedge \neg s_i \rightarrow \langle\overline{\sigma}_i'\rangle\top \wedge [\sigma_i' \cap \overline{\sigma}_i']\bot) \wedge (r \rightarrow \langle\rho'\rangle\top \wedge \neg r \rightarrow \langle\overline{\rho}'\rangle\top) \wedge [\rho' \cap \overline{\rho}']\bot)$$

If $\mathcal{S}, v \models Copy_s$ then propositions s_i, r are true at v iff σ_i', ρ' hold at the reflexive edge (v, v) and negative literals $\neg s_i, \neg r$ hold at v iff $\overline{\sigma}_i', \overline{\rho}'$ hold at (v, v), for $i < k$. Thus these reflexive programs exactly match the corresponding propositions and their negations.
Let

$$\theta_1 = \bigwedge_{s=s(\pi,n)} \chi_{s<\frac{n}{2}} \wedge \mathbf{inc}(s) \wedge Copy_s \wedge Alt_s$$

Define programs

$$\begin{aligned}
Same &= \bigcap_{i<k}((\sigma_i; \pi^*; \sigma_i') \cup (\overline{\sigma}_i'; \pi^*; \overline{\sigma}_i')) \\
Left &= \bigcup_{i<k}[(\sigma_i'; \pi^*; \overline{\sigma}_i') \cap \bigcap_{i<j<k}[(\sigma_i'; \pi^*; \sigma_i') \cup (\overline{\sigma}_i'; \pi^*; \overline{\sigma}_i')]] \\
Right &= \bigcup_{i<k}[(\overline{\sigma}_i'; \pi^*; \sigma_i') \cap \bigcap_{i<j<k}[(\sigma_i'; \pi^*; \sigma_i') \cup (\overline{\sigma}_i'; \pi^*; \overline{\sigma}_i')]]
\end{aligned}$$

We will see that if $(u, v) \in Same^{\mathcal{S}}$ ($\in Left^{\mathcal{S}}$, $\in Right^{\mathcal{S}}$ respectively) then $pos_s^{\mathcal{S}}(u) = pos_s^{\mathcal{S}}(v)$ ($pos_s^{\mathcal{S}}(u) < pos_s^{\mathcal{S}}(v), pos_s^{\mathcal{S}}(u) > pos_s^{\mathcal{S}}(v)$).
We are now ready to replace occurrences of $\pi^{=n}, \pi^{\leq n}$ and $\pi^{\geq n}$ in ϕ. For $\diamond \in \{=, \leq, \geq\}$ we define a program $\pi_{\diamond n}$ using only $\{\cap, \cup, ;, ^*\}$ and of polynomial length (in terms of the length of $\pi^{\diamond n}$). In models of $\mathbf{inc}(s) \wedge Copy_s \wedge Alt_s$, $\pi_{\diamond n}$ will evaluate to the same binary relation as $\pi^{\diamond n}$. Let

$$\begin{aligned}
\pi_{=n} &= [[(\rho'; \pi)^+; (\overline{\rho}'; \pi)^+; (\rho'; \pi)^*; \rho'] \cup [(\overline{\rho}'; \pi)^+; (\rho'; \pi)^+; (\overline{\rho}'; \pi)^*; \overline{\rho}']] \cap Same \\
\pi_{\leq n} &= [\ [(\rho'; \pi)^*; (\overline{\rho}'; \pi)^*] \cup [(\overline{\rho}'; \pi)^*; (\rho'; \pi)^*] \cup \\
&\quad (Left \cap ([(\rho'; \pi)^* (\overline{\rho}'; \pi)^*; (\rho'; \pi)^*] \cup [(\overline{\rho}'; \pi)^*; (\rho'; \pi)^*; (\overline{\rho}'; \pi)^*]))\ ] \\
\pi_{\geq n} &= [([(\rho'; \pi)^+; (\overline{\rho}'; \pi)^+; (\rho'; \pi)^+] \cup [(\overline{\rho}'; \pi)^+; (\rho'; \pi)^+; (\overline{\rho}'; \pi)^+]) \cap Right]; \pi^*
\end{aligned}$$

Claim 1: Suppose $\mathcal{T}, w \models \chi_{s<\frac{n}{2}} \wedge [\Delta^*](\mathbf{inc}(s) \wedge Copy_s \wedge Alt_s)$ and suppose every world of $\mathcal{T}$ is Δ-reachable from w. Then $\pi_{\diamond n}^{\mathcal{T}} = (\pi^{\diamond n})^{\mathcal{T}}$.
We prove the first case of the claim, $\diamond$ is $=$, the other cases are similar. Consider $\pi_{=n}$. Suppose $(u, v) \in \pi_{=n}^{\mathcal{T}}$, both u and v are Δ-reachable from w. Since $(u, v) \in Same$ we have $u \in s_i^{\mathcal{T}} \iff v \in s_i^{\mathcal{T}}$ for $i < k$, hence $pos_s^{\mathcal{S}}(u) = pos_s^{\mathcal{S}}(v)$. By definition of $\pi_{=n}$, there is a π-sequence $\lambda = (u = u_0, u_1, \ldots, u_p = v)$ from u to v and since all worlds in λ satisfy $\mathbf{inc}(s) \wedge Copy_s \wedge Alt_s$, λ must consist of a final segment of an r-block concatenated with a $\neg r$-block and an initial segment of an r-block, or similar with $r, \neg r$ reversed. Hence v is two blocks after u. Since $pos_s^{\mathcal{S}}(u) = pos_s^{\mathcal{S}}(v)$ it follows that the length of λ is exactly n, so $(u, v) \in (\pi^{=n})^{\mathcal{T}}$. Similarly for the converse, suppose $(u, v) \in (\pi^{=n})^{\mathcal{T}}$ and let $\lambda = (u = u_0, u_1, \ldots, u_n = v)$ be the π-path from u to v. Since each world satisfies $\chi_{s<\frac{n}{2}} \wedge \mathbf{inc}(s) \wedge Copy_s \wedge Alt_s$, as we increment through the sequence λ the positions

increment by one each time, and we pass from a block to the next block but one, hence $(u,v) \in \pi^{\mathcal{T}}_{=n}$. This proves the (first case of the) claim.

Claim 2: If $\mathcal{S}, w \models \phi$ and $\Delta^{\mathcal{S}}$ is a tree with root w, then there is a structure $\mathcal{S}'$ such that $\mathcal{S}', w \models \phi \wedge \chi_{s<\frac{n}{2}} \wedge [\Delta^*](\mathbf{inc}(s) \wedge Copy_s \wedge Alt_s)$.

Proof of claim 2 Define a structure $\mathcal{S}'$ with the same worlds as $\mathcal{S}$ and all propositions and programs have the same interpretation in $\mathcal{S}'$, except the new programs $\rho, \overline{\rho}, \sigma_i, \overline{\sigma}_i : i < k$ and propositions $r, s_i : i < k$. Hence $\mathcal{S}', w \models \phi$. The space counters s_i are given a valuation so that whenever $(u,v) \in \pi^{\mathcal{S}}$ we have $pos^{\mathcal{S}}_s(u), pos^{\mathcal{S}}_s(v) < \frac{n}{2}$ and $pos^{\mathcal{S}}_s(v) = pos^{\mathcal{S}}_s(u) + 1 (\mathsf{mod}\ \frac{n}{2})$. Since $\Delta^{\mathcal{S}}$ is a tree, this can be done. r is given a valuation so that π-paths form blocks, i.e. if $(u,v) \in \pi^{\mathcal{S}}$ and $pos^{\mathcal{S}}_s(u) < \frac{n}{2} - 1$ then $u \in r^{\mathcal{S}'} \iff v \in r^{\mathcal{S}'}$ and if $pos^{\mathcal{S}}_s(u) = \frac{n}{2} - 1$ then $u \in r^{\mathcal{S}'} \iff v \notin r^{\mathcal{S}'}$. The new programs are interpreted as reflexive relations such that $(u,u) \in \sigma_i^{\mathcal{S}'} \iff u \in s_i^{\mathcal{S}'} \iff (u,u) \notin \overline{\sigma}_i^{\mathcal{S}'}$ and similarly $(u,u) \in \rho^{\mathcal{S}'} \iff u \in r^{\mathcal{S}'} \iff (u,u) \notin \overline{\rho}^{\mathcal{S}'}$. From this definition, we have $\mathcal{S}', u \models \mathbf{inc}(s) \wedge Copy_s \wedge Alt_s$, at each world u of $\mathcal{S}'$. The claim follows, since $\mathcal{S}', w \models \phi$.

We are now ready to define ϕ'. Let ϕ_1 be obtained from ϕ by replacing each occurrence of $\pi^{=n}, \pi^{\leq n}, \pi^{\geq n}$ by $\pi_{=n}, \pi_{\leq n}, \pi_{\geq n}$ respectively. ϕ' is now defined by (4). To see that this reduction is correct, suppose $\mathcal{S}, w \models \phi$, where $\Delta^{\mathcal{S}}$ is a tree with root w. By claim 2, there is a structure $\mathcal{S}'$ such that $\mathcal{S}' \models \phi \wedge \bigwedge_{s=s(\pi,n)} \chi_{s<\frac{n}{2}} \wedge [\Delta^*](\mathbf{inc}(s) \wedge Copy_s \wedge Alt_s)$. By claim 1 we have $\pi^{\mathcal{S}'}_{\diamond n} = (\pi^{\diamond n})^{\mathcal{S}'}$, hence $\mathcal{S}', w \models \phi$ implies that $\mathcal{S}', w \models \phi_1$, so $\mathcal{S}', w \models \phi'$. Conversely if $\mathcal{T}, w \models \phi'$ and $\Delta^{\mathcal{T}}$ is a tree with root w then by claim 1 $\mathcal{T}, w \models \phi$. This proves that the reduction is correct.

2. The proof of the second reduction is similar. Let $\phi \in \mathcal{P}(\cap, \cup, ;, ^{\leq n}, ^{=n} : n \in \mathbb{N})$. The reduction maps ϕ to $\phi' = \phi_2 \wedge [\Delta^*]\theta_2 \in \mathcal{P}(\cap, \cup, ;, ^{\leq n} : n \in \mathbb{N})$. This time, the task is to eliminate all occurrences of $\pi^{=n}$ from ϕ. Here we let $k = \lceil \log_2 n \rceil$. As before, we use space propositions $s_i : i < k$ and corresponding programs $\sigma_i, \overline{\sigma}_i : i < k$, for each occurrence of $\pi^{=n}$ in ϕ, but we do not need r, ρ or $\overline{\rho}$ as we can use $\pi^{\leq n}$ to restrict the distance to a point. This time we define the identity program by $e = \pi^{\leq 0}$, where π can be any program and we let $\sigma'_i = \sigma_i \cap e$, $\overline{\sigma}'_i = \overline{\sigma}_i \cap e$, for $i < k$. For $m < 2^k$, let $\chi_{s=m}$, $\chi_{s<m}$, $\mathbf{inc}(s)$ be the same as in the previous part,though we have changed the definition of k here. The definition of $Copy_s$ is slightly simpler, as we are not using proposition r,

$$Copy_s = \bigvee_{i \leq p(n)} [(s_i \to \langle \sigma'_i \rangle \top) \wedge \ (\neg s_i \to \langle \bar{\sigma}'_i \rangle \top) \wedge [\sigma'_i \cap \bar{\sigma}'_i] \bot]$$

Let $\theta_2 = \bigwedge_{s=s(\pi,n)} \chi_{s<n} \wedge \mathbf{inc}(s) \wedge Copy_s$. We now define a program $\pi_{=n}$, intended to replace $\pi^{=n}$.

$$\pi_{=n} = \bigcap_{i<k} ([\sigma'_i; \pi; \pi^{\leq n-1}; \sigma'_i\ ] \cup \ [\overline{\sigma}'_i; \pi; \pi^{\leq n}; \overline{\sigma}'_i])$$

Observe, for any structure $\mathcal{S}$ where all points satisfy θ_2, that if $(v,w) \in \pi^{\mathcal{S}}_{=n}$ then all space counters have the same value at u and v, and there is a π-path of length at most n and length at least one, from u to v, hence the length of the path is exactly n and $(u,v) \in (\pi^{=n})^{\mathcal{S}}$. The other containment $(\pi^{=n})^{\mathcal{S}} \subseteq (\pi_{=n})^{\mathcal{S}}$ also holds, similarly. Let ϕ_2 be obtained from ϕ by replacing each occurrence of $\pi^{=n}$ by $\pi_{=n}$. The reduction maps ϕ to $\phi_2 \wedge [\Delta^*]\theta_2$. By the foregoing, if all points of a structure $\mathcal{S}$ satisfy θ_2 then $(\pi^{=n})^{\mathcal{S}} = (\pi_{=n})^{\mathcal{S}}$, hence the reduction is correct.

□

Corollary 37 *Let X be a signature containing $\{\cup, \cap, ;, *\}$ and contained in $\{\cup, \cap, ;, ^{=n}, ^{\leq n}, ^{\geq n} : n \in \mathbb{N}\}$. Then the satisfiability problem for $\mathcal{P}(X)$ is double exponential time complete.*

PROOF:

By proposition 27 and theorem 36(1). $\Box$

THEOREM 38 *The satisfiability problem for* $\mathcal{P}(\cap, \cup, ;, {}^{\leq n} : n \in \mathbb{N})$ *is* **EXPSPACE***-complete.*

PROOF:

$\mathcal{P}(\cap, \cup, ;, {}^{\leq n} : n \in \mathbb{N})$ is in **EXPSPACE**, by theorem 30. To prove **EXPSPACE**-hardness, take an exponential time **ATM** $\mathcal{M}$ and an input w to the machine, let the run time (and space) of $\mathcal{M}$ on input w be bound by $m = 2^{p(|w|)}$, for some polynomial p. By theorem 15, $\phi_{\mathcal{M},w}(\Box^{\leq m^2}, m) \in \mathcal{L}(\Box^{=n}, \Box^{\leq n} : n \in \mathbb{N})$ is satisfiable iff $w \in \mathcal{L}(\mathcal{M})$. Recall that the reduction of definition 31 replaces $\Box$ by $[a]$. Let $\phi_{\mathcal{M},w}([a^{\leq m^2}], m) \in \mathcal{P}^1({}^{=n}, {}^{\leq n} : n \in \mathbb{N})$ be the formula obtained from $\phi_{\mathcal{M},w}(\Box^{\leq m^2}, m)$ by applying this reduction. Observe that $\phi_{\mathcal{M},w}([a^{\leq m^2}], m)$ does not involve *, but it remains for us to eliminate occurrences of ${}^{=n}$. By the foregoing, $\phi_{\mathcal{M},w}([a^{\leq m^2}], m)$ is satisfiable iff $\mathcal{M}$ accepts w. Let h be the polynomial time reduction from $\mathcal{P}(\cap, \cup, ;, {}^{\leq n}, {}^{=n} : n \in \mathbb{N})$ to $\mathcal{P}(\cap, \cup, ;, {}^{\leq n} : n \in \mathbb{N})$ of theorem 36(2). To summarise: $\mathcal{M}$ accepts w iff $h(\phi_{\mathcal{M},w}([a^{\leq m^2}], m)) \in \mathcal{P}(\cap, \cup, ;, {}^{\leq n} : n \in \mathbb{N})$ is satisfiable, proving that $\mathcal{P}(\cap, \cup, ;, {}^{\leq n} : n \in \mathbb{N})$ is **EXPSPACE**-hard. $\Box$

7 Open Problems

- Find the complexities of $\mathcal{L}(\Box^{=n} : n \in \mathbb{N})$ and $\mathcal{L}(\Box^{\leq n} : n \in \mathbb{N})$-SAT over Kripke frames.
- Find the complexities of $\mathcal{L}(\Box^{=n}, \Box^{\leq n} : n \in \mathbb{N})$, $\mathcal{L}(\Box^{=n} : n \in \mathbb{N})$-SAT and $\mathcal{L}(\Box^{\leq n} : n \in \mathbb{N})$-SAT over the linear frame $(\mathbb{N}, <)$.
- Find the complexities of $\mathcal{P}(\cup, ;, {}^{\leq n} : n \in \mathbb{N})$-SAT and $\mathcal{P}(\cup, ;, {}^{=n} : n \in \mathbb{N})$-SAT.
- Find the complexity of $\mathcal{P}(\cap, \cup, ;, {}^{=n} : n \in \mathbb{N})$-SAT.

References

[AH94] Rajeev Alur and Thomas A. Henzinger. A Really Temporal Logic. Journal of the ACM 41:181-204, 1994.

[CKS81] A.K. Chandra, D.C.Kozen, L.J. Stockmeyer. Alternation. Journal of the ACM, 28(1): 114-133, January 1981.

[D84] R. Danecki. Non Deterministic propositional dynamic logic with intersection is decidable. Proceeding of the Fifth Symposium on Computation Theory, Poland. LNCS vol. 208, Springer 1984, pp. 34-53.

[EMH82] E. A. Emerson, J. Halpern. Decision Procedures and expressiveness in the temporal logic of branching time. In Proceeding of STOC 1982, pages: 169-180, ACM Press, 1982.

[EMJ86] E. A. Emerson, C.S. Juttla. Tree automata and the logics of programmes. SIAM-JC, 29(1): 132-158, 1986.

[EMSS92] E. A. Emerson, A. K. Mok, A. P. Sistla, and J. Srinivasan. Quantitative Temporal Reasoning. Real Time Systems, Volume 4: pages 331-352, 1992.

[F01] Fabio Massacci. Decision procedures for expressive Description Logics with intersection, composition, converse of roles and role identity. In Proc. of the 17th Int. Joint Conf. on Artificial Intelligence (IJCAI-2001), pp.193-198, 2001.

[FL79] Fischer, M.J. and Ladner, R.E. Propositional dynamic logic of regular programs. Journal of Computer and Systems Sciences. v18. 194-211, 1979.

[Hem96] E Hemaspaandra. The price of universality. *Notre Dame Journal of Formal Logic*, 37(2):174–203, 1996.

[HM92] Joseph Y. Halpern and Yoram Moses. A guide to completeness and complexity for modal logics of knowledge and belief. Artificial Intelligence archive Volume 54 , Issue 3 (April 1992) Pages: 319-379, 1992

[JL03] Jan Johannsen, Martin Lange. CTL + is complete for double exponential time. *Proc. 30th Int. Coll. on Automata, Logics and Programming, ICALP, volume 2719 of LNCS, pages 767 775* , 2003.

[L77] R. Ladner. The computational complexity of provability in systems of modal logic. *SIAM Journal On Computing, 6: 467-480*, 1977.

[LL05] M. Lange and C. Lutz. 2-ExpTime Lower Bounds for Propositional Dynamic Logics with Intersection. Journal of Symbolic Logic 70(4), pages 1072-1086, 2005.

[PN77] A. Pnueli. The temporal logic of programs. In Proceedings of the 18th Annual Symposium on Foundations of Computer Science (Providence, RI.). IEEE, New York, pages 46-57, 1977.

[PR79] V.R. Pratt. Models of program logics. In Proc. 20th IEEE Symp. Foundations of Computer Science, pages 115-222, 1979.

[SC82] A. P. Sistla and E. M. Clarke. The complexity of propositional linear temporal logics. Proceedings of the fourteenth annual ACM symposium on Theory of Computing, pages: 159 - 168, 1982.

[VS85] M. Vardi and L. Stockmeyer. Improved upper and lower bounds for modal logics of programs. In Proceedings of STOC 1985, ACM Press, 1985.