The complexity of model checking for intuitionistic logics and their modal companions

Martin Mundhenk and Felix Weiß

Universität Jena, Institut für Informatik, Jena, Germany
{martin.mundhenk,felix.weiss}@uni-jena.de

Abstract. We study the model checking problem for logics whose semantics are defined using transitive Kripke models. We show that the model checking problem is P-complete for the intuitionistic logic KC. Interestingly, for its modal companion S4.2 we also obtain P-completeness even if we consider formulas with one variable only. This result is optimal since model checking for S4 without variables is NC^1-complete. The strongest variable free modal logic with P-complete model checking problem is K4. On the other hand, for KC formulas with one variable only we obtain much lower complexity, namely LOGDCFL as an upper bound.

1 Introduction

We investigate the complexity of the model checking problem for intuitionistic propositional logics and for its modal companions. Intuitionistic propositional logic IPC (see e.g. [1]) is the part of classical propositional logic that goes without the use of the excluded middle $a \vee \neg a$. We will use its semantical definition by Kripke models with a partially ordered set of states and a monotone valuation function. A straightforward upper bound follows from the Gödel-Tarski translation (see e.g. [2, p.96]) that embeds intuitionistic logic into the modal logic S4. Since the model checking problem—given a formula and a model, does the model satisfy the formula (or does the formula evaluate to "true" under the model)?—for modal logic is in P [3], we obtain the same as an upper bound for the problem in intuitionistic logic. For classical propositional logic, the model checking problem can be solved in logarithmic space [4] and even better in alternating logtime [5]. Since the models for classical logic can be seen as a special case of Kripke models with one state only, we cannot expect such a low complexity for intuitionistic logic, where the models may consist of many states.

More generally, we will consider the classical propositional logic PC, the intuitionistic logics LC (Gödel-Dummet logic, see [6]), KC (Jankov's logic, see [6]), IPC, and BPL (Visser's basic propositional logic [7]), and their respective modal companions S5, S4.3, S4.2, S4, and K4 (see e.g. [2] for an overview). Remind that PC $\supset$ LC $\supset$ KC $\supset$ IPC $\supset$ BPL.

Our first hardness result (Theorem 2) is the P-hardness of the model checking problem for the superintuitionistic (or intermediate) logic KC. This hardness result consequently also holds for IPC and BPL and their companions S4.2, S4, and K4. Hence, the well-known upper bound [3] turns out to be the lower bound.

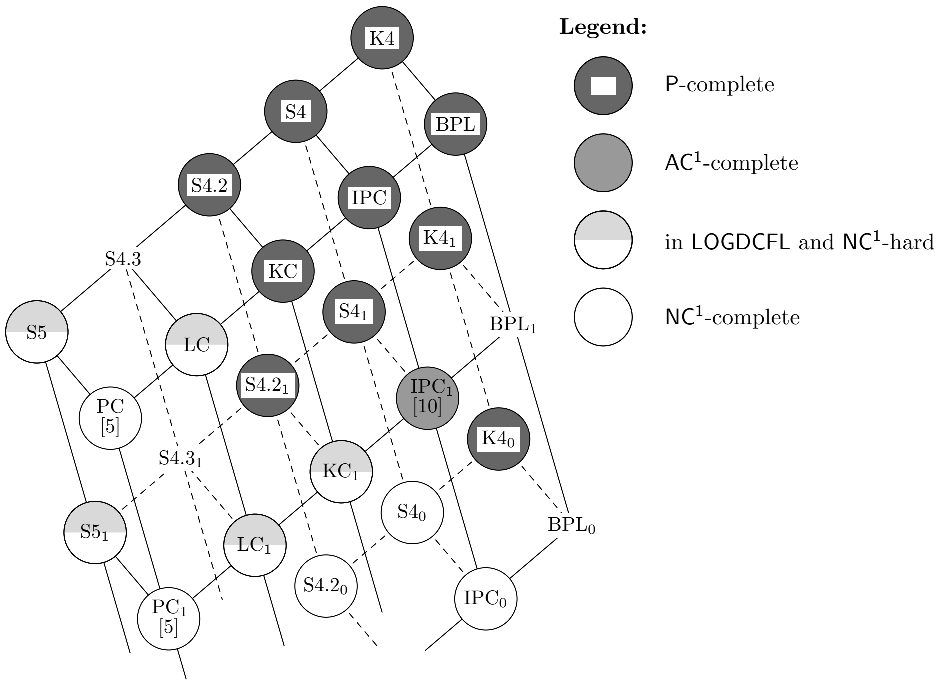

Fig. 1. Summary of results: the structure of the logics and the complexity of the model checking problem. Lower and upper bounds for the uncircled logics follow from their neighbourhoods, but non-trivial bounds are unknown.

Since the expressivity of intuitionistic logics seems to be much lower than that of their modal companions, it is somewhat surprising that all these logics have P-hard model checking problems. In fact, the satisfiability problem for S4.2 up to K4 are PSPACE-complete [8, 9], whereas the satisfiability problem for intuitionistic logic has the same complexity as that for classical logic—both are NP-complete. We can point out some differences for the model checking problem that can be seen as a result of the greater expressivity of modal logics. This difference appears if we consider formulas with one variable only or without any variables. In Theorem 3 we show that the model checking problem remains P-hard for S4.2, even if we consider formulas with one variable only. For K4 we show P-hardness, even if we consider formulas without variables (Theorem 4). These results are in contrast to the recent result in [10] showing that the model checking problem for IPC with one variable only is AC^1-complete. For KC with one variable only we will show that the complexity of model checking is even lower, namely in LOGDCFL (Theorem 7). Regarding the number of variables for S4.2 resp. S4, Theorem 3 is optimal. We show that model checking for the variable free fragment of S4 is NC^1-complete (Theorem 8).

Figure 1 summarizes our results. There, PC denotes classical propositional logic, and subscript 1 or 0 (e.g. $S4.2_1$) denotes the fragment with one variable only resp. without variables.

Technically, our hardness results use a reduction from the alternating graph accessibility problem AGAP, being one of the standard P-complete problems [11, 12]. It can straightforwardly be logspace reduced to the model checking problem for propositional modal logic by taking the alternating graph as the frame of a Kripke model (with an empty valuation function) and a formula essentially consisting of a sequence of $\Box$ and $\Diamond$ operators that simulates the search through the graph. This straightforward approach does not work anymore when we want to reduce to Kripke models with transitive frames—like for the modal logic S4 or intuitionistic propositional logic. On the one hand, making an alternating graph transitive, destroys essential properties it has, and on the other hand, a logspace reduction does not have enough computational power to calculate the transitive closure of a directed graph.

This paper is organized as follows. In Section 2 we introduce the notations for the logics under consideration, and we show P-completeness of a graph accessibility problem for a special case of alternating graphs that will be used for our P-hardness proofs. In Section 3 we give P-hardness proofs for the model checking problem for KC, $S4.2_1$, and $K4_0$. The upper bounds are presented in Section 4. The resulting completeness results and conclusions are drawn in Section 5.

2 Preliminaries

Kripke Models. We will consider different propositional logics whose formulas base on a countable set PROP of *propositional variables* (resp. atoms). A *Kripke model* is a triple $\mathcal{M} = (U, R, \xi)$, where U is a nonempty and finite set of *states*, R is a binary relation on U, and $\xi : \text{PROP} \to \mathfrak{P}(U)$ is a function — the *valuation function*. Informally spoken, for any variable it assigns the set of states in which this variable is satisfied. (U, R) can also be seen as a directed graph—it is called a *frame* in this context.

Modal Propositional Logic. The language $\mathcal{ML}$ of modal logic is the set of all formulas of the form

$$\varphi ::= \bot \mid p \mid \varphi \to \varphi \mid \Diamond\varphi\,,$$

where $p \in \text{PROP}$. As usual, we use the abbreviations $\neg\varphi := \varphi \to \bot$, $\top := \neg\bot$, $\varphi \vee \psi := (\neg\varphi) \to \psi$, $\varphi \wedge \psi := \neg(\varphi \to \neg\psi)$, and $\Box\varphi := \neg\Diamond\neg\varphi$.

The semantics is defined via Kripke models. Given a model $\mathcal{M} = (U, R, \xi)$ and a state $s \in U$, the *satisfaction relation for modal logics* $\models_M$ is defined as follows.

$$\begin{array}{lll}
\mathcal{M}, s \not\models_M \bot & & \\
\mathcal{M}, s \models_M p & \text{iff} & s \in \xi(p),\ p \in \text{PROP}, \\
\mathcal{M}, s \models_M \varphi \to \psi & \text{iff} & \mathcal{M}, s \not\models_M \varphi \text{ or } \mathcal{M}, s \models_M \psi, \\
\mathcal{M}, s \models_M \Diamond\varphi & \text{iff} & \exists t \in U : sRt \text{ and } \mathcal{M}, t \models_M \varphi.
\end{array}$$

A formula φ is *satisfied* by model $\mathcal{M}$ in state s iff $\mathcal{M}, s \models_M \varphi$. If it is satisfied by $\mathcal{M}$ in every state s of $\mathcal{M}$, then we write $\mathcal{M} \models_M \varphi$.

The modal logic defined in this way is called K (after Saul Kripke) and it is the weakest normal modal logic. We will consider the stronger modal logics K4, S4, S4.2, S4.3, and S5. The formulas in all these logics are the same as for $\mathcal{ML}$. Since we are interested in formula evaluation, we use the semantics defined by Kripke models. They will be defined by properties of the frame (i.e. graph) (U, R) that is part of the model. A frame (U, R) is *reflexive*, if xRx for all $x \in U$, and it is *transitive*, if for all $a, b, c \in U$, it follows from aRb and bRc that aRc. A reflexive and transitive frame is called a *preorder*. If a preorder (U, R) has the additional property that for all $a, b \in U$ there exists a $c \in U$ with aRc and bRc, then (U, R) is called a *directed preorder*. If for all $a, b \in U$ holds aRb or bRa, then (U, R) is called a *linear preorder*.

The semantics of several modal logics can be defined by restricting the class of Kripke frames under consideration. The semantics of K4 is defined by transitive frames. This means, that a formula α is a theorem of K4 if and only if $\mathcal{M} \models_M \alpha$ for all models $\mathcal{M}$ whose frame is transitive. The semantics of S4 is defined by preorders, of S4.2 by directed preorders, of S4.3 by linear preorders, and of S5 by equivalence relations (symmetric preorders). For any logic L, let L_i denote its fragment with i variables only. The fragment L_0 has no variables but the constant $\perp$ only.

Intuitionistic Propositional Logic. The language $\mathcal{IPC}$ of intuitionistic propositional logic is the same as that of propositional logic $\mathcal{PC}$, i.e. it is the set of all formulas of the form

$$\varphi ::= \perp \mid p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \varphi \rightarrow \varphi\,,$$

where $p \in \mathrm{PROP}$. As usual, we use the abbreviations $\neg\varphi := \varphi \rightarrow \perp$ and $\top := \neg\perp$. Because of the semantics of intuitionistic logic, one cannot express $\wedge$ or $\vee$ using $\rightarrow$ and $\perp$.

The semantics is defined via Kripke models $\mathcal{M} = (U, R, \xi)$ that fulfill certain restrictions. Firstly, R is a preorder on U, and secondly, the valuation function $\xi : \mathrm{PROP} \rightarrow \mathfrak{P}(U)$ is monotone in the sense that for every $p \in \mathrm{PROP}$, $a, b \in U$: if $a \in \xi(p)$ and aRb, then $b \in \xi(p)$. We will call such models *intuitionistic*.

Given an intuitionistic model $\mathcal{M} = (U, \leqslant, \xi)$ and a state $s \in U$, the *satisfaction relation for intuitionistic logics* $\models_I$ is defined as follows.

$$\begin{array}{lll}
\mathcal{M}, s \not\models_I \perp & & \\
\mathcal{M}, s \models_I p & \text{iff} & s \in \xi(p),\ p \in \mathrm{PROP}, \\
\mathcal{M}, s \models_I \varphi \wedge \psi & \text{iff} & \mathcal{M}, s \models_I \varphi \text{ and } \mathcal{M}, s \models_I \psi, \\
\mathcal{M}, s \models_I \varphi \vee \psi & \text{iff} & \mathcal{M}, s \models_I \varphi \text{ or } \mathcal{M}, s \models_I \psi, \\
\mathcal{M}, s \models_I \varphi \rightarrow \psi & \text{iff} & \forall n \geqslant s :\ \text{if } \mathcal{M}, n \models_I \varphi \text{ then } \mathcal{M}, n \models_I \psi
\end{array}$$

An important property of intuitionistic logic is the monotonicity property: if $\mathcal{M}, s \models_I \varphi$ then $\forall n \geqslant s$ holds $\mathcal{M}, n \models_I \varphi$, for all formulas φ.

int. logic	modal companion	frame properties
BPL	K4	transitive
IPC	S4	preorder
KC	S4.2	directed preorder
LC	S4.3	linear preorder
PC	S5	equivalence relation

Fig. 2. Intuitionistic logics, their modal companions, and the common frame properties

A formula φ is *satisfied* by an intuitionistic model $\mathcal{M}$ in state s iff $\mathcal{M}, s \models_I \varphi$. Intuitionistic propositional logic IPC is the set of $\mathcal{IPC}$-formulas that are satisfied by every intuitionistic model.

Notice that IPC is a proper subset of the tautologies in classical propositional logic PC.[1] The superintuitionistic (or intermediate) logics KC and LC are also subsets of the tautologies in classical propositional logic, but proper supersets of IPC. Syntactically, KC results from adding the weak law of the excluded third $\neg a \vee \neg\neg a$ to IPC. Its semantics is defined by Kripke frames that are directed preorders—similar as for S4.2. LC (also called Gödel-Dummett logic) results syntactically from adding $(a \rightarrow b) \vee (b \rightarrow a)$ to IPC. Its semantics is defined by Kripke frames that are linear preorders–similar as for S4.3. The logic BPL is Visser's basic propositional logic [7]. Its semantics is defined by transitive (not necessarily reflexive) Kripke models with monotone valuation functions. Hence it holds that BPL $\subseteq$ IPC. Finally, the classical propositional logic PC can syntactically be seen as IPC plus the law of the excluded third $a \vee \neg a$. Its semantics is defined by Kripke frames that are equivalence relations–similar as for S5. Notice that in a Kripke frame being an equivalence relation and having a monotone valuation function, all equivalent states satisfy exactly the same formulas. Therefore, evaluating a formula φ in a state w in such a model is the same as evaluating φ in the classical propositional sense under the assignment in which exactly those variables p with $w \in \xi(p)$ are set to true.

The Gödel-Tarski translation (see e.g. [2, p.96]) maps any $\mathcal{IPC}$-formula α to a modal formula by inserting a $\Box$ before every implication and every atom. For a formula α, let α_{GT} be its Gödel-Tarski translation. The goal of this translation is to preserve validity. I.e., α is a theorem for IPC (resp. BPL, KC, LC, PC) iff α_{GT} is a theorem for S4 (resp. K4, S4.2, S4.3, S5). Therefore, S4 (resp. K4, S4.2, S4.3, S5) is called a *modal companion* of IPC (resp. BPL, KC, LC, PC). Figure 2 gives an overview about the intuitionistic logics and their modal companions used here. The Gödel-Tarski translation also preserves satisfaction in the different logics.

Lemma 1. *Let α be a formula from $\mathcal{IPC}$, and an intuitionistic $\mathcal{M}$ with state s. Then $\mathcal{M}, s \models_I \alpha$ if and only if $\mathcal{M}, s \models_M \alpha_{GT}$.*

---

[1] The satisfiable formulas in intuitionistic logic are the same as in classical propositional logic.

Model Checking Problems. This paper examines the model checking problems L-MC for logics L whose formulas are evaluated on Kripke models with different properties.

Problem:	L-MC
Input:	$\langle \varphi, \mathcal{M}, s \rangle$, where φ is an L-formula, $\mathcal{M} = (U, R, \xi)$ is a Kripke model for L, and $s \in U$ is a state
Question:	Is φ satisfied by $\mathcal{M}$ in state s ?

We assume that formulas and Kripke models are encoded in a straightforward way. This means, a formula is given as a text, and the graph (U, R) of a Kripke model is given by its adjacency matrix that takes $|U|^2$ bits. Therefore, only finite Kripke models can be considered.

Notice that all instances $\langle \varphi, \mathcal{M}, s \rangle$ of IPC-MC have a graph (U, R) contained in $\mathcal{M}$ that is a preorder. Instances without this property can be assumed to be rejected. The same holds for S4-MC and $S4_1$-MC. Accordingly, KC-MC, S4.2-MC, and $S4.2_1$-MC (resp. LC-MC and S4.3-MC) have instances only where the graph underlying the model is a directed preorder (resp. linear preorder). Since we only consider finite models, every directed preorder must have a maximal element. Therefore, it can be easily decided whether the model has the order property under consideration.

Complexity. We assume familiarity with the standard notions of complexity theory as, e. g., defined in [13]. In particular, we will show results for the classes $\mathsf{LOGDCFL}$ and P. The notion of reducibility we use is the logspace many-one reduction $\leq_m^{\log}$. The Gödel-Tarski translation can be seen as a reduction between the model checking problems for intuitionistic logics and their modal companions, namely BPL-MC $\leq_m^{\log}$ K4-MC, IPC-MC $\leq_m^{\log}$ S4-MC, KC-MC $\leq_m^{\log}$ S4.2-MC, LC-MC $\leq_m^{\log}$ S4.3-MC, and PC-MC $\leq_m^{\log}$ S5-MC. The respective reducibilities also hold for the model checking problems for formulas with any restricted number of variables.

$\mathsf{LOGDCFL}$ is the class of sets that are $\leq_m^{\log}$-reducible to deterministic context-free languages. It is also characterized as sets decidable by deterministic Turing machine in polynomial-time and logarithmic space with additional use of a stack. The inclusion structure of the classes under consideration is as follows.

$$\mathsf{NC}^1 \subseteq \mathsf{L} \subseteq \mathsf{LOGDCFL} \subseteq \mathsf{AC}^1 \subseteq \mathsf{P}$$

L denotes logspace, the formula value problem for propositional logic is complete for NC^1 (= alternating logarithmic time) [5], and the model checking problem for IPC_1 is complete for AC^1 (= alternating logspace with logarithmically bounded number of alternations) [10].

P-complete problems. Chandra, Kozen, and Stockmeyer [11] have shown that the Alternating Graph Accessibility Problem AGAP is P-complete. In [12] it is mentioned that P-completeness also holds for a bipartite version.

An *alternating graph* $G = (V, E)$ is a bipartite directed graph where $V = V_\exists \cup V_\forall$ are the partitions of V. Nodes in $V_\exists$ are called *existential* nodes, and nodes in $V_\forall$ are called *universal* nodes. The property $apath_G(x, y)$ for nodes $x, y \in V$ is defined as follows.

1) $apath_G(x, x)$ holds for all $x \in V$
2a) for $x \in V_\exists$: $apath_G(x, y)$ iff $\exists z \in V_\forall : (x, z) \in E$ and $apath_G(z, y)$
2b) for $x \in V_\forall$: $apath_G(x, y)$ iff $\forall z \in V_\exists :$ if $(x, z) \in E$ then $apath_G(z, y)$

The problem AGAP consists of directed bipartite graphs G and nodes s, t that satisfy the property $apath_G(s, t)$. Notice that in bipartite graphs existential and universal nodes are strictly alternating.

Problem: AGAP
Input: $\langle G, s, t\rangle$, where G is a directed bipartite graph
Question: does $apath_G(s, t)$ hold?

Theorem 1. *[11, 12]* AGAP *is* P*-complete under* $\leq_{\mathrm{m}}^{\log}$*-reductions.*

For our purposes, we need an even more restricted variant of AGAP. We claim that the graph is *sliced*. An *alternating slice graph* $G = (V, E)$ is a directed bipartite acyclic graph with a bipartitioning $V = V_\exists \cup V_\forall$, and a further partitioning $V = V_1 \cup V_2 \cup \cdots \cup V_m$ (*m slices*, $V_i \cap V_j = \emptyset$ if $i \neq j$) where $V_\exists = \bigcup_{i \leq m, i \text{ odd}} V_i$ and $V_\forall = \bigcup_{i \leq m, i \text{ even}} V_i$, such that $E \subseteq \bigcup_{i=1,2,\ldots,m-1} V_i \times V_{i+1}$ — i.e. all edges go from slice V_i to slice V_{i+1} (for $i = 1, 2, \ldots, m-1$). Finally, we claim that all nodes in a slice graph excepted those in the last slice V_m have outdegree > 0.

Problem: ASAGAP
Input: $\langle G, s, t\rangle$, where $G = (V_\exists \cup V_\forall, E)$ is a slice graph with slices $V_1, \ldots, V_m$, and $s \in V_1 \cap V_\exists$, $t \in V_m \cap V_\forall$
Question: does $apath_G(s, t)$ hold?

It is not hard to see that this version of the alternating graph accessibility problem remains P-complete.

Lemma 2. ASAGAP *is* P*-complete under* $\leq_{\mathrm{m}}^{\log}$*-reductions.*

Sketch of Proof. ASAGAP is in P, since it is a special case of AGAP, that is known to be in P, and since instances $\langle G, s, t\rangle$ where G is not a slice graph or $s \notin V_1 \cap V_\exists$ or $t \notin V_m \cap V_\forall$ can easily be identified.

In order to show P-hardness of ASAGAP, it suffices to find a reduction AGAP $\leq_{\mathrm{m}}^{\log}$ ASAGAP. For an instance $\langle G, s, t\rangle$ of AGAP where G has n nodes it is straightforward to construct an instance $\langle G_n, s', t'\rangle$ of ASAGAP using the considerations from above. If $\langle G_n, s', t'\rangle \in$ ASAGAP, then there exists a tree being a subgraph of G_n, that witnesses this fact. This tree can directly be transformed to a witness for $\langle G, s, t\rangle \in$ AGAP. If $\langle G, s, t\rangle \in$ AGAP, this is also be witnessed by a (finite) tree T that can be seen to consist of copies of nodes and edges of G. This tree can be trimmed in a way that on every path from the root to a leaf, every node appears at most once. Hence T induces a tree that witnesses $\langle G_n, s', t'\rangle \in$ ASAGAP. □

3 Lower bounds

We now give hardness results for the model checking problem. The P-hardness proofs use logspace reductions from the P-hard problem AsAgap (Lemma 2). The slice graph is transformed to a frame to be used in an instance of the model checking problem. Since the semantics of the logics under consideration are defined by Kripke models with frames that are transitive (and reflexive), we need to produce frames that are transitive (and reflexive). The straightforward way would be to take the transitive closure of a slice graph. But we cannot compute the transitive closure of a directed graph in logspace. Fortunately, slice graphs can easily be made transitive by adding all edges that "jump" from a node to a node that is at least two slices higher. Clearly, the resulting graph is not anymore a slice graph, but it is a transitive supergraph of the transitive closure of the slice graph. We then will use the valuation function in order to let us rediscover in which slice a state is.

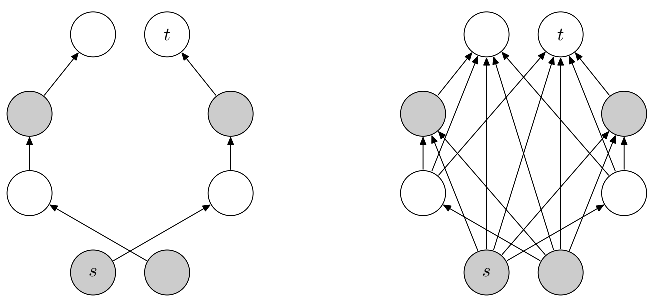

Fig. 3. A slice graph and its pseudo-transitive closure

Definition 1. *Let* $V_{\geq i} := \bigcup_{j=i,i+1,\ldots,m} V_j$*, and* $V_{\leq i} := \bigcup_{j=1,2,\ldots,i} V_j$*. The* pseudo-transitive closure *of a slice graph* $G = (V, E)$ *with* $V = V_1 \cup \ldots \cup V_m$ *is the graph* $G' = (V, E')$ *where*

$$E' \quad := \quad E \ \cup \bigcup_{i=1,2,\ldots,m-2} V_i \times V_{\geq i+2} \quad .$$

The reflexive and pseudo-transitive closure *of* G *is the graph* $G'' = (V, E'')$ *where*

$$E'' \quad := \quad E' \ \cup \ V \times V.$$

An example for a slice graph and its pseudo-transitive closure is shown in Figure 3.

Theorem 2. KC-Mc — *i.e. the model checking problem for* KC — *is* P*-hard.*

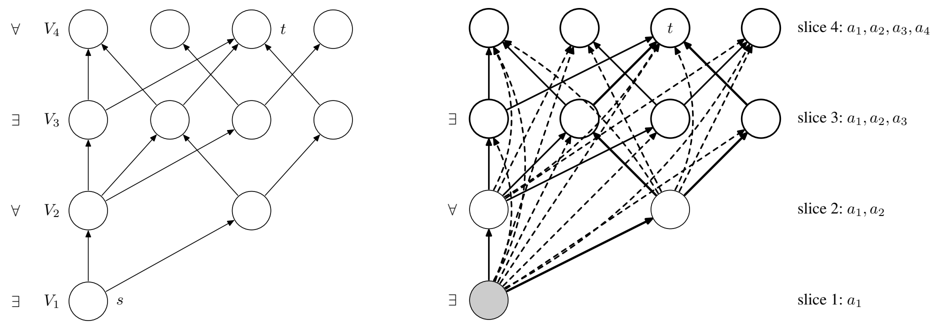

Fig. 4. A slice graph G, and the model $\mathcal{M}_G$ as constructed in the proof of Theorem 2. Pseudo-transitive edges are drawn dashed, and reflexive edges are not drawn for simplicity.

Sketch of Proof. We show AsAgap $\leq_{\mathrm{m}}^{\log}$ KC-Mc. The P-hardness of KC-Mc then follows from Lemma 2.

For simplicity, we informally sketch the ideas for the reduction AsAgap $\leq_{\mathrm{m}}^{\log}$ IPC-Mc. Given an AsAgap instance $\langle G, s, t\rangle$ where G has m slices, let (U, R) be its reflexive and pseudo-transitive closure. The valuation function ξ is defined for variables $t, a_1, \ldots, a_m$ as follows. t holds exactly in the state t of the graph ($\xi(t) = \{t\}$), and the variables $a_1, \ldots, a_i$ hold in slice i (for $i = 1, 2, \ldots, m$) ($\xi(a_i) = V_{\geq i}$) This yields the Kripke model $\mathcal{M}_G = (U, R, \xi)$. Figure 4 shows a slice graph G with $m = 4$ slices and Kripke model $\mathcal{M}_G = (U, R, \xi)$ that is transformed from it. The fat lines indicate that $\mathit{apath}_G(s, t)$ holds. The graph (U, R) is the reflexive and pseudo-transitive closure of G. The blue lines in Figure 4 are the pseudo-transitive edges, the reflexive edges are not depicted. The valuation function ξ is defined so that variable t holds exactly in the state t of the graph, and additionally the variables $a_1, \ldots, a_i$ hold in slice i (for $i = 1, 2, \ldots, m$).

The formulas $\psi_1, \ldots, \psi_m, \psi_{m+1}$ are inductively defined as follows.

1. $\psi_{m+1} := t$, and
2. $\psi_j := \psi_{j+1} \to a_{j+1}$ for all $j = m, m-1, \ldots, 1$.

Notice that $\psi_i = (\cdots((t \to a_{m+1}) \to a_m) \to \cdots \to a_{i+2}) \to a_{i+1}$. Therefore, ψ_i is satisfied in all slices where a_{i+1} is satisfied, i.e. the slices $V_{\geq i+1}$. In slice V_i, ψ_i and ψ_{i+1} behave like the mutual complement. Say that a state v is *good*, if $\mathit{apath}_G(v, t)$ holds, and otherwise it is *bad*. It turns out, that the good and the bad states can be distuingished using the formulas ψ_i as follows.

Claim. For all $i = 1, 2, \ldots, m$ and all $w \in V_i$ holds:
1. if i is odd: $\mathit{apath}_G(w, t)$ iff $\mathcal{M}_G, w \not\models_I \psi_{i+1}$, and
2. if i is even: $\mathit{apath}_G(w, t)$ iff $\mathcal{M}_G, w \not\models_I \psi_i$.

For our example, this means the following.

slice(s):	in every good state holds:	in every bad state holds:
4,3:	$\not\models_I t \to a_5$	$\models_I t \to a_5$
3,2:	$\models_I (t \to a_5) \to a_4$	$\not\models_I (t \to a_5) \to a_4$
2,1:	$\not\models_I ((t \to a_5) \to a_4) \to a_3$	$\models_I ((t \to a_5) \to a_4) \to a_3$
1:	$\models_I (((t \to a_5) \to a_4) \to a_3) \to a_2$	$\not\models_I \dots$

Since $\langle G, s, t\rangle \in$ AsAgap iff s is a good state, it now follows that $\langle G, s, t\rangle \in$ AsAgap if and only if $\mathcal{M}_G, s \models_I \psi_1$, i.e. $\langle \psi_1, \mathcal{M}_G, s\rangle \in$ IPC-Mc. By the simplicity of the construction it follows that AsAgap $\leq_{\mathrm{m}}^{\log}$ IPC-Mc.

In order to make this reduction work for KC-Mc, we add an additional *top*-state, to which every state is related and in which every variable is satisfied. □

It follows immediately from Lemma 1 that the model checking problem for S4.2—the modal companion of KC—is P-hard, too. In fact, we can improve the result and obtain P-hardness for the model checking problem for $\mathrm{S4}_1$—i.e. the fragment of S4 with formulas with one variable only. This result is optimal in the sense that the model checking problem for $\mathrm{S4}_0$ is easy to solve. A formula without any variables is either satisfied by every model w.r.t. S4, or it is satisfied by no model. This is because $\Diamond\top$ (resp. $\Box\top$) is satisfied by every state in every model, and $\Diamond\bot$ (resp. $\Box\bot$) is satisfied by no state in every model. Essentially, the modal operators can be ingnored and the remaining formula can be evaluated like a classical propositional formula—this problem is in NC^1 [5].

Theorem 3. $\mathrm{S4.2}_1$-Mc *is* P*-hard—i.e. the model checking problem for* S4.2 *is* P*-hard, even if we consider modal formulas with one variable only.*

Sketch of Proof. We show that AsAgap $\leq_{\mathrm{m}}^{\log}$ $\mathrm{S4.2}_1$-Mc. Since AsAgap is P-hard (Lemma 2), the P-hardness of $\mathrm{S4.2}_1$-Mc follows. For space reasons, we informally sketch the ideas for the reduction AsAgap $\leq_{\mathrm{m}}^{\log}$ $\mathrm{S4.2}_1$-Mc below.

Let $\langle G, s, t\rangle$ be an instance of AsAgap with a slice graph G with m slices (m even). First, we define the valuation function so that a holds in all nodes in all even slices. In order to be able to distinguish the goal node t from the other nodes, t gets a successor t', and t' is the only node in the new $m+1$st slice. Finally, we add a slice V_{m+2} with some nodes between which we also have edges. For all nodes in the other slices we add edges to all nodes in V_{m+2}. By the choice of edges in V_{m+2} there is now a node h that is the top node of this construction. We chose the valuation ξ for the nodes in V_{m+2} in a way that a certain formula γ' is satisfied in all states in V_{m+2}, and in all other states it is not satisfied. For the remaining slices $V_1, \dots, V_{m+1}$ it holds that $V_i \subseteq \xi(a)$ iff i is odd. Using this alternation of slices that satisfy a and that satisfy $\neg a$, we can estimate the slice to which a state belongs as follows using the inductively defined formulas δ_i (for $i = 1, 2, \dots, m$).

1. $\delta_m := \Diamond(a \wedge \neg\gamma')$
2. for odd i, $1 \leq i < m$: $\delta_i := \Diamond(\neg a \wedge \delta_{i+1})$
   for even i, $1 \leq i < m$: $\delta_i := \Diamond(a \wedge \delta_{i+1})$

For $x \in V_{\leq m}$ we now have that $\mathcal{M}_G, x \models_M \delta_i$ iff $x \in V_{\leq i}$.

The goal state t is the only state in V_m that satisfies $a \wedge \Diamond(\neg a \wedge \gamma')$. Using the δ_i formulas to verify an upper bound for the slice of a state, we can now simulate the alternating graph accessibility problem by the following formulas.

1. $\lambda_m := a \wedge \Diamond(\neg a \wedge \neg\gamma')$
2. for odd $i < m$: $\lambda_i := \neg a \wedge \Diamond(\delta_{i+1} \wedge \lambda_{i+1})$
   for even $i < m$: $\lambda_i := a \wedge \Box(\delta_{i+1} \rightarrow \lambda_{i+1})$

It follows that $\langle G, s, t\rangle \in$ AsAgap iff $\mathcal{M}_G, s \models_M \lambda_1$, i.e. $\langle \lambda_1, \mathcal{M}_G, s\rangle \in$ S4.2$_1$-Mc. Since the construction of $\mathcal{M}_G$ and λ_1 from G can be computed in logarithmic space, it follows that AsAgap $\leq_{\mathrm{m}}^{\log}$ S4.2$_1$-Mc. □

The reduction in the proof of Theorem 3 is not suitable for intuitionistic logics, since the constructed Kripke model lacks the monotonicity property of the variables. Moreover, in that proof we make extensive use of negation, that would have a very different meaning in intuitionistic logics.

In Theorem 4 we show P-hardness of the model checking problem for the modal logic K4, even if we consider formulas without any variables.

Theorem 4. K4$_0$-Mc *is* P-*hard.*

Sketch of Proof. The P-hardness of the model checking problem for the modal logic K4$_0$ can easily be obtained using the P-hardness of AsAgap from Lemma 2. The reduction from AsAgap to K4-Mc works as follows. Let $\langle G, s, t\rangle$ be an instance of AsAgap where G is a slice graph with m slices. Define $\mathcal{M}_G = (U, R, \xi)$ as follows.

– (U, R') is the pseudo-transitive closure of G.
– $R := R' \cup \{(v, v) \mid$ *for every vertex* $v \neq t$ *in the top slice* V_m *of* $G\}$.

Informally spoken, the model $\mathcal{M}_G$ is the pseudo-transitive closure of G and every state in the last slice except the state t has an edge to itself. We define φ_G as follows.

– $\alpha_i := \Diamond \ldots \Diamond\Box\bot$ with $m - i$ $\Diamond$s for $i \in \{2, \ldots, m-1\}$.
– $\varphi_{m-1} := \Diamond\Box\bot$
  for odd i, $m - 1 > i \geq 1$: $\varphi_i := \Diamond(\alpha_{i+1} \wedge \varphi_{i+1})$
  for even i, $m > i > 1$: $\varphi_i := \Box(\alpha_{i+1} \rightarrow \varphi_{i+1})$
– $\varphi_G := \varphi_1$

Notice that $\Box\bot$ is satisfied only in t because t is the only state without any successor. The subformula α_i is satisfied in state w, if there is a path in G from w to t with $m - i$ vertices. For this reason $\mathcal{M}, w \models_M \alpha_i$ implies $w \in V_{\leq i}$. With a straightforward induction it can be shown that for all $w \in V_{\leq i}$ holds: $\mathcal{M}_G, w \models_M \varphi_i$ iff $apath_G(w, t)$. Hence it follows that $\langle G, s, t\rangle \in$ AsAgap iff $\mathcal{M}_G, s \models_M \varphi_G$. □

4 Upper bounds

We give upper bounds for the complexity of the model checking problem for the logics under consideration. For S4, the model checking problem is in P [3]. By the properties of the Gödel-Tarski embedding of IPC into S4 (Lemma 1), the same upper bound follows immediately for IPC. The same holds for the more common fragments BPL and K4.

Theorem 5. *[3] The model checking problem for* K4 *and for* BPL *is in* P.

Consequently, the model checking problems for the superintuitionistic logics and their modal companions can also be solved in polynomial time. We now consider logics for which this goes even better.

Theorem 6. *The model checking problem for* LC *is in* $\mathsf{LOGDCFL}$.

Proof. The idea is as follows. Let $\mathcal{M} = (U, \leqslant, \xi)$ be an LC-model. This means that ξ is monotone and $(U, \leqslant)$ is a total preorder. For simplicity of notation we assume that $U = \{1, 2, \ldots, n\}$ and $\leqslant$ orders these states in the intuitive way, namely $1 \leqslant 2 \leqslant 3 \leqslant \ldots \leqslant n$. Because of the monotonicity of intuitionistic logic, for every formula α there exists an $i_\alpha \in \{1, 2, \ldots, n, n+1\}$ such that α is not satisfied in states $1, 2, \ldots, i_\alpha - 1$ and α is satisfied in states $i_\alpha, i_\alpha + 1, \ldots, n$. If $i_\alpha = n + 1$, then α is not satisfied in states $1, 2, \ldots, n$. We define a function g that maps formulas to this value. This function can inductively be defined as follows.

(1) $g(\bot) = n + 1$
(2) for atoms $\alpha = a$: $g(a) = \min\big(\{i \mid i \in \xi(a)\} \cup \{n+1\}\big)$
(3) for $\alpha = \beta \wedge \gamma$: $g(\beta \wedge \gamma) = \max(g(\beta), g(\gamma))$
(4) for $\alpha = \beta \vee \gamma$: $g(\beta \vee \gamma) = \min(g(\beta), g(\gamma))$
(5) for $\alpha = \beta \rightarrow \gamma$: $g(\beta \rightarrow \gamma) = \begin{cases} g(\gamma), & \text{if } g(\beta) < g(\gamma) \\ 1, & \text{otherwise} \end{cases}$

In order to decide $\mathcal{M}, 1 \models_I \alpha$ we calculate $g(\alpha)$ and decide whether this value equals 1. The calculation of $g(\alpha)$ can be done by a depth first search through the formula that we consider here as a tree. The "leaves" of this tree are variables resp. $\bot$. The g-values of these leaves can easily be computed in logarithmic space by inspecting the valuation function ξ. Every internal node of this tree represents a subformula of α. The g-value of each of these nodes can be computed using the g-values of its sons as described by the inductive definition of g above. Altogether, this search can be performed deterministically in polynomial time within logarithmic space and an additional stack. This shows that the model checking problem for LC is in $\mathsf{LOGDCFL}$. □

The model checking problem for KC_1 can be reduced to that of LC_1, and by Theorem 6 it also has $\mathsf{LOGDCFL}$ as upper bound. The reduction relies on algebraic properties of KC_1 according to [14, 15] and is left out here for space reasons.

Theorem 7. *The model checking problem for* KC_1 *is in* $\mathsf{LOGDCFL}$.

We obtain the same upper bound for S5-Mc.

Proposition 1. *The model checking problem for* S5 *is in* $\mathsf{LOGDCFL}$.

Sketch of Proof. Let $\langle \varphi, \mathcal{M}, s\rangle$ be an instance of S5-Mc for $\mathcal{M} = (U, R, \xi)$. Then R is a total relation on U. Therefore, every subformula of φ that begins with a modal operator (i.e. a subformula of the form $\Box\alpha$ or $\Diamond\alpha$) is either satisfied in all states of U or in no state of U. Now, φ can be evaluated as follows. First, evaluate the subformulas $\Box\alpha$ and $\Diamond\alpha$, where α is a propositional formula without any modal operators. In order to do this, check whether α is satisfied in every resp. in one state of U. This can be done in logspace. Replace these evaluated subformulas in φ by the propositional constants according to their satisfaction and evaluate the resulting formula. This must be repeated until one obtained a propositional formula that can straightforwardly be evaluated in the actual state.

This process can be implemented using a top down search through the formula, during which propositional formulas have to be evaluated in the states of U. The whole process then takes polynomial time, logarithmic space, and uses a stack for the top down search. This shows that S5-Mc can be solved in $\mathsf{LOGDCFL}$. □

In Theorem 8 we show NC^1-completeness of the model checking problem for the modal logic S4, even if we consider formulas without any variables. We sketch a proof for the upperbound. The NC^1-hardness follows immediately from [5].

Theorem 8. *The model checking problem for* $\mathrm{S4}_0$ *is* NC^1*-complete.*

Sketch of Proof. Notice that the S4 frames are reflexive and transitive. It is not possible to distinguish differents states in a reflexive and transitive frame with a variable free formula. Hence $\mathrm{S4}_0$ contains exactly all variable free formulas that can be satisfied by a reflexive and transitive Kripke model. For an $\mathrm{S4}_0$-Mc instance $\langle \mathcal{M}, \varphi\rangle$ it suffices to check whether $\varphi \in \mathrm{S4}_0$. Because we can not distinguish differents states, modal operators can be ignored. We define the operator free version φ_{of} of the $\mathrm{S4}_0$ formula φ as follows.

- $p_{of} = p$ for $p \in \{\bot, \top\}$
- $(\alpha \to \beta)_{of} = \alpha_{of} \to \beta_{of}$
- $(\Diamond\alpha)_{of} = \alpha_{of}$

It holds for an arbitrary $\mathcal{M}$ that $\langle \mathcal{M}, \varphi\rangle \in \mathrm{S4}_0$-Mc iff φ_{of} evaluates to true. Hence from [5] follows directly that $\mathrm{S4}_0$-Mc is NC^1-complete. □

5 Conclusion

The upper and lower bounds from the last sections (Theorems 2, 3, 4, and 5) combine to the following completeness results.

Theorem 9. *The following problems are* P*-complete.*

1. $K4_0$-Mc*—i.e. the model checking problems for* K4 *and formulas without variables.*
2. $K4_1$-Mc*,* $S4_1$-Mc*,* $S4.2_1$-Mc*—i.e. the model checking problems for* K4*,* S4 *resp.* S4.2 *and formulas with one variable only.*
3. KC-Mc*,* IPC-Mc*,* BPL-Mc*,* S4.2-Mc*,* S4-Mc*,* K4-Mc*—i.e. the model checking problems for* KC*,* IPC*,* BPL*,* S4.2*,* S4*, and* K4*.*

The one variable fragment IPC_1 of IPC is already deeply studied (see [16]). Recently it was shown that model checking for IPC_1 is AC^1-complete [10]. Our P-hardness proof of model checking for IPC uses an arbitrary number of variables. Rybakov [17] has shown that the tautology problem for the two variable fragment IPC_2 of IPC is already PSPACE-complete. This indicates that it is interesting to study whether model checking for IPC_2 is already P-complete.

O'Connor [18] gives a tautology-preserving translation from IPC formulas to those with two variables only. It is an open problem, whether such a translation to IPC_1 exists. From Theorem 2 and Proposition 7 follows, that we can exclude this for model checking for KC.

Theorem 10. KC-Mc $\not\leq_{\mathrm{m}}^{\log}$ KC_1-Mc*, unless* $\mathsf{P} \subseteq \mathsf{LOGDCFL}$*.*

The Gödel-Tarski translation from intuitionistic logic into S4 and the PSPACE-hardness of the tautology problem for IPC brought up the question for a "translation" from S4 into intuitionistic logic. In fact, this translation is expressed in terms of a reduction in [19]. Our results on the P-hardness of the model checking problem for S4.2 for formulas with one variable only (Theorem 3) and the contrasting LOGDCFL upper bound for KC_1 (Proposition 7) shows that those translations cannot omit the use of additional variables (unless $\mathsf{P} \subseteq \mathsf{LOGDCFL}$).

Theorem 11. $S4.2_1$-Mc $\not\leq_{\mathrm{m}}^{\log}$ KC_1-Mc*, unless* $\mathsf{P} \subseteq \mathsf{LOGDCFL}$*.*

At all, the LOGDCFL upper bounds for the model checking for LC, KC_1, and S5 are not really satisfactory. A LOGDCFL computation (polynomial time and logarithmic space with an additional stack) allows to explore a formula in a top down manner. This seems to be a very natural way to evaluate a formula. It is very surprising, that for classical propositional logic the stack is not needed [4, 5]. We conjecture that this is also possible for S5, and Proposition 1 could accordingly be improved. For KC_1, one can conclude from [14, 15] that there are only 7 equivalence classes of formulas, and only 3 types of models–all states of the model satisfy a, no state satisfies a, resp. all others. The third type is the type that makes the difference to classical propositional logic. Nevertheless, we expect that the LOGDCFL upper bound for KC_1 (Proposition 7) can be improved.

Notice that the logics KC_1 and LC_1 are the same. In [15] it is shown that $S4.3_1$—their modal companion—has infinitely many equivalence classes of formulas. Therefore it seems possible to find a lower bound for model checking for $S4.3_1$ that is above the upper bound for KC_1 and LC_1.

Acknowledgements. The authors thank Steve Awodey for his introduction to intuitionistic logic and many helpful discussions, Matthias Kramer for discussing predecessors of the proofs of Theorems 2 and 6, Vitezslav Svejdar for helpful discussions about intuitionistic logic, and Thomas Schneider for his support. The authors specially thank an anonymous referee for her/his idea to improve Theorem 3 by saving one variable.

References

1. van Dalen, D.: Logic and Structure. 4th edn. Springer, Berlin, Heidelberg (2004)
2. Chagrov, A., Zakharyaschev, M.: Modal Logic. Clarendon Press, Oxford (1997)
3. Fischer, M.J., Ladner, R.E.: Propositional dynamic logic of regular programs. J. Comput. Syst. Sci. **18**(2) (1979) 194–211
4. Lynch, N.A.: Log space recognition and translation of parenthesis languages. J. ACM **24**(4) (1977) 583–590
5. Buss, S.R.: The Boolean formula value problem is in ALOGTIME. In: Proc. 19th STOC, ACM Press (1987) 123–131
6. Dummett, M., Lemmon, E.: Modal logics between S4 and S5. Zeitschrift für Mathematische Logik und Grundlagen der Mathematik **14**(24) (1959) 250–264
7. Visser, A.: A propositional logic with explicit fixed points. Studia Logica **40** (1980) 155–175
8. Ladner, R.: The computational complexity of provability in systems of modal propositional logic. SIAM Journal on Computing **6**(3) (1977) 467–480
9. Spaan, E.: Complexity of Modal Logics. PhD thesis, Department of Mathematics and Computer Science, University of Amsterdam (1993)
10. Mundhenk, M., Weiß, F.: The model checking problem for intuitionistic logic with one variable is AC^1-complete. Unpublished manuscript (2010)
11. Chandra, A.K., Kozen, D., Stockmeyer, L.J.: Alternation. Journal of the Association for Computing Machinery **28** (1981) 114–133
12. Greenlaw, R., Hoover, H.J., Ruzzo, W.L.: Limits to Parallel Computation: P-Completeness Theory. Oxford University Press, New York (1995)
13. Papadimitriou, C.H.: Computational Complexity. Addison-Wesley, Reading, MA (1994)
14. Nishimura, I.: On formulas of one variable in intuitionistic propositional calculus. J. of Symbolic Logic **25** (1960) 327–331
15. Makinson, D.: There are infinitely many Diodorean modal functions. J. of Symbolic Logic **31**(3) (1966) 406–408
16. Gabbay, D.M.: Semantical investigations in Heyting's intuitionistic logic. D.Reidel, Dordrecht, Boston, London (1981)
17. Rybakov, M.N.: Complexity of intuitionistic and Visser's basic and formal logics in finitely many variables. In: Papers from the 6th conference on "Advances in Modal Logic", College Publications (2006) 393–411
18. O'Connor, M.: Embeddings into free Heyting algebras and translations into intuitionistic propositional logic. In: Proc. of the Int. Symp. on Logical Foundations of Computer Science, Berlin, Heidelberg, Springer-Verlag (2007) 437–448
19. Fernandez, D.: A polynomial translation of S4 into intuitionistic logic. J. of Symbolic Logic **71**(3) (2005) 989–1001