

Making Knowledge Explicit: How Hard It Is

Vladimir Brezhnev^a, Roman Kuznets^{b,1}

^a*Laboratory of Logical Problems in Computer Science,
Faculty of Mechanics and Mathematics, Moscow State University,
Vorob'evy Gory, Moscow, 119992, Russia*

^b*Ph.D. Program in Computer Science, CUNY Graduate Center,
365 Fifth Avenue, New York, NY 10016, USA*

Abstract

Artemov's logic of proofs LP is a complete calculus of propositions and proofs, which is now becoming a foundation for the evidence-based approach to reasoning about knowledge. Additional atoms in LP have form $t : F$, read as “ t is a proof of F ” (or, more generally, as “ t is an evidence for F ”) for an appropriate system of terms t called proof polynomials. In this paper, we answer two well-known questions in this area. One of the main features of LP is its ability to realize modalities in any S4-derivation by proof polynomials thus revealing a statement about explicit evidences encoded in that derivation. We show that the original Artemov's algorithm of building such realizations can produce proof polynomials of exponential length in the size of the initial S4-derivation. We modify the realization algorithm to produce proof polynomials of at most quadratic length. We also found a modal formula, any realization of which necessarily requires self-referential constants of type $c : A(c)$. This demonstrates that the evidence-based reasoning encoded by the modal logic S4 is inherently self-referential.

1 Introduction

The logic of proofs LP was introduced by Artemov as an explicit counterpart of Gödel's logic of provability S4. The main idea of LP was to replace atoms $\Box F$, read as “ F is provable,” by atoms $t : F$, understood as “ t is a proof of F ,”

Email addresses: brezhnev@yasenevo.ru (Vladimir Brezhnev),
RKuznets@gc.cuny.edu (Roman Kuznets).

¹ The author was supported in part by the Robert E. Gilleece fellowship from the CUNY Graduate Center.

with t being a proof object. The logic of proofs LP is supplied with an arithmetical semantics and enjoys completeness with respect to it (Artemov, [2]). LP is proved to be decidable (Mkrtychev, [9]). A complete overview can be found in [4]. Recent papers [5,8,3] show that LP could serve as a basis for evidence-based reasoning.

The logic of proofs provided a long sought provability semantics for S4 via interpretation in Peano arithmetic. This semantics appeared as a result of the realization procedure described in [1,2], which allows to realize an arbitrary cut-free S4-derivation by a corresponding LP-derivation. In this paper, we analyze and optimize this realization procedure and also show that S4-reasoning is inherently self-referential. The main results described in this paper are as follows:

- (1) It is possible to perform realization with at most a polynomial overhead for the modified version of the procedure (Brezhnev, Kuznets, Sect. 4).
- (2) There are theorems of S4 such that any of their realizations requires self-referential constants (Kuznets, Sect. 5).

Sect. 2 provides basic knowledge of the logics we use, and Sect. 3 gives a full-length discussion of the realization procedure and modifications necessary to make its complexity polynomial.

2 Main Definitions and Facts

Our main goal is to analyze the procedure that realizes modalities in S4-derivations by proof terms of the logic of proofs LP and to optimize this procedure in terms of complexity. To help the reader follow the details of the realization, we give here a complete list of axioms and rules of both theories.

Here is the Hilbert-style formulation of the logic of proofs we use (see [2]):

Definition 1 *The language of the logic of proofs LP contains*

- *the language of classical propositional logic which includes propositional variables, truth constants \top and \perp , and boolean connectives*
- *proof variables x_0, \dots, x_n, \dots , proof constants c_0, \dots, c_n, \dots*
- *functional symbols: monadic $!$, binary \cdot and $+$*
- *operator symbol of type “term:formula”*

The logic of proofs LP has the following axioms:

A0 *Finite set of axiom schemes of classical propositional logic*²

² To be absolutely precise, we use the propositional part of system Hc from [10].

A1	$t:F \rightarrow F$	“reflection”
A2	$t:(F \rightarrow G) \rightarrow (s:F \rightarrow (t \cdot s):G)$	“application”
A3	$t:F \rightarrow !t:(t:F)$	“proof checker”
A4	$s:F \rightarrow (s+t):F, \quad t:F \rightarrow (s+t):F$	“sum”

Inference rules are

R1	$(F \rightarrow G), F \vdash G$	“modus ponens”
R2	$\vdash c:A$, if A is an axiom A0–A4 and c is a proof constant	“axiom necessitation”

Proof terms built from proof variables and proof constants by means of functional symbols $!$, \cdot , and $+$ are called proof polynomials.

We will need the following well-known facts about LP (see [1, Lemma 2.17], [2, Lemma 5.4]).

Lemma 2 (Lifting Lemma) *Let $x_1:B_1, \dots, x_n:B_n \vdash F$, then there exists a term $t = t(x_1, \dots, x_n)$ such that $x_1:B_1, \dots, x_n:B_n \vdash t:F$.*

PROOF. Let ℓ be an LP-derivation of F from hypotheses $x_1:B_1, \dots, x_n:B_n$. By induction on the length of ℓ , we construct a term t and a new derivation ℓ^{lift} of $t:F$ from the same hypotheses in the following way: we replace each formula G in ℓ by a sequence of formulas that ends with $s:G$ for some term s . A formula G in ℓ can be obtained in four ways:

- (1) $G = A$, where A is an axiom of LP. Then in ℓ^{lift} we replace G by an instance $c:A$ of the axiom necessitation rule for a fresh proof constant c .
- (2) $G = x_i:B_i$ is one of the hypotheses. Then in ℓ^{lift} we replace G by the following three formulas:

$$x_i:B_i, \quad x_i:B_i \rightarrow !x_i:x_i:B_i, \quad !x_i:x_i:B_i .$$

The last formula is the desired lifted version of G .

- (3) G is obtained by *modus ponens* from $E \rightarrow G$ and E . By induction hypothesis, ℓ^{lift} contains $s_1:(E \rightarrow G)$ and $s_2:E$ for some proof polynomials s_1 and s_2 . We append ℓ^{lift} by the following three formulas:

$$s_1:(E \rightarrow G) \rightarrow (s_2:E \rightarrow s_1 \cdot s_2:G), \quad s_2:E \rightarrow s_1 \cdot s_2:G, \quad s_1 \cdot s_2:G .$$

The last formula is the desired lifted version of G .

- (4) $G = a:A$ is obtained by axiom necessitation, where A is an axiom and a is a proof constant. We replace G by the following three formulas:

$$a:A, \quad a:A \rightarrow !a:a:A, \quad !a:a:A .$$

Again, the last formula is the desired lifted version of G . \square

Note 1 If F is a theorem, it should be clear from the proof that term t is ground (does not contain variables). Moreover, in that case, t is built from constants and proof-checked constants by means of application only.

Lemma 3 For any proof variables \vec{x}_i and any formulas \vec{B}_i there exists a proof term $s = s(x_1, \dots, x_n)$ such that

$$\text{LP} \vdash x_1 : B_1 \wedge \dots \wedge x_n : B_n \rightarrow s : (x_1 : B_1 \wedge \dots \wedge x_n : B_n) .$$

PROOF. Obviously, $x_1 : B_1, \dots, x_n : B_n \vdash x_1 : B_1 \wedge \dots \wedge x_n : B_n$. Then by the Lifting Lemma we have just proved, there exists a term $s = s(x_1, \dots, x_n)$ such that $x_1 : B_1, \dots, x_n : B_n \vdash s : (x_1 : B_1 \wedge \dots \wedge x_n : B_n)$. Finally, simple propositional reasoning involving the Deduction Theorem³ provides the statement of the lemma.

Now we describe the Gentzen-style version of **S4** we use. We take system **G3s** from [10] and restrict it to the single modal operator \Box (also cf. note at the end of [10, Def. 9.1.3]).⁴ A similar system can be found in [7, *142.1] under the name of **S4***. In that system, an antecedent and a consequent are considered to be lists of formulas whereas in the system we use they are multisets of formulas. Since cut elimination is known to hold for this sequent calculus (see [10]), we initially formulate it without the cut rule. Another advantage of this system is that it has no structural rules; however, we have to allow some “weakening” in the $(\Rightarrow \Box)$ -rule.

Definition 4 The Gentzen-style formulation of the modal logic **S4** has the following axioms:

- (1) $S, \Gamma \Rightarrow \Delta, S$, where S is a propositional variable;
- (2) $\perp, \Gamma \Rightarrow \Delta$.

Rules are

$$\begin{array}{ccc} \frac{\Gamma \Rightarrow \Delta, A}{\neg A, \Gamma \Rightarrow \Delta} (\neg \Rightarrow) & & \frac{A, \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \neg A} (\Rightarrow \neg) \\ \\ \frac{A, B, \Gamma \Rightarrow \Delta}{A \wedge B, \Gamma \Rightarrow \Delta} (\wedge \Rightarrow) & & \frac{\Gamma \Rightarrow \Delta, A \quad \Gamma \Rightarrow \Delta, B}{\Gamma \Rightarrow \Delta, A \wedge B} (\Rightarrow \wedge) \\ \\ \frac{A, \Gamma \Rightarrow \Delta \quad B, \Gamma \Rightarrow \Delta}{A \vee B, \Gamma \Rightarrow \Delta} (\vee \Rightarrow) & & \frac{\Gamma \Rightarrow \Delta, A, B}{\Gamma \Rightarrow \Delta, A \vee B} (\Rightarrow \vee) \end{array}$$

³ LP is proved to enjoy the Deduction Theorem, see [2].

⁴ In our system, \Diamond is considered to be an abbreviation of $\neg\Box\neg$.

$$\frac{\Gamma \Rightarrow \Delta, A \quad B, \Gamma \Rightarrow \Delta}{A \rightarrow B, \Gamma \Rightarrow \Delta} (\rightarrow \Rightarrow) \qquad \frac{A, \Gamma \Rightarrow \Delta, B}{\Gamma \Rightarrow \Delta, A \rightarrow B} (\Rightarrow \rightarrow)$$

$$\frac{A, \Box A, \Gamma \Rightarrow \Delta}{\Box A, \Gamma \Rightarrow \Delta} (\Box \Rightarrow) \qquad \frac{\Box A_1, \dots, \Box A_n \Rightarrow B}{\Box A_1, \dots, \Box A_n, \Gamma \Rightarrow \Delta, \Box B} (\Rightarrow \Box)$$

Definition 5 To realize a modal formula F in the logic of proofs means to substitute proof polynomials for all occurrences of \Box in F in such a way that the result yields an LP-formula F^r .

Our goal is to realize all S4-theorems by some LP-theorems thereby explicating the provability operator \Box by specific proofs. Moreover, the scope of available realizations can be limited to the so-called *normal* realizations ([1,2]).

Definition 6 A realization r is called *normal* if all negative occurrences of modality are realized by proof variables.

Such a limitation arises from the fact that negative \Box 's constitute arguments of Skolem functions that emerge from existential quantifiers. These quantifiers are hidden in subformulas $\Box F$, understood as “there *exists* a proof of F .” Thus, it is reasonable to demand that those Skolem arguments are realized by proof variables rather than by more complicated proof polynomials. In [1,2], Artemov proved the following

Theorem 7 (Realization Theorem) If $S4 \vdash F$, then $LP \vdash F^r$ for some normal realization r .

In [1] and [2], Artemov describes slightly different variants of the realization procedure for constructing a realization r and an LP-derivation of the realized formula F^r from a given S4-derivation of F . The variants differ in that the procedure from [2] produces Gentzen-style LP-derivations whereas the procedure from [1] produces Hilbert-style derivations. The realization procedure given in this paper is based on the latter.

Kuznets showed that both variants of Artemov’s procedure give an exponential blow-up of the derivation size and Brezhnev suggested a modification to the procedure from [1] that produces at most a polynomial overhead. In the next section, we provide all the necessary details of that modification.

3 Realization Procedure

First, we divide all occurrences of modality \Box in the given derivation tree of $\Rightarrow F$ into families of related occurrences. Namely, in a rule

$$\frac{\Gamma \Rightarrow \Delta \quad [\Gamma' \Rightarrow \Delta']}{\Gamma'' \Rightarrow \Delta''},$$

each occurrence of \Box in a side formula A in the premise of the rule is related only to the corresponding occurrence of \Box in A in the conclusion of the rule. Similarly, each occurrence of \Box in an active formula of the rule, i.e. in a formula in the premise that is transformed by the rule, is related only to the corresponding occurrence of \Box in the principal formula of the rule, i.e. in the result of that transformation. As an important example, consider the $(\Box \Rightarrow)$ -rule, in which formulas A and $\Box A$ in the premise sequent are the active formulas and formula $\Box A$ in the conclusion sequent is the principal formula. Note that corresponding occurrences of \Box inside A in formulas A and $\Box A$ are related here. A *family* of \Box 's is simply an equivalence class with respect to the reflexive transitive closure of this relation.

Note that all the rules in the cut-free Gentzen system respect the polarity of formulas. Therefore, each family consists of \Box 's of the same polarity. We will call a family *positive* (*negative*) if it consists of positive (negative) \Box 's.

Since different \Box 's from the same family correspond to the same occurrence of \Box in the derived sequence, we have to realize all of them by the same proof polynomial that explicates that \Box . Moreover, all \Box 's from a negative family have to be realized by the same proof variable, due to the normality condition.

By induction on the depth of the derivation tree, we will simultaneously construct a realization and a Hilbert-style proof of the realized formula. We will also keep track of the constant specification used in that Hilbert-style proof, i.e. of all the instances of the axiom necessitation rule R2 used in it. The procedure described combines two ideas, namely the original method of translation proposed by Artemov in [1] and the methods used by Cook and Reckhow in [6].

Preliminary observations. There are only three ways of introducing new \Box 's in our system, namely

- (1) inside a side formula in an axiom,
- (2) inside a formula by which a sequent is “weakened” in a $(\Rightarrow \Box)$ -rule, or
- (3) the outer \Box in the principal formula of a $(\Rightarrow \Box)$ -rule.

A given derivation tree of $\Rightarrow F$ imposes a tree structure on each family of \Box 's whereby leaves of the family's tree are those nodes where \Box 's of this family

are first introduced (either at a leaf of the derivation tree or in some $(\Rightarrow \Box)$ -rule). We will call a positive family of \Box 's *essential*, if at least one of its leaves corresponds to a principal \Box in a $(\Rightarrow \Box)$ -rule, and *non-essential* otherwise.

Let us enumerate all $(\Rightarrow \Box)$ -rules in the tree and associate provisional variable u_i with the i th rule, more precisely with the principal \Box of that rule (in the course of the realization procedure, all these provisional variables will be replaced by proof polynomials).

First step. We choose distinct proof variables for each negative or non-essential positive family of \Box 's. All \Box 's from such a family will be realized by a proof variable corresponding to that family.

Second step. In an essential positive family of \Box 's, with $i_1 < i_2 < \dots < i_k$ being all the numbers of $(\Rightarrow \Box)$ -rules that introduce \Box 's from this family as the principle ones (case 3), all such \Box 's are initially realized by provisional term

$$u_{i_1} + u_{i_2} + \dots + u_{i_k}$$

(pluses are associated to the left). We also initialize a substitution σ that acts on those provisional variables to be the empty substitution. By the end of the realization procedure, this substitution σ will assign a certain proof polynomial to each provisional variable so that essential positive \Box 's will also be realized by proof polynomials that contain no provisional variables.

Now each modal formula A occurring in the sequent derivation is translated into an LP-formula A^r as follows: each occurrence of \Box in A is replaced by a proof polynomial $t\sigma$ that possibly contains provisional variables. Here t is the term realizing the family of that \Box , whereas σ is the current state of the substitution acting on provisional variables. This substitution is appended during the realization procedure, namely, during processing of $(\Rightarrow \Box)$ -rules.

Third step. By induction on the depth of the derivation tree of $\Rightarrow F$, for each sequent in the initial derivation, we will construct

- an LP-formula C that corresponds to that sequent;
- a proof polynomial t that contains no provisional variables;
- a Hilbert-style derivation of $t:C$.

The *external* polynomials t may seem superfluous, but they prove to be a vital part of eliminating the exponential blow-up. In the procedure, they are used while processing the $(\Rightarrow \Box)$ -rules of the initial **S4**-derivation.

A formula C is constructed in a natural way: namely, a sequent

$$A_1, A_2, \dots, A_n \Rightarrow B_1, B_2, \dots, B_m$$

is translated into a formula

$$(\dots (A_1^r \wedge A_2^r) \wedge \dots) \wedge A_n^r \rightarrow (\dots (B_1^r \vee B_2^r) \vee \dots) \vee B_m^r .^5$$

Both the antecedent and the consequent of a sequent are multisets, so the order of formulas is irrelevant in them. But normal Hilbert-style operations do not provide for such freedom. Therefore, we have to force some order on A_i^r 's and on B_j^r 's. Any order that allows for efficient sorting can be used, but this order should be uniform for all sequents. Otherwise, we won't be able to use Cook and Reckhow's idea of implementing each step of a Gentzen-style derivation by several steps of the corresponding Hilbert-style proof because the formulas on different branches of the tree simply would not match. For our purposes, let us choose the alphabetical order. An empty consequent constitutes an empty disjunction and is, therefore, translated as \perp ; an empty antecedent (empty conjunction) is translated as \top . In particular, $\Rightarrow F$ is translated as $\top \rightarrow F^r$.

For example, the two types of axioms of the **S4** sequent calculus (see Def. 4) for $\Gamma = \{A_1, \dots, A_n\}$, $\Delta = \{B_1, \dots, B_m\}$ are translated as

- (1) $A_1^r \wedge \dots \wedge A_{i-1}^r \wedge S \wedge A_i^r \wedge \dots \wedge A_n^r \rightarrow B_1^r \vee \dots \vee B_{j-1}^r \vee S \vee B_j^r \vee \dots \vee B_m^r$
- (2) $\perp \wedge A_1^r \wedge \dots \wedge A_n^r \rightarrow B_1^r \vee \dots \vee B_m^r$

(here we assume that A_k^r 's and B_l^r 's are already ordered alphabetically, $S^r = S$ becomes i th among A_k^r 's and j th among B_l^r 's upon insertion, and \perp is the first symbol of the alphabet).

Each implication C of this type (a translation of a Gentzen-style axiom into the Hilbert-style language) is clearly derivable in **LP**. Applying the Lifting Lemma to this derivation, we obtain a ground proof polynomial s and a derivation of $s:C$. This concludes the base of our induction.

Consider any propositional rule with one premise

$$\frac{\Gamma \Rightarrow \Delta}{\Gamma' \Rightarrow \Delta'} (\text{R})$$

Let C and C' be translations of $\Gamma \Rightarrow \Delta$ and $\Gamma' \Rightarrow \Delta'$ respectively. By induction hypothesis, we have a term t_C and a derivation ℓ_C of $t_C:C$. By purely propositional reasoning, there is a derivation of $C \rightarrow C'$. Using the Lifting Lemma again, we get a ground term t_R and a derivation ℓ_R of $t_R:(C \rightarrow C')$. Concatenating ℓ_C with ℓ_R and appending the result with the following sequence

$$t_R:(C \rightarrow C') \rightarrow (t_C:C \rightarrow t_R \cdot t_C:C'), \quad t_C:C \rightarrow t_R \cdot t_C:C', \quad t_R \cdot t_C:C' ,$$

⁵ For the rest of the paper, we will omit these parentheses; the convention will be that conjunctions and disjunctions are always associated to the left.

we obtain the term $t_{C'} = t_R \cdot t_C$ and the derivation $\ell_{C'}$ of $t_R \cdot t_C : C'$.

A case of a propositional rule with two premises is handled in a similar way.

Let us discuss modal rules in more detail. Consider a $(\Box \Rightarrow)$ -rule

$$\frac{A, \Box A, B_1, \dots, B_n \Rightarrow D_1, \dots, D_m}{\Box A, B_1, \dots, B_n \Rightarrow D_1, \dots, D_m} (\Box \Rightarrow) .$$

Without loss of generality, let us assume that the translation of the premise is

$$C = B_1^r \wedge \dots \wedge B_{i-1}^r \wedge A^r \wedge B_i^r \wedge \dots \wedge B_{j-1}^r \wedge x : A^r \wedge B_j^r \wedge \dots \wedge B_n^r \rightarrow D ,$$

where $D = D_1^r \vee \dots \vee D_m^r$ and x is the proof variable associated with the negative family of the outer modality in $\Box A$. Then the translation of the conclusion is

$$C' = B_1^r \wedge \dots \wedge B_{j-1}^r \wedge x : A^r \wedge B_j^r \wedge \dots \wedge B_n^r \rightarrow D .$$

Since $\text{LP} \vdash x : A^r \rightarrow A^r$, it is easy to derive $C \rightarrow C'$. Using the Lifting Lemma, we obtain a ground term $t_{(\Box \Rightarrow)}$ and a derivation $\ell_{(\Box \Rightarrow)}$ of $t_{(\Box \Rightarrow)} : (C \rightarrow C')$. The rest is the same as with the one-premise propositional rules.

The only rule that is treated differently is $(\Rightarrow \Box)$:

$$\frac{\Box A_1, \dots, \Box A_n \Rightarrow B}{D_1, \dots, D_k, \Box A_1, \dots, \Box A_n \Rightarrow \Box B, E_1, \dots, E_m} (\Rightarrow \Box) .$$

All main \Box 's in $\Box A_i$'s are negative and belong to different families, so they are realized by distinct proof variables x_i 's. Let k be the number of this $(\Rightarrow \Box)$ -rule and let its family be realized by $u_{s_1} + \dots + u_k + \dots + u_{s_l}$. By induction hypothesis, we have a term t_C and a derivation ℓ_C of

$$t_C : (x_1 : A_1^r \wedge \dots \wedge x_n : A_n^r \rightarrow B^r) .$$

By Lemma 3, we construct a term $s = s(x_1, \dots, x_n)$ and a derivation ℓ_1 of

$$x_1 : A_1^r \wedge \dots \wedge x_n : A_n^r \rightarrow s : (x_1 : A_1^r \wedge \dots \wedge x_n : A_n^r) .$$

Note that s does not contain any provisional variables. Now, it is easy to append ℓ_C together with ℓ_1 by the sequence

$$t_C : (\vec{x} : \vec{A}^r \rightarrow B^r) \rightarrow (s : (\vec{x} : \vec{A}^r) \rightarrow t_C \cdot s : B^r), \quad s : (\vec{x} : \vec{A}^r) \rightarrow t_C \cdot s : B^r$$

(we abbreviate the conjunction using a vector notation). Further, using the syllogism rule, we get a derivation of

$$x_1 : A_1^r \wedge \dots \wedge x_n : A_n^r \rightarrow t_C \cdot s : B^r .$$

Using the axiom A4 several times leads to a derivation of

$$x_1:A_1^r \wedge \dots \wedge x_n:A_n^r \rightarrow (u_{s_1}\sigma + \dots + t_C \cdot s + \dots + u_{s_l}\sigma):B^r .$$

Moreover, this derivation is easy to append to get a derivation ℓ_2 of formula C' that (modulo permutations) looks like

$$\vec{D} \wedge \vec{x}:\vec{A}^r \rightarrow (u_{s_1}\sigma + \dots + t_C \cdot s + \dots + u_{s_l}\sigma):B^r \vee \vec{E}$$

(here \vec{D} and $\vec{x}:\vec{A}^r$ stand for conjunctions, \vec{E} is a disjunction). As usual, now we use the Lifting Lemma to produce a ground term $t_{C'}$ and a derivation $\ell_{C'}$ of

$$t_{C'}:(\vec{D} \wedge \vec{x}:\vec{A}^r \rightarrow (u_{s_1}\sigma + \dots + t_C \cdot s + \dots + u_{s_l}\sigma):B^r \vee \vec{E}) .$$

Here lies the main improvement over the original procedure from [1]. This modification is what makes the whole procedure polynomial in the size of the initial S4-derivation: while lifting ℓ_2 , there is no need to lift its initial part ℓ_C since the only formula we use in the second part is $t_C:(\vec{x}:\vec{A}^r \rightarrow B^r)$. But this formula is easily lifted by adding to ℓ_C the following two formulas

$$t_C:(\vec{x}:\vec{A}^r \rightarrow B^r) \rightarrow !t_C:t_C:(\vec{x}:\vec{A}^r \rightarrow B^r) \quad \text{and} \quad !t_C:t_C:(\vec{x}:\vec{A}^r \rightarrow B^r) ,$$

the latter being the desired lifted version. This modified procedure also produces a ground term because t_C is ground.

In the original algorithm from [1], every time a $(\Rightarrow \Box)$ -rule is processed, the number of formulas in the Hilbert-style derivation is multiplied by a constant factor since most formulas in the initial derivation are replaced by three formulas in the lifted one. Thus, the size of the Hilbert-style derivation grows exponentially in the number of $(\Rightarrow \Box)$ -rules, as opposed to the polynomial growth in the modified version. To illustrate this difference consider

Example 8

$$\frac{\frac{\frac{S \Rightarrow S}{\Rightarrow S \rightarrow S}}{\Rightarrow \Box(S \rightarrow S)}}{\Rightarrow \Box\Box(S \rightarrow S)}$$

Omitting several irrelevant technicalities, our procedure first produces a ground external term t , a term t_1 , and a proof of formula $t:t_1:(S \rightarrow S)$, that corresponds to the third sequent. Processing the fourth sequent calls for a new external term t' and a term t_2 , such that $\text{LP} \vdash t':t_2:t_1:(S \rightarrow S)$. It is immediate that t_2 may be taken to be the previous external term t . This is done by the original procedure and the modified one as well. The way they construct term t' is drastically different. The original procedure suggested lifting the whole derivation of $t:t_1:(S \rightarrow S)$, making the derivation about three times longer.

In the modified procedure, we notice that t' can be taken to be $!t$ so that the length of the derivation stays almost the same.

Now, we append σ by a new substitution: $\sigma = \sigma + \{u_k \leftarrow t_C \cdot s\}$, and apply this substitution throughout the derivation.⁶ After that, there are no occurrences of u_k left in our derivation. As a result, we got rid of one provisional variable.

Final Touch. At the end of the procedure, the whole derivation tree of $\Rightarrow F$ is translated. By that time, all $(\Rightarrow \square)$ -rules have been processed and there are no provisional variables left. Thus, F^r is simply an LP-formula; moreover, we have a Hilbert-style derivation of $t: (\top \rightarrow F^r)$ for some ground term t . The following four formulas append that derivation to get the desired realization F^r :

$$t: (\top \rightarrow F^r) \rightarrow (\top \rightarrow F^r), \quad \top \rightarrow F^r, \quad \top, \quad F^r .$$

4 Complexity of the Realization Procedure

In this section, we evaluate the complexity of the realization procedure from Sect. 3. We show that the procedure gives at most a polynomial overhead with respect to the size of the initial S4-derivation.

First, we state several lemmas instrumental in evaluating the complexity of the realization procedure.

Lemma 9 *The size of a full Hilbert-style derivation corresponding to an instance of the syllogism rule used on formulas A , B , and C of lengths a , b , and c respectively, i.e.*

$$\frac{A \rightarrow B \quad B \rightarrow C}{A \rightarrow C} \text{ (Syl) } ,$$

is a linear function in a , b , and c .

It is not generally true that the size of the derivation grows linearly under the use of deduction or lifting.

Remark 10 *It is well known, that each deduction expands the size (number of formulas) in a proof by a constant factor, thus making the size exponential in the number of deductions. Thorough observation of the proof of the Lifting Lemma shows the same tendency. Moreover, the size of the term constructed in the proof is the size of the original derivation (modulo a multiplicative constant). Therefore, just the size of terms constructed by the consecutive*

⁶ LP is known to be closed under substitutions, see [2].

applications of the Lifting Lemma is already exponential in the number of applications.

However, the following lemmas may be proved via a thorough analysis of both procedures.

Definition 11 A set \mathcal{H} of Hilbert-style derivations is called an $[f(n), g(n)]$ -set if each derivation in \mathcal{H} that consists of $O(f(n))$ formulas has size of $O(g(n))$.

Lemma 12 Let \mathcal{H} be an $[f(n), g(n)]$ -set of derivations ℓ_i of $\Gamma_i, A_i \vdash B_i$ such that size of formulas in each derivation of $O(f(n))$ formulas does not exceed $O(h(n))$. (Formula A_i is considered to be present in ℓ_i .) Then set \mathcal{H}^{ded} of derivations ℓ_i^{ded} of $\Gamma_i \vdash A_i \rightarrow B_i$ obtained from ℓ_i 's via the deduction procedure is an $[f(n), g(n) + f(n)h(n)]$ -set.

The two most common instances of this lemma used in our evaluation are

Corollary 13 (1) If \mathcal{H} is an $[n, n^2]$ -set such that each derivation of $O(n)$ formulas consists of formulas of size $O(n)$, then \mathcal{H}^{ded} is an $[n, n^2]$ -set.
(2) If \mathcal{H} is an $[n^2, n^3]$ -set such that derivation of $O(n^2)$ formulas consists of formulas of size $O(n)$, then \mathcal{H}^{ded} is an $[n^2, n^3]$ -set.

Lemma 14 Let \mathcal{H} be an $[f(n), g(n)]$ -set of derivations ℓ_i of $(\vec{x})_i : (\vec{A})_i \vdash B_i$ such that each formula in ℓ_i serves as a premise to $O(1)$ modus ponens rules and size of formulas in each derivation of $O(f(n))$ formulas does not exceed $O(h(n))$. Then set $\mathcal{H}^{\text{lift}}$ of derivations ℓ_i^{lift} of $(\vec{x})_i : (\vec{A})_i \vdash t_i : B_i$ obtained from ℓ_i 's via the lifting procedure is an $[f(n), g(n) + (f(n) + h(n))h(n)]$ -set such that size of t_i for derivation ℓ_i of $O(f(n))$ formulas is $O(f(n))$.

The restriction that each formula is used only a limited number of times as a premise to a *modus ponens* rule is necessary because the size of t_i is linear in the number of steps in the initial Hilbert-style derivation represented as a tree. Normally, we are allowed to reuse formulas, so our derivations are more of a DAG than a tree. Nevertheless, this restriction is not too binding. All the derivations used in the realization procedure trivially satisfy it. Let us call such derivations *good*.

Corollary 15 (1) If \mathcal{H} is an $[n, n^2]$ -set such that each derivation of $O(n)$ formulas is good and consists of formulas of size $O(n)$, then $\mathcal{H}^{\text{lift}}$ is an $[n, n^2]$ -set. The size of the constructed terms is $O(n)$ for a derivation of $O(n)$ formulas.
(2) If \mathcal{H} is an $[n^2, n^3]$ -set such that each derivation of $O(n^2)$ formulas is good and consists of formulas of size $O(n)$, then $\mathcal{H}^{\text{lift}}$ is an $[n^2, n^4]$ -set. The size of the constructed terms is $O(n^2)$ for a derivation of $O(n^2)$ formulas.

In [6], Cook and Reckhow showed that a propositional Hilbert system is capa-

ble of simulating propositional Gentzen-style proofs with at most a polynomial overhead. Their argument runs as follows: axioms of a Gentzen calculus are translated into tautologies; each step of a Gentzen-style derivation is realized by a certain sequence of formulas in the target Hilbert-style derivation.

In this paper, we use a similar procedure, but in addition we lift each sequence to construct an external term corresponding to that sequence. It is essential for the complexity that we never work with the whole Hilbert-style derivation, but only with the part that corresponds to the last Gentzen-style step: in particular, while lifting for the $(\Rightarrow \Box)$ -rule, we do not lift ℓ_C (see Sect. 3).

Negative and non-essential positive \Box 's are realized by proof variables, i.e each " \Box " is replaced by two symbols, "proof variable" and "colon," thereby stretching the proof at most twice. For the purposes of complexity evaluation, it is convenient to postpone the actual evaluation of the size of terms $(u_{s_1} + \dots + u_{s_l})\sigma$ till the end of the procedure and to consider them to be of the size 1. A thorough analysis of the realization procedure shows the following. Consider a set \mathcal{G} of all Gentzen-style steps other than $(\Rightarrow \Box)$; let n denote the size of a conclusion sequent for a step. Then set \mathcal{H} of corresponding Hilbert-style derivations that realize steps from \mathcal{G} is an $[n^2, n^4]$ -set modulo adding a constant number of formulas that involve external term t_C to each derivation in \mathcal{H} . The size of the external term is increased by $O(n^2)$ per Gentzen step. Consider now a $(\Rightarrow \Box)$ -step; let n denote the size of a conclusion sequent. Realizing this step may involve as many as $O(n) + O(f)$ formulas, where f is the number of $(\Rightarrow \Box)$ -rules in the whole family of the \Box introduced by the step. $O(f)$ formulas are used to derive $t_C \cdot s : B^r \vdash (u_{s_1}\sigma + \dots + t_C \cdot s + \dots + u_{s_f}\sigma) : B^r$. The size of the Hilbert-style sequence that realizes this step is also more intricate: it is $O(n^2) + O(nf) + O(f^2)$; additionally, we need to use the previous external term t_C no more than $O(n + f)$ times. After the lifting the new external term will contain only one occurrence of $!t_C$ since before the lifting the formula $t_C : C$ is used as a premise to a *modus ponens* rule only once. Therefore, the size of the external term is increased by $O(n + f)$.

Bounding $\sum_i f_i$ by the size N of the initial Gentzen-style derivation and using inequality $\sum_i n_i^k \leq (\sum_i n_i)^k = N^k$, we conclude that the size of the realizing Hilbert-style proof is $O(N^4)$ if we consider any term realizing a \Box to be of size 1. Further, all external terms have a size $O(N^2)$ and the number of formulas in the realizing proof is also $O(N^2)$. It remains to note that substituting terms $t_C \cdot s$ of a size $O(N^2)$ for each of $\leq O(N^3)$ occurrences of the corresponding provisional variable u for each of f provisional variables of the corresponding family gives at most an $fO(N^5)$ overhead per family, and the sum over all the families may again be bounded by $O(N^6)$. We proved the following

Theorem 16 *For a given Gentzen-style cut-free S4-derivation of $\Rightarrow F$ of size N , the procedure from Sect. 3 produces a realization F^r and its Hilbert-*

style LP-derivation ℓ such that

- (1) ℓ contains $O(N^2)$ formulas;
- (2) the size of ℓ is $O(N^6)$;
- (3) the sizes of external terms used in ℓ are $O(N^2)$.

Note 2 One should not mistaken the complexity of realizing a given **S4**-proof into LP, which is polynomial in the size of that proof, for the complexity of constructing such a proof from a given formula and its further realization. The latter problem is most probably exponential in the size of the formula since **S4** is known to be PSPACE-complete.

5 Self-Referentiality of Modal Logic S4

In this section, we explore a fundamental question of whether proofs encoded by modal logic **S4** are self-referential. Our main result is that the realization of some **S4**-theorems necessarily calls for a constant specification with self-referential constants, i.e. for a \mathcal{CS} that involves axiom necessitation instances of type $c:A(c)$ for an axiom $A(c)$ that contains constant c .

Theorem 17 *Let*

$$\mathcal{CS} = \{c:A \mid A \text{ is an axiom that does not contain occurrences of } c\} ,$$

the largest non-self-referential⁷ constant specification. Let S be a propositional variable. Then for any proof polynomials t and t'

$$\text{LP}_{\mathcal{CS}} \not\vdash \neg t' : \neg(S \rightarrow t : S) .$$

Corollary 18 *S4-theorem $\neg \Box \neg(S \rightarrow \Box S)$ cannot be realized in LP without self-referential constants even if we drop requirement of a normal realization.*

PROOF of Theorem 17. We prove the claim by presenting for any proof polynomials t and t' an M-model⁸ of $\text{LP}_{\mathcal{CS}}$ where $\neg t' : \neg(S \rightarrow t : S)$ is false. Thus, by completeness, $\neg t' : \neg(S \rightarrow t : S)$ cannot be a theorem of $\text{LP}_{\mathcal{CS}}$.

We will briefly review the definition of M-models. To describe an *M-model* $\mathcal{M} = (*, v, \models)$ one has to define

⁷ To be absolutely precise, only one-step self-referentiality is ruled out.

⁸ M-models were called pre-models in [9].

- (1) an *evidence function* $*$ that assigns a set $*(s)$ of LP-formulas (not necessarily true) to each proof polynomial s . An evidence function has to satisfy the following three conditions:
 - (a) if $F \in *(s)$, then $(s:F) \in *(!s)$
 - (b) if $(F \rightarrow G) \in *(s_1)$ and $F \in *(s_2)$, then $G \in *(s_1 \cdot s_2)$
 - (c) $*(s_1) \cup *(s_2) \subseteq *(s_1 + s_2)$
- (2) a *truth assignment* v that assigns a truth value to each propositional variable.

Truth relation \models is defined as follows:

- (1) $\mathcal{M} \models S$ iff $v(S) = \text{true}$
- (2) boolean connectives are classical
- (3) $\mathcal{M} \models s:F$ iff $F \in *(s)$ and $\mathcal{M} \models F$

An evidence function $*$ is called a *CS-function* if $A \in *(c)$ for each axiom necessitation instance $c : A \in \mathcal{CS}$. An M-model $\mathcal{M} = (*, v, \models)$ is called a *CS-model* if $*$ is a CS-function.

Mkrtychev in [9] proved that for any set X of conditions $F \in *(s)$ on the evidence function there exists the smallest⁹ evidence function satisfying all conditions from X . This smallest function is the termwise intersection of all functions that satisfy conditions from X ; on the other hand, each true statement $G \in *(s')$ about the smallest evidence function may be derived from the conditions from X by using rules 1a–1c from the definition of an evidence function. In particular, for any constant specification \mathcal{CS} there exists the smallest CS-function satisfying all conditions from X .

Till the end of this section by \mathcal{CS} we will mean the largest non-self-referential constant specification defined in Theorem 17. Consider any proof polynomials t and t' that might realize modalities in $\neg\Box\neg(S \rightarrow \Box S)$. We now construct a CS-model that refutes the would-be realization $\neg t' : \neg(S \rightarrow t:S)$. Let $*$ be the smallest CS-function that satisfies condition $(\neg(S \rightarrow t:S)) \in *(t')$. Let truth assignment v assign *true* to all propositional variables. Then $\mathcal{M} = (*, v, \models)$ is the desired countermodel.

For $\neg t' : \neg(S \rightarrow t:S)$ to be false, $t' : \neg(S \rightarrow t:S)$ has to be true. Since $\neg(S \rightarrow t:S)$ is evidenced by t' , it remains to show that $\neg(S \rightarrow t:S)$ is true, meaning that S has to be true, while $t:S$ has to be false. The former is guaranteed by definition of v ; the latter means that either S has to be false (but we know otherwise) or S should not be evidenced by t . To summarize, it is sufficient to show that $S \notin *(t)$ for our evidence function $*$.

⁹ Evidence function $*'$ is said to be *smaller* than function $*$ if $*'(s) \subseteq *(s)$ for each term s .

Let us define an auxiliary evidence function $*'$ to be the smallest \mathcal{CS} -function. Obviously, $*'(s) \subseteq *(s)$ for each term s (because $*$ is also a \mathcal{CS} -function).

Lemma 19 *Let s be a subterm of t , then*

- (1) *If $F \in *(s)$ then F is a theorem (of $\text{LP}_{\mathcal{CS}}$) and F does not contain occurrences of t in it.*
- (2) *If $F \in *(s) \setminus *(s)$, then F has at least one occurrence of t in it. Moreover, if F is an implication, then $F = (S \rightarrow t : S) \rightarrow \perp$, which is the formula evidenced by t' in $*$.*

Corollary 20 *t is not an evidence for S according to evidence function $*$.*

PROOF. Assume $S \in *(t)$. S may or may not be evidenced by t in $*'$. In the former case, according to Lemma 19.1, S has to be a theorem, which it is not; in the latter case, according to Lemma 19.2, S has to contain t , which it does not. Contradiction. \square

PROOF of Lemma 19. We prove both claims by induction on complexity of subterm s .

- Case 1 (s is a proof constant c)** (1) Only axioms are evidenced by proof constant c in $*'$; all axioms are theorems. On the other hand, these axioms cannot contain occurrences of c because \mathcal{CS} is not self-referential; c is a subterm of t , so these axioms cannot contain t either.
- (2) Nothing new is evidenced by c in $*$ compared to $*'$ unless $t' = c$; in this latter case $\neg(S \rightarrow t : S) \in *(c) \setminus *(c)$, but this formula contains t and is the implication $(S \rightarrow t : S) \rightarrow \perp$.¹⁰

- Case 2 (s is a proof variable x)** (1) Nothing is evidenced by proof variables in $*'$. So claim 1 is vacuously true.
- (2) Nothing is evidenced by x in $*$ either, unless $t' = x$; in this latter case $\neg(S \rightarrow t : S) \in *(x) \setminus *(x)$; this formula satisfies all conditions of claim 2.

- Case 3 ($s = !s_1$)** (1) According to rule 1a, $F \in *'(!s_1)$ for the smallest \mathcal{CS} -function $*'$ only if $F = s_1 : G$, where $G \in *(s_1)$. By IH, G has to be a theorem that does not contain t . Since $!s_1$ is a subterm of t , term s_1 does not contain t either, so $s_1 : G$ does not contain t .

To show that $s_1 : G$ is a theorem we will use completeness. Consider any \mathcal{CS} -model \mathcal{M}' . $\mathcal{M}' \models G$ because $\text{LP}_{\mathcal{CS}} \vdash G$; G is evidenced by s_1 in \mathcal{M}' because s_1 is an evidence for G for the smallest \mathcal{CS} -function $*'$. Therefore, $\mathcal{M}' \models s_1 : G$. $s_1 : G$ is true in every \mathcal{CS} -model, therefore $\text{LP}_{\mathcal{CS}} \vdash s_1 : G$.

¹⁰Negation $\neg A$ is an abbreviation of $A \rightarrow \perp$.

- (2) If $F \in *(!s_1) \setminus *'(!s_1)$, then either
 (a) $F = s_1 : G$, where $G \in *(s_1) \setminus *'(s_1)$ or
 (b) $t' = !s_1$ and $F = \neg(S \rightarrow t : S)$.

The latter case is trivial and similar to the already considered cases of $s = x$ and $s = c$; in the former case, by IH formula G has to contain t , thus so does $s_1 : G$. It remains to note that $s_1 : G$ is not an implication.

Case 4 ($s = s_1 \cdot s_2$) (1) According to rule 1b, $F \in *'(s_1 \cdot s_2)$ for the smallest \mathcal{CS} -function $*'$ only if there exists G such that $(G \rightarrow F) \in *'(s_1)$ and $G \in *(s_2)$. By IH, $\text{LP}_{\mathcal{CS}} \vdash G \rightarrow F$ and $\text{LP}_{\mathcal{CS}} \vdash G$, so by *modus ponens* $\text{LP}_{\mathcal{CS}} \vdash F$. By IH, $G \rightarrow F$ does not contain t ; hence neither does F .

- (2) If $F \in *(s_1 \cdot s_2) \setminus *'(s_1 \cdot s_2)$, then either
 (a) $(G \rightarrow F) \in *(s_1) \setminus *'(s_1)$ and $G \in *(s_2)$ for some formula G , or
 (b) $(G \rightarrow F) \in *(s_1)$ and $G \in *(s_2) \setminus *'(s_2)$ for some formula G , or
 (c) $t' = s_1 \cdot s_2$ and $F = \neg(S \rightarrow t : S)$.

In case a, by IH, implication $G \rightarrow F$ would have to be $(S \rightarrow t : S) \rightarrow \perp$, so $G = S \rightarrow t : S$. Being evidenced by s_2 in $*$, $S \rightarrow t : S$ might or might not be evidenced by s_2 in $*'$. In the former case, by IH, $S \rightarrow t : S$ would have to be a theorem which it is not; in the latter case, being an implication, by IH, $S \rightarrow t : S$ would have to be $(S \rightarrow t : S) \rightarrow \perp$ which it is not either. The contradiction shows that case a is impossible.

In case b, by IH, G would have to contain t , so $G \rightarrow F$ would contain t too, and hence, by IH, $(G \rightarrow F) \in *(s_1) \setminus *'(s_1)$, impossibility of which was shown in case a. Hence case b is also impossible.

Case c is trivial.

Case 5 ($s = s_1 + s_2$) (1) According to rule 1c, $F \in *'(s_1 + s_2)$ for the smallest \mathcal{CS} -function $*'$ only if $F \in *'(s_1)$ or $F \in *'(s_2)$. In either case, by IH, F has to be a theorem that does not contain t .

- (2) If $F \in *(s_1 + s_2) \setminus *'(s_1 + s_2)$, then
 (a) $F \in *(s_1) \setminus *'(s_1)$, or
 (b) $F \in *(s_2) \setminus *'(s_2)$, or
 (c) $t' = s_1 + s_2$ and $F = \neg(S \rightarrow t : S)$.

In cases a–b, by IH, F has to contain t , plus the only implication of such type is $(S \rightarrow t : S) \rightarrow \perp$.

Case c is trivial. \square (of Lemma 19)

We have shown that S is not evidenced by t , thus $\mathcal{M} \not\models \neg t' : \neg(S \rightarrow t : S)$. By completeness, $\text{LP}_{\mathcal{CS}} \not\vdash \neg t' : \neg(S \rightarrow t : S)$. \square (of Theorem 17)

Acknowledgements

The authors are grateful to their scientific advisor professor Sergei Artemov for his wise supervision. Also, we are very thankful to Vladimir Krupski for valuable insights and support and to Tatiana Yavorskaya for helpful comments. We are indebted to Galina Savukova for editing the linguistic aspect of the article.

References

- [1] Sergei N. Artemov. Operational modal logic. Technical Report MSI 95–29, Cornell University, December 1995.
- [2] Sergei N. Artemov. Explicit provability and constructive semantics. *The Bulletin of Symbolic Logic*, 7(1):1–36, March 2001.
- [3] Sergei [N.] Artemov. Evidence-based common knowledge. Technical Report TR–2004018, CUNY Ph.D. Program in Computer Science, November 2004.
- [4] S[ergei] N. Artemov. Kolmogorov and Gödel’s approach to intuitionistic logic: current developments. *Russian Mathematical Surveys*, 59(2):203–229, 2004.
- [5] Sergei [N.] Artemov and Elena Nogina. Logic of knowledge with justifications from the provability perspective. Technical Report TR–2004011, CUNY Ph.D. Program in Computer Science, September 2004.
- [6] Stephen Cook and Robert Reckhow. On the lengths of proofs in the propositional calculus (preliminary version). In *Proceedings of the Sixth Annual ACM Symposium on Theory of Computing*, pages 135–148. ACM Press, 1974.
- [7] Robert Feys. *Modal Logics*. Paris, 1965.
- [8] Melvin Fitting. The logic of proofs, semantically. *Annals of Pure and Applied Logic*, 132(1):1–25, 2005.
- [9] Alexey Mkrtychev. Models for the logic of proofs. In Sergei I. Adian and Anil Nerode, editors, *Logical Foundations of Computer Science, 4th International Symposium, LFCS’97, Yaroslavl, Russia, July 6–12, 1997, Proceedings*, volume 1234 of *Lecture Notes in Computer Science*, pages 266–275. Springer, 1997.
- [10] A. S. Troelstra and H. Schwichtenberg. *Basic Proof Theory*, volume 43 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, second edition, 2000.