On Kripke-style semantics for the provability logic of Gödel's proof predicate with quantifiers on proofs

Rostislav Yavorskiy*

June 2, 2005

Abstract

Kripke-style semantics is suggested for the provability logic with quantifiers on proofs corresponding to the standard Gödel proof predicate. It is proved that the set of valid formulas is decidable. The arithmetical completeness is still an open issue.

1 Introduction

In this paper we suggest and study a Kripke-style semantics for the provability logic with quantifiers on proofs [1].

It is easy to see that there are at least two different ways of introducing quantifiers into the language of provability logic and into the logic of proofs. The first way is to extend the set of atomic formulas with predicate symbols and introduce individual variables and quantifiers for them. In this way we will be able to express different first order principles of provability similar to $\exists xp : R(x)$, for some individual value a the sentence $R(a)$ has a proof p, or $\forall yq : S(y)$, q proves $S(b)$ for every individual b. The corresponding first order logic of proofs QLP is an extension of the first order calculus in the language enriched with the proof terms and the proof operator. It was extensively studied in [2]. Main results in this area — parallel to those in the quantified provability logic — are negative. The logic is non-axiomatizable.

In this paper we consider another kind of first order extensions of the logic of proofs. We do not change the set of atomic formulas but allow to use quantifiers over proof variables. So, the scope of our interest includes

*The work is partially supported by grants from Russian Foundation for Basic Research, Program of the President of Russia to Support the Leading Scientific Schools, Russian Science Support Foundation, Utrecht University and NWO.

the following examples of formulas: $\forall x(\neg x : A)$, $\exists x(x : A \wedge \neg x : B)$, etc. In particular, the modal provability operator $\Box A$ is expressed in this language by the formula $\Box A \rightleftharpoons \exists x(x : A)$. One can also state all the operational properties of provability expressible in Artemov's logic of proofs LP, with the exception of the operation "+" specific for multi-conclusion proof predicates, by the corresponding $\forall\exists$-sentences, e.g.

$$\begin{array}{ll} (\cdot) & \forall xy\exists z(x : (A \to B) \to (y : A \to z : B)), \\ (!) & \forall x\exists y(x : A \to y : x : A). \end{array}$$

We can also formulate further valid principles such as

$$\begin{array}{ll} \text{negative introspection} & \forall x\exists y(\neg(x : A) \to y : \neg(x : A)), \\ \text{reflection rule} & \forall x\exists y(x : \Box A \to y : A), \\ \Box\forall\text{-rule} & \Box\forall xA \to \forall x\Box A. \end{array}$$

The validity of many principles expressible in the language depend on the kinds of considered proof predicates. A general definition of a proof predicate can be found, e.g., in [1]. Specific properties of Gödel's proof predicate and its multi-conclusion version are studied in [13], [14].

In general, a *proof predicate* is an arithmetical formula $P(x, y)$ with two free variables x and y which enumerates all theorems of a given theory T. That is,

$$T \vdash \varphi \;\Leftrightarrow\; P(n, \ulcorner\varphi\urcorner) \text{ holds for some n} \in \mathbb{N},$$

where $\ulcorner\varphi\urcorner$ stands for the Gödel number of a formula φ.

Clearly, this definition is very general, so the family of all possible proof predicates is very diverse. A *Gödel-type proof predicate* $Prf_T(x, y)$ straightforwardly formalizes the classical definition of derivation in T by means of Gödel's arithmetization of syntax (see e.g. [5], [10]):

$Prf_T(x, y) \rightleftharpoons x$ is the Gödel number of a sequence of formulas, which is a derivation in T of a formula with the Gödel number y.

In the context of this paper, $Prf_T(x, \ulcorner\varphi\urcorner)$ is the intended arithmetical interpretation for $x : \varphi$, see [12].

All such predicates have several nice properties, namely, the corresponding relation is primitive recursive, it is functional (every proof proves at most one formula), monotonic ($Prf(a, \ulcorner\psi\urcorner)$ implies $a \geq \ulcorner\psi\urcorner$), every provable formula has infinitely many proofs, etc. A *multi-conclusion version* of Gödel's proof predicate is a variant of $Prf(x, y)$ which is not functional. One such version states that x is a derivation in T, and y is a formula occurring in x. See [1], [14] for more examples.

If we restrict ourselves to the class of Gödel-type proof predicates, then we are able to express the relation of graphical equality of two formulas and the usual term equality relation by the following formulas:

$$\begin{array}{rcl}\ulcorner A\urcorner = \ulcorner B\urcorner & \rightleftharpoons & \forall x(x : (\bot \to A) \leftrightarrow x : (\bot \to B)),\\ s = t & \rightleftharpoons & \ulcorner A(s)\urcorner = \ulcorner A(t)\urcorner.\end{array}$$

By use of the equality we can express the separability property of the multi-conclusion version of the Gödel proof predicate:

$$\bigwedge_{i\in I} \Box A_i \wedge \bigwedge_{i\in I, j\in J} \ulcorner A_i\urcorner \neq \ulcorner B_j\urcorner \to \exists x(\bigwedge_{i\in I} x : A_i \wedge \bigwedge_{j\in J} \neg x : B_j),$$

and the infinity of the set of proofs for a given provable formula:

$$\forall x_1 \dots x_n(x_1 : A \wedge \dots \wedge x_n : A \to \exists y(y : A \wedge y \neq x_1 \wedge \dots \wedge y \neq x_n)).$$

Some non-axiomatizability results for different versions of provability logic with quantifiers on proofs are found in [11]. However, there is a hope that the logic corresponding to the standard (functional) Gödel proof predicate is decidable. This hope is very intriguing because positive results in this area are rare. If true, this would be a significant strengthening of Solovay's arithmetical completeness theorems.

Some partial positive results in this direction are presented in [12]. There, the author showed the decidability of the so-called prenex fragment of the provability logic with quantifiers over proofs.

The solution of this problem is closely related to the task of finding adequate Kripke-style semantics for this kind of logics; see also [2], [7], [8], [9], [13] and [14] for relevant research. In this paper we present a Kripke-style semantics for the logic of Gödel's proof predicate with quantifiers on proofs and show that the set of valid formulas in the sense of this semantics is decidable. However, the completeness of the logic of Kripke models with respect to the intended arithmetical interpretation remains an open problem.

2 Definitions

2.1 Language

The language LPq includes the following:

- propositional letters P, Q, $\dots$;
- proof variables x, y, $\dots$;

- Boolean connectives $\neg$, $\wedge$, $\vee$, $\rightarrow$ for propositions;
- quantifiers $\forall$, $\exists$ for the proof variables;
- the binary proof predicate $x : F$, "x is a proof for F", where x is a proof variable, F is a formula.

Formulas are built from the propositional letters by means of the Boolean connectives, quantifiers and the proof predicate. Formulas of the form $x : F$ are called *quasi-atomic formulas.* The set of free proof variables of the formula $x : F$ consists of all the free variables of the formula F and $\{x\}$. If all occurrences of proof variables in a formula are bound by a quantifier, the formula is called a *closed formula.*

For technical purposes it is convenient to consider an extension of the language LPq with an infinite set of specific proof constants $\delta_{i,\varphi}$. Each constant has two subscripts: the first one is a natural number, the second one is a closed formula of the extended language or the empty set symbol.

Definition 1 *The language $LPqC$ is the smallest extension of the language LPq such that it contains all constants of the form $\delta_{i,\emptyset}$ and all constants of the form $\delta_{i,F}$, for any closed formula F of $LPqC$.*

For a constant $a = \delta_{i,D(b_1,\ldots,b_m)}$ we say that a is *built from $b_1, \ldots, b_m$ by means of $D(x_1, \ldots, x_m)$.*

Informally, $\delta_{i,F}$ denotes the ith proof of formula F. The choice of the constants is motivated by the following properties of a standard proof system:

- Every proof proves only one formula.
- Every provable formula has infinitely many different proofs.
- There are infinitely many objects of the same nature (reasons, sequences of formulas, Gödel numbers, etc.) that are not proofs, i.e. they prove no formulas at all.

Notation. In what follows we use capital latin letters A, B, C to denote formulas of the language $LPqC$; a, b, c run over the constants of $LPqC$; h, i, j, k, l, m, n, p, q, r, s, t run over natural numbers; x, y, z, u, v denote proof variables.

2.2 Kripke-style semantics

A Kripke model is a triple $K = \langle \mathbf{K}, \prec, \Vdash \rangle$ where

K is a nonempty set of possible worlds;

$\prec$ is a transitive conversely well-founded order on **K**;

$\Vdash$ is a forcing relation that defines truth for propositional letters in every node of **K**.

We say that w' is accessible from w if $w \prec w'$.

Now we are going to jointly define the following two notions that depend on each other:

- the property of a constant a to be active in a world w, and
- the truth of a formula A in a world w.

The dependence is the following. The constant $\delta_{i,\varphi}$ is active in w if φ is true in all the worlds reachable from w. On the other hand, the set of active constants is used to define the truth of $\exists x F(x)$. The definition is non-circular because the accessibility relation is conversely well-founded.

Definition 2 (Activeness and truth in a model) Here we simultaneously define the set of active constants and the truth relation in a world w:

- All constants are active in any world w of depth 0, i.e. if $\neg\exists v(w \prec v)$.
- Constants of the form $\delta_{i,\emptyset}$ are active in any world.
- If constants $a_1, \ldots, a_m$ are active in w and $A(a_1, \ldots, a_m)$ is true in every world v accessible from w then all constants of the form $\delta_{i,A(a_1,\ldots,a_m)}$ are active in w for any $i \in \mathbb{N}$.
- The truth of propositional letters is given in the definition of Kripke model.
- For Boolean combinations of formulas use the truth tables for the connectives.
- $w \nVdash \delta_{i,\emptyset} : A$ for any closed formula A.
- Let A and B be closed formulas. Then, $w \Vdash \delta_{i,B} : A$ iff $\delta_{i,B}$ is active in w, and A coincides graphically with B.
- $w \Vdash \exists x A(x)$ if for some constant a active in w one has $w \Vdash A(a)$.

It is said that $K \models A$ if A is true in all worlds of K.

2.3 Auxiliary definitions

In what follows σ denotes a finite set of quasi-atomic formulas.

Let A be a formula of $LPqC$. The set of all quasi-atomic subformulas of A is denoted σ_A.

A constant a of language $LPqC$ is said to be *σ-restricted* if the formulas from σ are sufficient to build a. Namely:

- All constants of the form $\delta_{i,\emptyset}$ are σ-restricted.
- If $a_1, \dots, a_m$ are σ-restricted constants and $y : D(x_1, \dots, x_m) \in \sigma$ then all constants of the form $\delta_{i,D(a_1,\dots,a_m)}$ are σ-restricted too.

A constant of the form $\delta_{i,D(a_1,\dots,a_m)}$ is called *σ-compound.* Otherwise, it is called *σ-prime.* Namely, constants of the form $\delta_{i,\emptyset}$ are σ-prime for any σ, and if formula $y : H(x_1, \dots, x_l)$ does not belong to σ then $\delta_{i,H(a_1,\dots,a_l)}$ is σ-prime too.

Now, *σ-altitude* of a constant a, $\mathsf{alt}_\sigma(a)$, is the number of steps needed to build it from a σ-prime constant. Namely:

If a is σ-prime then $\mathsf{alt}_\sigma(a) = 0$.

If $y : D(x_1, \dots, x_m)$ is in σ then

$$\mathsf{alt}_\sigma(\delta_{i,D(a_1,\dots,a_m)}) = \max\{\mathsf{alt}_\sigma(a_1), \dots, \mathsf{alt}_\sigma(a_m)\} + 1.$$

Maximal index of a constant a, $\mathsf{maxi}(a)$, is the maximal natural number that occurs in the construction of a:

$\mathsf{maxi}(\delta_{k,\emptyset}) = k,$

$\mathsf{maxi}(\delta_{i,D(a_1,\dots,a_m)}) = \max\{i, \mathsf{maxi}(a_1), \dots, \mathsf{maxi}(a_m)\}.$

Assume $y : D(x_1, \dots, x_m)$ is in σ and $a = \delta_{i,D(b_1,\dots,b_m)}$. Then we say that constants $b_1, \dots, b_m$ *occur in the construction of a on σ-depth 1.* If some constant c occurs in the construction of either of $b_1, \dots, b_m$ on σ-depth h, then it occurs in the construction of a on σ_A-depth $h + 1$.

2.4 The equivalence relation

The equivalence relation defined below plays a crucial role in the proof of the decidability.

We want that, for two tuples of constants $\vec{a} = (a_1, \dots, a_m)$ and $\vec{b} = (b_1, \dots, b_m)$ of the same length and for a fixed formula A of the language

LPq, if $\vec{a} \approx_A^n \vec{b}$ then for any formula B with m free variables $x_1, \ldots, x_m$ such that $\sigma_B \subseteq \sigma_A$ and the quantifier depth of B over σ_A is not greater than n, one has

$$w \Vdash B(a_1, \ldots, a_m) \leftrightarrow B(b_1, \ldots, b_m)$$

provided both $\vec{a}$ and $\vec{b}$ consist of active in w constants. This is the main property to be proved in the next section. First, we give the formal definition.

Definition 3 (The equivalence relation) *Let $\vec{a} = (a_1, \ldots, a_m)$ and $\vec{b} = (b_1, \ldots, b_m)$ be two tuples of constants of the language $LPqC$. Let σ be a set of quasi-atomic formulas of LPq. We say that $\vec{a} \approx_\sigma^0 \vec{b}$ if for any $i, j, k, i_1, \ldots, i_r \in \{1, \ldots, m\}$ and $D(x_1, \ldots, x_r) \in \sigma$ the following holds:*

1. *$a_i = a_j$ if and only if $b_i = b_j$;*

2. *a_k is built from $a_{i_1}, \ldots, a_{i_r}$ by means of $D(x_1, \ldots, x_r)$ if and only if b_k is built from $b_{i_1}, \ldots, b_{i_r}$ by means of $D(x_1, \ldots, x_r)$.*

Informally, the relation $\vec{a} \approx_\sigma^0 \vec{b}$ means that dependencies between the constants inside $\vec{a}$ are the same as for the elements of $\vec{b}$ in the context defined by the set of quasi-atomic formulas σ.

Now we are going to define a refinement of this relation, $\vec{a} \approx_\sigma^n \vec{b}$, that informally means that the structure of the tuples and the dependencies are the same up to some deep level of the constants' structure.

To make this formal we need to introduce the notion of *extension of a tuple $\vec{a}$ up to σ-depth d, $\mathsf{ext}_d(\vec{a})$*. First of all, $\vec{a}$ is an initial segment of $\mathsf{ext}_d(\vec{a})$ and for $d = 0$ one has $\mathsf{ext}_d(\vec{a}) = \vec{a}$. To construct $\mathsf{ext}_d(\vec{a})$ for $d > 0$ one should write down at the end of $\vec{a}$ all the constants that occur in the construction of a_1 on σ-depth 1 then all the constants that occur in the construction of a_2 on σ-depth 1 and so on until the last element of $\vec{a}$; then, one should add in the same order all the constants that occur on σ-depth 2 and so on until σ-depth d.

Note that if $d_1 < d_2$ then $\mathsf{ext}_{d_1}(\vec{a})$ is an initial segment of $\mathsf{ext}_{d_2}(\vec{a})$.

Definition 4 (Equivalence up to power n)

$$\vec{a} \approx_\sigma^n \vec{b} \quad \rightleftharpoons \quad \forall d < 2^n \ (\mathsf{ext}_d(\vec{a}) \approx_\sigma^0 \mathsf{ext}_d(\vec{b})).$$

The next lemma follows immediately from the definitions.

Lemma 2.1 (Monotonicity of the equivalence relation) *Let $\vec{a}$ and $\vec{b}$ be tuples of constants of the same length. Suppose $\sigma_1 \subseteq \sigma_2$. Then $(\vec{a} \approx_{\sigma_2}^n \vec{b})$ implies $(\vec{a} \approx_{\sigma_1}^n \vec{b})$.*

3 The algorithm of the equivalent extension

Consider now two tuples $\vec{a} = (a_1, \ldots, a_m)$ and $\vec{b} = (b_1, \ldots, b_m)$. We are going to describe an algorithm, which for a given number $n > 0$ and an arbitrary constant a_{m+1} produces a constant b_{m+1} such that

$$(\vec{a} \approx_\sigma^n \vec{b}) \;\rightarrow\; [(a_1, \ldots, a_m, a_{m+1}) \approx_\sigma^{n-1} (b_1, \ldots, b_m, b_{m+1})]. \qquad (Ext)$$

There are two main cases:

1. First, check all the constants in $\mathsf{ext}_{2^{n-1}-1}(\vec{a})$. If a_{m+1} coincides with one of them, then b_{m+1} is the corresponding element in $\mathsf{ext}_{2^{n-1}-1}(\vec{b})$.

   That was the simple case. Otherwise, we construct b_{m+1} following the σ-structure of a_{m+1}.

2. Check all the constants that occur in the structure of a_{m+1} up to depth 2^{n-1}. If any of them appear in $\mathsf{ext}_{2^{n-1}-1}(\vec{a})$ then replace it with the corresponding constant from $\mathsf{ext}_{2^{n-1}-1}(\vec{b})$ (sometimes it may require to reshuffle the indices to avoid collisions). Then, replace all the σ-prime constants in the construction of a_{m+1} by constants of the form $\delta_{i,\emptyset}$ with the minimal fresh index i. Also, replace all of the constants on depth 2^{n-1} with constants of the form $\delta_{i,\emptyset}$ (again with the minimal fresh index i).

Due to the construction of b_{m+1} the property (Ext) trivially holds.

Lemma 3.1 (Bounds on the algorithm's result) *For the parameters described above $\vec{a} = (a_1, \ldots, a_m)$, $\vec{b} = (b_1, \ldots, b_m)$, a_{m+1} and b_{m+1} the following holds:*

(1) $\mathsf{alt}_\sigma(b_{m+1}) \leq 2^{n-1} + \max[\mathsf{alt}_\sigma(b_1), \ldots, \mathsf{alt}_\sigma(b_m)]$*;*

(2) $\mathsf{maxi}(b_{m+1}) \leq (\mathsf{w}_\sigma)^{2^{n-1}} + \max[\mathsf{maxi}(b_1), \ldots, \mathsf{maxi}(b_m)]$, *where* w_σ *denotes the maximal arity of formulas in* σ*.*

(3) Any occurrence of a σ*-prime constant in* b_{m+1} *either has the form* $\delta_{i,\emptyset}$*, or occurs in the construction of a constant among* $(b_1, \ldots, b_m)$*.*

Proof. Follows from the inspection of the constructed b_{m+1}. Namely:

(1) According to the algorithm, in order to construct b_{m+1} we start with sub-constants of $(b_1, \ldots, b_m)$ and fresh constants of the form $\delta_{i,\emptyset}$, and then

follow the structure of the upper 2^{n-1} levels in the constant a_{m+1}. This naturally gives us the upper bound on the altitude of b_{m+1}.

(2) To obtain an upper bound on the maximal index of b_{m+1} it is sufficient to estimate how many sub-constants have been used in the construction of b_{m+1}. Since the number of levels in the syntactical tree of b_{m+1} is not more than 2^{n-1} and the number of branches is not more than w_σ, the number of the leaves in the tree could be roughly estimated from above by $(\mathsf{w}_\sigma)^{2^{n-1}}$.

(3) Informally, the main idea of the algorithm is the following. Take a_{m+1} and inspect its structure down to the level $2^{n-1}-1$. Replace in (the inspected part of) a_{m+1} all occurrences of sub-constants from $\mathsf{ext}_{2^{n-1}-1}(\vec{a})$ by their counterparts from $\mathsf{ext}_{2^{n-1}-1}(\vec{b})$. Then, replace the rest of the constants in a_{m+1} that occur on level $2^{n-1}-1$ by fresh constants of the form $\delta_{i,\emptyset}$. It is clear that after this procedure all σ-prime sub-constants in the constructed b_{m+1} have the required form. ☑

Lemma 3.2 (Finite set of constants to restrict quantifiers) *For any natural number n and a tuple $\vec{b} = (b_1, \ldots, b_m)$ one can compute a finite set of constants $\Omega_n(\vec{b})$ such that for any tuple $\vec{a} = (a_1, \ldots, a_m)$ such that $\vec{a} \approx^n_\sigma \vec{b}$ and any constant a_{m+1} there exists a constant b_{m+1} in $\Omega_n(\vec{b})$ such that*

$$(a_1, \ldots, a_m, a_{m+1}) \approx^{n-1}_\sigma (b_1, \ldots, b_m, b_{m+1}).$$

Proof. Follows from lemma 3.1 above. Indeed, the third item of the lemma restricts the form of σ-prime constants used to build b_{m+1}. These σ-prime constants either occur in the construction of $(b_1, \ldots, b_m)$, or have the form $\delta_{i,\emptyset}$. The first and the second items of lemma 3.1 provide the upper bounds on σ-altitude of b_{m+1} and its maximal index. As the result we get the finite set $\Omega_n(\vec{b})$ of all possible alternatives for b_{m+1}. ☑

4 Decidability of the set of active constants

In what follows we consider a fixed finite Kripke model K.

We start with a very simple observation. Since in every world w of depth 0 every constant is active, trivially we have the following.

Lemma 4.1 (Basis of induction) *In every world w of depth 0 the set of active constants is decidable.*

This lemma has the following important generalization.

Lemma 4.2 (Inductive step on w) *For any set of quasi-atomic formulas σ if the set of true formulas over σ is decidable in every world up to the depth d then the set of active σ-restricted constants is decidable for all nodes up to depth $d+1$.*

Proof. According to the definition, any constant of the form $\delta_{i,\emptyset}$ is active everywhere. So this case is trivial.

To check whether $\delta_{i,D(b_1,\ldots,b_m)}$ is active in a given world of depth $d+1$, one has to test if $b_1,\ldots,b_m$ are active in this world, and to check truth of $D(b_1,\ldots,b_m)$ in all accessible worlds. Since constants $b_1,\ldots,b_m$ are less complex (have lower σ-altitude) than constant $\delta_{i,D(b_1,\ldots,b_m)}$, the initial task is eventually reduced to computing truth values of finite number of formulas in all worlds of smaller depth. This is decidable due to the conditions of the lemma.☑

This lemma gives us dependency between the decidability of truth and the decidability of activeness of constants. Namely, if the truth is decidable then the activeness is decidable too in all worlds on one level more.

The other way around, dependency comes from the definition of truth. To define whether a formula is true in a given world w one has to restrict quantifiers by the set of w-active constants. So, in order to prove decidability of truth in a given world w it is sufficient to restrict all quantifiers in a formula with a restricted (finite) set of constant and then to use decidability of the set of active constants. In such a way every formula will be algorithmically reduced to a quantifier free one.

5 Quantifier elimination

The following lemma gives us decidability for quantifier free formulas on the next level. This is the first part of the inductive step. The other part is to reduce all formulas to quantifier free formulas.

Lemma 5.1 (Inductive step for σ and w) *Suppose for all σ' such that $\sigma' \subset \sigma$ the set of true formulas over σ' is decidable for all worlds up to depth d. Then the set of quantifier free formulas over σ is decidable in all worlds of depth up to $d+1$.*

Proof. First, note that quantifier free formulas are Boolean combinations of propositional variables and quasi-atomic formulas. So, the decidability follows from the definition of the truth of formula $a : F$ in world w — it is reduced to the truth of a simpler formula F in the reachable worlds of smaller depth.☑

Now we are going to prove that for any formula A one can restrict any quantifier which is not in the scope of a proof operator by a (finite) set of σ_A-restricted constants with computable bounds on the constants' depth and maximal index.

Then, if the set of active σ_A-restricted constants is decidable in w, for any closed formula F over σ_A one can effectively find a quantifier free formula F' over σ_A such that $w \Vdash F \leftrightarrow F'$.

In order to prove that, we utilize the equivalence relation defined above and the algorithm of the equivalent extension.

Lemma 5.2 *Let $\vec{a} = (a_1, \ldots, a_m)$ and $\vec{b} = (b_1, \ldots, b_m)$ be tuples of constants of the same length m. Suppose all the constants in $\vec{a}$, $\vec{b}$ are active in w and $\vec{a} \approx^0_\sigma \vec{b}$. Then for any quantifier free formula $F(x_1, \ldots, x_m)$ over σ with m free variables one has*

$$w \Vdash F(a_1, \ldots, a_m) \leftrightarrow F(b_1, \ldots, b_m).$$

Proof. Follows immediately from the definition of the equivalence relation and the definition of truth in w.☑

Lemma 5.3 (The Ehrenfeucht-Fraissé argument) *Let w be a world of depth 0. Let $\vec{a} = (a_1, \ldots, a_m)$ and $\vec{b} = (b_1, \ldots, b_m)$ be tuples of constants of the same length m. Then for any formula $F(x_1, \ldots, x_m)$ over σ with m free variables such that the quantifier depth of F over σ is n or less one has*

$$w \Vdash F(a_1, \ldots, a_m) \leftrightarrow F(b_1, \ldots, b_m).$$

Proof. Straightforward induction by n. The case $n = 0$ is considered in the previous lemma. The inductive step follows from the properties of the algorithm of the equivalent extension. Use the standard Ehrenfeucht-Fraissé argument.☑

To prove the last lemma for any world w we need additional property of the algorithm of the equivalent extension, namely, the activeness of the produced constant b_{m+1}. Anyway, we obtain the following important fact.

Lemma 5.4 (Quantifier elimination in worlds of depth 0) *In any world w of depth 0, any LPq-sentence is effectively equivalent to a quantifier free formula of LPqC.*

Proof. Follows from lemmas 5.3, 3.2 by induction on the complexity of σ. This is also a standard part of the Ehrenfeucht-Fraissé technique [6].☑

Lemma 5.5 (Partial embedding into modal logic GL) *For any quantifier free sentence S of language LPqC one can effectively build a propositional modal formula $\mu(S)$ such that in any world w of any Kripke model one has*

$$w \Vdash S \Leftrightarrow w \Vdash \mu(S).$$

Proof. Straightforward induction on the length of S. For propositional variables the mapping is identical. The case of boolean combination is trivial. For the proof operator, $\delta_{i,A} : B$ is equivalent to $\bot$ in case A and B do not coincide. If S is $\delta_{i,A} : A$ then $\mu(S)$ is $\Box\mu(A)$.☑

Lemma 5.6 *Let σ be a finite set of quasi-atomic formulas such that formulas in σ has no occurrences of quantifiers (in the scope of the proof operator). Then for any σ-restricted constant c one can effectively build a propositional modal formula $\alpha(c)$ such that for any world w of any Kripke model*

$$w \Vdash \alpha(c) \ \Leftrightarrow \ c \text{ is active in } w.$$

Proof. Induction on the complexity of c. Basis, $\alpha(\delta_{i,\emptyset}) = \top$. The inductive step uses lemma 5.5:

$$\alpha(\delta_{i,A(c_1,\ldots,c_k)}) = \mu(A(c_1,\ldots,c_k)) \wedge \alpha(c_1) \wedge \cdots \wedge \alpha(c_k),$$

where $\mu(A(c_1,\ldots,c_k))$ is constructed as in lemma 5.5.☑

The last lemma could be slightly generalized in the following way.

Lemma 5.7 (Expressibility of activeness) *Let σ be a finite set of quasi-atomic formulas. Suppose, for any $\sigma' \subset \sigma$ there is an algorithm to transform any sentence S over σ (in the language LPqC) into a propositional modal formula $\mu(S)$ such that for any world w of any Kripke model one has*

$$w \Vdash S \ \Leftrightarrow \ w \Vdash \mu(S).$$

Then, for any σ-restricted constant c one can effectively build a propositional modal formula $\alpha(c)$ such that for any world w of any Kripke model

$$w \Vdash \alpha(c) \ \Leftrightarrow \ c \text{ is active in } w.$$

Proof. Inspect the proof of lemma 5.6 and note that if $y : A(x_1, \dots, x_k)$ is a quasi-atomic formula from σ then $A(x_1, \dots, x_k)$ is a formula over σ', where σ' is a proper subset of σ.☑

Lemma 5.8 (Correctness of the constructed constant) *Let $\vec{a} = (a_1, \dots, a_m)$ and $\vec{b} = (b_1, \dots, b_m)$ be tuples of constants of the same length m. Suppose all the constants in $\vec{a}$, $\vec{b}$ are active in w and $\vec{a} \approx^n_\sigma \vec{b}$. Then for any constant a_{m+1} active in w the corresponding constant b_{m+1} produced by the algorithm is also active in w.*

Proof. According to the definition of the activeness we need to check two properties. First, that all constants used in the construction of b_{m+1} are active. This follows from the inspection of the algorithm. Indeed, all constants used to construct b_{m+1} occur in $\vec{a}$, $\vec{b}$ or a_{m+1}. Second, one has to check that the formula used to construct b_{m+1} is valid in all accessible worlds. This could be done by induction on the complexity of the formula. It is sufficient to note that the substitution we make during construction of b_{m+1} does not affect the truth of the formulas, because we replace σ-prime constants with other σ-prime constants. ☑

Lemma 5.9 *If the set of active σ_A-restricted constants is decidable in w then the set of true formulas over σ_A is decidable in w.*

Proof. Follows from lemmas 3.2 and 5.8 by the standard technique of Ehrenfeucht-Fraissé games [6]. ☑

Lemma 5.10 (Inductive step for σ_A) *Suppose that for all σ_B such that $\sigma_B \subset \sigma_A$ the set of true formulas over σ_B is decidable in every world w. Then the set of true formulas over σ_A is also decidable in every world w.*

Proof. Joint induction on the complexity of σ_A and the depth of w. Use lemmas 4.1, 4.2 and 5.9. ☑

Theorem 1 (Decidability of truth in a model) *For any finite Kripke model K the set of true formulas of the language $LPqC$ is decidable.*

Proof. Follows from lemmas 4.1, 4.2, 5.9 and 5.10.☑

In fact, the Ehrenfeucht–Fraissé method gives us the following stronger fact. Namely, in every world w every σ_A-formula is equivalent to a boolean combination of quantifier free σ_A-formulas with constants from an appropriate set Ω mentioned in lemma 3.2 intersected with the set of w-active constants. Now, taking into account lemmas 5.5, 5.6 and 5.7 we obtain the following theorem.

Let GL denote the modal propositional provability logic [5].

Theorem 2 (Embedding into GL) *For any closed formula A of language LPq one can effectively construct a propositional modal formula φ such that for any Kripke model K with transitive conversely well-founded accessibility relation one has:*

$$K \models A \ \Leftrightarrow\ K \models \varphi.$$

Corollary 1 (Upper bound on the counter-model size) *For any closed formula A of language LPq one can compute a number n such that A is valid in all Kripke models if and only if A is valid in all Kripke models with n nodes or less.*

Proof. Follows from the similar fact for GL and the previous fact.☑

Theorem 3 (Decidability of validity) *The set of formulas of LPq valid in all Kripke-models is decidable.*

Proof. Follows form the decidability of GL and theorem 2.☑

6 Future work

Now the main task is to prove that that the set of Kripke valid formulas coincides with the set of arithmetically valid formulas. One can use here a technique of direct correspondence between Kripke models and models of arithmetic described in [12]. One can also try to prove the embedding theorem for the set of arithmetically valid formulas and then use the well-known arithmetical completeness of GL.

7 Acknowledgements

I am thankful to Sergei Artemov for drawing my attention to this problem, introducing me into the subject, and also for his regular support and encouragements. I am grateful to Albert Visser, Lev Beklemishev, Vladimir Krupski, and Tatiana Yavorskaya (Sidon) for numerous discussions on the subject and their helpful remarks.

References

[1] Sergei Artemov, *Explicit provability and constructive semantics*, Bulletin of Symbolic Logic, volume 7, No.1, pp. 1-36, 2001

[2] Sergei Artemov and Tatiana Yavorskaya. *On First Order Logic Of Proofs.* Moscow Mathematical Journal, vol. 1, No. 4, pp. 475-490, 2001.

[3] Sergei Artemov and L. Beklemishev. *Provability logic.* In D.Gabbay and F.Guenthner, editors, Handbook of Philosophical Logic, 2nd ed., volume 13, pages 229-403. Kluwer, Dordrecht, 2004.

[4] Sergei Artemov, *Problems, Spring of 2003.*
http://www.gc.cuny.edu/˜sartemov

[5] George Boolos. *The Logic of Provability.* Cambridge University Press, 1993.

[6] Jeanne Ferrante and Charles Rackoff. The computational complexity of logical theories, volume 718 of Lecture Notes in Mathematics. Springer-Verlag, 1979.

[7] Melvin Fitting, *Quantified LP.* CUNY Ph.D. Program in Computer Science Technical Reports TR-2004019, December 2004.

[8] Vladimir N. Krupski, *Operational Logic of Proofs with Functionality Condition on Proof Predicate.* Lecture Notes in Computer Science, vol. 1234 (1997). Eds. S. Adian and A. Nerode, pp. 167-177.

[9] Vladimir N. Krupski. *The single-conclusion proof logic and inference rules specification.* Annals of Pure and Applied Logic, v.110, No. 1-3, 2001.

[10] Craig Smorynski. **Self-Reference and Modal Logic.** New York, Springer Verlag, 1995.

[11] Rostislav Yavorsky. *Provability logic with quantifiers on proofs.* Annals of Pure and Applied Logic 113 (2002), pp. 373–387.

[12] Rostislav Yavorskiy. *On Prenex Fragment of Provability Logic with Quantifiers on Proofs.* Proceedings of the Steklov Institute of Mathematics. Vol. 242 (2003), pp. 112–124.

[13] Rostislav E. Yavorsky. *On arithmetical completeness of first-order logics of provability.* Advances in Modal Logic, vol. 3, F. Wolter, H. Wansing, M. de Rijke, and M. Zakharyaschev, eds. 2002, pp. 1-16.

[14] Rostislav E. Yavorsky. On the logic of the standard proof predicate. Springer Lecture Notes in Computer Science, vol. 1862, pp. 527-542. Proceedings of 14th International Conference of Computer Science Logic 2000.