A way to handle substitutions in first-order logic

Anders Lundstedt

2016

Abstract

I present a syntax and a substitution system for first-order arithmetic that handles substitution less naively than the standard "named variables and substitution by replacement" approach. This makes it suitable for formalization (as evidenced by a successful Coq implementation). The syntax is similar to elimination of named variables with de Bruijn levels and should generalize straightforwardly to any first-order language. I compare this approach with the standard approach by looking at their respective statements of universal elimination and the corresponding soundness proofs.

§1 Introduction

In formalized mathematics problems arise which are more or less trivially handled in non-formalized mathematics. One such problem is "substitution under binders". Non-formalized arguments tend to rely on human intuition about syntax and details involving substitution are often skipped or quickly looked over, because, well, our intuition tell us that they can safely be ignored. In formalized mathematics we do not have this option and reasoning about substitution quickly gets tedious, especially if using the standard "named variable" construction of syntax. For example, Russell O'Connor had the following comment on his Coq formalization of Gödel's first incompleteness theorem.

> Renaming bound variables turned out to be a constant source of work during development because variable names and terms were almost always abstract. In principle the variable names could conflict, so it was constantly necessary to consider this case and deal with it by renaming a bound variable to a fresh one. (O'Connor, 2005)

*

I experienced this problem firsthand when as part of a larger formalization I formalized Heyting arithmetic in Coq (Lundstedt, 2015). My solution was to use a syntax and a substitution system constructed such that any possible substitution is in a sense "safe". Thus there was no need to consider and deal with renaming cases or conditions such as terms being "free for substitution for" variables in formulas.

*

A "binder" is an operator which introduces bound variables. Examples are quantifiers in logic, abstraction in lambda calculus and differentials in calculus. The problem of "substitution under binders" is that substitution by replacement in expressions containing binders may inadvertently bind variables. An example is replacing x with y in the formula $\exists y(x \neq y)$. To deal with this, one either needs to single out the "safe" substitutions (e.g. by using predicates such as "free for substitution for") or define a more complex (than substitution by replacement) system where only "safe" substitutions are possible.

*

A common solution is to eliminate named variables using *de Bruijn indices*, introduced by de Bruijn (1972). De Bruijn also introduced another way of eliminating named variables: *de Bruijn levels*, which is more similar to my way of "naming" variables. Lescanne and Rouyer-Degli (1995) develop a lambda calculus with "explicit substitutions" based on de Bruijn levels. As they note, their choice of levels instead of indices is not the choice most commonly made. I have not seen any treatments of first-order logic using de Bruijn levels, which presumably would be similar to my solution.

*

The idea for my solution came from Erik Palmgren, who provided me with a Coq axiomatization of a specialization of fibred semantics (see e.g. Jacobs, 1999), which I further simplified and implemented as the system to be presented. I will try to present the solution in a self-contained way—in the sense that I will not presuppose familiarity with existing solutions to the problem of substitution under binders. I will present the solution for the language of arithmetic, but it should be applicable to any first-order language. Furthermore, I will only prove that substitution is well-behaved with respect to the standard interpretation, but this should also generalize straightforwardly—to any interpretation in a Tarski-style first-order structure.

*

For a point of comparison with the "named variable" approach I will use the statement and soundness proof of universal elimination. With named variables this is how we would state universal elimination:

$$\frac{\forall x A \qquad t \text{ free for substitution for } x \text{ in } A}{A[t \,/\, x]} \,.$$

The soundness of this rule follows from the following substitution lemma. Let v be a valuation of variables and let $v[t \,/\, x]$ be the valuation of variables obtained from v by replacing the valuation of x with the v-valuation of t. Then the v-interpretation of $A[t \,/\, x]$ is equivalent to the $v[t \,/\, x]$-interpretation of A, provided t is free for substitution for x in A.

*

For my solution, the statement of universal elimination is in §4. The corresponding substitution lemma and soundness proof is in §6. Neither of these involves a "free for substitution for" predicate.

*

While the main feature of my solution is an elegant substitution system, there are three other features that should be mentioned. First, the syntax should be more readable than a syntax based on de Bruijn indices. Second, all recursive definitions are structurally recursive, i.e. defined by recursion on the corresponding inductive structure (as opposed to being recursive with respect to some other well-order). (At least in Coq this allows for simpler and more straightforward definitions.) Third, terms and formulas are indexed by their "arity", where a term or formula of arity n has at most n free variables. In type theory an indexed family of sets becomes an indexed family of types. In this case this means that the type of a term or formula encodes its arity. Thus we state that a term or formula has arity n simply by giving its type, as opposed to by defining a predicate we require it to satisfy.

*

One disadvantage is that the only substitution defined is simultaneous substitution for all free variables. The only other substitution one can conveniently do is "substitution for the last variable". However at least this suffices to define and prove the soundness of a Hilbert-style deductive system (Lundstedt, 2015). Of course one could define additional substitutions, but then one would have to prove these "well-behaved" as well (by proving corresponding substitution lemmas).

§2 The basic idea

Consider the formula

$$A(x) \coloneqq \exists y(x \neq y)$$

in which two variables occur. The variable y occurs "bound" and the variable x occurs "free". How should we define $A(t)$—the substitution of a term t for the variable x in A? The lazy way is simple syntactic replacement, i.e.

$$A(t) \coloneqq \exists y(t \neq y).$$

This would allow for "bad" substitutions:

$$A(y) = \exists y(y \neq y).$$

Another solution is to rename bound variables when needed to keep these distinct from the variables in t. Then we could have e.g.

$$A(y) = \exists z(y \neq z).$$

*

Let us say that a formula or term has "arity" n (or is "n-ary") if its set of free variables is a subset of $\{x_0, \ldots, x_{n-1}\}$. Note: This does not define arity uniquely, viz. an n-ary formula will also be an $n+1$-ary formula.

*

Now more generally: Given an $n+1$-ary formula A^{n+1}, consider the n-ary formula $B^n \coloneqq \forall x_n A^{n+1}$. Assume that we have already defined "safe" simultaneous substitutions $A^{n+1}(t_0, \ldots, t_n)$ for all terms $t_0, \ldots, t_n$. Given n terms $t_0^m, \ldots, t_{n-1}^m$ each of arity m, how should we define the simultaneous substitution $B^n(t_0^m, \ldots, t_{n-1}^m)$? One solution is the following. Since x_m is a variable not free in $t_0^m, \ldots, t_{n-1}^m$ relabel the quantifier with x_m and simultaneously substitute x_m for x_n and each t_i for x_i in A^{n+1}:

$$B^n(t_0^m, \ldots, t_{n-1}^m) \coloneqq \forall x_m A^{n+1}(t_0^m, \ldots, t_{n-1}^m, x_m).$$

Note that the resulting formula has arity m, as is to be expected when simultaneously substituting each free variable in a formula for a term of arity m.

*

The choice of x_m as the new bound variable may seem arbitrary but it fits with the restricted syntax to be presented. In this syntax the arity of a term or formula is unique (which is accomplished by treating terms and formulas as "arity-indexed"). If A^{n+1} has arity $n+1$ then instead of $\forall x_n A^{n+1}$ we will write simply $\forall A^{n+1}$, the informal interpretation we have in mind being that the quantifier always and implicitly binds the variable x_n. Thus in this syntax x_m is the only "good" choice as the new bound variable (the choice in this syntax not including how to relabel the quantifier—as formally there is no such choice—but only what term to substitute for x_n in A^{n+1}).

*

How do we formally state that the simultaneous substitution outlined above is "well-behaved"? An n-ary formula A^n can be interpreted as a relation $[\![A^n]\!] \subseteq \mathbb{N}^n$. Similarly an n-ary term t^n can be interpreted as a function $[\![t^n]\!] \colon \mathbb{N}^n \to \mathbb{N}$. Substitution is "well-behaved" if for all n-ary formulas A^n, all m-ary terms $t_0^m, \ldots, t_{n-1}^m$ and all $\mathbf{y} \in \mathbb{N}^m$,

$$\mathbf{y} \in [\![A^n(t_0^m, \ldots, t_{n-1}^m)]\!] \iff ([\![t_0^m]\!](\mathbf{y}), \ldots, [\![t_{n-1}^m]\!](\mathbf{y})) \in [\![A^n]\!].$$

Thus once syntax (§3), substitution (§4) and interpretation (§5) have been formally defined I will prove this result (§6).

§3 Syntax

Definition 3.1 (Terms). For each natural number n, the set Term^n of *terms of arity n* is given inductively:

$$\frac{i < n}{x_i^n \in \mathrm{Term}^n} \qquad \frac{}{0^n \in \mathrm{Term}^n} \qquad \frac{t \in \mathrm{Term}^n}{S(t) \in \mathrm{Term}^n}$$

$$\frac{u, v \in \mathrm{Term}^n}{(u + v) \in \mathrm{Term}^n} \qquad \frac{u, v \in \mathrm{Term}^n}{(u \cdot v) \in \mathrm{Term}^n} \ .$$

Notation: The superscript in 0^n may be dropped when the arity is unimportant or can be inferred from the context. I will drop parentheses when possible, using the standard order of operations.

*

Example (Some variables). There are no variables of arity 0. The only variable of arity 1 is x_0^1. The variables of arity 2 are x_0^2 and x_1^2. The variables of arity 3 are x_0^3, x_1^3 and x_2^3.

*

Example (Some terms). Some terms of arity 0:

$$0^0, \ \ S(0^0), \ \ S(S(0^0)), \ \ 0^0 \cdot 0^0, \ \ S(0^0) + S(S(0^0)).$$

Some terms of arity 2:

$$0^2, \ \ S(x_1^2), \ \ S(x_1^2) + 0^2, \ \ S(x_0^2 + x_1^2), \ \ x_0^2 \cdot S(x_1^2).$$

*

Definition 3.2 (Atomic formulas). For each natural number n, the set Atom^n of *atomic formulas of arity n* is given inductively:

$$\frac{}{\bot^n \in \mathrm{Atom}^n} \qquad \frac{u, v \in \mathrm{Term}^n}{u \doteq v \in \mathrm{Atom}^n} \ .$$

Notation: The superscript in $\bot^n$ may be dropped when the arity is unimportant or can be inferred.

*

Example (The atomic Peano axioms). The defining equations for multiplication and addition may be stated as atomic formulas of arities 1 and 2:

$$\begin{aligned} x_0^1 + 0^1 &\doteq x_0^1, \\ x_0^2 + S(x_1^2) &\doteq S(x_0^2 + x_1^2), \\ x_0^1 \cdot 0^1 &\doteq 0^1, \\ x_0^2 \cdot S(x_1^2) &\doteq x_0^2 \cdot x_1^2 + x_0^2. \end{aligned}$$

*

Definition 3.3 (Formulas). For each natural number n, the set Form^n of *formulas of arity n* is given inductively:

$$\frac{A \in \mathrm{Atom}^n}{A \in \mathrm{Form}^n} \qquad \frac{A, B \in \mathrm{Form}^n}{(A \wedge B) \in \mathrm{Form}^n} \qquad \frac{A, B \in \mathrm{Form}^n}{(A \vee B) \in \mathrm{Form}^n}$$

$$\frac{A, B \in \mathrm{Form}^n}{(A \to B) \in \mathrm{Form}^n} \qquad \frac{A \in \mathrm{Form}^{n+1}}{\exists A \in \mathrm{Form}^n} \qquad \frac{A \in \mathrm{Form}^{n+1}}{\forall A \in \mathrm{Form}^n} \ .$$

Notation: $\neg A$ will be short for $A \to \bot$ and $u \neq v$ will be short for $\neg u \doteq v$. I will drop parentheses when possible, using the convention that $\to$ associates to the right and that $\forall$ and $\exists$ bind stronger than $\wedge$ and $\vee$, which bind stronger than $\to$.

*

Example (Non-atomic Peano axioms sans induction). The two non-atomic non-schematic Peano axioms can be stated as formulas of arities 1 and 2:

$$0^1 \neq S(x_0^1),$$
$$S(x_0^2) = S(x_1^2) \to x_0^2 = x_1^2.$$

They can of course also be stated as sentences, i.e. as formulas of arity 0:

$$\forall(0^1 \neq S(x_0^1)),$$
$$\forall\forall(S(x_0^2) = S(x_1^2) \to x_0^2 = x_1^2).$$

*

Example (Equality axioms). Since substitution is tricky, equality is preferably not axiomatized by a substitution scheme, but as an equivalence relation congruent with the function symbols:

$$x_0^1 \doteq x_0^1,$$
$$x_0^2 \doteq x_1^2 \to x_1^2 \doteq x_0^2,$$
$$x_0^3 \doteq x_1^3 \to x_1^3 \doteq x_2^3 \to x_0^3 \doteq x_2^3,$$
$$x_0^2 \doteq x_1^2 \to S(x_0^2) \doteq S(x_1^2),$$
$$x_0^4 \doteq x_1^4 \to x_2^4 \doteq x_3^4 \to x_0^4 + x_2^4 \doteq x_1^4 + x_3^4,$$
$$x_0^4 \doteq x_1^4 \to x_2^4 \doteq x_3^4 \to x_0^4 \cdot x_2^4 \doteq x_1^4 \cdot x_3^4.$$

§4 Substitution

From now on I will use superscripts to indicate arities, e.g. t^n will denote an n-ary term.

*

Definition 4.1 (Substitution in terms and vectors of terms). For a term t^n and a vector $\mathbf{t} = (t_0^m, \ldots, t_{n-1}^m)$, the *substitution of* $\mathbf{t}$ *in* t^n, written $t^n[\mathbf{t}]$, is a term of arity m defined by recursion on the structure of t^n:

$$\begin{aligned} 0^n[\mathbf{t}] &\coloneqq 0^m, \\ x_i^n[\mathbf{t}] &\coloneqq t_i^m, \\ S(t)[\mathbf{t}] &\coloneqq S(t[\mathbf{t}]), \\ (u+v)[\mathbf{t}] &\coloneqq u[\mathbf{t}] + v[\mathbf{t}], \\ (u \cdot v)[\mathbf{t}] &\coloneqq u[\mathbf{t}] \cdot v[\mathbf{t}]. \end{aligned}$$

For a vector $\mathbf{u} = (u_0^n, \ldots, u_{m-1}^n)$ and a vector $\mathbf{v}$, the *substitution of* $\mathbf{v}$ *in* $\mathbf{u}$, written $\mathbf{u}[\mathbf{v}]$, is defined as the substitution of $\mathbf{v}$ component-wise in $\mathbf{u}$:

$$\mathbf{u}[\mathbf{v}] \coloneqq (u_0^n[\mathbf{v}], \ldots, u_{m-1}^n[\mathbf{v}]).$$

*

Recall the intended interpretation of $B^n \coloneqq \forall A^{n+1}$ as $\forall x_n A^{n+1}$ in informal syntax and the corresponding definition of substitution:

$$B^n(t_0^m, \ldots, t_{n-1}^m) \coloneqq \forall x_m A^{n+1}(t_0^m, \ldots, t_{n-1}^m, x_m).$$

However the corresponding formulation in our arity-indexed syntax would not typecheck since x_m^{m+1} has arity $m+1$ while each t_i^m has arity m. The solution is to "promote" each t_i^m, increasing their arity by 1. To define this operation we could do a straightforward structural recursion or we could use substitution with a "promoted identity vector".

*

Definition 4.2 (Promotion of terms and vectors of terms). For a term t^n the corresponding term of arity $n+1$ is the *promoted term*

$$\widehat{t^n} \coloneqq t^n[(x_0^{n+1}, \ldots, x_{n-1}^{n+1})].$$

For a vector $\mathbf{t}^n$ the corresponding vector of arity $n+1$ is the *vector of promoted terms*

$$\widehat{\mathbf{t}^n} \coloneqq \mathbf{t}^n[(x_0^{n+1}, \ldots, x_{n-1}^{n+1})].$$

*

Definition 4.3 (Substitution in formulas). For a formula A^n and a vector $\mathbf{t}$ of n terms of arity m, the *substitution of* $\mathbf{t}$ *in* A^n, written $A^n[\mathbf{t}]$, is a formula of arity m defined by recursion on the structure of A^n:

$$\bot^n[\mathbf{t}] \coloneqq \bot^m,$$

$$
\begin{aligned}
(u \doteq v)[\mathbf{t}] &\coloneqq u[\mathbf{t}] \doteq v[\mathbf{t}],\\
(A \wedge B)[\mathbf{t}] &\coloneqq A[\mathbf{t}] \wedge B[\mathbf{t}],\\
(A \vee B)[\mathbf{t}] &\coloneqq A[\mathbf{t}] \vee B[\mathbf{t}],\\
(A \to B)[\mathbf{t}] &\coloneqq A[\mathbf{t}] \to B[\mathbf{t}],\\
(\exists A)[\mathbf{t}] &\coloneqq \exists A[(\hat{\mathbf{t}}, x_m^{m+1})],\\
(\forall A)[\mathbf{t}] &\coloneqq \forall A[(\hat{\mathbf{t}}, x_m^{m+1})].
\end{aligned}
$$

*

Example (Substitution under binder).

$$(\forall(x_0^2 \doteq x_1^2 \to x_1^2 \doteq x_0^2))[(S(x_4^5))] = \forall(S(x_4^6) \doteq x_5^6 \to S(x_4^6) \doteq x_5^6).$$

*

Definition 4.4 (Substitution for the last variable). *Substitution of a term* t^{n+1}, respectively *of a term* t^n, *for the last variable* in a formula A^{n+1} is given notation and defined by

$$A^{n+1}[t^{n+1}] \coloneqq A^{n+1}[(x_0^{n+1}, \ldots, x_{n-1}^{n+1}, t^{n+1})],$$

respectively by

$$A^{n+1}[t^n] \coloneqq A^{n+1}[(x_0^n, \ldots, x_{n-1}^n, t^n)].$$

*

Note that $A^{n+1}[t^{n+1}]$ has arity $n+1$, while $A^{n+1}[t^n]$ has arity n.

*

Example (The induction scheme). For every formula A^{n+1} the following formula is an instance of the induction scheme.

$$A^{n+1}[0^n] \to \forall(A^{n+1} \to A^{n+1}[S(x_n^{n+1})]) \to \forall A^{n+1}.$$

*

Example (Universal elimination). One way to state universal elimination is:

$$\frac{\forall A^{n+1}}{A^{n+1}[t^n]}\ .$$

(Note that since both premise and conclusion belong to Term^n we can also write universal elimination "Hilbert-style" as the implication $\forall A^{n+1} \to A^{n+1}[t^n]$.)

§5 The standard interpretation

Definition 5.1 (Standard term interpretation and standard term vector interpretation). For a term t^n the *standard term interpretation* is a function $[\![t^n]\!]\colon \mathbb{N}^n \to \mathbb{N}$ defined by recursion on the structure of t^n:

$$\begin{aligned}
[\![0]\!](\mathbf{y}) &\coloneqq 0,\\
[\![x_i^n]\!](y_0,\dots,y_{n-1}) &\coloneqq y_i,\\
[\![S(t)]\!](\mathbf{y}) &\coloneqq [\![t]\!](\mathbf{y}) + 1,\\
[\![u + v]\!](\mathbf{y}) &\coloneqq [\![u]\!](\mathbf{y}) + [\![v]\!](\mathbf{y}),\\
[\![u \cdot v]\!](\mathbf{y}) &\coloneqq [\![u]\!](\mathbf{y}) \cdot [\![v]\!](\mathbf{y}).
\end{aligned}$$

For a vector $\mathbf{t} = (t_0^m,\dots,t_{n-1}^m)$ the *standard term vector interpretation* is a function $[\![\mathbf{t}]\!]\colon \mathbb{N}^m \to \mathbb{N}^n$ defined as the component-wise interpretation of $\mathbf{t}$:

$$[\![\mathbf{t}]\!](\mathbf{y}) \coloneqq ([\![t_0^m]\!](\mathbf{y}),\dots,[\![t_{n-1}^m]\!](\mathbf{y})).$$

*

Definition 5.2 (Standard formula interpretation). For a formula A^n the *standard formula interpretation* is a relation $[\![A^n]\!]$ on $\mathbb{N}^n$ defined by recursion on the structure of A^n:

$$\begin{aligned}
[\![\bot]\!] &\coloneqq \emptyset,\\
[\![u \doteq v]\!] &\coloneqq \{\mathbf{y} : [\![u]\!](\mathbf{y}) = [\![v]\!](\mathbf{y})\},\\
[\![A \wedge B]\!] &\coloneqq [\![A]\!] \cap [\![B]\!],\\
[\![A \vee B]\!] &\coloneqq [\![A]\!] \cup [\![B]\!],\\
[\![A \to B]\!] &\coloneqq \{\mathbf{y} : \mathbf{y} \in [\![A]\!] \implies \mathbf{y} \in [\![B]\!]\},\\
[\![\exists A]\!] &\coloneqq \{\mathbf{y} : \exists y \in \mathbb{N}, (\mathbf{y}, y) \in [\![A]\!]\},\\
[\![\forall A]\!] &\coloneqq \{\mathbf{y} : \forall y \in \mathbb{N}, (\mathbf{y}, y) \in [\![A]\!]\}.
\end{aligned}$$

*

Recall that $\mathbb{N}^0$ is (most conveniently defined as) the unit set, so that an interpretation of a formula of arity 0 is either the unit set or the empty set. (The particular choice of unit set will of course not matter.)

*

Definition 5.3 (Standard validity). A formula A^n is *valid under the standard interpretation* if $[\![A^n]\!] = \mathbb{N}^n$.

§6 The substitution lemma

In the following substitution lemmas I have left the involved arities and vector lengths implicit. I encourage the reader to manually add these and make sure

the equations typecheck. (In each equation there are two degrees of freedom to choosing arities and vector lengths.)

*

Theorem 6.1 (Term substitution lemma).

$$[\![t[\mathbf{t}]]\!](\mathbf{y}) = [\![t]\!]([\![\mathbf{t}]\!](\mathbf{y})).$$

Proof. Straightforward induction on t. □

*

Theorem 6.2 (Substitution lemma).

$$\mathbf{y} \in [\![A[\mathbf{t}]]\!] \iff [\![\mathbf{t}]\!](\mathbf{y}) \in [\![A]\!].$$

Proof. Induction on A. The case $A = \bot$ is trivial. The case $A = u \doteq v$ follows from the term substitution lemma. The propositional cases are straightforward applications of induction hypotheses. The case for the existential quantifier is of course similar to the case for the universal quantifier, for which we have the following equivalences, each explained below.

$$\begin{aligned}
&(1) \quad \mathbf{y} \in [\![(\forall A^{n+1})[\mathbf{t}^m]]\!] &&\iff \mathbf{y} \in [\![\forall A^{n+1}[(\widehat{\mathbf{t}^m}, x_m^{m+1})]]\!] \\
&(2) && \iff \forall y \in \mathbb{N}, (\mathbf{y}, y) \in [\![A^{n+1}[(\widehat{\mathbf{t}^m}, x_m^{m+1})]]\!] \\
&(3) && \iff \forall y \in \mathbb{N}, [\![(\widehat{\mathbf{t}^m}, x_m^{m+1})]\!](\mathbf{y}, y) \in [\![A^{n+1}]\!] \\
&(4) && \iff \forall y \in \mathbb{N}, ([\![\widehat{\mathbf{t}^m}]\!](\mathbf{y}, y), [\![x_m^{m+1}]\!](\mathbf{y}, y)) \in [\![A^{n+1}]\!] \\
&(5) && \iff \forall y \in \mathbb{N}, ([\![\mathbf{t}^m]\!](\mathbf{y}), y) \in [\![A^{n+1}]\!] \\
&(6) && \iff [\![\mathbf{t}^m]\!](\mathbf{y}) \in [\![\forall A^{n+1}]\!].
\end{aligned}$$

(1) unfolds the substitution. (2) unfolds the standard interpretation. (3) rewrites with the induction hypothesis. (4) unfolds the term vector substitution. The term substitution lemma gives $[\![\widehat{\mathbf{t}^m}]\!](\mathbf{y}, y) = [\![\mathbf{t}^m]\!](\mathbf{y})$. Together with the immediate $[\![x_m^{m+1}]\!](\mathbf{y}, y) = y$ we have (5). (6) read from right to left is the unfolding of the standard term vector interpretation. □

*

Corollary 6.1 (Soundness of universal elimination). If $\forall A^{n+1}$ is valid then $A^{n+1}[t^n]$ is valid.

Proof. Suppose $\forall A^{n+1}$ is valid, i.e.

$$[\![\forall A^{n+1}]\!] = \mathbb{N}^n,$$

i.e.

$$\forall \mathbf{y} \in \mathbb{N}^n, \mathbf{y} \in [\![\forall A^{n+1}]\!],$$

i.e.

$$\forall \mathbf{y} \in \mathbb{N}^n, \forall y \in \mathbb{N}, (\mathbf{y}, y) \in [\![A^{n+1}]\!].$$

Instantiating y with $[\![t^n]\!](\mathbf{y})$ we have

$$\forall \mathbf{y} \in \mathbb{N}^n, (\mathbf{y}, [\![t^n]\!](\mathbf{y})) \in [\![A^{n+1}]\!],$$

which by the substitution lemma is equivalent to

$$\forall \mathbf{y} \in \mathbb{N}^n, \mathbf{y} \in [\![A^{n+1}[t^n]]\!],$$

i.e.

$$[\![A^{n+1}[t^n]]\!] = \mathbb{N}^n,$$

i.e. $A^{n+1}[t^n]$ is valid. □

References

de Bruijn, N.G. (1972). "Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the Church-Rosser theorem". *Indagationes Mathematicae (Proceedings)* 75.5, pp. 381–392.

Jacobs, Bart (1999). *Categorical Logic and Type Theory*. Studies in Logic and the Foundations of Mathematics 141, Elsevier.

Lescanne, Pierre and Jocelyne Rouyer-Degli (1995). "Explicit substitutions with de Bruijn's levels". In: Jieh Hsiang editor, *Rewriting Techniques and Applications*, Lecture Notes in Computer Science 914, pp. 294–308, Springer Berlin Heidelberg.

Lundstedt, Anders (2015). "Realizability in Coq". Master's thesis, KTH, Stockholm, Sweden. URL: http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-174109.

O'Connor, Russell (2005). "Essential incompleteness of arithmetic verified by Coq". In: Joe Hurd and Tom Melham, editors, *Theorem Proving in Higher Order Logics: 18th International Conference*, *Theorem Proving in Higher Order Logics: 18th International Conference, TPHOLs 2005, Oxford, UK, August 22–25, 2005. Proceedings,* Lecture Notes in Computer Science 3603, pp. 245–260, Springer Berlin Heidelberg.