A Completeness Proof for A Regular Predicate Logic with Undefined Truth Value

Antti Valmari
Faculty of Information Technology, University of Jyväskylä, FINLAND

Lauri Hella
Faculty of Information Technology and Communication Sciences,
Tampere University, FINLAND

This is the Author's accepted manuscript (with a different LaTeX style file) of a paper in
Notre Dame J. Formal Logic 64(1): 61–93 (February 2023). Copyright © 2023 University of Notre Dame
doi:10.1215/00294527-2022-0034, link to the paper at Project Euclid

Abstract

We provide a sound and complete proof system for an extension of Kleene's ternary logic to predicates. The concept of theory is extended with, for each function symbol, a formula that specifies when the function is defined. The notion of "is defined" is extended to terms and formulas via a straightforward recursive algorithm. The "is defined" formulas are constructed so that they themselves are always defined. The completeness proof relies on the Henkin construction. For each formula, precisely one of the formula, its negation, and the negation of its "is defined" formula is true on the constructed model. Many other ternary logics in the literature can be reduced to ours. Partial functions are ubiquitous in computer science and even in (in)equation solving at schools. Our work was motivated by an attempt to explain, precisely in terms of logic, typical informal methods of reasoning in such applications.

2010 MSC: Primary 03B50 Many-valued logic, 03F03 Proof theory, general (including proof-theoretic semantics); Secondary 03B10 Classical first-order logic

Keywords: ternary logic, partial functions, completeness

1 Introduction

Classical binary first-order logic assumes that all function symbols denote total functions. This assumption repeatedly fails in everyday mathematics and in theoretical and practical computer science. As a consequence, there is extensive literature on how to deal with partial functions in logical reasoning, presenting surprisingly many diverse approaches, including [1–3, 5, 6, 8–10, 13, 15, 16, 18, 22, 23, 25, 26, 28]. We faced the problem when developing computer support for school and elementary university mathematics education [27–29]. To introduce related work and our motivation, it is useful to first present a somewhat artificial example.

Assume that a student has been asked to find the roots of $3\sqrt{|x|-1} \geq x+1$ (that is, solve $3\sqrt{|x|-1} \geq x+1$) in the case of real numbers. One possible way to start is to get rid of the absolute value operator by splitting the problem to two cases. Intuitively it seems that this can be expressed with logical connectives as follows:

$$(x < 0 \land 3\sqrt{-x-1} \geq x+1) \lor (x \geq 0 \land 3\sqrt{x-1} \geq x+1) \tag{1}$$

The values $x \leq -1$ make $3\sqrt{-x-1} \geq x+1$ yield T (that is, true), and $-1 < x < 0$ makes it undefined. Furthermore, $0 \leq x < 1$ makes $3\sqrt{x-1} \geq x+1$ undefined, $1 \leq x < 2$ and $x > 5$ result in F (false), and $2 \leq x \leq 5$ results in T. Therefore, the student should answer something to the effect of $x \leq -1 \lor 2 \leq x \leq 5$.

To formalize this reasoning in some logic, $3\sqrt{|x|-1} \geq x+1$ should be in some sense equivalent to $x \leq -1 \lor 2 \leq x \leq 5$ in that logic. The trouble begins with the fact that when $-1 < x < 1$, then $3\sqrt{|x|-1} \geq x+1$ is undefined but $x \leq -1 \lor 2 \leq x \leq 5$ yields F.

We could try to sort this out by declaring that the domain of discourse is not $\mathbb{R}$ but $\{x \in \mathbb{R} \mid |x|-1 \geq 0\}$. Unfortunately, this idea would invalidate (1), because every real number makes some atomic formula in it undefined, and we obviously want the roots be in the domain of discourse. (A similar remark was made in [22].) Furthermore, with this kind of exercises it is usually the student's responsibility to find out that some values are not roots because they make something undefined. Those values must be in the domain of discourse of the reasoning whose purpose is to find them. In brief, we want a logic that justifies, not bans, (1).

Perhaps the next idea is *negative free logic*: every atomic formula that contains an undefined term yields F [8, 20, 22, 23]. It suffers from a problem illustrated by the following example. If $\geq$ is a relation symbol in the core language and $x < y$ is defined as an abbreviation of $\neg(x \geq y)$, then $3\sqrt{|x|-1} < x+1$ yields T when $-1 < x < 1$. It is against our intention. It is also clumsy: the user of the logic would have to remember which predicate symbols yield F and which yield T when applied to undefined terms. Defining both $<$ and $\geq$ in the core language would mean to axiomatize essentially the same thing twice. It would also result in the loss of the permission to replace $\neg(t \geq t')$ and $t < t'$ by each other when t and t' are potentially undefined terms.

In computer science, the idea of *underspecification* [10] is popular. In it every function always yields a value in the intended range, but we refuse to say anything else about the value when it would be undefined in the everyday mathematics sense. From [10]: "The value of $x/0$ could be 2.4 or $20 \cdot x$ or any value in $\mathbb{R}$; we simply don't say what it is." It has a variant where functions may be proper partial, but every relation symbol always yields F or T. In the words of [26, Sect. 2.5]: "If one or both of E_1 and E_2 are undefined, then we say that the predicates $E_1 = E_2$ and $E_1 \in E_2$ are *undetermined*: we do not know whether they are true or false. This does not mean that the predicates have some intermediate status in which they are 'neither true nor false', simply that we have chosen not to say whether they are true or not."

While underspecification works well for proving programs correct, it is unsuitable for our purpose. By its very nature, it denies complete axiomatizations. It explicitly leaves it open whether, for instance, 0 is a root of $3\sqrt{|x|-1} \geq x+1$, while in mainstream mathematics the intention is that it is not a root. On the other hand, [10] makes 0 a root of $\frac{1}{x} - x = \frac{1}{2x}$.

Both underspecification and negative (and positive) free logics support a semi-formal approach to dealing with undefined terms. Under mild assumptions, there is a straightforward recursive algorithm that, for each term t, produces a formula $\lceil t \rfloor$ that yields T when t is defined, and F otherwise [3, 5, 10]. For instance, $\frac{\sqrt{x}}{x-2}$ is defined precisely when $\sqrt{x}$ is defined, $x-2$ is defined, and $x-2 \neq 0$, that is, $x \geq 0 \land x \neq 2$.

Let $\varphi(X)$ be a first-order formula containing precisely one instance of the nullary relation symbol X, and not containing other connectives and quantifiers than $\neg$, $\land$, $\lor$, $\forall$, and $\exists$. Let n be the number of those subformulas of $\varphi(X)$ that are of the form $\neg\varphi'$, where φ' contains X. Let $R(t)$ be a formula, and let $\varphi(R(t))$ denote the result of putting it in place of X.

If n is even, then for any interpretation of all other non-logical symbols than X, either $\varphi(\mathsf{F})$ yields the same truth value as $\varphi(\mathsf{T})$, or $\varphi(\mathsf{F})$ yields F and $\varphi(\mathsf{T})$ yields T. In the former case, $\varphi(R(t))$, $\varphi(\lceil t \rfloor \land R(t))$, and $\varphi(\neg\lceil t \rfloor \lor R(t))$ yield the same truth value. In the latter case, $\varphi(\lceil t \rfloor \land R(t))$ yields F and $\varphi(\neg\lceil t \rfloor \lor R(t))$ yields T when t is undefined, and both agree with $\varphi(R(t))$ when t is defined. The case that n is odd can be returned to the even case by considering $\varphi(\neg(\neg R(t)))$. This makes it possible to choose the outcome of undefined terms as appropriate, when translating the practical problem at hand to logical formulas.

In the case of our running example, this approach asks someone to choose *informally* between $|x|-1 \geq 0 \land 3\sqrt{|x|-1} \geq x+1$ and $\neg(|x|-1 \geq 0) \lor 3\sqrt{|x|-1} \geq x+1$, after which the student should solve the chosen one formally. We wish the choice be made by the student, on the basis that those values of x that make $\sqrt{|x|-1}$ undefined are not roots. Furthermore, the reasoning behind the choice should be formalizable.

Many people share the intuition that an undefined formula is neither false nor true, even if that results in the loss of the Law of Excluded Middle. For instance, [4] reports on a survey that was participated by over 200 software developers. When asked how various undefined situations should be interpreted, between 74 % and 91 % chose "error/exception", the other options being "true", "false", and "other (provide details)". One of the situations was a programming analogue of $\frac{1}{0} = 0 \lor \frac{1}{0} \neq 0$.

So we turn our attention to 3-valued logics. We denote the third truth value with U and call it "undefined". (Some authors talk of it as the absence of truth value instead of a truth value.) Everybody seems to agree that $\neg\mathsf{U}$ yields U. Now neither $3\sqrt{|x|-1} \geq x+1$ nor $\neg(3\sqrt{|x|-1} \geq x+1)$ has 0 as a root, because both of them yield U when $x = 0$. The definition of $t < t'$ as a shorthand for $\neg(t \geq t')$ works well.

In the presence of U, there are three major interpretations of $\wedge$ and $\vee$. Fortunately, we need not elaborate them now, because we will see in Section 8 that the other two can be obtained from our choice in Definition 3.2(5) and (6) as shorthands. Our choice is the same as Kleene's [14] and Łukasiewicz's [17]. Our $\forall$ and $\exists$ are analogous. The other versions of $\forall$ and $\exists$ that we encountered can be mimicked by them. What $\rightarrow$ should mean in this context is a tricky issue, as we will argue in Section 8. We will mimic Kleene's version in Section 3, and Łukasiewicz's version in Section 5.

The important issue regarding related work is how to say that a term or formula is undefined. An obvious idea is to introduce a new atomic predicate $*t$ or connective $*\varphi$ that yields F if its input is undefined and T otherwise [2,6,9,13,18,20,23]. Alternatively, the idea of $\lceil t \rfloor$ discussed above can be used and naturally extended to formulas [3,5], at the cost of needing a new component in addition to the traditional signature and set of axioms. In the words of [3]: "we will assume that included with every signature Σ is a set Δ of domain formulas, one for each function and predicate symbol in Σ." The difference is that $*$ is but $\lceil\,\rfloor$ is not a symbol in the core language. Instead, $\lceil t \rfloor$ and $\lceil \varphi \rfloor$ are metalanguage expressions that denote some formulas that typically only contain $\neg$, $\wedge$, $\vee$, $\forall$, $\exists$, and atomic formulas. We will adopt the latter approach in Section 5, and argue in Section 8 that it can mimic the former.

The replacement of $\varphi(R(t))$ by $\varphi(\lceil t \rfloor \wedge R(t))$ or $\varphi(\neg\lceil t \rfloor \vee R(t))$ remains a powerful practical reasoning method also in the presence of U, and then it takes place within the formal logic. More generally, assume that no other connectives and quantifiers are used than (our versions of) $\neg$, $\wedge$, $\vee$, $\forall$, and $\exists$; ψ is a formula; and $\varphi(\psi)$ is a formula where ψ occurs within the scope of an even number of negations. Then every interpretation that makes $\varphi(\psi)$ yield T, also makes $\varphi(\lceil \psi \rfloor \wedge \psi)$ yield T, and vice versa. A similar claim holds for odd number of negations and $\varphi(\neg\lceil \psi \rfloor \vee \psi)$. The hard part in proving these is ruling out the possibility that ψ yields U, $\varphi(\psi)$ yields T and $\varphi(\lceil \psi \rfloor \wedge \psi)$ or $\varphi(\neg\lceil \psi \rfloor \vee \psi)$ yields F or U. It becomes easy by appealing to the notion of *regularity* proposed by Kleene and developed in Section 4.

In our running example, $\lceil 3\sqrt{-x-1} \rfloor$ is $-x-1 \geq 0$, $\lceil 3\sqrt{x-1} \rfloor$ is $x-1 \geq 0$, and there are no negations. Therefore, $3\sqrt{-x-1} \geq x+1$ can be replaced by $x \leq -1 \wedge 3\sqrt{-x-1} \geq x+1$ and $3\sqrt{x-1} \geq x+1$ by $x \geq 1 \wedge 3\sqrt{x-1} \geq x+1$, resulting in a formula that yields U for no value of x. In this way formulas can be formally converted to a form where undefinedness plays essentially no role, after which classical binary logic can be used for the rest of the reasoning (for further discussion and sources, please see [5]). This approach can be used by both humans and computers. It is simple to use, because $\lceil t \rfloor$ and $\lceil \varphi \rfloor$ are obtained with an algorithm. The algorithm will be presented in Defnition 5.4.

If $\varphi(\psi)$ is replaced by $\varphi((*\psi) \wedge \psi)$ instead of $\varphi(\lceil \psi \rfloor \wedge \psi)$, then the result contains $*$, which is not any of $\neg$, $\wedge$, $\vee$, $\forall$, and $\exists$. As a consequence, doing a second replacement is not necessarily sound. So the practical approach described above is lost. On the other hand, if the original formula contains $*$, it can be replaced by $\lceil\rfloor$, opening the way to the practical approach described above.

In the present study we develop a logic that uses U and $\lceil\rfloor$, present a proof system for it, and prove that the system is sound and complete. The number of rules in our proof system that differ from classical rules is small. We believe that this is the first completeness proof for a proof system that relies on $\lceil\rfloor$. Furthermore, we believe to be the first to point out the role of regularity in practice-oriented reasoning in this context (one example was above and another will be in Section 4).

Also [13] claims completeness, but using $*$ and only for finite axiomatizations. The source [9] presents a (in our opinion hard to read) tableaux-based completeness proof for a logic that uses $*$. Its notion of $\models$ is unusual in that both T and U are designated values on the right (but only T on the left). As a consequence, its proof system would be only indirectly applicable to our purposes. The completeness claim in [5] does not refer to Gödel's sense, but to what in this study is Lemma 5.5(3) and (4): "The procedure is complete [8,9], that is, the well-definedness condition generated from a formula is provable if and only if the formula is well-defined."

In terms of free logics, ours has *neutral* semantics [20]. Positive or negative semantics only use the two truth values F and T, while neutral semantics and *supervaluation* also use U. A recent study [23] covers positive and negative semantics, but leaves out the latter two, mentioning that they "up to now still lack the rigorous systematicity the other two family members enjoy". Supervaluation makes $\frac{1}{0} = \frac{1}{0}$ yield T on the basis that if $\frac{1}{0}$ is given any value, no matter what, then $\frac{1}{0} = \frac{1}{0}$ would yield T in classical logic. In our logic $\frac{1}{0} = \frac{1}{0}$ yields U. Neutral non-supervaluation semantics were surveyed in [15]. Our logic disagrees with all the systems summarized in the table on p. 328. For instance, unlike our logic, $\wedge$ and $\vee$ are *strict* in [16], that is, if φ or ψ or both are undefined, then $\varphi \wedge \psi$ and $\varphi \vee \psi$ are undefined as well.

The studies [19,24,30] discuss proof systems for Kleene's logic, but only cover propositional logic, while [1] focuses on equational logic without quantifiers.

The syntax and semantics of our core logic are presented in Sections 2 and 3. We already mentioned that the notion of regularity is developed in Section 4, and $\lceil\rfloor$ in Section 5. Section 6 is devoted to a proof system for our logic, together with its soundness proof. The system is proven complete (in the sense of Gödel, allowing infinite sets of axioms) in Section 7. In Section 8 we argue that most, if not all, other 3-valued logics for similar applications can be mimicked by ours.

2 Formal Languages

Our notion of a *formal language* is essentially the same as in classical binary first-order logic. A couple of details are affected by the needs of the rest of this study. We will comment on them after presenting the definition.

The alphabet of a formal language is the union of the following five mutually disjoint sets:

1. The set $\mathscr{L}$ of the following eleven symbols: $(\;) \; , \; = \; \mathsf{F} \; \mathsf{T} \; \neg \; \wedge \; \vee \; \forall \; \exists$
2. A countably infinite set $\mathscr{V}$ of *variable symbols* $\mathtt{v}_1$, $\mathtt{v}_2$, ...
3. A countable set $\mathscr{C}$ of *constant symbols* c_1, c_2, ...
4. A countable set $\mathscr{F}$ of *function symbols* f_1, f_2, ...
5. A countable set $\mathscr{R}$ of *relation symbols* R_1, R_2, ...

All formal languages have the same $\mathscr{L}$ and the same $\mathscr{V}$, but not necessarily the same $\mathscr{C}$, $\mathscr{F}$, or $\mathscr{R}$. In particular, we will assume that the variable symbols are literally $\mathtt{v}_1$, $\mathtt{v}_2$, and so on. To emphasize this, we write them as $\mathtt{v}_i$ instead of v_i. When we want to refer to a variable symbol without saying which one, we use x, y, x_1, and so on, as metalanguage variable symbols.

Each function symbol f and each relation symbol R has an *arity* $\alpha(f)$ or $\alpha(R)$. It is a positive integer. A *signature* is the quadruple $(\mathscr{C}, \mathscr{F}, \mathscr{R}, \alpha)$.

Terms, atomic formulas, and formulas are defined recursively as follows.

Definition 2.1. Let a signature be fixed.

1. A *term* is either a variable symbol; a constant symbol; or of the form $f(t_1, \ldots, t_{\alpha(f)})$, where f is a function symbol and t_1, ..., $t_{\alpha(f)}$ are terms.
2. An *atomic formula* is either F; T; of the form $(t_1 = t_2)$ where t_1 and t_2 are terms; or of the form $R(t_1, \ldots, t_{\alpha(R)})$, where R is a relation symbol and t_1, ..., $t_{\alpha(R)}$ are terms.
3. A *formula* is either an atomic formula or of any of the following forms, where φ and ψ are formulas and x is a variable symbol:

$$(\neg\varphi) \mid (\varphi \wedge \psi) \mid (\varphi \vee \psi) \mid (\forall x \; \varphi) \mid (\exists x \; \varphi)$$

An occurrence of a variable symbol x in a formula is *bound* if and only if it is in a subformula of the form $(\forall x \; \varphi)$ or $(\exists x \; \varphi)$. The other occurrences of x are *free*. A formula is *closed* if and only if it has no free occurrences of variable symbols, and *open* in the opposite case.

We make a distinction between variables and variable symbols, for the reason illustrated by the classical binary first-order logic formula $\exists \mathtt{v}_2 \; (\mathtt{v}_2 < \mathtt{v}_1 \wedge \exists \mathtt{v}_1 \; (\mathtt{v}_1 < \mathtt{v}_2))$ on real numbers. If, for instance, the free occurrence of $\mathtt{v}_1$ has the value 3, then the formula can be shown to hold by letting $\mathtt{v}_2 = 2$ and the bound occurrences of $\mathtt{v}_1$ have the value 1. Instead of thinking of the variable symbol $\mathtt{v}_1$ having simultaneously the values 3 and 1, we think of the free and bound occurrences of $\mathtt{v}_1$ as referring to two distinct variables which just happen to have the same name. In general, there are two kinds of *variables*: free and bound. Every variable symbol that occurs free introduces a *free variable*, and every $\forall$ and every $\exists$ introduces a *bound variable*.

We will use $\varphi(x)$ as a synonym for φ, and $\varphi(t)$ to denote the result of replacing every free occurrence of x in φ by t. The purpose of the notation $\varphi(x)$ is to make it clear which is the variable symbol whose free occurrences are replaced. By *t is free for x in φ* it is meant that no variable symbol in t becomes bound in $\varphi(t)$. As is well known, replacing x by t that is not free for x is often incorrect, because, intuitively speaking, a free variable in t disappears and another, bound variable with the same name takes its place.

The atomic formulas F and T, corresponding to the truth values false and true, have been included in the language for technical convenience. Although we will also talk about a third truth value U (undefined), we did not include a corresponding atomic formula in the language. Thanks to this design choice, our theory reduces to classical binary first-order logic when every function symbol is defined everywhere.

We will tell how to add Kleene's versions of the symbols $\rightarrow$ and $\leftrightarrow$ to the language in Section 3, and Łukasiewicz's version of $\rightarrow$ in Section 5.

We adopt some semiformal conventions to improve the readability of formulas. To reduce the need of (and), we let $\neg$ have the highest precedence, then $\wedge$, then $\vee$, and finally the quantifiers $\forall$ and $\exists$. The connectives $\wedge$ and $\vee$ associate to the left; that is, $\varphi \wedge \psi \wedge \chi$ denotes $((\varphi \wedge \psi) \wedge \chi)$, and similarly for $\vee$. In examples we may use the familiar syntax of the domain of discourse of the example. For instance, in the case of natural numbers we may write

$$\forall n\ (\neg \exists m\ (m \cdot m = n)) \vee \sqrt{n} \cdot \sqrt{n} = n$$

as a human-friendly semiformal representation of the formula

$$(\forall \mathtt{v}_1\ ((\neg(\exists \mathtt{v}_2\ (\cdot(\mathtt{v}_2, \mathtt{v}_2) = \mathtt{v}_1))) \vee (\cdot(\surd(\mathtt{v}_1), \surd(\mathtt{v}_1)) = \mathtt{v}_1)))$$

Let φ, ψ, and χ denote any formulas. In the metalanguage, we use $\varphi \cong \psi$ to denote that after unwinding all semiformal abbreviations, φ and ψ result in literally the same formula. For instance, because we have chosen that both $\varphi \vee \psi \vee \chi$ and $(\varphi \vee \psi) \vee \chi$ are abbreviations for $((\varphi \vee \psi) \vee \chi)$, we have $\varphi \vee \psi \vee \chi \cong (\varphi \vee \psi) \vee \chi$. On the other hand, we have $\varphi \vee \psi \vee \chi \ncong \varphi \vee (\psi \vee \chi)$, because $(\varphi \vee (\psi \vee \chi))$ is not literally the same formula as $((\varphi \vee \psi) \vee \chi)$.

3 Structures, Truth Values, and Models

Our notion of a structure differs from the standard one in that function symbols may denote partial functions. More formally, a *structure* $(\mathbb{D}, \_)$ on a signature $(\mathscr{C}, \mathscr{F}, \mathscr{R}, \alpha)$ consists of the following:

1. A non-empty set $\mathbb{D}$, called the *domain of discourse*.

2. For each $c \in \mathscr{C}$, an element $\underline{C}$ of $\mathbb{D}$. (We reserve the symbol $\underline{c}$ for another use.)

3. For each $f \in \mathscr{F}$, a partial function $\underline{f}$ from $\mathbb{D}^{\alpha(f)}$ to $\mathbb{D}$.

4. For each $R \in \mathscr{R}$, a subset $\underline{R}$ of $\mathbb{D}^{\alpha(R)}$.

The standard approach would continue by defining an assignment of values for variable symbols, and then defining a value for each term and a truth value for each formula. We will proceed in the opposite order, by first interpreting terms as partial functions and formulas as total functions, and then assigning values to variable symbols. We do so to simplify the formulation of the notion of "regularity" that will be presented in Definition 4.2.

In itself, reversing the order is insignificant, since it does not affect the fundamental ideas but only their formalization. However, because of the overall goal of our study, we also introduce two significant changes. First, terms need not yield a value. Second, the values of formulas are picked from among three truth values *false*, *undefined* and *true*, denoted by F, U and T, respectively.

To interpret terms and formulas as functions, we need to map variable symbols to argument positions. For instance, we need to decide whether $\mathtt{v}_3 + \mathtt{v}_2$ is interpreted as the function $\mathbb{D}^2 \to \mathbb{D}; (d_3, d_2) \mapsto d_3 + d_2$ where d_i denotes the value of $\mathtt{v}_i$, or as something else. We interpret it as $\mathbb{D}^3 \to \mathbb{D}; (d_1, d_2, d_3) \mapsto d_3 + d_2$. In general, we let the arguments correspond to $\mathtt{v}_1$, $\mathtt{v}_2$, and so on, in this order, as far as needed by the term or formula. This will make many argument lists contain positions whose corresponding variable symbol does not occur (free) in the term or formula, such as the first position and $\mathtt{v}_1$ in $\mathbb{D}^3 \to \mathbb{D}; (d_1, d_2, d_3) \mapsto d_3 + d_2$. This will not be much of a problem, because interpretation as functions is only an auxiliary tool. After assigning values to free variables, a standard kind of interpretation is obtained.

For each term t we define its arity $\alpha(t)$ as the biggest i such that $\mathtt{v}_i$ occurs in t. If no variable symbol occurs in t, then $\alpha(t) = 0$. It is easy to see that the arity of a compound term $f(t_1, \ldots, t_{\alpha(f)})$ is the maximum of the arities of its constituents t_1, …, $t_{\alpha(f)}$. Similarly, we define that the arity of a closed formula is 0, and the arity of an open formula is the biggest i such that $\mathtt{v}_i$ occurs free in it. It is possible that $\alpha((\forall x\ \varphi)) < \alpha(\varphi)$ and $\alpha((\exists x\ \varphi)) < \alpha(\varphi)$. Even so, the arity of a compound formula is at most the maximum of the arities of its constituents.

Definition 3.1. Given a signature $(\mathscr{C}, \mathscr{F}, \mathscr{R}, \alpha)$ and a structure $(\mathbb{D}, \_)$ on it, each term t defines a partial function $\underline{t}$ from $\mathbb{D}^{\alpha(t)}$ to $\mathbb{D}$ as follows. In the definition, d_1, …, d_n are arbitrary elements of $\mathbb{D}$.

1. If $\mathtt{v}_n \in \mathscr{V}$, then $\underline{\mathtt{v}_n}$ is the total function from $\mathbb{D}^n$ to $\mathbb{D}$ that maps $(d_1, \ldots, d_n)$ to d_n. That is, $\underline{\mathtt{v}_n}(d_1, \ldots, d_n) = d_n$.

2. If $c \in \mathscr{C}$, then $\underline{c}$ is the function with arity zero such that $\underline{c}() = \underline{C}$. That is, $\underline{c} : \mathbb{D}^0 \to \mathbb{D}; () \mapsto \underline{C}$.

3. If $f \in \mathscr{F}$ and $t_1, \dots, t_{\alpha(f)}$ are terms, then let $n = \max\{\alpha(t_1), \dots, \alpha(t_{\alpha(f)})\}$. We define $\underline{f(t_1, \dots, t_{\alpha(f)})}$ as the following partial function from $\mathbb{D}^n$ to $\mathbb{D}$.
   - If for $1 \le i \le \alpha(f)$, any of the $\underline{t_i}(d_1, \dots, d_{\alpha(t_i)})$ is undefined, then $\underline{f(t_1, \dots, t_{\alpha(f)})}(d_1, \dots, d_n)$ is undefined as well.
   - Otherwise, for $1 \le i \le \alpha(f)$ let $e_i = \underline{t_i}(d_1, \dots, d_{\alpha(t_i)})$. If $\underline{f}(e_1, \dots, e_{\alpha(f)})$ is defined, then $\underline{f(t_1, \dots, t_{\alpha(f)})}(d_1, \dots, d_n) = \underline{f}(e_1, \dots, e_{\alpha(f)})$; and otherwise $\underline{f(t_1, \dots, t_{\alpha(f)})}(d_1, \dots, d_n)$ is undefined.

The definition obeys the principle that if any subterm of a term is undefined, then the term as a whole is undefined as well. That is, our partial functions are *strict*.

Definition 3.2. Given a signature and a structure $(\mathbb{D}, \underline{\ })$ on it, each formula φ defines a total function $\underline{\varphi}$ from $\mathbb{D}^{\alpha(\varphi)}$ to $\{\mathsf{F}, \mathsf{U}, \mathsf{T}\}$ as follows. In the definition, d_1, $\dots$, d_n are arbitrary elements of $\mathbb{D}$. To avoid confusion with the formal symbol $=$, we write $\underline{\varphi}(d_1, \dots, d_{\alpha(\varphi)}) \asymp \mathsf{F}$ to denote that $\underline{\varphi}$ maps $(d_1, \dots, d_{\alpha(\varphi)})$ to F, and similarly with U and T.

1. We define that $\underline{\mathsf{F}}$ and $\underline{\mathsf{T}}$ are the functions with arity zero whose values are F and T, respectively.

2. Let $n = \max\{\alpha(t_1), \alpha(t_2)\}$. We define $\underline{(t_1 = t_2)}(d_1, \dots, d_n) \asymp$

   U, if $\underline{t_1}(d_1, \dots, d_{\alpha(t_1)})$ or $\underline{t_2}(d_1, \dots, d_{\alpha(t_2)})$ is undefined

   T, if $\underline{t_1}(d_1, \dots, d_{\alpha(t_1)})$ and $\underline{t_2}(d_1, \dots, d_{\alpha(t_2)})$ are defined, and
   $$\underline{t_1}(d_1, \dots, d_{\alpha(t_1)}) = \underline{t_2}(d_1, \dots, d_{\alpha(t_2)})$$

   F, if $\underline{t_1}(d_1, \dots, d_{\alpha(t_1)})$ and $\underline{t_2}(d_1, \dots, d_{\alpha(t_2)})$ are defined, and
   $$\underline{t_1}(d_1, \dots, d_{\alpha(t_1)}) \ne \underline{t_2}(d_1, \dots, d_{\alpha(t_2)})$$

3. Let $n = \max\{\alpha(t_1), \dots, \alpha(t_{\alpha(R)})\}$, and let $e_i = \underline{t_i}(d_1, \dots, d_{\alpha(t_i)})$ when the latter is defined. We define $\underline{R(t_1, \dots, t_{\alpha(R)})}(d_1, \dots, d_n) \asymp$

   U, if for $1 \le i \le \alpha(R)$, any of the $\underline{t_i}(d_1, \dots, d_{\alpha(t_i)})$ is undefined

   T, if for $1 \le i \le \alpha(R)$, each $\underline{t_i}(d_1, \dots, d_{\alpha(t_i)})$ is defined and $(e_1, \dots, e_{\alpha(R)}) \in \underline{R}$

   F, if for $1 \le i \le \alpha(R)$, each $\underline{t_i}(d_1, \dots, d_{\alpha(t_i)})$ is defined and $(e_1, \dots, e_{\alpha(R)}) \notin \underline{R}$

4. Clearly $\alpha((\neg\varphi)) = \alpha(\varphi)$. We define $\underline{(\neg\varphi)}(d_1, \dots, d_{\alpha((\neg\varphi))}) \asymp$

   F, if $\underline{\varphi}(d_1, \dots, d_{\alpha(\varphi)}) \asymp \mathsf{T}$

   T, if $\underline{\varphi}(d_1, \dots, d_{\alpha(\varphi)}) \asymp \mathsf{F}$

U, if $\underline{\varphi}(d_1,\ldots,d_{\alpha(\varphi)}) \asymp \mathsf{U}$

5. Let $n = \max\{\alpha(\varphi), \alpha(\psi)\}$. We define $\underline{(\varphi \wedge \psi)}(d_1,\ldots,d_n) \asymp$

   T, if $\underline{\varphi}(d_1,\ldots,d_{\alpha(\varphi)}) \asymp \underline{\psi}(d_1,\ldots,d_{\alpha(\psi)}) \asymp \mathsf{T}$
   F, if $\underline{\varphi}(d_1,\ldots,d_{\alpha(\varphi)}) \asymp \mathsf{F}$ or $\underline{\psi}(d_1,\ldots,d_{\alpha(\psi)}) \asymp \mathsf{F}$
   U, otherwise

6. Let $n = \max\{\alpha(\varphi), \alpha(\psi)\}$. We define $\underline{(\varphi \vee \psi)}(d_1,\ldots,d_n) \asymp$

   F, if $\underline{\varphi}(d_1,\ldots,d_{\alpha(\varphi)}) \asymp \underline{\psi}(d_1,\ldots,d_{\alpha(\psi)}) \asymp \mathsf{F}$
   T, if $\underline{\varphi}(d_1,\ldots,d_{\alpha(\varphi)}) \asymp \mathsf{T}$ or $\underline{\psi}(d_1,\ldots,d_{\alpha(\psi)}) \asymp \mathsf{T}$
   U, otherwise

7. If $i > \alpha(\varphi)$, we define $\underline{(\forall \mathrm{v}_i\ \varphi)}$ as $\underline{\varphi}$. Otherwise $1 \leq i \leq \alpha(\varphi)$, and we define $\underline{(\forall \mathrm{v}_i\ \varphi)}(d_1,\ldots,d_{\alpha((\forall \mathrm{v}_i\ \varphi))}) \asymp$

   T, if for every $e_i \in \mathbb{D}$ we have $\underline{\varphi}(d_1,\ldots,e_i,\ldots,d_{\alpha(\varphi)}) \asymp \mathsf{T}$
   F, if for at least one $e_i \in \mathbb{D}$ we have $\underline{\varphi}(d_1,\ldots,e_i,\ldots,d_{\alpha(\varphi)}) \asymp \mathsf{F}$
   U, otherwise

8. If $i > \alpha(\varphi)$, we define $\underline{(\exists \mathrm{v}_i\ \varphi)}$ as $\underline{\varphi}$. Otherwise $1 \leq i \leq \alpha(\varphi)$, and we define $\underline{(\exists \mathrm{v}_i\ \varphi)}(d_1,\ldots,d_{\alpha((\exists \mathrm{v}_i\ \varphi))}) \asymp$

   F, if for every $e_i \in \mathbb{D}$ we have $\underline{\varphi}(d_1,\ldots,e_i,\ldots,d_{\alpha(\varphi)}) \asymp \mathsf{F}$
   T, if for at least one $e_i \in \mathbb{D}$ we have $\underline{\varphi}(d_1,\ldots,e_i,\ldots,d_{\alpha(\varphi)}) \asymp \mathsf{T}$
   U, otherwise

It follows from (2) that if both t_1 and t_2 are defined, then $t_1 = t_2$ compares their values in the usual fashion, and if at least one of them is undefined, then $(t_1 = t_2) \asymp \mathsf{U}$. For instance, with real numbers, $\sqrt{-1} = \sqrt{-1}$ is not true but undefined.

More generally, by (1), (2), and (3), an atomic formula yields U only if it contains an undefined term. This restriction is only for technical convenience. It may be circumvented by introducing a new function symbol f that is undefined precisely when desired, making $(d_1,\ldots,d_{\alpha(R)}) \in \underline{R}$ when $\underline{f}(d_1,\ldots,d_{\alpha(f)})$ is undefined, and using $R(x_1,\ldots,x_{\alpha(R)}) \wedge (f(x_1,\ldots,x_{\alpha(f)}) = f(x_1,\ldots,x_{\alpha(f)}))$.

It is easy to check that (4), (5), and (6) make $\neg$, $\wedge$, and $\vee$ match the corresponding truth tables in Figure 1. Furthermore, Kleene's conditional and biconditional can be obtained by treating $\varphi \rightarrow \psi$ as a shorthand for $\neg\varphi \vee \psi$, and $\varphi \leftrightarrow \psi$ as a shorthand for $(\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$. We will show in Section 4 that Łukasiewicz's conditional $\twoheadrightarrow$ cannot be expressed in our language. However, Section 5 will reveal that any formula that contains it can be replaced by a formula in our language.

In (7) and (8), if $i > \alpha(\varphi)$, it is appropriate to define quantification so that it has no effect, because then v_i does not occur free in φ. The definitions for the case

$\neg$	
F	T
U	U
T	F

$\wedge$	F	U	T
F	F	F	F
U	F	U	U
T	F	U	T

$\vee$	F	U	T
F	F	U	T
U	U	U	T
T	T	T	T

$\rightarrow$	F	U	T
F	T	T	T
U	U	U	T
T	F	U	T

$\leftrightarrow$	F	U	T
F	T	U	F
U	U	U	U
T	F	U	T

$\twoheadrightarrow$	F	U	T
F	T	T	T
U	U	T	T
T	F	U	T

Figure 1: Truth tables of some propositional connectives. The symbols $\rightarrow$ and $\leftrightarrow$ are Kleene's conditional and biconditional, and $\twoheadrightarrow$ is Łukasiewicz's conditional

$i \leq \alpha(\varphi)$ are analogous to the definitions of $\wedge$ and $\vee$. In them, v_i may but need not occur free in φ.

If every function symbol in a formula φ is defined everywhere, then everywhere $\underline{\varphi}(d_1,\ldots,d_{\alpha(\varphi)}) \not\simeq \mathsf{U}$. If all function symbols of the language are defined everywhere, then Definitions 3.1 and 3.2 reduce to the classical binary first-order logic semantics represented in a function form.

It is helpful to think of F being smaller than U which is smaller than T. Then $\underline{(\varphi \wedge \psi)}(d_1,\ldots,d_{\alpha((\varphi\wedge\psi))})$ yields the minimum of the results of $\underline{\varphi}(d_1,\ldots,d_{\alpha(\varphi)})$ and $\underline{\psi}(d_1,\ldots,d_{\alpha(\psi)})$, and $\underline{(\varphi \vee \psi)}(d_1,\ldots,d_{\alpha((\varphi\vee\psi))})$ yields the maximum. Furthermore, $\underline{(\forall \mathrm{v}_i\ \varphi(\mathrm{v}_i))}(d_1,\ldots,d_{\alpha((\forall \mathrm{v}_i\ \varphi(\mathrm{v}_i)))})$ yields the minimum of $\underline{\varphi}'(e)$ for $e \in \mathbb{D}$, and $\underline{(\exists \mathrm{v}_i\ \varphi(\mathrm{v}_i))}(d_1,\ldots,d_{\alpha((\exists \mathrm{v}_i\ \varphi(\mathrm{v}_i)))})$ yields the maximum. Here $\underline{\varphi}'$ denotes the function from $\mathbb{D}$ to $\{\mathsf{F},\mathsf{U},\mathsf{T}\}$ obtained by using $d_1, \ldots, d_{\alpha(\varphi)}$ as other arguments of $\underline{\varphi}$ than the ith. That is, if $i > \alpha(\varphi)$, then $e \mapsto \underline{\varphi}(d_1,\ldots,d_{\alpha(\varphi)})$, and otherwise $e \mapsto \underline{\varphi}(d_1,\ldots,e,\ldots,d_{\alpha(\varphi)})$.

It is easy to check that De Morgan's laws hold in our logic:

Lemma 3.3.

1. *$\underline{(\neg(\varphi \wedge \psi))}$ is the same function as $\underline{((\neg\varphi) \vee (\neg\psi))}$.*
2. *$\underline{(\neg(\forall x\ \varphi))}$ is the same function as $\underline{(\exists x\ (\neg\varphi))}$.*

If a term is not free for a variable symbol in a formula, then the following lemma can be used to change the names of the bound variables in the formula, so that the term becomes free.

Lemma 3.4. *If y does not occur in $\varphi(x)$, then $\underline{(\forall y\ \varphi(y))}$ is the same function as $\underline{(\forall x\ \varphi(x))}$, and $\underline{(\exists y\ \varphi(y))}$ is the same function as $\underline{(\exists x\ \varphi(x))}$.*

Proof. If x and y are the same variable symbol, or if x does not occur free in $\varphi(x)$, then the claim is trivial. So we assume that they are distinct and x does occur free.

By construction, x does not occur free in $\varphi(y)$. By assumption, y does not occur free in $\varphi(x)$. Therefore, $\forall x\ \varphi(x)$ and $\forall y\ \varphi(y)$ have the same free variables. Let $n = \alpha((\forall x\ \varphi(x))) = \alpha((\forall y\ \varphi(y)))$.

Because y does not occur in $\varphi(x)$, all occurrences of y in $\varphi(y)$ are free, and they match precisely the free occurrences of x in $\varphi(x)$. Let i and j be such that x is v_i and y is v_j. The functions $\underline{\varphi(x)}$ and $\underline{\varphi(y)}$ have $\max\{n,i\}$ and $\max\{n,j\}$ arguments, respectively, but their values only depend on those arguments whose corresponding variable occurs free. The values of x and y go in via different argument positions, but are from then on treated identically. The values of all other free variables are treated fully identically.

Therefore, $\underline{\varphi(x)}(d_1,\ldots,d_n;e_i) \asymp \underline{\varphi(y)}(d_1,\ldots,d_n;e_j)$, where the notation has the following meaning. The symbols $d_1, \ldots, d_{\max\{n,i-1,j-1\}}$, and e_i denote arbitrary elements of $\mathbb{D}$, and $e_j = e_i$. If $i \leq n$, then $(d_1,\ldots,d_n;e_i)$ denotes $(d_1,\ldots,e_i,\ldots,d_n)$, and if $i > n$, it denotes $(d_1,\ldots,d_{i-1},e_i)$. Furthermore, $(d_1,\ldots,d_n;e_j)$ is defined similarly.

As a consequence, $\underline{(\forall x\ \varphi(x))}(d_1,\ldots,d_n) \asymp \underline{(\forall y\ \varphi(y))}(d_1,\ldots,d_n)$, and similarly with $\exists$. $\square$

Definition 3.5. Let a signature be fixed.

Given a structure $\sigma = (\mathbb{D}, \_)$ on it, an *assignment of values to free variables* is a total function ν from $\mathbb{Z}^+$ to $\mathbb{D}$. Given σ and ν, any term t yields $\underline{t}(\nu(1),\ldots,\nu(\alpha(t)))$, and any formula φ yields $\underline{\varphi}(\nu(1),\ldots,\nu(\alpha(\varphi)))$.

A *model* of a formula φ is a pair (σ,ν) such that $\underline{\varphi}(\nu(1),\ldots,\nu(\alpha(\varphi))) \asymp \mathsf{T}$. This is denoted with $(\sigma,\nu) \models \varphi$. If Γ is a set of formulas, then $(\sigma,\nu) \models \Gamma$ means that for every $\varphi \in \Gamma$ we have $(\sigma,\nu) \models \varphi$.

If x denotes the variable v_i, then by $\nu(x)$ we mean $\nu(i)$. Let $d \in \mathbb{D}$. By $\nu[x := d]$ we denote the assignment such that $\nu[x := d](x) = d$ and $\nu[x := d](y) = \nu(y)$ when y is not the same variable as x. For brevity, we will often write $\underline{t}(\nu)$ instead of $\underline{t}(\nu(1),\ldots,\nu(\alpha(t)))$ and $\underline{\varphi}(\nu)$ instead of $\underline{\varphi}(\nu(1),\ldots,\nu(\alpha(\varphi)))$. The following lemma is immediate from Definition 3.1.

Lemma 3.6. *Assume that $\underline{t}(\nu)$ is defined and yields the value d. If t is free for x in $\varphi(x)$, then $\underline{\varphi(t)}(\nu) \asymp \underline{\varphi(x)}(\nu[x := d])$.*

4 Regularity

In this section we introduce and discuss a notion that needs different technical background from the rest of this study. We first briefly introduce the necessary background.

By a *3-valued propositional logic* we mean a logic whose alphabet consists of F, U, T, proposition symbols, and a choice of propositional connectives. Any formula in the logic whose proposition symbols are among $P_1, \ldots, P_n$ can be interpreted as a truth function from $\{\mathsf{F},\mathsf{U},\mathsf{T}\}^n$ to $\{\mathsf{F},\mathsf{U},\mathsf{T}\}$. Kleene's 3-valued propositional logic [14,24] has the connectives $\neg$, $\wedge$, $\vee$, $\rightarrow$, and $\leftrightarrow$ introduced in Figure 1. Łukasiewicz's 3-valued propositional logic [17] has $\neg$, $\wedge$, $\vee$, $\twoheadrightarrow$, and a biconditional version of $\twoheadrightarrow$.

One can check from Figure 1 that the truth functions represented by $P \vee Q$, $P \rightarrow Q$, and $P \leftrightarrow Q$ can also be represented as $\neg(\neg P \wedge \neg Q)$, $\neg P \vee Q$, and $(P \rightarrow Q) \wedge (Q \rightarrow P)$, respectively. On the other hand, we will soon see that $P \twoheadrightarrow Q$ cannot be constructed from other connectives in the figure.

Kleene's 3-valued propositional logic has a useful property called *regularity*. Intuitively, it says that if the truth value of a formula depends on the truth value of P_i (while the truth values of the other proposition symbols remain unchanged), then the truth value of the formula is U when the truth value of P_i is U.

Definition 4.1. Let $\pi(P_1,\ldots,P_n)$ be a truth function. It is *regular* if and only if for each $1 \leq i \leq n$, for each j such that $1 \leq j \leq n$ and $j \neq i$, and for each $P_j \in \{\mathsf{F},\mathsf{U},\mathsf{T}\}$

1. either $\pi(P_1,\ldots,\mathsf{U},\ldots,P_n) \asymp \mathsf{U}$
2. or $\pi(P_1,\ldots,\mathsf{F},\ldots,P_n) \asymp \pi(P_1,\ldots,\mathsf{U},\ldots,P_n) \asymp \pi(P_1,\ldots,\mathsf{T},\ldots,P_n)$,

where the explicitly shown truth value F, U, or T, is assigned to P_i.

A formula is regular if and only if the truth function represented by it is regular. A propositional logic is regular if and only if all of its formulas are regular.

It is easy to see from Figure 1 that $P \twoheadrightarrow Q$ is not regular: $\mathsf{U} \twoheadrightarrow \mathsf{U} \asymp \mathsf{T} \not\asymp \mathsf{U}$, but $\mathsf{T} \twoheadrightarrow \mathsf{U} \asymp \mathsf{U} \not\asymp \mathsf{U} \twoheadrightarrow \mathsf{U}$. Therefore, Łukasiewicz's 3-valued propositional logic is not regular.

In Figure 1, excluding $\twoheadrightarrow$, each row and each column either has U in the middle, or its every entry is F or every entry is T. Therefore, $\neg P$, $P \wedge Q$, $P \vee Q$, $P \rightarrow Q$, and $P \leftrightarrow Q$ are regular. It is possible to prove (and we will de facto do so as part of the proof of Theorem 4.3) that every propositional formula that is composed only using proposition symbols, F, T, U, $\neg$, $\wedge$, $\vee$, $\rightarrow$, and $\leftrightarrow$ is regular. As a consequence, Kleene's propositional logic is regular. Therefore, $P \twoheadrightarrow Q$ cannot be constructed in it. It is also impossible to construct a formula $*(P)$ such that $*(\mathsf{U}) \asymp \mathsf{F}$ and $*(\mathsf{T}) \asymp *(\mathsf{F}) \asymp \mathsf{T}$, because it is irregular. (On the other hand, $*(P) \asymp \neg((P \twoheadrightarrow \neg P) \wedge (\neg P \twoheadrightarrow P))$, and $P \twoheadrightarrow Q \asymp \neg P \vee Q \vee \neg(*(P) \vee *(Q))$.)

Next we adapt the notion of regularity to our predicate logic. Although the value of a variable is never undefined, it is possible to assign an undefined term in the place of each free occurrence of the variable symbol. We will need handy notation for discussing such situations. Therefore, we introduce a new metalanguage symbol $\perp$, to be used only in this section, to represent the missing value of an undefined term.

We declare $\perp \notin \mathbb{D}$ and define $\mathbb{D}_\perp = \mathbb{D} \cup \{\perp\}$. Then we extend each $\underline{t}$ to a partial function $\underline{t}_\perp$ from $\mathbb{D}_\perp^{\alpha(t)}$ to $\mathbb{D}$, and each $\underline{\varphi}$ to a total function $\underline{\varphi}_\perp$ from $\mathbb{D}_\perp^{\alpha(\varphi)}$ to $\{\mathsf{F},\mathsf{U},\mathsf{T}\}$. (We do not follow the well-known approach of extending $\underline{t}$ to a total function from $\mathbb{D}_\perp^{\alpha(t)}$ to $\mathbb{D}_\perp$, because we want to use $\perp$ as little as possible.) The desired effect is obtained by rewriting Definition 3.1 and 3.2 such that $\underline{}_\perp$ is used instead of $\underline{}$, and 3.1(1) is replaced by the following:

> If $\mathrm{v}_n \in \mathscr{V}$, then $\underline{\mathrm{v}_n}_\perp$ is the partial function from $\mathbb{D}_\perp^n$ to $\mathbb{D}$ such that if $e_n \in \mathbb{D}$, then $\underline{\mathrm{v}_n}_\perp(e_1,\ldots,e_n) = e_n$, and otherwise $\underline{\mathrm{v}_n}_\perp(e_1,\ldots,e_n)$ is undefined.

By an "extended value" of a free variable v_i we mean an element of $\mathbb{D}_\perp$ as the ith argument of $\underline{t}_\perp$ or $\underline{\varphi}_\perp$. Intuitively, regularity says that for any free variable

v_i, if the truth value of a formula depends on the extended value of v_i (while the extended values of the other free variables remain unchanged), then the truth value of the formula is U when the extended value of v_i is undefined.

Definition 4.2. Let φ be a formula and $n = \alpha(\varphi)$. The formula φ is *regular* if and only if for each $1 \leq i \leq n$, for each j such that $1 \leq j \leq n$ and $j \neq i$, and for each $e_1 \in \mathbb{D}_\perp, \ldots, e_{i-1} \in \mathbb{D}_\perp, e_{i+1} \in \mathbb{D}_\perp, \ldots, e_n \in \mathbb{D}_\perp$

1. either $\underline{\varphi}_\perp(e_1,\ldots,\perp,\ldots,e_n) \asymp \mathsf{U}$
2. or $\underline{\varphi}_\perp(e_1,\ldots,e_i,\ldots,e_n) \asymp \underline{\varphi}_\perp(e_1,\ldots,\perp,\ldots,e_n)$ for every $e_i \in \mathbb{D}$,

where the value $\perp$ or e_i is used as the ith argument.

The following theorem is from [28], but we have improved its proof.

Theorem 4.3. *The logic in Definitions 2.1, 3.1, and 3.2 is regular.*

Proof. Let φ, n, i, and e_1, $\ldots$, e_n be like in Definition 4.2. For brevity, if ψ is any subformula of φ, we write $\underline{\psi}_\perp(e_i)$ instead of $\underline{\psi}_\perp(e_1,\ldots,e_{\alpha(\psi)})$ both when $1 \leq i \leq \alpha(\psi)$ and when $i > \alpha(\psi)$. We use induction on the structure of φ to show that $\underline{\varphi}_\perp(\perp) \asymp \mathsf{U}$ or $\underline{\varphi}_\perp(e_i)$ is the same for every $e_i \in \mathbb{D}_\perp$.

The base case consists of atomic formulas. By Definition 3.2(1), $\underline{\mathsf{F}}_\perp(e_i) \asymp \mathsf{F}$ and $\underline{\mathsf{T}}_\perp(e_i) \asymp \mathsf{T}$ independently of e_i. If v_i occurs in neither t_1 nor t_2, then $\underline{(t_1 = t_2)}_\perp(e_i)$ does not depend on e_i. Otherwise, if e_i is $\perp$, then t_1 or t_2 is undefined, so by 3.2(2) $\underline{(t_1 = t_2)}_\perp(\perp) \asymp \mathsf{U}$. By 3.2(3), similar reasoning applies to $R(t_1,\ldots,t_{\alpha(R)})$. So the atomic formulas are regular.

The induction step consists of five cases. By the induction assumption, the subformula(s) ψ, ψ_1, and ψ_2 of each case are regular.

Let φ be $\neg\psi$. By 3.2(4), if $\underline{\psi}_\perp(\perp) \asymp \mathsf{U}$, then also $\underline{\varphi}_\perp(\perp) \asymp \mathsf{U}$. If $\underline{\psi}_\perp(e_i) \asymp \mathsf{F}$ independently of e_i, then $\underline{\varphi}_\perp(e_i) \asymp \mathsf{T}$ independently of e_i. If $\underline{\psi}_\perp(e_i) \asymp \mathsf{T}$ independently of e_i, then $\underline{\varphi}_\perp(e_i) \asymp \mathsf{F}$ independently of e_i.

Let φ be $\psi_1 \wedge \psi_2$. By 3.2(5), if $\underline{\psi_1}_\perp(e_i) \asymp \underline{\psi_2}_\perp(e_i) \asymp \mathsf{T}$ independently of e_i, then also $\underline{\varphi}_\perp(e_i) \asymp \mathsf{T}$ independently of e_i. If $\underline{\psi_1}_\perp(e_i) \asymp \mathsf{F}$ or $\underline{\psi_2}_\perp(e_i) \asymp \mathsf{F}$ independently of e_i, then also $\underline{\varphi}_\perp(e_i) \asymp \mathsf{F}$ independently of e_i. In the remaining cases $\underline{\psi_1}_\perp(\perp) \not\asymp \mathsf{F} \not\asymp \underline{\psi_2}_\perp(\perp)$, and $\underline{\psi_1}_\perp(\perp) \asymp \mathsf{U}$ or $\underline{\psi_2}_\perp(\perp) \asymp \mathsf{U}$. Then $\underline{\varphi}_\perp(\perp) \asymp \mathsf{U}$.

Let φ be $\forall x\ \psi$. We write $\underline{\psi}_\perp(e_i;d)$ to indicate that the value of x is $d \in \mathbb{D}$. By 3.2(7), if for every $d \in \mathbb{D}$ we have $\underline{\psi}_\perp(e_i;d) \asymp \mathsf{T}$ independently of e_i, then also $\underline{\varphi}_\perp(e_i) \asymp \mathsf{T}$ independently of e_i. If for some $d \in \mathbb{D}$ we have $\underline{\psi}_\perp(e_i;d) \asymp \mathsf{F}$ independently of e_i, then also $\underline{\varphi}_\perp(e_i) \asymp \mathsf{F}$ independently of e_i. In the remaining cases at least one $d \in \mathbb{D}$ yields $\underline{\psi}_\perp(\perp;d) \asymp \mathsf{U}$, and no $d \in \mathbb{D}$ yields $\underline{\psi}_\perp(\perp;d) \asymp \mathsf{F}$. Then $\underline{\varphi}_\perp(\perp) \asymp \mathsf{U}$.

The cases $\psi_1 \vee \psi_2$ and $\exists x\ \psi$ are proven similarly using 3.2(6) and 3.2(8). $\square$

We gave an example in Section 1 that regularity is important in practical application of our logic. Just to give another example that can be explained briefly: let t and t' be terms such that they are free for x in $\varphi(x)$, and when t is defined, then $t = t'$. For instance, we may have $t = \frac{x-2}{x-2}$ and $t' = 1$. If the logic is regular, then $\varphi(t)$ implies $\varphi(t')$. This is because when $\underline{\varphi(t)} \asymp \mathsf{T}$ but t is undefined, then $\underline{\varphi(t')} \asymp \mathsf{T}$ by regularity. This makes it correct to solve $\frac{x-2}{x-2}(x^2-5x+7) = 1$ by replacing $\frac{x-2}{x-2}$ by 1, solving $1(x^2-5x+7) = 1$, and checking its roots 2 and 3 against the original equation. The root 2 fails and 3 passes the check, so 3 is the only root of $\frac{x-2}{x-2}(x^2-5x+7) = 1$.

From now on we will not use $\perp$ explicitly. Instead, when we appeal to regularity in the sequel, we will use the following corollary of Theorem 4.3.

Corollary 4.4. *If $\underline{t}(\nu)$ is undefined but $\underline{\varphi(t)}(\nu) \asymp \mathsf{T}$ or $\underline{\varphi(t)}(\nu) \asymp \mathsf{F}$, then for every $d \in \mathbb{D}$ we have $\underline{\varphi(x)}(\nu[x := d]) \asymp \underline{\varphi(t)}(\nu)$.*

Proof. In the notation of Definition 4.2, $\underline{\varphi(t)}(\nu)$ is $\underline{\varphi}_{\perp}(\nu(1),\ldots,\perp,\ldots,\nu(n))$ and $\underline{\varphi(x)}(\nu[x := d])$ is $\underline{\varphi}_{\perp}(\nu(1),\ldots,d,\ldots,\nu(n))$. □

5 Is Defined -Formulas

To understand the motivation of the topic of this section, consider adding a function symbol for multiplicative inverses to the theory of real closed fields. The standard axiom $\forall x\,(x = 0 \vee x \cdot \frac{1}{x} = 1)$ does most of the job. When $x = 0$, then $(x \cdot \frac{1}{x} = 1) \asymp \mathsf{U}$, but $(x = 0 \vee x \cdot \frac{1}{x} = 1) \asymp \mathsf{T}$ by Definition 3.2(6). What this axiom fails to do is to tell that $\frac{1}{0}$ has been intentionally left undefined. It leaves open many possibilities, including $\frac{1}{0} = 0$ and $\frac{1}{0} = 1$. It thus leaves the axiomatization incomplete.

In everyday mathematics it is natural to use a first-order formula to specify the domain of a function. For instance, in the case of real numbers, $\frac{y}{x}$ is defined precisely when $x \neq 0$; $\sqrt{x}$ is defined precisely when $x \geq 0$; and $\log x$ is defined precisely when $x > 0$.

To formalize this idea, we assume that in a formal theory, each function symbol f has an associated *isdef-formula* $\lceil f \rfloor$, defined soon. The name is an abbreviation of "is defined -formula". While in classical binary first-order logic a theory consists of two components: a signature and a set of formulas on it (the axioms), in our logic a theory consists of three components: the signature, the axioms, and the isdef-formulas. The isdef-formulas will be defined so that they never yield U. We use $\lceil \mathscr{F} \rfloor$ to denote the mapping from the function symbols to their isdef-formulas, and $\lceil f \rfloor$ denotes the image of $f \in \mathscr{F}$. The notation $\lceil\,\rfloor$ used in Section 1 applies the idea to terms and formulas. It will be defined in terms of $\lceil \mathscr{F} \rfloor$ in Definition 5.4.

As was discussed in more detail towards the end of Section 2, $\varphi \cong \psi$ means that φ and ψ denote the literally same formula.

Definition 5.1. *Isdef-formulas* on a signature $(\mathscr{C}, \mathscr{F}, \mathscr{R}, \alpha)$ are a function $\lceil \mathscr{F} \rfloor$ from $\mathscr{F}$ to the formulas on the signature such that for every $f \in \mathscr{F}$,

1. $\lceil f \rfloor$ contains no other free variables than $\mathrm{v}_1, \ldots, \mathrm{v}_{\alpha(f)}$, and
2. for every function symbol g in $\lceil f \rfloor$, we have $\lceil g \rfloor \cong \mathsf{T}$.

To improve readability, in examples we may write x, y, and z instead of $\mathtt{v}_1$, $\mathtt{v}_2$, and $\mathtt{v}_3$. Here are some examples on familiar function symbols on real numbers:

$$\lceil x+y \rfloor \cong \mathsf{T},\ \lceil \sqrt{x} \rfloor \cong (x \geq 0),\text{ and } \lceil \tfrac{x}{y} \rfloor \cong (\neg(y=0)).$$

Because T, $(x \geq 0)$, and $(\neg(y=0))$ contain no function symbols at all, they vacuously satisfy Definition 5.1(2). Because multiplication is defined on all pairs of natural numbers, we may choose $\lceil x \cdot y \rfloor \cong \mathsf{T}$. Then an isdef-formula of the square root on natural numbers could be $\exists y\,(y \cdot y = x)$. It contains the function symbol $\cdot$.

The intention is that each function is defined precisely when its isdef-formula yields T. The next definition expresses this property, and tells how isdef-formulas are taken into account in the notions of model and logical consequence. The notation $(\sigma,\nu) \models \varphi$ and $(\sigma,\nu) \models \Gamma$ was introduced in Definition 3.5.

Definition 5.2. Let Γ be a set of formulas and $\lceil \mathscr{F} \rfloor$ be the isdef-formulas.

1. A *model of* $\lceil \mathscr{F} \rfloor$ is a structure $(\mathbb{D}, \_)$ such that for every function symbol f and every $d_1 \in \mathbb{D}$, ..., $d_{\alpha(f)} \in \mathbb{D}$, the following holds:

$$\lceil f \rfloor (d_1,\ldots,d_{\alpha(\lceil f \rfloor)}) \asymp \mathsf{T} \text{ if and only if } \underline{f}(d_1,\ldots,d_{\alpha(f)}) \text{ is defined.}$$

   This is denoted by $(\mathbb{D}, \_) \models \lceil \mathscr{F} \rfloor$.

2. A *model of* $(\lceil \mathscr{F} \rfloor, \Gamma)$ is a pair (σ,ν) such that σ is a structure, ν is an assignment of values to free variables, $\sigma \models \lceil \mathscr{F} \rfloor$, and $(\sigma,\nu) \models \Gamma$.

3. A formula φ is a *logical consequence* of $\lceil \mathscr{F} \rfloor$ and Γ, denoted by $(\lceil \mathscr{F} \rfloor, \Gamma) \models \varphi$, or $\Gamma \models \varphi$ for brevity, if and only if every model of $(\lceil \mathscr{F} \rfloor, \Gamma)$ also is a model of φ.

To illustrate the contribution of $\lceil \mathscr{F} \rfloor$ to the notion of logical consequence, let $\alpha(f) = 1$, $\lceil f \rfloor_1 \cong \mathsf{T}$, $\lceil f \rfloor_2 \cong \mathsf{F}$, and $\varphi \cong (f(x) = f(x))$. We have $(\lceil \mathscr{F} \rfloor_1, \emptyset) \models \varphi$ but $(\lceil \mathscr{F} \rfloor_2, \emptyset) \not\models \varphi$.

In this study, given a signature, Γ will vary frequently, but $\lceil \mathscr{F} \rfloor$ will remain the same. As a consequence, Γ is informative but $\lceil \mathscr{F} \rfloor$ is dead weight in the notation $(\lceil \mathscr{F} \rfloor, \Gamma) \models \varphi$. Therefore, we prefer the notation $\Gamma \models \varphi$, but remind at places that the concept also depends on $\lceil \mathscr{F} \rfloor$.

Definition 5.3. A *3-valued first-order theory* is a triple $(\mathscr{S}, \Gamma, \lceil \mathscr{F} \rfloor)$, where:

1. $\mathscr{S}$ is a signature,

2. Γ is a set of formulas on $\mathscr{S}$ (known as the *axioms*), and

3. $\lceil \mathscr{F} \rfloor$ is isdef-formulas on $\mathscr{S}$.

For instance, the function $\frac{1}{x}$ can be added to the classical binary first-order theory of real closed fields as follows. First, the isdef-formula T is introduced for each original function symbol, to make the theory 3-valued. Then, the symbol $\frac{1}{-}$ is added to the signature; the formula $\neg(x = 0)$ is made its isdef-formula; and the formula $\forall x\ (x = 0 \lor x \cdot \frac{1}{x} = 1)$ is added to the axioms. The square root function can be added to the theory of natural numbers by introducing the isdef-formulas T, adding $\sqrt{\ }$ to the signature, giving it the isdef-formula $\exists y\ (y \cdot y = x)$, and adding the axiom $(\neg\exists y\ (y \cdot y = x)) \lor (\sqrt{x} \cdot \sqrt{x} = x)$.

It is intuitively clear that, for instance, in the case of real numbers, $\frac{\sqrt{x+y}}{y+1}$ is defined if and only if $x + y \geq 0 \land y + 1 \neq 0$. Next we present and show correct a straightforward algorithm that implements this intuition. Given isdef-formulas, for each term t it computes a formula $\lceil t \rfloor$ and for each formula φ it computes a formula $\lceil \varphi \rfloor$. Given a structure that models the isdef-formulas, $\lceil t \rfloor$ yields T if and only if t is defined, and $\lceil \varphi \rfloor$ yields T if and only if φ does not yield U. The formulas $\lceil t \rfloor$ and $\lceil \varphi \rfloor$ themselves never yield U.

Definition 5.4. Assume that a signature and isdef-formulas on it are given.

1. If t is a variable symbol or a constant symbol, then $\lceil t \rfloor \cong \mathsf{T}$.
2. $\lceil f(t_1,\ldots,t_{\alpha(f)}) \rfloor \cong \lceil t_1 \rfloor \land \cdots \land \lceil t_{\alpha(f)} \rfloor \land \lceil f \rfloor(t_1,\ldots,t_{\alpha(\lceil f \rfloor)})$
(Where necessary, rename bound variables in $\lceil f \rfloor$ as justified by Lemma 3.4, so that $t_1, \ldots, t_{\alpha(\lceil f \rfloor)}$ become free for $\mathtt{v}_1, \ldots, \mathtt{v}_{\alpha(\lceil f \rfloor)}$.)
3. $\lceil \mathsf{F} \rfloor \cong \lceil \mathsf{T} \rfloor \cong \mathsf{T}$
4. $\lceil t_1 = t_2 \rfloor \cong \lceil t_1 \rfloor \land \lceil t_2 \rfloor$
5. $\lceil R(t_1,\ldots,t_{\alpha(R)}) \rfloor \cong \lceil t_1 \rfloor \land \cdots \land \lceil t_{\alpha(R)} \rfloor$
6. $\lceil \neg\varphi \rfloor \cong \lceil \varphi \rfloor$
7. $\lceil \varphi \land \psi \rfloor \cong (\lceil \varphi \rfloor \land \lceil \psi \rfloor) \lor (\lceil \varphi \rfloor \land \neg\varphi) \lor (\lceil \psi \rfloor \land \neg\psi)$
8. $\lceil \varphi \lor \psi \rfloor \cong (\lceil \varphi \rfloor \land \lceil \psi \rfloor) \lor (\lceil \varphi \rfloor \land \varphi) \lor (\lceil \psi \rfloor \land \psi)$
9. $\lceil \forall x\ \varphi \rfloor \cong (\forall x\ \lceil \varphi \rfloor) \lor \exists x\ (\lceil \varphi \rfloor \land \neg\varphi)$
10. $\lceil \exists x\ \varphi \rfloor \cong (\forall x\ \lceil \varphi \rfloor) \lor \exists x\ (\lceil \varphi \rfloor \land \varphi)$

Lemma 5.5. *Assume* $(\mathbb{D}, \_) \models \lceil \mathscr{F} \rfloor$. *Let t be a term and φ be a formula.*

1. *Every free variable in $\lceil t \rfloor$ also occurs in t, and every free variable in $\lceil \varphi \rfloor$ also occurs free in φ.*
2. *If for every function symbol f and every $d_1 \in \mathbb{D}$, ..., $d_{\alpha(f)} \in \mathbb{D}$ we have $\lceil f \rfloor(d_1,\ldots,d_{\alpha(\lceil f \rfloor)}) \asymp \mathsf{T}$, then for every term t, every formula φ, and every $d_1 \in \mathbb{D}$, ... we have $\lceil t \rfloor(d_1,\ldots,d_{\alpha(\lceil t \rfloor)}) \asymp \mathsf{T}$ and $\lceil \varphi \rfloor(d_1,\ldots,d_{\alpha(\lceil \varphi \rfloor)}) \asymp \mathsf{T}$.*

3. *Let* $d_1 \in \mathbb{D}, \ldots, d_{\alpha(t)} \in \mathbb{D}$*. If* $\underline{t}(d_1,\ldots,d_{\alpha(t)})$ *is defined, then* $\lceil t \rfloor(d_1,\ldots,d_{\alpha(\lceil t \rfloor)}) \asymp \mathsf{T}$*. Otherwise* $\lceil t \rfloor(d_1,\ldots,d_{\alpha(\lceil t \rfloor)}) \asymp \mathsf{F}$*.*
4. *Let* $d_1 \in \mathbb{D}, \ldots, d_{\alpha(\varphi)} \in \mathbb{D}$*. If* $\underline{\varphi}(d_1,\ldots,d_{\alpha(\varphi)}) \asymp \mathsf{F}$ *or* $\underline{\varphi}(d_1,\ldots,d_{\alpha(\varphi)}) \asymp \mathsf{T}$*, then* $\lceil \varphi \rfloor(d_1,\ldots,d_{\alpha(\lceil \varphi \rfloor)}) \asymp \mathsf{T}$*. Otherwise* $\lceil \varphi \rfloor(d_1,\ldots,d_{\alpha(\lceil \varphi \rfloor)}) \asymp \mathsf{F}$*.*
5. *Assume that* $\lceil \mathscr{F} \rfloor$ *is a computable function. Then the functions* $t \mapsto \lceil t \rfloor$ *and* $\varphi \mapsto \lceil \varphi \rfloor$ *are computable.*

Proof.

1. It is easy to check from Definition 5.4 that no case introduces other free variables than those in $t_1, \ldots, t_n$, $\lceil t_1 \rfloor, \ldots, \lceil t_n \rfloor$, φ, ψ, $\lceil \varphi \rfloor$, and $\lceil \psi \rfloor$, where $t_1, \ldots, t_n$ are the proper subterms and φ and ψ are the proper subformulas of the case. The claim follows from this immediately by induction.
2. Each case in Definition 5.4 is T, $\lceil t_1 \rfloor \wedge \cdots \wedge \lceil t_{\alpha(f)} \rfloor \wedge \lceil f \rfloor(t_1,\ldots,t_{\alpha(\lceil f \rfloor)})$, $\lceil t_1 \rfloor \wedge \cdots \wedge \lceil t_n \rfloor$ for some n, or $\lceil \varphi \rfloor$; or contains the disjunct $\lceil \varphi \rfloor \wedge \lceil \psi \rfloor$ or $\forall x\, \lceil \varphi \rfloor$. By induction and the assumption of the claim, each such formula or disjunct, and thus each case, yields T everywhere.
3. We use induction on the structure of t. For brevity we drop parameter lists of the form $(d_1,\ldots,d_{\alpha(\ )})$, but do show those that are of other forms.

   The base case consists of variable and constant symbols. By Definition 3.1(1) and (2), $\underline{\mathsf{v}_i}$ and $\underline{c}$ are defined, and by Definition 5.4(1) $\underline{\lceil \mathsf{v}_i \rfloor} \asymp \underline{\lceil c \rfloor} \asymp \mathsf{T}$.

   The induction step consists of terms of the form $f(t_1,\ldots,t_{\alpha(f)})$. By the induction hypothesis and Definition 5.2(1), there are three cases, each of which can be dealt with 3.1(3) and 5.4(2):

   (a) At least one of the $\underline{t_i}$ is undefined and $\underline{\lceil t_i \rfloor} \asymp \mathsf{F}$. Then $\underline{f(t_1,\ldots,t_{\alpha(f)})}$ is undefined and $\underline{\lceil f(t_1,\ldots,t_{\alpha(f)}) \rfloor} \asymp \mathsf{F}$.

   (b) Every $\underline{t_i}$ is defined and has $\underline{\lceil t_i \rfloor} \asymp \mathsf{T}$; and $\underline{f}(d'_1,\ldots,d'_{\alpha(f)})$ is defined, where each d'_i is the value of $\underline{t_i}$. By 5.2(1) we have $\lceil f \rfloor(d'_1,\ldots,d'_{\alpha(\lceil f \rfloor)}) \asymp \mathsf{T}$. Then $\underline{f(t_1,\ldots,t_{\alpha(f)})}$ is defined and $\underline{\lceil f(t_1,\ldots,t_{\alpha(f)}) \rfloor} \asymp \mathsf{T}$.

   (c) Every $\underline{t_i}$ is defined and has $\underline{\lceil t_i \rfloor} \asymp \mathsf{T}$; but $\underline{f}(d'_1,\ldots,d'_{\alpha(f)})$ is undefined, where each d'_i is the value of $\underline{t_i}$. By 5.2(1) we have $\lceil f \rfloor(d'_1,\ldots,d'_{\alpha(\lceil f \rfloor)}) \asymp \mathsf{F}$. Then $\underline{f(t_1,\ldots,t_{\alpha(f)})}$ is undefined and $\underline{\lceil f(t_1,\ldots,t_{\alpha(f)}) \rfloor} \asymp \mathsf{F}$.
4. We use induction on the structure of φ, and again drop parameter lists of the form $(d_1,\ldots,d_{\alpha(\ )})$. The base case consists of atomic formulas.

   (a) By Definition 3.2(1) $\underline{\mathsf{F}} \asymp \mathsf{F}$ and $\underline{\mathsf{T}} \asymp \mathsf{T}$, which matches Definition 5.4(3).

   (b) If $\underline{t_1}$ or $\underline{t_2}$ is undefined, then by 3.2(2) $\underline{t_1 = t_2} \asymp \mathsf{U}$. By Claim 3, $\underline{\lceil t_1 \rfloor} \asymp \mathsf{F}$ or $\underline{\lceil t_2 \rfloor} \asymp \mathsf{F}$, so by 5.4(4) $\underline{\lceil t_1 = t_2 \rfloor} \asymp \mathsf{F}$. Otherwise $\underline{t_1}$ and $\underline{t_2}$ are defined. Then by 3.2(2) $\underline{t_1 = t_2} \asymp \mathsf{F}$ or $\underline{t_1 = t_2} \asymp \mathsf{T}$. By Claim 3, $\underline{\lceil t_1 \rfloor} \asymp \underline{\lceil t_2 \rfloor} \asymp \mathsf{T}$, so by 5.4(4) $\underline{\lceil t_1 = t_2 \rfloor} \asymp \mathsf{T}$.

(c) The case $\underline{R}(t_1,\ldots,t_{\alpha(R)})$ is proven similarly to (b) using 3.2(3) and 5.4(5).

The induction step consists of five cases.

(a) By 3.2(4), $\underline{\neg\varphi} \asymp \mathsf{U}$ if and only if $\underline{\varphi} \asymp \mathsf{U}$. This matches 5.4(6).

(b) If $\underline{\varphi} \asymp \underline{\psi} \asymp \mathsf{T}$, then by 3.2(5) $\underline{\varphi \land \psi} \asymp \mathsf{T}$. By the induction assumption and 5.4(7), $\underline{\lceil\varphi\rfloor}$, $\underline{\lceil\psi\rfloor}$, and $\underline{\lceil\varphi\land\psi\rfloor}$ yield T. If $\underline{\varphi} \asymp \mathsf{F}$, then $\underline{\varphi\land\psi} \asymp \mathsf{F}$, and $\underline{\lceil\varphi\rfloor}$, $\underline{(\lceil\varphi\rfloor \land \neg\varphi)}$, and $\underline{\lceil\varphi\land\psi\rfloor}$ yield T. Similarly if $\underline{\psi} \asymp \mathsf{F}$, then $\underline{\varphi\land\psi} \asymp \mathsf{F}$ and $\underline{\lceil\varphi\land\psi\rfloor} \asymp \mathsf{T}$. In the remaining cases, at least one of $\underline{\varphi}$ and $\underline{\psi}$ yields U while the other yields T or U, $\underline{\varphi\land\psi} \asymp \mathsf{U}$, at least one of $\underline{\lceil\varphi\rfloor}$ and $\underline{\lceil\psi\rfloor}$ yields F, and $\underline{\lceil\varphi\land\psi\rfloor} \asymp \mathsf{F}$.

(c) The case $\varphi \lor \psi$ is proven similarly to (b), but using 3.2(6) and 5.4(8), and with F and T swapped except when yielded by $\lceil\rfloor$.

(d) To deal with $\forall x\, \varphi$, let $\underline{\varphi}(;e)$ abbreviate $\underline{\varphi}(d_1,\ldots,d_{\alpha(\varphi)})$, if x is none of $\mathrm{v}_1, \ldots, \mathrm{v}_{\alpha(\varphi)}$; and otherwise $\underline{\varphi}(;e)$ abbreviates $\underline{\varphi}(d_1,\ldots,e,\ldots,d_{\alpha(\varphi)})$, where e is used in the place of x. If for every $e \in \mathbb{D}$ we have $\underline{\varphi}(;e) \asymp \mathsf{T}$, then by 3.2(7) $\underline{\forall x\, \varphi} \asymp \mathsf{T}$. By the induction assumption, for every $e \in \mathbb{D}$ we have $\underline{\lceil\varphi\rfloor(;e)} \asymp \mathsf{T}$. That is, $\underline{\forall x\, \lceil\varphi\rfloor} \asymp \mathsf{T}$. By 5.4(9), $\underline{\lceil\forall x\, \varphi\rfloor} \asymp \mathsf{T}$. If $e \in \mathbb{D}$ is such that $\underline{\varphi}(;e) \asymp \mathsf{F}$, then $\underline{\forall x\, \varphi} \asymp \mathsf{F}$ and $\underline{\lceil\varphi\rfloor}(;e) \asymp \mathsf{T}$. So $\underline{\exists x\, (\lceil\varphi\rfloor \land \neg\varphi)} \asymp \mathsf{T}$ and $\underline{\lceil\forall x\, \varphi\rfloor} \asymp \mathsf{T}$. The case remains where $\underline{\varphi}(;e) \asymp \mathsf{U}$ for at least one $e \in \mathbb{D}$ and $\underline{\varphi}(;e) \asymp \mathsf{F}$ for no $e \in \mathbb{D}$. Then $\underline{\forall x\, \varphi} \asymp \mathsf{U}$ and $\underline{\lceil\forall x\, \varphi\rfloor} \asymp \mathsf{F}$.

(e) The case $\exists x\, \varphi$ is proven similarly to (d), but using 3.2(8) and 5.4(10), and with F and T swapped except when yielded by $\lceil\rfloor$.

5. The algorithm is immediate from Definition 5.4, except perhaps the renaming of bound variables in (2). A simple possibility starts by scanning $t_1, \ldots, t_{\alpha(f)}$, to find the greatest i such that $i = 0$ or v_i occurs in at least one of them. Then it replaces the bound variables in $\lceil f\rfloor$ by v_{i+1}, v_{i+2}, and so on.

□

For example, $\left\lceil \frac{\sqrt{x+y}}{y+1} \right\rfloor$

$$
\begin{aligned}
&\cong \lceil\sqrt{x+y}\rfloor \land \lceil y+1\rfloor \land (\neg(y+1=0)) \\
&\cong (\lceil x+y\rfloor \land (x+y \geq 0)) \land (\lceil y\rfloor \land \lceil 1\rfloor \land \mathsf{T}) \land (\neg(y+1=0)) \\
&\cong ((\lceil x\rfloor \land \lceil y\rfloor \land \mathsf{T}) \land (x+y \geq 0)) \land (\mathsf{T} \land \mathsf{T} \land \mathsf{T}) \land (\neg(y+1=0)) \\
&\cong ((\mathsf{T} \land \mathsf{T} \land \mathsf{T}) \land (x+y \geq 0)) \land (\mathsf{T} \land \mathsf{T} \land \mathsf{T}) \land (\neg(y+1=0))
\end{aligned}
$$

By Definition 3.2(1) and (5) and the usual definition of $\neq$, this formula expresses the same function as $x+y \geq 0 \land y+1 \neq 0$.

We emphasize that there is no symbol $\lceil\rfloor$ in the formal language. Every instance of $\lceil\rfloor$ is a metalanguage expression that represents the formal language expression that is obtained by Definition 5.4.

Now $\varphi \twoheadrightarrow \psi$ can be introduced as a shorthand for $\neg\varphi \vee \psi \vee \neg(\lceil\varphi\rfloor \vee \lceil\psi\rfloor)$. This is not in contradiction with the facts that our logic is regular and $\twoheadrightarrow$ cannot be expressed in a regular logic, because $\lceil\varphi\rfloor$ only exists in the metalanguage. It does not represent a truth function but a function from formulas to formulas.

6 Proof System and Its Soundness

Our proof system is loosely based on a proof system for classical binary first-order logic in [7, Section 3.1]. The most important (but not only) difference is that our system depends on the isdef-formulas $\lceil f\rfloor$ of the function symbols $f \in \mathscr{F}$. Therefore, its soundness proof will use Lemma 5.5(3) and (4). Thanks to 5.5(5), if the set of axioms is recursive and the function $\lceil\mathscr{F}\rfloor$ is computable, then also the proof system is recursive. If $\mathscr{F}$ is finite, then it is trivial that $\lceil\mathscr{F}\rfloor$ is computable.

The notation $(\lceil\mathscr{F}\rfloor, \Gamma) \vdash \varphi$ means that φ can be proven from $\lceil\mathscr{F}\rfloor$ and Γ using the proof system. We usually write it more briefly as $\Gamma \vdash \varphi$, just like we do with $\models$, because $\lceil\mathscr{F}\rfloor$ never changes during our argumentation. The meaning of $\Gamma \models \varphi$ was given in Definition 5.2(3). Given a signature and the isdef-formulas, a proof system is *sound* if and only if for every Γ and φ, $\Gamma \vdash \varphi$ implies $\Gamma \models \varphi$.

In what follows, φ, ψ and χ are arbitrary formulas, Γ and Δ are arbitrary sets of formulas, $t, t_1, \ldots, t_n$ are arbitrary terms, and x and y are arbitrary variable symbols. The rule schemas that differ from or are absent in classical binary first-order logic have been marked with (*) at the end of the first line of the rule schema. After each group of rule schemas, we show their soundness if it is not immediately obvious. The soundness proof is by induction, where the induction assumption says that every $\vdash$ in the if-part of the schema is sound.

The first three rule schemas allow thinking of proofs as maintaining a set of axioms and proven formulas, which grows each time a new formula is proven.

P1 $\{\varphi\} \vdash \varphi$

P2 If $\Gamma \vdash \varphi$ then $\Gamma \cup \Delta \vdash \varphi$.

P3 If $\Gamma \vdash \varphi$ and $\Gamma \cup \{\varphi\} \vdash \psi$, then $\Gamma \vdash \psi$.

Lemma 6.1. *P1, P2, and P3 are sound.*

Proof.

P1 For every σ and ν such that $\sigma \models \lceil\mathscr{F}\rfloor$ and $(\sigma,\nu) \models \{\varphi\}$, obviously $(\sigma,\nu) \models \varphi$. This means that by Definition 5.2(3), P1 is sound.

P2 If $(\sigma,\nu) \models \Gamma \cup \Delta$, then clearly $(\sigma,\nu) \models \Gamma$. If, furthermore, $\sigma \models \lceil\mathscr{F}\rfloor$ and $\Gamma \vdash \varphi$, then by the induction assumption $(\sigma,\nu) \models \varphi$, showing that P2 is sound.

P3 If $\sigma \models \lceil\mathscr{F}\rfloor$, $(\sigma,\nu) \models \Gamma$, and $\Gamma \vdash \varphi$, then $(\sigma,\nu) \models \Gamma \cup \{\varphi\}$. Then $\Gamma \cup \{\varphi\} \vdash \psi$ yields $(\sigma,\nu) \models \psi$. Therefore, P3 is sound.

□

The next rule schema expresses the Law of Excluded Fourth, which replaces the Law of Excluded Middle in classical logic. The next two are the basis of proof by contradiction. The fourth one says that if a formula is true, then it is also defined.

C1 $\emptyset \vdash (\varphi \vee \neg\varphi) \vee \neg\lceil\varphi\rfloor$ (*)

C2 $\{\mathsf{F}\} \vdash \varphi$

C3 $\{\varphi, \neg\varphi\} \vdash \mathsf{F}$

D1 $\{\varphi\} \vdash \lceil\varphi\rfloor$ (*)

Lemma 6.2. *C1, C2, C3, and D1 are sound.*

Proof. Assume $\sigma \models \lceil\mathscr{F}\rfloor$. By Lemma 5.5(4), if $\underline{\varphi}(\nu)$ yields neither T nor F, then $\underline{\lceil\varphi\rfloor}(\nu) \asymp \mathsf{F}$. So C1 is sound. The soundness of D1 follows immediately from the same lemma. The rule schemas C2 and C3 are vacuously sound, because there are no σ and ν such that $(\sigma,\nu) \models \mathsf{F}$ or $(\sigma,\nu) \models \{\varphi,\neg\varphi\}$. □

Here is an example of a rule instance generated by D1:

$$\left\{\frac{\sqrt{x}}{x-1} > 0\right\} \vdash \big(\mathsf{T} \wedge (x \geq 0)\big) \wedge \big(\mathsf{T} \wedge \mathsf{T} \wedge \mathsf{T}\big) \wedge \big(\neg(x-1=0)\big)$$

Here is an example of a rule instance generated by C1, made more human-readable by dropping a number of parentheses and "$\mathsf{T} \wedge$":

$$\emptyset \vdash \left(\frac{\sqrt{x}}{x-1} > 0\right) \vee \neg\left(\frac{\sqrt{x}}{x-1} > 0\right) \vee \neg\Big((x \geq 0) \wedge \neg(x-1=0)\Big)$$

We now illustrate the use of some rule schemas introduced this far, and obtain two results that are needed later. C4 is actually a theorem schema and its proof is a proof schema. They become a theorem and a proof in our system by putting a formula in the place of φ. Also C5 is a theorem schema. Our proof of it is not a proof schema but a demonstration that the set of rules that C5 generates is a subset of those generated by C4.

Lemma 6.3. *For every formula φ, our proof system proves the following:*

C4 $\{\varphi, \neg\lceil\varphi\rfloor\} \vdash \mathsf{F}$

C5 $\{\neg\varphi, \neg\lceil\varphi\rfloor\} \vdash \mathsf{F}$

Proof.

C4 By D1, $\{\varphi\} \vdash \lceil\varphi\rfloor$. By P2, $\{\varphi, \neg\lceil\varphi\rfloor\} \vdash \lceil\varphi\rfloor$. Let this be called (1). Using $\lceil\varphi\rfloor$ in the place of φ in C3, we get $\{\lceil\varphi\rfloor, \neg\lceil\varphi\rfloor\} \vdash \mathsf{F}$. By P2, $\{\varphi, \neg\lceil\varphi\rfloor, \lceil\varphi\rfloor\}$ $\vdash \mathsf{F}$. Let this be called (2). By (1), (2) and P3, $\{\varphi, \neg\lceil\varphi\rfloor\} \vdash \mathsf{F}$.

C5 With $\neg\varphi$ in the place of φ, C4 yields $\{\neg\varphi, \neg\lceil\neg\varphi\rfloor\} \vdash \mathsf{F}$. By Definition 5.4(6) $\lceil\neg\varphi\rfloor$ is literally the same formula as $\lceil\varphi\rfloor$. Therefore, $\{\neg\varphi, \neg\lceil\neg\varphi\rfloor\} \vdash \mathsf{F}$ is literally the same as $\{\neg\varphi, \neg\lceil\varphi\rfloor\} \vdash \mathsf{F}$.

□

It is clear that the above style of proof is clumsy indeed. Therefore, we now argue that a finite sequence of subproofs and sets $\Gamma_0, \ldots, \Gamma_n$ can also be thought of as a proof, where $n > 0$, $\Gamma_0 = \Gamma$, $\Gamma_i = \Gamma_{i-1} \cup \{\varphi_i\}$, and φ_i is obtained from some subset of Γ_{i-1} by some rule schema or subproof. We do that by verifying by induction that if $0 \leq k \leq n$, then $\Gamma_{n-k} \vdash \varphi_n$. The claim then follows by choosing $k = n$.

By P1 $\{\varphi_n\} \vdash \varphi_n$. So by P2, $\Gamma_{n-0} \vdash \varphi_n$. Thus the claim holds for $k = 0$. By the induction hypothesis $\Gamma_{n-k} \vdash \varphi_n$. By P2 and the assumption on how φ_i is obtained, $\Gamma_{(n-k)-1} \vdash \varphi_{n-k}$. Since $\Gamma_{n-k} = \Gamma_{n-k-1} \cup \{\varphi_{n-k}\}$, P3 yields $\Gamma_{n-(k+1)} \vdash \varphi_n$.

By this observation, proofs can be presented in a compressed form $\Gamma \vdash_{\mathrm{x}_1} \varphi_1 \vdash_{\mathrm{x}_2} \cdots \vdash_{\mathrm{x}_n} \varphi_n$, where $\mathrm{x}_1, \ldots, \mathrm{x}_n$ are labels of rule schemas or lemmas, and φ_i can be obtained by x_i from some subset of $\Gamma \cup \{\varphi_1, \ldots, \varphi_{i-1}\}$. Furthermore, more information can be added to the index x_i, to tell about how the rule schema or lemma is applied. In this representation, the proof of C4 may be written as $\{\varphi, \neg\lceil\varphi\rfloor\} \vdash_{\mathrm{D1}} \lceil\varphi\rfloor \vdash_{\mathrm{C3}} \mathsf{F}$.

The following rule schemas for conjunction and disjunction are mostly trivial. The schema $\vee$-E expresses the principle of proof by cases.

$\wedge$-I $\{\varphi, \psi\} \vdash \varphi \wedge \psi$

$\wedge$-E1 $\{\varphi \wedge \psi\} \vdash \varphi$

$\wedge$-E2 $\{\varphi \wedge \psi\} \vdash \psi$

$\vee$-I1 $\{\varphi\} \vdash \varphi \vee \psi$

$\vee$-I2 $\{\psi\} \vdash \varphi \vee \psi$

$\vee$-E If $\Gamma \cup \{\varphi\} \vdash \chi$ and $\Gamma \cup \{\psi\} \vdash \chi$, then $\Gamma \cup \{\varphi \vee \psi\} \vdash \chi$.

Lemma 6.4. *$\wedge$-I, $\wedge$-E1, $\wedge$-E2, $\vee$-I1, $\vee$-I2, and $\vee$-E are sound.*

Proof. If $\sigma \models \lceil\mathscr{F}\rfloor$ and $(\sigma, \nu) \models \Gamma \cup \{\varphi \vee \psi\}$, then $(\sigma, \nu) \models \Gamma$ and either $(\sigma, \nu) \models \varphi$ or $(\sigma, \nu) \models \psi$ or both. In the former case $\Gamma \cup \{\varphi\} \vdash \chi$ yields $(\sigma, \nu) \models \chi$; and in the latter case $\Gamma \cup \{\psi\} \vdash \chi$ yields $(\sigma, \nu) \models \chi$. Therefore, $\vee$-E is sound. The soundness proofs of the other five are immediate. □

The following lemma illustrates subproofs and proofs by cases, and gives another five results that are needed later. The last three of them are examples of the ability of the proof system of exploiting the commutativity and associativity of $\vee$.

Lemma 6.5. *For every formula φ, ψ, and χ, our proof system proves the following:*

D2 $\emptyset \vdash \lceil\varphi\rfloor \vee \neg\lceil\varphi\rfloor$

C6 $\emptyset \vdash \mathsf{T}$

$\vee$**-C** $\{\varphi \vee \psi\} \vdash \psi \vee \varphi$

$\vee$**-A** $\{(\varphi \vee \psi) \vee \chi\} \vdash \varphi \vee (\psi \vee \chi)$

$\vee$**-A'** $\{(\varphi \vee \psi) \vee \chi\} \vdash \varphi \vee (\chi \vee \psi)$

Proof.

D2 Since $\{\varphi\} \vdash_{\text{D1}} \lceil\varphi\rfloor$ and $\{\neg\varphi\} \vdash_{\text{D1}} \lceil\neg\varphi\rfloor \cong_{5.4(6)} \lceil\varphi\rfloor$, we get $\{\varphi \vee \neg\varphi\} \vdash_{\vee\text{-E}} \lceil\varphi\rfloor \vdash_{\vee\text{-I1}} \lceil\varphi\rfloor \vee \neg\lceil\varphi\rfloor$. On the other hand, $\{\neg\lceil\varphi\rfloor\} \vdash_{\vee\text{-I2}} \lceil\varphi\rfloor \vee \neg\lceil\varphi\rfloor$. Therefore, $\emptyset \vdash_{\text{C1}} (\varphi \vee \neg\varphi) \vee \neg\lceil\varphi\rfloor \vdash_{\vee\text{-E}} \lceil\varphi\rfloor \vee \neg\lceil\varphi\rfloor$.

C6 Since $\{\mathsf{T}\} \vdash_{\text{P1}} \mathsf{T}$ and $\{\neg\mathsf{T}\} \vdash_{\text{D1}} \lceil\neg\mathsf{T}\rfloor \cong \lceil\mathsf{T}\rfloor \cong_{5.4(3)} \mathsf{T}$, we reason $\emptyset \vdash_{\text{D2}} \lceil\mathsf{T}\rfloor \vee \neg\lceil\mathsf{T}\rfloor \cong \mathsf{T} \vee \neg\mathsf{T} \vdash_{\vee\text{-E}} \mathsf{T}$.

$\vee$-C $\{\varphi\} \vdash_{\vee\text{-I2}} \psi \vee \varphi$ and $\{\psi\} \vdash_{\vee\text{-I1}} \psi \vee \varphi$, thus $\{\varphi \vee \psi\} \vdash_{\vee\text{-E}} \psi \vee \varphi$.

$\vee$-A $\{\varphi\} \vdash_{\vee\text{-I1}} \varphi \vee (\psi \vee \chi)$ and $\{\psi\} \vdash_{\vee\text{-I1}} \psi \vee \chi \vdash_{\vee\text{-I2}} \varphi \vee (\psi \vee \chi)$, hence $\{\varphi \vee \psi\} \vdash_{\vee\text{-E}} \varphi \vee (\psi \vee \chi)$. We also have $\{\chi\} \vdash_{\vee\text{-I2}} \psi \vee \chi \vdash_{\vee\text{-I2}} \varphi \vee (\psi \vee \chi)$. Therefore, $\{(\varphi \vee \psi) \vee \chi\} \vdash_{\vee\text{-E}} \varphi \vee (\psi \vee \chi)$.

$\vee$-A' The proof is like the previous proof, but builds $\chi \vee \psi$ instead of $\psi \vee \chi$.

□

Lemma 6.6. *If α, β, γ is any permutation of φ, ψ, χ, then $\{(\varphi \vee \psi) \vee \chi\} \vdash \alpha \vee (\beta \vee \gamma)$.*

Proof. $\{(\varphi \vee \psi) \vee \chi\} \vdash_{\vee\text{-A}} \varphi \vee (\psi \vee \chi) \vdash_{\vee\text{-C}} (\psi \vee \chi) \vee \varphi \vdash_{\vee\text{-A}} \psi \vee (\chi \vee \varphi) \vdash_{\vee\text{-C}} (\chi \vee \varphi) \vee \psi \vdash_{\vee\text{-A'}} \chi \vee (\psi \vee \varphi) \vdash_{\vee\text{-C}} (\psi \vee \varphi) \vee \chi \vdash_{\vee\text{-A}} \psi \vee (\varphi \vee \chi) \vdash_{\vee\text{-C}} (\varphi \vee \chi) \vee \psi \vdash_{\vee\text{-A}} \varphi \vee (\chi \vee \psi) \vdash_{\vee\text{-C}} (\chi \vee \psi) \vee \varphi \vdash_{\vee\text{-A'}} \chi \vee (\varphi \vee \psi)$ □

By Lemma 5.5(4) and Definition 3.2(4), for every formula φ and for each interpretation, precisely one of $\underline{\varphi}$, $\underline{\neg\varphi}$ and $\underline{\neg\lceil\varphi\rfloor}$ yields T. Our proof system reflects the same idea in the following way. The set Γ is *inconsistent* if and only if $\Gamma \vdash \mathsf{F}$. C3, C4, and C5 tell that if Γ is consistent, then at most one of φ, $\neg\varphi$, and $\neg\lceil\varphi\rfloor$ can be proven. The next lemma tells that if two of them lead to a contradiction, then the third one can be proven.

Lemma 6.7. *Assume that α, β, γ is any permutation of $\varphi, \neg\varphi, \neg\lceil\varphi\rfloor$. If $\Gamma \cup \{\alpha\} \vdash \mathsf{F}$, then $\Gamma \vdash \beta \vee \gamma$. If $\Gamma \cup \{\alpha\} \vdash \mathsf{F}$ and $\Gamma \cup \{\beta\} \vdash \mathsf{F}$, then $\Gamma \vdash \gamma$.*

Proof. $\emptyset \vdash_{\text{C1}} (\varphi \vee \neg\varphi) \vee \neg\lceil\varphi\rfloor$, from which Lemma 6.6 yields $\emptyset \vdash \alpha \vee (\beta \vee \gamma)$. Assume $\Gamma \cup \{\alpha\} \vdash \mathsf{F}$. Then $\Gamma \cup \{\alpha\} \vdash_{\text{C2}} \beta \vee \gamma$. On the other hand, $\Gamma \cup \{\beta \vee \gamma\} \vdash_{\text{P1}} \beta \vee \gamma$. Thus $\Gamma \cup \{\alpha \vee (\beta \vee \gamma)\} \vdash_{\vee\text{-E}} \beta \vee \gamma$, and $\Gamma \vdash_{\text{P3}} \beta \vee \gamma$. If also $\Gamma \cup \{\beta\} \vdash \mathsf{F}$, then $\Gamma \cup \{\beta\} \vdash_{\text{C2}} \gamma$ and $\Gamma \cup \{\gamma\} \vdash_{\text{P1}} \gamma$. So $\Gamma \cup \{\beta \vee \gamma\} \vdash_{\vee\text{-E}} \gamma$, and by $\Gamma \vdash \beta \vee \gamma$ we get $\Gamma \vdash_{\text{P3}} \gamma$. □

Next we discuss rule schemas for equality. The first of them differs from classical binary first-order logic in that it insists the term in question be defined. Without this restriction, we could, for instance, prove $\frac{1}{0} = \frac{1}{0}$ using =-1.

=-1 $\{\lceil t \rfloor\} \vdash (t = t)$ (*)

=-2 If $\varphi(x_1, \ldots, x_{\alpha(\varphi)})$ is a formula, $1 \leq i \leq \alpha(\varphi)$, and t_i and t'_i are free for x_i in $\varphi(x_1, \ldots, x_{\alpha(\varphi)})$, then $\{(t_i = t'_i), \varphi(t_1, \ldots, t_{\alpha(\varphi)})\} \vdash \varphi(t_1, \ldots, t'_i, \ldots, t_{\alpha(\varphi)})$.

Lemma 6.8. *=-1 and =-2 are sound.*

Proof.

=-1 Assume $\sigma \models \lceil \mathscr{F} \rfloor$ and $(\sigma, \nu) \models \lceil t \rfloor$. The latter means that $\underline{\lceil t \rfloor}(\nu) \asymp \mathsf{T}$. By Lemma 5.5(3), $\underline{t}(\nu)$ is defined. So $\underline{(t = t)}(\nu) \asymp_{3.2(2)} \mathsf{T}$, that is, $(\sigma, \nu) \models (t = t)$.

=-2 If $(\sigma, \nu) \models (t_i = t'_i)$, then by Definition 3.2(2) both $\underline{t_i}(\nu)$ and $\underline{t'_i}(\nu)$ are defined, and they yield the same value. Because both t_i and t'_i are free for x_i in $\varphi(x_i)$, the values yielded by $\underline{t_i}(\nu)$ and $\underline{t'_i}(\nu)$ are treated in the same way in $\underline{\varphi(t_i)}(\nu)$ and $\underline{\varphi(t'_i)}(\nu)$. As a consequence, if $(\sigma, \nu) \models \varphi(t_i)$, then also $(\sigma, \nu) \models \varphi(t'_i)$.

□

The next lemma tells that our proof system allows the substitution of a term for an equal term inside a defined term. Furthermore, it can exploit symmetry and transitivity of $=$.

Lemma 6.9. *For every function symbol f, every $1 \leq i \leq \alpha(f)$, and every term t_i, t'_i, t, t', t_1, t_2, and t_3, our proof system proves the following:*

=-3 $\{(t_i = t'_i), \lceil f(t_1, \ldots, t_{\alpha(f)}) \rfloor\} \vdash (f(t_1, \ldots, t_{\alpha(f)}) = f(t_1, \ldots, t'_i, \ldots, t_{\alpha(f)}))$

=-4 $\{(t = t')\} \vdash (t' = t)$

=-5 $\{(t_1 = t_2), (t_2 = t_3)\} \vdash (t_1 = t_3)$

Proof.

=-3 Let $\varphi(x_i) \cong (f(t_1, \ldots, t_{\alpha(f)}) = f(t_1, \ldots, x_i, \ldots, t_{\alpha(f)}))$. It contains no quantifiers, so t_i and t'_i are free for x_i in it. We have $\{\lceil f(t_1, \ldots, t_{\alpha(f)}) \rfloor\} \vdash_{=\text{-}1} \varphi(t_i)$ and $\{(t_i = t'_i), \varphi(t_i)\} \vdash_{=\text{-}2} \varphi(t'_i) \cong (f(t_1, \ldots, t_{\alpha(f)}) = f(t_1, \ldots, t'_i, \ldots, t_{\alpha(f)}))$. The claim now follows by P2 and P3.

=-4 Let $\varphi(x) \cong (x = t)$. We reason $\{(t = t')\} \vdash_{\text{D1}} \lceil t = t' \rfloor \cong_{5.4(4)} \lceil t \rfloor \wedge \lceil t' \rfloor \vdash_{\wedge\text{-E1}} \lceil t \rfloor \vdash_{=\text{-}1} (t = t) \cong \varphi(t)$. Clearly $\{(t = t'), \varphi(t)\} \vdash_{=\text{-}2} \varphi(t') \cong (t' = t)$. The claim now follows by P3.

=-5 Let $\varphi(x) \cong (x = t_3)$. By abuse of $\cong$ we reason $\{(t_1 = t_2), (t_2 = t_3)\} \cong \{(t_1 = t_2), \varphi(t_2)\} \vdash_{=\text{-}4} (t_2 = t_1) \vdash_{=\text{-}2} \varphi(t_1) \cong (t_1 = t_3)$.

□

Rule schemas for quantifiers have one difference from classical binary first-order logic: ∀-E insists that the term t must be defined. This prevents us from deriving, for instance, $0 \cdot \frac{1}{0} = 0$ from $\forall x\ (0 \cdot x = 0)$. We will see that thanks to Corollary 4.4, ∃-I does not need a similar condition.

∀-E If t is free for x in $\varphi(x)$, then $\{\lceil t \rfloor, (\forall x\ \varphi(x))\} \vdash \varphi(t)$. (*)

∀-I If $\Gamma \vdash \varphi(x)$ and x does not occur free in Γ, then $\Gamma \vdash (\forall x\ \varphi(x))$.

∃-I If t is free for x in $\varphi(x)$, then $\{\varphi(t)\} \vdash (\exists x\ \varphi(x))$.

∃-E If $\Gamma \cup \{\varphi(y)\} \vdash \psi$ and y does not occur in Γ, $(\exists x\ \varphi(x))$, nor in ψ, then $\Gamma \cup \{(\exists x\ \varphi(x))\} \vdash \psi$.

Lemma 6.10. *∀-E, ∀-I, ∃-I, and ∃-E are sound.*

Proof.

∀-E If $\sigma \models \lceil \mathscr{F} \rfloor$ and $(\sigma, \nu) \models \lceil t \rfloor$, then $\underline{t}(\nu)$ is defined. Let its value be denoted by d. If also $(\sigma, \nu) \models (\forall x\ \varphi(x))$, then $(\sigma, \nu[x := d]) \models \varphi(x)$. Because t is free for x, also $(\sigma, \nu) \models \varphi(t)$ by Lemma 3.6. Thus ∀-E is sound.

∀-I If $(\sigma, \nu) \models \Gamma$ and x does not occur free in Γ, then for any $d \in \mathbb{D}$ we have $(\sigma, \nu[x := d]) \models \Gamma$. If also $\sigma \models \lceil \mathscr{F} \rfloor$ and $\Gamma \vdash \varphi(x)$, then $(\sigma, \nu[x := d]) \models \varphi(x)$. This means that $(\sigma, \nu) \models (\forall x\ \varphi(x))$, implying that ∀-I is sound.

∃-I Assume $(\sigma, \nu) \models \varphi(t)$. Because t is free for x in $\varphi(x)$, if $\underline{t}(\nu)$ is defined, then $(\sigma, \nu[x := \underline{t}(\nu)]) \models \varphi(x)$ by Lemma 3.6. Therefore, $(\sigma, \nu) \models (\exists x\ \varphi(x))$. Otherwise, $\underline{t}(\nu)$ is undefined. Because $(\sigma, \nu) \models \varphi(t)$, by Corollary 4.4 $(\sigma, \nu[x := d]) \models \varphi(x)$ for every $d \in \mathbb{D}$. Because $\mathbb{D} \neq \emptyset$ by definition, this implies $(\sigma, \nu[x := d]) \models \varphi(x)$ for at least one $d \in \mathbb{D}$, that is, $(\sigma, \nu) \models (\exists x\ \varphi(x))$. So ∃-I is sound.

∃-E Assume $(\sigma, \nu) \models \Gamma \cup \{(\exists x\ \varphi(x))\}$. So $(\sigma, \nu[x := d]) \models \varphi(x)$ for at least one $d \in \mathbb{D}$. Because y does not occur in $(\exists x\ \varphi(x))$, y is free for x in $\varphi(x)$. Thus also $(\sigma, \nu[y := d]) \models \varphi(y)$. Because y does not occur in Γ, $(\sigma, \nu) \models \Gamma$ implies $(\sigma, \nu[y := d]) \models \Gamma$. If also $\sigma \models \lceil \mathscr{F} \rfloor$ and $\Gamma \cup \{\varphi(y)\} \vdash \psi$, we get $(\sigma, \nu[y := d]) \models \psi$. Because y does not occur in ψ, we have $(\sigma, \nu) \models \psi$. This completes the soundness proof of ∃-E.

□

Altogether, there are only four differences from classical binary first-order logic: two that make each closed formula yield precisely one of three truth values instead of two; one that enforces that an undefined term is not equal to anything; and one that reflects the principle that variables range over defined values only.

7 Existence of Models and Completeness

Our proofs for the model existence theorem and completeness theorem mimic [21], Proposition 5.7, pp. 107–110, which we have adapted to 3-valued first-order logic and our proof system. We have also attempted to clarify many technicalities. Ultimately the proofs are based on the well-known construction by Leon Henkin [12].

Let Γ be a set of formulas. Because it may be infinite, it is possible that every variable symbol occurs in it. However, the Henkin construction needs infinitely many additional variable symbols. Further headache is caused by the fact that a term may be not free for a variable symbol in a formula. In what follows we cannot rely on Lemma 3.4, because the Henkin construction uses the formula literally as it is. Therefore, we will use a provably equivalent term instead that is substitutable in the original formula. So for each finite set of variable symbols, each term must have a provably equivalent term that contains none of the variable symbols. To deal with these, we introduce a new indexing for the variable symbols and a modified Γ called Γ' as follows.

The function $\iota : \mathbb{Z}^+ \times \mathbb{Z}^+ \to \mathbb{Z}^+; (i, j) \mapsto \frac{1}{2}(i+j-1)(i+j-2)+j$ is a bijection. Therefore, we get an alternative indexing for the variable symbols by, for $i \in \mathbb{Z}^+$ and $j \in \mathbb{Z}^+$, defining that $\mathtt{v}_{i,j}$ is the same variable symbol as $\mathtt{v}_{\iota(i,j)}$. The set Γ' is obtained by replacing all variable symbols $\mathtt{v}_i$ in Γ by the variable symbols $\mathtt{v}_{2i,1}$, and then, for $j > 1$, adding the formulas $(\mathtt{v}_{i,1} = \mathtt{v}_{i,j})$. The function $\lceil \mathscr{F} \rfloor$ need not be modified, because of the following. For each f, the only occurrence of $\lceil f \rfloor$ in our formalism is in Definition 5.4(2). There the names of its variables are insignificant, because free variables have been substituted by terms and bound variables have been chosen so that the substitution is legal.

Let $\varphi(x)$ be any formula, t any term, and $\mathtt{v}_{i,j}$ any variable symbol that occurs in t and becomes bound in $\varphi(t)$. Because every formula is finite, only finitely many variable symbols occur in $\varphi(x)$. So there is some $k \in \mathbb{Z}^+$ such that $\mathtt{v}_{i,k}$ does not occur in $\varphi(x)$. Because $j = 1$ or $(\mathtt{v}_{i,1} = \mathtt{v}_{i,j}) \in \Gamma'$, and $k = 1$ or $(\mathtt{v}_{i,1} = \mathtt{v}_{i,k}) \in \Gamma'$, the replacement of $\mathtt{v}_{i,j}$ by $\mathtt{v}_{i,k}$ in t results in a term that seems intuitively equivalent to t. Based on this idea, we will be able to work around the "not free for x" problem in the sequel.

The variable symbols $\mathtt{v}_{2i-1,1}$ are almost but not entirely unused, because they occur in the formulas $(\mathtt{v}_{2i-1,1} = \mathtt{v}_{2i-1,j})$ in Γ'. To discuss this, we introduce new concepts. A *duplicate* of $\mathtt{v}_{i,1}$ is any $\mathtt{v}_{i,j}$ with $j > 1$. Let Υ be any superset of Γ'. By $\mathtt{v}_{i,1}$ is *vacant* in Υ we mean that for $j \in \mathbb{Z}^+$, every occurrence of $\mathtt{v}_{i,j}$ in Υ is in a formula of the form $(\mathtt{v}_{i,1} = \mathtt{v}_{i,j})$. All $v_{2i-1,1}$ are vacant in Γ', but only those $v_{2i,1}$ are vacant in Γ' where v_i does not occur in Γ.

Lemma 7.1. *Let Υ be a set of formulas such that $\Gamma' \subseteq \Upsilon$.*

1. *Every variable symbol x has infinitely many variable symbols y such that $\Gamma' \vdash (x = y)$.*

2. *If $\Upsilon \setminus \Gamma'$ is finite, then infinitely many variable symbols are vacant in Υ.*

3. *If $i \in \mathbb{Z}^+$, $\Upsilon \vdash \mathsf{F}$, and for $j > 1$, no $\mathrm{v}_{i,j}$ occurs in Υ except in $(\mathrm{v}_{i,1} = \mathrm{v}_{i,j})$, then Υ has a finite subset Δ such that $\Delta \vdash \mathsf{F}$ and no $\mathrm{v}_{i,j}$ with $j > 1$ occurs in Δ.*

4. *If $\Gamma' \vdash \mathsf{F}$, then $\Gamma \vdash \mathsf{F}$.*

5. *If $(\lceil\mathscr{F}\rfloor, \Gamma')$ has a model (Definition 5.2(2)), then $(\lceil\mathscr{F}\rfloor, \Gamma)$ has a model.*

Proof.

1. The claim follows from the $(\mathrm{v}_{i,1} = \mathrm{v}_{i,j})$ by =-4 and =-5.

2. All the $v_{2i-1,1}$ are vacant in Γ'. Each additional formula in Υ can make only a finite number of them non-vacant, because formulas are finite.

3. Because every proof is finite, the proof $\Upsilon \vdash \mathsf{F}$ only uses a finite number of elements of Υ. There is thus a finite $\Upsilon' \subseteq \Upsilon$ such that $\Upsilon' \vdash \mathsf{F}$. Because it is finite, it has a minimal subset Δ such that $\Delta \vdash \mathsf{F}$. It remains to be proven that no $\mathrm{v}_{i,j}$ with $j > 1$ occurs in Δ. We prove it by deriving a contradiction from the assumption that Δ contains a formula of the form $(\mathrm{v}_{i,1} = \mathrm{v}_{i,j})$ where $j > 1$.

   Let $\Delta' = \Delta \setminus \{(\mathrm{v}_{i,1} = \mathrm{v}_{i,j})\}$. By definition, $\Delta' \cup \{(\mathrm{v}_{i,1} = \mathrm{v}_{i,j})\} = \Delta \vdash \mathsf{F}$. Because $\emptyset \vdash_{\text{C6; }\wedge\text{-I}} \mathsf{T} \wedge \mathsf{T}$, we have $\Delta' \cup \{\neg\lceil \mathrm{v}_{i,1} = \mathrm{v}_{i,j}\rfloor\} \vdash_{\text{P1}} \neg\lceil \mathrm{v}_{i,1} = \mathrm{v}_{i,j}\rfloor \cong_{5.4(4,1)} \neg(\mathsf{T} \wedge \mathsf{T}) \vdash_{\text{C3}} \mathsf{F}$. By Lemma 6.7 we have $\Delta' \vdash \neg(\mathrm{v}_{i,1} = \mathrm{v}_{i,j})$. By construction, $\mathrm{v}_{i,j}$ does not occur in Δ'. Therefore, $\Delta' \vdash_{\forall\text{-I}} \forall x\, \neg(\mathrm{v}_{i,1} = x) \vdash_{\forall\text{-E}} \neg(\mathrm{v}_{i,1} = \mathrm{v}_{i,1}) \vdash_{=\text{-1; C3}} \mathsf{F}$, where $\lceil \mathrm{v}_{i,1}\rfloor \cong \mathsf{T}$ by 5.4(1). This contradicts the minimality of Δ.

4. By (3), there is $\Delta \subseteq \Gamma'$ such that $\Delta \vdash \mathsf{F}$ and no variable symbol of the form $\mathrm{v}_{i,j}$, where $i \geq 1$ and $j > 1$, occurs in Δ. It implies that Δ does not contain any $\mathrm{v}_{2i-1,1}$ either. Because every proof is finite, $\Delta \vdash \mathsf{F}$ only uses a finite number of variable symbols. So all occurrences of variable symbols of other forms than $\mathrm{v}_{2i,1}$ in $\Delta \vdash \mathsf{F}$ can be replaced by so far unused variable symbols of the form $\mathrm{v}_{2i,1}$, resulting in a proof of F from Δ that only uses variable symbols of the form $\mathrm{v}_{2i,1}$. Now replacing each $\mathrm{v}_{2i,1}$ by v_i results in a proof of F from Γ.

5. If $(\sigma, \nu') \models \Gamma'$, then $(\sigma, \nu) \models \Gamma$, where for $i \in \mathbb{Z}^+$ we have $\nu(i) = \nu'(\iota(2i, 1))$.

□

It is our goal to prove that if $\Gamma \not\vdash \mathsf{F}$, then $(\lceil\mathscr{F}\rfloor, \Gamma)$ has a model. It follows from Lemma 7.1(4) and (5) that it suffices to prove that if $\Gamma' \not\vdash \mathsf{F}$, then $(\lceil\mathscr{F}\rfloor, \Gamma')$ has a model. In that proof, we may assume what Lemma 7.1(1), (2), and (3) state.

Next we extend Γ' so that the extended set Γ_ω is consistent and for each formula φ, precisely one of φ, $\neg\varphi$ and $\neg\lceil\varphi\rfloor$ is in Γ_ω. Furthermore, for each formula of the form $\exists x\, \psi(x)$ in Γ_ω it contains a formula of the form $\psi(y)$ as well, so that later in this section we can appeal to $\psi(y)$ to justify $\exists x\, \psi(x)$. Similarly every $\neg\forall x\, \psi(x)$

in Γ_ω is accompanied by $\neg\psi(y)$ for some y. The y are called *Henkin witnesses*. Technically they are free variables. However, our eventual goal is to build a model (σ, ν) for Γ', and in it each free variable v_i will have a single value $\nu(i)$. So the Henkin witnesses will eventually represent constant values.

The set Γ_ω is built by processing all formulas φ (also those that are in Γ') in some order $\varphi_1, \varphi_2, \ldots$ and forming a sequence $\Gamma_0, \Gamma_1, \Gamma_2, \ldots$ of sets of formulas as follows. Let $\Gamma_0 := \Gamma'$. For $i > 0$, we construct Γ_i from Γ_{i-1} and φ_i according to the first item in the list below whose condition is satisfied by φ_i (we will later show that at least one item matches). Please notice that each $\Gamma_i \setminus \Gamma_{i-1}$ contains at most two formulas, and thus each $\Gamma_i \setminus \Gamma'$ is finite. (Conditions of Cases 4 and 5 mention facts that could be derived from the failure of earlier conditions. This is to simplify subsequent discussion.)

1. If $\Gamma_{i-1} \cup \{\lceil\varphi_i\rfloor\} \vdash \mathsf{F}$, then $\Gamma_i := \Gamma_{i-1} \cup \{\neg\lceil\varphi_i\rfloor\}$.

2. If $\Gamma_{i-1} \cup \{\varphi_i\} \nvdash \mathsf{F}$ and φ_i is of the form $\exists x\ \psi(x)$, then $\Gamma_i := \Gamma_{i-1} \cup \{\varphi_i, \psi(y)\}$, where y is a variable symbol that is vacant in Γ_{i-1} and does not occur nor its duplicates occur in φ_i. By Lemma 7.1(2) such an y exists.

3. If $\Gamma_{i-1} \cup \{\neg\varphi_i\} \nvdash \mathsf{F}$ and φ_i is of the form $\forall x\ \psi(x)$, then $\Gamma_i := \Gamma_{i-1} \cup \{\neg\varphi_i, \neg\psi(y)\}$, where y is a variable symbol that is vacant in Γ_{i-1} and does not occur nor its duplicates occur in φ_i nor $\lceil\psi(x)\rfloor$. By Lemma 7.1(2) such an y exists.

4. If $\Gamma_{i-1} \cup \{\varphi_i\} \nvdash \mathsf{F}$ and φ_i is not of the form $\exists x\ \psi(x)$, then $\Gamma_i := \Gamma_{i-1} \cup \{\varphi_i\}$.

5. If $\Gamma_{i-1} \cup \{\neg\varphi_i\} \nvdash \mathsf{F}$ and φ_i is not of the form $\forall x\ \psi(x)$, then $\Gamma_i := \Gamma_{i-1} \cup \{\neg\varphi_i\}$.

Clearly $\Gamma' = \Gamma_0 \subseteq \Gamma_1 \subseteq \Gamma_2 \subseteq \cdots$. We choose $\Gamma_\omega := \Gamma_0 \cup \Gamma_1 \cup \ldots$.

Lemma 7.2. *If* $\Gamma' \nvdash \mathsf{F}$, *then* $\Gamma_\omega \nvdash \mathsf{F}$.

Proof. We prove first by induction that each Γ_i is consistent. The assumption $\Gamma' \nvdash \mathsf{F}$ gives the base case $\Gamma_0 \nvdash \mathsf{F}$. The induction assumption is that $\Gamma_{i-1} \nvdash \mathsf{F}$. The induction step is divided into five cases according to how Γ_i is formed. In Cases 4 and 5 $\Gamma_i \nvdash \mathsf{F}$ by the condition of the case. In the remaining cases we assume $\Gamma_i \vdash \mathsf{F}$ and derive a contradiction.

In Case 1 $\Gamma_{i-1} \cup \{\lceil\varphi_i\rfloor\} \vdash \mathsf{F}$. If $\Gamma_i \vdash \mathsf{F}$, then $\Gamma_{i-1} \cup \{\neg\lceil\varphi_i\rfloor\} \vdash \mathsf{F}$. These yield $\Gamma_{i-1} \vdash_{\text{D2}} \lceil\varphi_i\rfloor \vee \neg\lceil\varphi_i\rfloor \vdash_{\vee\text{-E}} \mathsf{F}$, contradicting the induction assumption.

If in Case 2 $\Gamma_i \vdash \mathsf{F}$, then $\Gamma_{i-1} \cup \{(\exists x\ \psi(x)), \psi(y)\} \vdash \mathsf{F}$. Because y does not occur in $\exists x\ \psi(x)$, it is free for x in $\psi(x)$. Therefore, $\{\psi(y)\} \vdash_{\exists\text{-I}} \exists x\ \psi(x)$. Hence $\Gamma_{i-1} \cup \{\psi(y)\} \vdash_{\text{P3}} \mathsf{F}$. By 7.1(3) there is a finite subset Δ of Γ_{i-1} such that $\Delta \cup \{\psi(y)\} \vdash \mathsf{F}$ and y does not occur in Δ. Thus the assumptions of $\exists$-E hold such that Γ, φ, and ψ are Δ, $\psi(y)$, and F, respectively. We obtain $\Delta \cup \{\exists x\ \psi(x)\} \vdash_{\exists\text{-E}} \mathsf{F}$. So $\Gamma_{i-1} \cup \{\exists x\ \psi(x)\} \vdash_{\text{P2}} \mathsf{F}$, contradicting the condition of the case.

In Case 3 we denote by Γ'_i the set $\Gamma_{i-1} \cup \{\neg\forall x\ \psi(x)\}$. So $\Gamma_i = \Gamma'_i \cup \{\neg\psi(y)\}$. If $\Gamma_i \vdash \mathsf{F}$, then $\Gamma'_i \cup \{\neg\psi(y)\} \vdash \mathsf{F}$. Again, by 7.1(3) there is a finite subset Δ of Γ'_i such that $\Delta \cup \{\neg\psi(y)\} \vdash \mathsf{F}$ and y does not occur in Δ. Because y does not occur in $\neg\forall x\ \psi(x)$, we may assume $(\neg\forall x\ \psi(x)) \in \Delta$. Lemma 6.7 yields

$$\Delta \vdash \psi(y) \vee \neg\lceil\psi(y)\rfloor. \tag{2}$$

On the other hand, $\Delta \vdash_{\mathrm{D1}} \lceil\neg\forall x\ \psi(x)\rfloor \cong_{5.4(6,9)} (\forall x\ \lceil\psi(x)\rfloor) \vee \exists x\ (\lceil\psi(x)\rfloor \wedge \neg\psi(x))$. We show that both $\Delta \cup \{\forall x\ \lceil\psi(x)\rfloor\}$ and $\Delta \cup \{\exists x\ (\lceil\psi(x)\rfloor \wedge \neg\psi(x))\}$ prove F. They imply $\Delta \vdash_{\vee\text{-E}} \mathsf{F}$ and $\Gamma'_i \vdash_{\mathrm{P2}} \mathsf{F}$, which contradicts the condition of the case.

First consider $\Delta \cup \{\forall x\ \lceil\psi(x)\rfloor\}$. We have $\lceil y\rfloor \cong \mathsf{T}$ by 5.4(1). Because y does not occur in $\lceil\psi(x)\rfloor$, we get $\{\forall x\ \lceil\psi(x)\rfloor\} \vdash_{\forall\text{-E}} \lceil\psi(y)\rfloor$. Since $\{\lceil\psi(y)\rfloor\} \cup \{\psi(y)\} \vdash_{\mathrm{P1}} \psi(y)$ and $\{\lceil\psi(y)\rfloor\} \cup \{\neg\lceil\psi(y)\rfloor\} \vdash_{\mathrm{C3}} \mathsf{F} \vdash_{\mathrm{C2}} \psi(y)$, we obtain $\Delta \cup \{\forall x\ \lceil\psi(x)\rfloor\} \vdash_{(2);\,\vee\text{-E}} \psi(y) \vdash_{\forall\text{-I}} \forall y\ \psi(y)$, since y does not occur in Δ or $\lceil\psi(x)\rfloor$. Because $\psi(y)$ is obtained from $\psi(x)$ by substituting y for x, we have $\{\forall y\ \psi(y)\} \vdash_{\forall\text{-E}} \psi(x) \vdash_{\forall\text{-I}} \forall x\ \psi(x)$. Therefore, $\Delta \cup \{\forall x\ \lceil\psi(x)\rfloor\} \vdash \forall x\ \psi(x) \vdash_{\mathrm{C3}} \mathsf{F}$, because $(\neg\forall x\ \psi(x)) \in \Delta$.

To deal with $\Delta \cup \{\exists x\ (\lceil\psi(x)\rfloor \wedge \neg\psi(x))\}$, let z be a variable symbol that does not occur in it. We obtain $\Delta \vdash_{(2);\,\forall\text{-I}} \forall y\ (\psi(y) \vee \neg\lceil\psi(y)\rfloor) \vdash_{\forall\text{-E}} \psi(z) \vee \neg\lceil\psi(z)\rfloor$. Clearly $\{\lceil\psi(z)\rfloor \wedge \neg\psi(z)\} \cup \{\psi(z)\} \vdash \mathsf{F}$ and $\{\lceil\psi(z)\rfloor \wedge \neg\psi(z)\} \cup \{\neg\lceil\psi(z)\rfloor\} \vdash \mathsf{F}$, which implies $\Delta \cup \{\lceil\psi(z)\rfloor \wedge \neg\psi(z)\} \vdash_{\vee\text{-E}} \mathsf{F}$. Thus $\Delta \cup \{\exists x\ (\lceil\psi(x)\rfloor \wedge \neg\psi(x))\} \vdash_{\exists\text{-E}} \mathsf{F}$.

This completes the proof of $\Gamma_i \nvdash \mathsf{F}$ for all $i \in \mathbb{N}$.

We still have to prove that $\Gamma_\omega \nvdash \mathsf{F}$. If $\Gamma_\omega \vdash \mathsf{F}$, let Γ'_ω be the set of formulas in Γ_ω that occur in the proof of the contradiction. The set Γ'_ω is finite because all proofs are finite. Therefore, there is such an index $i \in \mathbb{N}$ that $\Gamma'_\omega \subseteq \Gamma_i$. By construction, $\Gamma'_\omega \vdash \mathsf{F}$. This and P2 imply $\Gamma_i \vdash \mathsf{F}$, which contradicts the earlier result. □

Lemma 7.3. *Assume $\Gamma' \nvdash \mathsf{F}$. For each formula φ, the set Γ_ω contains exactly one of the formulas φ, $\neg\varphi$, and $\neg\lceil\varphi\rfloor$.*

Proof. If, when φ_i is processed in the construction of Γ_ω, none of the conditions of Cases 2, 3, 4, and 5 holds, then by Lemma 6.7 $\Gamma_{i-1} \vdash \neg\lceil\varphi_i\rfloor$, and thus the condition of Case 1 holds. So Γ_ω contains, for each φ, at least one of $\neg\lceil\varphi\rfloor$, φ, and $\neg\varphi$. By Lemma 7.2, C3, C4, and C5, Γ_ω contains at most one of φ, $\neg\varphi$, and $\neg\lceil\varphi\rfloor$. □

Lemma 7.4. *Assume $\Gamma' \nvdash \mathsf{F}$. Then $\Gamma_\omega \vdash \varphi$ if and only if $\varphi \in \Gamma_\omega$.*

Proof. If $\varphi \in \Gamma_\omega$, then $\Gamma_\omega \vdash_{\mathrm{P1}} \varphi$.

Assume $\Gamma_\omega \vdash \varphi$. If $(\neg\varphi) \in \Gamma_\omega$, then $\Gamma_\omega \vdash_{\mathrm{C3}} \mathsf{F}$, which contradicts Lemma 7.2. Therefore, $(\neg\varphi) \notin \Gamma_\omega$. A similar reasoning using C4 yields $(\neg\lceil\varphi\rfloor) \notin \Gamma_\omega$. By Lemma 7.3 at least one of φ, $\neg\varphi$, $\neg\lceil\varphi\rfloor$ is in Γ_ω. So $\varphi \in \Gamma_\omega$. □

Lemma 7.5. *Assume $\Gamma' \nvdash \mathsf{F}$. For each formula φ, the set Γ_ω contains exactly one of $\lceil\varphi\rfloor$ and $\neg\lceil\varphi\rfloor$. For each term t, the set Γ_ω contains exactly one of $\lceil t\rfloor$ and $\neg\lceil t\rfloor$.*

Proof. If $\varphi \in \Gamma_\omega$ or $(\neg\varphi) \in \Gamma_\omega$, then $\Gamma_\omega \vdash_{\mathrm{D1}} \lceil\varphi\rfloor \cong \lceil\neg\varphi\rfloor$. By Lemma 7.4 we have $\lceil\varphi\rfloor \in \Gamma_\omega$. Otherwise, Lemma 7.3 yields $(\neg\lceil\varphi\rfloor) \in \Gamma_\omega$. By C3 we have $\{\lceil\varphi\rfloor, (\neg\lceil\varphi\rfloor)\} \not\subseteq \Gamma_\omega$, completing the proof of the first claim.

If $\lceil t = t\rfloor \in \Gamma_\omega$, then $\lceil t\rfloor \in \Gamma_\omega$, because $\lceil t = t\rfloor \cong_{5.4(4)} \lceil t\rfloor \wedge \lceil t\rfloor$. Otherwise, by the first claim $(\neg\lceil t = t\rfloor) \in \Gamma_\omega$. We use Lemma 6.7 to show $\{\neg\lceil t = t\rfloor\} \vdash \neg\lceil t\rfloor$. Clearly $\neg\lceil t = t\rfloor \cong \neg(\lceil t\rfloor \wedge \lceil t\rfloor)$, so $\{\neg\lceil t = t\rfloor\} \cup \{\lceil t\rfloor\} \vdash_{\wedge\text{-I}} \lceil t\rfloor \wedge \lceil t\rfloor \vdash_{\mathrm{C3}} \mathsf{F}$. Furthermore, $\{\neg\lceil t = t\rfloor\} \vdash_{\mathrm{D1}} \lceil\neg\lceil t = t\rfloor\rfloor \cong \lceil\lceil t = t\rfloor\rfloor \cong \lceil\lceil t\rfloor \wedge \lceil t\rfloor\rfloor \cong_{5.4(7)} (\lceil\lceil t\rfloor\rfloor \wedge \lceil\lceil t\rfloor\rfloor) \vee (\lceil\lceil t\rfloor\rfloor \wedge \neg\lceil t\rfloor) \vee (\lceil\lceil t\rfloor\rfloor \wedge \neg\lceil t\rfloor)$, so $\{\neg\lceil t = t\rfloor\} \cup \{\neg\lceil\lceil t\rfloor\rfloor\} \vdash_{\vee\text{-E}} \mathsf{F}$. $\square$

By Lemma 7.3, we may unambiguously assign each formula χ a truth value $\overline{\chi}$ as follows:

$$\begin{aligned} \overline{\chi} &\asymp \mathsf{T}, \text{ if } \chi \in \Gamma_\omega \\ \overline{\chi} &\asymp \mathsf{F}, \text{ if } \neg\chi \in \Gamma_\omega \\ \overline{\chi} &\asymp \mathsf{U}, \text{ if } \neg\lceil\chi\rfloor \in \Gamma_\omega \end{aligned} \tag{3}$$

We show next that these truth values respect the propositional constants and connectives.

Lemma 7.6. *If* $\Gamma' \nvdash \mathsf{F}$, *then* $\overline{\mathsf{F}} \asymp \mathsf{F}$, $\overline{\mathsf{T}} \asymp \mathsf{T}$, $\overline{\neg\varphi} \asymp \neg\overline{\varphi}$, $\overline{\varphi \wedge \psi} \asymp \overline{\varphi} \wedge \overline{\psi}$, *and* $\overline{\varphi \vee \psi} \asymp \overline{\varphi} \vee \overline{\psi}$.

Proof.

T, F By C6 and Lemma 7.4, $\mathsf{T} \in \Gamma_\omega$, so $\overline{\mathsf{T}} \asymp \mathsf{T}$. Because $\neg\lceil\mathsf{F}\rfloor \cong \neg\mathsf{T}$ by Definition 5.4(3), we cannot have $(\neg\lceil\mathsf{F}\rfloor) \in \Gamma_\omega$. Lemmas 7.2 and 7.4 rule out $\mathsf{F} \in \Gamma_\omega$. The only remaining possibility is $(\neg\mathsf{F}) \in \Gamma_\omega$, so $\overline{\mathsf{F}} \asymp \mathsf{F}$.

$\neg\varphi$ Let $\chi \cong \neg\varphi$. If $\overline{\varphi} \asymp \mathsf{F}$, then $(\neg\varphi) \in \Gamma_\omega$. That is, $\chi \in \Gamma_\omega$, so $\overline{\chi} \asymp \mathsf{T}$. If $\overline{\varphi} \asymp \mathsf{U}$, then $(\neg\lceil\varphi\rfloor) \in \Gamma_\omega$. By Definition 5.4(6) $\lceil\chi\rfloor \cong \lceil\varphi\rfloor$. So $(\neg\lceil\chi\rfloor) \in \Gamma_\omega$ and $\overline{\chi} \asymp \mathsf{U}$. Finally, let $\overline{\varphi} \asymp \mathsf{T}$, that is, $\varphi \in \Gamma_\omega$. If $\overline{\chi} \asymp \mathsf{T}$, then $\{\varphi, \neg\varphi\} \subseteq \Gamma_\omega \vdash_{\mathrm{C3}} \mathsf{F}$. If $\overline{\chi} \asymp \mathsf{U}$, then $\{\varphi, \neg\lceil\neg\varphi\rfloor\} = \{\varphi, \neg\lceil\varphi\rfloor\} \subseteq \Gamma_\omega \vdash_{\mathrm{C4}} \mathsf{F}$. Thus $\overline{\chi} \asymp \mathsf{F}$.

$\varphi \wedge \psi$ Let $\chi \cong \varphi \wedge \psi$. If $\overline{\varphi} \asymp \mathsf{T}$ and $\overline{\psi} \asymp \mathsf{T}$, then $\{\varphi, \psi\} \subseteq \Gamma_\omega \vdash_{\wedge\text{-I}} \varphi \wedge \psi$. By Lemma 7.4 $\overline{\varphi \wedge \psi} \asymp \mathsf{T}$.

By Definition 5.4(7) $\lceil\varphi \wedge \psi\rfloor \cong (\lceil\varphi\rfloor \wedge \lceil\psi\rfloor) \vee (\lceil\varphi\rfloor \wedge \neg\varphi) \vee (\lceil\psi\rfloor \wedge \neg\psi)$. If $\overline{\varphi} \asymp \mathsf{U}$ and $\overline{\psi} \asymp \mathsf{T}$, then $\{(\neg\lceil\varphi\rfloor), \psi\} \subseteq \Gamma_\omega$ and $\Gamma_\omega \cup \{\lceil\varphi \wedge \psi\rfloor\} \vdash_{\wedge\text{-E1};\, \wedge\text{-E2};\, \mathrm{C3};\, \vee\text{-E}} \mathsf{F}$. So by Lemma 7.5 $(\neg\lceil\varphi \wedge \psi\rfloor) \in \Gamma_\omega$, that is, $\overline{\varphi \wedge \psi} \asymp \mathsf{U}$. Similar reasoning applies to $\overline{\varphi} \asymp \mathsf{T}$ and $\overline{\psi} \asymp \mathsf{U}$, and to $\overline{\varphi} \asymp \mathsf{U}$ and $\overline{\psi} \asymp \mathsf{U}$.

If $\overline{\varphi} \asymp \mathsf{F}$ then $\neg\varphi \in \Gamma_\omega \vdash_{\mathrm{D1}} \lceil\neg\varphi\rfloor \cong \lceil\varphi\rfloor \vdash_{\wedge\text{-I}} \lceil\varphi\rfloor \wedge \neg\varphi \vdash_{\vee\text{-I2};\, \vee\text{-I1}} \lceil\varphi \wedge \psi\rfloor$, ruling out $\overline{\varphi \wedge \psi} \asymp \mathsf{U}$. Also $\overline{\varphi \wedge \psi} \asymp \mathsf{T}$ is impossible, because $\{\varphi \wedge \psi\} \vdash_{\wedge\text{-E1}} \varphi$. So $\overline{\varphi \wedge \psi} \asymp \mathsf{F}$. Similarly, if $\overline{\psi} \asymp \mathsf{F}$, then $\overline{\varphi \wedge \psi} \asymp \mathsf{F}$.

$\varphi \vee \psi$ Let $\chi \cong \varphi \vee \psi$. If $\overline{\varphi} \asymp \mathsf{T}$, then $\varphi \in \Gamma_\omega \vdash_{\vee\text{-I1}} \varphi \vee \psi$, so $\overline{\varphi \vee \psi} \asymp \mathsf{T}$. Similarly if $\overline{\psi} \asymp \mathsf{T}$, then $\overline{\varphi \vee \psi} \asymp \mathsf{T}$.

By Definition 5.4(8) $\lceil \varphi \vee \psi \rfloor \cong (\lceil \varphi \rfloor \wedge \lceil \psi \rfloor) \vee (\lceil \varphi \rfloor \wedge \varphi) \vee (\lceil \psi \rfloor \wedge \psi)$. Like above, it yields a contradiction with any combination of $\overline{\varphi}$ and $\overline{\psi}$, where one of them is U and the other is U or F. Therefore, these combinations make $\overline{\varphi \vee \psi} \asymp \mathsf{U}$.

If $\overline{\varphi} \asymp \mathsf{F}$ and $\overline{\psi} \asymp \mathsf{F}$, then $(\neg\varphi) \in \Gamma_\omega \vdash_{\text{D1}} \lceil \varphi \rfloor$ and $(\neg\psi) \in \Gamma_\omega \vdash_{\text{D1}} \lceil \psi \rfloor$, so $\Gamma_\omega \vdash_{\wedge\text{-I}} \lceil \varphi \rfloor \wedge \lceil \psi \rfloor \vdash_{\vee\text{-I1};\,\vee\text{-I1}} \lceil \varphi \vee \psi \rfloor$. This rules out $\overline{\varphi \vee \psi} \asymp \mathsf{U}$. Clearly $\Gamma_\omega \cup \{\varphi\} \vdash_{\text{C3}} \mathsf{F}$ and $\Gamma_\omega \cup \{\psi\} \vdash_{\text{C3}} \mathsf{F}$, so $\Gamma_\omega \cup \{\varphi \vee \psi\} \vdash_{\vee\text{-E}} \mathsf{F}$, ruling out $\overline{\varphi \vee \psi} \asymp \mathsf{T}$. Therefore, $\overline{\varphi \vee \psi} \asymp \mathsf{F}$.

□

Assuming that Γ' is consistent, we next build a model for Γ_ω. By Definition 5.2(2), it has to obey Definition 3.5 and 5.2(1). The former will be proven now, and the latter as Lemma 7.8.

Lemma 7.7. *If $\Gamma' \nvdash \mathsf{F}$, then there are $\sigma = (\mathbb{D}, \_)$ and ν such that $(\sigma, \nu) \models \Gamma_\omega$.*

Proof. Let us define the set of *defined terms* as $\mathscr{T}_{\mathsf{def}} = \{t \mid \lceil t \rfloor \in \Gamma_\omega\}$. If $t \in \mathscr{T}_{\mathsf{def}}$ and if $(t = t') \in \Gamma_\omega$ or $(t' = t) \in \Gamma_\omega$, then also $t' \in \mathscr{T}_{\mathsf{def}}$, because $\{t_1 = t_2\} \vdash_{\text{D1}} \lceil t_1 = t_2 \rfloor \cong \lceil t_1 \rfloor \wedge \lceil t_2 \rfloor$ by Definition 5.4(4). By Lemma 7.4, =-1, =-4, and =-5, the relation $\{(t_1, t_2) \mid (t_1 = t_2) \in \Gamma_\omega\}$ is an equivalence on $\mathscr{T}_{\mathsf{def}}$. For each $t \in \mathscr{T}_{\mathsf{def}}$, let $\overline{t}$ be the equivalence class that t belongs to. That is, $\overline{t} = \{t' \mid (t = t') \in \Gamma_\omega\}$. We let $\mathbb{D}$ be the set of these equivalence classes, that is, $\mathbb{D} = \{\overline{t} \mid \lceil t \rfloor \in \Gamma_\omega\}$. It is non-empty, because each variable symbol is a defined term. If $t \notin \mathscr{T}_{\mathsf{def}}$, we leave $\overline{t}$ undefined. To summarize:

- $\lceil t \rfloor \in \Gamma_\omega$ if and only if $\overline{t} \in \mathbb{D}$; furthermore, every element of $\mathbb{D}$ is of this form.
- $\lceil t \rfloor \notin \Gamma_\omega$ if and only if $\overline{t}$ is undefined.
- $(t = t') \in \Gamma_\omega$ if and only if $\overline{t} = \overline{t'}$; furthermore, then both $\overline{t}$ and $\overline{t'}$ are defined.

We need to define _ and ν so that $(\sigma, \nu) \models \Gamma_\omega$ (Definition 3.5) and $\sigma \models \lceil \mathscr{F} \rfloor$ (Definition 5.2(1)). We will choose them so that we can prove by induction that

$$\underline{t}(\nu) = \overline{t} \text{ for each } t \in \mathscr{T}_{\mathsf{def}}, \text{ and} \tag{4}$$

$$\underline{t}(\nu) \text{ is undefined for each } t \notin \mathscr{T}_{\mathsf{def}}. \tag{5}$$

By Definition 5.4(1) and C6, variable and constant symbols are defined. For each $n \in \mathbb{Z}^+$ we let $\nu(n) = \overline{\mathsf{v}_n}$. Then $\underline{\mathsf{v}_n}(\nu) = \overline{\mathsf{v}_n}$ by 3.1(1). To make 3.1(2) yield $\underline{c}(\nu) = \overline{c}$, we choose $\underline{C} = \overline{c}$. This is the base case of the induction proof.

Our next task is, for each function symbol f, to define $\underline{f}$ so that it makes $\underline{t}(\nu)$ be undefined or $\overline{t}$ as appropriate. Let $\overline{t_1} \in \mathbb{D}, \ldots, \overline{t_{\alpha(f)}} \in \mathbb{D}$. We choose

$$\underline{f}(\overline{t_1}, \ldots, \overline{t_{\alpha(f)}}) \begin{cases} = \overline{f(t_1, \ldots, t_{\alpha(f)})} & \text{if } \lceil f(t_1, \ldots, t_{\alpha(f)}) \rfloor \in \Gamma_\omega \\ \text{is undefined} & \text{otherwise.} \end{cases}$$

To show that this definition does not depend on the choice of the $t_i \in \overline{t_i}$, choose any $1 \leq i \leq \alpha(f)$. Let $t_i' \in \overline{t_i}$, that is, $(t_i = t_i') \in \Gamma_\omega$. We leave terms out for brevity; for instance, $f(t_i)$ means $f(t_1,\ldots,t_{\alpha(f)})$. If $\lceil f(t_i) \rfloor \in \Gamma_\omega$, then =-3 yields $(f(t_i) = f(t_i')) \in \Gamma_\omega$. So $\overline{f(t_i)} = \overline{f(t_i')}$. If $\lceil f(t_i) \rfloor \notin \Gamma_\omega$ then also $\lceil f(t_i') \rfloor \notin \Gamma_\omega$, because otherwise =-3 yields $(f(t_i') = f(t_i)) \in \Gamma_\omega$, implying $\lceil f(t_i) \rfloor \in \Gamma_\omega$.

We now complete the induction proof regarding $\underline{t}(\nu)$ and $\overline{t}$. There are three cases.

- Assume that $\overline{t_i}$ is defined for every $1 \leq i \leq \alpha(f)$, and $\lceil f(t_1,\ldots,t_{\alpha(f)}) \rfloor \in \Gamma_\omega$. By Definition 3.1(3), the induction assumption, and the definition of $\underline{f}$ we have $\underline{f(t_1,\ldots,t_{\alpha(f)})}(\nu) = \underline{f}(\underline{t_1}(\nu),\ldots,\underline{t_{\alpha(f)}}(\nu)) = \underline{f}(\overline{t_1},\ldots,\overline{t_{\alpha(f)}}) = \overline{f(t_1,\ldots,t_{\alpha(f)})}$.

- Assume that $\overline{t_i}$ is defined for every $1 \leq i \leq \alpha(f)$, but $\lceil f(t_1,\ldots,t_{\alpha(f)}) \rfloor \notin \Gamma_\omega$. By the definition of $\overline{t}$, $\overline{f(t_1,\ldots,t_{\alpha(f)})}$ is undefined. On the other hand, $\underline{f}(\overline{t_1},\ldots,\overline{t_{\alpha(f)}})$ is undefined by the definition of $\underline{f}$. By the induction assumption, $\overline{t_i} = \underline{t_i}(\nu)$ for $1 \leq i \leq \alpha(f)$, so $\underline{f}(\underline{t_1}(\nu),\ldots,\underline{t_{\alpha(f)}}(\nu))$ is undefined. By 3.1(3) $\underline{f(t_1,\ldots,t_{\alpha(f)})}(\nu)$ is undefined.

- Assume that $\overline{t_i}$ is undefined for some $1 \leq i \leq \alpha(f)$. We have $\lceil t_i \rfloor \notin \Gamma_\omega$, from which $\lceil f(t_1,\ldots,t_{\alpha(f)}) \rfloor \notin \Gamma_\omega$ by 5.4(2), $\wedge$-E1, and $\wedge$-E2. By the definition of $\overline{t}$, $\overline{f(t_1,\ldots,t_{\alpha(f)})}$ is undefined. On the other hand, by the induction assumption $\underline{t_i}(\nu)$ is undefined, so $\underline{f(t_1,\ldots,t_{\alpha(f)})}(\nu)$ is undefined by 3.1(3).

This completes the proof of (4) and (5).

In (3) we defined that if φ is a formula, then $\overline{\varphi}$ yields T, F, or U according to which one of φ, $\neg\varphi$, and $\neg\lceil\varphi\rfloor$ is in Γ_ω. We will show by induction that

$$\underline{\varphi}(\nu) \asymp \overline{\varphi} \text{ for every } \varphi.$$

This will imply that if $\varphi \in \Gamma_\omega$, then $\underline{\varphi}(\nu) \asymp \overline{\varphi} \asymp \mathsf{T}$. As a consequence, $(\sigma,\nu) \models \Gamma_\omega$.

The base case of the induction consists of the atomic formulas. Definition 3.2(1) and Lemma 7.6 tell that $\underline{\mathsf{F}}(\nu) \asymp \mathsf{F} \asymp \overline{\mathsf{F}}$ and $\underline{\mathsf{T}}(\nu) \asymp \mathsf{T} \asymp \overline{\mathsf{T}}$. Next we show $\underline{(t_1 = t_2)}(\nu) \asymp \overline{t_1 = t_2}$.

- If $\overline{t_1}$ is undefined, then $\lceil t_1 \rfloor \notin \Gamma_\omega$. We have $\lceil t_1 = t_2 \rfloor \cong (\lceil t_1 \rfloor \wedge \lceil t_2 \rfloor) \notin \Gamma_\omega$ by Definition 5.4(4) and $\wedge$-E1. By Lemma 7.5, $(\neg\lceil t_1 = t_2 \rfloor) \in \Gamma_\omega$, that is, $\overline{(t_1 = t_2)} \asymp \mathsf{U}$. On the other hand, because $\overline{t_1}$ is undefined also $\underline{t_1}(\nu)$ is undefined by (5), so $\underline{(t_1 = t_2)}(\nu) \asymp \mathsf{U}$ by 3.2(2). The same argument applies if $\overline{t_2}$ is undefined.

- If $\overline{t_1}$ and $\overline{t_2}$ are defined, then $\underline{t_1}(\nu) = \overline{t_1}$ and $\underline{t_2}(\nu) = \overline{t_2}$ by (4).

  If $\overline{t_1} = \overline{t_2}$, then $(t_1 = t_2) \in \Gamma_\omega$, that is, $\overline{(t_1 = t_2)} \asymp \mathsf{T}$. On the other hand, $\underline{t_1}(\nu) = \underline{t_2}(\nu)$, so by 3.2(2) we have $\underline{(t_1 = t_2)}(\nu) \asymp \mathsf{T}$.

If $\overline{t_1} \neq \overline{t_2}$, then $(t_1 = t_2) \notin \Gamma_\omega$. Furthermore, $\lceil t_1 \rfloor \in \Gamma_\omega$ and $\lceil t_2 \rfloor \in \Gamma_\omega$. By $\wedge$-I we have $\lceil t_1 = t_2 \rfloor \in \Gamma_\omega$, implying $(\neg \lceil t_1 = t_2 \rfloor) \notin \Gamma_\omega$. By Lemma 7.3 we have $(\neg(t_1 = t_2)) \in \Gamma_\omega$, that is, $\overline{(t_1 = t_2)} \asymp \mathsf{F}$. On the other hand, $\underline{t_1}(\nu) \neq \underline{t_2}(\nu)$, so by 3.2(2) we have $\underline{(t_1 = t_2)}(\nu) \asymp \mathsf{F}$.

By Definition 5.4(5), $\overline{R(t_1,\ldots,t_{\alpha(R)})} \asymp \mathsf{U}$ (that is, $(\neg \lceil R(t_1,\ldots,t_{\alpha(R)}) \rfloor) \in \Gamma_\omega$) precisely when at least one of the $\overline{t_i}$ is undefined. This is in harmony with 3.2(3), according to which $\underline{R(t_1,\ldots,t_{\alpha(R)})}(\nu) \asymp \mathsf{U}$ if and only if at least one $\underline{t_i}(\nu)$ is undefined. When all the $\overline{t_i}$ are defined, we define $\underline{R} = \{(\overline{t_1},\ldots,\overline{t_{\alpha(R)}}) \mid (R(t_1,\ldots,t_{\alpha(R)})) \in \Gamma_\omega\}$. This makes $\overline{R(t_1,\ldots,t_{\alpha(R)})}$ yield T and F when it should by 3.2(3). By =-2, whether or not $(R(t_1,\ldots,t_{\alpha(R)})) \in \Gamma_\omega$, is independent of the choice of the representatives t_1, …, $t_{\alpha(R)}$ of the equivalence classes. Because $R(x_1,\ldots,x_{\alpha(R)})$ is an atomic formula, it contains no bound variables, so the "free for x_i" condition in =-2 is satisfied. We have shown $\underline{R(t_1,\ldots,t_{\alpha(R)})}(\nu) \asymp \overline{R(t_1,\ldots,t_{\alpha(R)})}$.

The base case of the induction proof is now complete. The induction assumption is that subformulas obey $\underline{\varphi}(\nu) \asymp \overline{\varphi}$. Our proof of the induction step follows a pattern that we now illustrate with $\wedge$. We have $\underline{\varphi \wedge \psi}(\nu) \asymp \underline{\varphi}(\nu) \wedge \underline{\psi}(\nu) \asymp \overline{\varphi} \wedge \overline{\psi} \asymp \overline{\varphi \wedge \psi}$ by Definition 3.2(5), the induction assumption, and Lemma 7.6. The cases $\neg$ and $\vee$ follow similarly from 3.2(4) and (6), respectively.

We still have to deal with the quantifiers. By the induction assumption, (4), and Lemma 3.6 we have the following:

$$\text{If } \overline{t} \in \mathbb{D} \text{ and } t \text{ is free for } x \text{ in } \psi(x)\text{, then } \overline{\psi(t)} \asymp \underline{\psi(t)}(\nu) \asymp \underline{\psi(x)}(\nu[x := \overline{t}]). \qquad (6)$$

If $\overline{\forall x\ \psi(x)} \asymp \mathsf{T}$, then $(\forall x\ \psi(x)) \in \Gamma_\omega$. Assume $\overline{t}$ is an arbitrary element of $\mathbb{D}$. Then $\lceil t \rfloor \in \Gamma_\omega$. By Lemma 7.1(1), $t \in \overline{t}$ can be chosen so that it is free for x in $\psi(x)$. Then $\{\forall x\ \psi(x)\} \vdash_{\forall\text{-E}} \psi(t)$. Therefore, $\psi(t) \in \Gamma_\omega$, that is, $\overline{\psi(t)} \asymp \mathsf{T}$. So $\underline{\psi(x)}(\nu[x := \overline{t}]) \asymp \mathsf{T}$ by (6). By Definition 3.2(7) $\underline{\forall x\ \psi(x)}(\nu) \asymp \mathsf{T}$.

If $\overline{\forall x\ \psi(x)} \asymp \mathsf{F}$, then $(\neg \forall x\ \psi(x)) \in \Gamma_\omega$. When $\forall x\ \psi(x)$ was dealt with in the construction of Γ_ω, Case 1 was not chosen, because otherwise we would have $\{(\neg \forall x\ \psi(x)), (\neg \lceil \forall x\ \psi(x) \rfloor)\} \subseteq \Gamma_\omega \vdash_{\text{C5}} \mathsf{F}$. Case 2 was not chosen since $\forall x\ \psi(x)$ is of wrong form for it. The condition of Case 3 was satisfied, as otherwise Γ_ω would be inconsistent. In Case 3, the formula $\neg \psi(y)$ was added to Γ_ω, making $\overline{\psi(y)} \asymp \mathsf{F}$. By 5.4(1) $y \in \mathscr{T}_{\mathsf{def}}$, so by (6) $\underline{\psi(x)}(\nu[x := \overline{y}]) \asymp \mathsf{F}$. By Definition 3.2(7) $\underline{\forall x\ \psi(x)}(\nu) \asymp \mathsf{F}$.

If $\overline{\forall x\ \psi(x)} \asymp \mathsf{U}$, then $(\neg \lceil \forall x\ \psi(x) \rfloor) \in \Gamma_\omega$. If $(\neg \psi(t)) \in \Gamma_\omega$ for any $t \in \mathscr{T}_{\mathsf{def}}$ that is free for x in $\psi(x)$, then $\Gamma_\omega \vdash_{\text{D1; }\wedge\text{-I}} \lceil \neg \psi(t) \rfloor \wedge \neg \psi(t) \vdash_{\text{5.4(6); }\exists\text{-I}} \exists x\, (\lceil \psi(x) \rfloor \wedge \neg \psi(x)) \vdash_{\vee\text{-I2}} \lceil \forall x\ \psi(x) \rfloor \vdash_{\text{C3}} \mathsf{F}$, because by 5.4(9), $\lceil \forall x\ \psi(x) \rfloor \cong (\forall x\, \lceil \psi(x) \rfloor) \vee \exists x\, (\lceil \psi(x) \rfloor \wedge \neg \psi(x))$. So there is no $\overline{t} \in \mathbb{D}$ such that $\overline{\psi(t)} \asymp \mathsf{F}$, that is, $\underline{\psi(x)}(\nu[x := \overline{t}]) \asymp \mathsf{F}$. By 3.2(7), to show $\underline{\forall x\ \psi(x)}(\nu) \asymp \mathsf{U}$, it remains to be proven that there is $t \in \mathscr{T}_{\mathsf{def}}$ that is free for x in $\psi(x)$ such that $\underline{\psi(x)}(\nu[x := \overline{t}]) \asymp \mathsf{U}$, that is, $\overline{\psi(t)} \asymp \mathsf{U}$, that is, $(\neg \lceil \psi(t) \rfloor) \in \Gamma_\omega$. We have $(\forall x\, \lceil \psi(x) \rfloor) \notin \Gamma_\omega$, because otherwise $\Gamma_\omega \vdash_{\vee\text{-I1}} \lceil \forall x\ \psi(x) \rfloor \vdash_{\text{C3}} \mathsf{F}$. On the other hand, $\emptyset \vdash_{\text{D2}} \lceil \psi(x) \rfloor \vee \neg \lceil \psi(x) \rfloor \vdash_{\text{D1; }\vee\text{-E}} \lceil \lceil \psi(x) \rfloor \rfloor \vdash_{\forall\text{-I}} \forall x\, \lceil \lceil \psi(x) \rfloor \rfloor$

$\vdash_{\vee\text{-I1}} \lceil \forall x \lceil \psi(x) \rfloor \rfloor$, so $(\neg \lceil \forall x \lceil \psi(x) \rfloor \rfloor) \notin \Gamma_\omega$. By Lemma 7.3, $(\neg \forall x \lceil \psi(x) \rfloor) \in \Gamma_\omega$. It is of the form $(\neg \forall x \ldots) \in \Gamma_\omega$ that was discussed above, so there is a variable symbol y such that $(\neg \lceil \psi(y) \rfloor) \in \Gamma_\omega$, that is, $\underline{\psi(x)}(\nu[x := \overline{y}]) \asymp \overline{\psi(y)} \asymp \mathsf{U}$.

If $\overline{\exists x\, \psi(x)} \asymp \mathsf{T}$, then $(\exists x\, \psi(x)) \in \Gamma_\omega$. Consider the step $\varphi_i \cong \exists x\, \psi(x)$ in the construction of Γ_ω. Case 1 was not chosen, because otherwise both $\varphi_i \in \Gamma_\omega$ and $(\neg \lceil \varphi_i \rfloor) \in \Gamma_\omega$. Because $(\exists x\, \psi(x)) \in \Gamma_\omega \nvdash \mathsf{F}$, Case 2 was chosen. There the formula $\psi(y)$ was added to Γ_ω, making $\overline{\psi(y)} \asymp \mathsf{T}$. Thus $\underline{\exists x\, \psi(x)}(\nu) \asymp \mathsf{T}$ by Definition 3.2(8).

If $\overline{\exists x\, \psi(x)} \asymp \mathsf{F}$, then $(\neg \exists x\, \psi(x)) \in \Gamma_\omega$. Assume $\overline{t}$ is an arbitrary element of $\mathbb{D}$. By Definition 3.2(8) we get $\underline{\exists x\, \psi(x)}(\nu) \asymp \mathsf{F}$, if we show $\underline{\psi(t)}(\nu) \asymp \mathsf{F}$. We have $\lceil t \rfloor \in \Gamma_\omega$. By Lemma 7.1(1), t can be chosen so that it is free for x in $\psi(x)$ and $\lceil \psi(x) \rfloor$. We show $\overline{\psi(t)} \asymp \mathsf{F}$, that is, $(\neg \psi(t)) \in \Gamma_\omega$, by ruling out the other two possibilities. If $\psi(t) \in \Gamma_\omega$, then $\Gamma_\omega \vdash_{\exists\text{-I}} \exists x\, \psi(x) \vdash_{\text{C3}} \mathsf{F}$. The case $(\neg \lceil \psi(t) \rfloor) \in \Gamma_\omega$ can be split to two via $\Gamma_\omega \vdash_{\text{D1}} \lceil \neg \exists x\, \psi(x) \rfloor \cong_{5.4(10)} (\forall x \lceil \psi(x) \rfloor) \vee \exists x\, (\lceil \psi(x) \rfloor \wedge \psi(x))$. We have $\Gamma_\omega \cup \{\forall x \lceil \psi(x) \rfloor\} \vdash_{\forall\text{-E}} \lceil \psi(t) \rfloor \vdash_{\text{C3}} \mathsf{F}$. Furthermore, $\Gamma_\omega \cup \{\exists x\, (\lceil \psi(x) \rfloor \wedge \psi(x))\} \vdash_{\exists\text{-E}} \mathsf{F}$, since $\{(\lceil \psi(z) \rfloor \wedge \psi(z)), \neg \exists x\, \psi(x)\} \vdash_{\wedge\text{-E2};\, \exists\text{-I}} \mathsf{F}$.

If $\overline{\exists x\, \psi(x)} \asymp \mathsf{U}$, then $(\neg \lceil \exists x\, \psi(x) \rfloor) \in \Gamma_\omega$. There cannot be any $\overline{t} \in \mathbb{D}$ such that $\psi(t) \in \Gamma_\omega$ (where t is chosen so that it is free for x in $\psi(x)$), because otherwise $\Gamma_\omega \vdash_{\text{D1};\, \wedge\text{-I}} \lceil \psi(t) \rfloor \wedge \psi(t) \vdash_{\exists\text{-I}} \exists x\, (\lceil \psi(x) \rfloor \wedge \psi(x)) \vdash_{\vee\text{-I2}} \lceil \exists x\, \psi(x) \rfloor \vdash_{\text{C3}} \mathsf{F}$. By Definition 3.2(8), it remains to be proven that there is $\overline{t} \in \mathbb{D}$ such that $(\neg \lceil \psi(t) \rfloor) \in \Gamma_\omega$. We have $(\forall x \lceil \psi(x) \rfloor) \notin \Gamma_\omega$, because otherwise $\Gamma_\omega \vdash_{\vee\text{-I1}} \lceil \exists x\, \psi(x) \rfloor \vdash_{\text{C3}} \mathsf{F}$. The rest of the proof is the same as in the case $\overline{\forall x\, \psi(x)} \asymp \mathsf{U}$. □

Lemma 7.8. *The* $\mathbb{D}$ *and* _ *defined in the proof of Lemma 7.7 satisfy* $(\mathbb{D}, \_) \models \lceil \mathscr{F} \rfloor$.

Proof. By Definition 5.2(1) and the choice of $\mathbb{D}$ in the proof of 7.7, for every function symbol f and $t_1 \in \mathscr{T}_{\mathsf{def}}, \ldots, t_{\alpha(f)} \in \mathscr{T}_{\mathsf{def}}$ we have to show the following:

$$\underline{\lceil f \rfloor}(\overline{t_1}, \ldots, \overline{t_{\alpha(\lceil f \rfloor)}}) \asymp \mathsf{T} \text{ if and only if } \underline{f}(\overline{t_1}, \ldots, \overline{t_{\alpha(f)}}) \text{ is defined.}$$

By the construction in the proof of 7.7, $\underline{f}(\overline{t_1}, \ldots, \overline{t_{\alpha(f)}})$ is defined if and only if $\lceil f(t_1, \ldots, t_{\alpha(f)}) \rfloor \in \Gamma_\omega$, that is, $\overline{\lceil f(t_1, \ldots, t_{\alpha(f)}) \rfloor} \asymp \mathsf{T}$. We have

$$\overline{\lceil f(t_1, \ldots, t_{\alpha(f)}) \rfloor} \asymp \overline{\lceil f \rfloor(t_1, \ldots, t_{\alpha(\lceil f \rfloor)})} \asymp \underline{\lceil f \rfloor(t_1, \ldots, t_{\alpha(\lceil f \rfloor)})}(\nu) \asymp$$
$$\underline{\lceil f \rfloor}(\underline{t_1}(\nu), \ldots, \underline{t_{\alpha(\lceil f \rfloor)}}(\nu)) \asymp \underline{\lceil f \rfloor}(\overline{t_1}, \ldots, \overline{t_{\alpha(\lceil f \rfloor)}})$$

by the following. The first step follows from $\overline{t_1} \in \mathbb{D}, \ldots, \overline{t_{\alpha(f)}} \in \mathbb{D}$ and 5.4(2). The second and last step hold because by the proof of 7.7, for every formula φ we have $\underline{\varphi}(\nu) = \overline{\varphi}$ and for every defined term t we have $\underline{t}(\nu) = \overline{t}$. The third step follows from 3.2(2), 3.2(3), and the fact that each t_i is free for x_i in $\lceil f \rfloor(x_1, \ldots, x_{\alpha(\lceil f \rfloor)})$ by 5.4(2). □

Theorem 7.9. *If* $(\lceil \mathscr{F} \rfloor, \Gamma) \nvdash \mathsf{F}$, *then* $(\lceil \mathscr{F} \rfloor, \Gamma)$ *has a model. If* $(\lceil \mathscr{F} \rfloor, \Gamma) \models \varphi$, *then* $(\lceil \mathscr{F} \rfloor, \Gamma) \vdash \varphi$.

Proof. If $\Gamma \not\vdash \mathsf{F}$, then Lemma 7.1(4) yields $\Gamma' \not\vdash \mathsf{F}$. By Lemma 7.7 and 7.8 Γ_ω has a model. It is a model of Γ' as well, because $\Gamma' \subseteq \Gamma_\omega$. So Γ has a model by Lemma 7.1(5).

If $\Gamma \not\vdash \varphi$, then by Lemma 6.7 there is $\psi \in \{\neg\varphi, \neg\lceil\varphi\rfloor\}$ such that $\Gamma \cup \{\psi\} \not\vdash \mathsf{F}$. By the previous claim $\Gamma \cup \{\psi\}$ has a model. By Definition 5.2(3) it contradicts the assumption $\Gamma \models \varphi$. □

8 Discussion

Let us first discuss the mimicking of other logics by ours. Some logics [6, 16] use strict logical connectives; that is, if φ or ψ or both are undefined, then also $\varphi \wedge \psi$ and $\varphi \vee \psi$ are undefined. They can be reduced to our system by interpreting them as shorthands for $(\varphi \wedge \neg\varphi) \vee (\psi \wedge \neg\psi) \vee (\varphi \wedge \psi)$ and $(\varphi \vee \neg\varphi) \wedge (\psi \vee \neg\psi) \wedge (\varphi \vee \psi)$, respectively. The $\forall x\, \varphi(x)$ and $\exists x\, \varphi(x)$ of [6, 18] are our $(\forall x\, \varphi(x)) \vee \exists x\, (\varphi(x) \wedge \neg\varphi(x))$ and $(\exists x\, \varphi(x)) \wedge \forall x\, (\varphi(x) \vee \neg\varphi(x))$, respectively.

Some logics [18] (a variant in [6]) interpret $\varphi \wedge \psi$ and $\varphi \vee \psi$ like $\varphi \wedge (\neg\varphi \vee \psi)$ and $\varphi \vee (\neg\varphi \wedge \psi)$, respectively, are interpreted in our logic. It corresponds to how "and" and "or" work in many programming languages, assuming that U represents program crash or undefined behavior. In the case of "and", if φ yields F, then F is returned without evaluating ψ; if the evaluation of φ crashes, then the evaluation of "and" has crashed; and if φ yields T, then ψ is evaluated resulting either in a crash or a truth value F or T that is returned. "Or" is computed analogously. In programming literature, this is often called "short-circuit evaluation".

Relation symbols R' that are not defined everywhere can be simulated as follows. A new function symbol f is introduced that is undefined precisely when desired. We choose $(d_1, \ldots, d_{\alpha(R)}) \in \underline{R}$ when $\underline{f}(d_1, \ldots, d_{\alpha(f)})$ is undefined, and use $R(x_1, \ldots, x_{\alpha(R)}) \wedge (f(x_1, \ldots, x_{\alpha(f)}) = f(x_1, \ldots, x_{\alpha(f)}))$ in the place of R'.

The sources [3, 9, 25] introduce an if-then-else operator for terms. It is obviously useful for defining functions in a recursive fashion, which is common practice in computer science. The atomic formula $R(\text{if } \chi \text{ then } t_1 \text{ else } t_2)$ can be treated as an abbreviation of $(\chi \wedge R(t_1)) \vee (\neg\chi \wedge R(t_2)) \vee (\chi \wedge \neg\chi)$. This was exemplified in (1), where $|x| = (\text{if } x < 0 \text{ then } -x \text{ else } x)$, and $(\chi \wedge \neg\chi)$ was omitted because in this case it always yields F. This makes it also possible to introduce non-strict functions, such as $(\text{if } x = 0 \text{ then } 0 \text{ else if } y = 0 \text{ then } 0 \text{ else } x \cdot y)$, which is a version of multiplication that yields 0 also when one argument is 0 and the other argument is undefined. (A function is strict if and only if an undefined argument always makes the result undefined.)

Some logics contain a non-strict unary connective that inputs a truth value and yields T if the input is F or T, and F if the input is U. That is, it is otherwise like our $\lceil\varphi\rfloor$, but it is a connective while $\lceil\rfloor$ is an abbreviation. In [6] it is $\#\varphi$, in [9, 13] it is $\Delta\varphi$, and in [18] it is $*\varphi$. Let us use $*$. In the presence of $*$ and some way to express the constant U, any truth function of arity $n+1$ (including the irregular ones) can be expressed recursively as $(P_{n+1} \wedge *P_{n+1} \wedge \varphi_{\mathsf{T}}) \vee (\neg P_{n+1} \wedge *P_{n+1} \wedge \varphi_{\mathsf{F}}) \vee$

$(\neg * P_{n+1} \wedge \varphi_{\mathsf{U}})$, where φ_{T}, φ_{F}, and φ_{U} express truth functions of arity n. In our logic, this can be mimicked by using $\lceil \varphi \rfloor$ instead of $*\varphi$. To obtain U one may declare a unary function symbol f with $\lceil f \rfloor \cong \mathsf{F}$ (that is, f is defined nowhere), and use $f(x) = f(x)$. In this way, any propositional connective can be mimicked.

If the axiom system that is to be mimicked does not contain a natural counterpart for $\lceil f \rfloor$, then a new relation symbol R_f may be introduced and used as $\lceil f \rfloor$. This is not void of content, because information about R_f may then follow from other axioms. Let us illustrate this with an example. Assume $(\forall x\, (x = 0 \vee x \cdot \frac{1}{x} = 1)) \in \Gamma$. We have $\{x \cdot \frac{1}{x} = 1\} \vdash_{\mathrm{D1}} \lceil x \cdot \frac{1}{x} = 1 \rfloor \vdash_{5.4(4);\, \wedge\text{-E1}} \lceil x \cdot \frac{1}{x} \rfloor \vdash_{5.4(2);\, \wedge\text{-E1};\, \wedge\text{-E2}} \lceil \frac{1}{x} \rfloor \cong R_{\frac{1}{x}}(x)$. Thus $\Gamma \cup \{\neg(x = 0)\} \vdash_{\forall\text{-E};\, \vee\text{-E}} R_{\frac{1}{x}}(x)$. Assume that originally $\neg * (\frac{1}{0} = \frac{1}{0})$ was used to express that $\frac{1}{0}$ is undefined. We mimic it by letting $(\neg \lceil \frac{1}{0} = \frac{1}{0} \rfloor) \in \Gamma$. Then $\Gamma \vdash \neg R_{\frac{1}{x}}(0)$, because $\{R_{\frac{1}{x}}(0)\} \vdash \mathsf{T} \wedge \mathsf{T} \wedge R_{\frac{1}{x}}(0) \cong \lceil \frac{1}{0} \rfloor \vdash_{=\text{-1}} \frac{1}{0} = \frac{1}{0} \vdash_{\mathrm{D1}} \lceil \frac{1}{0} = \frac{1}{0} \rfloor$.

The $E!t$, where t is a term, of free logics can be mimicked with $\lceil t \rfloor$. The $\exists x\, \varphi(x)$ of [16] can be mimicked with $\exists x\, (\lceil \varphi(x) \rfloor \wedge \varphi(x))$.

The observations above suggest that most, if not all, ternary first-order logics for partial functions can be mimicked by our logic. On the other hand, our completeness result can be generalized to also cover $*$. Because $\neg*(\frac{1}{0} = 0)$ is true but $\exists x\, \neg*(x = 0)$ is false on real numbers, $\exists$-I is not sound in the presence of $*$. Therefore, we replace it by "If t is free for x in $\varphi(x)$, then $\{\lceil t \rfloor, \varphi(t)\} \vdash \exists x\, \varphi(x)$". This makes the soundness proof go through despite the fact that due to $*$, the logic is no longer regular. Then $*$ is given proof rules as suggested by Definition 5.4. It is not hard to check that every instance of $\exists$-I in our completeness proof uses as t either a variable symbol or a term such that $\bar{t} \in \mathbb{D}$, that is, $\lceil t \rfloor \in \Gamma_\omega$. Therefore, also the completeness proof goes through.

Let us now briefly discuss which familiar laws must be changed when switching from binary to our logic. We have already mentioned that C1, D1, =-1, $\forall$-E, and the modified $\exists$-I differ from the binary case. Of the 21 propositional laws in [11, Table 6.3] that do not use $\rightarrow$ or $\leftrightarrow$, only the following four fail in our logic: $P \vee \neg P$ equals T (Law of Excluded Middle), $P \wedge \neg P$ equals F (Law of Noncontradiction), $P \wedge (\neg P \vee Q)$ equals $P \wedge Q$, and $P \vee (\neg P \wedge Q)$ equals $P \vee Q$ (short-circuit vs. ordinary conjunction and disjunction). All of the 14 quantifier laws in [11, (7.1), (7.6), p. 380] that do not use $\rightarrow$ or $\leftrightarrow$ are valid in our logic.

The situation with $\rightarrow$ is less clear, starting from the question what it should mean. All the 3-valued logics that we checked use either Kleene's [3, 9, 13], strict [6, 16, 18], or no [5, 15] $\rightarrow$. (Some denote it with $\Rightarrow$.) In classical binary first-order logic, $\rightarrow$ is closely linked to $\vdash$ via Modus Ponens ($\{(\varphi \rightarrow \psi), \varphi\} \vdash \psi$) and Deduction Theorem (if $\Gamma \cup \{\varphi\} \vdash \psi$, then $\Gamma \vdash \varphi \rightarrow \psi$). Kleene's, Łukasiewicz's, and strict implication satisfy the former, but, as we now show, not the latter. Let c be a constant symbol, f be a unary function symbol with $\lceil f \rfloor \cong \mathsf{F}$, $\varphi \cong (f(c) = c)$, and $\psi \cong \neg(c = c)$. Then in any model $\underline{\varphi}(v) \asymp \mathsf{U}$ and $\underline{\psi}(v) \asymp \mathsf{F}$, and $\underline{\varphi \rightarrow \psi}(v) \asymp \mathsf{U}$ by Figure 1 or strictness. Furthermore, $\emptyset \cup \{\varphi\} \vdash_{\mathrm{D1}} \lceil \varphi \rfloor \cong_{5.4(4),(2)} \lceil c \rfloor \wedge \lceil f \rfloor(c) \wedge \lceil c \rfloor \cong_{5.4(1)} \mathsf{T} \wedge \mathsf{F} \wedge \mathsf{T} \vdash_{\wedge\text{-E1},\, \wedge\text{-E2}} \mathsf{F} \vdash_{\mathrm{C2}} \psi$. If Deduction Theorem holds, we get $\emptyset \vdash \varphi \rightarrow \psi$, demonstrating that the proof system is unsound.

Both Modus Ponens and Deduction Theorem hold in our logic if we define $\varphi \rightarrow \psi$ as a shorthand for $\neg(\varphi \wedge \lceil\varphi\rfloor) \vee \psi$ or $\neg(\varphi \wedge \lceil\varphi\rfloor) \vee (\psi \wedge \lceil\psi\rfloor)$. On the other hand, of the 8 laws in [11, Table 6.3] that contain $\rightarrow$, Kleene's implication violates only the law that $P \rightarrow P$ equals T, while Łukasiewicz's violates 3 laws, $\neg(\varphi \wedge \lceil\varphi\rfloor) \vee \psi$ violates 4, $\neg(\varphi \wedge \lceil\varphi\rfloor) \vee (\psi \wedge \lceil\psi\rfloor)$ violates 5, and strict implication violates 6. As a consequence, maintaining Deduction Theorem and maintaining as many familiar practical laws as possible seem conflicting goals. Therefore, we leave it open what $\rightarrow$ should mean in the context of our logic.

The main message of this section is the following. It is often possible to use $\lceil\rfloor$ instead of $*$ without losing completeness. By doing so other connectives than $\neg$, $\wedge$, $\vee$, $\forall$, and $\exists$ can be eliminated, facilitating some useful practical reasoning methods.

Acknowledgements. We thank Esko Turunen for the help he gave in checking earlier versions of our proofs; Cliff Jones, Scott Lehmann and Fred Schneider for helpful discussions on the topic; and the anonymous reviewers for their hard work. Our special thanks go to the reviewer who pointed out that for the proof system to be recursive, also the function from function symbols to their isdef-formulas must be recursive.

References

[1] H. Andréka, W. Craig, and I. Németi. A system of logic for partial functions under existence-dependent Kleene equality. *J. Symbolic Logic*, 53(3):834–839, 1988.

[2] H. Barringer, J. H. Cheng, and C. B. Jones. A logic covering undefinedness in program proofs. *Acta Inform.*, 21(3):251–269, 1984.

[3] Sergey Berezin, Clark Barrett, Igor Shikanian, Marsha Chechik, Arie Gurfinkel, and David L. Dill. A practical approach to partial functions in CVC Lite. In *Selected papers from the workshops on disproving and the second international workshop on pragmatics of decision procedures (PDPAR 2004), Cork, Ireland, 2004*, pages 13–23. Amsterdam: Elsevier, 2005.

[4] Patrice Chalin. Logical foundations of program assertions: What do practitioners want? In *Third IEEE International Conference on Software Engineering and Formal Methods (SEFM 2005), 7-9 September 2005, Koblenz, Germany*, pages 383–393. IEEE Computer Society, 2005.

[5] Ádám Darvas, Farhad Mehta, and Arsenii Rudich. Efficient well-definedness checking. In *Automated reasoning. 4th international joint conference, IJCAR 2008, Sydney, Australia, August 12–15, 2008 Proceedings*, pages 100–115. Berlin: Springer, 2008.

[6] Hans de Nivelle. Theorem proving for classical logic with partial functions by reduction to Kleene logic. *J. Logic Comput.*, 27(2):509–548, 2017.

[7] David A. Duffy. *Principles of automated theorem proving*. Wiley Professional Computing. John Wiley & Sons, Ltd., Chichester, 1991.

[8] William M. Farmer and Joshua D. Guttman. A set theory with support for partial functions. *Studia Logica*, 66(1):59–78, 2000. Partiality and modality (Montréal, QC, 1995).

[9] Antonio Gavilanes-Franco and Francisca Lucio-Carrasco. A first order logic for partial functions. *Theoret. Comput. Sci.*, 74(1):37–69, 1990.

[10] David Gries and Fred B. Schneider. Avoiding the undefined by underspecification. In *Computer science today*, volume 1000 of *Lecture Notes in Comput. Sci.*, pages 366–373. Springer, Berlin, 1995.

[11] James L. Hein. *Discrete Structures, Logic, and Computability*. Boston, MA: Jones and Bartlett Publishers, 1995.

[12] Leon Henkin. The completeness of the first-order functional calculus. *J. Symbolic Logic*, 14:159–166, 1949.

[13] C. B. Jones and C. A. Middelburg. A typed logic of partial functions reconstructed classically. *Acta Inform.*, 31(5):399–430, 1994.

[14] Stephen Cole Kleene. *Introduction to metamathematics*. D. Van Nostrand Co., Inc., New York, N. Y., 1952.

[15] Scott Lehmann. Strict Fregean free logic. *J. Philos. Logic*, 23(3):307–336, 1994.

[16] Scott Lehmann. “no input, no output” logic. In *New essays in free logic. In honour of Karel Lambert*, pages 147–155. Dordrecht: Kluwer Academic Publishers, 2001.

[17] J. Łukasiewicz. Philosophische Bemerkungen zu mehrwertigen Systemen des Aussagenkalküls. *C. R. Soc. Sci. Varsovie, Cl. III*, 23:51–77, 1931.

[18] John McCarthy. Predicate calculus with “undefined” as a truth-value. Technical report, Stanford Artificial Intelligence Project Memo 1, 1963.

[19] Maurizio Negri. An algebraic completeness proof for Kleene’s 3-valued logic. *Boll. Unione Mat. Ital. Sez. B Artic. Ric. Mat. (8)*, 5(2):447–467, 2002.

[20] John Nolt. Free logic. In *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Winter 2020 edition, 2020.

[21] Christos H. Papadimitriou. *Computational complexity*. Addison-Wesley Publishing Company, Reading, MA, 1994.

[22] David Lorge Parnas. Predicate logic for software engineering. *IEEE Trans. Software Eng.*, 19(9):856–862, 1993.

[23] Edi Pavlović and Norbert Gratzl. A more unified approach to free logics. *J. Philos. Logic*, 50(1):117–148, 2021.

[24] Yaroslav Petrukhin. Natural deduction for three-valued regular logics. *Log. Log. Philos.*, 26(2):197–206, 2017.

[25] Birgit Schieder and Manfred Broy. Adapting calculational logic to the undefined. *Comput. J.*, 42(2):73–81, 1999.

[26] J. M. Spivey. *The Z Notation. A Reference Manual*. New York etc.: Prentice Hall, 1989.

[27] Antti Valmari. Automated checking of flexible mathematical reasoning in the case of systems of (in)equations and the absolute value operator. In *Proceedings of the 13th International Conference on Computer Supported Education, CSEDU 2021, Online Streaming, April 23-25, 2021, Volume 2*, pages 324–331. SCITEPRESS, 2021.

[28] Antti Valmari and Lauri Hella. The logics taught and used at high schools are not the same. In *Proceedings of the Fourth Russian Finnish Symposium on Discrete Mathematics*, volume 26 of *TUCS Lecture Notes*, pages 172–186, Turku, Finland, 2017. Turku Centre for Computer Science. editors: Juhani Karhumäki and Yuri Matiyasevich and Aleksi Saarela.

[29] Antti Valmari and Johanna Rantala. Arithmetic, logic, syntax and MathCheck. In *Proceedings of the 11th International Conference on Computer Supported Education, CSEDU 2019, Heraklion, Crete, Greece, May 2-4, 2019, Volume 2.*, pages 292–299, Setúbal, Portugal, 2019. SciTePress. editors: H. Lane and Susan Zvacek and James Uhomoibhi.

[30] Stefan Wintein. On all strong Kleene generalizations of classical logic. *Studia Logica*, 104(3):503–545, 2016.