A BASIC FORMAL EQUATIONAL PREDICATE LOGIC – PART I

Abstract

We present, in two parts, the details of a formalization of Equational Predicate Logic based on a “propositional” version of the Leibniz rule (Propositional Strong Leibniz, or PSL), and Equanimity (EQN). Both rules are “strong”, that is, they are applicable to arbitrary premises (not just to absolute theorems).

We show (Part II) that a strong “no-capture” Leibniz, and a weak “full-capture” version are derived rules (both access the interior of quantifier scopes).

We also derive general rules MON (monotonicity) and AMON (antimonotonicity), which allow us to “calculate” appropriate conclusions ⊢ C[p \ A] ⇒ C[p \ B] or ⊢ C[p \ A] ⇔ C[p \ B] from the assumption ⊢ A ⇒ B – “[p \ …]” denoting full-capture substitution. We show that these rules are “as strong as possible” for our logic. Finally, we show that our logic is sound and complete.

0. Introduction

Equational or Calculational Logic, introduced by Dijkstra and Scholten [2] and further promoted in the recent text [5] and papers [4], [6], [7] of Gries and Schneider, is finding its way in Computer Science curricula as a front-end for the “Discrete Math” and subsequent “theoretical” components.

It is argued by its proponents (e.g., [4], [6], [7]) that Equational Logic is the pedagogically proper setting to do proofs – i.e., work within the theory. The argument maintains that the Equational or Calculational approach provides a powerful yet natural and user-friendly “Calculus”, which,
George Tourlakis

once mastered by the practitioner, will provide him or her with a methodology of reasoning that is suitable to use in diverse computer science and mathematics fields of application.

It is also a fair statement that Equational Logic has had a more cumbersome metatheory than that of the “standard” approaches (e.g., [3], [8], [9]), because its user-friendliness has often come at the expense of succinctness: Far too many axioms and far too many rules of inference are often adopted (by comparison, again, to standard approaches). But this does not have to be so.

The purpose of [10] was to faithfully formalize the perceived intentions – and their semi-formal realization as practices – of [5] and prove that this framework is both sound and complete, or that it can be made to be so with minimal tinkering that does not betray its underlying philosophy.\(^1\)

The concluding remarks in [10] suggested a simpler way of going about founding Equational Logic: A manner where the axiom set is smaller, the set of rules of inference is smaller, and the inference rule Substitution is not included anymore (being replaced by the adoption of axiom schemata).

The purpose of the present paper (Parts I and II) is to implement the above idea in detail, and to briefly follow-up some of its metamathematical and practical consequences. Thus, we propose here a formalization which, unlike the one in [10], bases the proof-apparatus solely on two propositional rules of inference one of which, of course, is a version of “Leibniz”. This entails an unconstrained Deduction Theorem (contrast with [10]), which in turn further simplifies the steps of calculational reasoning. Even though our primary rules include just a propositional version of Leibniz – that is, substitutions are done into Boolean variables that are not in the scope of a quantifier – we prove the validity of two derived rules that allow the use of Leibniz-style substitution within the scope of a quantifier after all.

The layout of the paper is as follows: In Part I, section 1 introduces the formal language. Section 2 the axioms, the rules of inference, and sets the rules of the game (definitions of the two main types of substitution, of theorem and of proof). Section 3 introduces a few metatheorems, including the Deduction Theorem – all these tools needed in Part II.

In Part II we resume section numbering from where we left in Part I. Section 4 derives some powerful Leibniz rules. Section 5 is on monotonicity

\(^1\)[6] also address this metamathematical concern of soundness and completeness, but only for the Propositional segment of the Logic.
and antimonotonicity. Section 6 is a rapid look at semantics, mostly a set of links to the literature (in particular to [10]). An Appendix discusses some fine points regarding “strong” versus “weak” premises in the application of Leibniz.

1. Syntax

Equational (first order) logic, like all (first order) logic, is “spoken” in a first order language, $L$. $L$ is a triple $(V, \text{Term}, \text{Wff})$, where $V$ is the alphabet used to built terms and formulas. $V$ contains the usual symbols of a first order language, namely, an enumerable set of object variables $x_1, x_2, \ldots$ (metalinguistically denoted by $x, y, z, u, v, w$ with or without primes); equality between “objects”, “$\approx$”;\footnote{Following the practice in Enderton [3], we use “$\approx$” for formal equality and “$=$” for informal (metamathematical) equality. This will enable us to write in the metatheory, for example, $A = B$ where $A$ and $B$ are formulas, meaning that the strings $A$ and $B$ are identical. A conflict does arise though, since one also uses “$=$” quasi-formally (in equational style proofs) as a conjunctional alias of “$\equiv$”.} “(“ and “)”\footnote{}; the Boolean (or propositional) connectives, $\equiv, \Rightarrow, \lor, \land, \neg$; the universal quantifier, $\forall$ (the existential quantifier, $\exists$, will be taken to be a meta-symbol, introduced definitionally); a set of symbols (possibly empty) for individual constants (metalinguistically denoted by $a, b, c, d, e$ with or without primes – if we have no alternative “standard” notation in mind such as $\emptyset$, 0, $\omega$, etc.); a set of symbols for predicates or relations (possibly empty) for each possible “arity” $n > 0$ (metalinguistically denoted by $P, Q, R$ with or without primes – if we have no alternative “standard” notation in mind such as $<$, $\in$, etc.); and a set of symbols for functions (possibly empty) for each possible “arity” $n > 0$ (metalinguistically denoted by $f, g, h$ with or without primes – if we have no alternative “standard” notation in mind such as $+$, $\times$, etc.). Additionally, $V$ contains the special Boolean (propositional) “constants” true and false; and an enumerable set $v_1, v_2, \ldots$, of Boolean or Propositional variables (metalinguistically denoted by $p, q, r$ with or without primes).

Remarks 1.1.

(1) Any two symbols of $V$ are distinct. Moreover (if they are built from simpler “sub-symbols”, e.g., $x_1, x_2, x_3, \ldots$ might really be $x|x,x||x,x||x,\ldots$), none is a substring (or subexpression) of any other.
(2) Boolean variables are introduced solely to make the application of “Rule Leibniz” transparent. We see in section 6 (Part II) that they are, theoretically, redundant. The Boolean constants are also introduced solely for convenience.

(3) When we say “constant” anywhere below here we mean “individual constant symbol”, not a Boolean constant (true or false). The latter we always refer to by name, or qualify: “Boolean”.

We omit the standard inductive definitions for the set of terms, Term, and the set of well-formed formulas, Wff (see [3]). The only difference here will be in the definition of the set of atomic formulas, Af, needed in the definition of Wff: We must add to the standard definition ([3]) that the formulas p (for any p), true and false are also atomic.

Remarks 1.2.

(1) We will let the letters A, B, C, D, E (with or without primes) be names for arbitrary formulas.

(2) We introduce a meta-symbol (∃) solely in the metalanguage via the definition “((∃x).A) stands for, or abbreviates, (¬((∀x)(¬A))).”

(3) We often write more explicitly, ((∀x).A[x]) and ((∃x).A[x]) for ((∀x).A) and ((∃x).A). This is intended to draw attention to the variable x that may or may not occur free in A, which has now become “bound”.

(4) To minimize the use of brackets in the metanotation we adopt standard priorities, that is, ∀, ∃, and ¬ have the highest, and then we have (in decreasing order of priority) ∧, ∨, ⇒, ≡. All associativities are right.

(5) We omit the standard (e.g., [3]) inductive definition of free and bound variables. A term or formula is closed iff no free variables occur in it. A closed formula is also called a sentence.

2. Axioms and Rules of Inference

We choose (logical) axioms and rules of inference so that a logic equivalent to that in [1], [3] results. The characterizing feature of such logics is that the primary rules of inference are propositional (they do not access the interior of quantifier scopes).

We will need a precise definition of tautologies in our first order language L.
Definition 2.1. [Prime formulas in Wff] A formula \( A \in \text{Wff} \) is a prime formula if it is either atomic or a formula of the form \((\forall x)A\).

Remark 2.2. Let \( \mathcal{P} \) denote the set of all prime formulas in our language, except the formulas true and false. Using \( \mathcal{P} \) as the (new) set of propositional variables, the symbols true and false, brackets, and the Boolean connectives, we may define the set of Propositional Calculus formulas over \( \mathcal{P} \), say Prop-Wff, in the usual way. Clearly, \( A \in \text{Wff} \) iff \( A \in \text{Prop-Wff} \).

Definition 2.3. [Tautologies in Wff] A formula \( A \in \text{Wff} \) is a tautology iff it is so when viewed as a formula of Prop-Wff. We call the set of all tautologies, as defined here, \( \text{Taut} \). The symbol \( \models_{\text{Taut}} \) says \( A \in \text{Taut} \).

Definition 2.4. [Tautologically Implies, for formulas in Wff] Given formulas \( B \in \text{Wff} \) and \( A_i \in \text{Wff} \), for \( i = 1, \ldots, m \). We say that \( A_1, \ldots, A_m \) tautologically implies \( B \), in symbols \( A_1 \Rightarrow \ldots \Rightarrow A_m \Rightarrow B \), iff \( \models_{\text{Taut}} A_1 \Rightarrow \ldots \Rightarrow A_m \Rightarrow B \).

Definition 2.5. [Substitutions] We will have two types of substitutions:

Contextual (or “context-sensitive”) Substitution: \( A[p := W] \) and \( A[x := t] \) denote, respectively, the result of replacing all occurrences of \( p \) by the formula \( W \) and all free occurrences of \( x \) by the term \( t \), provided no variable of \( W \) or \( t \) was “captured” (by a quantifier) during substitution. If the proviso is not valid, then the substitution is undefined.

Uniform (or “context free”) Substitution: \( A[p \setminus W] \) and \( A[x \setminus t] \) denote, respectively, the result of replacing all occurrences of \( p \) by the formula \( W \) and all free occurrences of \( x \) by the term \( t \). No restrictions.

The symbols \( [p := \ast] \) and \( [p \setminus \ast] \) above, where \( \ast \) is a \( W \) or a \( t \), lie in the metalanguage. These metasymbols have the highest priority.

Remark 2.6. An inductive definition (by induction first on terms \( s \) and then on formulas \( A \)) of the string \( A[x := t] \) is standard ([3]) and therefore omitted.

Similarly, one defines \( A[p := W] \) by (below \( \circ \in \{\equiv, \lor, \land, \Rightarrow\} \)):

\[ A[p := W] \] is the same string as

\[ \equiv \circ \ast \]

\[ \equiv \circ \ast \]

\[ \equiv \circ \ast \]

\[ \equiv \circ \ast \]

\[ \equiv \circ \ast \]

\[ \equiv \circ \ast \]

\[ \equiv \circ \ast \]

\[ \equiv \circ \ast \]

In any context, of course, propositional variables are whatever symbols we say are “propositional variables”!
The cases for $A[x \setminus t]$ and $A[p \setminus W]$ have exactly the same inductive definitions, except that we drop the hedging “if defined” throughout, and we also drop the restriction that $y$ (of $(\forall y)$) be not free in $W$ or $t$.

We say that $B$ is a partial generalization of $A$ iff it is the expression consisting of $A$, prefixed with zero or more strings “$(\forall x)$” for various “$x$” – $x$ may or may not occur free in $A$. Repetitions of the same prefix-string “$(\forall x)$” are allowed. The well known “universal closure of $A$”, that is, $(\forall x_1)(\forall x_2)\ldots(\forall x_n)A$ – where $x_1, x_2, \ldots, x_n$ is the full list of free variables in $A$ – is a special case.

**Definition 2.7.** [Axioms and Axiom schemata] The axioms (schemata) are all the possible “partial” generalizations of the following (exactly as in [3])

- **Ax1.** All formulas in Taut.
  - For every formula $A$, $(\forall x)A \Rightarrow A[x := t]$, for any term $t$.
  - By 2.5, the notation implies no capture substitution. We say that “$t$ must be substitutable in $x$”.

- **Ax2.** For every formula $A$, $(\forall x)A \Rightarrow A[x := t]$, for any term $t$.
  - NB. We often see the above written (in metalinguistic argot) as $(\forall x)A[x] \Rightarrow A[t]$ or even $(\forall x)A \Rightarrow A[t]$, where the presence of $A[x]$ (or $(\forall x)A$, or $(\exists x)A$) and $A[t]$ in the same context means that $t$ replaces contextually all $x$ occurrences in $A$.

- **Ax3.** For every formula $A$ and variable $x$ not free in $A$, $A \Rightarrow (\forall x)A$.
  - **Ax4.** For every formulas $A$ and $B$, $(\forall x)(A \Rightarrow B) \Rightarrow (\forall x)A \Rightarrow (\forall x)B$.

- **Ax5.** For each object variable $x$, the formula $x \approx x$.

- **Ax6.** (Leibniz’s characterization of equality–1st order version.) For any atomic formula $A$, any object variable $x$ and any term $t$, the formula $x \approx t \Rightarrow (A \equiv A[x := t])$.

**NB.** The above is written usually as $x \approx t \Rightarrow (A[x] \equiv A[t])$ or even $x \approx t \Rightarrow (A \equiv A[t])$. 
The following two are the primary rules of inference. We emphasize that the domain of the rules we describe below is the entire set $\text{Wff}$. That is why we call the rules “strong”. A “weak” rule applies on a proper subset of $\text{Wff}$ only – that of “absolute theorems” – a subset that is not yet defined. We say that any set $S \subseteq \text{Wff}$ is closed under some rule of inference iff whenever the rule is applied to formulas in $S$, it also yields formulas in $S$.

**Definition 2.8. [Rules of Inference]**

**Inf1.** (Propositional (Strong) Leibniz, $\text{PSL}$) For any formulas $A, B, C$ and any propositional variable $p$ (which may or may not occur in $C$)

$$A \equiv B \quad C[p := A] \equiv C[p := B]$$

(*PSL*)

on the condition that $p$ is not in the scope of a quantifier. Given the condition on $p$, it makes no difference if in $\text{PSL}$ above we used $[p \setminus *]$ instead of $[p := *]$.

**Inf2.** (Equanimity, $\text{EQN}$) For any formulas $A, B$

$$A, A \equiv B \quad \frac{}{B}$$

(*$\text{EQN}$*)

**Remark 2.9.** The Leibniz rule (or its variants) is at the heart of equational or calculational reasoning. In “standard” approaches to logic it rather appears as the well known “derived rule” that if $\Gamma \vdash A \equiv B$ \footnote{The meaning of the symbol $\vdash$ is defined in 2.10 below.} and if we replace one or more occurrences of the subformula $A$ of a formula $D$ (here $D$ is $C[p := A]$) by $B$, to obtain $D'$ (that is $C[p := B]$), then $\Gamma \vdash D \equiv D'$. It turns out that no restriction on $p$ is necessary (see section 4) after all. Shoenfield [9] calls this derived rule “the equivalence theorem”\footnote{Actually, the syntactic apparatus in [9], but not the one in this paper – see section 4 – allows a stronger “Leibniz”. It allows Inf1 with uniform substitution and without the caveat on the variable $p$. See also [10].}.

We next define $\Gamma$-theorems, that is, formulas we can prove from the set of formulas $\Gamma$ ($\Gamma$ may be empty).

**Definition 2.10. [$\Gamma$-theorems]** The set of $\Gamma$-theorems, $\text{Thm}_\Gamma$, is the $\subseteq$-smallest subset of $\text{Wff}$ that satisfies the following: Th1. $\text{Thm}_\Gamma$ contains...
as a subset all the axioms defined in 2.7. We call these formulas the logical
axioms. Th2. $\Gamma \subseteq \text{Thm}_\Gamma$. We call every member of $\Gamma$ a nonlogical axiom.
Th3. $\text{Thm}_\Gamma$ is closed under each rule $\text{Inf1–Inf2}$.

We write $\Gamma \vdash A$ for $A \in \text{Thm}_\Gamma$ and say that $A$ is proved from $\Gamma$, or
that it is a $\Gamma$-theorem. If $\Gamma = \emptyset$, then rather than $\emptyset \vdash A$ we write $\vdash A$. We
say in this case that $A$ is absolutely provable (or provable with no nonlogical
axioms). We write $A, B, \ldots, D \vdash E$ for $\{A, B, \ldots, D\} \vdash E$.

Remark 2.11. Now we can spell out what a “weak” rule of inference is:
It is a rule whose domain is restricted to be $\text{Thm}_\emptyset$. None of $\text{Inf1–Inf2}$ is
weak.

Definition 2.12. ($\Gamma$-proofs) A finite sequence $A_1, \ldots, A_n$ of members of
$\text{Wff}$ is a $\Gamma$-proof iff every $A_i$, for $i = 1, \ldots, n$ is one of $\text{Pr1}$. A logical
axiom (2.7). $\text{Pr2}$. A member of $\Gamma$. $\text{Pr3}$. The result of a rule $\text{Inf1–Inf2}$
applied to (an) appropriate formula(s) $A_j$ with $j < i$.

Remarks 2.13.

(1) It is a well known result on inductive definitions that $\Gamma \vdash A$ is
equivalent to “$A$ appears in some $\Gamma$-proof” – in the sense of the above
definition – and also equivalent to “$A$ is at the end of some $\Gamma$-proof”.

(2) It follows from 2.12 that if each of $A_1, \ldots, A_n$ has a $\Gamma$-proof and
$B$ has an $\{A_1, \ldots, A_n\}$-proof, then $B$ has a $\Gamma$-proof. We refer to this
phenomenon as “the transitivity of $\vdash$”.

(3) If $\Gamma \subseteq \Delta$ and $\Gamma \vdash A$, then also $\Delta \vdash A$ as it follows from 2.10
or 2.12. In particular, $\vdash A$ implies $\Gamma \vdash A$ for any $\Gamma$.

3. Basic metatheorems

Meta-theorem 3.1. [“Redundant true”] For any formula $A$ and any set
of formulas $\Gamma$, $\Gamma \vdash A$ iff $\Gamma \vdash A \equiv \text{true}$.

Proof. $\models \text{Taut } A \equiv (A \equiv \text{true}), \models \text{Taut } (A \equiv \text{true}) \equiv A$ and EQN. $\Box$

Meta-theorem 3.2. [Modus ponens] $A, A \Rightarrow B \vdash B$ for any formulas
$A$ and $B$.

NB. From 2.10 follows that any one of the primary rules can be writ-
ten “linearly” rather than as a “fraction”. That is, premises first, followed
A basic formal equational logic – Part I

by ⊢, followed by the conclusion. We (almost) always use this linear format for derived rules.

**Proof.** Here Γ = \{A, A \Rightarrow B\}. Thus,

\[ A \Rightarrow B = \langle 3.1 \text{ and } \Gamma \vdash A; \text{ plus PSL} \rangle \text{true} \Rightarrow B \]

\[ = \langle \models \text{Taut}(\text{true} \Rightarrow B) \equiv B \rangle \]

Since Γ ⊢ A ⇒ B, the above “calculation” and EQN yield Γ ⊢ B.

**Remark 3.3** The rule “transitivity”, namely \( A \equiv B, B \equiv C \vdash A \equiv C \), is a derived rule as one can verify by applying PSL to \( A \equiv p \) with premise \( B \equiv C \), and following this up by an application of EQN.

Let us call our logic, that is, any language \( L \) along with the adopted axioms (2.7), Inf1, Inf2, and the definition (2.10) of Γ-theorems (an) E-logic (“E” for Equational).

Let us call En-logic what we obtain by keeping all else the same, but adopting modus ponens as the only primary rule of inference. This is, essentially, the logic in [3] (except that [3] allows neither Boolean variables nor Boolean constants).

We may subscript the symbol \( \vdash \) by an E or En to indicate in which logic we are working. Thus, e.g., \( \Gamma \vdash_{En} A \) means we deduced \( A \) from \( \Gamma \) working in En-logic.

We introduce En-logic as a convenience because its metatheory is a bit simpler due to the presence of just one solitary rule of inference. For this reason, in this paper we often work in En-logic when proving metatheorems. We next show that E-logic and En-logic are equivalent.

**Lemma 3.4.** [The extended tautology theorem] If \( A_1, \ldots, A_n \models_{\text{Taut}} B \) then, \( A_1, \ldots, A_n \vdash B \), in either E-logic or En-logic.

---

6Recall that we a have a different language for each choice of the set of “nonlogical symbols”: constants, functions, predicates.
Proof. The assumption reads

\[ \vdash \text{Taut } A_1 \Rightarrow \ldots \Rightarrow A_n \Rightarrow B \]  

(1)

Thus, since the formula in (1) is an axiom of both logics,

\[ A_1, \ldots, A_n \vdash A_1 \Rightarrow \ldots \Rightarrow A_n \Rightarrow B \]  

(2)

Applying modus ponens to (2), \( n \) times, we deduce \( B \).

\begin{proof}
Metatheorem 3.5. For any \( \Gamma \) and any formula \( A \), \( \Gamma \vdash E_n A \) iff \( \Gamma \vdash E A \).

By 3.2, if \( \Gamma \vdash E_n A \), then \( \Gamma \vdash E A \).

Conversely, since each rule \text{Inf1-Inf2} of E-logic is derived in En-logic (by 3.4), \( \Gamma \vdash E A \) implies \( \Gamma \vdash E_n A \).
\end{proof}

Metatheorem 3.6. [Generalization] For any \( \Gamma \) and any \( A \), if \( \Gamma \vdash A \) with a restriction, then \( \Gamma \vdash (\forall x)A \). The restriction is: There is a \( \Gamma \)-proof of \( A \), such that \( x \) does not occur free in any formula of \( \Gamma \) that was used in the proof.

Proof. By induction on the length of a \( \Gamma \)-proof that deduces \( A \) while respecting the restriction on \( x \). In view of 3.5 we work with En-logic (see [3]). Let \( B_1, B_2, \ldots, B_n \) be such a \( \Gamma \)-proof effected in En-logic, where \( B_n = A \). We show (by induction on \( n \)), that in the sequence

\[ (\forall x)B_1, (\forall x)B_2, \ldots, (\forall x)B_n \]  

(1)

every formula is a \( \Gamma \)-theorem of En-logic, hence also of E-logic.

Basis \( n = 1 \). If \( B_1 \in \Gamma \), then \( x \) is not free in \( B_1 \). By \text{Ax3}, and modus ponens, \( B_1 \vdash (\forall x)B_1 \), hence \( \Gamma \vdash (\forall x)B_1 \) by transitivity of \( \vdash \).

If \( B_1 \) is logical, then \( (\forall x)B_1 \) is also logical (partial generalization, see p.48). Thus, again, \( \Gamma \vdash (\forall x)B_1 \) (by 2.10 or 2.12).

Assume the claim for \( n \leq k \) (Induction Hypothesis, in short I.H.).

\footnote{Actually Enderton requires an unnecessarily strong condition: That \( x \) be not free in any formula in \( \Gamma \). He does so, presumably, because he offers a proof by induction on \( \Gamma \)-theorems. Induction on \( \Gamma \)-proofs, as we opted for here, is satisfied with a lesser restriction, imposed on only finitely many formulas of \( \Gamma \).}
We look at \( n = k + 1 \). If \( B_n \) is logical or in \( \Gamma \), we already have seen how to handle it. Suppose then that \( B_n \) is actually there because we had applied modus ponens, namely, \( B_i, B_i \Rightarrow B_n \vdash B_n \), and that \( B_i \Rightarrow B_n \) is the formula \( B_j \), where, \( i \) and \( j \) are each less than \( n \). Thus, by I.H.,

\[
\Gamma \vdash (\forall x)B_i
\]

and

\[
\Gamma \vdash (\forall x)(B_i \Rightarrow B_n)
\]

Applying modus ponens - via (2) and (3) - twice to the following instance of \( \text{Ax4} \), \( (\forall x)(B_i \Rightarrow B_n) \Rightarrow (\forall x)B_i \Rightarrow (\forall x)B_n \), we get \( \Gamma \vdash (\forall x)B_n \). \( \square \)

An important observation flows immediately from the proof of 3.6: The sequence (1) can be “padded” to be a \( \Gamma \)-proof without using any additional \( \Gamma \)-formulas beyond those used to derive \( A \) in the first place.

Corollary 3.7. [“Weak” Generalization] For any formula \( A \), if \( \vdash A \), then \( \vdash (\forall x)A \).

Proof. Take \( \Gamma = \emptyset \) above. \( \square \)

NB. We trivially have a derived rule specialization, sort of the converse of generalization. It says that if \( \Gamma \vdash (\forall x)A \), then \( \Gamma \vdash A[x := t] \). This follows from \( \text{Ax2} \) by modus ponens. In particular, we may take \( t = x \) to obtain \( \Gamma \vdash A \) (it is easy to check that \( x \) is always substitutable in \( x \), and that \( A[x := x] = A \)). We can thus also state: \( \vdash A \) iff \( \vdash (\forall x)A \).

“Weak” implies that there is a “strong” generalization rule as well. This is the “rule” \( \vdash (\forall x)A \). This rule is not derivable in E-logic. We will see why once we have proved the Deduction Theorem.

Metatheorem 3.8. [The Deduction Theorem] For any formulas \( A \) and \( B \) and set of formulas \( \Gamma \), if \( \Gamma, A \vdash B \), then \( \Gamma \vdash A \Rightarrow B \).

NB. \( \Gamma, A \) means \( \Gamma \cup \{A\} \). The converse of the metatheorem is also trivially true by modus ponens.

Proof. The proof is by induction on \( \Gamma, A \)-theorems and, once again, it is carried out in En-logic.

Basis. Let \( B \) be logical or nonlogical. Then \( \Gamma \vdash B \) (2.10). Since, \( B \models \text{Taut} \) \( A \Rightarrow B \), 3.4 and the transitivity of \( \vdash \) yield \( \Gamma \vdash A \Rightarrow B \).
If $B$ is the same string as $A$, then $A \Rightarrow B$ is a logical axiom (tautology), hence $\Gamma \vdash A \Rightarrow B$ (2.10).

*Modus ponens* induction step. Let $\Gamma, A \vdash C$, and $\Gamma, A \vdash C \Rightarrow B$. By I.H., $\Gamma \vdash A \Rightarrow C$ and $\Gamma \vdash A \Rightarrow C \Rightarrow B$. Since $A \Rightarrow C, A \Rightarrow C \Rightarrow B \models \text{Taut} A \Rightarrow B$, we have $\Gamma \vdash A \Rightarrow B$.

**RemarKs 3.9.**

(1) E-logic (equivalently, En-logic) does not support strong generalization $A \vdash (\forall x)A$. Otherwise, in particular, by 3.8,

$$\vdash x \approx 0 \Rightarrow (\forall x)x \approx 0$$

which, intuitively, is not “valid” (e.g., not true over the integers).

In some expositions the Deduction theorem is constrained by requiring that $A$ be closed ([9], [10]), or a complicated condition on its variables is given ([8]). Which version is right? Both are, each in its context. If all the primary rules of inference are propositional, as they are here, then the Deduction theorem is unconstrained because we do not have strong generalization. If, on the other hand, the rules of inference manipulate object variables via quantification (e.g., strong generalization, or other “stronger” rules are present – see [8], [9], [10]), then we must constrain the application of the Deduction theorem, lest we want to derive the invalid (i) above.

Here is the trade-off: Unconstrained Deduction Theorem goes with a constrained generalization (3.6). Unconstrained generalization ($A \vdash (\forall x)A$) goes with a constrained Deduction Theorem.

(2) This divergence of approach in choosing rules of inference has some additional repercussions. One has to be careful in defining the semantic counterpart of $\vdash$, namely, $\models$ (see section 6). One wants the two symbols to “track each other” faithfully (Gödel’s completeness theorem).

**MétaTheorem 3.10.** [The variant or dummy renaming metatheorem] For any formula $(\forall x)A$, if $z$ does not occur in it (i.e., is neither free nor bound), then $\vdash (\forall x)A \equiv (\forall z)A[x := z]$.

**Proof.** Since $z$ is substitutable in $x$ under the stated conditions, $A[x := z]$ is defined. Thus, by specialization (p.53), $(\forall x)A \vdash A[x := z]$. By 3.7, since $z$ is not free in $(\forall x)A$, we also have $(\forall x)A \vdash (\forall z)A[x := z]$. By 3.8, $\vdash (\forall x)A \Rightarrow (\forall z)A[x := z]$.

Noting that $x$ is not free in $(\forall z)A[x := z]$ and is substitutable in $z$
(indeed, $A[x := z][z := x] = A$) we can repeat the above argument to get $(\forall z)A[x := z] \vdash A$, hence (by 3.7) $(\forall z)A[x := z] \vdash (\forall x)A$, and, finally, \[
\vdash (\forall z)A[x := z] \Rightarrow (\forall x)A.\]

**Metatheorem 3.11. [Distributivity of $\forall$ over $\Rightarrow$]** Suppose we have a $\Gamma$-proof of $A \Rightarrow B$, where $x$ does not occur free in whatever nonlogical axioms were used. Then $\Gamma \vdash (\forall x)A \Rightarrow (\forall x)B$.

**Proof.** By 3.6, $\Gamma \vdash (\forall x)(A \Rightarrow B)$. The result follows by Ax4 and modus ponens.

**Corollary 3.12. [Distributivity of $\exists$ over $\Rightarrow$]** Suppose we have a $\Gamma$-proof of $A \Rightarrow B$, where $x$ does not occur free in whatever nonlogical axioms were used. Then $\Gamma \vdash (\exists x)A \Rightarrow (\exists x)B$.

**Proof.** A trivial exercise, using 3.11 and the definition of the “text” $(\exists x)A)$, namely, $(\neg((\forall x)(\neg A)))$.

**References**


**Acknowledgements.** I wish to thank my colleague Jonathan Ostroff for the many illuminating discussions we have had about Equational Logic.

Department of Computer Science  
York University  
Toronto, Ontario  
M3J 1P3  
Canada  
e-mail: gt@cs.yorku.ca