LOGIC AND APPLICATIONS OF LOGIC

Place and time: In M105 on Friday, Jan 5, at 10:30–12:00
Organizers: Esko Turunen (Tampere University of Technology)
Miikka Vilander (Tampere University of Technology)
Contact email: esko.turunen@tut.fi

The Logic of the Undefined

Antti Valmari *(Tampere University of Technology)*, antti.valmari@jyu.fi
Lauri Hella *(University of Tampere)*

Abstract. Undefined expressions are frequent in everyday mathematics. However, textbooks on logic do not tell what we do when we use them in reasoning, because they assume that function symbols are always defined. We solve this not very important but irritating problem.

1. The Problem. The real-valued roots of

$$7\sqrt{|x|-1} = x+9 \tag{1}$$

are -2, 5, and 26. How can this be expressed in familiar logical notation? Consider the following logical equivalence on real numbers:

$$7\sqrt{|x|-1} = x+9 \;\Leftrightarrow\; x=-2 \vee x=5 \vee x=26 \tag{2}$$

When $x=1$, both sides yield F (i.e., false), and when $x=5$, both sides yield T (i.e., true). However, when $x=0$, the right hand side yields F but the left hand side is undefined.

When simplifying arithmetic expressions, this problem can be avoided by assuming that each expression represents a function whose domain is $E \cap D$, where E is the explicitly mentioned domain and D is the set where the expression is defined. The function is of all variables occurring in the simplification chain. For instance, by $\tan x = \frac{\sin x}{\cos x}$ we mean that the equality holds when $\cos x \neq 0$. Unless 0 is explicitly ruled out, $\frac{x}{x} = 1$ is incorrect, because 0 is in the domain of 1 but not in the domain of $\frac{x}{x}$, so 1 and $\frac{x}{x}$ express different functions of x.

Direct application of this idea to (2) would consist of saying that we do not consider all reals, we only consider the set $\{x \in \mathbb{R} \mid x \leq -1 \vee x \geq 1\}$. Unfortunately, then we would not be saying that 0 is not a root of (1); instead, we would be saying that we do not consider the question whether 0 is its root. What is more, a natural first step towards solving it would be case analysis:

$$7\sqrt{|x|-1} = x+9 \;\Leftrightarrow$$

$$(x<0 \wedge 7\sqrt{-x-1} = x+9) \vee (x \geq 0 \wedge 7\sqrt{x-1} = x+9)$$

Here $x=5$ makes the right hand side yield T, but the left hand side is undefined because of $\sqrt{-x-1}$. So our attempt to avoid undefined situations failed.

One might argue that when $x=5$, $x<0$ yields F, so $x<0 \wedge \varphi$ yields F no matter what φ is, even if it is undefined. We agree. The goal of this study is to make this idea precise!

Undefined expressions can be dealt with using binary logic. In [2, p. 40–41] the convention has been adopted that a predicate with an undefined term as an argument does yield either F or T, but it is not known which one. This convention does not meet our need, however, because it does not say that (1) does not hold when $x = 0$; it says that it is not known whether it holds.

Indeed, our mission cannot be accomplished without any modifications to familiar laws. Because we do not want to accept 0 as a root of $\frac{1}{x} = \frac{1}{x}$, $\frac{1}{x} \geq 0$, and $\frac{1}{x} < 0$, we must agree that $\frac{1}{0} = \frac{1}{0}$, $\frac{1}{0} \geq 0$, and $\frac{1}{0} < 0$ do not yield T. So the law $t = t$ cannot remain universally valid if t may be undefined. With binary logic, also the law $t \geq u \Leftrightarrow \neg(t < u)$ would have to be modified.

Because $\frac{1}{0} < 0$ does not yield T, in binary logic it must yield F. So $\neg(\frac{1}{0} < 0)$ yields T. Furthermore, $\frac{1}{x} < 0 \Rightarrow x < 0$ is a correct implication in binary logic. Contraposition does not yield $x \geq 0 \Rightarrow \frac{1}{x} \geq 0$ but $x \geq 0 \Rightarrow \frac{1}{x} \geq 0 \vee x = 0$, because $\neg(\frac{1}{x} < 0)$ holds also when $\frac{1}{x}$ is undefined. We see that the use of binary logic would not save us from the need to treat undefined expressions in a special way.

We believe that instead of letting $\frac{1}{0} < 0$ yield F and $\neg(\frac{1}{0} < 0)$ yield T, it is more natural to employ a third truth value U (undefined) [1] and let both $\frac{1}{0} < 0$ and $\neg(\frac{1}{0} < 0)$ yield it [3]. This has many advantages, including that $t \geq u \Leftrightarrow \neg(t < u)$ need not be modified.

2. The Solution. We adopt the principle that if f is a function symbol and, for some value combination of variables, t is an undefined term, then $f(\ldots, t, \ldots)$ is undefined for the value combination in question. Let c be a defined constant term. When we say that $x = c$ is a root of $t(x) = u(x)$, we mean that both $t(c)$ and $u(c)$ are defined and they yield the same value.

Every predicate with an undefined term as an argument yields U. The negation of U is U. If we say that $\mathsf{F} < \mathsf{U} < \mathsf{T}$, then $P \wedge Q$ yields the minimum of the truth values of P and Q, and $P \vee Q$ yields the maximum. A similar claim holds on $\forall$ and $\exists$. We define $P \to Q$ as $\neg P \vee Q$, and $P \leftrightarrow Q$ as $(P \to Q) \wedge (Q \to P)$. This is similar to Kleene but different from Łukasiewicz who made $\mathsf{U} \to \mathsf{U}$ yield T.

This is important, because it implies that every truth function $\varphi(P)$ that can be constructed from these operators is *regular*, that is, either $\varphi(\mathsf{U})$ yields U or $\varphi(P)$ does not depend on P. Also every predicate that can be constructed from these operators, arithmetic operators, and the comparisons "=", "<", and so on, is regular: either $\varphi(t)$ does not depend on t, or yields U when t is undefined. For instance, the correctness of $\varphi(\frac{1}{1/x}) \Rightarrow x \neq 0 \wedge \varphi(x)$ relies on φ being regular (and not identically T).

If t is a term, then let $\lfloor t \rceil$ denote the claim that t is defined. It can be constructed recursively. If t is a constant or variable, then $\lfloor t \rceil$ is T. If t is $u + v$, then $\lfloor t \rceil$ is $\lfloor u \rceil \wedge \lfloor v \rceil$. If t is $\sqrt{u}$, then $\lfloor t \rceil$ is $u \geq 0 \wedge \lfloor u \rceil$. If t is $\frac{u}{v}$, then $\lfloor t \rceil$ is $v \neq 0 \wedge \lfloor u \rceil \wedge \lfloor v \rceil$, and so on. If φ is a predicate, then we define $\lfloor \varphi \rceil$ as follows. If φ is $t = u$, $t < u$, or so on, then $\lfloor \varphi \rceil$ is $\lfloor t \rceil \wedge \lfloor u \rceil$. $\lfloor \neg \varphi \rceil$ is $\lfloor \varphi \rceil$. $\lfloor \varphi \wedge \psi \rceil$ is $(\lfloor \varphi \rceil \wedge \neg \varphi) \vee (\lfloor \psi \rceil \wedge \neg \psi) \vee (\lfloor \varphi \rceil \wedge \lfloor \psi \rceil)$, and so on. Both $\lfloor t \rceil$ and $\lfloor \varphi \rceil$ never yield U. Please notice that $\lfloor \varphi \rceil$ is not a function of the truth value of φ but of its syntactic expression.

The operators "$\Rightarrow$" and "$\Leftrightarrow$" do not yield a truth value. Instead, they express reasoning steps [3]. We stick to the principle that $\varphi \Leftrightarrow \psi$ is valid if and only if both $\varphi \Rightarrow \psi$ and $\psi \Rightarrow \varphi$ are valid. The roots of $\frac{1}{x} = \frac{1}{x}$ are all reals but 0, so

$\frac{1}{x} = \frac{1}{x} \Leftrightarrow x \neq 0$ must be valid. In [3], it was shown that this implies $\mathsf{F} \Rightarrow \mathsf{U}$, $\mathsf{U} \Rightarrow \mathsf{F}$, $\mathsf{U} \Rightarrow \mathsf{U}$, and $\mathsf{U} \Rightarrow \mathsf{T}$. Furthermore, $\mathsf{T} \Rightarrow \mathsf{U}$ is not valid. In other words, the laws are what is obtained by adding $\mathsf{U} \Leftrightarrow \mathsf{F}$ to the laws of "$\Rightarrow$" and "$\Leftrightarrow$" in binary logic. So at this level, each undefined claim is treated as not holding.

This does not cause problems with negation, because "$\Rightarrow$" and "$\Leftrightarrow$" cannot be in the scope of "$\neg$", because they are not propositional operators but reasoning operators. Because of the regularity property, each undefined sub-claim either does not affect the truth value of the claim as a whole, or makes it U which is treated similarly to F.

3. Reasoning in the Presence of the Undefined. That φ holds can be expressed as $\varphi \Leftrightarrow \mathsf{T}$. On the other hand, $\varphi \Leftrightarrow \mathsf{F}$ and $\varphi \Leftrightarrow \mathsf{U}$ mean the same. So $\frac{1}{0} < 0 \Leftrightarrow \neg(\frac{1}{0} < 0) \Leftrightarrow \frac{1}{0} \geq 0 \Leftrightarrow \mathsf{F}$.

Any claim φ can be reduced to binary logic by replacing $\varphi \wedge \lfloor\varphi\rceil$ for it. When replacing a sub-claim within the scope of an odd number of negations, $\varphi \vee \neg\lfloor\varphi\rceil$ should be used instead, so that U is mapped to F at the level of the claim as a whole. Unfortunately, doing this extensively is clumsy, so also other methods of reasoning are needed.

Most laws of propositional logic are valid as such also in the presence of U. The most important exceptions are the following, shown here in their modified form:

$$\varphi \vee \neg\varphi \vee \neg\lfloor\varphi\rceil \;\Leftrightarrow\; \mathsf{T}$$

$$\varphi \vee \neg\varphi \;\Leftrightarrow\; \varphi \rightarrow \varphi \;\Leftrightarrow\; \varphi \leftrightarrow \varphi \;\Leftrightarrow\; \lfloor\varphi\rceil$$

The reasoning operators "$\Rightarrow$" and "$\Leftrightarrow$" are usually interpreted assuming a set of known or given facts, such as the axioms of real numbers, the case selector in case analysis, and the negation of the goal in a proof by contradiction. Let Γ denote this set. That is, when we write $\varphi \Rightarrow \psi$, we mean, roughly speaking, what would be written as $\frac{\Gamma \vdash \varphi}{\Gamma \vdash \psi}$ in natural deduction. Modus Ponens can now be rephrased as follows:

$\varphi \Rightarrow \psi$ is valid if and only if for every intrepretation
that satisfies Γ, $\varphi \wedge \lfloor\varphi\rceil \rightarrow \psi$ yields T.

On the right hand side, $\lfloor\psi\rceil$ is not needed, because $P \rightarrow \mathsf{U}$ yields T if and only if $P \rightarrow \mathsf{F}$ yields T.

The law of contraposition becomes: $\varphi \Rightarrow \psi$ if and only if $\neg\psi \vee \neg\lfloor\psi\rceil \Rightarrow \neg\varphi \vee \neg\lfloor\varphi\rceil$. Therefore, if $\varphi \Rightarrow \psi$, then $\neg\psi \Rightarrow \neg\varphi \vee \neg\lfloor\varphi\rceil$. Furthermore, if $\varphi \Rightarrow \psi$ and $\lfloor\psi\rceil \Rightarrow \lfloor\varphi\rceil$, then $\neg\psi \Rightarrow \neg\varphi$.

We already saw that $t = t$ is not universally valid. Instead, we have $\lfloor t\rceil \Rightarrow t = t$. Furthermore, if $\lfloor t\rceil \Leftrightarrow \lfloor u\rceil$, $\lfloor t\rceil \Rightarrow t = u$, and t and u are free for x in $\varphi(x)$, then $\varphi(t) \Leftrightarrow \varphi(u)$.

The $\forall$-introduction, $\exists$-elimination, and $\exists$-introduction laws are the same as in binary logic. $\forall$-elimination becomes:

If t is free for x in $\varphi(x)$, then $\lfloor t\rceil \wedge \forall x : \varphi(x) \Rightarrow \varphi(t)$.

The following laws facilitate substituting a sub-claim:

If $\varphi \Leftrightarrow \psi$ and $\lfloor\varphi\rceil \Leftrightarrow \lfloor\psi\rceil$, then $\xi(\varphi) \Leftrightarrow \xi(\psi)$.

If $\varphi \Rightarrow \psi$ and P is not in the scope of an odd number of negations in $\xi(P)$, then $\xi(\varphi) \Rightarrow \xi(\psi)$.

Some more laws can be found in [3].

4. It Works! Consider $t(x) = u(x) \Leftrightarrow x = c_1 \vee \cdots \vee x = c_n$, where $c_1, \ldots, c_n$ are terms that contain no variables and are not undefined. We use $c_1, \ldots, c_n$ also to denote the values that they evaluate to. We now show that this logical equivalence holds for every x if and only if the set of the roots of $t(x) = u(x)$ is $\{c_1, \ldots, c_n\}$.

If x is a root and $x \in \{c_1, \ldots, c_n\}$, then both sides yield T. If x is not a root and $x \notin \{c_1, \ldots, c_n\}$, then the left hand side yields F or U, and the right hand side yields F. In both cases, the equality holds. If x is a root but $x \notin \{c_1, \ldots, c_n\}$, then the left hand side yields T and the right hand side yields F. If x is not a root but $x \in \{c_1, \ldots, c_n\}$, then the left hand side yields F or U, and the right hand side yields T. In both cases, the equality fails.

[1] Kleene, S.C.: Introduction to Metamathematics, North-Holland, 1964

[2] Spivey, J.M.: Z Notation – A Reference Manual (2. ed.), Prentice Hall International Series in Computer Science, 1992

[3] Valmari, A., Hella, L.: The Logics Taught and Used at High Schools Are Not the Same. In: Karhumäki, J., Matiyasevich, Y., Saarela A. (eds.) Proceedings of the Fourth Russian Finnish Symposium on Discrete Mathematics, TUCS Lecture Notes No 26, May 2017, 172–186

Moniarvoinen ristiriitoja sietävä päättely

Esko Turunen ja Miikka Vilander *(Tampere University of Technology)*, `esko.turunen@tut.fi`

Abstract. Matemaattinen logiikka syntyi vaatimuksesta matematiikan perusteiden täsmällisyyteen ja ristiriidattomuuteen. Matemaattisen ideamaailman ulkopuolella insinööritieteissä, taloudessa ja muilla reaalimaailman alueilla tarvitsemme loogisia apuvälineitä, joissa haasteena on mallintaa epätäsmällisiä ja ristiriitaisia ilmiöitä.

Ensimmäiset moniarvologiikat kehitti puolalainen Jan Lukasiewicz jo 1920-luvulla; ne jäivät kuitenkin logiikan tutkimuksen marginaaliin viideksi vuosikymmeneksi, kunnes kiinnostus moniarvoiseen logiikkaan virisi. Merkittävä virstanpylväs oli tshekkiläisen Jan Pavelkan työ vuonna 1979; se oli lähtölaukaus matemaattisen sumean logiikan tutkimukselle [2]. Toinen merkittävä avaus oli amerikkalaisen Nuel Belnapin tutkimus parakonsistentista logiikasta vuonna 1977 [1]. Parakonsistentti logiikka sietää ristiriitoja; vaikka olisi perusteita hyväksyä totena sekä lause α että sen negaatio $\neg\alpha$, ei tästä kuitenkaan voida päätellä, että kaikki lauseet olisivat hyväksyttävissä, päinvastoin kuin klassisessa logiikassa on asianlaita.

Uutena käsitteenä parakonsistentissa logiikassa on evidenssi; jos lause on tosi, on meillä tietysti jotakin evidenssiä lauseen puolesta. Käänteinen ei kuitenkaan päde. Vaikka meillä olisi evidenssiä lauseen puolesta, tästä ei seuraa, että lause olisi tosi; voi olla myös evidenssiä lauseen negaation puolesta.

Vuonna 2010 julkaisimme paperin [3], jossa yhdistetään Pavelkan moniarvoinen logiikka ja Belnapin parakonsistentti logiikka, työnimeltään parakonsis-

tentti Pavelkan logiikka. Todistimme tämän logiikan täydellisyyslauseen ja eräitä perusominaisuuksia, joita esittelemme. Näytämme myös kahdella esimerkillä robotiikasta, miten kyseistä logiikkaa voidaan soveltaa.

[1] Belnap, N.D.: A useful four–valued logic. In Dunn, J.M. & Epstein. G. (eds.): Modern Uses of Multiple–Valued Logic. D. Reidel (1977), 5–37.

[2] Pavelka, J.: On fuzzy logic I, II, III. Zeitsch. f. Math. Logik **25**(1979) 45–52, 119–134, 447–464.

[3] Turunen, E., Öztürk, M. and Tsoukias, A.: Paraconsistent semantics for Pavelka style fuzzy sentential logic. Fuzzy Sets and Systems 161 **14**(2010), 1926–1940.