Decidability and Notation

Timm Lampert

Abstract

This paper first defines the concept of an iconic notation for a property P by a notation providing decision criteria for P. This definition distinguishes an iconic notation from a symbolic notation. The notion of an iconic proof is then defined by an algorithmic translation of a symbolic notation into an iconic notation. The defined concepts are illustrated by examples from mathematics and monadic logic. The definitions and examples then serve as a background for a discussion of the decision problem that asks for the possibility of an algorithmic translation of first-order formulas into a proper iconic notation for the whole realm of first-order logic.

Keywords: Iconic Logic · Diagrammatic Reasoning · Decision Problem · Proof Theory

1 Introduction

The difference between iconic and symbolic logic and, more generally, between iconic reasoning and reasoning according to inference rules has been drawn in different ways. One may, e.g., draw the distinction by distinguishing (i) the kind of mental activity that is prevailing (seeing as vs. thinking), (ii) the kind of expressions that are used (icons vs. symbols), or (iii) the way in which information is encoded (by features of expressions vs. merely by rules applied to expressions). As a consequence of (iii), the peculiarities of iconic reasoning are spelled out by the possibility of free rides (Shimojima (1996)), direct and indirect interpretation (Stenning (2000)), multiple readings (Shin (2002), Macbeth (2012)), operational iconicity (Stjernfelt (2011)), or observational advantages (Stapleton & Jamnik & Shimojima (2017)). The account presented in this paper rejects (i), specifies (ii) and is in line with (iii). In addition to the efforts to spell out approach (iii), the paper puts forth that the peculiarity of iconic expressions consists of providing decision criteria. These criteria are features of expressions that allow identifying the formal (logical or mathematical) properties in question. Expressions that provide these criteria are iconic in the sense that they identify properties by their own properties. In the following, I first

outline and illustrate this approach and then apply it to the decision problem in logic.

My purpose in relating iconicity to decidability is threefold: (α) to explain Peirce's claim that 'deduction consists in constructing an icon' (Peirce (1885, 182)), (β) to provide an account of iconicity that is independent of mental activity or intuition and that applies to algebraic notations, and (γ) to argue that the probative force of iconic reasoning in logic is stronger than hypothetical informal reasoning regarding its limits.

2 Iconic Notation

Peirce distinguishes icon and symbol by the way in which the signified is represented: by similarity or by convention. In the following, I confine the discussion to mathematics and logic. In mathematics and logic, it is neither clear whether expressions signify anything at all nor, if so, how to measure similarity. This is why I propose an alternative to Peirce's distinction of icon and symbol. I use the term 'expression' (rather than 'sign') as a neutral, general term of which icons and symbols are specific variants. An expression is not an icon nor a symbol per se. Instead, the classification depends on what purpose it satisfies.

According to Peirce, 'a great distinguishing property of the icon is that by the direct observation of it other truths [...] can be discovered than those which suffice to determine its construction' Peirce (1993-58, 2.279). This quote expresses the peculiarity of encoding information by icons (approach (iii) above). In the case of mathematics and logic, I propose explicating what is 'directly observed' if 'truths are discovered' in terms of decision criteria. A decision criterion is a property of an iconic expression that it has iff the formal property in question holds. A non-iconic, symbolic notation gives rise to the question whether a certain formal property holds and this question is answered by translating the initial symbolic expression to its iconic equivalent. Thus, I define the differentia specifica of an iconic expression with respect to its capability to provide decision criteria for mathematical or logical properties (including relations).

Definition 1 An expression Ψ is iconic with respect to property P iff it has a property that serves as a decision criterion for property P.

The definition of an iconic notation is likewise relative to the capability of deciding certain properties:

Definition 2 A notation is iconic with respect to property P iff it is a notation for iconic expressions deciding P.

In contrast, symbols of a symbolic notation do not provide decision criteria. Instead, they have to be reduced to iconic expressions of an iconic notation to determine mathematical or logical properties.

Let me illustrate the distinction between symbolic and iconic notation by two simple examples, one from arithmetic and one from logic. The next section 3 provides further examples.

Arithmetic expressions such as $3{\cdot}(2{+}1){-}6 > (4{-}1){\cdot}2{-}3$ and $3{\cdot}(2{+}2){-}6 > (4-2)\cdot 2-3$ do not provide a criterion to determine whether the property of being greater than ($>$) holds for the arithmetic terms. There is no general property common to all arithmetic terms that identifies the relation between the numbers expressed by the arithmetic terms. One must apply rules that reduce the arithmetic terms to iconic expressions to determine the relation. One way to do so is the reduction to unary notation. In this case, the first expression reduces to $||| > |||$, and the second, to $|||||| > |$. This makes it possible to determine the property in question by a property of the resulting stroke notation. In contrast to the symbolic notation of the initial arithmetic expressions (using decimal numbers and arithmetic operations), the unary notation is iconic with respect to the property of (in)equality, because it provides a general criterion to determine the (in)equality of arithmetic terms by correlating strokes.

In logic, the A,E,I,O-notation or, likewise, the modern notation of first-order logic does not provide criteria to determine the validity of Aristotelian syllogisms. However, Venn diagrams allow one to decide upon the logical property of validity by a property of the diagrams: A syllogism is valid iff no area that is not excluded by the premises is excluded by the conclusion. Table 1 illustrates how a valid syllogism (first syllogism) is distinguished from an invalid one (second syllogism) by Venn-diagrams (cf. the red areas in the diagrams).

3 Determining Impossibility

The decidability of formal properties in mathematics and logic typically concerns the question of what is possible. Prominent examples are the solvability of equations of a certain form (e.g., algebraic equations) by numbers of a certain kind (e.g., rational numbers or radicals) or the provability of a formula from certain axioms (e.g., the expression of Goldbach's conjecture or its negation within the language of Robinson Arithmetic (Q) from the axioms of Q). While the possibility to solve an equation or to prove a formula can be demonstrated by finite means in solving the equation or proving the formula, the impossibility to do so within a given calculus first and foremost means that applying the rules of the calculus to solve an equation or to prove a formula does not terminate. This situation raises the question of how it is possible to determine this impossibility.

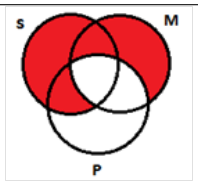
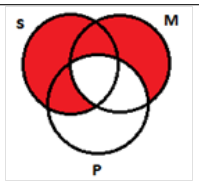
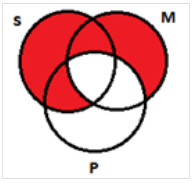
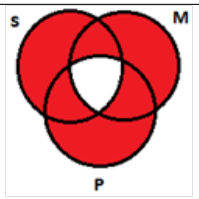
Symbolic Notation Syllogism	Iconic Notation Premises	 Premises & Conclusion
$\forall x(Sx \to Mx)$ $\underline{\forall x(Mx \to Px)}$ $\forall x(Sx \to Px)$		
$\forall x(Sx \to Mx)$ $\underline{\forall x(Mx \to Px)}$ $\forall x(Px \to Sx)$		

Table 1: Symbolic vs. Iconic Notation of Syllogisms

This section provides examples for deciding impossibilities in mathematics; section 6 takes up these examples to outline analogous strategies for determining provability in first-order logic.

Questions concerning infinity, such as the question of the non-terminating application of rules, are not undecidable per se. The impossibility to compute solutions of an equation within a calculus computing rational solutions might be shown by the possibility to compute irrational solutions of the equation within another calculus. Similarly, one might prove that a certain formula is not provable within a correct logical calculus by the method of identifying finite counter-models if available.

However, changing the calculus is not necessary to prove unsolvability or unprovability. Instead, the decidability of these properties depends on whether a finite pattern can be identified that shows that the application of the rules of a calculus will repeat ad infinitum. The Euclidean division algorithm may serve as a most simple example. Applied to the division of natural numbers, it yields decimal numbers. The application may result in a finite decimal number, such as in the case of $1 \div 4 = 0.25$, or in a periodic decimal number, such as in the case of $1 \div 3 = 0.\overline{3}$. The repeating pattern is induced by a repetition of dividing 10 by 3 with a remainder 1, and this repetition can serve as a criterion to decide that the rational number $\frac{1}{3}$ has no finite representation within the decimal notation. Thus, the property of a rational number to be representable by a finite decimal number is decidable within the Euclidean division algorithm. Therefore, the decimal notation is iconic with respect to this property even in the case where no finite representation of the initial rational number is available in the decimal notation.

The decimal notation, in contrast, is not iconic with respect to the question of whether equations of the form $a^2 = b$ with $b \in \mathbb{N}$ are solvable such that $a \in \mathbb{Q}$. Applying an algorithm to approximate the solutions within the decimal notation (e.g., by Dedekind cuts) may yield an 'irrational number', e.g., in the case of $b = 2$. In this case, approximating the solution within the decimal notation does not yield a repeating pattern. Thus, there is no criterion available that allows inferring that the computation will go on forever.

Yet, this 'irrationality' or lack of a finite pattern is a property of the notation, not of the number. Solving the equation $a^2 = 2$ by computing regular continued fractions does yield a periodic pattern: $1+\frac{1}{2+\frac{1}{2+\frac{1}{\cdots}}}$, or $[1;\overline{2}]$ in the short notation for regular continued fractions. Since all 'irrational' square roots have periodic regular continued fractions and all rational numbers have finite regular continued fractions, this notation is iconic with respect to the property of being a rational square root.

The notation of regular continued fractions, in turn, may seem irrational in the cases of other numbers such as π. However, moving on to the more general notation of irregular continued fractions, which also allows for partial denominators other than 1, yields again a repeating pattern such as $3+\frac{1^2}{6+\frac{3^2}{6+\frac{5^2}{6+\frac{7^2}{\cdots}}}}$.

Determining the solvability of an equation by numbers of a certain kind needs not reveal itself within a notation for numbers. One may also refer to a pattern within the derivation of equations in an attempt to solve the equation within a certain calculus. In this case, the iconic notation consists of sequences of equations rather than of sequences of digits. The classical proof of the irrationality of $\sqrt{2}$, for example, in fact proves that the attempt to solve $(\frac{a}{b})^2 = 2$ with $a, b \in \mathbb{N}$ inevitably runs in a loop, namely, to repeatedly solve an equation of form $\frac{c^2}{d^2} = 2$ with $c, d \in \mathbb{N}$. There is no need to reconstruct this proof as an informal proof of hypothetical reasoning using reductio ad absurdum. Instead, one may reconstruct the proof on the basis of a general algorithm to decide whether equations of the form $(\frac{a}{b})^2 = x$ with $x \in \mathbb{N}$ have a solution such that $a, b \in \mathbb{N}$ by either generating a solution or ending up with a repeating pattern. In contrast to the Euclidean division algorithm, this algorithm is not applied to numbers but to equations involving letters. Thus, it does not yield repeating digits but repeating equations using an algebraic notation. This notation is not iconic or symbolic per se. Instead, its iconicity depends on the property in question and the fact of whether it is possible to determine this property by means of a feature of the algebraic notation.

Peirce often emphasized the iconic aspect of an algebraic notation despite its abstract use of letters and operational symbols.[1] The decision procedure for

[1] Cf. the continuation of the quote above from Peirce (1993-58, 2.279):

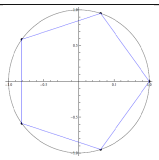
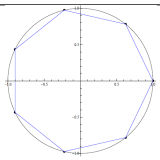
Symbolic Notation	Iconic Notation
	$x^5 - 1 = (-1 + x)(1 + x + x^2 + x^3 + x^{\underline{\underline{2 \cdot 2}}})$
	$x^7 - 1 = (-1 + x)(1 + x + x^2 + x^3 + x^4 + x^5 + x^{\underline{\underline{2 \cdot 3}}})$

Table 2: Symbolic vs. Iconic Notation of the Construction of Polygons

the possibility of constructing polygons in Euclidean geometry by the algebraic notation nicely illustrates this. It makes evident the fact that diagrams are not necessarily iconic, while the non-diagrammatic algebraic notation may serve as an iconic notation. It is not possible to decide from a diagram of a polygon whether it can be constructed with a straightedge and compass. However, the question becomes decidable within the algebraic notation by the factorization of polynoms over $\mathbb{Q}$. An n-gon is constructable iff the degree of the cyclotomic polynomial (cf. the first irreducible polynom with degree > 1 in table 2) in the factorization over $\mathbb{Q}$ of the n-th unit root is a power of 2 (cf. table 2, which underlines the degree in question, showing that the pentagon can be constructed, while the heptagon cannot).

The solution of decision problems of Euclidean geometry is an example in which one has to change the calculus (operations with a straightedge and compass vs. factorization of polynomials) and its language (geometrical figures vs. polynomials) by transforming constructions with a straightedge and compass into expressions of nested square roots (= constructible numbers). The operations of the Euclidean calculus are embedded in the algebraic calculus, meaning that more is decidable in the latter than in the former. This makes it possible to identify the limits of the Euclidean calculus.

In section 6, I draw an analogy between deciding logical properties in first-order logic and determining the solvability of polynomials by algebraic numbers of a certain kind (e.g., by radicals and, in the case of radicals, by constructible numbers). For this sake, it is important to note that the algorithm used to determine the solvability of unit roots by constructible numbers is, in fact, twofold:

> This capacity of revealing unexpected truth is precisely that wherein the utility of algebraical formulae consists, so that the iconic character is the prevailing one.

The first step is the factorization over $\mathbb{Q}$; the second step is the application of Galois theory, which computes possible types of solutions from the irreducible polynomials generated in step 1. By Galois theory, minimal polynomials can be generated from the cyclotomic polynomial, those minimal polynomials being solved by the real part of the first primitive unit root (= the real part of the point of the first edge in the first quadrant in the figures of table 2 with imaginary part > 0). The degree of these minimal polynomials is a power of two iff the degree of the cyclotomic polynomial is a power of two. Thus, the real part of the first primitive unit root is a constructible number iff the degree of the cyclotomic polynomial is a power of two. This determines the question of the possibility to construct a regular polygon because this question can be reduced to a question of whether the real part of the first primitive unit root is a constructible number.

For our later discussion, it is important to note that the application of Galois theory does not presuppose the whole realm of the complex numbers $\mathbb{C}$; the decision procedure does not make use of a complete linear factorization over $\mathbb{C}$. Instead, Galois theory investigates formal properties of polynomials and internal relations between their solutions and deduces the kind of their possible solutions hereof. It explains why properties of polynomials that are irreducible over $\mathbb{Q}$ allow determining the formal properties in question. Thus, it makes it possible to interpret a certain normal form notation (factorization over $\mathbb{Q}$) as an iconic notation, hereby making superfluous a complete factorization in linear factors and their localization within $\mathbb{C}$.

I argue in section 6 that one can likewise elaborate on a twofold algorithm in first-order logic: one that results in disjunctions of conjunctions of so-called primary (anti-prenex) formulas in first-order logic (FOLDNF), which corresponds to factorization into irreducible polynomials over $\mathbb{Q}$, and one that ensures that the resulting FOLDNFs are sufficiently decomposed to read off logical properties such as satisfiability. Prior to the enumeration of concrete models, the structure of possible models is thus investigated by identifying properties of decomposing normal forms and of their derivations.

4 Iconic Proofs

A decision problem is first and foremost the problem of a symbolic notation: this notation does not provide decision criteria. The solution of a decision problem specifies an algorithmic equivalence procedure transforming initial symbolic expressions ϕ into iconic expressions ψ. This procedure must preserve the property in question; it is an equivalence procedure with respect to this property. I define iconic proofs as results of decision procedures in terms of reducing initial expressions ϕ to iconic expressions ψ:

Definition 3 A proof is iconic with respect to property P iff it reduces initial expressions ϕ to iconic expressions ψ by means of an algorithmic equivalence procedure to determine P.

I already sketched four examples of iconic proofs: (i) reducing arithmetic expressions to stroke notation to decide (in)equality, (ii) converting the symbolic notation of syllogisms to Venn diagrams to decide logical (in)validity, (iii) translating decimal numbers into continued fractions to decide (ir)rationality, and (iv) embedding diagrams of polygons drawn by hand in factorized polynomials to decide the constructibility of polygons. Iconic proofs are essentially heterogeneous since they start from symbolic expressions and result in iconic expressions. This process makes a property that is only implicitly represented by rules governing the initial expression explicit by providing identity criteria for the property in question. It is this difference in representation that matters for iconic proofs.

Iconic proofs can be opposed to axiomatic proofs. While iconic proofs apply the method of analysis in terms of starting from what is in question and resulting in providing the evidence for the property in question, axiomatic proofs apply the method of synthesis in terms of starting with what is taken for granted and deriving a proposition to prove. The axioms of a negation complete axiomatic system that (dis)proves a proposition stating property P iff its equivalent iconic proof procedure (dis)proves that P holds encodes implicitly what the iconic procedure identifies explicitly. The axiom-schemas of so-called Baby-Arithmetic and an equivalent iconic proof procedure provide an example hereof. Baby-Arithmetic decides variable-free Δ_0-formulas ϕ by logically deducing either ϕ or $\neg\phi$ from the axiom-schemas, whereas the iconic procedure translates arithmetic terms ϕ to icons ψ of an unary notation.

One might question whether axiomatic proofs are able to do justice to the feature of deductive reasoning, namely, to show that a certain property necessarily holds, since all they show is that certain propositions (theorems) are derivable from other propositions (axioms) by certain rules. Iconic proofs, however, do justice to the necessity of deductive reasoning by making evident that the formal property in question necessarily holds due to nothing more than the possibility to represent it. Deductive reasoning is necessary not in some dubious metaphysical sense but due to the possibility to reduce the representation of properties to icons, which by their properties determine the property in question. I take this as the reason for Peirce's reduction of deductive reasoning to the construction and manipulation of icons, Peirce (1885, 182):

> The truth, however, appears to be that all deductive reasoning, even simple syllogism, involves an element of observation; namely,

> deduction consists in constructing an icon or diagram the relations of whose parts shall present a complete analogy with those of the parts of the object of reasoning, of experimenting upon this image in the imagination, and of observing the result so as to discover unnoticed and hidden relations among the parts.

Any iconic proof, however, is based on an equivalence procedure. That this procedure preserves the property in question cannot, in turn, be proven by an iconic proof. A correctness and termination proof of the algorithm, proving that all its transitions preserve the property in question, and, in fact, result in an iconic expression, alludes to the understanding of the rules laid down and is inevitably informal (non-algorithmic).[2] One has to understand transitions between different possibilities to express the same (with respect to a certain property in question). Meta-algorithmic proofs are based on analytic judgements concerning the concepts in question and the possibility to present them. We explicate our understanding of natural numbers and the operations applied to them if we convert $2 + 2$ via $|| + ||$ and $(||)(||)$ to $||||$; likewise, we explicate our understanding of a proposition and operations applied to them if we convert $\neg\neg P$ via $\neg\neg(T\text{-}P\text{-}F)$ and $T\text{-}F\text{-}T\text{-}P\text{-}F\text{-}T\text{-}F$ to $T\text{-}P\text{-}F$. The former shows that natural numbers can be used to count and that $2 + 2$ is equivalent to 4; the latter shows that propositions are capable of being true (T) and false (F) and that $\neg\neg P$ is equivalent to P.

Thus, iconic proofs serve to explicate mathematical or logical properties by an analytical, stepwise process that makes implicit knowledge explicit. Transformation of an arithmetical expression to stroke notation, translation of the ordinary, linear notation of syllogisms into Venn diagrams, writing square roots in regular continued fractions or encoding the construction of geometrical figures by algebraic formulas explicate our understanding of the initial symbolic expressions by an iconic proof procedure. Iconic proofs extend explicit, not implicit, knowledge. When they involve a stepwise, analytic process of converting symbols to icons, their understanding involves thinking; when they involve an identification of properties of resulting icons, their understanding involves aspect seeing (cf. account (i) in section 1).

5 Undecidability Proofs

While iconic proofs can be opposed to axiomatic proofs, meta-algorithmic proofs that prove (or explain why) a given decision procedure generating iconic proofs

[2] This is even true if (algorithmic or handmade) formalization is involved in correctness or termination proofs like in verification. This is so, because the correctness of the formalization has, in turn, to be proven and such a proof cannot be other than informal. Thus, the necessity of an informal proof is merely shifted. The same applies to undecidability proofs based on the algorithmic formalization of Turing machines, cf. section 5, p. 12.

achieves what it purports to achieve can be opposed to metamathematical undecidability proofs.

While meta-algorithmic proofs argue that a certain given procedure is, in fact, a correct and terminating decision procedure for a property in question, undecidability proofs argue that a certain decision problem cannot be solved. The informal reasoning of the former is based on explicating given rules and what follows from applying them, while the reasoning of the latter is not only informal but necessarily hypothetical since it is argued for the impossibility to define a decision procedure without considering a concrete procedure for a property in question. This kind of impossibility proof is opposed to the impossibility proofs sketched in section 3 that are based on decision procedures distinguishing what is possible and what is not possible by iconic proofs. Instead, undecidability proofs argue that an iconic procedure is impossible for certain properties. Thus, undecidability proofs purport to prove the limits of an iconic conception of proofs.

The all-important difference between undecidability proofs and correctness proofs of a given decision procedure is the application of the diagonal method. Meta-algorithmic correctness proofs prove that a certain property expressed by initial symbolic expressions holds iff a certain property of the final iconic expression holds; they intend to prove an equivalence. Undecidability proofs, in contrast, prove that this equivalence is impossible to obtain by whatever algorithm. The method used to prove this is the diagonal method; it reduces the assumption to absurdity that an equivalence between a property in question and some computable characteristic function exists.

Let me first illustrate the application of the diagonal method in the case of the undecidability proof of the halting problem. In contrast to undecidability proofs of first-order logic, this undecidability proof is independent of expressing properties in question by propositional functions within a logical symbolism. It is only this latter feature that I question. Thus, I neither question the reliability of the diagonal method in general nor the validity of undecidability proofs applying this method as long as they do not involve expressing formal properties by propositional functions.

I take for granted that any decision procedure is definable by a Turing machine (Turing's thesis) and that Turing machines are encoded by their numbers. In the case of the halting problem, the property in question is the property that an arbitrary Turing machine n halts if started with an arbitrary input k. The diagonal method reduces the possibility that this property is decidable by a hypothetically assumed universal Turing machine H to absurdity by considering the hypothetical case that the halting machine H determines this property for a machine X involving H. To apply the diagonal method, X must involve not

only H but additionally a machine that ensures self-application and a machine that inverses the result of H. Thus, let us define X by CHD, i.e., the composition of a copy machine C, the halting machine H and the dithering machine D, cf. Boolos & Burgess & Jeffrey (2007, 39f.). C copies the input k and returns the pair (k, k) to H, which returns 1 if the machine with the number k halts if started with input k and 2 if it does not halt if started with k. D is a dithering machine that does not halt if started with 1 and otherwise halts. From these definitions, it follows that a machine CHD that starts with its own number would not halt iff H returns the value 1 to D, which contradicts the assumption that H computes the halting function for any arbitrary machine. Since C and D are known to exist, the assumption that H is a universal machine computing the halting function even for a case such as CHD started with its own number cannot be true. Therefore, there is no Turing machine H such that, for any arbitrary Turing machine M, H returns 1 iff M halts if started with an arbitrary input k and 2 iff M does not halt if started with k.

This reasoning is informal and hypothetical because no machine H is defined in terms of a concrete Turing machine. The existence of such a machine H or, more precisely, the definability of H is only hypothetically assumed, and this assumption is reduced to absurdity by considering a specific case of self-application that contradicts the assumption of a universal machine that computes a property in question for every machine, including machines that include this very machine in question. Thus, this undecidability proof, in fact, is an undefinability proof proving that a certain function or property cannot be defined by a computable function.

There are many other applications of the diagonal method in mathematics, theoretical informatics or mathematical logic that prove the impossibility to define a property, a number or a function in a certain way without implying to express what is in question within the language of first-order logic. All these proofs may legitimately be analysed as proofs that limit the realm of decidability and, thus, of iconic proofs. In the following, I argue that the same does not apply to undecidability proofs of first-order logic (the so-called Church-Turing theorem). I confine my argument to a standard proof based on Turing machines (in short, 'Turing's proof', although I do not distinguish between Turing's original proof and modern variants of it). The question is whether Turing's proof proves that it is impossible to specify a procedure that determines the property of first-order validity (or, likewise, FOL-provability or other interdefinable properties such as FOL-satisfiability) by iconic proofs.

Turing's proof proves that the halting problem is decidable if first-order logic is decidable. He does so by specifying an algorithm that encodes Turing machines (including their inputs and configurations) by propositional functions of first-order logic. He then intends to prove a Lemma that states the equivalence

between a property of Turing machines in question, e.g., halting, and a property of the formula $Un(M)$ designed to encode it, e.g., provability of $Un(M)$. This part of his proof is nothing but an example of the informal reasoning for the correctness of a given algorithm. If this meta-algorithmic proof is valid, deciding the provability of first-order formulas $Un(M)$ could be done to solve the halting problem. The halting machine H would then consist of the composition $TFOL$ of a translation machine T that translates the code number of a Turing machine M and the number of its input to the respective logical formula $Un(M)$ and of a hypothetical assumed (undefined) decision machine FOL for first-order logic that returns 1 if $Un(M)$ is provable and 2 if not. However, since the halting problem is known to be unsolvable, the decision problem must be unsolvable given Turing's Lemma.

Yet, Turing's proof of his Lemma does not consider that the Lemma must also hold in the diagonal case in which $CTFOLD$ is started with its own number and FOL processes the formula $Un(CTFOLD)$. Instead, his proof of the direction 'If the formalization $Un(M)$ of machine M is provable, then M halts' rests on the general principle that any intended interpretation of a provable formula is true[3] without discussing the application of this principle to diagonal cases. Turing's Lemma as well as his principle are strong universal statements that do not allow for any exception if his proof shall work. His principle does not follow trivially from the correctness of first-order logic since the correctness of first-order logic implies that intended interpretations obey the semantic principles of first-order logic and, thus, are well-defined admissible interpretations. However, it is questionable whether the intended interpretation of $Un(CTFOLD)$ is admissible in the specific diagonal case in which $CTFOLD$ is started with its own number and FOL decides $Un(CTFOLD)$. Instead, provability and truth necessarily fall apart in this case according to the reasoning in the undecidability proof of the halting problem. This outcome could be used as a criterion for the inadmissibility of the intended interpretation instead as a criterion for the non-existence (or impossibility) of a machine FOL. One might intend to interpret $Un(CTFOLD)$ as a statement about the behaviour of $CTFOLD$ started with its own number, but it is this intention that is inconsistent, not the assumption of a well-defined machine FOL.

Turing is satisfied with having laid down rules how to interpret $Un(M)$ as

[3] In Turing's words, Turing (1936, 262):

> If we substitute any propositional function for function variables in a provable formula, we obtain a true proposition.

This is the first sentence of his proof of his Lemma 2, which concerns one direction of the equivalence stated in his Lemma. The whole proof of Lemma 2 consists of only two sentences. The second sentence simply applies Turing's general principle to $Un(M)$ and its intended interpretations. Modern variants do not essentially differ from Turing's original proof, cf., e.g., Boolos & Burgess & Jeffrey (2007, 130).

a statement about the behaviour of the formalized machine, hereby implicitly (without further argument) presuming that these intended interpretations are also admissible in the hypothetical diagonal case. However, one must consider that the method of diagonalization is designed to show that certain equivalences do not hold because it is impossible to express (or define) certain functions within a certain framework. The equivalence in question in Turing's proof does not concern the definability of a machine FOL but Turing's Lemma and, thus, the correctness of the algorithm of T in respect to expressing a property of Turing machines by a logical property of the resulting formula $Un(M)$. As a consequence, the question arises whether one can presume that the intended interpretations of the propositional functions involved in $Un(M)$ indeed do express the behaviour of $CTFOLD$ if started with its own number. Turing's informal reasoning for his Lemma does not prove this, since it simply presumes the admissibility of his interpretations (and, thus, assumes what is in question). Instead of inferring that first-order logic is undecidable, one might well reject the hypothetical assumption that the halting function (or, more specifically, a Turing-computable function for FOL-provability) is expressible by formulas $Un(M)$ generated by a Turing machine T because diagonal cases such as processing $Un(CTFOLD)$ when $CTFOLD$ is started with its own number rule out inferring that $CTFOLD$ started with its own number halts if FOL proves the formula $Un(CTFOLD)$.

Unlike the undecidability proof of the halting problem, the undecidability proof of first-order logic does not show that a Turing machine FOL is not definable. The mere assumption of this machine is not inconsistent; the inconsistency arises only if Turing's Lemma is conceded. One might consider a machine FOL and reject the possibility to apply its decisions to decide the halting problem in cases involving interpretations of a problematic self-application. From the viewpoint of an iconic logic, this is a most reasonable option. The notation of first-order logic is based on the concept of propositional functions, not on the concept of computable functions. The iconic conception of proof identifies computable properties or functions by properties of iconic expressions, not by propositional functions.[4] Thus, it may well be that first-order logic is not a suit-

[4] The idea that propositional functions are not suitable to express computable formal properties to a full extent can be traced back to Wittgenstein (1994, 4.126):

> Formal concepts [e.g., 'x is provable'] cannot, in fact, be represented by means of a [propositional] function, as concepts proper can. For their characteristics, formal properties, are not expressed by my means of functions. The expression of a formal property is a feature of certain symbols. So the sign for the characteristics of a formal concept is a distinctive feature of all symbols whose meaning fall under the concept.

Wittgenstein rejects expressing diagonal cases by propositional functions, cf. his distinction of operations and functions (Wittgenstein (1994, 5.251)) as well as his analysis of Russell's Paradox (Wittgenstein (1994, 3.33-3.333)). He also conjectured that first-order logic is de-

able iconic notation for properties of Turing machines or computable functions in general.[5]

Undecidability proofs for FOL are underdetermined. They do not limit the possibility to decide properties of logical formulas such as their provability by iconic proofs. Instead, undecidability proofs of FOL, in fact, prove that if FOL is decidable, the decision procedure cannot be used to solve certain undecidable problems such as the halting problem, because this application cannot be correct in the diagonal case.

6 Towards a Solution to the Decision Problem

Instead of limiting the possibility to specify an iconic decision procedure for FOL by hypothetical reasoning, one should rather attempt to spell out a decision procedure. The probative force of reasoning based on nothing but a given equivalence procedure within first-order logic that applies well-known and uncontroversial rules and culminates in uncontroversial criteria of logical properties is stronger than any hypothetical reasoning based on questionable intended interpretations of logical formulas related to properties that exceed logic.

In the following, I sketch some principles of a decision procedure I worked out for FOL and implemented in a program called the FOL-Decider. The details of this procedure and the proof of its correctness and termination are given elsewhere, cf. Lampert (2019b). This paper argues that the conceptual framework and the examples given in sections 2 to 4 provide a detailed heuristic for what one is looking for.

The algebraic tradition in mathematics was the predominant paradigma for the emergence of modern logic. This notably affected propositional logic and monadic first-order logic. Similar to factorization in algebra, disjunctive normal forms (DNFs) in propositional logic or Venn diagrams in monadic first-order

cidable and stuck to this conjecture throughout his life, even after he was confronted with Turing's proof, cf. Lampert (2019a).

[5] In the case of Church's proof, similar problems arise if one considers expressing recursive functions within a language based on first-order logic. The problem with Church's proof is not the proof that if FOL is decidable, Robinson Arithmetic Q is decidable. Instead, the problem is the underlying proof that Q is undecidable. This proof is based on the 'theorem' that recursive functions are expressible by Σ_1-formulas within the arithmetic language L_A of Q that is based on the language of first-order logic. The question is whether this so-called theorem also applies to expressing the diagonalization of a hypothetically assumed decision function for Q-provability in L_A. It may well be that provability in Q is definable by a recursive function but that this is not expressible within L_A due to the diagonal case. The informal proof of the so-called theorem that any recursive function is expressible in L_A is not conclusive for similar reasons that hold for the proof of Turing's Lemma; for more details, cf. Lampert (2019b).

logic can be seen as a decomposition of formulas to the effect that the formal properties in question are identified from the resulting decomposed structure. Linear factorization, e.g., identify zeros, DNFs and Venn diagrams can be used to identify models. The question is how possible is it to generalize this account to the whole realm of first-order logic.

It is not difficult to specify an algorithm to convert first-order formulas to a disjunction of conjunctions of anti-prenex formulas of FOL (FOLDNF).[6] This algorithm converts formulas to negation normal forms (NNFs), then applies PN-laws that are used to convert formulas into prenex normal forms in the opposite direction (including scope transformations for a most powerful miniscoping), and finally applies distributive laws to yield an FOLDNF. Furthermore, auxiliary laws can be applied to minimize and, thus, simplify FOLDNFs.

It is a curious fact of the history of logic that minimized FOLDNFs were not studied in detail. The discussion in mathematical logic is dominated by prenex normal forms; computational accounts of FOL are oriented by the method of resolution or tableaux; and iconic traditions such as Peirce's existential graphs or Frege's two-dimensional notation are neither taken from nor isomorphic to FOLDNFs. However, similar to DNFs of propositional logic, FOLDNFs are transparent normal forms for identifying conditions of truth (or the structure of models) of instances of initial formulas, which are converted to FOLDNFs: primary, anti-prenex formulas are the equivalents to atomic formulas in propositional logic that identify primitive structures from which sufficient conditions of truth can be composed by conjunction, which in turn are composed by disjunction to make up a necessary condition of truth (of instances of) the initial formula.

However, there is one important difference between minimized FOLDNFs and DNFs: The atomic propositions of a DNF are primitive in an absolute sense; they cannot be analysed any further and they are logically independent. Drawing the analogy to factorization in algebra, they correspond to linear factors. Primary formulas of FOLDNFs, however, are only 'irreducible' relative to the algorithm converting first-order formulas to FOLDNFs. This algorithm, e.g., does not exclude that a primary formula (or a conjunction of them) can be reduced further to an inconsistency and it does not result in factors that are necessarily independent. One might consider models in FOL as the most reasonable equivalent to linear factors (localized in $\mathbb{C}$). However, an equivalence procedure in terms of an algorithm resulting in enumerating alternative models (e.g., Herbrand instances) is impossible, since these models are infinite (in number as well as by themselves given an infinite domain). Thus, an iconic proof procedure cannot reduce formulas to models. Instead, this procedure

[6] The history of FOLDNFs as well as the algorithm used to yield minimized FOLDNFs and their use for identifying structures of models are elaborated in detail in Lampert (2017).

yields FOLDNFs and a relative notion of irreducible factors. In this respect, FOLDNFs are similar to a factorization of polynomials over $\mathbb{Q}$.

According to this analogy, an analogon to Galois theory is needed for FOL. Due to the acceptance of the Church-Turing theorem, however, this analogon has not been recognized as a relevant research problem. This may be one reason for the ignorance of FOLDNFs. An analogon to Galois theory in FOL amounts to a theory studying structures of models by investigating nothing but properties of FOLDNFs and their derivations to identify structures of models and to generate invariant kinds of models without starting from a denumerable or even non-denumerable infinite set of interpretations. This is an ambitious enterprise that has not yet been tackled in logic research.[7] In the following, I confine myself to the more specific decision problem for the satisfiability of single disjuncts of FOLDNFs.

In contrast to DNFs and Venn diagrams, one cannot directly read off satisfiability from a conjunction of primary formulas. Instead, an algorithm is needed that ensures that a conjunction of primary, anti-prenex formulas cannot further be reduced to a contradiction and, thus, indeed specifies identity criteria for models of an initial formula. This algorithm would amount to a decision procedure in first-order logic, since an FOLDNF is satisfiable iff one of its disjuncts is satisfiable.

Let me illustrate this by an example. The following, initial formula provides no criteria to read off its logical properties:

$$\neg\exists y_1\forall x_1\exists y_2(\neg Fy_1x_1 \vee (\neg Fy_2x_1 \wedge Fy_2y_1) \vee \neg\forall x_2\neg Fx_2x_2) \tag{1}$$

One cannot, e.g., identify from (1) whether the formula is satisfiable or, if so, what kind of structure its models have. With respect to its logical properties (e.g., satisfiability), the formula is a symbolic expression, and the conventional first-order notation is symbolic (not iconic). Converting (1) to an FOLDNF results in a conjunction of two primary (anti-prenex) formulas:

$$\forall x_1\exists y_1(Fx_1y_1 \wedge \forall x_3(Fx_3y_1 \vee \neg Fx_3x_1)) \wedge \forall x_2\neg Fx_2x_2 \tag{2}$$

(1) and the equivalent formula (2) have only infinite models. Thus, an algorithm going through finite interpretations will never yield a model, though

[7] One might consider Sheffer (1921) as an exception. However, Sheffer's elaboration of his project is sketchy and not yet studied in detail, cf. Urquhart (2012) for an overview and Langford (1926) for an interesting and fruitful application of Sheffer's approach. I am thankful to Victor Rodych for notifying Sheffer's project to me.

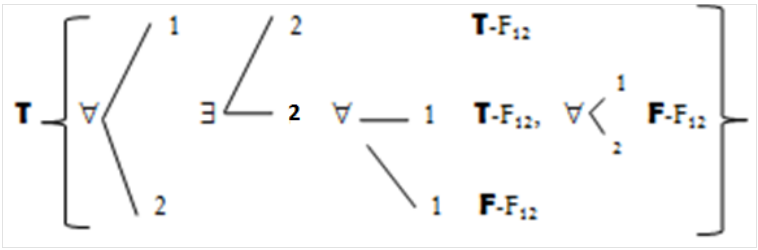


Figure 1: TF-diagram for (2)

the formulas are satisfiable. An algorithm going through single interpretations would be similar to an algorithm approximating irrational solutions of equations (e.g., $x^2 = 2$) within the decimal number notation: no pattern is identified that allows one to determine questions involving infinity.

However, there is no need to go through single finite interpretations if one can read off the structure (or pattern) of models from (2). How one may do so becomes manifest by converting (2) further to a two-dimensional notation (TF-diagram) that explicates the way each part of the FOLDNF (2) contributes to identify a structure of models, cf. figure 1.

TF-diagrams make manifest the fact that the role of bound variables is to identify positions of the propositional functions and that the role of conjunction and disjunction within the scope of quantifiers in primary formulas is to identify irreducible structures of models rather than to identify how a whole structure is composed of these irreducible structures (as conjunction and disjunction do outside the scope of quantifiers in FOLDNFs). Furthermore, TF-diagrams make it possible to read them from the outside to the inside by a mechanical reading algorithm that explicates how to paraphrase the (finite) structure of (possible infinite) models from the TF-diagram.[8] Thus, the TF-diagram of figure (1) can be paraphrased in terms of a description of the structure of models as follows:

- Instances of the initial formula (1) are true iff

  I1 All objects, the same in the first position of pairs of type 1 and the second position of pairs of type 3, combined with some object, the same in the second position of pairs of type 1 and in the second position of pairs of type 2, combined with all objects, distributed among those in the first position of pairs of type 2 and those in the first position of pairs of type 3, make up pairs of type 1 and 2 satisfying and pairs of type 3 not satisfying the dyadic propositional function F, and

[8] I have specified the algorithm to generate TF-diagrams and their mechanical paraphrase as well as their interpretation as an iconic notation in Lampert (2018).

I2 All objects, the same in the first and second position, make up pairs of type 4 not satisfying the dyadic propositional function F.

This reading demonstrates that the TF-diagram of figure (1) is an iconic expression that may provide criteria to identify the structure of models for the initial symbolic expression (1).

Humans might be able to imagine (and machines to generate successively) the combination of pairs satisfying the conditions described in I1 and I2 within an infinite domain and to consider a consistent composition hereof, thus interpreting the above paraphrase as a rule to generate infinite models such as the following (overlined pairs signify that the pairs do not satisfy $\Im(F)$):

Domain: $\{1, 2, 3, \ldots\}$.

$$\Im(F): \quad \begin{array}{lllll} \{ \ \overline{(1,1)}, & (1,2), & (1,3), & (1,4), & \ldots, \\ \overline{(2,1)}, & \overline{(2,2)}, & (2,3), & (2,4), & \ldots, \\ \overline{(3,1)}, & \overline{(3,2)}, & \overline{(3,3)}, & (3,4), & \ldots, \\ \vdots & \ldots & \ldots & \vdots & \ddots \quad \ldots \ \} \end{array}$$

The envisaged construction of a model, however, must not turn out to be inconsistent at some step of its generation. For a human or a machine to legitimately interpret TF-diagrams as descriptions of the structure of computable models that satisfy certain patterns in an infinite domain, it must be decidable in advance that disjuncts of FOLDNFs (such as (2)) and their TF-diagrams (such as figure (1)) do not imply a contradiction but, in fact, identify a satisfiable structure.

Let me illustrate the basic idea of a decision procedure satisfying this claim by analogy to algorithms such as the Euclidean division algorithm for determining whether a rational has a finite decimal representation or an analogous algorithm that decides whether square roots have a rational solution (cf. section 3). These algorithms result in either a finite solution or a repeating pattern that may serve as a criterion for the impossibility to derive a finite solution within a certain notation.

The same is possible in the case of disjuncts D_i of FOLDNFs if one considers an exhaustive search for proofs of minimal length within a correct and complete calculus (called the 'NNF-calculus') that uses the rule $\wedge I$ ($A \dashv\vdash A \wedge A$) as the decisive rule that increases complexity. Proofs of minimal length for D_i within this calculus are proofs that make use of $\wedge I$ to a minimal extent. A search for $\wedge$I-minimal proofs of D_i is exhaustive if all possible alternatives available to unify pairs of literals from D_i to derive contradictions by proofs of minimal

length are considered. The most important point is that one can define a schema for the exhaustive proof search for ∧I-minimal proofs within the NNF-calculus such that either a proof of contradiction is found or each branch of the proof tree results in a repeating derivation pattern, which serves as a criterion to break off the search on this path. In this case, the iconic notation for deciding upon satisfiability is a notation for sequences of formulas in a derivation tree of an exhaustive proof search within the NNF-calculus starting from disjuncts D_i of FOLDNFs. This notation keeps track of all applications of ∧I as well as the resulting unifications of literals and yields either an explicit contradiction on a proof path or patterns of repetition on all proof paths.

It may suffice to illustrate this briefly by considering one branch of the proof search for (2). (2) contains two and only two pairs of unifiable literals: $\{Fx_1y_1, \neg Fx_3x_1\}$ and $\{Fx_3y_1, \neg Fx_2x_2\}$.[9] To unify them, universal x variables must be replaced by existential y variables. However, to do so, one must consider optimized prenex normal forms such that a universal quantifier is in the scope of an existential quantifier if the universal variable is to be replaced by the existential variable. There is only one optimized prenex for a prenex normal form of (2), which I indicate by an ordered list of the bound variables: $\{x_1, y_1, x_2, x_3\}$. However, to unify the first unifiable pair of literals, x_1 must be replaced with y_1, which is not possible in a logically valid way with respect to the optimized prenex. In this case, the universal expression starting with $\forall x_1$ has to be 'multiplied' by ∧I to replace x_1 (which is now bound by two different universal quantifiers) in different scopes by different y variables, namely, by a new variable y_0 that is bound by a new existential quantifier preceding any optimized prenex by stipulation and by y_1. Thus, x_1 can replaced by y_0 in the literal Fx_1y_1 (stemming from one conjunct of the ∧I-application) and by y_1 in the literal Fx_3x_1 (stemming from another conjunct) of the ∧I-application). This, in turn, makes it necessary to replace x_3 with y_0 in the first unifiable pair $\{Fx_1y_1, \neg Fx_3x_1\}$ and with y_1 in the second unifiable pair $\{Fx_3y_1, \neg Fx_2x_2\}$. To identify variables bound by different quantifiers, it is convenient to rename variables after ∧I is applied so that, strictly speaking, substitutions with indices of depth 2 are to be considered after a first application of ∧I. The general idea of the search for ∧I-minimal proofs is to compute all necessary substitutions to unify unifiable pairs of literals to derive a contradiction. If a sequence of ∧I-applications results in a loop, roughly meaning that the same sort of pairs of literals are again to be unified by replacing the same sort of x variables with the same sort of y variables, the proof path on this branch terminates because no minimal proof of contradiction can be found on a path in which a ∧I-application

[9]Simple tests for unifiability exclude all other pairs of literals, including, e.g., $\{Fx_3y_1, \neg Fx_3x_1\}$, which is not unifiable due to the order of quantifiers that bind the variables occurring in the pair if one abstains from all other parts of the formula. The same holds for $\{Fx_1y_1, \neg Fx_2x_2\}$.

causes a repetition of itself. This 'Loop-Criterion' allows one to decide upon satisfiability by a finite pattern and, thus, makes going through an infinite number of interpretations superfluous.

The topic of this paper was neither to spell out in detail nor to prove the sketched decision procedure. Instead, this paper argued (i) that an iconic proof conception goes hand in hand with a critique of undecidability proofs of FOL, (ii) that it is possible to offer a clear idea for realizing an iconic proof conception in terms of a decision procedure for the whole realm of FOL and (iii) that this proof conception is in line with well-known decision procedures of the algebraic tradition. Decidability is a matter of reducing decision problems to an iconic notation and decidability of FOL is not a matter that can be settled by encoding decision problems within the propositional language of FOL.

References

Boolos, G.S., Burgess, J.P. & Jeffrey, R.C. (2007), Computability and Logic, fifth edition, Cambridge University Press, Cambridge.

Lampert, T. (2017), 'Minimizing disjunctive normal forms of pure first-order logic', The Logic Journal of the IGPL, 25(3), 325–347.

Lampert, T. (2018), 'Iconic Logic and Ideal Diagrams: The Wittgensteinian Approach', in Chapman, P., Stapleton, G., Moktefi, A., Perez-Kriz, S. & Bellucci, F. (eds), Diagrammatic Representation and Inference, Springer, Cham, 624–639.

Lampert, T., 'Wittgenstein's Conjecture', in Mras, G.M, Ritter, B. & Weingartner, P. (eds.) Proceedings of the 41st International Wittgenstein Symposium 2018, Philosophy of Logic and Mathematics, DeGruyter, 443–462.

Lampert, T., 'A Decision Procedure for Pure First-order Logic'.

Langford, C. 'Analytic completeness of sets of postulates', Proceedings of the London Mathematical Society 25, 115–142.

Macbeth, D. (2012), 'Diagrammatic reasoning in Frege's Begriffsschrift', Synthese 186(1), 289–314.

Peirce, C. (1981-58), Collected Papers, Harvard, Cambridge.

Peirce, C. (1885), 'On the Algebra of Logic: A Contribution to the Philosophy of Notation', American Journal of Mathematics 7(2), 180–196.

Sheffer, H.M. (1921), The general theory of notational relativity, Typewritten manuscript of 61pp. Available from the Harvard Widener Library.

Shimojima, A. (1996), 'Operational constrains in diagrammatic reasoning', in Barwise, J. & Allwein, G. (eds), Logical Reasoning with Diagrams, 27–48.

Shin, S-J. (2002), The Iconic Logic of Peirce's Graphs, MIT, Cambridge.

Stapleton, G., Jamnik, M., Shikojima, A. (2017), 'What Makes an Effective Representation of Information: A Formal Account of Observational Advantages', Journal of Logic, Language and Information 26(2), 143–177.

Stenning, K. (2000), 'Distinctions with Differences: Comparing Criteria for Distinguishing Diagrammatic from Sentential Systems', in Anderson, M., Cheng, P. & Haarslev, V. (eds.), Diagrams 2000, Springer, Berlin, 132–148.

Stjernfelt, F. (2011), 'On operational and optimal iconicity in Peirce's diagrammatology', Semiotica 186, 395–419.

Turing, A.: 'On Computable Numbers, with an Application to the Entscheidungsproblem', Proceedings of the London Mathematical Society 2(42), pp. 230–265.

Urquhart, A., 'Henry M. Sheffer and Notational Relativity', History and Philosophy of Logic 33(1), 33–47.

Wittgenstein, L. (1994), Tractatus Logico-philosophicus, Routledge, London.