Triggering a Copernican Shift in Logic through Sequenced Evaluations

Erik Thomsen
Principal Scientist
Charles River Analytics
Cambridge, MA USA

Abstract

We propose a novel view of the arguments and predicates of first-order logic in terms of differences in evaluation sequencing. We describe three main benefits of this novel view. (1) Higher order quantification is recast as asserting in relation to first-order logic assertions where the concepts used as predicates in one assertion are used as arguments in another (2) Wffs are no longer assumed to be evaluable propositions. Rather a truth evaluation process must be attempted to determine whether wff are or are not evaluable. And, all and only those that are determined to be evaluable are granted status as propositions which are guaranteed bi-valence. The distinction between evaluable and unevaluable wffs is then used to resolve the liar. (3) The relationships between logical propositions and their physical representations expands to include multiple internal physical representations (e.g., different memory locations) for the same internal logical proposition supporting applied research in database optimization as well as theoretical research in para-consistent (for example contraction-free) logic as well as non-verbal and non-linguistic representations which supports research in natural language understanding

Introduction

Over the past fifty years, the fabric of our society has been radically transformed by successful logic-based applications. In today's world, logic chips (i.e., CPUs) and the logic-grounded software that controls them support nearly all socio-economic infrastructure, from banking to defense. However, inference problems characteristic of widely used logic-grounded software technologies [Thomsen 2003b], and representation problems characteristic of logic-based research into natural language understanding (Berant, Chou, Frostig et al., 2013) suggest that there are parts of logic whose foundations are not yet completely understood.

At the center of logic's foundations is the characterization of the components of a proposition as *argument* and *predicate*. From Aristotle to Boole, Frege, and Russell, the consensus view of both components has been referential. That is to say arguments and predicates refer to entities in their respective domains in the world. Arguments refer to substances or objects; predicates refer to attributes. First order logic (FOL) was built with this referential view as a foundation.

This essay proposes a novel view of the notions of argument and predicate, treating them as matter of differences in evaluation sequencing or function and not as difference in domain (objects vs. attributes). This new view of arguments and predicates simplifies our logical understanding (e.g., eliminating paradoxes of impredicativity and the need for higher order quantification) and extends it to account for truth evaluation processes and transformations between different physical representations of the same logical proposition.

Towards that end, this essay is organized as follows:

1. Learning from empirical evidence about the parts of logic that do and do not work,
2. Using that evidence to revisit and reconsider unresolved foundational issues and
3. Proposing a functional (or function-role)- based view of the components of a proposition

Section I Empirical evidence

What is specified and appears to work perfectly

Classical first-order logic – hereafter 'FOL' – appears to work perfectly for truth functional applications when propositions are guaranteed to have a truth value. This after all is the basis for computer chips; and why we're celebrating the anniversary of Boole's invention of what he called 'signs and their laws'; what we call 'Boolean Logic'(Boole, 1854).

What is specified and works well but with certain restrictions

Relational Databases (RDBs) are the backbone upon which corporations and governments process information. Oracle and other RDB products are variably faithful implementations of the Relational Data Model first proposed in 1970 by Edgar Codd, whose seminal paper, *A Relational Model of Data for Large Shared Databanks* (Codd, 1970) relies explicitly on FOL as the basis for how information is queried (i.e., via use of the relational calculus). Codd's goal was to provide a logical description of information which could be queried without having to know where or how the data are physically represented. This principle became known as data independence. The success of RDBs shows that FOL works very well for querying large data sets. As successful as RDBs are, however, they have important shortcomings traceable to FOL and certain consensus assumptions about how FOL is to be used (highlighted with *underlines and italics*).

1. The first shortcoming is that the expressivity of RDB languages such as SQL was restricted to avoid the sorts of impredicativity paradoxes to which higher order logics are exposed. So for example, given a propositional function parent(x), the Relational Calculus provides no method for defining all x (i.e., an ancestor definition)[(Aho & Ullman, 1979)

2. RDBs enforce a distinction between data definition languages (DDLs) that *define* models and data manipulation languages (DMLs) that *query* models already built. This may be traceable to the *FOL emphasis on creating expressions that get evaluated against a pre-existing closed world*. The distinction between DDLs and DMLs led to artificial divisions between those who build models and those who analyze them. This in turn had the negative impact of making it hard to design an RDB that can modify its own model in the face of changing information -- which led to the recent revolt against pre-defined schemas, most conspicuously in the rise of NOSQL and similar minimalist products, frequently referred to by means of the misleading term 'schema-less databases' (Maluf & Tran, 2003), a term that is at best useful only for marketing purposes.

3. The consensus view is that *FOL statements are to be evaluated against a predefined world where world facts are known in advance*. Truth, for RDBs, is typically assumed to reside in the model. When a new assertion enters the system that disagrees with prior knowledge in the RDB's model, the new assertion is assumed to be wrong. (as contrasted against the interactive view of data and models coming from the domain of statistical process control (Shewhart, 1925; Thomsen, 2003a), contributed to the artificial divide between data and knowledge where new assertions are data and the background model holds the static knowledge.

4. Codd's dream of a universal query language was thwarted by the fact that RDBs did not initially pay attention to typing systems. Codd's original paper acknowledged the existence of data types or domains as pools of values (Codd 1970). But otherwise provided no guidance as to what was required of domains.[1] This lack of attention to domains can be traced back to *the lack of attention paid to data types (e.g., Integers, Rationals, Time, Space) within the predicate calculus*. Early RDB vendors provided minimal data type support: principally categoricals and integers. As a result, there was a flood of research in the 1970s into the 1980s on richer data typing systems for RDBs, including a paper by Codd called 'Extending the Relational Data Model to capture more meaning' (Codd, 1979)[2]. By the late 1980s, the RDB's lack of data typing support created a balkanization of technologies and standards. For example, both multidimensional and object databases were created because of the lack of sufficient data typing support in RDBs(Thomsen 2003). And the result, for application developers, was akin to Babel after the fall: a morass of competing standards for overlapping representational systems (e.g., SQL-based standards, OLAP standards, Object standards (Melton, 2003; Vassiliadis & Sellis, 1999)).

5. *Shortcoming #4 above combined with* the tacit assumption that logic is distinct from computing (an aspect of the logicist enterprise unrealized), led to a multitude of disconnected inference technologies - some logic-oriented, some mathematics-oriented, all of which needed to get manually woven together in the building of large scale information systems (LSIS). For example, even modestly simple

[1] Since 1995 Chris Date, a close colleague of Codd's and now the acknowledged standard bearer for the Relational Model has been advocating the need for the Relational Model to adopt a rich typing system. See for example Date, C. J.& Darwen, H. (2000). Foundations for Future Database Systems. Boston: Addison Wesley.
[2] See also Smith and Smith, (Smith & Smith, 1977)…..

real world application expressions may require a blend of mathematical computation and logical inference such as in the example below which is selecting all cases where an IF-THEN assertion evaluated to False (i.e., sales increased but bonuses did not increase accordingly).

```
Select all regions where
        IF Sales this period > Sales last period
        THEN increase Bonus by at least ½ times the % change in sales
Is false
```

Mathematical computation is required to compare the sales values. Logic is required to connect the IF statement with the THEN statement and mathematics is required to calculate the bonus if applicable and finally logic is used to quantify over the collection of such IF THEN expressions picking out those where the condition was true but the consequent false. Ideally one should be able to specify and view inferences that blend logical and mathematical expressions as shown above, rather than relegate each component to a separate kind of inference engine as is currently the case. For example, in LSIS, it is common to see inference methods physically realized in an RDB as column formulas, and as triggers and as stored procedures, plus a multidimensional database for aggregations and matrix operations plus a theorem prover for logical deductions plus a rules engine for linking the different databases.

What is specified and does not work

Rules of inference that apply to collections of wffs when some of the wffs in the collection are unevaluable . The World Bank maintains one of the world's largest collections of time series data consisting of tens of thousands of economic, environmental and social time series data (i.e., wff) for its over two hundred member countries. During the 1990s, it was this author's job to define the logic by which that data was aggregated and fed to downstream inference processes. The problem was that many of the country- and time-specific assertions (i.e., wff) to be aggregated were not evaluable for some reason; either because they were missing (e.g., caloric intake data or a predicate constraint on caloric intake data such as 'Is < 2000 calories per day' is missing for a sub-Saharan country for a particular year) or not applicable (e.g., agricultural wages for a city state like Hong Kong). And the fact that the syntactic representation of logically missing predicates is physically indistinguishable from logically inapplicable predicates is indistinguishable from logical zeros. Treating all the input wff as evaluable because they look like zeros without verifying evaluability would have meant that all wff regardless of whether missing, present or not applicable would get treated the same way by downstream analyses (e.g., a wff whose predicate says 'zero' might trigger a process based on it being true that some value = 0). But this can create wrong outputs for operations that range over collections of wffsuch as aggregations.

Regardless of circumstances, when a physically empty predicate slot is logically missing *some default must be substituted into the open variable prior to aggregation* In contrast, averaging wff when the empty predicate slot is logically not applicable creates a wrong denominator. And so in this case, the inapplicable variable needs to be eliminated. To preserve logical correctness in downstream analyses, it was necessary to find and differentially treat all physically empty logically missing and physically empty logically-inapplicable wff before downstream predicates were evaluated. This was accomplished using a computable definition of evaluability combined with substitution and elimination rules, which though not standard logical practice are still in use twenty years later at the

World Bank because they work. Deductive operations like summing should never produce empirical mistakes. It would appear that something in logic may be missing or wrong.

What is underspecified

In addition to LSIS' useful empirical feedback on the efficiency, consistency and/or completeness of logic empirical feedback also comes from research into natural language understanding where logic is used as a canonical form to represent the meaning extracted from natural language (Domingos, Kok, Poon et al., 2006). Three common themes emerge from current research directions as diverse as the mapping of lexical analyses into first order logic , the discovery of latent canonical forms in natural language (Turmo, Ageno, Català et al., 2006)the joint discovery of semantic parses for natural language and images (Evangelopoulos, Zlatintsi, Potamianos et al., 2013) ,and the mapping of natural language sentences into an underspecified logical form as an intermediate step to producing a fully specified logical form QUA canonical meaning(Berant & Liang, 2014)

To illustrate the first theme, consider the example below

```
The old man who lives by the sea is a good surfer.
```

A fully specified canonical representation might look as follows.

```
good-surfer(℩x(man(x) & old(x) & lives-by-the-sea(x))
```

where ℩ is meant to be the upside down iota operator for definite descriptions

In natural language understanding, the existence of the argument (e.g., ℩x(man(x) & old(x) & lives-by-the-sea(x)) is not presupposed. Nor is the appropriate collection of predicates needed to individuate x. Furthermore, current linguistics research strongly suggests that the argument evaluates before the predicate [Stalnaker 2002]. This can be illustrated by looking at discourse between individuals. Considering again the sentence above, if this sentence were uttered by some person A to a person B, and person A had in mind some person C in making the statement, and person B knows many men of different ages who live in different places some of whom are good surfers, the selection of qualifiers by person A is intended to enable person B to narrow down which man to just one person so that person B can evaluate the predicate 'is a good surfer' of the argument 'the old man who lives by the sea'. But the qualifiers selected by person A (man, old, lives by the sea) might not work. If, for example, it resolved to zero, person B might respond "What old man who lives by the sea?" or "Old man Jackson doesn't live by the sea. Is that who you're referring to? He lives in town now." And if it resolved to more than one, person B might respond "Which old man who lives by the sea? The nice guy or the grouch?" More likely still, based on the grammatical construction, person B would first evaluate 'old man' and only if 'old man' evaluated to one or more candidates would person B then proceed to evaluate 'who lives by the sea'.

The empirical feedback for logic would appear to be that 1) the unambiguous referencing of an argument within a proposition may depend on the receiver's knowledge, 2) allowances need to be made for refining argument specification so that argument reference succeeds and 3) the argument must finish evaluating before the predicate begins to evaluate.

The second theme is that regardless of whether (and independent of flavor of) spatio-temporal event semantics, logic needs to bind with such semantics before being used to connect with (or match to) the sense data representative of external phenomena such as words, tactile patterns, sounds and pictures. In other words, even where researchers are talking about using “domain-independent underspecified logical forms” (Kwiatkowski, Choi, Artzi et al., 2013)the expressions matched against the page are richly typed (e.g., persons as special biological objects, visits as special events, annual as a period of time, dates as time references, new York as a place).

The third theme is that as soon as representations cover different languages or sensory pathways, the link between logical and physical representations, becomes paramount. The distinct physical representations (e.g., tactile, visual olfactory, textual-English, textual-French) each of which may be thought of as a distinct wff need to be treated as distinct physical representations of a common collection of logical propositions (i.e. internal understanding).

Section II Re-visiting relevant foundational issues

The main logical issues that appear to be problematic based on the empirical evidence presented above and that are reflected in foundational discussions can be grouped into the following three categories of questions which will be addressed in this section.

1. Are there sequential constraints on the process of truth evaluation of logical expressions? If so, what are they?
2. Is the evaluation state of a logical expression of logical concern? If yes, are there assumptions under which it can be ignored?
3. What is the natural relationship between logical expressions and physical representations?

Summary characterization of consensus logic

The modern classical consensus has been reasonably well formed since the 1930s[3]. For example:

1. Frege’s characterization of a proposition[4] in terms of f(a) where ‘a’ denoted an N-adic argument and ‘f’ denoted the predicate,
2. Russell’s theory of descriptions linking natural language to the predicate calculus via systematic descriptions of arguments,
3. Mill’s, Frege’s and Russell’s distinction between denotation or reference and meaning or sense, and its application to substitution rules, and
4. Pierce and Wittgenstein’s classification of truth functions

have continued in use until this day[5]

[3] There was no specific day. But the 1927 second edition of Principia Mathematica with its inclusion of Russell’s logical atomism that he took from Wittgenstein is a good marker. In any event the key notions of well formedness, rules of equivalence and sentential connectives were reasonably stable by then.

[4] Whether in conjunction with an existential quantifier (e.g., “There exists an X such that f(x)”), or a universal quantifier, (e.g., “For all ‘x’ f(x)”), Hilbert, Peano, Russell, Carnap, Quine and Putnam to name but a few, all used a symbology based on the notion of a predicate ‘f’ and N-adic argument ‘(a)’ to formally denote and reason about propositions.

[5] Over the years alternate terminology has been introduced and used. As far back as the turn of last century Sheffer used the term ‘propositional forms’ instead of propositional function. Quine spoke of ‘sentential functions’ instead of propositional functions. Some , like Lee used R(X,Y,Z) instead of F(X,Y,Z) and spoke of relations instead of predicates. See for example “Symbolic Logic” Harold Lee Random House 1961

However, that consensus covered over acknowledged foundational problems. For example, both Frege and Russell were aware of problems of identity and substitution in the predicate calculus (haddock, 2012). Russell discovered and passed on to Frege the problems with the predication of all members of a class or set, especially when there were dependencies between the act of determining that an object belonged in the set, and the set itself. The Wittgenstein of the Tractatus also voiced concerns about the referential interpretation of the components of a proposition f and a. [TLP 3.333 ref]. Foundational concerns notwithstanding, with the passage of time, consensus logic didn't evolve so much as solidify into formalisms (Russell, 1995) Foundational holes were patched over by clever restrictions and technical gymnastics (Bargiela & Pedrycz, 2006)..

Sequential constraints on the process of truth evaluation

Given the impact of Boole's 'The Laws of thought', we start there for a closer look at whether sequence needs to play a role in the evaluation of wff. In chapter II on *signs and their laws* Boole treats multi-part descriptors as set operations. And in the case of multi-term adjectives or verbs associated with an object, as intersections. Thus he writes " If x alone stands for white things and y for sheep, let xy stand for white sheep" [page 28]. This is normally understood in an extensional sense, (often with the help of a Venn diagram) as the intersection of the set of white things and the set of sheep things. And the intersection is indifferent to order of operation: xy = yx. In chapter IV *division of propositions*, he treats propositions with the same methods. He distinguishes subjects and predicates. But there is no distinction in processing. For example Boole writes *To say 'Snow is white'* is for the ends of logic equivalent to saying that 'Snow is a white thing' which is the commutatively expressible intersection of snow things and white things. There is no mention of evaluation sequence; that the subject (or argument) needs to be successfully evaluated before the process of evaluating the predicate can begin. Is sequence or order of operation significant? Consider the sentence "The old man who lives by the sea is a surfer." And consider the following series of purportedly negating assertions PNA each with its own label for subsequent discussion.

PNA1: No, the old man who lives by the sea is not a surfer
PNA2: No, the old man who lives by the sea is a writer
PNA3: No, the young man who lives by the sea is a surfer
PNA4: No, the old man who lives in town is a surfer
PNA5: No, the young woman who lives by the sea is a surfer
PNA6: No, the young woman who works in the city is a surfer
PNA7: No, the dog who ran up the mountain is a loud barker

PNA1 represents the case where the argument and predicate type are the same between the original assertion and its purported negation. It suggests a non commutative process where arguments must match before predicates are compared
PNA2 represents the case where the argument is the same between the original assertion and its purported negation. But the predicate type is not. It too suggests a non commutative process where arguments must match before predicates are compared
PNA3-6 represent the case where the predicate is the same between the original assertion and its purported negation, where the argument is not the same between the original assertion and its purported negation, but the argument type is. It suggests one of two things. Either a commutative process with the constraint that either arguments or predicates must match before the complement (i.e. predicates or arguments) are compared. Or as a recasting of the original assertion and subsequent agreement relative to the recast assertion. For example, given the original assertion one might instead

of PNA3 respond with the following: "If you mean by 'the old man who lives by the sea' that guy with the orange surfboard, he's actually pretty young; but yes he is a surfer".
PNA7 represents the case where neither the argument type nor predicate type is the same across the original assertion and its purported negation. It suggests a commutative process between argument and predicate matching with no constraints. And so any assertion whatsoever could fit into this category.

Any treatment of a proposition as the output of a collection of commutative operations erases the distinction between non-matching arguments and non-matching predicates. With each increase in the number of terms that can vary between an assertion and a negation the number of propositions that would count as a negation of a given proposition grows. For example, under the PNA7 interpretation of negation, in a collection of propositions P, Q, R… any proposition other than P could be considered a negation of P. So too could any proposition other than Q be considered as a negation of Q and so on. Therefore the negation of the negation of P would not necessarily equal P. So bivalence or any finite valence for that matter would have to be abandoned. Only PNA 1 and 2 which treat a proposition as the output of a non-commutative process that matches on argument type and value before comparing predicates would appear to support bivalence.

Evaluability constraints on the process of truth evaluation
If arguments must evaluate successfully before predicates can be evaluated at all and bivalence only applies to evaluated (or assumed evaluated) wff it should be possible to resolve the Liar paradox by demonstrating that 'this sentence is false' though a wff is not evaluable and so never converts into an evaluated proposition. Consider Russell's classic version of the Liar, 'This sentence is false'. It passes sentence level well formedness and reads verb phrase(noun phrase). But is it a proposition?. Can it evaluate? Although the classical view of the problem was that it stems from the self referential nature of the sentence and so Russell's solution was to create a hierarchy of sentence types, the problem as shown by Barwise and Echtemendy in their 1989 book "The Liar Paradox" has nothing to do with self reference, per se. For example, "This sentence has five words." is no less self referential than "This sentence is false". But it is true, not paradoxical. (As, "This sentence has six words.", is false; not paradoxical.)

Barwise and Echtemendy made the case that the flip flopping truth values of the Liar reflected flip flopping contexts within which the terms 'this sentence' were to be interpreted. And within each interpreted context, the meaning and truth value of 'this sentence is false' is stable. Their insightful analysis of hypersets and of Russellian versus Austinian semantics notwithstanding, Barwise and Echtemendy fell into the same trap that caught Frege and Russell before. Namely the trap of thinking that just because a sentence is grammatically well formed that it denoted one or more evaluable propositions.

We hypothesize the consensus view is in error when it jumps the gun in assigning a propositional variable and one or more truth values to the paradoxical sentence. The reason the consensus approach is problematical is that at no point in the substitution process does the sentence ever denote an evaluable truth value- bearing proposition. Since that possibility was brought up by Barwise and Echtemendy[6] but then rejected for lack of evidence, we now meet their challenge to "offer an explanation for the failure of

[6] "The Liar Paradox" pps 13, 14

the Liar sentence to express a claim one that is either true or false"[7], thereby demonstrating that evaluability is the fundamental issue. Consider the process of trying to convert the wff 'This sentence is false' into an evaluable form[8].

Step 1: Receive string; parse into words

- 'This sentence is false'

Step 2: Assign each word a role as argument or predicate

- Is false(this sentence)

Step 3: Test for the possible presence of variables for which values need to be substituted

- Discover that the two words 'this sentence' in the argument could constitute a variable.

Step 4: Check whether the candidate variables discovered in step 3 are the output of a prior substitution process and check whether the words match the form required for the predicate operator. There are three cases to process the results:

Case 1: If the words are not the result of a prior substitution process then substitute for the variables the string that is what the variables refer to and return with that string to step one

Case 2: If the words are the result of a prior substitution process and the words match the form required for the predicate function then run/execute the predicate function on the argument

Case 3: If the words are the result of a prior substitution process but they do not match the form required for the predicate function then substitute for the variables the string that is what the variables refer to and return with that string to step one

- Given the output of Step 3 above, step four triggers case 1. And so

Repeating Step 1: Receive string; parse into words

- is false ('This sentence is false')

Repeating step 2: Assign each word a role as argument or predicate

- is false ('is false(this sentence)')

Repeating Step 3: Test for the possible presence of variables

- Discover that the two words 'this sentence' in the argument could constitute a variable.

Repeating Step 4: Check whether the candidate variables discovered in step 3 are the output of a prior substitution process and check whether the words match the form required for the predicate function .

- Discover the candidate variables are the output of a prior substitution process and the words do not match the form required for the predicate function

Step 4 now triggers Case 3: If the words are the result of a prior substitution process and the words do not match the form required for the predicate function then substitute for the variable 'this sentence' the string that is what the variable refers to and return with that string to step one

- Return to step 1

Repeating Step 1: Receive string; parse into words

- is false(is false ('This sentence is false'))

The process will loop indefinitely hitting Case 3 in step four each time. (Contrast with the sentence 'this sentence has five words' which will resolve in its second pass through by hitting Case 2 in step 4.) So when the process of trying to convert a wff into an evaluable proposition is examined in detail, the classically paradoxical sentence 'this sentence is false' is seen as an infinite loop, not of contradictory true/false assertions but of attempts to substitute an assertion for a variable referring to what is supposed to be, but in fact is not, an assertion. The substitution process never comes to a halt; no assertion is ever found (unless the term 'this sentence' points somewhere else). The number of

[7] Ibid page 14

[8] For a more thorough treatment see Thomsen 2012

nested variable substitutions grows without end. The process of finding an argument that is evaluable relative to the predicate never terminates. So the predicate evaluation never even starts. And no truth value is ever produced. The sentence is never converted into a proposition; and for entirely understandable reasons.

Section III. Suggestions for logic

There are three major suggestions for the future of logic numbered 1-3 below and bolded. Following the first are a subordinately numbered series of related suggestions.

1. Truth evaluation needs to have a sequenced evaluation process. Any wff with an f(a) structure, denotes a bi-valent proposition IFF the predicate *f* successfully evaluates of its argument *a* AND *a* was successfully evaluated prior to the start of the evaluation of the predicate. Argument evaluation must come first. If argument evaluation fails, the predicate is not evaluated. And the wff is deemed unevaluable for reasons of an unevaluable argument. Note that, consistent with Aristotle's view, the only way to know the division of argument and predicate in a proposition is to know the question that the proposition is answering. Absent a prior question, the division can only be assumed. In the example below, we use the standard interpretation of grammatical cues (i.e., that the verb phrase to the right of the copula predicates of the argument specified by the noun phrase to the left). And recognize, consistent with the prior analysis of Boolean semantics, that division is contingent. Follow an evaluation of "The old man who lives by the sea *is a surfer*." Where for the sentence and the evaluation process the argument is colored purple and the predicate is colored green and italicized.

```
Find all men
    If count = 0 argument not found
    If count > = 1
        For each man found test 'is old'
            If count = 0 argument not found
            If count > = 1
                Apply restrictions 'who lives by the sea'
                    If count remaining = 0.....argument not found
                    **If count remaining = 1** apply predicate function
                        Evaluates yes or no
                    **If count > 1 AND predicate function = aggregation**
                        apply  predicate function
                            Evaluates yes or no
                    ELSE IF count > 1 argument is under specified
                        Learn which argument
```

For those cases highlighted in **bold** where the argument evaluates and the predicate, afterwards, does evaluate, the wff converts to a bivalent proposition.

The evaluation of arguments prior to predicates and the need for both to evaluate in order for the wff to denote a proposition should hold regardless of what the wff is being compared with. For example, if a person is evaluating the wff, the evaluation could take place against that person's prior knowledge by description (e.g., a collection of assertions treated as ground truth). Or it could take place against that person's prior or current knowledge by acquaintance (e.g., a collection of observations treated as ground truth). Or it could take place against that person's collection of rules pertaining to whether old men who live by the sea do or do not surf.

1.1 Given a sentential wff in the form of an assertion any word could be an argument or predicate. For example, in the sentences below and their associated f(a) notation, the word(s) in red italics is/are the asserted predicate; the remainder of the words being used to portray the argument.

The old man who lives by the sea is a *surfer*.	*Surfer*(the old man who lives by the sea)
The *old* man who lives by the sea is a surfer.	*Old*(the surfer man who lives by the sea)
The old man who *lives by the sea* is a surfer.	*Lives by the sea*(the old man who is a surfer)
The old *man* who lives by the sea is a *surfer*.	*man*(the old surfer who lives by the sea)

1.2 So any words could function as argument or predicate. For example, *man* can play argument or predicate; *old* can play argument or predicate, *surfer* can play argument or predicate.. So the notions of argument and predicate can be reinterpreted to signify purely functional roles (in our proposal, which of the two evaluation sequences) instead of referring to domains. To be a functional role is not to refer or say anything about the world. It does commit to a two-step sequence for truth evaluation (i.e., it says something about the functioning of its own execution machinery), but no particular external ontology or semantics.

1.3 Sentential WFF that are determined to be evaluable are propositions. Instead of words or words with sorts, propositions should be represented as roles of types to facilitate the creation of inference functions. Future logic will need to be integrated with a variant of the typing systems(Sowa, 1999; Turner, 1984; Curry, Feys, Craig et al., 1972; Church, 1960) most often associated with the predicate Calculus that captures the computational structures in modern typing systems but without losing the truth semantics advocated in this essay that comes from treating propositions as abstract objects just like numbers(Martin-Löf, 1994). The type roles are the standard roles for data types:

- type_label such as color or size or date or integer;
- type values such as red, blue, green for color or Monday, Tuesday, Wednesday for Days-of-the-week
- type operators such as Negate which works on any type, Next which works on rank ordered types

Propositions can be represented as either fully specified typed expressions e.g.,

Profession.surfer(Age.old ∧Critter.man, ∧ Domicile.by-the-sea)

or as some kind of propositional function that ranges over a collection such as

Profession.surfer(∀(Age.old ∧Critter.man, ∧ Domicile.by-the-sea))

Read: all old men who live by the sea are surfers. Or

Profession.surfer(∃ (Age.old ∧Critter.man, ∧ Domicile.by-the-sea)

Read: There exist an old man who live by the sea are surfers

Or the standard quantifiers can be replaced with generalized quantifiers as in

Profession.surfer(3-of-7 (Age.old ∧Critter.man, ∧ Domicile.by-the-sea)
Read: Three of the seven old men who live by the sea are surfers

Or the fully specified typed expressions can be replaced by their propositional identifier (e.g., P, Q, R etc..) in the standard form where unadorned means asserted (i.e., P means P is true).

1.4 It is hypothesized that the logical operators (e.g., negation, disjunction, conjunction, Implication, bi-conditional) will be understood to be those operators that apply to any well-formed type. So all typed expressions may utilize logical operators. And those comprised of more specialized types (e.g., Integers, Rationals, Hierarchies) will also support additional specialized operators (e.g., *division* requiring at least Rationals; *parent* requiring a hierarchy). And the blending of logical and mathematical operators will be facilitated. Consider an alternative view of Irrationals, not as numbers but as logically incommensurate metrics [Shavel, Thomsen 1990] where for example the square root of 2 would be defined in the following way. Start with Euclidean space defined as the cross product of two Rationals X and Y with a metric (i.e., a valid path) defined as for any (X,Y) Delta (X) XOR Delta (Y). Then define the diagonal as a path such that Delta(X) / Delta(Y) = K where K is a constant defining the slope and direction of the diagonal. This path is a specialization of the more general path Delta(X) IFF Delta(Y) By definition, and rearranging the terms slightly to make it clearer IFF(Delta(X), Delta(Y)) is the logical opposite of the Euclidean metric = XOR(Delta(X), Delta(Y). Not only are the paths allowed by the metric and the path required of the diagonal logically incommensurate, but the Euclidean metric required logic just to be specified.

2. Logical expressions (wff, propositions) need to be tied to evaluation states. Those states include

- Not potentially evaluable which applies to sentences that fail a preliminary test of well formedness
- Potentially evaluable which applies to any sentence/wff
- Argument evaluating which applies during the process of evaluating the argument for a wff
- Argument successfully evaluated
- Argument failed to evaluate because zero arguments were found
- Argument failed to evaluate because more than one argument was found
  Predicate evaluating
- Predicate failed to evaluate
- Predicate successfully evaluated. The wff is a proposition

The expression of any state has the form Evaluation state(expression) = some evaluation state
For example, Evaluation state(Surfer(old man who lives by the sea)) = evaluating argument tells us that the sentence's argument is currently being evaluated.

Explicit capturing of evaluation states is critical for designing logic-based applications in any real world domain where sentences cannot all be assumed to evaluate to propositions. This is because once evaluation states are tracked, they can fuel inferences related to dealing with the process of large scale sentence-to-proposition evaluations. And aggregate proposition evaluations such as those seen in section I above. These may include application logic for dealing with lack of arguments. For example,

IF evaluation state (surfer(old man who lives by the sea)) = zero arguments found THEN prompt user "The system is not aware of an old man who lives by the sea."
Or for dealing with more than one argument (e.g., 'Which old man who lives by the sea?' The tall one or the short one?') Or for dealing with missing predicates etc..

Aside from the fact that FOL can be thought of as the special case where
Evaluation state(P) = predicate successfully evaluated,
evaluation states offer two benefits to pure logic. First they provide a way of representing all wff whose argument fails to evaluate (i.e., cases where presupposition fails) without resorting to any meta language. Second, it is hypothesized that they provide a method for resolving impredicative sets. This we believe[9] is because the essential challenge with the set of all non-self membered sets is the inter-relationships between the dynamic evaluation states of the various propositional pieces.

3. Logical expressions may have many physical representations and vice versa

Standards such as Common Logic support dialects that enable the specification of special purpose formal languages that can be translated into the common core. In the future, this notion that there can be many representations of a common canonical form needs to be extended in three ways.

First the notion needs to be extended to different internal physical representations of the same logical expression. Any working logic system may have multiple internal physical representations for the same proposition. For example, start with the proposition *Billy's credit limit = $500.* Assuming the proposition is an actual representation of Billy's credit limit it must exist someplace. For example, it may exist physically represented in the bank's 'Central server storage'. The moment Billy goes to an ATM machine to make a balance enquiry, a duplicate physical representation will be created at the banks central server storage and sent to the ATM where Billy is located. Thus there will exist two different physical representations of the same logical expression of Billy's credit limit. As soon as there exist more than one physical representation of the same logical expression, it is possible for the two expressions to differ. For example if Billy made a request to withdraw funds based on his having $500, but during the time it took for the central server to send the ATM the physically represented logical expression, Billy's ex-girlfriend withdrew all his money directly form the bank where the central server is located, there is the possibility that Billy withdraws money that he no longer has. Of course large scale systems are built to avoid these kinds of problems (e.g., locking and two phase commit). The point is that the issue is quite real. And in this sense, future logic needs to be contraction free because in practice it already is. Contraction could still hold for logical expressions whose physical representations are jointly constrained (e.g., to be in the same physical representation). We believe that baking these distinctions into the fabric of Logic will make it much easier to use logic for processing natural language.

Second the notion needs to be extended to natural language. When this happens, the challenge becomes the fact that not only can multiple surface language expressions map to the same logical expression. But a single surface language expression (e.g., green) can map to multiple logical expressions (e.g., a value of a color type or a value of a political party etc..)

Third the notion needs to be extended to non verbal and linguistic interactions such as visual recognition of objects, scenes, gestures and poses. In this fashion external physically represented linguistic and non linguistic expressions may be mapped to internal logical expressions of a new generation of smarter logic-based systems.

[9] See Thomsen 2013 for a detailed treatment of the set of all non self membered sets paradox

This essay has attempted to show that the conflation of wff with propositions is a serious problem both in pure logic where it causes paradox and in the application of logic to large scale information processing where it can lead to empirically inaccurate results.. And it tried to demonstrate that key to disentangling wff from propositions is the re-interpretation of the components of a proposition - arguments and predicates - as differences in evaluation sequencing rather than differences in domain reference. Treating arguments and predicates as purely functional distinctions provides a more consistent basis for separating logic from ontology and cleanly supports the further differentiation between logical and physical representations needed for logic to support applications in natural language understanding.

References:

Aho, A. V.& Ullman, J. D. (1979). Universality of data retrieval languages. In *Proceedings of the 6th ACM SIGACT-SIGPLAN symposium on Principles of programming languages*, 110-119.

Bargiela, A.& Pedrycz, W. (2006). The roots of granular computing. In *GrC*, 806-809.

Barwise, J.& Etchemendy, J. (1989). *The liar: An essay on truth and circularity*.: Oxford University Press.

Berant, J., Chou, A., Frostig, R., and Liang, P. (2013). Semantic Parsing on Freebase from Question-Answer Pairs. In *EMNLP*, 1533-1544.

Berant, J.& Liang, P. (2014). Semantic parsing via paraphrasing. In *Proceedings of ACL*, 7, 92.

Boole, G. (1854). *An investigation of the laws of thought: on which are founded the mathematical theories of logic and probabilities*.: Dover Publications.

Church, A. (1960). Application of recursivc arithmetic to the problem of circuit synthesis. In" Summaries of Talks Presented at the Summer Institute for Symbolic Logic, Cornell University, 1957.". *Institute for Defense Analysis, Princeton*, 3.

Codd, E. F. (1970). A relational model of data for large shared data banks. *Communications of the ACM*, 13.

Codd, E. F. (1979). Extending the database relational model to capture more meaning. *ACM Transactions on Database Systems (TODS)*, 4, 397-434.

Curry, H. B., Feys, R., Craig, W., Hindley, J. R., and Seldin, J. P. (1972). *Combinatory logic*.: North-Holland Amsterdam.

Date, C. J.& Darwen, H. (2000). *Foundations for Future Database Systems*. Boston: Addison Wesley.

Domingos, P., Kok, S., Poon, H., Richardson, M., and Singla, P. (2006). Unifying Logical and Statistical AI. In *AAAI*, 6, 2-7.

Evangelopoulos, G., Zlatintsi, A., Potamianos, A., Maragos, P., Rapantzikos, K., Skoumas, G., and Avrithis, Y. (2013). Multimodal saliency and fusion for movie summarization based on aural, visual, and textual attention. *Multimedia, IEEE Transactions on*, 15, 1553-1568.

Haddock, G. E. R. (2012). *A critical introduction to the philosophy of Gottlob Frege*.: Ashgate Publishing, Ltd.

Kwiatkowski, T., Choi, E., Artzi, Y., and Zettlemoyer, L. (2013). Scaling semantic parsers with on-the-fly ontology matching.

Landini, G. (2015). The Collected Papers of Bertrand Russell, Volume 5: Toward Principia Mathematica, 1905–1908. *History and Philosophy of Logic*, 1-17.

Lee, H. N. (1961). Symbolic logic.

Maluf, D. A.& Tran, P. B. (2003). NETMARK: A Schema-Less Extension for Relational Databases for Managing Semi-structured Data Dynamically. In *Foundations of Intelligent Systems*, 231-241.

Martin-Löf, P. (1994). Analytic and synthetic judgements in type theory. *Kant and contemporary epistemology*, 87-99.

Melton, J. (2003). ISO International Standard (IS) Database Language SQL-Part 2: SQL/Foundation. *ISO/IEC*, 9075-9077.

Russell, B. (1995). *My philosophical development*.: Psychology Press.

Shavel, Thomsen 'A Tractarian Approach to Information Modeling' published in Wittgenstein-Towards a Re-evaluaiotn Verlag Holder-Pichler-Tempsky Wien 1990

Shewhart, W. A. (1925). The application of statistics as an aid in maintaining quality of a manufactured product. *Journal of the American Statistical Association*, 20, 546-548.

Smith, J. M.& Smith, D. C. (1977). Database abstractions: aggregation and generalization. *ACM Transactions on Database Systems (TODS)*, 2, 105-133.

Sowa, J. F. (1999). Knowledge representation: logical, philosophical, and computational foundations.Stalnaker, R. 'Common Ground' in Linguistics and Philosophy December 2002 Volume 25

Thomsen, E. (2002). *OLAP solutions: building multidimensional information systems*.: John Wiley & Sons, Inc. New York, NY, USA.

Thomsen, E. (2003a). BI's promised land. *INTELLIGENT ENTERPRISE-SAN MATEO-*, 6, 20-25.

Thomsen, E. (2003b). *Information Value*. Dordrecht, The Netherlands: Kluwer Academic Publishers.

Thomsen, E. (2012). *Knowing the World Is Language*.

Turmo, J., Ageno, A., and Català, N. (2006). Adaptive information extraction. *ACM Computing Surveys (CSUR)*, 38, 4.

Turner, R. (1984). *Logics for artificial intelligence*.: Horwood.

Vassiliadis, P.& Sellis, T. (1999). A survey of logical models for OLAP databases. *ACM Sigmod Record*, 28, 64-69.

Wittgenstein, L. (1973). Tractatus logico-philosophicus (1921). *Madrid: Alianza*.