Master's thesis

Consistency of LSO with syntactic equality

Yannick Spörl

UNIVERSITY
OF BERGEN

Faculty of Mathematics and Natural Sciences
Department of Informatics
Supervisor: Michał Walicki
01.06.2023

Abstract

LSO, Logic of Sentential Operators, is defined by extending first-order logic by sentential quantification and sentential operators. Its semantics is defined by a digraph, with kernels reflecting consistent valuations of all sentences and consistency of the language. It is possible, by using self-reference, to construct paradoxes and LSO need not be consistent with arbitrary valuations for its operators. While certain choices for operators allow existence of a kernel, not all do.

The syntactic equality operator, only true for syntactically identical sentences, raises the question, whether the language with it is consistent. Analyzing structure and complexity of sentences shows that certain subsets of sentences are consistent and that under some assumption the whole language is consistent. Even without a conclusive answer to the question of consistency of LSO with syntactic equality, consistency appears to hold and results provide a solid base for a potential proof showing the whole claim.

Contents

1 Introduction

1.1 Propositional and first-order logic

What makes the language of *first-order logic*, or *FOL*, more expressive than classical *propositional logic* is the means of stating facts about objects by quantification. As a proposition formulates a fact about only one specific object, we can only express the relation of some of these facts regarding a few selected objects.

We might express that *If Mount Everest has not been climbed yet, it will be climbed by someone* or

$$\neg everestClimbed \rightarrow everestWillBeClimbed$$

Semantically, our sentence would be considered true, if our proposition *everestClimbed* was true or if *everestWillBeClimbed* was true. So we may decide ourselves, whether we believe the propositions to be true or not and will with certainty know if our sentence about Mount Everest follows. But we may also be interested in more mountains and state a similar fact about *Mount Blanc* as well, but stating it for every mountain existing would be very tedious, as we are limited to explicitly mentioning selected objects.

In FOL, however, quantification over variables enables us to make the statement generic, talking about any set of objects we desire: *For every mountain holds, either it has been climbed by somebody, or it will be climbed by somebody*, or

$$\begin{aligned}
&\forall m\ (isMountain(m) \rightarrow (\\
&\qquad \exists h\ (isHuman(h) \wedge hasClimbed(h, m)) \vee \\
&\qquad \exists h\ (isHuman(h) \wedge willClimb(h, m))))
\end{aligned}$$

Of course, there need not be a connection between what we think that $isHuman(h)$ should mean and its semantics in a model for the sentence, or between objects on earth and what *the universe* is that we quantify over. A model making the sentence true would simply specify a set of objects and how they are related to each other. So what we quantify over is *some* set of objects.

1.2 Logic of sentential operators

What if these objects were sentences? Not only could we talk about sentences and their relations to each other but also about their truth value. By extending FOL with quantification over sentences as well as sentential operators, s-operators, we obtain the so-called *Logic of sentential operators*, or *LSO*. In LSO we could therefore state things, such as *Either there is no mountains, or all sentences made about mountains are false*, or

$$S = \neg\exists m\ (isMountain(m)) \vee \forall\phi\ (M(\phi) \to \neg\phi)$$

In this way, LSO contains both object language as well as its metalanguage. Drawing a parallel to axiomatic truth theories, where there is a metalanguage embedded into an object language, one might wonder if LSO is set up the same way. Like classical truth theories, LSO is a classical language with more expressive metalanguage on top of an object language.

[HL22] say on truth theories that "[...] a truth predicate is defined for a language, the so-called object language. This definition is carried out in a metalanguage [...] which is typically taken to include [...] another strong theory or expressively rich interpreted language". A crucial difference is that LSO's semantics makes no use of a meta-hierarchy, i.e. a *meta*-metalanguage, but instead remains on one meta-level.

1.3 Paradoxes

It is little surprising that with help of this metalanguage, we could now create paradoxes. We could, for instance, try and evaluate sentence S; a model making S true would be one where no object is a mountain. We might want to, however, fix our valuation of the object language and focus on the metalanguage.

In such case, our fixed valuation might propose the existence of mountains and then S's value might depend on whether or not all sentences in M are false. We could argue that $M = \{S\}$, since S talks about mountains and is our only sentence. So if S is true, all sentences about mountains are false, especially S itself, while if S is false, there must be a sentence about mountains that is true, which must be the only sentence in M, making S true, thus creating a paradox. In short

$$S \Rightarrow \forall\phi\ (M(\phi) \to \neg\phi) \Rightarrow M(S) \to \neg S \Rightarrow \neg S$$

and

$$\neg S \Rightarrow \neg\forall\phi\ (M(\phi) \rightarrow \neg\phi) \Rightarrow \exists\phi\ (M(\phi) \land \phi) \Rightarrow S$$

We can therefore not find a consistent valuation for S and might deduce that *sentential operator* M might be the culprit. Indeed, the choice $M = \{S\}$ is the cause of the paradox here.

A way that paradoxes can be dealt with are multi-valued theories of truth, for instance Kripke's theory where "[...] the truth predicate is only partially defined, that is, it only applies to some of the sentences of the language [and] a three-valued logic is employed, that is, a logic which operates with a third value, undefined, in addition to the truth values true and false." ([Bol17]).

LSO is distinct from such multi-valued logics in that it permits only truth and falsehood and thus has a classical reasoning system. One might thereafter wonder, how occurring paradoxes can be dealt with and if such operators always prevent us from finding consistent valuations.

1.4 Task and outline

Any n-nary operator P expresses some relation of sentences to each other. These properties or relations need not be connected to the truth- or falsehood of those sentences, they might as well express some other arbitrary notion, including syntax.

One operator of interest might be the notion of *syntactic equality*. Expressed with infix notation, the operator $S \doteq T$ could be defined as true, iff sentences S and T are syntactically identical, i.e. are *the same sentence*. This operator reflects some of the expressive power of LSO, as it states facts about the syntax of sentences. This work investigates consistency of the language LSO, where the only s-operator is "$\doteq$".

After this introductory part, in section 2, the syntax of LSO and its language graph semantics is introduced and defined with additional comments on sentential operators and consistency in general.

Equipped with the fundamental concepts and definitions of LSO, the syntactic equality operator $\doteq$ is defined in the main body of this work, section 3. Firstly some smaller but helpful results about the operators behavior are established; then it is investigated if LSO with syntactic equality can be consistent, by first focusing on more primitive sentences and then widening the scope on more complex sentences. The rather technical results guarantee consistency for a restricted set of sentences and establish a proof pattern that could show consistency for the whole language, even though consistency of the whole language is not proved in this work.

Section 4 reflects on what results have been found and how, what they suggest, and what remains to be investigated.

2 Basics of LSO

2.1 Syntax of LSO

In order to start talking about the *Language of Sentential Operators*, about the valuation of sententially quantified sentences and operators over them, we need to introduce some preliminary concepts, definitions and notation and see some preliminary results.

Following [Wal], we define *LSO* extending classical first-order logic FOL to FOL$^+$ (similarly one could extend higher-order logic). Firstly, we define FOL symbols and sentences. Assuming the reader to be familiar with FOL, we move through the definitions quickly.

Definition 2.1 (Symbols of FOL)**.**
The symbols of a language of FOL are a countable set of FOL-constants $\mathcal{C}^- = \{c_1, c_2, \dots\}$, a set of FOL-variables $\mathcal{V}^- = \{v_1, v_2, \dots\}$ and a countable set of FOL-predicate symbols $\mathcal{P}^- = \{P_1^-, P_2^-, \dots\}$, each with its finite arity.

The lack of function symbols in the definition is deliberate as any n-ary function can be expressed by a $(n+1)$-ary predicate, the graph of the function.

The super-script "$-$" helps us to distinguish between FOL-predicates and -variables and sentential operators and -variables. Next, we define symbols of the extended language FOL$^+$ and sentences of of FOL$^+$.

Definition 2.2 (Symbols of FOL$^+$)**.**
The symbols of a language of FOL$^+$ consist of a set of FOL$^+$-variables $\mathcal{V} = \{\phi_1, \phi_2, \dots\}$ and a set of FOL$^+$-operator symbols $\mathcal{P} = \{P_1, \dots P_n\}$, each with its finite arity (possibly 0).

We call members of $\mathcal{V}$ s-variables or sentential variables and members of $\mathcal{P}$ s-operators or sentential operators.

Definition 2.3 (Logical symbols and formulas of FOL$^+$)**.**
Logical symbols are "$($", "$)$", "$\neg$", "$\wedge$" and "$\forall$".
The set of formulas $\mathbf{F}$ *of a language of FOL$^+$ is defined by*

- $P_i^-(t_1, \dots t_n) \in \mathbf{F}$, *if* $P_i^- \in \mathcal{P}^-$, *has arity* n *and* $t_1, \dots t_i \in (\mathcal{V}^- \cup \mathcal{C}^-)$
- $\phi_i \in \mathbf{F}$, *if* $\phi_i \in \mathcal{V}$

- $P_i(F_1, \dots F_n) \in \mathbf{F}$, *if* $P_i \in \mathcal{P}$, *has arity* n *and* $F_1, \dots F_i \in \mathbf{F}$
- $\neg F \in \mathbf{F}$, *if* $F \in \mathbf{F}$
- $(F_1 \wedge F_2) \in \mathbf{F}$, *if* $F_1, F_2 \in \mathbf{F}$
- $\forall v_i\ (F) \in \mathbf{F}$, *if* $v_i \in (V^-)$ *and* $F \in \mathbf{F}$
- $\forall \phi_i\ (F) \in \mathbf{F}$, *if* $\phi_i \in (V)$ *and* $F \in \mathbf{F}$

We also define the following abbreviations:

- $(F_1 \to F_2) := \neg(F_1 \wedge \neg F_2)$
- $(F_1 \vee F_2) := (\neg F_1 \to F_2)$
- $\exists v_i\ (F) := \neg \forall v_i\ \neg(F)$
- $\exists \phi_i\ (F) := \neg \forall \phi_i\ \neg(F)$

Any FOL$^+$ operator P_i could also be allowed to take regular *FOL* terms $t \in (\mathcal{V}^- \cup \mathcal{C}^-)$ as arguments. This possibility is omitted here, as for all purposes later on, the object language will be assumed to have a fixed valuation already, hence making P_i behave like defined over only elements of $\mathbf{F}$.

The notions of *nominal occurrence* and *sentential occurrence* are intuitive: F occurs in nominal position, if it occurs as (a subformula of) an argument of some operator P that is atomic in S, e.g. $S = \forall \phi\ X \wedge (\phi \vee P(F \wedge F'))$. If an occurrence is not nominal then it is sentential. Typically, this F will be some variable ϕ and there will be distinction made between ϕ being in exclusively nominal, exclusively sentential or both positions.

Both object and sentential variables can have free occurrences, according to standard rules.

Definition 2.4 (Free variables)**.**
FOL-variable v_i *occurs freely in formula* F, *iff*

- $F = P_i^-(t_1, \dots t_n)$ *and* v_i *is one of* $t_1, \dots t_n$
- $F = P_i(F_1, \dots F_n)$ *and* v_i *is free in one of* $F_1, \dots F_n$
- $F = \neg F'$ *and* v_i *is free in* F'

- *$F = (F_1 \wedge F_2)$ and v_i is free in F_1 or F_2*
- *$F = \forall v_j\ (F')$, $v_i \neq v_j$ and v_i is free in F'*
- *$F = \forall \phi_i\ (F')$ and v_i is free in F'*

Analogously s-variable ϕ_i is free in formula F, iff

- *$F = \phi_i$*
- *$F = P_i(F_1, \dots F_n)$ and ϕ_i is free in one of $F_1, \dots F_n$*
- *$F = \neg F'$ and ϕ_i is free in F'*
- *$F = (F_1 \wedge F_2)$ and ϕ_i is free in F_1 or F_2*
- *$F = \forall v_i\ (F')$ and ϕ_i is free in F'*
- *$F = \forall \phi_j\ (F')$, $\phi_i \neq \phi_j$ and ϕ_i is free in F'*

Definition 2.5 (Sentences of FOL$^+$)**.**
S is a FOL$^+$-sentence, $S \in \mathbf{S}$, iff $S \in \mathbf{F}$ and S has no free variables. We define the set of atomic sentences $\mathbf{A}$, by $S \in \mathbf{A}$, iff $S = P_i^-(t_1, \dots t_n)$ and each t_i is no variable or $S = P_i(S_1, \dots S_n)$ and each S_i is an atomic sentence.

We define FOL^+_M as FOL$^+$ extended by constant symbols for all elements of some FOL interpretation domain M and $\mathbf{S}_M$ and $\mathbf{A}_M$ as (atomic) sentences of FOL^+_M.

Having repeated all necessary definitions of FOL and inserted some extending notions, it seems a good moment to pause and reflect on what we are able to construct now.

We have ordinary variables of FOL, that we can quantify over and have defined FOL-predicates over terms, which form the atoms of our formulas. We have, however, new types of atoms: firstly, we have so-called *s-variables*, that we can also quantify over, and we have *s-operators*, that take as arguments arbitrary formulas.

It is important to note that even if we may have touched upon the definition of formulas, we deal, as our FOL$^+$ is a *Language of sentential operators*, solely with sentences and such have defined quantification over just sentences as well as operators applied to just sentences.

Example 2.6.
Let S be an arbitrary FOL-sentence. With our extended language we can apply s-operators to any sentences, such that P_1 is an unary meta-operator, for which either $S \in P_1$ or $S \notin P_1$.

$$S' = (P_1(S) \to \neg S)$$

S' now says that, if indeed $S \in P_1$, S ought to be false. Additionally, we may express this notion about arbitrary sentences, by quantifying over s-variables.

$$A = \forall \phi_1 \ (P_1(\phi_1) \to \neg \phi_1)$$

We might want to think of P_1 as a meta-property that sentences can have, for instance, that P_1 contains all contradictions. Then we would want to consider A as true, since any negated contradiction is a tautology. In that sense, S' depends on both the meta-property P_1 as well as the sentence S.

It might be worthwhile mentioning that if LSO reminds of quantified modal logic, it is not it, even though it might be able to capture it, as it has no possible worlds semantics, but semantics defined over a language graph (as seen in the next section).

In the example, we touched vaguely upon semantics and valuation of sentences, and mentioned a distinction between *object-level* and *meta-level*.

One quirk of this meta-level is that if a sentence, such as A, claims some fact about all sentences, it also states it about itself.

How the semantics of FOL$^+$ is defined and specifically how this circularity is dealt with, is elaborated in the next section.

2.2 Graph semantics for LSO

2.2.1 Language graphs

Having a sentence A, such as in Example 2.6, we notice how A quantifies over all sentences. Not only are there too many sentences to manually check, but in fact infinitely many. A *language graph* is therefore not only a convenient way to visualize A and its relationship to other sentences, but in fact how [Wal] defines semantics of FOL$^+$, by *kernels* of language graphs.

The language graph of a language is a directed graph (and from this point on *graph* shall refer to a *digraph*), with sentences as its vertices and edges going from one sentence to some other sentences, whose falsehood make the sentence true. More precisely, we define the language graph of LSO:

Definition 2.7 (Language graph of FOL$^+$).
For some interpretation domain M, the language graph $\mathcal{G}_M(FOL^+)$ is a directed graph with

- *all sentences $\mathbf{S}_M$ as vertices, where*
- *each atomic sentence $S \in \mathbf{A}_M$ has an outgoing and incoming edge from and to its negation $\neg S$ and*
- *each non-atomic sentence $S \notin \mathbf{A}_M$ has outgoing edges to*
  - *the sentence F, if $S = \neg F$*
  - *the sentences $\neg F_1$ and $\neg F_2$, if $S = F_1 \wedge F_2$*
  - *$\neg F(x)$ for all $x \in M$, if $S = \forall v_i \ (F(v_i))$*
  - *$\neg F(S')$ for all $S' \in \mathbf{S}$, if $S = \forall \phi_i \ (F(\phi_i))$*

While the whole graph $\mathcal{G}_M(\text{FOL}^+)$ is certainly not visualizable, we might want to look at the subgraph with A from Example 2.6 as root. From this point speaking of $\mathcal{G}_M$ refers to $\mathcal{G}_M(\text{FOL}^+)$, if not specified otherwise.

Example 2.8.
To draw the graph for A (Figure 1), we first unfold its definition.

$$A = \forall \phi_1 \ (P_1(\phi_1) \to \neg \phi_1) = \forall \phi_1 \ \neg(P_1(\phi_1) \wedge \phi_1)$$

$$\begin{tikzcd}
 & & & & & & \bullet \arrow[dll, bend right] \\
 & & & & A \arrow[dllll] \arrow[dll] \arrow[d] \arrow[dr] & & \\
(P_1(S_1) \wedge S_1) \arrow[d] \arrow[dr] & & (P_1(S_2) \wedge S_2) \arrow[d] \arrow[dr] & & \cdots & (P_1(A) \wedge A) \arrow[d] \arrow[dr] & \\
\bullet \arrow[d] & \bullet \arrow[d] & \bullet \arrow[d] & \bullet \arrow[d] & & \bullet \arrow[d] & \bullet \arrow[d] \\
P_1(S_1) & S_1 & P_1(S_2) & S_2 & & P_1(A) & A \arrow[uuuu, bend right]
\end{tikzcd}$$

Figure 1: The subgraph for A.

There are some details hidden in the graph. Firstly, as all, infinitely many, sentences are substituted into A, we chose to only display some selected ones S_1, S_2 and A and abbreviate the rest with an edge to "$\ldots$" and can deduce that they would follow the same patterns as the explicit ones.

Moreover, of course A (as well as $\neg A$ and $\neg\neg A$, $\ldots$) are also substituted. In that, and all similar cases, cycles back to the root of the tree appear.

The leaves of the tree appear to be all atoms $P_1(\phi_1)$ as well as all sentences ϕ_1. These sentences are generally not atomic and are therefore just the roots of their own subtrees, with some of them having cycles back to A.

While $\bullet$ represent some intermediate sentences (here negations of their targets) and simplify visualization, there is also duplicate vertices, in this case the vertex for A; as shown in [Wal], these auxiliary vertices do not alter the semantics we intend the graph to have.

Having convenient means of looking at a sentence and its relation to other sentences, we now define how truth and falsehood relate to that graph.

2.2.2 Kernels and solvability

For an arbitrary graph $\mathcal{G} = (V, E), E \subseteq V \times V$, and the notations $E^- = \{(y, x) \mid (x, y) \in E\}$ and $E(X) = \bigcup_{x \in X}\{y \mid (x, y) \in E\}$, we define kernels.

Definition 2.9 (Kernel)**.**
$K \subseteq V$ is a kernel of $\mathcal{G} = (V, E)$, iff $V \setminus K = E^-(K)$. Then $K \in ker(\mathcal{G})$.

So the kernel of a graph is an independent set of vertices, such that any vertex that is not in the kernel has an edge to the kernel. As edges in our language graph represent negations, the kernel is, informally, a maximal set not contradicting itself.

The definition of the kernel is, meanwhile, equivalent to an assignment of truth-values to all sentences, that is consistent. Therefore a kernel $K \in sol(\mathcal{G}_M)$, as of [NM44], a solution to the language graph $\mathcal{G}_M$, is also an assignment $v \in 2^V$, where $S \in K$ $v(S) = \mathbf{1}$, meaning each sentence in the kernel is considered true and $S' \notin K$ iff $v(S') = \mathbf{0}$, meaning everything outside the kernel (negating some sentence in the kernel) is considered false. Hence $\mathcal{G}_M$ is solvable, if it has a kernel, a consistent truth-assignment to all sentences.

Such does [Wal] define the solvability of the graph and shows that $\mathcal{G}_M$ without any operators is always solvable, and thus has a kernel. He continues to suggest that the corresponding proof can be extended to also include operators, given some restrictions on the operators interpretations, and thus that $\mathcal{G}_M$ remains solvable, even with operators.

This claim might appear surprising at first glance, as we have mentioned paradoxes and circularity earlier but not yet addressed it and indeed do we look more closely into it and comment on it in the next subsection.

Before, however, we revisit our sentence and subgraph from the previous example and see if we can find a consistent valuation for it.

Example 2.10.
Our familiar sentence $A = \forall\phi_1\ (P_1(\phi_1) \to \neg\phi_1)$ seems a good example, as its subgraph has cycles. If $\mathcal{G}_M$ has a kernel v, A obtains a truth value, so let us see what happens, if we assign A. (It might be helpful to keep the graph from Example 2.8 close by and check the respective truth-assignments to the vertices.)

- *If $v(A) = \mathbf{1}$, all $v(P_1(S) \wedge S) = \mathbf{0}$, so for each true S, $S \notin P_1$, so also $A \notin P_1$. That is a crucial restriction and reflects, where the paradox could occur. If A and $P_1(A)$ would be in the kernel, $\neg A$ would be forced into the kernel as well, making it not a kernel, as there is an edge between the two kernel members A and $\neg A$, reflecting the occurring inconsistency of valuating both as true.*

- *Conversely, if $v(A) = \mathbf{0}$, there is a sentence S, such that $v(P_1(S) \wedge S) = \mathbf{1}$, so $S \in P_1$ and $v(S) = \mathbf{1}$. Now if there would only be $A \in P_1$ and no other sentence, then A would have to be true, forcing A back into the kernel, thus having A both in and outside the kernel, reflecting a* *paradox. Thus for the kernel that excludes A, A can not be the sole sentence in P_1.*

We conclude that the danger of paradoxes is existent, for certain choices regarding the operator P_1, for instance that $P_1 = \{A\}$. Thus we may want to believe that there is always a kernel covering the graph and notice that "solving of the graph" includes fixing the proper interpretation of our operators.

2.2.3 Semantic equivalence

We certainly want to use the notion of semantic equivalence, alas, write $S \Leftrightarrow S'$ if and only if S and S' are equivalently evaluated under any interpretation. However, with semantics defined over graph kernels this notion needs to be treated with extra care.

Definition 2.11 (Semantic equivalence $\overset{\mathcal{G}_M}{\Longleftrightarrow}$)**.**

We call S and S' semantically equivalent, $S \overset{\mathcal{G}_M}{\Longleftrightarrow} S'$, iff for any kernel K of $\mathcal{G}_M$, $S \in K$ iff $S' \in K$.

We saw already that a kernel K is a truth-assignment v to all sentences, hence $v(S) = \mathbf{1}$ iff $S \in K$ iff $S' \in K$ iff $v(S') = \mathbf{1}$. So $S \overset{\mathcal{G}_M}{\Longleftrightarrow} S'$ guarantees S and S' obtain the same truth value. Further, $\overset{\mathcal{G}_M}{\Longleftrightarrow}$ always holds relative to some fixed object language interpretation and its corresponding language graph $\mathcal{G}_M$. Without providing proofs, we rely upon $\overset{\mathcal{G}_M}{\Longleftrightarrow}$, which contains

- PL equivalence,
- FOL equivalence,
- that all tautologies/contradictions are equivalent, $\top \overset{\mathcal{G}_M}{\Longleftrightarrow} \top / \perp \overset{\mathcal{G}_M}{\Longleftrightarrow} \perp$.

Additionally to these cases, there might as well be other arguments, justifying why some sentences always reside in the same kernels and thus $S \overset{\mathcal{G}_M}{\Longleftrightarrow} S'$.

Going forward, we will, for readability, abbreviate $\overset{\mathcal{G}_M}{\Longleftrightarrow}$ by $\Leftrightarrow$.

Example 2.12.
Consider as an example the sentence $\forall\phi\ (\phi \vee (S \wedge S))$. By propositional equivalence and existence of false sentences, $\forall\phi\ (\phi \vee (S \wedge S)) \Leftrightarrow S$. Taking some A assumed to be false and drawing the corresponding subgraph (Figure 2) where A is substituted for ϕ confirms that both vertices of the equivalence above obtain the same truth value.

It can be verified easily (by changing $A^{\mathbf{0}}$ to $A^{\mathbf{1}}$ and $\neg A^{\mathbf{1}}$ to $\neg A^{\mathbf{0}}$ in the graph) that the same holds for some A assumed true, hence for any arbitrary sentence.

$$\begin{tikzcd}
 & & \forall\phi(\phi \vee (S \wedge S))^{\mathbf{1/0}} \arrow[dll] \arrow[d] \arrow[dr] & \\
\neg A \wedge \neg(S \wedge S)^{\mathbf{0/1}} \arrow[d] \arrow[dr] & & A \wedge \neg(S \wedge S)^{\mathbf{0/0}} \arrow[dl] \arrow[d] & \cdots \\
A^{\mathbf{0}} & S \wedge S^{\ \mathbf{1/0}} \arrow[d] & \neg A^{\mathbf{1}} \arrow[d] & \\
 & \neg S^{\ \mathbf{0/1}} \arrow[d] & A^{\mathbf{0}} & \\
 & S^{\ \mathbf{1/0}} & &
\end{tikzcd}$$

Figure 2: The subgraph for $\forall\phi\ (\phi \vee (S \wedge S))$.

The superscripts $\bullet^{\mathbf{1/0}}$ indicate possible valuation of its vertices. For instance $S^{\mathbf{1/0}}$ and $\neg S^{\mathbf{0/1}}$ indicates that under the assignment $v(S) = \mathbf{1}$ we have $v(\neg S) = \mathbf{0}$ and vice versa.

What can be read from the graph is that for $v(S) = \mathbf{1}$ we have $v(S \wedge S) = \mathbf{1}$, while $v(S) = \mathbf{0}$ and $v(S \wedge S) = \mathbf{0}$, making S equivalent to $S \wedge S$. Further, every sentence is substituted for ϕ, due to the $\forall\phi$ quantifier and hence, whenever some sentence A is substituted, so is $\neg A$. So if this A were valuated to $\mathbf{0}$, $v(A \wedge \neg(S \wedge S)) = \mathbf{0}$, while $v(\neg A \wedge \neg(S \wedge S)) = v(\neg S)$. It can be concluded that all direct children of the root vertex are either equivalent to $\perp$ or equivalent to $\neg S$. Thus the root sentence is equivalent to S, i.e. always obtains the same value that was assigned to S.

2.2.4 Definitional extension for operators

[Wal] shows that a consistent language can be extended with *definitional extension* of operators, while preserving solvability of the graph, as the extended language has essentially the same kernels as before. The extension mechanism for a new (e.g. unary) operator P is the definition of a sentence $S^P = \forall\phi\ (P(\phi) \leftrightarrow F(\phi))$, where $F(\phi)$ is an arbitrary formula in the language without P and with only ϕ freely. It holds then that if the language without operator P had a kernel, the language with P has *essentially* the same kernel and that S^P is in that kernel.

Definitional extension is thereby neat means of introduction of new operators that preserve consistency. However, so far we have only certainty that FOL$^+$ without operators is consistent and defining a new operator on that basis does not promise too interesting results. Consider for instance the extension $S^T = \forall\ (T(\phi) \leftrightarrow \phi)$. We know with certainty that the language with operator T is consistent and a kernel will always contain $T(X)$ if it contained X, thus T reflecting truth of sentences; but we have not really gained any expressivity with $T(\phi)$, if only introduced a more awkward way of saying ϕ.

Hence, at this point at least, definitional extension is a potentially interesting mechanism, but give us only essentially trivial consistent operators of little expressivity. We therefore move onward in search of new consistent operators and bear in mind that, if we find such operators, we can use definitional extensions to combine them as we wish into new operators.

2.2.5 Solvability and s-operators

Knowing that we can find consistent truth-values for sentences of FOL$^+$ without operators and possibly for sentences with operators, even though that might be trickier, according to Example 2.10, we can examine the impact of operators further.

In [Wal] it was sketched how adding operators that behave like boolean functions should retain solvability, as they, intuitively speaking, can not systematically confuse the boolean valuation we want to assign to a sentence.

Formally it was suggested that if an n-ary operator P satisfies one of the following conditions, $\mathcal{G}_M$ is solvable[1]:

[1] A previous version of [Wal] contained the cited conditions; after making the author aware of the following counter-examples to his conditions, the work has been updated since and the (faulty) conditions omitted from the most recent version of [Wal].

1. for all sentences $\phi_i \in \mathbf{S}$: $P(\phi_1, \dots \phi_i, \dots \phi_n) = P(\phi_1, \dots \neg\phi_i, \dots \phi_n)$
2. for all sentences $\phi_i \in \mathbf{S}$: $P(\phi_1, \dots \phi_i, \dots \phi_n) = \neg P(\phi_1, \dots \neg\phi_i, \dots \phi_n)$

Unfortunately, however, it is possible to construct a counter-example to these conditions, so an interpretation of operators that conform with the conditions but still form a paradox.

Example 2.13.
Let $X = \forall\phi_1\ ((P_1(\phi_1) \land P_2(\phi_1)) \rightarrow \neg\phi_1)$. Further, let

- *$X \in P_1$ and, if $\phi \in P_1$, then also $\neg\phi \in P_1$.*
  So $P_1 = \{X, \neg X, \neg\neg X, \neg\neg\neg X, \dots\}$ and therefore it is in accordance with condition 1.
- *Further, we define P_2's interpretation inductively.*
  - *$S \in P_2$, for any atomic sentence $S \in \mathbf{A}$.*
  - *$(S \land S') \in P_2$ for any sentences $S, S' \in \mathbf{S}$.*
  - *$\neg\neg S \in P_2$ for any sentence $S \in P_2$.*

  Less formally, P_2 contains sentences that start with exactly an even number of negations, thus is in accordance with condition 2.

Let, for simplicity, $\forall\phi_1\ (P_3(\phi_1) \leftrightarrow P_1(\phi_1) \land P_2(\phi_1))$, so that P_3 is a definitional extension of P_1 and P_2. Then, by unfolding definitions, $X = \forall\phi_1\ (P_3(\phi_1) \rightarrow \neg\phi_1) = \forall\phi_1\ \neg(P_3(\phi_1) \land \phi_1)$.

We note that for any sentence ϕ, $P_3(\phi) \Leftrightarrow \phi = \underbrace{\neg \dots \neg}_{2n} X$, so P_3 is true for exactly X and all its double-negations.

To see the paradoxical situation we have created, consider the relevant subgraph (Figure 3) we want to assign truth-values to.

The superscripts $\bullet^{\mathbf{1}}/\bullet^{\mathbf{0}}$ on vertices represent an already fixed truth-assignment, while $\bullet^{?}$ shows that the assignment is not yet clear. The central observations here are

- *for all sentences S that are not $P_3 = \{X, \neg\neg X, \neg\neg\neg\neg X, \dots\}$, $(P_3(S) \land S)$ obtain value $\mathbf{0}$, no matter the value of S itself. Only the values of $\{X, \neg\neg X, \neg\neg\neg\neg X, \dots\}$ seem to actually matter.*

$$\begin{tikzcd}
\bullet^{?} \arrow[rrr, bend left] & & & X^{?} \arrow[dll] \arrow[d] \arrow[drr] \arrow[drrrr] & & & & & \\
& (P_3(X) \wedge X)^{?} \arrow[dl] \arrow[dr] & & (P_3(\neg\neg X) \wedge \neg\neg X)^{?} \arrow[d] \arrow[dr] & & \cdots & & (P_3(S) \wedge S)^{\mathbf{0}} \arrow[dl] \arrow[dr] & \\
\bullet^{?} \arrow[d] & & \bullet^{\mathbf{0}} \arrow[d] & \bullet^{?} \arrow[d] & \bullet^{\mathbf{0}} \arrow[d] & & \bullet^{\mathbf{1}} \arrow[d] & & \bullet \arrow[d] \\
X^{?} \arrow[uuu, bend left] & & P_3(X)^{\mathbf{1}} & \neg\neg X^{?} \arrow[dl, bend left] & P_3(\neg\neg X)^{\mathbf{1}} & & P_3(S)^{\mathbf{0}} & & S \\
& & \bullet^{?} \arrow[ull, bend left] & & & & & &
\end{tikzcd}$$

Figure 3: The subgraph for X has unbroken odd cycles.

- *The value of all vertices* $(P_3((\neg\neg)^* X) \wedge (\neg\neg)^* X)$ *depend solely on the value for* $((\neg\neg)^* X)$.
- *Each* $((\neg\neg)^* X)$ *is involved in an odd cycle with* X.

The unbroken odd cycles prevent us from finding a consistent truth-assignment. If there would be a consistent assignment v, *with* $v(X) = \mathbf{1}$, *then* $v(P_3(X) \wedge X) = \mathbf{1}$ *making* $v(X) = \mathbf{0}$, *hence contradicting itself.*

If, conversely, $v(X) = \mathbf{0}$, *then there must be some* $Y = \underbrace{\neg \ldots \neg}_{2n} X$, *such that* $v(Y) = \mathbf{1}$, *which implies that* $v(X) = \mathbf{1}$.

Concluding, in terms of the graph, we have odd-cycles involving X, *that are not broken, as all outgoing edges from the cycle are* $\mathbf{0}$. *In terms of solvability that means that for this interpretations of operators* P_1 *and* P_2 *we can not find a consistent truth-assignment, hence have constructed a paradox. As the operators conform with the "boolean-function" requirement, but still lead to a paradox, conditions 1 and 2 are not sufficient.*

Consequently, the suggested conditions for operators are not closed under the definitional extension by means of existing operators, as $P_3(\phi) \leftrightarrow P_1(\phi) \wedge P_2(\phi)$ violates the conditions, while both P_1 and P_2 fulfil them. Therefore the conditions as proposed by [Wal], while already restrictive, can not be sufficient for solving the language graph.

This issue could be fixed by strengthening the conditions. A possible alteration could be to keep the conditions, but require them to be fulfilled for all operators as well as any operator defined by combinations of existing operators, i.e.:

Definition 2.14 (Suggested conditions for consistent operators).
For every operator P, as well as every operator definable by definitional extension, one of the following conditions must hold:

1. *for all sentences* $\phi_i \in \mathbf{S}$*:* $P(\phi_1, \dots \phi_i, \dots \phi_n) = P(\phi_1, \dots \neg\phi_i, \dots \phi_n)$
2. *for all sentences* $\phi_i \in \mathbf{S}$*:* $P(\phi_1, \dots \phi_i, \dots \phi_n) = \neg P(\phi_1, \dots \neg\phi_i, \dots \phi_n)$

These strengthened conditions force a closure under definitional extension, however, the operators lose even more of their expressivity, limiting their usefulness. It is trivial that a restriction like $P = \emptyset$ assigns a constant value to all occuring operators and thus keeps the language consistent but adds no expressivity to the language at all. Another possibility would be to see if there are not completely different conditions that could be imposed onto the operators and would guarantee solvability.

An interesting observation connected to the previous example is that the two operators, even despite their intended "boolean functionality", were defined over syntactic properties of sentences. For instance, if a sentence $A \in P_2$, then $\neg A \notin P_2$. But on the other hand $(\neg A \wedge \neg A) \in P_2$, illustrating the syntactic character of P_2 or at least, its lacking closure under semantic equivalence. Concluding, our meta-operators are causing trouble, especially, when being syntactic, distinguishing sentences not by their boolean value, but their syntactic structure.

3 The operator "Syntactic equality"

3.1 Introducing syntactic equality

After we looked closer into the behaviour of arbitrary operators, their influence on solvability and saw some examples, we move forward and focus on one specific operator. In Example 2.13, we already saw that operators can be defined on a syntactic level, signifying some syntactic property of sentences, for instance, if a sentence "starts with an even number of negations".

Another, at first glance simpler, notion is *syntactic equality*, that is true precisely when both its arguments are the same sentence, so the same string of symbols.

Definition 3.1 (Syntactic equality $\doteq$)**.**
The binary operator P_1 with the fixed interpretation where for any two distinct sentences $S, S' \in \mathbf{S}$, $v(P_1(S,S)) = \mathbf{1}$ and $v(P_1(S,S')) = \mathbf{0}$ is called syntactic equality and is denoted by $(S \doteq S') := P_1(S,S')$.

For readability we abbreviate $(S \not\doteq S') := \neg(S \doteq S')$.

One pitfall to avoid, when examining sentences with the $\doteq$-operator that contain variables, is to mistake when the operator obtains its boolean value. It is of course legitimate to have a sentence like $\exists\phi_1\ (\phi_1 \doteq S)$. When just looking at the literal $\phi_1 \doteq S$, one might think that $v(\phi_1 \doteq S) = \mathbf{0}$, since ϕ_1 and S are clearly not the same string of symbols. ϕ_1 is, however an open formula and not a sentence.

So we do not evaluate $\phi_1 \doteq S$ yet, but rather when ϕ_1 has been substituted by a sentence (at the leaves of the subtree of the sentence). Thus we encounter $\doteq$-literals containing open formulas, that only gain a determined truth-value once being substituted into fully.

To get more familiar with the operator and get a feel for what we can express with it let us take a look at some exemplary sentences.

Example 3.2.
Simple examples include

- $(S \doteq T)$*, which is a contradiction, while*
- $(S \doteq S)$ *is a tautology.*
- $\forall\phi_1\ ((\phi_1 \doteq S) \rightarrow \neg\phi_1)$ *is saying that S is false, while*

- $\forall\phi_1\exists\phi_2\ (\phi_1 \doteq \phi_2)$ *says that every sentence is syntactically equal to some sentence*

We can even express quite sophisticated things about the syntax of sentences.

- $\mathit{isConjunction}(\phi_1) \leftrightarrow \exists\phi_2\exists\phi_3\ (\phi_1 \doteq (\phi_2 \wedge \phi_3))$
- $\mathit{doubleNegated}(\phi_1) \leftrightarrow \exists\phi_2\ (\phi_1 \doteq (\neg\neg\phi_2))$
- $\mathit{propositionalAxiom1}(\phi_1) \leftrightarrow \exists\phi_2\exists\phi_3\ (\phi_1 \doteq (\phi_2 \rightarrow (\phi_3 \rightarrow \phi_2)))$

In Definition 3.1, we defined that $v(S \doteq S) = \mathbf{1}$ and $v(S \doteq S') = \mathbf{0}$. One might wonder about the value of $(\forall\phi_1(\phi_1) \doteq \forall\phi_2(\phi_2))$, as it could be argued that they are the same sentence after all, just with different bound variables. In such cases we valuate $v(\forall\phi_1(\phi_1) \doteq \forall\phi_2(\phi_2)) = \mathbf{1}$ as the sentences are identical up to renaming of bound variables.

In Definition 2.11, we established semantic equivalence and one might be tempted to also regard equivalent sentences as identical. Certainly

$$\forall\phi_1(\phi_1) \Leftrightarrow \forall\phi_2(\phi_2)$$

But this equivalence also holds for syntactically different sentences, such as $(S \wedge S) \Leftrightarrow S$. While $(S \wedge S)$ is a conjunction S is perhaps not, thus they are not syntactically identical (and no renaming can fix that) and we want to therefore valuate $v((S \wedge S) \doteq S) = \mathbf{0}$.

Thus, the operator is closed under renaming of bound variables, and when stating facts like "$\phi \doteq S$ is only true for exactly one sentence ϕ", we bear in mind that it is true for exactly one sentence ϕ up to renaming of bound variables.

Without addressing how big the expressive power of a operator is, it is safe to say that the syntactic equality operator exceeds very strict conditions (such as $\doteq$ being the empty set) that guarantee existence of a kernel of the language graph, as the operator behaves certainly not like a boolean function. Its expressive power is however still limited.

We recall Example 2.13, where we constructed sentence

$$X = \forall\phi_1\ ((P_1(\phi_1) \wedge P_2(\phi_1)) \rightarrow \neg\phi_1)$$

We then also fixed an interpretation of the operator P_1, such that $X \in P_1$. So in some sense the operator P_1 contained a sentence that contained P_1 and thus P_1 contained itself. This very vague notion could be seen as the operator "talking about itself". This can, however, not happen with our $\doteq$-operator and is a limitation.

Let us say we wanted to construct a similar sentence for $\doteq$ and write an expression like $Y = \forall\phi_1 \left((\phi_1 \doteq Y) \to \neg\phi_1\right)$. It might look, at first glance, like we created a sentence Y that speaks about itself, but on further examination, Y is not a sentence at all, as of course Y can not contain itself as a proper substring.

These vague intuitions give a picture of how the $\doteq$-operator might differ from arbitrary operators, and what can and can not be expressed with it. The question that remains, of course, is whether or not this operator is consistent, thus, with its fixed valuation always permits us to find a consistent boolean valuation for all sentences, hence a kernel of the whole language graph.

3.2 Preliminary results

Before actually proving a first consistency result for the syntactic equality operator, we need some preliminary work. After fixing some notation we look closely at sentence-structure and interactions of the operator with other literals.

Definition 3.3 (Notation).
We denote arbitrary LSO-sentences by S_i or by T_i. For variables $\phi_1, \dots \phi_m$ we denote formulas that may or may not contain none, some, or all of $\phi_1, \dots \phi_m$ freely, but certainly no others, by $S_i(\phi_1, \dots \phi_m)$, or $T_i(\phi_1, \dots \phi_m)$. We write $\top/\bot$ for any formula that is a tautology/contradiction.

We now have the two different notations of S and $S(\dots)$, depending on whether any variables occur freely in it.

Further we examine the structure of arbitrary sentences. A crucial detail that becomes very handy, shown in [Wal], is that an arbitrary sentence is semantically equivalent to a sentence in *Prenex Disjunctive Normal Form*, or *PDNF*. While *DNF* guarantees a sentence is a disjunction of conjunctions of literals, *PDNF* additionally requires all quantifiers to be at the beginning of the sentence, thus the sentence having the general pattern of

$$S = \exists\!\!\!\forall\phi_1 \dots \exists\!\!\!\forall\phi_n \; (C_1(\phi_1, \dots \phi_n) \vee \dots C_c(\phi_1, \dots \phi_n))$$

where $\exists\!\!\!\forall$ stands for any of the quantifiers $\forall$ or $\exists$ and each C_i is a conjunction of literals, where a literal is a (possibly negated) atomic formula.

As each sentence is equivalent to a sentence in PDNF, we assume, when speaking of sentences, without loss of generality, the sentence to be in PDNF, if not explicitly stated otherwise.

Definition 3.4 (PDNF, conjunctions, literals)**.**
By $D(\phi_1, \ldots \phi_n)$ we denote a disjunctive matrix, thus any PDNF sentence S has the form $S = \exists\!\!\!\forall \phi_1 \ldots \exists\!\!\!\forall \phi_n \ D(\phi_1, \ldots \phi_n)$. By $C_i(\phi_1, \ldots \phi_n)$ we denote a conjunction of literals, such that

$$D(\phi_1, \ldots \phi_n) = (C_1(\phi_1, \ldots \phi_n) \vee \ldots C_c(\phi_1, \ldots \phi_n))$$

and further, each

$$C_i(\phi_1, \ldots \phi_n) = (L_1(\phi_1, \ldots \phi_n) \wedge \ldots L_l(\phi_1, \ldots \phi_n))$$

where a literal $L_i(\phi_1, \ldots \phi_n)$ is either

1. *ϕ_k, for some $1 \leq k \leq n$*
2. *$\neg\phi_k$, for some $1 \leq k \leq n$*
3. *$(S_1(\phi_1, \ldots \phi_n) \doteq S_2(\phi_1, \ldots \phi_n))$*
4. *$(S_1(\phi_1, \ldots \phi_n) \not\doteq S_2(\phi_1, \ldots \phi_n))$*
5. *a FOL atom, $P_1^-(\ldots)$, with no s-variables*
6. *a negated FOL atom, $\neg P_1^-(\ldots)$, with no s-variables*

Within PDNF there might appear regular FOL atoms, going forward, however, those will not be considered and we will rather focus on "pure" FOL$^+$, that is on sentences without any FOL atoms. The reason for this is, with the distant goal of consistency of LSO, we want to fix valuation for object language, so an interpretation with object domain M, and to show that this still allows a consistent valuation of all sentences, all vertices in the language graph $\mathcal{G}_M$. Under these circumstances, with object language not being circular in the graph, their role is merely a constant contribution of **1**/**0** to sentences with metalanguage. Without more details or proof, we stay aware that object language neither causes nor contributes to the potential inconsistency of LSO and will largely be omitted.

It is now time to inspect occurrences of the syntactic equality operator more closely. The operator applied on two formulas with free variables is either trivially satisfied, unsatisfiable or dictates what variables must be identified with which subformulae.

Lemma 3.5.
For any formulas $S(\phi_1, \dots \phi_n)$ and $T(\phi_1, \dots \phi_n)$, either

$$(S(\phi_1, \dots \phi_n) \doteq T(\phi_1, \dots \phi_n)) \Leftrightarrow \bot/\top$$

or

$$(S(\phi_1, \dots \phi_n) \doteq T(\phi_1, \dots \phi_n)) \Leftrightarrow (L_1(\phi_1, \dots \phi_n) \wedge \dots L_n(\phi_1, \dots \phi_n))$$

where

$$L_i(\phi_1, \dots \phi_n) = (\phi_i \doteq S(\phi_1, \dots \phi_{i-1}, \phi_{i+1}, \dots \phi_n)).$$

Proof. If $S(\phi_1, \dots \phi_n)$ and $T(\phi_1, \dots \phi_n)$ are not unifiable under any renaming of bound variables (say, by FOL-Martelli-Montanari algorithm, treating quantifiers with variables, $\forall\phi, \exists\phi$, as operator names), then

$$(S(\phi_1, \dots \phi_n) \doteq T(\phi_1, \dots \phi_n)) \Leftrightarrow \bot$$

If they are unifiable, then the most general unifier provides for each ϕ_i a formula $S(\phi_1, \dots \phi_{i-1}, \phi_{i+1}, \dots \phi_n)$ it needs to be identified with, while $(\phi \doteq \phi) \Leftrightarrow \top$. □

A special case of Lemma 3.5 is, when $n = 1$, thus both formulas contain only up to one free variable. Then either

$$(S(\phi_1) \doteq T(\phi_1)) \Leftrightarrow \top/\bot$$

or

$$(S(\phi_1) \doteq T(\phi_1)) \Leftrightarrow (\phi_1 \doteq S')$$

Example 3.6.
Let X, Y and Z denote some sentences.

- $(X \wedge (\phi_1 \doteq Y)) \doteq (\phi_2 \wedge (Z \doteq Y)) \Leftrightarrow ((\phi_1 \doteq Z) \wedge (\phi_2 \doteq X))$
- $(\phi_1 \wedge (X \doteq Y)) \doteq ((\phi_2 \vee Z) \wedge (X \doteq Y)) \Leftrightarrow (\phi_1 \doteq (\phi_2 \vee Z))$
- $(\phi_1 \wedge (X \vee \phi_2)) \doteq ((\phi_1 \vee Y) \wedge Z) \Leftrightarrow \bot$

We may also inspect the conjunctions of a DNF and find that we may always assume exactly one sentential occurrence of a given variable in each conjunction.

Lemma 3.7.
For any DNF-matrix $D(\phi_1, \dots \phi_n)$ and some variable ϕ_i, it holds that $D(\phi_1, \dots \phi_n) \Leftrightarrow D'(\phi_1, \dots \phi_n)$ where for each conjunct $C_i(\phi_1, \dots \phi_n)$ in $D'(\phi_1, \dots \phi_n)$ it holds that

$$C_i(\phi_1, \dots \phi_n) = (\phi_i \wedge R(\phi_1, \dots \phi_n))$$

or

$$C_i(\phi_1, \dots \phi_n) = (\neg\phi_i \wedge R(\phi_1, \dots \phi_n))$$

where $R(\phi_1, \dots \phi_n)$ is an arbitrary conjunction of literals with no sentential occurrences of ϕ_i.

Proof. Assuming that duplicate occurrences of ϕ_i have been reduced already, the conjunction must be of one of the following forms:

1. $C_i(\phi_1, \dots \phi_n) = (\phi_i \wedge R(\phi_1, \dots \phi_n))$ or $C_i(\phi_1, \dots \phi_n) = (\neg\phi_i \wedge R(\phi_1, \dots \phi_n))$, in which case the claim holds already.

2. $C_i(\phi_1, \dots \phi_n) = (\phi_i \wedge \neg\phi_i \wedge R(\phi_1, \dots \phi_n))$. The conjunction is a contradiction and thus can be omitted in the equivalent DNF-matrix.

3. $C_i(\phi_1, \dots \phi_n)$ contains neither ϕ_i nor $\neg\phi_i$ sententially. By applying the equivalence

$$C_i(\phi_1, \dots \phi_n) \Leftrightarrow (\phi_i \wedge C_i(\phi_1, \dots \phi_n)) \vee (\neg\phi_i \wedge C_i(\phi_1, \dots \phi_n))$$

   we replace the conjunction by two new conjunctions for an equivalent DNF.

$\square$

Thus we can always find an equivalent DNF where each conjunction contains exactly one sentential occurrence of ϕ_i.

What might not be apparent at first glance is how the previous Lemma did not assume syntactic equality or any specific sentential operators in the sentence. It merely involved sentential variables and is therefore applicable to any DNF, no matter which specific operators are being considered.

Before turning back to syntactic equalities, we prove another fact about arbitrary sentences.

Lemma 3.8.
Any sentence $S = \forall\phi_1 \dots \forall\phi_n\ D(\phi_1, \dots \phi_n)$ is equivalent to a conjunction of up to 2^n sentences, $S \Leftrightarrow S^0 \wedge \dots S^{2^n-1}$ where

$$S^i = \forall\phi_1 \dots \forall\phi_n\ (([\neg]\phi_1 \wedge \dots [\neg]\phi_n) \rightarrow D^i(\phi_1, \dots \phi_n))$$

with $D^i(\phi_1, \dots \phi_n)$ having no variables in sentential positions and i being a binary representation i^b of length n, such that if i^b's $(x-1)$th digit is 1, then $[\neg]\phi_x = \phi_x$, while if its $(x-1)$th digit is 0, then $[\neg]\phi_x = \neg\phi_x$, thus i coding which sentential variables occur negated in S^i.

Proof. The claim can be shown by propositional equivalences.

$$\begin{aligned}
S = \forall\phi_1 \dots \forall\phi_n\ & D(\phi_1, \dots \phi_n) && (1)\\
\Leftrightarrow \forall\phi_1 \dots \forall\phi_n\ & \big[(\phi_1 \wedge D^1(\phi_1, \dots \phi_n)) \vee (\neg\phi_1 \wedge D^0(\phi_1, \dots \phi_n))\big]\\
\Leftrightarrow \forall\phi_1 \dots \forall\phi_n\ & \big[(\phi_1 \wedge \phi_2 \wedge D^3(\phi_1, \dots \phi_n))\vee\\
& (\phi_1 \wedge \neg\phi_2 \wedge D^2(\phi_1, \dots \phi_n))\vee\\
& (\neg\phi_1 \wedge \phi_2 \wedge D^1(\phi_1, \dots \phi_n))\vee\\
& (\neg\phi_1 \wedge \neg\phi_2 \wedge D^0(\phi_1, \dots \phi_n))\big]\\
\Leftrightarrow\ \dots &\\
\Leftrightarrow \forall\phi_1 \dots \forall\phi_n\ & \big[(\phi_1 \wedge \phi_2 \wedge \dots \phi_n \wedge D^{2^n-1}(\phi_1, \dots \phi_n))\vee && (2)\\
& (\phi_1 \wedge \phi_2 \wedge \dots \neg\phi_n \wedge D^{2^n-2}(\phi_1, \dots \phi_n))\vee\\
& \dots\\
& (\neg\phi_1 \wedge \neg\phi_2 \wedge \dots \neg\phi_n \wedge D^0(\phi_1, \dots \phi_n))\big]\\
\Leftrightarrow \forall\phi_1 \dots \forall\phi_n\ & \big[(\phi_1 \wedge \phi_2 \wedge \dots \phi_n) \rightarrow D^{2^n-1}(\phi_1, \dots \phi_n)\big]\wedge && (3)\\
\forall\phi_1 \dots \forall\phi_n\ & \big[(\phi_1 \wedge \phi_2 \wedge \dots \neg\phi_n) \rightarrow D^{2^n-2}(\phi_1, \dots \phi_n)\big]\wedge\\
& \dots\\
\forall\phi_1 \dots \forall\phi_n\ & \big[(\neg\phi_1 \wedge \neg\phi_2 \wedge \dots \neg\phi_n) \rightarrow D^0(\phi_1, \dots \phi_n)\big]\\
\Leftrightarrow\ \ & S^{2^n-1} \wedge S^{2^n-2} \wedge \dots S^0 && (4)
\end{aligned}$$

From (1) to (2) we get by repeated application of Lemma 3.7 and distributivity laws. (3) utilizes the idea that for any combination of sentences substituted for $\phi_1, \dots \phi_n$, only exactly one of the conjunctions $([\neg]\phi_1 \wedge \dots [\neg]\phi_n)$ is true and all others false. For (4) it yields up to 2^n new sentences, as each variable occuring both positively and negated doubles the number of combinations of negated and non-negated variables.

The same equivalences can be utilized as shown, for some variable occuring only positively, only negatively, or not at all, in which case the number of combinations remains smaller than 2^n. □

The last fact we need at the moment, is that syntactic equality literals can absorb each other in some cases.

Lemma 3.9.
For every conjunction $C(\phi_1)$ of literals with at most one free variable ϕ_1, which occurs only in nominal positions, one of the following equivalences holds.

1. $C(\phi_1) \Leftrightarrow (\phi_1 \doteq S_1)$, *or*
2. $C(\phi_1) \Leftrightarrow ((\phi_1 \not\doteq T_1) \wedge \ldots \wedge (\phi_1 \not\doteq T_m))$, *or*
3. $C(\phi_1) \Leftrightarrow \top/\bot$.

Proof. By Lemma 3.5 we need only consider atoms of the form $(\phi_1 \doteq S)$, so we consider

$$C(\phi_1) \Leftrightarrow ((\phi_1 \doteq S_1) \wedge \ldots (\phi_1 \doteq S_s) \wedge (\phi_1 \not\doteq T_1) \wedge \ldots (\phi_1 \not\doteq T_t))$$

We consider no literals that are equivalent to $\top/\bot$, as one $\bot$-literal makes the whole conjunction equivalent to $\bot$, while literals equivalent to $\top$ can be omitted (if they are the only literals, the whole conjunction is equivalent to $\top$).

- If $s > 1$, then $C(\phi_1) \Leftrightarrow \bot$, as no sentence ϕ_1 satisfies $(\phi_1 \doteq S_1) \wedge (\phi_1 \doteq S_2)$ for $S_1 \neq S_2$.
- If $s = 0$, then $C(\phi_1) \Leftrightarrow ((\phi_i \not\doteq T_1) \wedge \ldots (\phi_i \not\doteq T_i))$.
- If $s = 1$ and $t = 0$, then $C(\phi_1) \Leftrightarrow (\phi_i \doteq S_1)$.
- If $s = 1$, $t > 0$ and $S_1 = T_i$ for some i, then $C(\phi_1) \Leftrightarrow \bot$, no matter the substituted sentence.
- If $s = 1$, $t > 0$ and $S_1 \neq T_i$ for all i, then $C(\phi_1) \Leftrightarrow (\phi_1 \doteq S_1)$, since the conjunction is true exactly when S_1 is substituted for ϕ_1.

□

3.3 Sentences with one quantifier

What we would like to gain at some point is the assurance that our fixed interpretation of the syntactic equality operator allows a consistent boolean valuation of all sentences. That is, however, quite a strong claim. We now move in this direction, but prove a weaker claim for a smaller set of sentences.

We show that, when having fixed valuation of our underlying object language, the truth of all sentences with just one quantifier, depends only on the truth of some of its subsentences occurring nominally. By $[\neg]T$ we denote T or $\neg T$.

Theorem 3.10.
For every valuation of object language and for any sentence A with exactly one sentential quantifier, either $A \Leftrightarrow \top/\bot$ or there is some set of sentences, $\mathcal{T} = \{T_1, \dots T_t\}$, occurring nominally in A, such that $A \Leftrightarrow [\neg]T_1 \wedge \dots [\neg]T_t$ or $A \Leftrightarrow [\neg]T_1 \vee \dots [\neg]T_t$.

Proof. We consider $A = \forall\phi_1\ D(\phi_1)$. In case $A = \exists\phi_1\ D(\phi_1)$, we consider instead $\neg A$ (with its negation pushed inwards) to obtain the set $\mathcal{T}$ for $\neg A$, hence for A as well. By Lemma 3.8, A is equivalent to a conjunction of two sentences, $A \Leftrightarrow A^0 \wedge A^1$, where $A^0 = \forall\phi_1\ (\neg\phi_1 \rightarrow D^0(\phi_1))$ and $A^1 = \forall\phi_1\ (\phi_1 \rightarrow D^1(\phi_1))$.

We keep track that in our equivalent sentences, both $D^0(\phi_1)$ and $D^1(\phi_1)$ are still free from sentential ϕ_1. We can now turn to the simplified sentence

$$A^1 = \forall\phi_1\ (\phi_1 \rightarrow D^1(\phi_1))$$

knowing that A^0 can be treated the same way.

$D^1(\phi_1)$ being free from sentential occurrences of ϕ_i allows application of Lemma 3.9, thus restricting the types of conjunctions that appear in the DNF. A degenerate case is of course when $D^1(\phi_1) = D^1$, i.e. has no free variables at all. Then, of course, D^1 obtains a constant boolean value and we are done. Otherwise,

$$\begin{aligned} D^1(\phi_1) = \big[&(\phi_1 \doteq S_1) \vee \dots \\ &(\phi_1 \doteq S_s) \vee \\ &((\phi_1 \not\doteq T^1_1) \wedge \dots (\phi_1 \not\doteq T^1_{n^1})) \vee \dots \\ &((\phi_1 \not\doteq T^t_1) \wedge \dots (\phi_1 \not\doteq T^t_{n^t}))\big] \end{aligned}$$

Now, depending on which conjunctions are exactly in the DNF, different interactions might happen. First of all, if

- $t = 0$, then $D^1(\phi_1) \Leftrightarrow (\phi_1 \doteq S_1) \vee \dots (\phi_1 \doteq S_s)$ and so $A^1 \Leftrightarrow \bot$, as there are true sentences ϕ_1 that are different from all $S_1, \dots S_s$.
- If $t > 0$, we notice, that if for any two conjunctions $((\phi_1 \not\doteq T_1^i) \wedge \dots (\phi_1 \not\doteq T_{n^i}^i))$ and $((\phi_i \not\doteq T_1^j) \wedge \dots (\phi_1 \not\doteq T_{n^j}^j))$ there is no T_x^y common to both of them, then $D^1(\phi_1) \Leftrightarrow \top$ as for any sentence ϕ_1, one of the conjunctions is true and so $A^1 \Leftrightarrow \forall\phi_1\ (\phi_1 \to D^1(\phi_1)) \Leftrightarrow \forall\phi_1\ (\phi_1 \to \top) \Leftrightarrow \top$.
- Otherwise, all conjunctions $((\phi_1 \not\doteq T_1^i) \wedge \dots (\phi_1 \not\doteq T_{n^i}^i))$ have some common sentences $\mathcal{T} := \{T_1, \dots T_t\}$. Let then be $\mathcal{S} := \{S_1, \dots S_s\}$ the sentences from all conjunctions $(\phi_1 \doteq S_1) \vee \dots (\phi_1 \doteq S_s)$. We know that $((\phi_i \not\doteq T_1^1) \wedge \dots (\phi_i \not\doteq T_{n^1}^1)) \vee \dots ((\phi_i \not\doteq T_1^t) \wedge \dots (\phi_i \not\doteq T_{n^t}^t))$ is false for any $T_i \in \mathcal{T}$ but otherwise true.

  Thus, if

  - $\mathcal{T} \subseteq \mathcal{S}$, we know that for any $T_i \in \mathcal{T}$ the conjunction $(\phi_1 \doteq T_i)$ is true and so $A^1 \Leftrightarrow \forall\phi_1\ (\phi_1 \to D^1(\phi_1)) \Leftrightarrow \forall\phi_1\ (\phi_1 \to \top) \Leftrightarrow \top$.
  - Conversely, if $(\mathcal{T} \setminus \mathcal{S}) \neq \emptyset$, then there are some sentences, for which none of the conjunctions are true, and therefore $D^1(\phi_1) \Leftrightarrow (\phi_1 \not\doteq T_1) \wedge \dots (\phi_1 \not\doteq T_t)$ for all $T_i \in (\mathcal{T} \setminus \mathcal{S})$.

For a quick recap, we have found that for the sentence $A^1 = \forall\phi_1\ (\phi_1 \to D^1(\phi_1))$, the DNF-matrix $D^1(\phi_1)$ is equivalent to $\top/\bot$ in most cases, except for when all conjunctions with only negative literals have some sentences in common, that do not occur in any conjunction with positive literals, in which case the DNF-matrix is true for all sentences except those in question. Unfolding this idea now brings the proof to a close.

$$\begin{aligned}
A^1 &= \forall\phi_1\ \left[\phi_1 \to D^1(\phi_1)\right] \\
&\Leftrightarrow \forall\phi_1\ \left[\phi_1 \to ((\phi_1 \not\doteq T_1) \wedge \dots (\phi_1 \not\doteq T_t))\right] \\
&\Leftrightarrow \forall\phi_1\ \left[(\phi_1 \to (\phi_1 \not\doteq T_1)) \wedge \dots (\phi_1 \to (\phi_1 \not\doteq T_t))\right] \\
&\Leftrightarrow \forall\phi_1\ \left[\phi_1 \to (\phi_1 \not\doteq T_1)\right] \wedge \dots \forall\phi_1\ \left[\phi_1 \to (\phi_1 \not\doteq T_t)\right] \\
&\Leftrightarrow \forall\phi_1\ \left[(\phi_1 \doteq T_1) \to \neg\phi_1\right] \wedge \dots \forall\phi_1\ \left[(\phi_1 \doteq T_t) \to \neg\phi_1\right] \\
&\Leftrightarrow \neg T_1 \wedge \dots \neg T_t
\end{aligned}$$

The very last equivalence might come unexpected, as we seem to eliminate the quantifier. It is justified, however, as $\forall\phi_1\ ((\phi_1 \doteq T_i) \to \neg\phi_1)$ is trivially true for all sentences apart from T_i, and if T_i is substituted, then T_i needs to be false as per the implication.

Thus, if A was not equivalent to $\top/\bot$ in the first place, we have found a set of sentences $\mathcal{T}$, that occurred only nominally in A, such that $A \Leftrightarrow [\neg]T_1 \wedge \ldots [\neg]T_t$ (if $A = \exists\phi_1\ D(\phi_1)$, then $A \Leftrightarrow [\neg]T_1 \vee \ldots [\neg]T_t$). $\square$

The proof made use of a technical detail that provides a very convenient equivalence between formulas, which is often used in the remainder of this work, both explicitly and implicitly.

We used the fact that $\forall\phi\ ((\phi \doteq S) \to \phi) \Leftrightarrow S$. Generalizing this instance, we get that in formulas

$$(\phi_1 \doteq S(\phi_2, \ldots \phi_n)) \wedge D(\phi_1, \phi_2, \ldots \phi_n)$$

or

$$(\phi_1 \doteq S(\phi_2, \ldots \phi_n)) \to D(\phi_1, \phi_2, \ldots \phi_n)$$

we can always replace all occurrences of ϕ_1 within $D(\phi_1, \phi_2, \ldots \phi_n)$ by the formula $S(\phi_2, \ldots \phi_n)$ without changing its boolean value, to get

$$(\phi_1 \doteq S(\phi_2, \ldots \phi_n)) \wedge D(S(\phi_2, \ldots \phi_n), \phi_2, \ldots \phi_n)$$

or

$$(\phi_1 \doteq S(\phi_2, \ldots \phi_n)) \to D(S(\phi_2, \ldots \phi_n), \phi_2, \ldots \phi_n)$$

This accomplishes the elimination of all occurrences of ϕ_1, except the one in $(\phi_1 \doteq S(\phi_2, \ldots \phi_n))$ and also puts the formula $S(\phi_2, \ldots \phi_n)$ from formerly only nominal position into a sentential position.

Before thinking about what this result entails, let us look at an example and see how exactly the equivalences used in the proof come into play, and how a sentence can be logically equivalent to nominal subsentence of itself.

Example 3.11.
Let us consider a non-trivial sentence.

$$\begin{aligned} A = \forall\phi_1\ ((\phi_1 \doteq S_1) \vee (\phi_1 \wedge (\phi_1 \doteq S_2)) \vee \\ (\phi_1 \wedge (\phi_1 \not\doteq S_1) \wedge (\phi_1 \not\doteq S_3)) \vee (\neg\phi_1 \wedge (\phi_1 \not\doteq S_5))) \end{aligned}$$

We can now, step by step, apply the different equivalences to get an instance of the Theorem (labels on the side reference the used Lemma or definition).

$$\begin{aligned}
A = \; & \forall\phi_1 \left[(\phi_1 \doteq S_1) \vee (\phi_1 \wedge (\phi_1 \doteq S_2)) \vee \right. && (3.4,\ 3.5)\\
& \quad \left. (\phi_1 \wedge (\phi_1 \not\doteq S_1) \wedge (\phi_1 \not\doteq S_3)) \vee (\neg\phi_1 \wedge (\phi_1 \not\doteq S_4))\right] \\
\Leftrightarrow \; & \forall\phi_1 \left[(\phi_1 \wedge (\phi_1 \doteq S_1)) \vee (\neg\phi_1 \wedge (\phi_1 \doteq S_1)) \vee (\phi_1 \wedge (\phi_1 \doteq S_2)) \vee \right. \\
& \quad \left. (\phi_1 \wedge (\phi_1 \not\doteq S_1) \wedge (\phi_1 \not\doteq S_3)) \vee (\neg\phi_1 \wedge (\phi_1 \not\doteq S_4))\right] \\
\Leftrightarrow \; & \forall\phi_1 \left[(\phi_1 \wedge ((\phi_1 \doteq S_1) \vee (\phi_1 \doteq S_2) \vee ((\phi_1 \not\doteq S_1) \wedge (\phi_1 \not\doteq S_3)))) \vee \right. \\
& \quad \left. (\neg\phi_1 \wedge ((\phi_1 \not\doteq S_1) \vee (\phi_1 \not\doteq S_4)))\right] \\
\Leftrightarrow \; & \forall\phi_1 \left[\phi_1 \rightarrow ((\phi_1 \doteq S_1) \vee (\phi_1 \doteq S_2) \vee ((\phi_1 \not\doteq S_1) \wedge (\phi_1 \not\doteq S_3)))\right] \wedge && (3.8)\\
& \forall\phi_1 \; \left[\neg\phi_1 \rightarrow ((\phi_1 \not\doteq S_1) \vee (\phi_1 \not\doteq S_4))\right] \\
\Leftrightarrow \; & \forall\phi_1 \left[\phi_1 \rightarrow (\phi_1 \not\doteq S_3)\right] \wedge \forall\phi_1\left[\neg\phi_1 \rightarrow \top\right] && (*)\\
\Leftrightarrow \; & \forall\phi_1 \left[(\phi_1 \doteq S_3) \rightarrow \neg\phi_1\right] \wedge \top \\
\Leftrightarrow \; & \neg S_3
\end{aligned}$$

(∗) In the proof of Theorem 3.10 we argued how

- $(\phi_1 \doteq S_2) \vee ((\phi_1 \not\doteq S_1) \wedge (\phi_1 \not\doteq S_3)) \Leftrightarrow ((\phi_1 \not\doteq S_1) \wedge (\phi_1 \not\doteq S_3))$, *since* S_2 *is not equal to* S_1 *or* S_3
- *Also a non-negated and negated equality to* S_1 *cancel each other out, such that* $(\phi_1 \doteq S_1) \vee ((\phi_1 \not\doteq S_1) \wedge (\phi_1 \not\doteq S_3)) \Leftrightarrow (\phi_1 \not\doteq S_3)$
- *Meanwhile* $(\phi_1 \not\doteq S_1) \vee (\phi_1 \not\doteq S_4)$ *is vacuously true, as no sentence can be equal to* S_1 *and* S_4 *at the same time.*

The vague intuition mentioned before, that the operator $\doteq$ is not able to express facts about itself, seems to hold. If a sentence is equivalent to just $\top/\bot$, it is independent of any other sentences' truth-assignment, including itself, and is therefore not self-referential at all. And a sentence that is equivalent to some of its nominal subsentences is at least not directly self-referential; but as any sentence (other than A) might appear as a subsentence of A, we can not quite be sure of what kinds of self-reference we might still get. The claim we proved was restricted to sentences with one quantifier, and so a big part of the graph and a major part of the whole picture is still missing.

Another, more abstract, observation is how the only troublesome interaction we seem to have at this level is between an occurrence of a variable ϕ in both nominal and sentential position.

3.4 Sentences with multiple quantifiers

3.4.1 Sentences with only nominal or only sentential occurrences of variables

Our closing observation of the previous section was how the troublesome interaction was between a ϕ occuring both sententially and nominally. Indeed, it is possible to show that sentences with an arbitrary number of quantifiers get a fixed boolean valuation, as long as all its quantified variables $\phi_1 \dots \phi_n$ appear exclusively nominally, thus in form of $S(\phi_1 \dots \phi_n) \doteq S'(\phi_1 \dots \phi_n)$, or exclusively sententially, in form of ϕ_i.

Theorem 3.12.
For every valuation of object language, each sentence

$$A = \exists\!\!\!\forall \phi_1 \exists\!\!\!\forall \phi_2 \dots \exists\!\!\!\forall \phi_n \; D(\phi_1, \dots \phi_n)$$

with every ϕ_i occuring only in nominal position of some operator P has a fixed boolean value, $A \Leftrightarrow \top/\bot$.

Proof. We consider $\exists\!\!\!\forall \phi_1 = \forall \phi_1$, but the same arguments can be employed for $\exists\!\!\!\forall \phi_1 = \exists \phi_1$.

By definition of semantics over the language graph $\mathcal{G}_M$, the vertex of A has a double edge to all $A^S = \exists\!\!\!\forall \phi_2 \dots \exists\!\!\!\forall \phi_n \; D(S, \phi_2, \dots \phi_n)$ for every sentence S. Each A^S has of course also only nominal occurrences of variables, thus A's subgraph has no cycles and is a tree with all leaves being various atoms of the form $P(S_1, \dots S_n)$, which has a fixed boolean value and no outgoing edges (see Figure 4). Therefore A's value is induced bottom up and fixed. $\square$

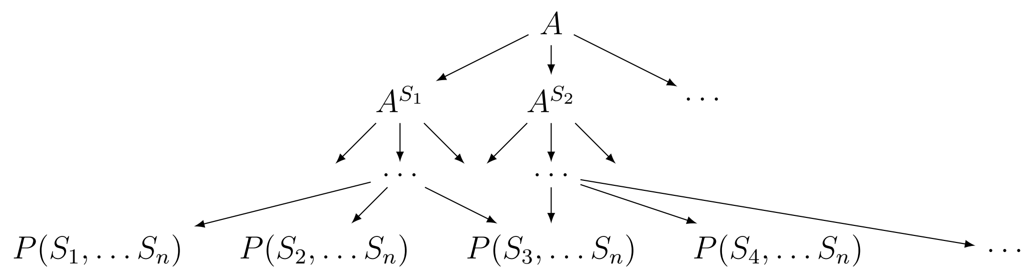

Figure 4: The subgraph of A has no cycles.

The converse case, all sentences with every ϕ_i occuring only in sentential position, is merely a consequence of [Wal]'s proof that $\mathcal{G}_M$ without operators always has a solution. Every sentence with s-variables only in sentential position has only atoms of the operator with fixed arguments, if at all, which obtain a fixed value, based on the semantics of the operator. They contribute just a constant $\mathbf{1}/\mathbf{0}$ to its supersentence and thus sentences have a fixed value, as per existence of a kernel of $\mathcal{G}_M$ without the operator.

However, utilizing equivalences from this work, a different approach to proving consistency of all sentences with every ϕ_i occuring only in sentential position can be taken, and thus the proof of the following Theorem 3.14, together with the new Lemma 3.13, confirming the already known result, is different and more compact than [Wal]'s proof.

The central observation that is used in the next Lemma is that a sentence of the form $\forall\phi\ldots\ ((\phi \to D) \wedge (\neg\phi \to D'))$ acts like a case distinction on ϕ's value. Coupled with the $\forall$-quantifier, ϕ is always substituted by S as well as its negation $\neg S$ and thus, provided ϕ does not occur in D and D', jointly inducing the same value, allowing distribution of other quantifiers (even if existential) over the outermost conjuncts.

Lemma 3.13.
For any valuation of all sentences and any sentence

$$\forall\phi_1\ A(\phi_1) = \forall\phi_1 \exists\!\!\!\forall\phi_2 \ldots \exists\!\!\!\forall\phi_n \left[(\phi_1 \to D(\phi_2,\ldots\phi_n)) \wedge (\neg\phi_1 \to D'(\phi_2,\ldots\phi_n))\right]$$

with ϕ_1 not occuring in $D(\phi_2,\ldots\phi_n)$ and $D(\phi_2,\ldots\phi_n)$,

$$\forall\phi_1\ A(\phi_1) \Leftrightarrow \exists\!\!\!\forall\phi_2 \ldots \exists\!\!\!\forall\phi_n\ D(\phi_2,\ldots\phi_n) \wedge \exists\!\!\!\forall\phi_2 \ldots \exists\!\!\!\forall\phi_n\ D'(\phi_2,\ldots\phi_n)$$

Proof. In the subgraph of $\forall\phi_1\ A(\phi_1)$ (see Figure 5), due to the $\forall\phi_1$ quantifier, every sentence is being substituted for ϕ_1. Thus, for an arbitrary S that is substituted, so is $\neg S$, and we can draw the subgraph for specifically S and $\neg S$, knowing the same pattern reappears for any sentence and its negation.

For $A(S) = \exists\!\!\!\forall\phi_2 \ldots \exists\!\!\!\forall\phi_n \left[(S \to D(\phi_2,\ldots\phi_n)) \wedge (\neg S \to D'(\phi_2,\ldots\phi_n))\right]$ we then abbreviate the subtree, where every ϕ_i gets substituted by some T_i by a triangle and focus on one arbitrary branch with the vertex $\left[(S \to D(T_2,\ldots T_n)) \wedge (\neg S \to D'(T_2,\ldots T_n))\right]$ at its end. It can now be reasoned, why value of the root is induced by

$\exists\!\!\!\forall\phi_2 \ldots \exists\!\!\!\forall\phi_n \; D(\phi_2, \ldots \phi_n)$ and $\exists\!\!\!\forall\phi_2 \ldots \exists\!\!\!\forall\phi_n \; D'(\phi_2, \ldots \phi_n)$ and independent of S's value (*left* and *right subtree* refers to the subtrees of $A(S)$ and $A(\neg S)$, respectively):

- In the left subtree, for each branch ending in $(S \to D(T_2, \ldots T_n)) \wedge (\neg S \to D'(T_2, \ldots T_n))$ there is an identical branch in the right subtree ending in $(\neg S \to D(T_2, \ldots T_n)) \wedge (S \to D'(T_2, \ldots T_n))$.
- If S's value was **1**, the branch in the left subtree gets its values induced solely by $D(T_2, \ldots T_n)$, while the branch in the right subtree gets its values induced solely by $D'(T_2, \ldots T_n)$.
- The same applies to any other arbitrary branch in the subtree. $A(S)$'s value depends solely on the values of all branches, which all get their values induced by $D(\phi_2, \ldots \phi_n)$, the converse holds for $A(\neg S)$ and $D'(\phi_2, \ldots \phi_n)$.
- Thus, in case S's value was **1**, $A(S) \Leftrightarrow \exists\!\!\!\forall\phi_2 \ldots \exists\!\!\!\forall\phi_n \; D(\phi_2, \ldots \phi_n)$ and $A(\neg S) \Leftrightarrow \exists\!\!\!\forall\phi_2 \ldots \exists\!\!\!\forall\phi_n \; D'(\phi_2, \ldots \phi_n)$.
- If S's value was **0** the same situation arises, only with both subtrees swapped, thus the left subtree getting its values induced by $D'(\phi_2, \ldots \phi_n)$ and the right subtree by $D(\phi_2, \ldots \phi_n)$.
- It can be concluded that no matter S's valuation, both subtrees jointly induce the same value, $\exists\!\!\!\forall\phi_2 \ldots \exists\!\!\!\forall\phi_n \; D(\phi_2, \ldots \phi_n) \wedge \exists\!\!\!\forall\phi_2 \ldots \exists\!\!\!\forall\phi_n \; D'(\phi_2, \ldots \phi_n)$.
- S was chosen arbitrarily and so for all sentences, coupled together with their negation, the same equivalence holds and overall,

$$\forall\phi_1 \; A(\phi_1) \Leftrightarrow \exists\!\!\!\forall\phi_2 \ldots \exists\!\!\!\forall\phi_n \; D(\phi_2, \ldots \phi_n) \wedge \exists\!\!\!\forall\phi_2 \ldots \exists\!\!\!\forall\phi_n \; D'(\phi_2, \ldots \phi_n)$$

$\square$

Theorem 3.14.
For every valuation of object language and, any sentence

$$A = \exists\!\!\!\forall\phi_1 \exists\!\!\!\forall\phi_2 \ldots \exists\!\!\!\forall\phi_n \; D(\phi_1, \ldots \phi_n)$$

with every ϕ_i occuring only in sentential position has a fixed boolean value, $A \Leftrightarrow \top/\bot$.

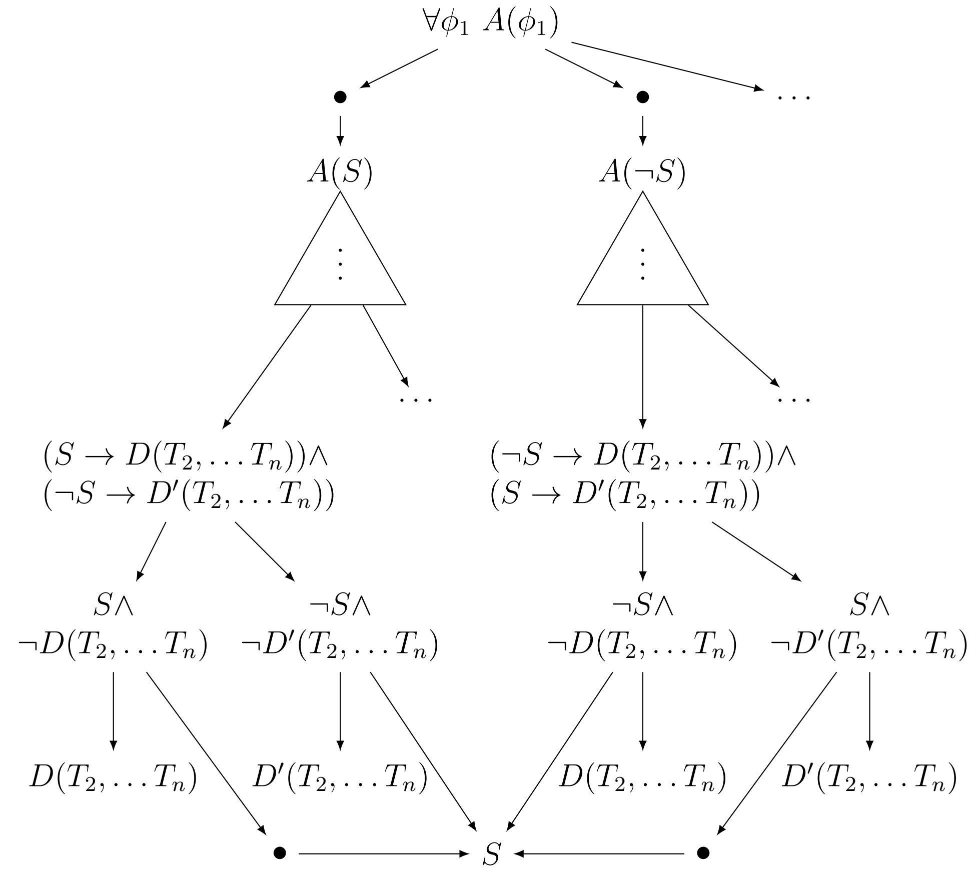

Figure 5: S's value does not contribute to the valuation of the subgraph.

Proof. We consider only $A = \forall\phi_1 \text{Ɐ}\phi_2 \dots \text{Ɐ}\phi_n\ D(\phi_1, \dots \phi_n)$, while the proof for $\text{Ɐ}\phi_1 = \exists\phi_1$ goes essentially the same way.

This proof relies on an induction on the number of quantifiers in A. The base case, $A = \forall\phi_1\ D(\phi_1)$ ($A = \exists\phi_1\ D(\phi_1)$) clearly has a fixed boolean value as the only possible disjuncts in $D(\phi_1)$ are

(1) (ϕ_1)

(2) $(\neg\phi_1)$

(3) $(\phi_1 \wedge \neg\phi_1)$

If among D's disjuncts are both (1) and (2) (either (1) or (2) or both), then $A \Leftrightarrow \top$, otherwise $A \Leftrightarrow \bot$.

For an induction hypothesis, we assume any A with $(n-1)$ quantifiers to have a fixed boolean value.

It remains the induction step for A with n quantifiers.

$$\begin{aligned}
A \Leftrightarrow & \forall\phi_1 \exists\!\!\!\forall\phi_2 \dots \exists\!\!\!\forall\phi_n \ \big[(\phi_1 \wedge C_1(\phi_2, \dots \phi_n)) \vee \dots (\phi_1 \wedge C_c(\phi_2, \dots \phi_n)) \vee && (1) \\
& (\neg\phi_1 \wedge C'_1(\phi_2, \dots \phi_n)) \vee \dots (\neg\phi_1 \wedge C'_{c'}(\phi_2, \dots \phi_n))\big] \\
\Leftrightarrow & \forall\phi_1 \exists\!\!\!\forall\phi_2, \dots \exists\!\!\!\forall\phi_n \ \big[(\phi_1 \wedge (C_1(\phi_2, \dots \phi_n) \vee \dots C_c(\phi_2, \dots \phi_n))) \vee \\
& (\neg\phi_1 \wedge (C'_1(\phi_2, \dots \phi_n) \vee \dots C'_{c'}(\phi_2, \dots \phi_n)))\big] \\
\Leftrightarrow & \forall\phi_1 \exists\!\!\!\forall\phi_2 \dots \exists\!\!\!\forall\phi_n \ \big[(\phi_1 \rightarrow (C_1(\phi_2, \dots \phi_n) \vee \dots C_c(\phi_2, \dots \phi_n))) \wedge \\
& (\neg\phi_1 \rightarrow (C'_1(\phi_2, \dots \phi_n) \vee \dots C'_{c'}(\phi_2, \dots \phi_n)))\big] \\
\Leftrightarrow & \forall\phi_1 \exists\!\!\!\forall\phi_2 \dots \exists\!\!\!\forall\phi_n \ \big[(\phi_1 \rightarrow D(\phi_2, \dots \phi_n)) \wedge && (2) \\
& (\neg\phi_1 \rightarrow D'(\phi_2, \dots \phi_n))\big] \\
\Leftrightarrow & \quad \exists\!\!\!\forall\phi_2 \dots \exists\!\!\!\forall\phi_n \ D(\phi_2, \dots \phi_n) \wedge && (3) \\
& \quad \exists\!\!\!\forall\phi_2 \dots \exists\!\!\!\forall\phi_n \ D'(\phi_2, \dots \phi_n)
\end{aligned}$$

By Lemma 3.7 we can assume exactly one sentential occurrence in each disjunct of A's PDNF and by assumption no other occurrences (1). To (2) we get by trivial propositional laws. Lemma 3.13 allows to eliminate ϕ_1 entirely from the sentence (3).

Thus A is equivalent to some sentence with $(n-1)$ quantifiers, completing the induction step. So any A with only sentential variables has a fixed boolean value. $\square$

This chapter and most of the proven results were of course concerning the operator syntactic equality $\doteq$. However, the observation that only nominal *and* sentential occurrences of variables together are problematic, is in fact applicable to any operator. Indeed, the previous facts did not rely on the fact the language contained the operator $\doteq$, thus hold for any (number of) arbitrary operators.

We have now certainty that sentences with variables in only nominal or only sentential position do not cause paradoxes; the question remains if arbitrary sentences permit consistency as well.

3.4.2 A possible induction

Turning our eye towards arbitrary sentences, one way of addressing the infinite number of sentences with multiple quantifiers could be an induction on the number of quantifiers occuring. If we could show that any sentence with n quantifiers is semantically equivalent to one with $(n-1)$ quantifiers, we would find for any sentence an equivalent sentence with 1 quantifier, which would be the base case and which we have treated in section 3.3.

This approach is not sufficient, however. Theorem 3.10 shows for sentences with one quantifier an equivalence to nominal subsentences which in turn may have an arbitrary number of quantifiers again. Thus, when for an induction hypothesis assuming the claim for sentences with $(n-1)$ quantifiers, this hypothesis is hardly ever applicable when already in the base case the number of quantifiers might increase.

Broadening the scope of what Theorem 3.10 actually guarantees gives a potentially more promising induction. We could start adding up the total number of quantifiers occuring in the sentence, both the "outer" quantifiers that actually quantify the sentence and the "inner" quantifiers which occur with some sentence in nominal position.

Definition 3.15 (Quantifier-total, inner and outer quantifiers).
For any formula F, the quantifier-total $qt(F) = t$, iff F contains exactly t quantifiers as symbols.

$F = \text{Ɐ}\phi_1 \dots \text{Ɐ}\phi_n D(\phi_1 \dots \phi_n)$, being in PDNF, is said to have n outer quantifiers, $qo(F) = n$. The number of inner quantifiers of F is $qi(F) = qt(F) - qo(F) = t - n$.

With this new definition we get a new perspective on what Theorem 3.10 could accomplish, namely reducing the number of total quantifiers.

Lemma 3.16.
For any sentence A with one outer quantifier $qo(A) = 1$ there exists an equivalent sentence $A' \Leftrightarrow A$ such that $qt(A') < qt(A)$.

Proof. Let A be an arbitrary sentence with $qo(A) = 1$ and $qi(A) = i$ and so $qt(A) = i + 1$. By Theorem 3.10, $A \Leftrightarrow [\neg]T_1 \wedge \dots [\neg]T_t$ (or $A \Leftrightarrow \top/\bot$), with each T_n nominal in A. Thus $qt([\neg]T_1 \wedge \dots [\neg]T_t) \leq qi(A) < qt(A)$. □

A possible induction over the quantifier-total qt of sentences lies now at hand. The base case with $qt(A) = 0$ is trivial.

Base case 3.17.
For every valuation of object language, any sentence A with $qt(A) = 0$ obtains a fixed value $A \Leftrightarrow \top/\bot$.

Proof. If $qt(A) = 0$ then $qo(A) = 0$. Thus A is a propositional matrix of atomic sentences ($X \doteq Y$). These obtain fixed values which fixes the value of A. □

The induction hypothesis assumes the claim for sentences with $qt(A) = n - 1$ and is investigated in the rest of the work.

Induction hypothesis 3.18.
For an induction hypothesis we assume a fixed valuation of object language and that for any sentence A with $qt(A) \leq (n-1)$, A has a fixed value $A \Leftrightarrow \top/\bot$.

Having established a base case, it remains to check if by assumption of the induction hypothesis, the induction step can be accomplished. This is possible for various special cases.

3.4.3 Induction step for sentences with one block of quantifiers

First of all, with help of the following Lemma 3.19, we can prove the induction step for Σ_1- and Π_1-sentences.

Lemma 3.19.
For any disjunction of $\doteq$-atoms, each having bare ϕ_i on its left-hand side, i.e.

$$D(\phi_1, \dots \phi_i, \dots \phi_n) = (\phi_i \doteq S_1(\phi_1, \dots \phi_n)) \vee \dots (\phi_i, \doteq S_s(\phi_1, \dots \phi_n))$$

there are assignments $\phi_i \mapsto T$ and $\phi_i \mapsto T'$ such that $D(\phi_1, \dots T, \dots \phi_n) \Leftrightarrow D(\phi_1, \dots T', \dots \phi_n) \Leftrightarrow \bot$ and $T \Leftrightarrow \top$ and $T' \Leftrightarrow \bot$.

Proof. Each $S_i(\phi_1 \dots \phi_n)$ is either a conjunction, a negation, or starting with a number m quantifiers, i.e.

- $S_i(\phi_1, \dots \phi_n) = F \wedge F'$ or
- $S_i(\phi_1, \dots \phi_n) = \neg F$ or
- $S_i(\phi_1, \dots \phi_n) = \exists\!\!\!\forall \phi_{x_1} \dots \exists\!\!\!\forall \phi_{x_m} F$

for some formulas F, F'.

Thus any $S_j(\phi_1, \dots \phi_n)$ starts with either 0 or m quantifiers and choosing for ϕ_i a sentence starting with more than the maximum of quantifiers m suffices to falsify every instance of $\phi_i \doteq S_j(\phi_1, \dots \phi_n)$.

Therefore if

$$T = \forall\phi_1 \dots \forall\phi_{m+1}(\phi_1 \vee \neg\phi_1 \vee \dots \phi_{m+1} \vee \neg\phi_{m+1}) \Leftrightarrow \top$$

and

$$T' = \forall\phi_1 \dots \forall\phi_{m+1}(\phi_1 \wedge \neg\phi_1 \wedge \dots \phi_{m+1} \wedge \neg\phi_{m+1}) \Leftrightarrow \bot$$

then $D(\phi_1, \dots T, \dots \phi_n) \Leftrightarrow D(\phi_1, \dots T', \dots \phi_n) \Leftrightarrow \bot$. □

Example 3.20.
Let us for an example of Lemma 3.19 take a disjunction $D(\phi_1, \phi_2, \phi_3)$ with ϕ_1 alone in nominal position.

$$\begin{aligned} D(\phi_1, \phi_2, \phi_3) =& (\phi_1 \doteq (X \wedge (\phi_2 \to \phi_3))) \vee (\phi_1 \doteq \forall\phi_1\forall\phi_2(\phi_1 \wedge \phi_2)) \vee \\ & (\phi_1 \doteq \exists\phi_1(\phi_2 \wedge \phi_1)) \vee (\phi_1 \doteq \neg\forall\phi_1(\phi_1 \vee \phi_3)) \end{aligned}$$

The maximum number of quantifiers heading the sentences is 2, hence sentences with 3 leading quantifiers for ϕ_1 falsify all the equalities. Take $T = \forall\phi_1\forall\phi_2\forall\phi_3(\phi_1 \vee \neg\phi_1 \vee \phi_2 \vee \neg\phi_2 \vee \phi_3 \vee \neg\phi_3)$ and $T' = \forall\phi_1\forall\phi_2\forall\phi_3(\phi_1 \wedge \neg\phi_1 \wedge \phi_2 \wedge \neg\phi_2 \wedge \phi_3 \wedge \neg\phi_3)$. Clearly T is a tautology and T' is a contradiction.

$$\begin{aligned} D(S, \phi_2, \phi_3) =& (T \doteq (X \wedge (\phi_2 \to \phi_3))) \vee (T \doteq \forall\phi_1\forall\phi_2(\phi_1 \wedge \phi_2)) \vee \\ & (T \doteq \exists\phi_1(\phi_2 \wedge \phi_1)) \vee (T \doteq \neg\forall\phi_1(\phi_1 \vee \phi_3)) \\ \Leftrightarrow & \bot \vee \bot \vee \bot \vee \bot \\ \Leftrightarrow & \bot \end{aligned}$$

It is easy to verify that T (as well as T') falsifies the disjunction, no matter what sentences would be substituted for ϕ_2 and ϕ_3.

Next, we define the type of sentences that we want to prove the induction step for, Σ_1/Π_1-sentences.

Definition 3.21 (PDNF hierarchy)**.**
A formula F in PDNF is a

- Σ_1*-formula, iff* $F = \exists\phi_1 \dots \exists\phi_n \ P(\phi_1, \dots \phi_n)$ *and* $qo(P) = 0$*,*
- Π_1*-formula, iff* $F = \forall\phi_1 \dots \forall\phi_n \ P(\phi_1, \dots \phi_n)$ *and* $qo(P) = 0$*,*
- Σ_{i+1}*-formula, iff* $F = \exists\phi_1 \dots \exists\phi_n \ P(\phi_1, \dots \phi_n)$ *and* P *is a* Π_i*-formula,*
- Π_{i+1}*-formula, iff* $F = \forall\phi_1 \dots \forall\phi_n \ P(\phi_1, \dots \phi_n)$ *and* P *is a* Σ_i*-formula.*

S is a Σ_i/Π_i-sentence, iff S is a sentence and a Σ_i/Π_i-formula.

Theorem 3.22 (Induction step for Σ_1/Π_1)**.**
Under the assumption of the induction hypothesis (3.18), for any Σ_1/Π_1-sentence A with $qt(A) = n$, $A \Leftrightarrow \top/\bot$.

Proof. We assume A to be Π_1; if it is Σ_1, then there is a Π_1-sentence A', such that $A \Leftrightarrow \neg A'$ and finding a value for A' yields a value for A.

By Lemma 3.8 we know that $A \Leftrightarrow A^0 \wedge \dots A^{2^n-1}$ where

$$A^i = \forall\phi_1 \dots \forall\phi_n \ (([\neg]\phi_1 \wedge \dots [\neg]\phi_n) \rightarrow D^i(\phi_1, \dots \phi_n))$$

with $D^i(\phi_1, \dots \phi_n)$ having no variables in sentential positions. Finding values for all A^i yields a fixed value for A, so we turn our attention to A^0 and make sure any other A^i follows alongside. We transform $D^i(\phi_1, \dots \phi_n)$ into an equivalent CNF $D_1(\phi_1, \dots \phi_n) \wedge \dots D_d(\phi_1, \dots \phi_n)$ and apply distributivity to get

$$\begin{aligned}
A^0 =& \forall\phi_1 \dots \forall\phi_n \ \left[(\neg\phi_1 \wedge \dots \neg\phi_n) \rightarrow D^i(\phi_1, \dots \phi_n)\right] \\
\Leftrightarrow& \forall\phi_1 \dots \forall\phi_n \ \left[(\neg\phi_1 \wedge \dots \neg\phi_n) \rightarrow (D_1(\phi_1, \dots \phi_n) \wedge \dots D_d(\phi_1, \dots \phi_n))\right] \\
\Leftrightarrow& \forall\phi_1 \dots \forall\phi_n \ \big[((\neg\phi_1 \wedge \dots \neg\phi_n) \rightarrow D_1(\phi_1, \dots \phi_n)) \wedge \dots \\
& \qquad ((\neg\phi_1 \wedge \dots \neg\phi_n) \rightarrow D_d(\phi_1, \dots \phi_n))\big] \\
\Leftrightarrow& \forall\phi_1 \dots \forall\phi_n \ \left[((\neg\phi_1 \wedge \dots \neg\phi_n) \rightarrow D_1(\phi_1, \dots \phi_n))\right] \wedge \dots \\
& \forall\phi_1 \dots \forall\phi_n \ \left[((\neg\phi_1 \wedge \dots \neg\phi_n) \rightarrow D_d(\phi_1, \dots \phi_n))\right] \\
=& A^0_1 \wedge \dots A^0_d
\end{aligned}$$

Having split A^0 into simpler sentences $A^0_1, \dots A^0_d$ we again only focus on one A^0_i, knowing that a value for each A^0_i yields a fixed value for A^0. To make sure we are still eligible to apply the induction hypothesis 3.18, a quick look at the applied Lemma 3.8 as well as the applied distributivity laws suffices to see that $qi(A^0_i) \leq qi(A^0) \leq qi(A)$ (while $qo(A^0_i) = qo(A^0) = qo(A) = n$).

$$\begin{aligned}A_i^0 =&\forall\phi_1 \dots \forall\phi_n\ ((\neg\phi_1 \wedge \dots \neg\phi_n) \rightarrow D_i(\phi_1, \dots \phi_n))\\ \Leftrightarrow&\forall\phi_1 \dots \forall\phi_n\ (\phi_1 \vee \dots \phi_n \vee L_1(\phi_1, \dots \phi_n) \vee \dots L_l(\phi_1, \dots \phi_n))\end{aligned}$$

As per Lemma 3.8, each $L(\phi_1, \dots \phi_n)$ represents a literal, but not a variable. We can also assume that Lemma 3.5 had been applied in the beginning and thus every $L(\phi_1, \dots \phi_n) = [\neg](\phi_i \doteq S(\phi_1, \dots \phi_{i-1}, \phi_{i+1}, \dots \phi_n))$. Seeing that L can occur negated and non-negated gives a case distinction.

- If there is a negated literal $L(\phi_1, \dots \phi_i, \dots \phi_n) = (\phi_i \not\doteq S(\phi_1, \dots \phi_{i-1}, \phi_{i+1}, \dots \phi_n))$, then we can rearrange A_i^0 and then replace all instances of ϕ_i with the sentence it is required to equal to:

$$
\begin{aligned}
A_i^0 \Leftrightarrow & \forall\phi_1 \ldots \forall\phi_i \ldots \forall\phi_n \, \big[\phi_1 \vee \ldots \phi_i \vee \ldots \phi_n \vee \\
& \phi_i \not\doteq S(\phi_1, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_n) \vee \\
& L_1(\phi_1, \ldots \phi_i, \ldots \phi_n) \vee \ldots L_l(\phi_1, \ldots \phi_i, \ldots \phi_n)\big] \\
\Leftrightarrow & \forall\phi_1 \ldots \forall\phi_i \ldots \forall\phi_n \, \big[(\phi_i \doteq S(\phi_1, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_n)) \rightarrow & (1) \\
& (\phi_1 \vee \ldots \phi_i \vee \ldots \phi_n \vee \\
& L_1(\phi_1, \ldots \phi_i, \ldots \phi_n) \vee \ldots L_l(\phi_1, \ldots \phi_i, \ldots \phi_n))\big] \\
\Leftrightarrow & \forall\phi_1 \ldots \forall\phi_i \ldots \forall\phi_n \, \big[(\phi_i \doteq S(\phi_1, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_n)) \rightarrow & (2) \\
& (\phi_1 \vee \ldots \phi_{i-1} \vee S(\phi_1, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_n) \vee \phi_{i+1} \vee \ldots \phi_n \vee \\
& L_1(\phi_1, \ldots \phi_{i-1}, S(\phi_1, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_n), \phi_{i+1} \ldots \phi_n) \vee \ldots \\
& L_l(\phi_1, \ldots \phi_{i-1}, S(\phi_1, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_n), \phi_{i+1}, \ldots \phi_n))\big] \\
\Leftrightarrow & \forall\phi_1 \ldots \forall\phi_{i-1} \forall\phi_{i+1} \ldots \forall\phi_n \, \big[& (3) \\
& \forall\phi_i(\phi_i \not\doteq S(\phi_1, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_n)) \vee \\
& (\phi_1 \vee \ldots \phi_{i-1} \vee S(\phi_1, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_n) \vee \phi_{i+1} \vee \ldots \phi_n \vee \\
& L_1(\phi_1, \ldots \phi_{i-1}, S(\phi_1, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_n), \phi_{i+1}, \ldots \phi_n) \vee \ldots \\
& L_l(\phi_1, \ldots \phi_{i-1}, S(\phi_1, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_n), \phi_{i+1}, \ldots \phi_n))\big] \\
\Leftrightarrow & \forall\phi_1 \ldots \forall\phi_{i-1} \forall\phi_{i+1} \ldots \forall\phi_n \, \big[\bot \vee & (4) \\
& (\phi_1 \vee \ldots \phi_{i-1} \vee S(\phi_1, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_n) \vee \phi_{i+1} \vee \ldots \phi_n \vee \\
& L_1(\phi_1, \ldots \phi_{i-1}, S(\phi_1, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_n), \phi_{i+1}, \ldots \phi_n) \vee \ldots \\
& L_l(\phi_1, \ldots \phi_{i-1}, S(\phi_1, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_n), \phi_{i+1}, \ldots \phi_n))\big] \\
\Leftrightarrow & \forall\phi_1 \ldots \forall\phi_{i-1} \forall\phi_{i+1} \ldots \forall\phi_n \, \big[& (5) \\
& \phi_1 \vee \ldots \vee \phi_{i-1} \vee S(\phi_1, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_n) \vee \phi_{i+1} \vee \ldots \phi_n \vee \\
& L_1(\phi_1, \ldots \phi_{i-1}, S(\phi_1, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_n), \phi_{i+1}, \ldots \phi_n) \vee \ldots \\
& L_l(\phi_1, \ldots \phi_{i-1}, S(\phi_1, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_n), \phi_{i+1}, \ldots \phi_n)\big] \\
= & A'
\end{aligned}
$$

To get (1), we transform the disjunction into an implication. We then replace all ϕ_i in the consequent with the formula it is required to equal (2). Pulling the quantifier $\forall\phi_i$ inside yields subsentence $\forall\phi_i(\phi_i \not\doteq S(\phi_1, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_n))$ (3); no matter what sentences are substituted for $\phi_1 \ldots \phi_n$, if we take

$$\phi_i = S(\phi_1, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_n)$$

which is clearly a sentence, our subsentence is false (4) and can hence be eliminated entirely (5).

We have thereby eliminated $\forall\phi_i$ from the sentence. By replacing the sentential ϕ_i with the nominal $S(\phi_1 \ldots \phi_n)$ some quantifiers may have shifted from inner to outer, but the total number has been reduced by one.

$qi(A') = qi(A_i^0) - x$ and $qo(A') = qo(A_i^0) - 1 + x$ and thus $qt(A') = qi(A') + qo(A') = qi(A_i^0) - x + qo(A_i^0) - 1 + x = qi(A_i^0) + qo(A_i^0) - 1 = qt(A_i^0) - 1 < qt(A_i^0)$.

We can therefore apply induction hypothesis 3.18 and $A' \Leftrightarrow \top/\bot$.

- The other case is if there is no negated literal, thus all literals are positive $L(\phi_1, \ldots \phi_i, \ldots \phi_n) = (\phi_i \doteq S(\phi_1, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_n))$. It is then possible to find an assignment $\phi_1 \mapsto T_1, \ldots \phi_n \mapsto T_n$ such that $A_i^0 \Leftrightarrow \bot$. To find this assignment, we choose any ϕ_i occurring bare in the left-hand side of some $\doteq$-atoms, $(\phi_i \doteq S_1(\phi_1, \ldots \phi_n)) \vee \ldots (\phi_i \doteq S_1(\phi_1, \ldots \phi_n))$. Lemma 3.19 allows us to find an assignment $\phi_i \mapsto T_i$, such that all these atoms are false and T_i itself is false as well. Substituting this T_i for ϕ_i in A_i^0 yields a new sentence.

$$\begin{aligned}
A_i' =& \forall\phi_1 \ldots \forall\phi_i \ldots \forall\phi_n \left[\phi_1 \vee \ldots \phi_i \ldots \phi_n \vee\right. \\
& (\phi_i \doteq S_1(\phi_1, \ldots \phi_n)) \vee \ldots (\phi_i \doteq S_s(\phi_1, \ldots \phi_n)) \\
& \left.\vee D(\phi_1, \ldots \phi_i, \ldots \phi_n)\right] \\
\Rightarrow& \forall\phi_1 \ldots \forall\phi_n \left[\phi_1 \vee \ldots T_i \vee \ldots \phi_n \vee\right. \\
& (T_i \doteq S_1(\phi_1, \ldots \phi_n)) \vee \ldots (T_i \doteq S_1(\phi_1, \ldots \phi_n)) \\
& \left.\vee D(\phi_1, \ldots T_i, \ldots \phi_n)\right] \\
\Rightarrow& \forall\phi_1 \ldots \forall\phi_n \left[\phi_1 \vee \ldots \bot \vee \ldots \phi_n \vee\right. \\
& \bot \vee \ldots \bot \vee \\
& \left.D(\phi_1, \ldots T_i, \ldots \phi_n)\right] \\
\Rightarrow& \forall\phi_1 \ldots \forall\phi_n \left[\phi_1 \vee \ldots \phi_n \vee D(\phi_1, \ldots T_i, \ldots \phi_n)\right]
\end{aligned}$$

This process can be repeated until all bare variables and atoms have been eliminated leaving the empty disjunction. We have thus found an assignment for all $\phi_1 \dots \phi_n$ making the disjunction empty and so witnesses for $A_i^0 \Leftrightarrow \bot$.

$\square$

3.4.4 Induction step for sentences with a variable only in sentential position

Having proven the induction step for the special case of Σ_1/Π_1-sentences, there are more special cases, for which the induction step can be shown: for instance, sentences with a variable ϕ_i occuring only in sentential position.

Theorem 3.23.
Under the assumption of the induction hypothesis (3.18), for any sentence $A = \exists\!\!\!\forall \phi_1 \dots \forall \phi_i \dots \exists\!\!\!\forall \phi_n (D(\phi_1, \dots \phi_i, \dots \phi_n))$ with ϕ_i occuring only in sentential position, $A \Leftrightarrow \top/\bot$.

Proof.

$$\begin{aligned}
A \Leftrightarrow\ & \exists\!\!\!\forall \phi_1 \dots \forall \phi_i \dots \exists\!\!\!\forall \phi_n \big[D(\phi_1, \dots \phi_i, \dots \phi_n)\big] \\
\Leftrightarrow\ & \exists\!\!\!\forall \phi_1 \dots \forall \phi_i \dots \exists\!\!\!\forall \phi_n \big[(\phi_i \wedge D_1(\phi_1, \dots \phi_n)) \vee \dots && (1) \\
& \qquad (\phi_i \wedge D_d(\phi_1, \dots \phi_n)) \vee \\
& \qquad (\neg\phi_i \wedge D'_1(\phi_1, \dots \phi_n)) \vee \dots \\
& \qquad (\neg\phi_i \wedge D'_{d'}(\phi_1, \dots \phi_n))\big] \\
\Leftrightarrow\ & \exists\!\!\!\forall \phi_1 \dots \forall \phi_i \dots \exists\!\!\!\forall \phi_n \big[(\phi_i \rightarrow D(\phi_1, \dots \phi_n)) \wedge && (2) \\
& \qquad (\neg\phi_i \rightarrow D'(\phi_1, \dots \phi_n))\big] \\
\Leftrightarrow\ & \exists\!\!\!\forall \phi_1 \dots \exists\!\!\!\forall \phi_{i-1} \big[\quad (\exists\!\!\!\forall \phi_{i+1} \dots \exists\!\!\!\forall \phi_n D(\phi_1, \dots \phi_n)) \wedge && (3) \\
& \qquad (\exists\!\!\!\forall \phi_{i+1} \dots \exists\!\!\!\forall \phi_n D'(\phi_1, \dots \phi_n))\big] \\
=\ & A'
\end{aligned}$$

By Lemma 3.7, we can assume exactly one sentential occurrence of ϕ_i in each PDNF-disjunct (1). With distributivity (2) and Lemma 3.13, ϕ_i can be eliminated entirely from the sentence (3), yielding a new sentence A'.

So not only $A' \Leftrightarrow A$ but also $qi(A') = qi(A)$ and $qo(A') = qo(A) - 1$ and thus $qt(A') = qi(A') + qo(A') = qi(A) + qo(A) - 1 = qt(A) - 1 < qt(A)$. Having found an equivalent A' with less quantifiers allows application of the induction hypothesis and therefore $A \Leftrightarrow \top/\bot$. $\square$

3.4.5 Induction step for sentences with the innermost quantified variable alone in nominal positions

For another special case of sentences, the induction step can be shown. If a sentence with the innermost quantified variable ϕ_n ($\exists\!\!\!\forall\phi_n$ in $\exists\!\!\!\forall\phi_1\exists\!\!\!\forall\phi_2\dots\exists\!\!\!\forall\phi_n\ D(\phi_1,\phi_2,\dots\phi_n)$) has ϕ_n in nominal positions, but always as a bare variable, thus all operators containing ϕ_n have the form $\phi_n, \doteq \phi_i$, $\phi_n, \doteq S$ or $\phi_n \doteq S(\phi_1,\dots\phi_{n-1})$ (and never $\phi_i, \doteq S(\phi_1\dots\phi_n)$), then we can find a boolean value for the sentence.

Theorem 3.24.
Under the assumption of the induction hypothesis (3.18) for any sentence $A = \exists\!\!\!\forall\phi_1\dots\exists\!\!\!\forall\phi_n(D(\phi_1,\dots\phi_n))$*, if all* $\doteq$*-atoms containing* ϕ_n *are of the form* $\phi_n \doteq S(\phi_1,\dots\phi_{n-1})$ *(*ϕ_n *may still occur in sentential position), then* $A \Leftrightarrow \top/\bot$*.*

Proof. The proof relies on Lemma 3.19 and propositional equivalences. By Lemma 3.7 we assume exactly one sentential occurrence of ϕ_n in every disjunct. Without loss of generality, we assume $\exists\!\!\!\forall\phi_n = \forall\phi_n$ (if $\exists\!\!\!\forall\phi_n = \exists\phi_n$, then there is A' with $A \Leftrightarrow \neg A'$ with $\exists\!\!\!\forall\phi_n = \forall\phi_n$ in A', while a fixed value of A' fixes A's value).

$$
\begin{aligned}
A =\ & \exists\!\!\!\forall\phi_1\dots\exists\!\!\!\forall\phi_{n-1}\forall\phi_n\big[D(\phi_1,\dots\phi_n)\big]\\
\Leftrightarrow\ & \exists\!\!\!\forall\phi_1\dots\exists\!\!\!\forall\phi_{n-1}\forall\phi_n\big[(\phi_n\wedge C_1(\phi_1,\dots\phi_n))\vee\dots(\phi_n\wedge C_c(\phi_1,\dots\phi_n))\vee\\
& \qquad (\neg\phi_n\wedge C'_1(\phi_1,\dots\phi_n))\vee\dots(\neg\phi_n\wedge C'_{c'}(\phi_1,\dots\phi_n))\big]\\
\Leftrightarrow\ & \exists\!\!\!\forall\phi_1\dots\exists\!\!\!\forall\phi_{n-1}\forall\phi_n\big[(\phi_n\wedge(C_1(\phi_1,\dots\phi_n)\vee\dots C_c(\phi_1,\dots\phi_n)))\vee\\
& \qquad (\neg\phi_n\wedge(C'_1(\phi_1,\dots\phi_n)\vee\dots C'_{c'}(\phi_1,\dots\phi_n)))\big]\\
\Leftrightarrow\ & \exists\!\!\!\forall\phi_1\dots\exists\!\!\!\forall\phi_{n-1}\forall\phi_n\big[(\phi_n\rightarrow(C_1(\phi_1,\dots\phi_n)\vee\dots C_c(\phi_1,\dots\phi_n)))\wedge\\
& \qquad (\neg\phi_n\rightarrow(C'_1(\phi_1,\dots\phi_n)\vee\dots C'_{c'}(\phi_1,\dots\phi_n)))\big]\\
\Leftrightarrow\ & \exists\!\!\!\forall\phi_1\dots\exists\!\!\!\forall\phi_{n-1}\big[\quad \forall\phi_n\ (\phi_n\rightarrow(C_1(\phi_1,\dots\phi_n)\vee\dots C_c(\phi_1,\dots\phi_n)))\wedge\\
& \qquad \forall\phi_n\ (\neg\phi_n\rightarrow(C'_1(\phi_1,\dots\phi_n)\vee\dots C'_{c'}(\phi_1,\dots\phi_n)))\big]
\end{aligned}
$$

The equivalences are justified by trivial propositional laws and distributivity. The DNF $C_1(\phi_1,\dots\phi_n)\vee\dots C_c(\phi_1,\dots\phi_n)$ is transformed into CNF $D_1(\phi_1,\dots\phi_n)\wedge\dots D_d(\phi_1,\dots\phi_n)$ and the quantifier $\forall\phi_n$ distributed.

$$
\begin{aligned}
A \Leftrightarrow\ & \exists\!\!\!\forall\phi_1 \ldots \exists\!\!\!\forall\phi_{n-1}\big[\forall\phi_n(\phi_n \to (C_1(\phi_1,\ldots\phi_n) \vee \ldots C_c(\phi_1,\ldots\phi_n)))\wedge \\
& \qquad\qquad\qquad \forall\phi_n(\neg\phi_n \to (C'_1(\phi_1,\ldots\phi_n) \vee \ldots C'_{c'}(\phi_1,\ldots\phi_n)))\big] \\
\Leftrightarrow\ & \exists\!\!\!\forall\phi_1 \ldots \exists\!\!\!\forall\phi_{n-1}\big[\forall\phi_n(\phi_n \to (D_1(\phi_1,\ldots\phi_n) \wedge \ldots D_d(\phi_1,\ldots\phi_n)))\wedge \\
& \qquad\qquad\qquad \forall\phi_n(\neg\phi_n \to (D'_1(\phi_1,\ldots\phi_n) \wedge \ldots D'_{d'}(\phi_1,\ldots\phi_n)))\big] \\
\Leftrightarrow\ & \exists\!\!\!\forall\phi_1 \ldots \exists\!\!\!\forall\phi_{n-1}\big[\forall\phi_n((\phi_n \to D_1(\phi_1,\ldots\phi_n)) \wedge \ldots \\
& \qquad\qquad\qquad (\phi_n \to D_d(\phi_1,\ldots\phi_n)))\wedge \\
& \qquad\qquad\qquad \forall\phi_n((\neg\phi_n \to D'_1(\phi_1,\ldots\phi_n)) \wedge \ldots \\
& \qquad\qquad\qquad (\neg\phi_n \to D'_{d'}(\phi_1,\ldots\phi_n)))\big] \\
\Leftrightarrow\ & \exists\!\!\!\forall\phi_1 \ldots \exists\!\!\!\forall\phi_{n-1}\big[\forall\phi_n(\phi_n \to D_1(\phi_1,\ldots\phi_n)) \wedge \ldots \\
& \qquad\qquad\qquad \forall\phi_n(\phi_n \to D_d(\phi_1,\ldots\phi_n))\wedge \\
& \qquad\qquad\qquad \forall\phi_n(\neg\phi_n \to D'_1(\phi_1,\ldots\phi_n)) \wedge \ldots \\
& \qquad\qquad\qquad \forall\phi_n(\neg\phi_n \to D'_{d'}(\phi_1,\ldots\phi_n))\big]
\end{aligned}
$$

Thus, for A to hold, no matter what has been substituted for ϕ_1 to ϕ_{n-1}, all subsentences $\forall\phi_n([\neg]\phi_n \to D_i(\phi_1,\ldots\phi_n))$ have to hold.

Each $D_i(\phi_1,\ldots\phi_n)$ is a disjunction of literals $L(\phi_1,\ldots\phi_n)$ which may or may not contain ϕ_n (let $D'(\phi_1,\ldots\phi_{n-1})$ be the disjunction of all literals without ϕ_n).

$$
\begin{aligned}
& \forall\phi_n\big[\neg\phi_n \to D_i(\phi_1,\ldots\phi_n)\big] \\
\Leftrightarrow\ & \forall\phi_n\big[\phi_n \vee D_i(\phi_1,\ldots\phi_n)\big] \\
\Leftrightarrow\ & \forall\phi_n\big[\phi_n \vee L_1(\phi_1,\ldots\phi_n) \vee \ldots L_l(\phi_1,\ldots\phi_n)\big] \vee D'(\phi_1,\ldots\phi_{n-1})
\end{aligned}
$$

Considering all literals, by assumption, we know that they all are of the form $\phi_n \doteq S(\phi_1,\ldots\phi_{n-1})$ or $\phi_n \not\doteq S(\phi_1,\ldots\phi_{n-1})$ or don't contain ϕ_n.

- If one of the literals is negated, $L(\phi_1,\ldots\phi_n) = (\phi_n \not\doteq S(\phi_1,\ldots\phi_{n-1}))$, then

$$
\begin{aligned}
&\forall\phi_n\big[(\phi_n \mathrel{\dot{\neq}} S(\phi_1,\dots\phi_{n-1})) \vee \phi_n \\
&\qquad \vee L_2(\phi_1,\dots\phi_n) \vee \dots L_l(\phi_1,\dots\phi_n)\big] \\
\Leftrightarrow &\forall\phi_n\big[(\phi_n \doteq S(\phi_1,\dots\phi_{n-1})) \rightarrow (\phi_n \vee && (1)\\
&\qquad L_2(\phi_1,\dots\phi_n) \vee \dots L_l(\phi_1,\dots\phi_n))\big] \\
\Leftrightarrow &\forall\phi_n\big[(\phi_n \doteq S(\phi_1,\dots\phi_{n-1})) \rightarrow (S(\phi_1,\dots\phi_{n-1}) \vee && (2)\\
&\qquad L_2(\phi_1,\dots\phi_{n-1} S(\phi_1,\dots\phi_{n-1})) \vee \dots \\
&\qquad L_l(\phi_1,\dots\phi_{n-1} S(\phi_1,\dots\phi_{n-1})))\big] \\
\Leftrightarrow &\forall\phi_n\big[(\phi_n \mathrel{\dot{\neq}} S(\phi_1,\dots\phi_{n-1})) \vee S(\phi_1,\dots\phi_{n-1}) \vee \\
&\qquad L_2(\phi_1,\dots\phi_{n-1} S(\phi_1,\dots\phi_{n-1})) \vee \dots \\
&\qquad L_l(\phi_1,\dots\phi_{n-1} S(\phi_1,\dots\phi_{n-1}))\big] \\
\Leftrightarrow &\forall\phi_n\big[\phi_n \mathrel{\dot{\neq}} S(\phi_1,\dots\phi_{n-1})\big] \vee S(\phi_1,\dots\phi_{n-1}) \vee && (3)\\
&\qquad L_2(\phi_1,\dots\phi_{n-1} S(\phi_1,\dots\phi_{n-1})) \vee \dots \\
&\qquad L_l(\phi_1,\dots\phi_{n-1} S(\phi_1,\dots\phi_{n-1})) \\
\Leftrightarrow &\qquad \bot \vee S(\phi_1,\dots\phi_{n-1}) \vee && (4)\\
&\qquad L_2(\phi_1,\dots\phi_{n-1} S(\phi_1,\dots\phi_{n-1})) \vee \dots \\
&\qquad L_l(\phi_1,\dots\phi_{n-1} S(\phi_1,\dots\phi_{n-1})) \\
\Leftrightarrow &\qquad S(\phi_1,\dots\phi_{n-1}) \vee && (5)\\
&\qquad L_2(\phi_1\dots\phi_{n-1} S(\phi_1,\dots\phi_{n-1})) \vee \dots \\
&\qquad L_l(\phi_1,\dots\phi_{n-1} S(\phi_1,\dots\phi_{n-1}))
\end{aligned}
$$

We can transform the sentence into an implication (1), to then replace all occurrences of ϕ_n in the consequent by $S(\phi_1 \dots \phi_{n-1})$ (2). The created subsentence $\forall\phi_n$ $(\phi_n \mathrel{\dot{\neq}} S(\phi_1 \dots \phi_{n-1}))$ (3) is false (as some ϕ_n will certainly satisfy the equality) (4) and thus ϕ_n can be eliminated entirely (5).

Thus the whole subsentence has an equivalent sentence without ϕ_n.

- If none of the literals are negated, we can, by Lemma 3.19 find some false sentence T for which, when substituted for ϕ_n, the whole subsentence is equivalent to $\bot$, thus making the subsentence false.

$$\begin{aligned}
&\forall \phi_n(\phi_n \vee L_1(\phi_1, \ldots \phi_n) \vee \ldots L_l(\phi_1, \ldots \phi_n))\\
\Rightarrow&(T \vee L_1(\phi_1, \ldots \phi_{n-1}, T) \vee \ldots L_l(\phi_1, \ldots \phi_{n-1}, T))\\
\Rightarrow&\bot
\end{aligned}$$

Analogously we can find some true sentence $T \Leftrightarrow \top$ in case we have $\neg\phi_n$. Thus, the subsentence with no negated literals is simply false and hence can be *omitted* (giving raise to a sentence equivalent to A without the particular false subsentence).

The case distinction shows that we can construct $A' \Leftrightarrow A$, where ϕ_n does no longer occur in A'. Obviously then $qt(A') < qt(A)$ and we can apply the induction hypothesis, thus $A \Leftrightarrow \top/\bot$. □

3.4.6 Missing induction step

Having seen a variety of results and having drawn up a possible induction to show consistency for LSO with syntactic equality, the question remains, how close we have come to that goal. LSO sentences can of course be partitioned by their syntactic structure:

- Object language sentences, which are assumed to have a boolean value.
- Sentences without sentential quantifiers, which trivially have a fixed boolean value, being propositional matrices of $S \doteq S'$ instances.
- Sentences with sentential quantifiers, can be further distinguished into
  - Sentences with no quantified variable appearing in sentential positions, showed to have a fixed boolean value in Theorem 3.12.
  - Sentences with no quantified variable appearing in nominal positions, showed to have a fixed boolean value in Theorem 3.14.
  - Sentences with quantified variables in both sentential and nominal position. For these sentences we drew up an induction scheme over the total number of quantifiers occuring in the sentence. The base case is handled in 3.17, and as an induction hypothesis 3.18 is assumed (and actually only necessary for sentences non-conforming with the aforementioned special cases).

- ∗ The induction step was shown for Σ_1/Π_1-sentences in Theorem 3.22.
- ∗ For sentences with some variable only occuring in sentential positions in Theorem 3.23.
- ∗ For sentences with the innermost quantified variable only occuring alone in a nominal position, in Theorem 3.24.
- ∗ It was, however *not* shown, that the induction step holds for any arbitrary sentence, non-conforming with the explicit special cases. Thus, the induction step *being* proven for some special cases says nothing about consistency of these special cases, as they relied on an unproven assumption. Hence, the induction proof is incomplete and until completed, provides merely an intuition, how consistency for the whole language could be proven. Of course the induction proof not being complete does not say that the language *is* inconsistent, but just that inconsistency can not be ruled out with certainty.

Having seen all proven results and their limited applicability, let us, before drawing a close to this work, consider a last example. We construct a sentence that conforms with none of the contemplated special cases and might involve self-reference for a potential paradoxical situation.

Example 3.25.

$$A = \exists\phi_1 \left[\forall\phi_2\Big(\big[(\phi_1 \doteq S(\phi_2)) \vee (\phi_2 \doteq S'(\phi_1))\big] \rightarrow \phi_2\Big) \leftrightarrow \neg\phi_1\right]$$

A is not covered by any of the special cases as

- *it is a Σ_2-sentence,*
- *has both ϕ_1 and ϕ_2 in nominal and sentential positions, as well as*
- *both ϕ_1 and ϕ_2 not alone in one side of $\doteq$-atoms*

Still, along the lines of previous proofs, some known equivalences are applicable. We let $S(\phi_2) = (X \wedge \phi_2)$ and $S'(\phi_1) = (\phi_1 \wedge Y)$ for sentences X and Y.

$$\begin{aligned}
A =&\exists\phi_1 \Big[\ \forall\phi_2\Big(\big[(\phi_1 \doteq (X \wedge \phi_2)) \vee (\phi_2 \doteq (\phi_1 \wedge Y))\big] \to \phi_2\Big) \leftrightarrow \neg\phi_1\Big] \\
\Leftrightarrow&\exists\phi_1 \Big[\ \forall\phi_2\Big(\big[(\phi_1 \not\doteq (X \wedge \phi_2)) \wedge (\phi_2 \not\doteq (\phi_1 \wedge Y))\big] \vee \phi_2\Big) \leftrightarrow \neg\phi_1\Big] && (1) \\
\Leftrightarrow&\exists\phi_1 \Big[\ \forall\phi_2\Big(\big[(\phi_1 \not\doteq (X \wedge \phi_2)) \vee \phi_2\big] \wedge \big[(\phi_2 \not\doteq (\phi_1 \wedge Y)) \vee \phi_2\big]\Big) \leftrightarrow \neg\phi_1\Big] \\
\Leftrightarrow&\exists\phi_1 \Big[\Big(\forall\phi_2\big[(\phi_1 \not\doteq (X \wedge \phi_2)) \vee \phi_2\big] \wedge && (2) \\
&\qquad \forall\phi_2\big[(\phi_2 \not\doteq (\phi_1 \wedge Y)) \vee \phi_2\big]\Big) \leftrightarrow \neg\phi_1\Big] \\
\Leftrightarrow&\exists\phi_1 \Big[\Big(\forall\phi_2\big[(\phi_1 \doteq (X \wedge \phi_2)) \to \phi_2\big] \wedge \\
&\qquad \forall\phi_2\big[(\phi_2 \doteq (\phi_1 \wedge Y)) \to \phi_2\big]\Big) \leftrightarrow \neg\phi_1\Big] \\
\Leftrightarrow&\exists\phi_1 \Big[\Big(\forall\phi_2\big[(\phi_1 \doteq (X \wedge \phi_2)) \to \phi_2\big] \wedge && (3) \\
&\qquad \forall\phi_2\big[(\phi_2 \doteq (\phi_1 \wedge Y)) \to (\phi_1 \wedge Y)\big]\Big) \leftrightarrow \neg\phi_1\Big] \\
\Leftrightarrow&\exists\phi_1 \Big[\Big(\forall\phi_2\big[(\phi_1 \doteq (X \wedge \phi_2)) \to \phi_2\big] \wedge && (4) \\
&\qquad \big[\forall\phi_2(\phi_2 \not\doteq (\phi_1 \wedge Y)) \vee (\phi_1 \wedge Y)\big]\Big) \leftrightarrow \neg\phi_1\Big] \\
\Leftrightarrow&\exists\phi_1 \Big[\Big(\forall\phi_2\big[(\phi_1 \doteq (X \wedge \phi_2)) \to \phi_2\big] \wedge \big[\bot \vee (\phi_1 \wedge Y)\big]\Big) \leftrightarrow \neg\phi_1\Big] && (5) \\
\Leftrightarrow&\exists\phi_1 \Big[\Big(\forall\phi_2\big[(\phi_1 \doteq (X \wedge \phi_2)) \to \phi_2\big] \wedge \phi_1 \wedge Y\Big) \leftrightarrow \neg\phi_1\Big] && (6)
\end{aligned}$$

Applied equivalences were de Morgan's laws (1), *distributivity (2), replacing* $\forall\phi_2\big[(\phi_2 \doteq (\phi_1 \wedge Y)) \to \phi_2\big]$ *by* $\forall\phi_2\big[(\phi_2 \doteq (\phi_1 \wedge Y)) \to (\phi_1 \wedge Y)\big]$ (3), *giving raise to the false subsentence* $\forall\phi_2(\phi_2 \not\doteq (\phi_1 \wedge Y))$ (4), *by* $\phi_1 \wedge Y$ *as witness* (5), *hence being eliminated* (6). *From this point on, a case distinction will reveal that A need not be self-referential, but rather has a fixed boolean value.*

A proposes existence of some ϕ_1 *and*

- *if this* $\phi_1 = Z$ *for some sentence* $Z \neq (X \wedge \ldots)$*, we get*

$$\begin{aligned}
A \Leftrightarrow&\forall\phi_2\ \big[(Z \doteq (X \wedge \phi_2)) \to \phi_2\big] \wedge (Z \wedge Y) \leftrightarrow \neg Z \\
\Leftrightarrow&\forall\phi_2\ \big[(\bot) \to \phi_2\big] \wedge (Z \wedge Y) \leftrightarrow \neg Z \\
\Leftrightarrow&\top \wedge (Z \wedge Y) \leftrightarrow \neg Z \\
\Leftrightarrow&(Z \wedge Y) \leftrightarrow \neg Z
\end{aligned}$$

Hence, A's value depends on its arbitrary nominal subsentence Y; if $v(Y) = \mathbf{1}$ then $v(A) = \mathbf{0}$, while if $v(Y) = \mathbf{0}$ we can find an appropriate witness Z, making A true.

- *On the contrary, if $\phi_1 = X \wedge Z$, for some sentence Z*

$$\begin{aligned}
A \Leftrightarrow & \forall\phi_2 \ \big[((X \wedge Z) \doteq (X \wedge \phi_2)) \rightarrow \phi_2\big] \wedge (X \wedge Z \wedge Y) \leftrightarrow \neg(X \wedge Z) \\
\Leftrightarrow & \forall\phi_2 \ \big[(Z \doteq \phi_2) \rightarrow \phi_2\big] \wedge (X \wedge Z \wedge Y) \leftrightarrow \neg(X \wedge Z) \\
\Leftrightarrow & \forall\phi_2 \ \big[(Z \doteq \phi_2) \rightarrow Z\big] \wedge (X \wedge Z \wedge Y) \leftrightarrow \neg(X \wedge Z) \\
\Leftrightarrow & \big[\forall\phi_2 \ (Z \not\doteq \phi_2) \vee Z\big] \wedge (X \wedge Z \wedge Y) \leftrightarrow \neg(X \wedge Z) \\
\Leftrightarrow & \big[\bot \vee Z\big] \wedge (X \wedge Z \wedge Y) \leftrightarrow \neg(X \wedge Z) \\
\Leftrightarrow & Z \wedge (X \wedge Z \wedge Y) \leftrightarrow \neg(X \wedge Z) \\
\Leftrightarrow & (X \wedge Z \wedge Y) \leftrightarrow \neg(X \wedge Z)
\end{aligned}$$

Hence, A's value again depends on the values for its nominal subsentences X and Y. If $v(X) = \mathbf{0}$, then $v(A) = \mathbf{0}$, while if $v(X) = \mathbf{1}$ and $v(Y) = \mathbf{0}$, we can again find an appropriate witness for Z that gives $v(A) = \mathbf{1}$.

Both cases together yield that $A \Leftrightarrow \neg Y$ and A's boolean value depends only on its nominal subsentence Y, just like sentences with one quantifier, as shown in Theorem 3.10.

While A is of course just an example of some rather short sentence, it exceeds already the grounds covered by proofs in this work. Still, it obeyed the familiar mechanics and is, again, equivalent to some nominal subsentence.

As explained more thoroughly before, this signals that self-reference should perhaps not be expected with the syntactic equality operator and gives reason for optimism towards consistency of LSO with $\doteq$.

4 Conclusion

4.1 Summarizing results

This work's purpose is to investigate LSO, the language of sentential operators, and its consistency in regard to the operator $\doteq$, the syntactic equality operator. Hence the most important introduced concepts were

- The syntax of *LSO*, defined by extending first-order logic, to FOL^+, FOL with quantification over sentences and operators on sentences.
- *Language graphs*, i.e. digraphs with sentences as vertices, were defined and FOL^+'s semantics was defined over *kernels* of this graph, corresponding to a boolean assignment of its sentences.
- The notion of *semantic equivalence*, extending FOL equivalence, was introduced.
- The operator *syntactic equality*, $\doteq$ and its semantics was introduced.

On grounds of [Wal], originally introducing LSO, concepts from his work were adopted, but also results used as the groundwork for further investigation. Namely, it was used that

- LSO, restricted to only sentential quantification but no operators and LSO with sentential operators with trivial semantics have consistent boolean valuation and further that
- *definitional extensions* can introduce new operators while keeping the language consistent and that
- any sentence has an equivalent sentence in *Prenex Disjunctive Normal Form*, *PDNF*, thus a sentence with all quantifiers in its beginning and a disjunctive matrix as its body.

Then a variety of new results has been proven within this work, some of smaller, more technical nature and some more general with wider implications. Few results were regarding LSO with arbitrary operators, while most results were directly related to the very central syntactic equality operator, amongst others:

- That [Wal]'s proposed conditions for operators that preserve LSO's consistency are not sufficient, shown with a counter-example, also with a suggestion on how they could be fixed, in section 2.2.5.
- That sentences with only one quantifier are semantically equivalent to a conjunction of some of its nominal subsentences, section 3.3.
- That restricted subsets of LSO, such as all sentences with only nominal or only sentential occurrences of sentential variables are in fact consistent, in section 3.4.1.
- How consistency of LSO could be shown with an induction on the total number of quantifiers in a sentence, section 3.4.2 and
- that a possible induction step holds for at least a number of special forms of sentences (sections 3.4.3, 3.4.4 and 3.4.5), if not all.

4.2 Reflection

The desired result, unrestricted consistency of LSO with syntactic equality, was not proven. Within this work some intuitions have been illustrated and expressed that LSO *appears* to be consistent. The underlying idea being that paradoxes come in form of self-reference of sentences that can not be resolved, as they involve unbroken odd cycles in the language graph, all proven results point towards this phenomenon not occuring.

With arbitrary operators, paradoxes can be constructed already with syntactically primitive sentences (e.g. $\forall\phi(P(\phi) \to \neg\phi)$), while this work showed that such sentences, along a variety of more complex sentences (e.g. sentences with more quantifiers and nested expressions in nominal positions instead of a bare s-variable), can be consistent, thus sowing doubt that paradoxes could lurk among the sentences that have not yet been covered.

Undoubtedly, however, this work demonstrates that the fairly intuitive notion of syntactic equality behaves in a non-trivial way, enables sophisticated expressions and connects syntax and semantics in a potentially dangerous, but hopefully safe way. Many results, in their core, show how a sentence within nominal position does influence the value of its super-sentence (e.g. $\forall\phi((\phi \doteq S) \to \phi) \Leftrightarrow S$), which is an interesting revelation in itself.

While this adds to a crucial understanding of how sentences with syntactic equality behave, the rather technical propositional equivalences, that have been proven with it, paint an optimistic picture. This may not be the entire picture, however, as it seemed to not suffice to prove consistency of LSO.

Hence it is unclear, whether another approach could be more promising, as there has been little consideration towards the language graph of LSO and possible meta-arguments that could augment the understanding of why syntactic equality behaves consistent and how odd cycles might resolve in the graph in this work. The incomplete induction that has been drawn out in the last section focuses on the number of quantifier-symbols in sentences to tame the vast number of arbitrary sentences, which is of course not the only viable approach. An induction over the nesting-depth of the operator in itself was another approach considered, but discarded.

Thus, it can be concluded from this work that there are a variety of different approaches, both continuing directly results from this work, combining results with a new idea, or focusing on a whole other way of arguing that could potentially bring the proof to a close and show consistency.

4.3 A look ahead

Considering possible continuations of this work, it is clear that the major result to add would be to show consistency of LSO with syntactic equality. As emphasized in the previous subsection, there is no lack of options of how this problem could be addressed and even a completely new angle of viewing the problem might highly benefit from some of the technical results established in this work.

Very much related and perhaps even crucial to tackle the former problem is missing understanding of what makes an arbitrary syntactic operator consistent. Operators, having of course some arbitrary arity, can be unary in their simplest form and one step towards understanding them could be to define a non-trivial and provably consistent unary operator. If this can indeed be accomplished, the natural next step would be to investigate if the same holds for multiple operators in the language as well. Ultimately, it seems very desireable to search for general conditions making an operator consistent and ideally, theses conditions being as narrow as possible, being both sufficient and necessary.

Thus one direction to go towards is to find and generalize conditions making any n-ary operator consistent, from which consistency of syntactic equality (being a binary operator) would follow. The other direction would be to prove consistency of syntactic equality and try to infer and generalize such conditions from it.

In any case, syntactic equality serves as a convenient, already defined and investigated example of an operator and any future result for operators must of course be applicable to it as well, with this work already giving some benchmark of what could be expected.

If, against all optimism expressed in this work, syntactic equality is proven to be inconsistent, possible continuations of this work lose none of their relevance, as some trivial operators are certain to be consistent and the question arises of what properties of syntactic equality make it inconsistent and what conditions could be imposed that distinguish syntactic equality from consistent operators.

References

[Bol17] Thomas Bolander. “Self-Reference”. In: *The Stanford Encyclopedia of Philosophy*. Ed. by Edward N. Zalta. Fall 2017. Metaphysics Research Lab, Stanford University, 2017.

[HL22] Volker Halbach and Graham E. Leigh. “Axiomatic Theories of Truth”. In: *The Stanford Encyclopedia of Philosophy*. Ed. by Edward N. Zalta. Spring 2022. Metaphysics Research Lab, Stanford University, 2022.

[NM44] John von Neumann and Oskar Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944.

[Wal] Michał Walicki. “Logic of Sentential Operators”. `https://www.ii.uib.no/~michal/LSO.pdf`. Submitted.