Logical Form *vs.* Logical Form: How Does the Difference Matter for Semantic Computationality?

Prakash Mondal

Indian Institute of Technology Delhi, Hauz Khas
New Delhi 110016, India
mndlprksh@yahoo.co.in

Abstract. This paper aims at pointing out a range of differences between logical form as used in logic and logical form (LF) as used in the minimalist architecture of language. The differences will be shown from different angles based on the ways in which they differ in form and represent some natural language phenomena. The implications as following on from such differences will be then linked to the issue of whether semantic realization in mind/brain is computational. It will be shown that the differences between logical form as used in logic and logical form (LF) as used in the minimalist architecture of language will help us latch on to the realization that there is no determinate way in which semantics can be computational or computationally realized.

Keywords: Logical form, logic, minimalist architecture, semantic realization.

1 Introduction

Is semantics really computationally realized? How much of meaning can be computationally realized? And how much cannot? The path toward an answer to such questions can be tremendously difficult to follow given the fact that such questions are still faintly understood or grasped given a huge dearth in understanding what meaning really is. Here in this paper such a path will be traced through tracking the differences between logical form as used in logic and logical form in the minimalist architecture of language which will be extrapolated to approach the question of how such differences can throw light on whether meaning can be computationally realized. Both logical form as used in logic and logical form in the minimalist architecture of language *represent* semantics of natural language. If they can really represent meaning in natural language, their computability can have ramifications over how and to what extent semantics is computational. The question is whether semantics or meaning in language can be computational or computationally realized in mind/brain at all on the basis of concrete facts that the differences between logical form as used in logic and logical form in the minimalist architecture of language will provide us with. An important proviso has to be added right at this juncture. The question is not to scout out and magnify differences between logical form in logic and logical form as used in generative grammar. Such differences are quite well-known. The focus is

H.D. Pfeiffer et al. (Eds.): ICCS 2013, LNAI 7735, pp. 254–265, 2013.

rather on how such differences matter for and unlock aspects of semantic computationality given that both meta-languages *represent* natural language semantics. Why do differences in formal representation actually matter given that logical form as used in logic and logical form in the minimalist architecture of language formally *represent* semantics of natural language in different ways? One can write numbers in binary or decimal and countless other ways. That does not certainly change the fact that arithmetical operations are algorithmic; the details of the algorithm just vary appropriately based on the representation used. But the case in point is here semantics, *not* mathematical facts. Semantics is different from mathematical facts both in form and nature. The latter may well lie in the Platonic realm, but the former cannot perhaps be such given the fact that the very metalanguages that encode or represent semantics are not uniform in their representational faithfulness. This is *not* true of mathematical facts or objects as one can really write numbers in binary or decimal formats without any differences in the faithfulness with which decimal format or binary format can represent numbers. This is what will be shown below in the section 2 and these differences are crucial as far as the matter about the question of whether semantics is computational or not is concerned.

1.1 Logical Form in Logic and Logical Form (LF) in Generative Grammar

Logical Form in Logic. In brief, logical form of sentences of natural language is what determines their logical properties and logical relations. Logical form of natural language sentences is always constructed relative to a theory of logical form in the language of a theory of logic (say, first order logic) [1]. To schematize what we have in mind when we talk about logical form in logic, we can have

$$T = \{T_1 \ldots T_n\},\ L = \{L_1 \ldots L_m\},\ \mathcal{L} = \{\mathcal{L}_1 \ldots \mathcal{L}_k\}$$
$$A = \{A_1 \ldots A_j\},\ B = \{B_1 \ldots B_i\} \qquad (1)$$

Here T is the set of theories of logical form; L is the set of all possible logical forms and $\mathcal{L}$ is the set of theories of logic. $L_{\mathcal{L}_k}^{T_m}$ is the set of logical forms relative to a theory of logical form T_m and in the formulas of a theory of logic $\mathcal{L}_k$. Then:

$$L_{\mathcal{L}_k}^{T_m} \subseteq L \ \ \&\ \ \Psi: L_{\mathcal{L}_k}^{T_m} \rightarrow A \times B \qquad (2)$$

Here A is the set of logical properties and B is the set of logical relations. Logical properties consist in truth values as fixed by the terms assigned to predicates and logical relations are relations between sentences which are linked by chains of different types of inferences; so entailment, implication, equivalence etc. are different kinds of logical relations which are defined with respect to a set of sentences which must not be a singleton set. The function Ψ ensures a proper mapping from a set of logical forms to ordered pairs containing logical properties and logical relations. The mapping Ψ might be a little idealized given certain approximations that might exist at the interface between what we call logical properties and what we brand as logical relations. In sum, logical forms are a way of making out what logical properties and logical relations are. The following example can perhaps make it clearer.

(1) Peter danced and Clare sang. [D(p) $\wedge$ S(c)]
(2) Peter danced. [D(p)]

Here we have two sentences with their logical forms alongside. The logical form of each determines whether each is true if the given circumstances hold true of them. This is what logical property is. And then the sentence in (1) entails the sentence in (2). Again it can be determined by looking at the logical forms of both (1) and (2). This falls under logical relations. It now becomes clear that the logical forms of the sentences in (1-2) lead us to the logical properties and logical relations in question.

Logical Form (LF) in Generative Grammar. Logical Form (LF) as used in generative grammar has a little different, if not too distant, sense embedded in it. From now onwards 'LF' will be used to denote logical form as used in generative grammar and the full form will be reserved for the identical term in logic, just to avoid any confusion. So to come to the point of discussion, Logical form (LF) in an architecture of grammar is a level of syntactic representation which is interpreted by semantics. LF represents properties of syntactic form relevant to semantic interpretations or aspects of semantic structure that are expressed syntactically [2]. As May [2] argues, it all starts with Russell's and Freges' concerns with the relation of logical form to the syntax of natural language in that the logical form representing the semantic structure is not akin to the syntactic form of natural language (in fact, it dates back to the Greek thinkers including Aristotle who bothered about this mismatch, and then it lasted well into the twentieth century pervading all thinking about language and logic). Logical form is masked by the syntactic structure of natural language. An example can be given to show this:

(3) Coffee grows in Africa.

Here one might want to say the grammatical subject is 'coffee' and the rest is the predicate. But logically, 'be in Africa' characterizes the property-so it is the logical predicate and 'the growth of coffee' is the logical subject [3]. It can be written as:

(4) + **P** (Be in Africa) ([growth of coffee])

In this sense LF has a similarity of purpose with logical form in logic. More on this will be drawn up later. So they are two different strata of representational structures. Since LF is a syntactic level of representation, the question of representations at this level and meanings assigned to structures at this level are of paramount significance. In reality such a level gains its theoretical justification through the existence of a number of independent descriptive levels each of which has its own well-formedness conditions and formal representations as maintained throughout the main streams of thought in generative grammar. Overall, LF attempts to characterize the extent to which a class of semantic interpretations that can be assigned to syntactic structures at this level are a function of their grammatical properties; but it does not mean that LF has any commitment to all possible semantic interpretations that can be assigned to syntactic structures which are derived out of their grammatical properties. LF was actually motivated on facts about quantificational NP movement which fulfills the purpose of showing the difference between surface structure and covert structure in natural languages. An epitome of such a case is the following:

(5) Every woman loves a man.

(6) i. $[_S$ a man_2 $[_S$ every $woman_1$ $[_S$ e_1 loves e_2]]]

ii. $[_S$ every $woman_1$ $[_S$ a man_2 $[_S$ e_1 loves e_2]]]

(5) can have two different LF representations based on the two different scopal interpretations shown in (6i-ii). So much for LF. Let's now look at the parallels between logical form and LF. This will give us a handle to an exploration of the ways they differ from each other in their fundamental nature and form too.

1.2 Some Parallels between Logical Form and LF

Given that we have got a rough outline of what logical form and LF are, here are some parallels between them that can be brought out. Throughout this article we will be using first-order logic for any discussion on logical form; it is not due to any bias toward it but because of its more widespread use. However, these parallels can be highlighted on three grounds: (i) They are both aimed at uncovering the semantic/logical properties masked by grammatical forms; (ii) They are both translations of natural language sentences in a kind of meta-form. (iii) They are both 'paraphrases of natural language sentences', to use Quine's words [4]. A simple example can exhibit the parallels most succinctly.

(7) Every boy likes a game.

Logical Form (Logic): (i) $\forall x \exists y$ [L(x,y)], (ii) $\exists y \forall x$ [L(x,y)]

Logical Form (LF): (i) $[_S$ a $game_2$ $[_S$ every boy_1 $[_S$ e_1 likes e_2]]]
(ii)$[_S$ every boy_1 $[_S$ a $game_2$ $[_S$ e_1 likes e_2]]]

Even if such parallels between logical form and LF might seem to be apparently evident, they mask the fundamental differences between them. Let's now turn to those differences.

1.3 Of the Differences between Logical Form and LF

The differences between logical form and LF can now be put forward. They will be traced out from a number of angles in terms of how they behave. Extrapolating Quine's [4] postulated difference between logical form and deep structure, let's say that logical form and LF are used for quite different purposes. Logical form of natural language sentences is used in logic for logical calculations and inferential implications. Whereas LF is a level of syntactic representation generated through a sequence of derivational operations used for further semantic interpretations. This leads us to a much better characterization of the differences. Here's is how. Logical form as used in logic is externally motivated, but LF is internally motivated as it is a part of internalist architecture of grammar in that LF is a part or component of an internalist architecture of language regarded as the faculty of language which is itself a part or component of mind/brain. LF is a level of syntactic operations/computations which feeds semantic interpretations at Conceptual-Intentional (C-I) interface in the minimalist architecture of the language faculty. Logical form is not anchored to any such system; so logical form cannot be characterized in that way. The differences between them can now be zoomed in on from a number of focal perspectives.

Differences in Ontology. Logical form and LF have differences in ontology too. Logical forms are constructed in the language of theory of logic which consists of two quantifiers (existential and universal). LFs in natural language represent a whole range of quantifiers apart from the two, like 'most', 'many', 'two', 'few', 'likely', 'seem' etc. etc [5]. Such differences in ontology pave the way for fundamental differences between logical form and LF come into a clearer view.

Differences in Formal Representations. Logical forms and LF have remarkable differences in formal representations which unmask the distinction in terms of their fundamental forms. The examples below show this clearly enough.

(8) Sam killed every tiger.
LF: $[_S$ every tiger_1 $[_S$ Sam killed e_1]]
Logical Form: $\forall x$ [Tiger(x) $\rightarrow$ Killed (s, x)]

(9) Most linguists sleep.
LF: $[_S$ most linguists_1 $[_S$ e_1 sleep]]
Logical Form: (most x: x is a linguist) [sleep(x)]

(10) Few philosophers like cats.
LF: $[_S$ few philosophers_1 $[_S$ e_1 sleep]]
Logical Form: (few x: x is a philosopher) $\exists y$ [Cat(y) $\wedge$ Like(x, y)]

But the point to be noted is that even if the formulas in (9-10) use restricted quantification, it is done through an extension of natural language quantifiers like 'few', 'most' into logic. Hence it cannot be said that logical forms do *not* lack all quantifiers found in natural languages.

Differences in Restriction. It is quite well known that that in logical forms quantifiers range over a universe of individuals, as in (11) below.

(11) Every linguist drinks.
$\forall x$ [linguist(x) → drinks(x)]

But in LF the range is restricted by the head noun, as in 'few good girls' by 'good girls'.

Differences as Seen from the Phenomenon of Crossed Binding. The phenomenon of crossed binding is interesting because it opens a window onto the crucial differences between logical form and LF. Crossed binding is a problem for LF as has been seen in Bach-Peters sentence. Let's see how in the example in (12) taken from May[2].

(12) Every pilot who shot at it hit some MIG that chased him.

(13) i. [[Every pilot who shot at it]$_1$ [[some MIG that chased him]$_2$ [e_1 hit e_2]]]
ii. [[some MIG that chased him]$_2$ [[Every pilot who shot at it]$_1$ [e_1 hit e_2]]]

This sentence in (12) can have two LF representations in (13).As can be seen above, in (13i) the pronoun 'him' is bound by the hierarchically higher antecedent 'every pilot…'; but in (13ii) only the pronoun 'it' is bound by the antecedent 'some MIG…'. Both these two bindings are not represented in any single LF representation. To alleviate this situation, May has proposed 'absorption':

$$\ldots [NP_i \ \ldots \ [\ NP_j \ \ldots \rightarrow \ldots [NP_i \ \ NP_j\ ]_{i\ ,j} \ldots \qquad (3)$$

Such a representation turns (n-tuples of) unary quantifiers into binary (n-ary) quantifiers. Crossed binding is not a problem for logical forms. Need for absorption does not arise either. Let's see how:

(14) i. $\forall x \exists y$ [[pilot(x) ∧ [MIG(y) ∧ Shot at(x, y)]] → [Hit(x, y) ∧ Chased (y, x)]]
ii. $\exists y \forall x$ [[pilot(x) ∧ [MIG(y) ∧ Shot at(x, y)]] → [Hit(x, y) ∧ Chased (y, x)]]

Interesting to note is the fact that the LF representations in (13) can be mapped in a partial manner to the ones in (14). Thus (13i) can be mapped to (14i) and (13ii) to (14ii), but crossed binding is reflected in the either of logical forms in (14i-ii), but not in any of (13i-ii). The LF representation with the mechanism of 'absorption' applied can be mapped to both in (14i-ii). Hence again, it will be a case of partial homology if we try to map LF structures to logical forms. Meaning representation is blocked due to a bottleneck in the mapping process itself. This will have its repercussions across other cases to be discussed below.

Differences as Seen from the Phenomenon of Crossover. Other differences between logical form and LF can be telescoped through the phenomenon of crossover. Let' see look at the sentences below.

(15) *His_i cat loves every boy_i.
(16) *Her_i friend loves some $spinster_i$.

The indexes indicate co-reference between the NPs. As has been argued and shown throughout the generative literature, this is due to the covert movement of the QNPs (quantificational noun phrases) 'every boy' and 'some spinster'. So the LF representations will look like:

(17) [$_S$ every boy_1 [$_S$ his_i cat loves e_1]]]
(18) [$_S$ some $spinster_1$ [$_S$ her_i friend loves e_1]]]

Logical forms do not reflect such problems so much so that we can have perfectly fine logical forms for (15) and (16), contrary to facts in natural language as shown below:

(19) $\forall x\ [Boy(x) \rightarrow \exists y[x\text{'s } cat(y) \wedge Loves(y, x)]]$
(20) $\exists x\ [Spinster(x) \wedge \exists y[x\text{'s } friend(y) \wedge Loves(y, x)]]$

Again this reveals the fact that logical forms can sometimes overgenerate or overrepresent natural language sentences, LF do not. One could, of course, argue that some further syntactic rules may be added to formal logic to capture some constraints that will bar the constructions in (15-16); but this begs the question as the lack of existence of these to-be-postulated syntactic constraints or rules is the reason why we find (19) and (20) to be problematic as far as logical form is concerned. Again this reveals the fact that logical forms can sometimes overgenerate or overrepresent natural language sentences, LF do not. Of course, in the case of (19), one may argue that it is a representation for the sentence "Every boy is loved by his cat". But a fact that is basic and obvious but not of trivial significance can be driven home from this. It is that logical form does *not* distinguish between the two. That is where the problem lies.

Differences as Seen from the Phenomenon of Binding. Further evidence can be accumulated regarding the nature of differences between logical form and LF. This can come from further facts about binding. The examples taken from Miyagawa [6] below exhibit this clearly.

(21) Some students from his_i class appear to every $professor_i$ to be idiots.
(22) $Jack_i$'s mother seems to him_i to be wonderful.

(23) ∃x [Student(x) ∧ ∀y [Professor(y) → ∃z [y's class(z) ∧ From (x, z) ∧ Appear-to-be-idiot(x, y)]]]
(24) ∃x [j's mother(x) ∧ Seem-to-be-wonderful(x, j)]

The logical forms of (21-22) are (23-24). (23) does not represent the fact that the surface form and LF do not coincide in (21) in that the QNP 'some students…' has moved from the position below 'every professor'; and (24) does not reflect the fact that in (22) the surface form and LF correspond with each other as had it not been the case the sentence would have created a violation of binding principle C when the referring expression 'Jack' if lowered is c-commanded by the pronoun 'him'. This has significant implications for the differences between LF and logical form. LFs are thus sequence-dependent and sensitive to levels of representations in an architecture of grammar; logical forms are not sequence dependent in this way and are self-contained.

2 What Does It All Reveal?

It is now the time to wrap up the differences between logical form and LF into a space of important generalizations and implications on the differences as shown above. Let's now flesh them out. LF is a stage in a derivational sequence of computations. Let's call it $<D_1 \ldots D_n>$ where each D_i is computed from the output generated by D_{i-1}. Let's assume that D_n is the stage where LF is computed. Since $<D_1 \ldots D_n>$ is driven by computational considerations of locality, economy and other syntactic constraints (global or local), LF is also sensitive to such constraints. LFs are constructed on the basis of the computational operations as required by the derivations. But logical forms are constructed without any reference to any prior or posterior stages in a sequence of operations. Hence the differences in representational forms too! Logical forms cannot be specified this way. Hence the problems above that LF faces do not get reflected in logical forms. LFs are also sensitive to the requirement of generating licit sentences, while logical forms are not. Moreover *logical conservatism* which goes in for an economy in extensions in a logical theory and *ontological conservatism* which favors fewer ontological commitments constrain the form of logical forms [1]; LF, on the other hand, is constrained by *computational parsimony* which favors fewer computations in Merge operations. *Inclusive Condition* which bans entities not present in the Numeration (selected items from the lexicon) as defined in the minimalist architecture of grammar can at best carry the tenets of ontological conservatism but it is more global if we want to draw some parallels between the constraints governing logical form and those governing LF.

2.1 Fodor's Isomorphy, Logical Form and LF

To see how the differences between logical form and LF play out at the level of semantic representations in mind/brain, it is necessary to look into the roles they each play in Fodor's [7] postulated supervenience of logical forms of propositional attitudes on the syntactic properties of mental representations. This supervenience is also a sort of isomorphy as whatever the nature of logical form of a propositional attitude like belief is, the corresponding mental representation will have the same syntactic property. Such an isomorphy runs into fiendish problems in assignment of meanings. Let's see how.

(25) a. M1~ John walks. $\longrightarrow$ $F(j)$
b. M2 ~ Max walks. $\longrightarrow$ $F(m)$
(26) a. M1~ Crystal is bright. $\longrightarrow$ $G(c)/ \exists x\ [C(x) \wedge G(x)]$
b. M2 ~ John is bright. $\longrightarrow$ $G(j)$
(27) a. M1~ Crystal is bright. $\longrightarrow$ $G(c)$ | $\exists x\ [C(x) \wedge G(x)]$
b. M2~ Summer is bright. $\longrightarrow$ $G(s)$ |

Here *M* refers to mental representation. The logical forms of the sentences are placed alongside the sentences as indicated by arrows. In (25), the logical forms are different based on a difference in terms in that in (25a) the logical form contains 'John' and in (25b) it is 'Max' the predicate is about. So the corresponding mental representations M1 and M2 will also have different syntactic properties aligned with the respective logical forms. What happens in (26) is pretty interesting. In (26a), the sentence will have two possible logical forms based on whether we interpret 'Crystal' as a common noun or a proper name. But in (26b) this problem does not arise. What is of significance is that the indeterminacy present in (26a) cannot be resolved from within the sentence in question; it needs context which is not a syntactic property. Overall, on one hand, logical form does not supervene on the syntactic property of the mental presentation as we see in (26a); on the other logical form supervenes on the syntactic property of the mental representation as in (26b). The case in (27) leads to inconsistency in that both the sentences, on one hand, have two different logical forms and on the other possess the same logical form too. The inconsistency is again due to the unavailability of context which is not a syntactic property.

This suggests that the postulated isomorphy between logical forms of propositional attitudes and the syntactic properties of mental presentations is misleading and based on a shaky ground. Interestingly LFs may not run into this problem as it is anchored to C-I (Conceptual-Intentional) system. In generative grammar semantic structures- whatever their form is- are determined by syntactic computations. Hence the relation between syntax and semantics is much more restricted and constrained than is supposed to be. Much of semantics has been pushed into the mapping between C-I (Conceptual-Intentional) interface and the domain of concepts, meanings. C-I system might resolve the indeterminacy when the pairs are interpreted at C-I system after being shipped to it. If this is the case and the fact that only logical form-syntax isomorphy runs into problems as shown above but LF does not, then it follows that logical form and LF are different in kind and phenomenology, a fortiori.

3 What Does It Mean for Semantics to Be Computational?

The question of what it means for semantics to be computational needs to be keyed onto how computationality is involved in semantics. This needs a little more elaboration given a faint understanding and absence of a full grasp of what meaning is. The same can be said about the notion of computation. What is it that is meant when a question on whether semantics is computational or not is asked? Computation is one of the most confounded and unclear notions employed in cognitive science [8], [9]. So when a question on whether semantics is computational or not is asked, much hinges on the fact that the right concept of computation is applied to the phenomenon that is under the scan of the evaluation criteria of computation [8]. So here the notion of semantic computationality will be used in the classical sense of computation where inputs are mapped to outputs according to some well-defined rules by means of symbolic manipulation of digital vehicles in the form of linguistic strings. This notion of computation is the narrowest in the hierarchy of notions of digital computation [8]. The reason behind the employment of this notion of computation is that this is the very notion of computation that has been keyed to much of generative linguistics. The question of whether semantics is computational in the analog sense of computation or in the *generic sense* [8] that encompasses both digital and analog computation will not be touched upon here in that the differences between logical form as used in logic and logical form in the minimalist architecture of language are targeted as the pedestal on which the issue of semantics being computational will be teased apart. And these are the two metalanguages that represent apparently intangible semantics, which is what has been capitalized on for the sake of an investigation into whether semantics is computationally realized or not. But of course, the question of whether semantics is computational in the analog sense of computation or in the *generic sense* can be sharpened to a larger degree from the following discussion as we will see below.

4 Semantic Computationality, Logical Form and LF

Now let’s turn to the issue of semantic computationality as we gather the implications derived from what we have shown above so far. The discussion above indicates that LFs are semantically more accessible and transparent, while logical forms are not, given the problems pointed out above. Apart from that, the correspondence between logical forms and syntactic properties might be a case of partial homology, but not a full correlation. If this is so, logical forms weaken the case for semantic computationality as they are in a patchy correspondence with syntactic properties which are actually computational properties. LFs do strengthen the case for semantic computationality as LF is anchored to the C-I system thereby being more semantically accessible and LF representations for cases in (25-27) will be identical in parts which contain the subjects that will be treated uniformly as the same in being all amenable to interpretation only at the later stage at C-I system. In addition, the fact that at LF aspects of semantic structure supervene on syntactic properties, qua Fodor [10] further regiments aspects of meaning being computational by dint of being represented at LF. But if we co-opt Fodor’s analysis and view of what is computational, we may run into severe problems as we have already encountered problems of other kinds derived from his isomorphy of logical form with syntactic properties. It is because he has allowed for the possibility that C-I system intentions, beliefs and other inferential (global) processes supervene on syntactic properties (internal or internal), qua Fodor [7]. So by that means, such processes are also computational. Therefore everything at and beyond LF becomes computational and it is computational all around! Such a conclusion seems to be unwarranted and uncalled for given that it forces the case for globality in semantic computationality all the way throughout the entire gamut of cognition. It also leads to the absurd conclusion that semantic properties or aspects of semantics are actually syntactic! Nothing then prevents intentions, beliefs and other inferential (global) processes from flowing into (narrow) syntax. In fact Fodor's notion of computation is based on his classical notion of computation; and hence if it is Turing or Von Neumann style of computation, then semantics at LF are computational, but those from the C-I system are not. But as Langendoen and Postal [11] show in their NL (Natural Language) Non-Constructivity Theorem, this is also flawed under a closer analysis in that there is no Turing machine-style-constructive-procedure for generating either syntactic rules or semantic rules at LF if NL (Natural Language) Non-Constructivity Theorem is to be believed to be true.

But there are a range of views of computation-causal, functional and semantic [12]; and in addition, there are a lot of problems inherent in the notion of computation itself that blocks the path that differentiates computing systems from non-computing systems [13]. This borders on Searle's [14] conclusion that any physical system computes an algorithm leading to an emptiness in the notion of computation. Semantic computationality becomes a non-issue over which we are all perhaps cudgeling our brains as everything computed by the brain is computational. The matter becomes more complicated as we try to home in on the issue of semantic realization being computational. LF, on one hand, trivializes the notion of semantic computationality by overextending, *over*specifying its domain; logical form, on the other, *under*specifies it. LF cannot act as a standard against which we can assess semantic computationality chiefly because there are other parallel logical systems including logical form which do a similar job of representing meaning and these parallel logical systems project a different picture of semantic computationality. We thus fall into the trap of a relativistic notion of semantic computationality. Worse than that is the fact that LF and logical form are not *inter*convertible and *inter*translatable without any change in meaning as shown in the sections above as they cannot be mapped onto one another without much readjustment in meanings. Therefore, there cannot even appear any sense in which we can assess or examine semantic computationality as this issue cannot be checked against any standard, nor can it be put on the pedestal of test as both the notational/representational languages (LF and logical form) project varying pictures of semantic computationality even though Steedman and Stone [15] have defended a realist interpretation of semantics within which semantics can be conceived of in computational terms by keeping semantics from aspects of processing. For semantics to be representable, we need some representational (meta)language which can be checked for how much space it allows for so that semantics can be seen to be computationally realizable. But the discussion above shows that no such representational (meta)language is consistent and uniform with respect to the way meaning can be shown to be computationally realized. Hence, this fact extends to conceptual graphs as well which also represent meaning. The problem of logical form equivalence in cases of semantic distinctions [16] makes it more unlikely that logical representational (meta)languages can actually represent meaning fully, let alone reveal the extent of semantic computationality. There is then no determinate way to determine whether semantics is computationally realized in mind/brain.

5 Conclusion

To conclude, semantic computationality is very much an issue to be determined through a thorough consideration of the representational devices that represent semantics. But these very devices or systems of representation do not provide a constant testing ground on which semantic computationality can be scrutinized. Rather what we find is that either there is an exaggeration of semantic computationality or there cannot be any case for semantic computationality at all. We lose out in both ways. Further thinking and research can clarify which path is the better one.

References

1. Menzel, C.: Logical Form. In: Craig, E. (ed.) Routledge Encyclopedia of Philosophy, vol. 5. Routledge, London (1998)
2. May, R.: Logical Form: Its Structure and Derivation. MIT Press, Cambridge (1985)
3. Seuren, P.A.: Language in Cognition, vol. 1. Oxford University Press, New York (2009)
4. Qunie, W.V.O.: Methodological Reflections on Current Linguistic Theory. Synthese 21(3-4), 386–398 (1970)
5. Harman, G.: Deep Structure as Logical Form. Synthese 21, 275–297 (1970)
6. Miyagawa, S.: Why Agree? Why Move? MIT Press, Cambridge (2010)
7. Fodor, J.: The Mind does not Work that Way. MIT Press, Cambridge (2000)
8. Piccinini, G., Scarantino, A.: Information Processing, Computation and Cognition. Journal of Biological Physics 37, 1–38 (2011)
9. Fresco, N.: Concrete Digital Computation: What does it Take for a Physical System to Compute? Journal of Logic, Language and Information 20, 513–537 (2011)
10. Fodor, J.: LOT2. Oxford University Press, New York (2008)
11. Langendoen, D.T., Postal, P.: The Vastness of Natural Languages. Basil Blackwell, Oxford (1984)
12. Fresco, N.: Explaining Computation without Semantics: Keeping it Simple. Minds and Machines 20, 165–181 (2010)
13. Shagrir, O.: Why we View the Brain as a Computer. Synthese 153, 393–416 (2006)
14. Searle, J.: The Rediscovery of the Mind. MIT Press, Cambridge (1992)
15. Steedman, M., Stone, M.: Is Semantics Computational? Theoretical Linguistics 32(1), 73–89 (2006)
16. Shieber, S.M.: The Problem of Logical Form Equivalence. Computational Linguistics 19, 179–190 (1993)