J Log Lang Inf
DOI 10.1007/s10849-017-9261-3

Yoad Winter's *Elements of Formal Semantics*, 2016, Edinburgh Advanced Textbooks in Linguistics (Edinburgh University Press)

Paperback, pp. 258. ISBN 978 0 7486 4043 0

Edward L. Keenan[1]

© Springer Science+Business Media B.V. 2017

Abstract *Elements of Formal Semantics* (EFS) has already been reviewed twice (Rett in Glossa 1(1):42, 2016; Erlewine in Comput Linguist 42(4):837–839, 2017). As well, the website for the work is accompanied by evaluative quotes by noted scholars. All are very positive concerning its clarity and its utility as an introduction to formal semantics for natural language. As I agree with these evaluations my interest in reiterating them in slightly different words is limited. So my reviews of the content chapters will be accompanied by a Reflections section consisting of my own reflections on the foundations of model theoretic semantics for natural language as laid out in EFS. The issues I address—alternate ways of accomplishing the tasks Winter treats—should not be included in an introductory work but they may be helpful for those who teach classes for which EFS is an appropriate text. They might also help with queries about the content of the text by those using it. I note that a mark of a clear text is that it allows the reader to reflect on its content not its presentation.

Keywords Formal semantics · Yoad Winter · Review · Entailment · Compositionality

Elements of Formal Semantics (EFS) has already been reviewed twice (Rett 2016; Erlewine 2017). As well, the website for the work is accompanied by evaluative quotes by noted scholars. All are very positive concerning its clarity and its utility as an introduction to formal semantics for natural language. As I agree with these evaluations my interest in reiterating them in slightly different words is limited. So my reviews of the content chapters will be accompanied by a *Reflections* section consisting of myown

Full disclosure: I myself have written extensively on generalized quantifiers and with Larry Moss (2016) published a formal semantics text (less introductory than Winter's).

✉ Edward L. Keenan
edward.keenan1@gmail.com

[1] Department of Linguistics, UCLA, Los Angeles, CA, USA

Published online: 16 November 2017

reflections on the foundations of model theoretic semantics for natural language as laid out in EFS. The issues I address—alternate ways of accomplishing the tasks Winter treats—should <u>not</u> be included in an introductory work but they may be helpful for those who teach classes for which EFS is an appropriate text. They might also help with queries about the content of the text by those using it. I note that a mark of a clear text is that it allows the reader to reflect on its content not its presentation.

EFS consists of seven chapters, of which 2 through 6 express the primary content. Chapter 1 discusses the use of the book by instructors, the intended readership and background that readers will find useful, the role of the exercises, and some of the formal notation used in the book.

But one point there, which is easy to skip: Winter links formal semantics and cognitive neuroscience (Dehaene 2014) on the one hand and artificial intelligence (Liang and Potts 2015) on the other. Quoting from Dehaene: "only mathematical theory can explain how the mental reduces to the neural…". EFS does not further pursue these related fields, but does note, rightly to my mind, that mathematical formulation can allow us to relate results in all three areas. Recall here Galileo's dictum that mathematics is the language of science, and the much later but oft cited article by the physicist Wigner 1960 "The unreasonable effectiveness of mathematics in the natural sciences".

Chapter 7 ends the book, recapitulating some points but its main, and not inconsiderable value, is in drawing the readers' attention to 11 areas of current semantic research which go well beyond EFS. Each of these areas—anaphora, plurality, tense and aspect, implicatures and presuppositions,…is given a paragraph or two with several up to date references. This enhances nicely the didactic value of EFS, enabling the reader to pursue on her own, work in many areas that build on the background EFS offers.

Chapter 2 *Meaning and Form*, introduces the basic concepts of model theoretic semantics: (1) **entailment**, (2) **compositionality** and (3) **model** and **denotation in a model**. It is, in my judgment, the fundamental chapter in the book and a driving force behind the positive comments in the reviews.

Winter introduces entailment as based on everyday drawing of inferences from assertions people make. E.g. from an assertion of (1a) we infer (1b), meaning that (1b) is true in any situation in which (1a) is, noted (1a) $\Rightarrow$ (1b) and read "(1a) entails (1b)".

(1) a. *Tina is tall and thin* b. *Tina is thin*

Replacing *and* in (1a) by *or* the result, (1a)$'$, *Tina is tall or thin*, does not entail (1b), as (1a)$'$ is true if Tina is tall but not thin. An obvious metalinguistic inference from this pattern is that *and* and *or* do not mean the same thing. The point of this unsurprising claim is that we have shown it using *entailment* as a basic semantic relation. And the fact that (1a) and (1a)$'$do not mean the same thing is established by showing that they have different entailments.

We should note that even in this initial analysis of very simple Sentences (Ss) not all the entailment paradigms Winter adduces are as simplistic in appearance as (1). For example, in Winter's (2.3), sentence (2a) entails (2b): (2a) $\Rightarrow$ (2b).

(2) a. Tina is tall and Mrs. Turner is not tall
b. Tina is not Mrs. Turner

Capturing entailments is Winter's primary adequacy criterion of a semantic analysis of a natural language. Formally, he defines what it means for a sentence to be interpreted as true in a model and then (standardly) defines formal entailment, which he notes $\leq$, by: for all sentences S_1 and S_2, $S_1 \leq S_2$ iff for all models M, if S_1 is interpreted as true in M then so is S_2. Then the adequacy criterion (the TCC p.21) is: Whenever some $S_1 \Rightarrow S_2$—that is, S_2 is judged true in all situations in which S_1 is judged true—then $S_1 \leq S_2$.

One (small) criticism: EFS amply illustrates by example that whether some S entails some S' depends on what the Ss *mean*. But I think it would not hurt to overtly state that what a sentence entails depends on its meaning, so two Ss have different meanings if one entails something the other does not. This amounts to showing that they are not mutually entailing.

Winter defines *true in a model* standardly: a model is a pair (E, {0,1}), E a non-empty set of *entities* and {0,1} the set of truth values (0 = false, 1 = true), together with a function $[\![\cdot]\!]$ which maps expressions of the language to objects (their denotations in that model) defined in terms of (E, {0,1}). Denotations of lexical items are stipulated and those of syntactically complex expressions are given recursively as a function of the denotations of their immediate constituents. E.g. proper nouns like *tina* are mapped to elements of E, that is, $[\![\text{tina}]\!] \in E$. Predicate adjectives like *tall* are mapped to subsets of E, *is* denotes the set membership relation, and the S *Tina is tall* is mapped to 1 if $[\![\text{tina}]\!] \in [\![\text{tall}]\!]$ and to 0 if $[\![\text{tina}]\!] \notin [\![\text{tall}]\!]$. In models as developed so far *and* just combines with predicate adjectives, e.g. *tall and thin* and denotes set intersection.

Finally the semantic interpreting function $[\![\cdot]\!]$ is required to be compositional. We cannot accept that (3a) below would have the meaning expressed by (3b).

(3) a. Tina is tall and thin   b. Most rabbits like cabbage

(3a) says nothing about rabbits or cabbage, but it does mention *Tina*, *tall* and *thin* as well as the constants *is* and *and*. Why would we use the words we do in (3a) if the meaning we want to express is that of (3b)? Generalizing, how could we understand complex novel utterances if their interpretation was unrelated to that of the expressions they were built from? Thus EFS, like other formal approaches to natural language semantics, requires the semantic interpreting function to satisfy Compositionality: The interpretation of a syntactically complex expression is a function of that of its immediate constituents and how they are combined.

The Compositionality EFS espouses is *Direct Compositionality*—natural language expressions directly denote in models, rather than being translated into some other formalism (Montague's Intensional Logic in PTQ (1973) or the "LFs" of Heim and Kratzer 1998) which is then interpreted in a model.

EFS does note one issue with Direct Compositionality (addressed, in effect, in Chapter 5). Namely, we find structurally ambiguous English Ss which are pronounced word for word the same but which have different meanings. EFS illustrates with the slightly cumbersome example (4a):

(4) a. Tina is not tall and thin
b. Tina is not both tall and thin
c. Tina is not tall but is thin

(4a) is claimed to be semantically ambiguous as between the meanings more naturally expressed in (4b) and (4c). Structurally (4a) is ambiguous according as *and* has combined with the complex adjective *tall and thin* and then *not* combines with that, or according as *not* just negates *tall* and then *not tall* and *thin* are combined with *and*. So technically (the point is largely ignored later) EFS claims that the domain of the interpreting function $[\![\cdot]\!]$ is not simply phonological or gestural strings but rather entire derivational structures (labeled trees).

At one point here EFS is less explicit than it might be: Compositionality would have been more convincingly illustrated if the fragment of English considered had been given an explicit grammar. Since there is so little in it at this point we could have listed the lexical items by category and given the (context free) rules in simple "if-then" form: if s is of category C and t of category C$'$ then s concatenated with t is of category C$''$. Then we see that there are two ways to construct *Tina is not tall and thin*, each with its own compositional interpretation.

As a final, very positive, remark: this chapter and the other content chapters, are accompanied by many exercises that extend (often considerably) the text in a natural way permitting self instruction. For example in this chapter additional boolean connectives such as *or…, neither…nor…* and *both…and…* are introduced on the pattern of *and* given in the body of the text.

Reflections on Chapter 2

1. An early explicit goal of Generative Grammar was to account for how humans produce and understand novel utterances. We cannot learn them case by case as we do with lexical items, as there unboundedly many. Positing a mental grammar accounts for the production part of this goal, and Compositionality the comprehension part. Compositionality is not explicitly related to this historical goal by EFS, but it does satisfy it.

2. Chapter 2 is a marvelously streamlined introduction to the primary concepts used in a model theoretic interpretation. It is tempting to want to enrich it and refine it. I took one such step above suggesting that we add a grammar for the English fragment used to illustrate Compositionality. Another, worth a footnote, concerns Winter's adequacy criterion (the TCC) in which he just says that <u>if</u> S_1 entails S_2 <u>then</u> in all models M, $[\![S_1]\!] \leq [\![S_2]\!]$. But this "if-then" must be an "if and only if", otherwise for example a function $[\![\cdot]\!]$ which interpreted all Ss as true would satisfy it but fail to characterize our pretheoretical judgments of entailment. But if we start making the treatment more precise at this point, introducing jargon like *iff* and so on, we will lose EFS's easy appeal to the newcomer. Rather it is better, much better in my judgment (see also Dennett 2017) to learn to manipulate the model theoretic apparatus before trying to understand how to improve it. E.g. learn first how *Tina is tall and thin* formally entails *Tina is thin*, whereas *Tina is either tall or thin* does not. Then after converting inference patterns to habits it becomes more effective to reflect on how we might improve a given proposal for a semantic analysis.

3. In the terminology of Keenan (1982) the notion of interpretation in a model used in this chapter is "ontologically perfect". That is, the primitives of the semantics are all denotable by expressions: truth values are denoted by Ss and entities (in E) by proper names. It would somehow seem unnatural if the denotations of complex expressions depended on objects we had no way of referring to in our language. Yet by Chapter 6 the revised model theory proposed by Winter (and many others) loses ontological perfection.

4. A newcomer to semantic analysis might wonder why entailment is the defining criterion of an adequate analysis. There are other semantic properties and relations we want to study, *presupposition* for example (Erlewine 2017). But I agree with Winter that entailment is the best place to start. My experience in using entailment in various languages reveals judgments to be pretty "stable" as Winter says. We may be unclear about just how the world must be for *John loves Mary* to be true. But we find it rather easy to agree that <u>if</u> the world is such that *It is Mary who John loves* is true <u>then</u> it is such that *John loves Mary* is true. That is, the first entails the second. And EFS in no wise rules out studying other semantic relations. Plausibly for example (5a,b) are true in the same conditions and so mutually entailing, but only (5a) presupposes (5c).

(5) a. The king of France is bald
b. France has a king and he is bald
c. France has a king

But I often find it hard to infer presuppositions of Ss not in subject-predicate form. Classically (6a) presupposes (6b). I feel this carries over to the conditional (7a) but not to the conditional (7b).

(6) a. The king of France is doomed
b. France has a king
(7) a. If countries raise tarifs then the king of France is doomed
b. If France and Spain both have kings then the king of France is doomed

Chapter 3 *Meaning and Form*. This chapter provides a systematic way of classifying expressions according to the model-theoretically defined sets in which they denote. EFS enriches the class of Ss considered to include ones built from intransitive verbs, *Tina smiled,* and transitive verbs, *Tina praised Mary*. For any model M, *smiled* denotes a function from E^M, the set of entities of M, into {0,1}, and the truth value denoted by *Tina smiled* is whatever value the **smile** function assigns to the **tina** entity (tense ignored). Similarly *praised* denotes a function from E into the set of functions from E into {0,1}. The truth value of *Tina praised Mary* in a model M is the truth value **praise**(**mary**) assigns to **tina**.

This approach seems at variance with Ch 2 where the truth of *Tina is tall* was given set theoretically. But Winter notes (standardly) that there is a natural correspondence between the subsets of E and the functions from E into {0,1}. A subset A of E is *identified* by that function from E into {0,1} which maps to True (1) just the elements of A, all other elements of E being mapped to 0. This function is called the *characteristic* function of A. Interpreting intransitive verbs as such functions allows us to refer to the set they characterize. Writing **smile*** for the set of objects that the **smile** function maps to 1 we say indifferently that *Tina smiled* is true in M iff **tina** ∈ **smiled*** or **smiled**(**tina**) = 1.

Thus EFS shifts between equivalent set theoretic and function-argument statements. To systematically characterize the sets in which complex expressions denote, the function-argument approach is usual. EFS first inductively defines a set of *types*: the least set that contains e and t and contains (a, b), usually noted just ab, whenever it contains both a and b. Expressions in the language are indexed by the types, which determine the sets in which objects of those types denote. For d a type and M a model, we write $\mathrm{D_d}^{\mathrm{M}}$ for the set in which expressions of type d denote in M. (The superscript M is often omitted). It is required that for each M, $\mathrm{D_e} = \mathrm{E}$, $\mathrm{D_t} = \{0, 1\}$ and in general $\mathrm{D_{ab}}$ is the set of functions from $\mathrm{D_a}$ into $\mathrm{D_b}$.

Syntactically expressions u of type ab concatenate (in any order) with expressions v of type a to yield uv (or vu) of type b. Semantically $[\![\mathrm{uv}]\!]^{\mathrm{M}}$, the interpretation of uv in the model M, is $[\![\mathrm{u}]\!]^{\mathrm{M}}([\![\mathrm{v}]\!]^{\mathrm{M}})$, as is $[\![\mathrm{vu}]\!]^{\mathrm{M}}$. So for example $[\![\mathrm{Tina_e\ smiled_{et}}]\!]^{\mathrm{M}} = [\![\mathrm{smiled_{et}}]\!]^{\mathrm{M}}\ ([\![\mathrm{Tina_e}]\!]^{\mathrm{M}})$. When no confusion results we just write **smile** for $[\![\mathrm{smiled_{et}}]\!]^{\mathrm{M}}$, etc.

The typing notation serves a dual purpose. Syntactically an expression of type ab combines with one of type a to yield one of type b and semantically it denotes a function from $\mathrm{D_a}$ to $\mathrm{D_b}$. This often enables us to infer a type for an expression given the types for expressions it has combined with. For example (EFS p.61) treating *Tina is tall* as of type t and *Tina* as of type e we infer that if *is tall* has a type—there are other options—it is (e,t). And given that one of *is* and *tall* must have type (x,et) and the other type x so by function application *is tall* has type (e,t). Winter assigns *tall* type (e,t) and *is* type (et,et), the desired result.

But there are some surprises. In Ch 2 *is* denoted the set membership relation. Now EFS stipulates it to denote the identity map of type (et,et). So it maps each subset A of E to itself (equivalently, each function F from E into {0,1} to itself).

There are objections (later) concerning EFS's analysis of *is tall*. But the main point here is that the type theoretic formalism gives us as way to reason out possible categories for expressions given those in the syntactic context in which they occur. Mainstream generative syntax tends to assume a fixed set of categories rather than a mechanism for creating categories as we go along.

We illustrate this freedom with *a rosy cheeked girl* (*flat footed policeman*, *broad shouldered athlete*,…) not discussed in EFS. Assign *girl* type (e,t) and *rosy cheeked* type (et,et) as it modifies *girl,* and *rosy* itself type (et,et) as it modifies *cheek*, of type (e,t). But *rosy* is not syntactically optional: **a cheeked girl*, **a footed policeman*, etc. We may capture this by assigning to *-ed* a type that looks for an (e,t) to make something that requires an (et,et) to make an (et,et). Thus *-ed* has type: (e,t),((et,et),(et,et)). Then *rosy cheeked* modifies *girl* but we do not generate *cheeked girl*. And to reiterate: This interlude is intended solely to illustrate the freedom we have in a type theoretic system of categories.

Chapter 3 now (surprisingly) introduces a widely used tool of semantic analysis, **lambda abstraction**, a systematic way of building expressions which denote functions. Specifically if x is a variable of type τ and φ an expression of type σ then $\lambda x.\varphi$ is an expression of type $\tau\,\sigma$. (EFS so far eschews the term *variable*). Semantically, in each model M it maps any object **a** of type τ to the interpretation of φ when x is set to denote **a**. As EFS notes in Chapter 5, lambda abstraction is a kind of "inverse" of function application. The latter applies functions of some type $\tau\,\sigma$ to objects of type τ to yield objects of type σ. So in semantic representations it eliminates functional expressions in favor of their values. Lambda abstraction starts with an expression of some type σ and builds a function of type $\tau\,\sigma$, so it introduces functional expressions.

Lambda abstraction is enormously useful in studying binding and scope ambiguities (chapter 5) as well as the *de re / de dicto* distinction (chapter 6). No text on current semantic theory can lack such discussion. But the reasons for introducing lambda abstraction in chapter 3 are minimal. The immediate one seems to be to present a way of saying that **is** is the identity function of type (*et*,*et*), e.g. it maps **tall** to **tall**, so **is tall** = **tall**. But *voila*, we just said what *is* denotes, we don't need a more complicated notation like $\lambda g_{et}.g$. EFS also motivates lambda abstraction with reflexives like *herself* in *Tina praised herself.* But this occurs after EFS shows how we might assign a denotation of type (eet,et) to *herself*. It would map **praise** for example to that function of type *et* which maps each **a** of type *e* to (**praise**(**a**))(**a**). To be sure, pers EFS, we can also define this function elegantly using the lambda operator: *himself/herself* = $\lambda R_{eet}.\lambda x_e.Rxx$. This will map **praise** to λx_e.**praise** xx, which maps **tina** to (**praise tina**)**tina**. Ok, but awkward, as we have some tendency to think that *Tina praised Tina* is used when the speaker is thinking of two different people named *Tina*. Examples more appealing to the beginner are ones like *Every student praised himself vs Every student praised every student*. Here both Ss are unproblematic in both form and content, and they clearly differ in meaning, so binding is not semantically merely the replacement of an NP by a reflexive in certain contexts.

There are several other difficulties associated with the use of lambda abstraction and variable binding operators (VBOs) in general. Winter notes (p.68) that for readability he skips over conditions on substitution when a $\lambda x.\varphi$ combines with an appropriate argument α—only "free" *x* get replaced, α can't contain free variables that become bound by other variable binding operators in φ, etc. These simplifications are fully justified in my experience. Beginning students can be led to correct usage of the lambda operator just by following good examples. Later they can learn the slightly tricky traps to avoid.

A further point developed in this chapter concerns the distinction between predicate-level *and* and *not* vs their sentence (propositional) level use. EFS relates them but ultimately still claims there are two *and*s: and^{t} and and^{et}. Matters presumably escalate as a greater variety of syntactic categories are considered: Transitive verbs: *He both praised and criticized each student*, Modifying adjectives: *a tall and handsome professor*; Adverbs: *He works slowly and carefully*. This discussion follows up a point adumbrated in chapter 2 concerning the difference between lexical expressions with a fixed interpretation over all models such as *is* and *not* and ones that are interpreted freely in the denotation set associated with their type. In addition now EFS notes that some modifying adjectives are restricting (subsective): *Tina is a tall pianist* entails *Tina is a pianist* but does not entail *Tina is tall*. In contrast, intersective adjectives such as *Chinese* do have the corresponding entailment: *Tina is a Chinese pianist* entails *Tina is Chinese* and *Tina is a pianist.* Ultimately EFS treats *Chinese* as ambiguous, having an *et* interpretation and also an (*et*,*et*) one.

Reflections on Chapter 3

1. An alert beginner may observe a serious issue with lambda abstraction: compositionality. Consider a paradigm use, $\lambda x_e \varphi$, φ of type *t*. (For our illustrative purposes here we assume all lambda binding is with variables of type e). Now how can we compositionally interpret $\lambda x_e \varphi$ if λx_e is a function with just two arguments, 0 and 1? This implies that $[\![\lambda x_e \varphi]\!]^M$ and $[\![\lambda x_e \psi]\!]^M$ would be the same (e,t) sets when φ and ψ have the same truth value. And in any model M with φ, ψ and θ logically independent, at least two of $[\![\lambda x_e \varphi]\!]^M$, $[\![\lambda x_e \psi]\!]^M$ and $[\![\lambda x_e \theta]\!]^M$ would denote the same set. This is obviously incorrect. And $[\![\lambda x_e$ (x is a doctor)$]\!]^M$ should hold of just the entities that $[\![$doctor$]\!]^M$ holds of. But if E^M has at least two elements (hence at least four subsets) there are more $[\![$doctor$]\!]^M$ denotable sets than $[\![\lambda x_e$(x is a doctor)$]\!]^M$ denotable ones.

This problem arises in various guises in chapters 5 and 6. Basically once we allow productive use of VBOs we must systematically modify the interpretations of all expressions to take functions from variables (of type *e*) into E as arguments, yielding as value objects in our original type for the expression. This approach is tedious, and soporific, but it preserves compositionality. As this point is not obvious let us illustrate it with the basic case: $\lambda x_e.\varphi$. Now, in a model M, φ is not interpreted not as a truth value but as a function from variables of type *e* into E. Such functions are called *assignments* (of values to the variables). When α is such an assignment, x a variable of type *e*, and a an element of E, write $\alpha^{x \to a}$ for that assignment which maps x to a and each variable $y \neq x$ to $\alpha(y)$. Then we interpret $\lambda x_e \varphi$ as follows: $[\![\lambda x_e \varphi]\!]^M(\alpha)(a) = [\![\varphi]\!]^M(\alpha^{x \to a})$. Here it is clear that the interpretation of $\lambda x_e \varphi$ is given as a function of that of φ. We have preserved compositionality by enriching the denotations of expressions.

These assignments might be thought of as a mathematical artifice, but it is more reasonable to think of them as (a first approximation to) *contexts*. An assignment α assigns values to free variables while holding constant the "freely" interpreted expressions (*smiles*, etc.). That is why variables are so named, their denotation may vary independently of how other expressions are interpreted in a model. Free variables (ones not in the scope of a variable binding operator, like λ) behave rather like pronouns that lack an "antecedent", as in *He's tired*, which we might assert while pointing to someone who is obviously tired. Here we say that the reference of *he* is given by "context". But, formally speaking, what are contexts? A first answer is that they are assignments, as they tell us what the denotations of unbound variables / pronouns are. Doubtless modeling contexts requires much more than this, but assignments are a start.

2. EFS treats derived expressions as being binary branching, facilitating interpretation by function application. But in my opinion this leads to two shortcomings. Many categories have natural subcategories of semantic significance which would be naturally represented with unary branching constituents. For example EFS cites certain entailment patterns, such as the mutual entailment of (8a,b) below, which hold when the subject NP is a proper noun but not when it is an arbitrary (quantified) NP:

(8) a. Tina thanked Mary and Tina praised herself $\Leftrightarrow$
    b. Tina (both) thanked Mary and praised herself
(9) a′. Some woman thanked Mary and some woman praised herself $\not\Rightarrow$
    b′. Some woman (both) thanked Mary and praised herself

EFS acknowledges this disparity and could have handled it by taking NP_{prop} as a subcategory of NP. But NP in general has type $((et,t)$ so it would have been necessary to state just what subset of these functions proper nouns denoted, and that would have rendered the initial semantics more complicated. So failure to capture this entailment paradigm—a primary goal of EFS recall—results from the need to keep the opening sections easy to read.

A similar, but easier case is the difference between intersective and merely restricting adjectives. The latter are a subtype of (*et*,*et*), say $(et,et)_{\text{rest}}$, whose functions F meet the condition that F(A) $\subseteq$ A. The intersective subcategory meets a stronger condition: F(A) $=$ A$\cap$F(E). So the Chinese pianists are the pianists who are Chinese individuals. (We note that the set of intersective functions is isomorphic to the set of subsets of E, the map sending each intersective F to F(E) being an isomorphism). This would obviate the need for two *Chinese*.

3. The binary branching commitment also (to my mind but not to everyone's) renders coordination (*both*)…*and*…, (*either*)…*or*…, *neither*…*nor*…unnecessarily awkward. If *and*, etc. takes two arguments directly then its natural representation is as a ternary (or even quaternary: *both p and q*) branching structure. To be sure we can treat [x and y] as [x [and y]] interpreting, as EFS does, *and*, etc. as its curried[1] variant: **and(y)** maps **x** to a function that maps **y** to whatever binary **and** maps the pair (**x**,**y**) to. But is this syntactic juggling semantically enlightening? It enforces binary branching but why should we care? There is certainly nothing cognitively problematic about the use of two place functions, either in logic (*and*, *or*,…), set theory ($\cap$, $\cup$) or elementary mathematics ($+$, $-$, $\cdot$).

Note that selectional restrictions treat conjuncts on a par: In *[x and y] laughed* both x and y are understood as the kinds of things that can laugh: *every child and every teacher laughed*…, **every child and every floor laughed;* but something properly subordinate to the first NP (DP) need not: *Every child on the floor laughed* does not imply that the floor laughed (mercifully). Similarly *No student in any class laughed at that joke* is natural, where the npi *any* is in the scope of *no*. But in **No student and any teacher laughed*…the npi is not natural as it is not c-commanded by *no student* (on the ternary branching view).

And syntactically [x [and y]] turns [and y] into an x-modifier. E.g. if x is an intransitive verb as in *Ed [laughed [and cried]]* we see that *and cried* has the same type (*et*,*et*) as an adverb, so they should coordinate: *[[and cried] and loudly]*, which seems nonsensical.

4. In positing different *and*s when combining with different categories EFS has not presented the semantic generalization uniting their meaning (even if different words are used for *and* with different categories). Given his work on boolean structure (Winter 2001) and his working familiarity with earlier work (Keenan and Faltz 1985) Winter could have presented enough boolean structure to characterize *and* (later *all*) and *or* (later *some*) as greatest lower bound and least upper bound operators. I interpret not doing so as, again, a concession to keeping the introduction introductory. But in this case it is more costly, as the entailment paradigms we observe in natural lan-

[1] There is some discussion in the literature about the origin of "currying". It is named for the logician Haskell B. Curry (1930), who built (explicitly) on the somewhat earlier work of Moses Schoenfinkel (1924). And Hindley and Seldin (2008) note that the core idea (but not the explicit formalization) is already present in Frege (1893). I give all references here.

guage depend very significantly (not entirely—the restricting condition on adjectives and adverbs is important) on the boolean connectives and universal and existential quantification.

5. EFS rightly draws attention to the difference between fixed denotation lexical items and ones that denoted freely in their type-set. It would go beyond the introductory nature of the text to state what their common semantic property of the fixed ones is. It is a property which also applies to the conditions that distinguish restricting and intersective adjectives.

6. Reflect for a moment on the difficulty in presenting an introductory semantics text starting with expressions as syntactically and semantically as simple as possible and building up to more complex ones. Imagine, for one totally exciting moment, that we started with Malagasy examples instead of English ones. How would the naive reader (assumed unfamiliar with Malagasy) feel confident that the semantic analysis was not drawing on unsuspected properties peculiar to Malagasy? This is a fair qualm, but it applies to English as well. EFS starts with *Tina is tall*, with a proper noun subject and a stative predicate, allowing us to avoid considering whether **tina** is Agent or Patient, or the predicate an unbounded state or an on-going activity, etc.

Now EFS interprets *tall* simply as a subset of the domain of the model. But whether Tina is tall depends on who we are comparing her to: maybe she is a tall first grader but not tall compared with all the kids in her school. So there is an unexpressed context dependency here, one that would have been absent had we chosen predicate nominals: *Tina is a doctor*. Unfortunately in such cases English uses an article *a*. English seems the odd man out here, as many languages (Malagasy, Hebrew, Russian) would just say *Tina doctor* (or *Doctor Tina*). Others might use a copula 'is' but without an article: French, Farsi, German. Yet others, Malay and Mandarin, would use both a copula and a numeral ('one') + classifier in front of *doctor*: *Tina is one-person doctor*. EFS treats both the *a* here and the *is* as the identity function of the appropriate type, but perhaps we should think of *a* as a kind of classifier. Surely we don't want to interpret *a* as the identity function in *A student called while you were out*. Note that EFS interprets *Tina is is is a is a tall* the same as *Tina is tall*, but this is in my judgment unproblematic. We are concerned to represent the meanings of English expressions, what we do with non-expressions in English is of no concern.

So languages vary even with regard to the syntactically and semantically simplest sentence types, rendering it difficult to pick a neutral, generic starting point of analysis. (And proper names also vary: in Greek and Catalan they carry particles expressing number and gender; in some W. Austronesian other features may be marked, such (roughly) adult vs. youngster in Malagasy).

Chapter 4 *Quantified Noun Phrases* is in many ways the reward for having learned the basic material in chapters 2 and 3. It exhibits denotations, called *generalized quantifiers* (GQ), for quantified NPs such as *every woman*, *some man*, *no cook* assigning them type (et,t) so they will combine with one place predicates (P_1s) to form Ss; semantically they map subsets A of E (or their characteristic functions) to truth values. And Determiners such as *every*, *some*, *no*, *exactly one*, *fewer than five, at least half* are assigned type $(et(et,t))$ and have fixed denotations which determine many natural entailment patterns.

Some simple patterns, about which our pretheoretical intuitions are quite good, concern *monotonicity* properties of GQs. A GQ F is (*monotone*) *increasing* (*upward monotone*) iff it meets the condition that when $A \subseteq B$ and $F(A) = 1$ then $F(B) = 1$. For example *every man*, *some man*, and *at least five men* are increasing: So since the people who ran rapidly in some model are a subset of those who ran it follows that *Every man ran rapidly* $\Rightarrow$ *Every man ran*. Similarly replacing *every man* with *some man*, *at least five men*, etc.

Dually a GQ F is *decreasing* (*downward monotone*) iff whenever $A \subseteq B$ and $F(B) = 1$ then $F(A) = 1$. *No man* and *fewer than five men* are decreasing: *No man ran* $\Rightarrow$ *No man ran rapidly*. In contrast *exactly one man* is neither increasing nor decreasing: *Exactly one man ran rapidly* $\not\Rightarrow$ *Exactly one man ran*. And *Exactly one man ran* $\not\Rightarrow$ *Exactly one man ran rapidly*.

These entailment patterns are, obviously, determined by the fixed interpretation of the Determiners. E.g. (**every**(A))(B) = 1 iff $A \subseteq B$; (***some***(A))(B) = 1 iff $A \cap B \neq \emptyset$, (**no**(A))(B) = 1 iff $A \cap B = \emptyset$ and (**exactlyone**(A))(B) = 1 iff $|A \cap B| = 1$, and (**at least half**(A))(B) = 1 iff $|A \cap B| \geq ½|A|$.

Monotonicity also applies directly to Dets. E.g. **some** is increasing on its argument since for all A, A′ with $A \subseteq A'$, (**some**(A))(B) $\Rightarrow$ (**some**(A′))(B), all B. So *Some tall student ran* $\Rightarrow$ *Some student ran* since **tall student** $\subseteq$ **student**.

Dually **no** is decreasing on its argument: *No student ran* $\Rightarrow$ *No tall student ran*. And **exactly one student** is neither upward nor downward monotone.

Of interest is that **every** is decreasing on its argument, but a GQ it builds, e.g. **every man** is increasing on its argument. So *Every man ran rapidly* $\Rightarrow$ *Every tall man ran rapidly*, which in turns entails *Every tall man ran* (as **every tall man** is increasing on its argument).

Thus Dets can be logically classified according as they are increasing, decreasing or neither on their two arguments. So **every** may be noted ↓**every**↑ meaning that it is decreasing on its (first) argument and increasing on its second (the argument of the NP it builds). **No** is noted ↓**no**↓, **some** as ↑**some**↑, and **at least half** as −**at least half**↑ as it is non-monotonic on its first argument but increasing on its second.

A strong appeal of this work is that the reader sees that monotonicity properties are not limited to just one or two lexical items, but rather represent very general properties shared by many (not just lexical) expressions.

EFS also notes the tantalizing correlation between the presence of negative polarity items (*any*, *ever*) and being in the scope of a monotone decreasing expression. Thus we find patterns like those in (10) and (11):

(10) a. No child saw any bears in the forest
 b. *Every / *Some child saw any bears in the forest
(11) a. No / Every child who saw any bears in the forest…
 b. *Some child who saw any bears in the forest

EFS discusses a final property of Determiner denotations known as *conservativity*, a very general (?universal) constraint on the (*et*,*t*) functions they can denote. A function D of type (*et*,*t*) is *conservative* iff for all sets A,B,

(12) $D(A)(B) = D(A)(A \cap B)$

This may appear a little "algebraic" but it is quite contentful. It says that to determine whether Det As are Bs at most we need to know which individuals are As and which of those are Bs. We don't need to know about Bs that are not As. For example a statement equivalent to conservativity is: for all sets A, B, B′ if $A \cap B = A \cap B'$ then $DAB = DAB'$.

Also in this chapter EFS considers coordination of NPs and their "distributive" behavior (which invokes a new *and*, $and^{(et,t)}$):

(13) Every man and every woman ran ⇔ Every man ran and every woman ran

Lastly this chapter shows how entities like **tina** can be "lifted" to what EFS calls *individual substitutes* $\mathbf{I_{tina}}$ of type (*et*,*t*), which maps an (*et*) function p to a truth value as follows: $\mathbf{I_{tina}}(p) = p(\mathbf{tina})$. (Set theoretically, $\mathbf{I_{tina}}(p^*) = 1$ iff $\mathbf{tina} \in p^*$). This allows us to interpret coordination of proper nouns with other proper nouns and quantified NPs: *John and Mary, John and every student*, etc.

Reflections on Chapter 4

1. This chapter has the widest appeal to a general audience, especially scholars in areas that neighbor linguistics. It offers several generalizations that are both not immediately obvious and which do depend straightforwardly and non-trivially on a formal semantic analysis. Specifically (1) the claims concerning the licensing of negative polarity items by decreasing functional expressions, and (2) the Conservativity constraint on possible Determiner (Quantifier) denotations in natural language. The former derives from the work of Fauconnier (1979) and Ladusaw (1983). The term *conservativity* is first used to characterize Determiner denotations in Keenan (1981). Barwise and Cooper (1981), cited in EFS, used *lives on* in the same sense. Keenan and Stavi (1986), cited in EFS, show that over an arbitrarily large but finite domain all conservative functions are denotable by English expressions, hence no constraint stronger than conservativity can hold.

2. Conservativity is quite non-trivial in that there are many functions from pairs of subsets of E into {0,1} which fail it. Consider $D(A)(B) = 1$ iff $|A| \leq |B|$. Clearly $D(\{a, b\}, (\{b, c\}) = 1$ but $D(\{a, b\}, (\{a, b\} \cap \{b, c\}) = D(\{a, b\}, \{b\}) = 0$, so this D fails conservativity.

3. It would have increased the generality of the claims concerning npi's if EFS had stated just what decreasing Dets and NPs had in common with predicate level *not*, which also licenses npi's. In type (e, t), $A \subseteq B \Rightarrow$ not $B \subseteq$ not A. That is, if all As are Bs then all non-Bs are non-As. So npi licensors are expressions that reverse the natural $\leq$ (partial order) relation on the set of objects they apply to.

4. Having classified Dets by the monotonicity of their two arguments it is natural to ask whether all 9 possible categories are realized. E.g. are there Dets which are increasing on their first argument but non-monotonic on their second? Are there constraints here we might posit as universal?

Chapter 5 *Long Distance Meaning Relationships* This chapter extends the class of expressions analyzed by focusing on object NPs: quantified, as in *Tina praised every student* and relativized, as in *some teacher that Mary praised*. This analysis leads

Winter to enrich the "architecture" of his formal semantics with the Lambek-van Benthem calculus, which addresses, in principle, the issue of defining explicitly the language we are semantically interpreting.

5.1a The occurrence of quantified objects, as in (14), constitutes a "type mismatch" according to EFS:

(14) Tina praised every student

EFS's problem is that *praise* has been assigned type $(e, (e, t))$ and *every student* type $((e, t), t)$, a pair of types to which function application (FA) does not apply. Now EFS already has at hand one solution to this problem: (15).

(15) (every student)(λx_e.Tina praised x)

In (15) the two major constituents have the right types for FA to apply, and its application yields the correct interpretation. This usage also suffices to represent classical scope ambiguities in Ss like *Every student praised some teacher*.

Reflection on 5.1a The above solution leaves the reader wondering why we do not interpret (15) directly (as we do *Tina praised Mary*). Do we really need recourse to an operator, λx_e, not denoted by any English expression? The answer is a clear NO. Keenan (2016) (not available when EFS was written) discusses this issue in detail. Keenan (1993, 2002) and Keenan and Westerstähl (1997), all provide a mathematically explicit direct interpretations of such expressions:

(16) For F a possible DP denotation and R a binary relation on E,

$$F(R) =_{def} \{a \in E|F(aR) = 1\}, \text{ where } aR \ =_{def} \{b \in E|(a,b) \in R\}$$

So $[\![$praise every student$]\!]$ = (**every**(**student**))(**praise**) is the set of entities x which are such that (**every student**) holds of the set of entities that x praised. Note crucially here that the value an NP denotation assigns to a binary relation R is determined by the value it assigns to the subsets of E (such as aR, often noted R_a). So we define a generalized quantifier by giving its values on the sets, as EFS does, and this uniquely determines its values at binary (in fact all n-ary) relations. (16) extends to n+1 ary relations in general by: $F(R) = \{a \in E^n|F(aR) = 1\}$.

So I find no reason to evoke undenotable operators here, just getting the domain of NPs right the first time does the trick. If we want to force denotations into a type notation we can enrich the notation to e.g. (e^{n+1}, e^n), writing e^0 for t. Or perhaps (p_{n+1}, p_n) would be better, acknowledging that NPs combine with n+1 place predicates to form n place ones.

5.1b EFS turns now to relative clauses, as in (17a,b):

(17) a. *that praised Mary* as in [*some* [*teacher* [*that* [*praised Mary*]]]]
b. *that Mary praised* as in [*some* [*teacher* [*that* [*Mary praised*]]]]

Praised Mary is assumed of type (e,t), so *that* combines with an (e,t) to yield something that combines with an (e,t), *teacher*, to yield *teacher that praised Mary*, of type (e,t). So EFS treats *that* in (17a) as of type $((e, t), (et, et))$ stipulated to, in effect, denote set intersection. However EFS finds interpreting (17b) more problematic as EFS does not refer to linear order, so there is no way to say that *Mary praised* is not a constituent. If it were interpreted just by FA given the type assignment to its parts it would mean **praise**(**mary**), which is incorrect for (17b).

The solution EFS offers is to posit a Function Introduction rule which introduces in effect a free variable, *u*, marked as one that will become bound. The (*e*,*t*) expression that *that* combines with denotes λu_e.**praise**(u)(**mary**). The same mechanism could have been used to derive subject relatives as well, with the post-*that* phrase denoting λu_e.**praise**(**mary**)(u).

EFS does draw an appealing analogy between hypothetical reasoning (to prove $p \rightarrow q$ assume p, derive q, then infer $p \rightarrow q$ from no assumptions) introducing free variables and then binding them. But this reader at least fails to see how this analogy helps us to build semantic representations. We already use lambda abstraction and that suffices.

Reflection on 5.2b As with *Tina praised every student* we can ask if there is not a direct way of interpreting *…teacher that Mary praised*. And there is, namely by function composition. Recall, if f is a function from A to B and g a function from B to C then $(g\circ f)$ is that function from A to C given by: $(g\circ f)(a) = g(f(a))$. Now note that *Mary* can have type $((e, t), t)$ and *praise* has type $(e, (e, t))$ so we could set $[\![\textit{Mary praise}]\!]^M =$ **mary**∘**praise** which is the correct value, mapping an object a of type *e* to **mary**(**praise**(a)). This analysis does imply that we have a grammar for the expressions we interpret, so we can see that *praise Mary* is a constituent of type (*e*,*t*) and *Mary* and *praise* are adjacent expressions that do not form a constituent. The generalization would be that in such cases, where FA does not apply, we can interpret the string by Function Composition. So Direct Compositionality does work in these simple cases. And it does as well in the "unbounded" binding in *…teacher that Tina believes Mary praised* that EFS considers. Whether it can handle all instances of long distance binding would have to be studied.

5.2 The last section of this chapter enriches the notion of expression to what EFS calls *linguistic signs*. These are ordered pairs (p,c) where p is perceptual representation—so it includes the phonological and syntactic structure of the expression, and c is its conceptual representation. So what we combine in building binary semantic representations are pairs of signs, each itself a pair, such as (*praise*, **praise**) and (*mary*,**mary**). This approach seems to me completely reasonable if notationally a little heavy, and if developed we could use it to clarify representations for strings with scope ambiguities, such as ((*not tall*)*and thin*) vs *not*(*tall and thin*). However the proposals in EFS are programmatic, asserting that the type theoretical approach can be extended to the perceptual component.

Chapter 6 *Intensionality and Possible Worlds*. This chapter concerns itself with the interpretation of expressions that depends on the conceptual meaning of others rather than the objects EFS has been calling their denotations. Two widely noted cases are *belief* contexts and transitive verbs of intent and desire (*seek*/*look for, want*). EFS treats *believe* as in (18) as taking a sentential argument, *Mary smiled* to yield an (*e*,*t*) expression:

(18) John [believes [Mary smiled]]

But the interpretation of *believes Mary smiled* in a model M is not a function of the truth of *Mary smiled* in M. EFS illustrates this by observing that (19a) does not entail (19b).

(19) a. Mary smiled, and Tina danced, and John believes Mary smiled
b. John believes Tina danced

Given (19a), *Mary smiled* and *Tina danced* have the same truth value, now called the *extension* of the sentence, but different conceptual content, now called their *intension*. Since their intensional interpretations are different, the function denoted by *believes* can take different values when applied to them. So *believe* and similar verbs expressing propositional attitudes (*doubt*, *hope*, etc.) are *intensional* verbs. The syntactic expressions following *believe* as in (19) are called *opaque* (or *non-transparent*). Characteristically these contexts do not admit of interchanging distinct names with the same denotation preserving the truth of the original S. Thus (20a,b) may have different truth values even though the two proper names denote the same individual.

(20) a. John believes that Lewis Carroll wrote *Alice in Wonderland*
b. John believes that Charles Dodgson wrote *Alice in Wonderland*

A second case, analogous to (20), is (21), where (21a) fails to entail (21b):

(21) a. All pianists are composers and all composers are pianists, and John is looking for a composer.
b. John is looking for a pianist

EFS notes that if *is looking for* is everywhere replaced in (21a,b) with *is talking to* the entailment is sound. So *is talking to* is extensional on its argument while *is looking for* is not. EFS also notes a few other opacity inducing expressions.

The (standard) approach EFS takes to representing intensional expressions is by enriching the notion of model with a new primitive, a non-empty index set W (whose elements are, often, called *possible worlds*). A new primitive type, s, is added with $D_s = W$. Ss now denote functions from W into {0,1}; equivalently, subsets of W, dubbed *propositions.* So they have type (s, t), and *smile* has type $(e, (s, t))$ and *praise* type $(e, e, (s, t))$. (Later the type of *smile* is enriched to $((s, e), (s, t))$, and analogously for *praise*). The logical operators like *and* are engineered to behave extensionally, so e.g. for p,q propositions, $\mathbf{and}^{st}$(p)(q) maps each w in W to $\mathbf{and}^{t}$(p(w),q(w)). (We ignore currying here). Then the crucial definition of entailment is given by: a sentence X *entails* a sentence Y iff for all models M, the set of worlds denoted by $[\![X]\!]^M$ is a subset of those denoted by $[\![Y]\!]^M$.

This semantics will allow models in which *Mary smiled* and*Tina danced* determine distinct but overlapping sets of worlds. Say *Mary smiled* maps to 1 just worlds w_1 and w_2, and *Tina danced* maps just w_2 and w_3 to 1. So the two Ss take the same value at w_2 but are interpreted as different propositions and so can be assigned different values by **believe**.

This chapter ends with *de re* / *de dicto* ambiguities, certainly of interest to the beginner. EFS represents them using lambda binding. Compare:

(22) a. Tina believes some Englishman won the race

b. Some Englishman $\lambda x_e.x$ won the race

(b), the *de re* interpretation, just says there is an individual (who happens to be English) about whom Tina believes that he won the race. She may not know that he is an Englishman. On the *de dicto* reading in (22a) Tina may have no idea which individual won the race, but she knows that only Englishmen were allowed to participate.

Reflections on Chapter 6

1. I would have found it helpful to have formal examples illustrating the difference between the representation of properly intensional expressions and properly extensional ones. For example *believe* is well illustrated as intensional, but we are given no S taking verb that is extensional. Similarly EFS informally distinguished between intensional *look for* and extensional *talk to* but does not say how to constrain the interpretation of *talk to* to make it extensional.

Another clear case are intensional vs extensional adjectives. Clearly in a world in which the composers and the pianists are the same it does not follow that the skillful composers and the skillful pianists are the same. But it does follow that the female composers and the female pianists are the same. It is not hard to state this condition and it would have been helpful to do so. Similarly while quantifiers like *some* are (correctly) forced to be extensional, can we not find any Determiners that are properly intensional? *Too many*, *Damn few* come to mind. The doctors and the lawyers might be the same in some world but *Too many lawyers attended the meeting* and *Too many doctors attended the meeting* might have different truth values in that world.

2. EFS is to be commended in my view for emphasizing that the "possible worlds" just constitute an index set with no structure. Winter likens them to E, the set of entities. But that is not quite right as we have expressions, like *Tina*, that denote elements of E (or whose extensions lie in E). So we think of elements of E as objects, including human beings, but also inanimate and abstract objects (that we can count). Thus we have some intuition regarding what is intended when we refer to elements of E or quantify over E. But not so with W. It is just a bare, non-empty set.

Why should the truth value of Ss vary with the choice of some unknown inscrutable "indices"? It makes sense that the truth of Ss varies with models since expressions may have different denotations in different models (even with the same universe) and whether Ss are true or not depends on what their constituents denote. But possible worlds cannot be models (just as sets cannot be members of themselves), they are proper parts of models.

One might respond to my dissatisfaction above by agreeing that adding a non-empty set of indices to our models is quite arbitrary, but it works! After all we can show that putative entailments in (21a,b) fail, which is desirable. We may not comprehend why it works, but let's keep riding a winner until we have a better option (or until we come to understand it better). And a step towards better understanding here would be to study expressions which, arguably, denote objects that the truth of Ss varies with (call them "possible worlds" if you have a poetic bent). It seems to me that we don't have far to look. Consider (23a,b,c):

(23) a. Ted is laughing
b. Ted is laughing in the kitchen

c. Ted is laughing in Ben's picture

(23a) expresses an activity which, without further specification, is understood to be taking place at a location (here) at a time (now, speech time). (23b) entails (23a), having **now** as default time as (23a) does, but specifying the location. (23c) however, similar in form to (23b), does not entail (23a) as it locates the event in a different spatio-temporal frame altogether, namely Ben's picture. Note that if the picture is held constant the S *Ted is laughing in Ben's picture now but he wasn't yesterday* is anomalous. So phrases like *Ben's picture*, *here and now*, *yesterday afternoon at Sam's place*, are candidates for denoting "possible worlds". They are of course locative and temporal phrases which comport an enormous amount of their own semantic structure (they are not merely sets of unstructured points or times; they carry ordering relations, etc). See de Swart (1996) for an insightful start. And of course adding them to our language would restore ontological perfection.

3. A last, well recognized issue with possible world semantics (not specific to EFS; see p.225) is that intensional expressions like *believe* must assign the same value to different arguments that are logically equivalent and so have the same value at every possible world index. Thus in any model, *John believes* (*that*)*no poets are vegetarians* must have the same extension (truth value) in every world as *John believes* (*that*)*no vegetarians are poets*. In this case we may hesitate, as the logical equivalence of the sentential objects of *believes* is fairly obvious. But as the sentential objects become increasingly different syntactically their logical equivalence may become increasingly less obvious and thus it is increasingly plausible that a rational person could believe one without having an opinion about the other. Plausibly for example John could believe (24a) but not have an opinion about (24b). (Example adapted from Peters and Westershåhl (2006)):

(24) a. There are more linguists than logicians
b. There are more linguists who are not logicians than logicians who are not linguists

In sum EFS is a clear, readable, and most useful introduction to formal semantics of natural language—both the semantics part and the mathematical formulation.

References

Barwise, J., & Cooper, R. (1981). Generalized quantifiers in natural language. *Linguistics and Philosophy*, *4*, 159–219.

Curry, H. B. (1930). Grundlagen der Kombinatorischen Logik. *American Journal of Mathematics*, *52*(3), 509–536.

De Swart, H. (1996). Quantification over time. In J. van der Does & J. van Eijck (Eds.), *Quantifiers, logic and language* (pp. 311–337). Stanford: CSLI.

Dehaene, S. (2014). *Consciousness and the brain: Deciphering how the brain codes our thoughts*. New York: Viking Penguin.

Dennett, D. C. (2017). *From bacteria to Bach and back: The evolution of minds*. New York City: W.W Norton & Co.

Erlewine, M. Y. (2017). Review of *Elements of formal semantics*: An introduction to the mathematical theory of meaning in natural language. *Computational Linguistics*, *42*(4), 837–839.

Fauconnier, G. (1979). Implication reversal in natural language. In F. Guenthner & S. Schmidt (Eds.), *Formal semantics for natural language*. Dordrecht: D. Reidel.

Frege, G. (1893). *Grundgesetze der Arithmetik.* Vol 1. Section 4. Verlag Hermann Pohle, Jena. Noted in Hindley and Seldin 2008. Footnote 2, p. 3.

Hindley, J. R., & Seldin, J. P. (2008). *Lambda-Calculus and Combinators*. New York: CUP.

Heim, I., & Kratzer, A. (1998). *Semantics in generative grammar*. Malden: Blackwell.

Keenan, E. L. (1981). A Boolean approach to semantics. In J. Gronendijk, et al. (Eds.), *Formal methods in the study of language* (pp. 343–379). Amsterdam: Mathematics Center, University of Amsterdam.

Keenan, E. L. (1982). Eliminating the universe (a study in ontological perfection). In D. Flickinger, et al. (Eds.), *WCCFL 1*. Stanford: Stanford Linguistics Association.

Keenan, E. L. (1993). Natural language, sortal reducibility and generalized quantifiers. *The Journal of Symbolic Logic*, *58*(1), 314–325.

Keenan, E. L. (2002). Some properties of natural language quantifiers: Generalized quantifier theory. *Linguistics and Philosophy*, *25*, 627–654.

Keenan, E. L. (2016). In situ interpretation without type mismatches. *Journal of Semantics*, *32*(1), 1–20.

Keenan, E. L., & Faltz, M. L. (1985). *Boolean semantics for natural language*. New York City: D. Reidel.

Keenan, E. L., & Moss, L. S. (2016). *Mathematical structures in language*. CSLI Lecture Notes No. 218.

Keenan, E. L., & Stavi, J. (1986). A semantic characterization of natural language determiners. *Linguistics and Philosophy*, *9*, 253–326.

Keenan, E. L., & Westerstähl, D. (1997). Generalized quantifiers in linguistics and logic. In J. van Benthem & A. ter Meulen (Eds.), *Logic and language* (pp. 837–895). Amsterdam: North Holland.

Ladusaw, W. (1983). Logical form and conditions on grammaticality. *Linguistics and Philosophy*, *6*, 177–197.

Liang, P., & Potts, C. (2015). Bringing machine learning and compositional semantics together. *Annual Review of Linguistics*, *1*(1), 355–376.

Montague, R. (1973). In J. Hintikka, J. Moravcsik & P. Suppes (Eds.), *Approaches to Natural Language: Proceedings of the 1970 Stanford Workshop on Grammar and Semantics* (pp. 221–242). Dordrecht: D. Reidel Pub. Co. (Reprinted in *Formal Philosophy*, pp. 247–271, by R. Thomason, Ed., 1974, New Haven: Yale University Press.

Peters, S., & Westershåhl, D. (2006). *Quantifiers in language and logic*. Oxford: OUP.

Rett, J. (2016). Book review of Yoad Winter's. Elements of formal semantics. *Glossa*, *1*(1), 42.

Schoenfinkel, M. (1924). Ueber die Bausteine der mathematischen Logik. *Mathematische Annalen, 92*, 305–316. (Trans. In J. Van Heijenoort *From Frege to Gödel.* Harvard University Press 1967, pp. 355–366.)

Wigner, E. P. (1960). The unreasonable effectiveness of mathematics in the natural sciences. *Communication on Pure and Applied Mathematics*, *13*, 1–14.

Winter, Y. (2001). *Flexibility principles in Boolean semantics*. Cambridge: MIT Press.