Chapter 1
Basic Categorial Syntax

Chapter 1 Basic Categorial Syntax
1. The Task of Grammar
2. Artificial versus Natural Languages
3. Recursion
4. Category-Governed Grammars
5. Example Grammar – A Tiny Fragment of English
6. Type-Governed (Categorial) Grammars
7. Recursive Definition of Types
8. Examples of Types
9. First Rule of Composition
10. Examples of Type-Categorial Analysis
11. Quantifiers and Quantifier-Phrases
12. Compound Nouns
13. Perils of Concatenation
14. Adjectives Reconsidered
15. Copular-Be (is-of-predication)
16. Transitive-Be (is-of-identity)
17. Indefinite Articles
18. Adjectival Prepositions
19. Relative-Pronouns and Relative-Clauses
20. Restrictive Relative Clauses
21. Syntactic-Categories versus Syntactic-Types
A. Appendices
22. Appendix 1 – Summary of Basic Categorial Syntax
23. Appendix 2 – Examples of Type Assignments
24. Appendix 3 – Grammatical Niceties
25. Appendix 4 – Rank and Order
26. Appendix 5 – Comparison with Montague's Typology
1. The Task of Grammar

Language use consists in processing (including producing and recognizing) signs/symbols, which are variously realized by the following sorts of things.¹

(1) gestures gesticular language²
(2) sounds spoken language
(3) graphs written language

The signs of a language can generally be de-composed into smaller components, and ultimately into smallest (elementary) components.³ Conversely, the elementary components can be re-combined into compounds, the number of possible compounds being infinite. On the other hand, not every possible compound is admissible in the given language.

The principal concern of grammar is, for a given language, to demarcate symbol-combinations into those that are admissible and those that are inadmissible. As originally envisaged by Chomsky, grammar has three components, respectively known as syntax, semantics, and phonology.⁴ These three components of grammar are distinct inasmuch as they judge admissibility on the basis of different criteria. Semantics judges admissibility with regard to meaning, and phonology judges admissibility with regard to pronunciation.⁵ On the other hand, syntax judges admissibility without regard to either meaning or pronunciation, but by more abstract criteria that ultimately can only be described as "syntactic". Syntax is that part of grammar that remains after you remove meaning and pronunciation.

We begin our investigation with syntax, since that is generally thought to be the core of grammar; indeed, for many, ‘grammar’ just means syntax. The idea is that, central to our language-processing facility is a syntactic-module, which delivers material to the other two language modules for further processing – the phonetic-module (pronunciation) and the semantic-module (understanding).

Since we are ultimately concerned with semantics, we will be primarily interested in the syntactic material that is delivered to the semantic-module, and we will largely ignore the material that is delivered to the phonetic-module. This allows us to examine considerably simpler syntactic models than we would have to otherwise.

2. Artificial versus Natural Languages

This brings us to the distinction between artificial and natural languages. An artificial language is created from scratch, and its grammar is simply a matter of proclamation. By contrast, a natural language is a social-historical phenomenon whose grammar must be discovered. Natural languages, like most realms of the natural world, are profoundly complex, and uncovering their grammar is an on-going (never-ending) enterprise, just like other branches of science.

In recent years, it has become fairly standard practice to use artificial-language tools to study natural-language grammars.⁶ This is accomplished by examining various well-defined fragments of natural language, and treating them (in effect) as artificial languages. The goal then is to produce ever-richer and more accurate fragments of the natural languages under scrutiny.

---

¹ Most human communication involves all three symbolic components – sounds, gestures, graphs – usually with intimate connections among them. The following joke illustrates: a Roman walks into a bar, and holds up his index and middle finger, and says “five beers please”.

² Including mostly importantly sign-languages.

³ For example, there are smallest sound-elements (phonemes), smallest gestural-elements (gestemes), and smallest graphical-elements (graphemes). Note that the constituents of a compound expression need not themselves be meaningful. For example, ‘trees’ divides phonetically into ‘tree’ and ‘s’, which both have meaning, whereas ‘tree’ phonetically divides into ‘tr’ and ‘ee’, which do not have meaning.


⁵ Narrowly understood, phonology pertains to pronunciation, and accordingly only applies to spoken languages – or "tongues" as they are sometimes called. Bear in mind that ‘language’ and ‘linguistics’ derive from the Latin word ‘lingua’, which means ‘tongue’. Broadly understood, in this context, phonology pertains to whatever procedure is used to physically realize the symbols of a language.

⁶ Montague, “English as a Formal Language”.
3. Recursion

A person who knows a language knows its grammar – which, at a minimum, means that he/she can judge which symbolic compounds are admissible, and which are inadmissible. Indeed, a language user is capable of generating/recognizing infinitely-many admissible compounds, and avoiding/rejecting equally many inadmissible compounds.

How is this possible? For our purposes, the neurological details are of less concern than the theoretical underpinnings. Suffice it to say that, for us at least, the working hypothesis is that our brains/minds have somehow encoded/programmed in them a generative grammar, which in particular consists of the following two components.\(^7\)

1. finitely-many basic elements
2. finitely-many rules for forming compounds

If there are few enough basic elements and rules, then they can be encoded in the brain. The remaining question then is how finite resources can generate infinitely-many grammatical products. The trick that the mind uses is known as recursion.

In Britain and her many colonies at least, children were once taught the concept of linguistic recursion at a very young age. As with many of life’s early lessons, this lesson was taught via a nursery rhyme, which in this particular instance goes as follows.

```
this is the house that Jack built
this is the malt that lay in the house that Jack built
this is the rat that ate the malt that lay in the house that Jack built
this is the cat that killed the rat that ate the malt that lay in the house that Jack built
```

The lesson of this indefinitely-long poem is the principle of recursion in English. In particular, English has built-in the ability to construct phrases of arbitrary length, and hence the ability to construct infinitely-many phrases.\(^8\)

What makes a rule-system recursive? We propose the following semi-formal definition.

An input-output procedure is recursive if and only if it produces output that it also admits as input.

A fairly simple concrete example of a recursive procedure is agriculture. We plant corn-seeds, which we grow into corn-plants, from which we harvest corn-seeds, some of which we eat, and some of which we recycle back into the process. It is the recycling of seed-corn back into the process that is the key to the success of the process, which enables it to be sustained indefinitely.\(^9\)

The same holds for recursive grammars, which we informally define as follows.

A grammar is recursive if and only if it involves constructions that produce output that the constructions also admit as input.

4. Category-Governed Grammars

One way to make a grammar recursive is to make it category-governed. To say that a grammar \(\mathcal{G}\) is category-governed is to say that it satisfies the following two principles.

1. every admissible item of \(\mathcal{G}\) is assigned a category;
2. no item figures in a grammatical construction except in virtue of its category.

---

\(^7\) This thesis forms the basis of Steven Pinker's *Words and Rules: The Ingredients of Language*.

\(^8\) Of course, most of the phrases cannot be physically-realized – not because of grammar, but because of the briefness of our time in this world.

\(^9\) There are natural limitations to agriculture, including the finiteness of key ingredients, including the finiteness of the sun, and even the universe, not to mention the finiteness of our culture and our species.
In traditional syntax, categories very closely resemble the *parts-of-speech* that appear in dictionaries. For example, ‘lie’ is both a *verb* and a *noun*, the latter categories being very prominent in traditional grammar.

In this connection, the following is a relevant definition.

> A syntactic category is a set of words and/or phrases in a language which share a significant number of common characteristics. The classification is based on similar structure and sameness of distribution (the structural relationships between these elements and other items in a larger grammatical structure), and not on meaning. In generative grammar, a syntactic category is symbolized by a node label in a constituent structure tree.

### 5. Example Grammar – A Tiny Fragment of English

By way of illustration, we consider a fairly simple grammar for a tiny fragment of English, which we call $G$. First, $G$ proposes the following categories, which are accompanied by our proposed abbreviations.

1. *sentence* $S$
2. *proper-noun* $PN$
3. *noun-phrase* $NP$
4. *verb-phrase* $VP$
5. *transitive-verb* $TV$
6. *attitude-verb* $AV$

Second, $G$ proposes the following fundamental elements and category assignments.

1. (e1) *believes* attitude-verb
2. (e2) *respects* transitive-verb
3. (e3) *Jay* proper-noun
4. (e4) *Kay* proper-noun
5. (e5) *Elle* proper-noun

Finally, $G$ proposes the following formation rules.

1. (n1) a proper-noun is a noun-phrase;
2. (n2) nothing else is a noun-phrase.
3. (s1) a noun-phrase followed by a verb-phrase is a sentence;
4. (s2) nothing else is a sentence.
5. (v1) a transitive-verb followed by a noun-phrase is a verb-phrase;
6. (v2) an attitude-verb followed by a sentence is a verb-phrase;
7. (v3) nothing else is a verb-phrase.

Notice that $G$ is recursive, since sentences go into making verb-phrases, and verb-phrases go into making sentences, which means that $G$ generates infinitely-many sentences.

The following is an example sentence, together with its phrase-structure.

---

10 From SIL.org.

11 These fundamental elements have further parts – both graphical and phonetic – but these further parts are not considered grammatically relevant to $G_0$. A more detailed grammar would uncover these additional details.

12 The mathematical structure depicted here is known as a *tree-ordered set*, or simply a *tree*. See Appendix for more details.
Jay believes Kay respects Elle

Concerning infinite-generation, notice that this sentence can be embedded in the following matrix,

Kay believes ( … )

and the resulting sentence can be embedded in the following matrix,

Elle believes ( … )

and so on ad infinitum.

6. Type-Governed (Categorial) Grammars

A special sort of category-governed grammars are type-governed grammars, which are also called categorial grammars. We prefer the former nomenclature, and we propose to employ the term ‘type’ to refer to the affiliated categories, and we propose to employ the term ‘category’ to refer to traditional categories, such as NP (noun phrase) and VP (verb phrase). Also, in order to distinguish syntax from semantics, we will occasionally prefix the terms ‘syntactic’ and ‘semantic’.

What makes type-governed (i.e., categorial) grammars special is that the categories (i.e., types) they propose are recursively constructed. In particular, a type-governed grammar proposes:

1. an initial finite set of primitive types,
2. finitely-many rules for constructing derivative types,

in virtue of which the grammar proposes infinitely-many types.

There are many ways of implementing a type-governed grammar. For example, according to the scheme originally proposed by Kazimierz Ajdukiewicz (1935), the types include just two primitive types.

<table>
<thead>
<tr>
<th>N</th>
<th>nouns</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>sentences</td>
</tr>
</tbody>
</table>

An immediate problem arises, since nouns seem to divide into two sub-categories.¹³

<table>
<thead>
<tr>
<th>proper-nouns</th>
<th>Jay, Kay, Elle, Rex</th>
</tr>
</thead>
<tbody>
<tr>
<td>common-nouns</td>
<td>dog, cat, mouse, person</td>
</tr>
</tbody>
</table>

Accordingly, some authors propose two primitive noun-categories rather than just one, given as follows.¹⁴

---

¹³ This is more obvious to a native speaker of English than it would be to a native speaker of Polish, such as Ajdukiewicz. In Polish, but not English, common nouns serve as subjects and objects of verbs, just like proper nouns.

¹⁴ This parallels David Lewis (1970) who proposes types n (definite noun), c (common-noun), and s.
In addition to primitive types, there are derivative types, which are generated by rules of type-composition. According to Ajdukiewicz, these types correspond to functors. Whereas nouns and sentences are thought of as complete, functors are thought of as incomplete, in the sense that they have gaps that call for complementary phrases.

As a simple example of a functor, consider the word ‘the’. One strongly senses the incompleteness of this expression; if one hears ‘the’ used by itself, one naturally asks “the what?”. In other words, the expression ‘the’ is missing something – a complementary phrase. One way to complete ‘the’ is to affix a common noun – e.g.,

| dog | cat | house | barn |

by which we obtain a noun phrase – e.g.:

| the dog | the cat | the house | the barn |

Functors are analogous to functions, so we adopt/adapt the customary mathematical terminology. In particular, we can regard a functor as a function, the complementary phrase as its argument, and the resulting phrase as its value, as in the following diagram.

Indeed, this analogy is the founding idea of categorial grammar, which categorizes each functor according to:

1. what type of phrase it takes as input.
2. what type of phrase it produces as output.

The notation for functor-types varies from author to author. The most common notation goes as follows.

For example, using this notation, we can type-categorize ‘the’ as follows.

| type(the) | = | D / C |

In other words, ‘the’ produces a definite-noun-phrase (D) given a common-noun-phrase (C).

For various reasons, to be elucidated as we proceed, we prefer the following notation.

---

15 This category corresponds roughly to the category singular-term in logic. Basically, this type applies to any expression that behaves – syntactically and semantically – like a proper noun. Note carefully that this is not a purely-syntactic category, since semantic criteria are involved in identifying which phrases have this type.

16 The word “functor” has numerous uses; for us a functor is a syntactic object that purports to denote a function, which is a mathematical object.

17 Note that we concentrate for the moment on one-place functors. We consider multi-place functors later.

18 In particular, in Category Theory, A→B signifies the functions from A to B. The arrow-notation is also analogous to Chemistry, where arrow means in effect “yields”. Most importantly, the arrow-notation is analogous to Logic, where arrow means “if…then…”. This analogy becomes the cornerstone of Expanded Categorial Grammar.
<table>
<thead>
<tr>
<th>input type</th>
<th>&quot;yields&quot;</th>
<th>output type</th>
</tr>
</thead>
<tbody>
<tr>
<td>( T_1 )</td>
<td>( \rightarrow )</td>
<td>( T_2 )</td>
</tr>
</tbody>
</table>

In particular:

<table>
<thead>
<tr>
<th>a phrase of type</th>
<th>( T_1 )</th>
<th>( \rightarrow )</th>
<th>( T_2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>takes a phrase of type</td>
<td>( T_1 )</td>
<td>and produces a phrase of type</td>
<td>( T_2 )</td>
</tr>
</tbody>
</table>

Using this notation, we type-categorize ‘the’ as follows.

\[
\text{type(\text{the})} = C \rightarrow D
\]

In other words, ‘the’ takes a common-noun-phrase (\( C \)) and produces a definite-noun-phrase (\( D \)).

### 7. Recursive Definition of Types

As mentioned earlier, types are recursively generated from:

1. finitely-many primitive types;
2. finitely-many rules for constructing derivative types.

So far, our account can be formally specified by the following inductive definition.

<table>
<thead>
<tr>
<th>(1)</th>
<th>S is a type</th>
<th>sentences</th>
<th>primitive types</th>
</tr>
</thead>
<tbody>
<tr>
<td>(2)</td>
<td>D is a type</td>
<td>definite noun phrases</td>
<td></td>
</tr>
<tr>
<td>(3)</td>
<td>C is a type</td>
<td>common noun phrases</td>
<td></td>
</tr>
<tr>
<td>(4)</td>
<td>if ( A ) and ( B ) are types, then so is ( (A \rightarrow B) )</td>
<td>monadic functors</td>
<td>derivative types</td>
</tr>
<tr>
<td>(5)</td>
<td>nothing else is a type</td>
<td></td>
<td>extremal clause</td>
</tr>
</tbody>
</table>

### 8. Examples of Types

There are, in principle, infinitely-many types generated by this scheme. First, there are the 0-level types.

\[
(0) \quad \text{D} \quad \text{C} \quad \text{S}
\]

These can be combined via \( \rightarrow \) to make the following 1-level types.

\[
(1) \quad \text{D} \rightarrow \text{D} \quad \text{D} \rightarrow \text{C} \quad \text{D} \rightarrow \text{S} \quad \text{C} \rightarrow \text{D} \quad \text{C} \rightarrow \text{C} \quad \text{C} \rightarrow \text{S} \quad \text{S} \rightarrow \text{D} \quad \text{S} \rightarrow \text{C} \quad \text{S} \rightarrow \text{S}
\]

The resulting cumulative collection can in turn be combined via \( \rightarrow \) to make the following 2-level types, among others.

\[
(2) \quad \text{S} \rightarrow (\text{S} \rightarrow \text{S}) \quad \text{D} \rightarrow (\text{D} \rightarrow \text{S}) \quad \text{C} \rightarrow (\text{D} \rightarrow \text{S}) \quad (\text{D} \rightarrow \text{S}) \rightarrow \text{C} \quad (\text{D} \rightarrow \text{S}) \rightarrow \text{S} \quad (\text{D} \rightarrow \text{S}) \rightarrow (\text{D} \rightarrow \text{S})
\]

The resulting cumulative collection can in turn be combined via \( \rightarrow \) to make the following 3-level types, among others.

\[
(3) \quad \text{S} \rightarrow (\text{S} \rightarrow (\text{S} \rightarrow \text{S})) \quad \text{D} \rightarrow (\text{D} \rightarrow (\text{D} \rightarrow \text{S})) \quad \text{C} \rightarrow [(\text{D} \rightarrow \text{S}) \rightarrow \text{S}] \quad \text{D} \rightarrow [(\text{D} \rightarrow \text{S}) \rightarrow \text{S}]
\]
As the reader probably suspects, most of the available types do not correspond to phrases in English or any other natural language! Nevertheless, they are there waiting in case we need them. Examples of types that do serve natural language analysis include the following.

<table>
<thead>
<tr>
<th>type</th>
<th>category</th>
<th>input</th>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td>D→D</td>
<td>function-sign</td>
<td>DNP</td>
<td>DNP</td>
</tr>
<tr>
<td>S→S</td>
<td>sentence-adverb</td>
<td>sentence</td>
<td>sentence</td>
</tr>
<tr>
<td>D→S</td>
<td>predicate</td>
<td>DNP</td>
<td>sentence</td>
</tr>
<tr>
<td>S→D</td>
<td>subnective</td>
<td>sentence</td>
<td>DNP</td>
</tr>
<tr>
<td>D→(D→S)</td>
<td>transitive verb</td>
<td>DNP</td>
<td>predicate</td>
</tr>
<tr>
<td>(D→S)→S</td>
<td>2nd-order predicate</td>
<td>predicate</td>
<td>sentence</td>
</tr>
<tr>
<td>((D→S)→S)→S</td>
<td>3rd-order predicate</td>
<td>2nd-order predicate</td>
<td>sentence</td>
</tr>
<tr>
<td>(D→S)→(D→S)</td>
<td>predicate-adverb</td>
<td>predicate</td>
<td>predicate</td>
</tr>
</tbody>
</table>

9. **First Rule of Composition**

The remaining question is how do we utilize types to formulate rules of grammar by which we analyze the grammatical structure of English sentences. This is accomplished by stating type-governed composition-rules. This turns out to be an ongoing enterprise, which we begin by stating the most fundamental rule of (type-governed) composition.

<table>
<thead>
<tr>
<th>Function-Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>a phrase of type</td>
</tr>
<tr>
<td>combines with a phrase of type</td>
</tr>
<tr>
<td>to produce a phrase of type</td>
</tr>
</tbody>
</table>

10. **Examples of Type-Categorial Analysis**

By way of illustration, we begin with our earlier phrase ‘the dog’, which can be analyzed as follows.

1. **the dog**

   ![Diagram of the dog analysis](image)

In other words, we propose that ‘the dog’ has type D, and is constructed from ‘the’ (C→D) and ‘dog’ (C) via Function-Application.

A slightly more complicated example goes as follows.

---

19 One is reminded of the final line of Milton's poem ‘On his Blindness’: They also serve who only stand and wait.
2. the brown dog

D

C→D

the

C→C

brown

dog

Notice that this analysis treats ‘brown dog’ as a common-noun-phrase (CNP), just like ‘dog’. On the other hand, ‘brown’ is treated as an adjective (CNP-modifier), which has type C→C. Notice that ‘brown dog’ (C) is constructed from ‘brown’ (C→C) and ‘dog’ by function-application.

The following is our first example phrase that is a sentence.

3. the brown dog sees the cat

S

D

C→D

the

C→C

brown

dog

D

C→D

sees

the

cat

This introduces two new types.

<table>
<thead>
<tr>
<th>type</th>
<th>category</th>
<th>instance</th>
</tr>
</thead>
<tbody>
<tr>
<td>D→S</td>
<td>verb phrase</td>
<td>sees the cat</td>
</tr>
<tr>
<td>D→(D→S)</td>
<td>transitive verb</td>
<td>sees</td>
</tr>
</tbody>
</table>

This illustrates a big difference between Linguistics and Logic concerning the analysis of transitive verbs. According to elementary logic textbooks, one understands a sentence like

4. Jay respects Kay

as constructed from three directly-subordinate phrases, as follows.

S

D

Jay

D^2→S

respects

D

Kay

Notice the type D^2→S, which is a polyadic type. In logic, phrases of this type are known as two-place predicates, whereas phrases of type D→S are known as one-place predicates. Contrast this analysis with the following.

S

D

Jay

D→(D→S)

respects

D

Kay

The theoretical difference between these two analyses is that the latter analysis, but not the former, treats the string ‘respects Kay’ as a constituent, which is to say a relatively-autonomous-meaningful unit.

Is there evidence that we "hear" (process) this intermediate string? In this connection, consider the following expanded example.

---

20 We examine polyadic types in detail in a later Chapter.
5. Jay respects Kay, and so does Elle

As we process this string, when we get to ‘so does’ we look for a phrase that ‘so does’ alludes to. What does ‘so does’ allude to? What is it that Elle also does? Answer: ‘so does’ alludes to ‘respects Kay’; what Kay also does is *respect Kay*. It accordingly seems plausible to propose that ‘respects Kay’ is a constituent.

The syntactic analysis then looks thus, where we treat ‘so does’ as a lexical-phrase.\(^{21}\)

\[
\begin{array}{c}
S \\
| \downarrow \downarrow \downarrow \\
D \rightarrow (D \rightarrow S) & D \rightarrow (S \rightarrow (S \rightarrow S)) & D \rightarrow S & D \\
\text{Jay} & \text{and} & \text{So-does} & \text{Elle} \\
\text{respects Kay} & \text{respects Kay} & \text{so-does Elle} & \text{respects Kay}
\end{array}
\]

Note that this treats ‘and so does Elle’ as a constituent, just as it treats ‘respects Kay’ as a constituent. Whereas there is abundant linguistic evidence for treating ‘respects Kay’ as a constituent,\(^{22}\) the linguistic evidence for treating ‘and so does Elle’ as a constituent is not so compelling. Rather, the chief reason for treating ‘and so does Elle’ as a constituent is that it allows us to exclusively employ binary-branching and unary-functors in our grammatical analysis.

The above analysis contrasts with the conventional logical analysis of ‘and’, according to which it is a two place connective, which is to say its type is \(S^2 \rightarrow S\).\(^{23}\) If we pursue this treatment of ‘and’, the above sentence is analyzed as follows.\(^{24}\)

\[
\begin{array}{c}
S \\
| \downarrow \downarrow \downarrow \\
D \rightarrow (D \rightarrow S) & D \rightarrow (S \rightarrow (S \rightarrow S)) & D \rightarrow S & D \\
\text{Jay} & \text{and} & \text{So-does Elle} & \text{So-does Elle} \\
\text{respects Kay} & \text{respects Kay} & \text{so-does Elle} & \text{respects Kay}
\end{array}
\]

11. Quantifiers and Quantifier-Phrases

Consider the following sentences.

6. every woman respects Kay
7. some woman respects Kay
8. no woman respects Kay

The words ‘every’, ‘some’, and ‘no’ are prominent examples of *quantifiers*, which are governed by the following basic syntactic rule.

---

\(^{21}\) By a *lexical phrase*, we mean a phrase whose meaning is not computed from its syntactic parts, but is rather read directly from the lexicon [dictionary].

\(^{22}\) We have only considered one piece of evidence.

\(^{23}\) This is a polyadic type, which we examine in detail in a later chapter.

\(^{24}\) We reconsider this analysis in Chapter 10: Finitary Junctions.
A quantifier followed by a common-noun-phrase is a quantifier-phrase,\textsuperscript{25} which is a species of noun-phrase.\textsuperscript{26}

We accordingly have the following phrase structure for the first example above.

\[
\begin{array}{c}
S \\
| \\
NP \\
| \\
QP \\
| \\
respects Kay \\
\triangleleft 27 \\
| \\
Q \\
| \\
every \\
| \\
CNP \\
| \\
woman \\
\end{array}
\]

Here, we employ traditional syntactic categories. How do we convert these into types? As an initial conjecture, consider the following type-rendering of this structure.

\[
\begin{array}{c}
S \\
| \\
D \\
| \\
C \\
| \\
\triangleright S \\
| \\
\triangleleft \\
| \\
| \\
\triangleleft \\
\end{array}
\]

In other words, ‘every’ has the same type as ‘the’, and so ‘every woman’ has the same type as ‘the woman’, which has the same type as ‘Kay’.

Syntactically speaking, this seems entirely plausible; quantifier-phrases behave syntactically just like determiner-phrases,\textsuperscript{28} which behave just like proper-nouns. In particular, a determiner-phrase and a quantifier-phrase are syntactically permitted wherever a proper-noun is permitted.

On the other hand, semantically speaking, quantifier-phrases don’t behave like determiner-phrases and proper-nouns. For example, the proper-noun ‘Kay’ denotes a particular individual; similarly, the definite-noun-phrase ‘the brown dog’ denotes a particular individual. On the other hand, the quantifier-phrase ‘every woman’ does not denote a particular individual.\textsuperscript{29}

The syntactic data we wish to reproduce is that a quantifier-phrase followed by a verb-phrase is a sentence. One way to achieve this categorially is to treat QPs as definite-noun-phrases. In this case, a VP takes a QP as argument and yields a sentence, as in the following diagram.

<table>
<thead>
<tr>
<th>argument</th>
<th>function</th>
</tr>
</thead>
<tbody>
<tr>
<td>⟨every woman⟩</td>
<td>⟨respects Kay⟩</td>
</tr>
</tbody>
</table>

But this does not work semantically for QPs. Another way to achieve this categorially is to turn the tables, and treat the VP as the argument and treat the QP as the functor, as in the following diagram.

<table>
<thead>
<tr>
<th>function</th>
<th>argument</th>
</tr>
</thead>
<tbody>
<tr>
<td>⟨respects Kay⟩</td>
<td>⟨every woman⟩</td>
</tr>
</tbody>
</table>

\textsuperscript{25} More generally, quantifier is a sub-species of determiner, which includes non-quantifier words like ‘the’, ‘this’, and ‘my’. Accordingly, quantifier-phrase is a sub-species of determiner-phrase, which is a sub-species of noun-phrase.

\textsuperscript{26} See appendix for discussion of "X phrases".

\textsuperscript{27} The symbol ‘\(\triangle\)’ indicates that the phrase has further structure that we (momentarily) ignore or take for granted.

\textsuperscript{28} See footnote 25.

\textsuperscript{29} We are currently concentrating on singular constructions. Whereas singular proper nouns denote particular singular-entities, plural proper nouns denote particular plural-entities. The key concept is particular, not individual.
In that case, it is natural to propose the following type-categorization.\footnote{Type-categorization is a way to assign types to expressions in a logical or formal system.}

\[
\text{type}(QP) = (D \rightarrow S) \rightarrow S
\]

We then obtain the following syntactic analysis of our example sentence.

Notice the affiliated type-rendering of quantifiers.

\[
\text{type}(Q) = C \rightarrow [(D \rightarrow S) \rightarrow S]
\]

**12. Compound Nouns**

Consider the following example.

9. **Amelia Earhart is a woman pilot**

The words ‘woman’ and ‘pilot’ are both common-nouns, and the phrase ‘woman pilot’ is a common-noun-phrase, so all these expressions have type $C$. The following is moreover a natural parse of ‘woman pilot’.

According to this analysis, two $C$-phrases combine to form another $C$-phrase. Notice that this does not conform to our current rule of composition, Function-Application, since neither expression is a function that takes the other as an argument. Rather, it seems we have simply slammed together two CNPs to make a bigger CNP.

In order to incorporate this sort of composition into our grammatical framework, we propose the following additional rule of composition.\footnote{This rule appears in Heim and Kratzer, *Semantics in Generative Grammar*, p. 65, where it is called *Predicate Modification*. Later, this rule will be seen to be a special case of a more fundamental and general rule.}

\[
\begin{array}{|c|c|}
\hline
\text{Conjunction} & \\
\hline
\text{a phrase of type} & C \\
\hline
\text{combines with a phrase of type} & C \\
\hline
\text{to produce a phrase of type} & C \\
\hline
\end{array}
\]

The following example sentence illustrates the new rule.
10. every woman pilot admires Amelia Earhart

\[ S \rightarrow [(D \rightarrow S) \rightarrow S] \]

\[ \Delta \]

\[ (D \rightarrow S) \rightarrow S \]

\[ \text{admires A.E.} \]

\[ \text{every} \]

\[ \text{woman} \]

\[ \text{pilot} \]

13. **Perils of Concatenation**

The semantics of concatenated nouns is fraught with peril. Consider the difference between the following.

11. Thomas Jefferson is a gentleman farmer
12. Thomas Jefferson is a potato farmer

Both of these have two readings, although not equally plausible.

Jefferson is a gentleman who farms [a farmer who is a gentleman].
Jefferson is a potato who farms [a farmer who is a potato].
Jefferson is a farmer who grows gentlemen.
Jefferson is a farmer who grows potatoes.

The following two examples are even crazier.

13. Danica Patrick is a woman racecar driver
14. Captain Ahab is a baby whale killer

Does Ms. Patrick drive woman racecars? Is she a woman-racecar who drives? What are woman racecars? On the other hand, is Ahab a baby that kills whales? A baby whale that kills things? A thing that kills baby whales?

To continue examples of perilous concatenation, consider the following examples.

15. trout fishing
16. fly fishing
17. ice fishing

If we insisted on rigorous role-marking, these might be clarified as follows.

15. for-trout fishing
16. with-fly fishing
17. on-ice fishing

Finally, consider the following examples.

18. Danish baker
19. dandruff shampoo
20. productivity software

The distinction between these various readings is taken up in a later chapter. For the moment at least, we concentrate the simplest readings, provided by Conjunction.

---

32 The captain of the whaling vessel *Pequod* in Melville’s *Moby Dick*. Note that he is never given a first name.

33 Note that a Danish is a kind of pastry, as well as a nationality. Still there is a role-ambiguity.

34 I submit that some software companies treat productivity software the same way that pharmaceutical companies treat dandruff shampoo.

35 +++cross-reference+++
14. Adjectives Reconsidered

Recall the earlier phrase.

21. brown dog

This analysis treats the adjective ‘brown’ as a C-modifier [type C→C], and treats the composition of ‘brown’ and ‘dog’ as an instance of Function-Application. But now we have an additional rule of composition – Conjunction. So the following would also be an acceptable composition.

Is it grammatically plausible to treat ‘brown’ as a common-noun-phrase?

It is fairly common for adjectives to be repurposed as common-nouns. For example, there is a 1960 song with the following lyrics.36

only the lonely
know the way I feel tonight;
only the lonely
know this feeling ain’t right.

The analysis of the words ‘only’, ‘know’, etc., is complicated. For the moment, we merely note that ‘lonely’ here means “lonely people”; in particular, ‘lonely’ is used as a common-noun, which combines with ‘the’ to form a noun-phrase (NP).

Another example: a few years ago,37 sports broadcasters started using the word ‘big’ as a common-noun, although in a limited manner, as in the following example.

22. the *bigs* need to step up tonight

When I first heard basketball announcers use ‘big’ like this, it sounded quite odd, but subsequent exposure has made it seem ordinary.38

By way of accounting for the fact that adjectives are sometimes used as common-nouns, the simplest hypothesis simply asserts that *some* adjectives behave like common-nouns. We say *some* because there are adjectives that clearly don’t behave this way. For example, I doubt anyone would proffer something like the following.

23. ? the *former* need to step up tonight.39
24. ? the *alleged* need to step up tonight.
25. ? the *imitation* need to step up tonight.

The difference between the latter adjectives and ‘big’ is primarily semantic, being summarized in the following distinction.

<table>
<thead>
<tr>
<th>Conjunctive Composition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A and B compose conjunctively</td>
</tr>
<tr>
<td>if and only if</td>
</tr>
<tr>
<td>being</td>
</tr>
</tbody>
</table>

36 Written by Roy Orbison and Joe Melson; sung by Roy Orbison; 1960.
37 2013 or so.
38 This use of ‘bigs’ might be analyzed as a variant pronunciation of ‘big ones’, which in some rural dialects might be pronounced “big-ənz”, which is similar to ‘young’un’s’.
39 Note that ‘the former’ is also used anaphorically. I own a cat and a dog; alas, the *former* hates the latter.
For example:

<table>
<thead>
<tr>
<th>being</th>
<th>is the same as being</th>
</tr>
</thead>
<tbody>
<tr>
<td>a brown dog</td>
<td>a dog <strong>and</strong> brown</td>
</tr>
<tr>
<td>a dog in the yard</td>
<td>a dog <strong>and</strong> in the yard</td>
</tr>
<tr>
<td>a dog that I own</td>
<td>a dog <strong>and</strong> (something) that I own</td>
</tr>
<tr>
<td>a woman racecar-driver</td>
<td>a woman <strong>and</strong> a racecar-driver</td>
</tr>
</tbody>
</table>

By contrast,

<table>
<thead>
<tr>
<th>being</th>
<th>is NOT the same as being</th>
</tr>
</thead>
<tbody>
<tr>
<td>a former player</td>
<td>a former <strong>and</strong> a player</td>
</tr>
<tr>
<td>an alleged criminal</td>
<td>an alleged <strong>and</strong> a criminal</td>
</tr>
</tbody>
</table>

By contrast,

<table>
<thead>
<tr>
<th>being</th>
<th>MIGHT be the same as being</th>
</tr>
</thead>
<tbody>
<tr>
<td>a potato farmer</td>
<td>a potato <strong>and</strong> a farmer</td>
</tr>
<tr>
<td>a racecar driver</td>
<td>a racecar <strong>and</strong> a driver</td>
</tr>
</tbody>
</table>

We do not propose a general account of non-conjunctive modification at the moment, and accordingly concentrate on conjunctive modification, and so concentrate on phrases that combine conjunctively.

For example, in an earlier example, we treat ‘brown’ as a C-modifier \([C\to C]\). We can now treat it as having type C, as combining with another C via conjunction. Our earlier example then gets redone as follows.

26. the brown dog sees the cat

\[
\begin{array}{c}
\text{S} \\
\text{D} \\
\text{C\to D} \\
\text{the} \\
\text{C} \\
\text{brown} \\
\text{D} \\
\text{C} \\
\text{dog} \\
\text{D\to (D\to S)} \\
\text{sees} \\
\text{D} \\
\text{C\to D} \\
\text{C} \\
\text{the} \\
\text{C} \\
\text{cat}
\end{array}
\]

15. **Copular-Be (is-of-predication)**

So far, our VPs have all involved transitive verbs, whereas the logically simplest examples of predication involve predicative-be, also called copular-be, also called the is-of-predication, which is illustrated in the following.

27. the brown dog **is** friendly

It seems that ‘friendly’ is formally like ‘brown’, which means we can treat it as having type C. What remains, then, is to categorize ‘is’ in the above sentence, which we propose to do as follows.

\[
\text{type(cop-be)} = C\to(D\to S)
\]

The present example then gets analyzed as follows.
16. **Transitive-Be (is-of-identity)**

In addition to the *is-of-predication*, logicians identify another form of ‘is’ (‘be’), which they call the *is-of-identity*, and which linguists call *transitive-be*, which behaves like a transitive verb, which accordingly is type-categorized as $D \rightarrow (D \rightarrow S)$.\(^{40}\) The following sentences illustrate:\(^{41}\)

28. the brown dog is Penny
29. Penny is the brown dog

which are categorially analyzed as follows.

Notice, in particular, that *is* is symmetrical here; both sentences are equally good. Contrast that with the following examples that employ *copular-be* [*is-of-predication*].

30. Penny is friendly
31. friendly is Penny

Whereas the first sentence sounds perfectly prosaic, the second sentence seems odd, or at best poetic.

\(^{40}\) In addition to the *is-of-identity*, and the *is-of-predication*, logicians also acknowledge the *is-of-existence*, which figures prominently in Hamlet’s famous line: *to be, or not to be; that is the question.*

\(^{41}\) Transitive verbs in English take a nominative argument (the subject) and an accusative argument (the object). On the other hand, transitive-*be* takes a complement that is known as a *predicate-nominative*, which accordingly some think should be in nominative case. If I answer the phone, and the person asks for Mister Hardegree, then I would tend to say ‘this is *he*’ [nominative case], which sounds better than ‘this is *him*’ [accusative]. On the other hand, if I see a photo of a person I recognize to be *me*, I would tend to say ‘that is *me*’ [accusative], not ‘that is *I*’. Notice in the example the phrase ‘recognize to be *me*’ [accusative]. What is going on here?
17. Indefinite Articles

In English, and many other languages, a common-noun-phrase may be prefixed by an indefinite article, the resulting phrase being an *indefinite noun phrase*. The following is a simple example from English, where ‘a’ serves as an indefinite article.

32. the brown dog is a collie

The following is a plausible (partial) categorial analysis.

\[
\begin{array}{c}
S \\
\text{D} \rightarrow S \\
\text{C} \\
\text{D} \rightarrow \text{C} \\
\text{C} \\
\text{C} \\
\end{array}
\]

![Diagram]

The article ‘a’ is not accounted for yet. But before we do that, we first note that, if we delete the word ‘a’, we obtain the phrase

\[
\text{the brown dog is collie}
\]

which standard English grammar treats as syntactically ill-formed. On the other hand, languages that lack indefinite articles – the biggest of which are Latin, Russian, and Mandarin – have no problem saying sentences like this. Furthermore, even English eschews indefinite articles when the common-nouns are plural-nouns or mass-nouns, as in the following examples.

33. the brown things are fish [fishes]
34. the brown stuff is fish [fish-matter]

So, how does ‘a’ operate in English. We propose treat ‘a’ is a number-word synonymous with ‘one’. In particular, ‘a’ behaves semantically as a non-conjunctive C-modifier (C→C). Its semantic role is to cut down the class of entities to singular-entities (individuals).

In any case, the following all have basically the same syntactic form.

<table>
<thead>
<tr>
<th>The thing in the bowl</th>
<th>is</th>
<th>a</th>
<th>fish</th>
<th>singular noun</th>
</tr>
</thead>
<tbody>
<tr>
<td>The things in the bowl</td>
<td>are</td>
<td>∅</td>
<td>fish</td>
<td>plural noun</td>
</tr>
<tr>
<td>The stuff in the bowl</td>
<td>is</td>
<td>∅</td>
<td>fish</td>
<td>mass noun</td>
</tr>
<tr>
<td>The stuff/thing in the bowl</td>
<td>is</td>
<td>∅</td>
<td>fishy</td>
<td>adjective</td>
</tr>
</tbody>
</table>

The following tree (partially) analyzes the first one.

---

42 Supposing we reject the reading according to which ‘collie’ is a proper-name, and the reading according to which ‘collie’ is a mass-noun [referring presumably to collie-matter].
43 For example in Russian, one can say ‘korichnevaya sobaka kolli’.
44 Note carefully, however, that colloquial spoken English often employs unstressed ‘some’ [“səm”] as an indefinite article, which can prefix all the nouns above. Also note that Spanish has plural indefinite articles – ‘unos’ (masculine), ‘unas’ (feminine), and French has a plural indefinite article ‘des’ and a mass indefinite article ‘de’.
45 See Chapter 9: Number Words. Indeed, English is fairly peculiar among Western European languages in that it has separate words for ‘a’ and ‘one’. Note that ‘one’ is also used as a common-noun, as in the following example.
46 In elementary logic, this is redundant, since ‘dog’ already means ‘dog-individual’. But it is not redundant if we think ‘dog’ includes in its extension dog-pluralities as well as dog-matter.
35. the thing in the bowl is a fish

\[
\begin{array}{c}
S \\
D \\
C \rightarrow D \\
\text{the} \\
C \\
? \\
C \rightarrow (D \rightarrow S) \\
\text{thing in the bowl} \\
C \\
\rightarrow C \\
C \\
\end{array}
\]

18. Adjectival Prepositions

Notice that the expression ‘in the bowl’ is not analyzed above. It is an example of a prepositional-phrase, and ‘in’ is an example of a preposition. As classified in dictionaries, prepositions include well over one hundred words and word-combinations, including the following.

- about, above, across, after, against, along, amid, among, around, at,
- according-to, ahead-of, across-from, along-with, away-from, as-of, as-for

The prepositions we are currently interested in behave as follows – each one combines with an NP (D) to form a conjunctive-adjective (C), and accordingly is type-categorized as follows.

\[
\text{type}(\text{Prep}) = D \rightarrow C
\]

The following is an example analysis.

36. the cat is on the mat

\[
\begin{array}{c}
S \\
D \\
C \rightarrow D \\
\text{the} \\
C \\
\rightarrow C \\
\text{cat} \\
D \rightarrow S \\
\text{is} \\
C \\
\rightarrow C \\
\text{on} \\
C \rightarrow D \\
\text{the} \\
C \\
\rightarrow C \\
\text{mat} \\
\end{array}
\]

The example from the previous section can now be completed as follows.

37. the thing in the bowl is a fish

\[
\begin{array}{c}
S \\
D \\
C \rightarrow D \\
\text{the} \\
C \\
\rightarrow C \\
\text{thing} \\
D \rightarrow S \\
\text{is} \\
C \\
\rightarrow C \\
\text{a} \\
C \rightarrow (D \rightarrow S) \\
\text{fish} \\
C \\
\rightarrow C \\
\text{in} \\
C \rightarrow D \\
\text{the} \\
C \\
\rightarrow C \\
\text{bowl} \\
\end{array}
\]

---

47 Propositional phrases also serve as adverbial modifiers, as in the following example.
- the dog swam in the river [modifies VP]
- an hourglass is narrow in the middle [modifies adjective]
- in Camelot, it never rains till after sundown [modifies Sentence]
The following involves two prepositions.

38. every squirrel in the yard is under the bird-feeder

\[
\begin{array}{c}
\text{S} \\
\text{C \rightarrow [(D \rightarrow S) \rightarrow S]} \\
\text{every} \\
\text{squirrel} \\
\text{in} \\
\text{the yard} \\
\text{is} \\
\text{under} \\
\text{the} \\
\text{bird-feeder}
\end{array}
\]

19. Relative-Pronouns and Relative-Clauses

Notice that the following are equivalent.

39. every squirrel in the yard is under the bird-feeder

40. every squirrel that is in the yard is under the bird-feeder

The phrase ‘that is in the yard’ is an example of a relative-clause. Every relative clause begins with a relative-pronoun. A relative-pronoun (aka wh-pronoun) is a function-word that begins with ‘wh’ or at least would if English were more uniformly spelled and/or pronounced. These include:

who, whom, whose, when, where, why, which, what, how

On the other hand, a relative-clause (wh-clause) is a phrase headed by a wh-pronoun, which behaves in many ways like a sentence [thus the terminology ‘clause’]; e.g., it has a subject, a verb, and perhaps an object or two. Relative clauses divide into four classes, as follows:

<table>
<thead>
<tr>
<th>interrogative clause</th>
<th>who is the greatest philosopher?</th>
</tr>
</thead>
<tbody>
<tr>
<td>complementary clause</td>
<td>I don’t know who the greatest philosopher is</td>
</tr>
<tr>
<td>restrictive relative clause</td>
<td>the greatest philosopher who studied with Frege is Carnap</td>
</tr>
<tr>
<td>non-restrictive relative clause</td>
<td>the greatest philosopher, who studied with Frege, is Carnap</td>
</tr>
</tbody>
</table>

In the latter two sentences, note carefully the punctuation. The following are similar examples.

| restrictive relative clause | there is exactly one philosopher-who-studied-with-Frege |
| non-restrictive relative clause | there is exactly one philosopher… who studied with Frege |

Here the punctuation has been exaggerated to make a point. In the first one, ‘philosopher-who-studied-with-Frege’ is a unit. In the second one, ‘there is exactly one philosopher’ is a unit. The first sentence does not say how many philosophers there are, but it does say that exactly one of them studied with Frege. The second one says there is exactly one philosopher, and it further claims that this philosopher studied with Frege.

---

48 Note, in particular, that there are two pronunciation groups [at least according to current dictionaries].

when, where, why, which, what ['h' (aspiration) followed by 'w']

who, whom, whose, how ['h' not followed by 'w']

Note also that ‘that’ also serves (only) as a restrictive-relative pronoun.

49 The word ‘how’ is deficient as a wh-pronoun – although it can be used to form an interrogative clause and a complementary clause, it cannot be used to form a relative clause, either restrictive or non-restrictive.

50 An interrogative clause is not merely like a sentence; it is a sentence.
20. Restrictive Relative Clauses

For the time being, we concentrate on restrictive-relative-clauses, as in the following example.

41. every woman who respects Kay is friendly

The phrase structure goes as follows, where known types are employed where possible.

\[
\begin{array}{c}
S \\
(D \rightarrow S) \rightarrow S \\
C \rightarrow [(D \rightarrow S) \rightarrow S] \\
C \\
\text{woman} \\
\text{who respects Kay}
\end{array}
\]

How do we type-render the ?-marked phrases? In the above example, the relative-clause combines with a C to form another C. Does it combine conjunctively? Are the following equivalent?

- \(x\) is a woman who respects Kay
- \(x\) is a woman and \(x\) is (someone) who respects Kay

Yes! Accordingly, we can treat the relative clause as having type \(C\), and accordingly, we can treat the relative pronoun as having type \((D \rightarrow S) \rightarrow C\). So we have the following sub-tree.

\[
\begin{array}{c}
C \\
C \\
\text{woman} \\
(D \rightarrow S) \rightarrow C \\
\text{who respects Kay}
\end{array}
\]

21. Syntactic-Categories versus Syntactic-Types

Although we have not officially discussed semantics in any detail, we have proposed many type-assignments based upon informal semantic considerations. This has led to numerous discrepancies between conventional classification and type-categorization, which illustrate the following general theme of this opus.

The syntactic-type of a phrase, which takes into account its semantic behavior, need not mimic the syntactic-category of that phrase.

The following are examples.
We have not discussed ‘my’ before, but it is useful to consider this word, since it provides an excellent further example of the discrepancy between syntax and semantics. As indicated above, ‘my’ is categorized as a determiner, just like ‘every’, ‘the’ and ‘a’, but notice that these four phrases are assigned four different types.\(^51\)

The discrepancy between syntax (category) and semantics (type) has numerous consequences, chief among which being that many strings are judged differently as to admissibility. For example, the following string

42. every my dog is friendly

is syntactically ill-formed.\(^53\) On the other hand, it is well-formed according to the following categorial analysis.

\[
\begin{align*}
S \quad &\quad \quad (D\rightarrow S)\rightarrow S \\
(C\rightarrow[(D\rightarrow S)\rightarrow S]) \quad &\quad \quad \text{is friendly} \\
\text{every} \quad &\quad \quad \text{C} \\
\text{my} \quad &\quad \quad \text{C} \\
\text{dog} \quad &\quad \quad \text{C} \\
\end{align*}
\]

According to this analysis ‘my dog’ is a common-noun-phrase, which accordingly refers to a class of individuals, which in this example are presumably dogs that belong to me. It is furthermore plausible to suppose that ‘dog’ refers to dogs, and ‘my’ refers to things that belong to me,\(^54\) so by semantic-conjunction, ‘my dog’ refers to dogs that belong to me.\(^55\)

The opposite situation also arises – some strings do not set off syntax alarms, but nevertheless make no sense, and are accordingly semantically ill-formed. Probably the most famous example of such a sentence is offered by Chomsky (1957).

---

51 Later [Chapter 11: Definite Descriptions] we reconsider in great detail how best to type-categorize ‘the’.
52 Ultimately, it is even more complicated, since ‘my dog’ means something quite different from ‘my mother’, as we later see when we discuss possessive versus genitive. Once again the syntax and semantics to not align.
53 This syntactic failure can be classified as a violation of the syntactic rule that prohibits double-determiners.
54 Keep in mind that this is the possessive use of ‘my’, not the genitive use. My mother is presumably not a mother that belongs to me. Oh my; it’s also not the exclamatory use of ‘my’.
55 Notice that this account of ‘my’ poses a problem when we analyze: my dog is friendly

We diagnose the problem as a special case of a more general problem of understanding how C-phrases can serve as subjects and objects of verbs. See Chapter 8 [Indefinite Noun Phrases].
43. colorless green ideas sleep furiously

This seems to be patent nonsense. On the other hand, it has a straightforward syntactic tree.\[56\]

Indeed, it seems that this can be computed to a semantic-value, which is always false, since ‘colorless’ and ‘green’ contradict each other.

The following example is much more subtle and interesting.\[57\]

44. more people have been to Paris than I have

We don't have the resources yet to demonstrate that this is syntactically OK, but semantically bad. Nevertheless, we trust that the reader does not immediately say “OUCH!”’, but only starts to feel uncomfortable when s/he tries to figure out what this sentence says (which turns out to be nothing)!\[58\]

A. Appendices

22. Appendix 1 – Summary of Basic Categorial Syntax

1. Basic Principles

(1) Phrases are the principal subject matter of grammar.
(2) Phrases are related to one another by a part-whole relation, which in particular is a tree-order.
(3) Molecular (compound) phrases have proper parts.
(4) Atomic (simple) phrases have no proper parts.
(5) Every phrase is assigned a type.
(6) Types include primitive-types and derivative-types.
   (a) primitive-type phrases are complete, and accordingly do not take input.
   (b) derivative-type phrases (functors) are incomplete, and accordingly take input.
(7) Functors are categorized according to what types of phrases they take as input and what types of phrases they produce as output.
(8) Phrases combine in accordance with Type-Governed Syntactic Rules, including:
   (a) categorial rules.
   (b) formatting rules.

2. Tree-Order

A partial-order (relation) is a relation \( \leq \) satisfying the following conditions.

(1) \( x \leq x \) [reflexive]
(2) if \( x \leq y \) and \( y \leq z \), then \( x \leq z \) [transitive]
(3) if \( x \leq y \) and \( y \leq x \), then \( x = y \) [anti-symmetric]

A tree-order (relation) is a partial-order satisfying the following further condition.

(4) if \( x \leq y \) and \( x \leq z \), then \( y \leq z \) or \( z \leq y \) [semi-linear]

Note carefully that the converse of a partial-order relation is also a partial-order relation, but the converse of a tree-order relation need not be a tree-order relation. So it is important to be clear what the relation \( \leq \) expresses. In the case of grammar, \( \alpha \leq \beta \) means \( \alpha \) is a part (sub-phrase) of \( \beta \).

56 We need to add a covert determiner to make the CNP into an NP.
57 The origin of this example is quite interesting; see Geoffrey K. Pullum's log (2004):
http://itre.cis.upenn.edu/~myl/languagelog/archives/000860.html
58 In particular, it does not compute to FALSE; rather, it computes to NIL.
3. **Left-Right Order**

We concentrate on trees when we think mathematically about grammar, but syntactic admissibility also involves proper left-right organization. For example, the following phrase passes muster according to purely categorial norms.

\[
\text{is collie a dog brown the}
\]

\[
\begin{array}{c}
\text{S} \\
\text{D} \rightarrow \text{S} \\
\text{C} \rightarrow (\text{D} \rightarrow \text{S}) \\
\text{is} \\
\text{C} \rightarrow \text{C} \\
\text{collie} \\
\text{a} \\
\text{dog} \\
\text{brown} \\
\text{the}
\end{array}
\]

The moral of this exercise is that syntactic admissibility involves some restrictions that are not purely categorial in nature. These are very important for syntax, but since we are principally concerned with semantics, we will largely ignore non-categorial restrictions.

4. **Official (Inductive) Definition of Types**

| (1) | S is a type. | sentences |
| (2) | D is a type. | definite noun phrases |
| (3) | C is a type. | common noun phrases |
| (4) | if A and B are types, then so is (A → B). | monadic functors |
| (5) | nothing else is a type. |

Definite-noun and common-noun are understood semantically: whereas the former denote particulars, the latter denote universals.

5. **Composition Rules**

1. **Function-Application**

   | a phrase of type | A → B |
   | combines with a phrase of type | A |
   | to produce a phrase of type | B |

2. **Conjunction**

   | a phrase of type | C |
   | combines with a phrase of type | C |
   | to produce a phrase of type | C |
## 23. Appendix 2 – Examples of Type Assignments

<table>
<thead>
<tr>
<th>Phrase</th>
<th>Category</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jay</td>
<td>proper-noun</td>
<td>D</td>
</tr>
<tr>
<td>Kay</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elle</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Penny</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lassie</td>
<td></td>
<td></td>
</tr>
<tr>
<td>respects</td>
<td>transitive-verb</td>
<td>D→(D→S)</td>
</tr>
<tr>
<td>admires</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sees</td>
<td></td>
<td></td>
</tr>
<tr>
<td>is [identity]</td>
<td>copula</td>
<td>C→(D→S)</td>
</tr>
<tr>
<td>dog</td>
<td>common-noun</td>
<td>C</td>
</tr>
<tr>
<td>cat</td>
<td></td>
<td></td>
</tr>
<tr>
<td>man</td>
<td></td>
<td></td>
</tr>
<tr>
<td>woman</td>
<td></td>
<td></td>
</tr>
<tr>
<td>house</td>
<td></td>
<td></td>
</tr>
<tr>
<td>barn</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mat</td>
<td></td>
<td></td>
</tr>
<tr>
<td>collie</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bowl</td>
<td></td>
<td></td>
</tr>
<tr>
<td>squirrel</td>
<td></td>
<td></td>
</tr>
<tr>
<td>yard</td>
<td></td>
<td></td>
</tr>
<tr>
<td>brown</td>
<td>adjective</td>
<td>C</td>
</tr>
<tr>
<td>friendly</td>
<td></td>
<td></td>
</tr>
<tr>
<td>in</td>
<td>preposition</td>
<td>D→C</td>
</tr>
<tr>
<td>on</td>
<td></td>
<td></td>
</tr>
<tr>
<td>under</td>
<td></td>
<td></td>
</tr>
<tr>
<td>every</td>
<td></td>
<td></td>
</tr>
<tr>
<td>some</td>
<td></td>
<td></td>
</tr>
<tr>
<td>no</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>determiner</td>
<td>C→C</td>
</tr>
<tr>
<td>the</td>
<td></td>
<td>D→C</td>
</tr>
<tr>
<td>my</td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>and</td>
<td>connective</td>
<td>S→(S→S)</td>
</tr>
<tr>
<td>who</td>
<td>relative pronoun</td>
<td>(D→S)→C</td>
</tr>
<tr>
<td>that</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
24. Appendix 3 – Grammatical Niceties

1. Words and Phrases

Grammatical terms such as ‘word’, ‘phrase’, ‘sentence’ are not as clear-cut as one might wish. Fortunately, from the viewpoint of semantics, which is our principal concern, we can simplify our terminology.

Semantics hypothesizes that all linguistic *items* can be analyzed into *fundamental items*, called *morphemes*. A *string* then is any linear combination of morphemes, and a *phrase* is a *semantically-admissible* string.

Between morphemes and phrases are words, and the linguistic study of how morphemes combine into words is known as *morphology*. An excellent illustration of the difference between a word and a phrase is the difference between ‘darkroom’ and ‘dark room’. Notice in particular that the two expressions are pronounced differently – emphasis on the first syllable, or the second syllable.

On the other hand, *word* is not a semantic notion, but is rather a purely syntactic notion. There are many ways in which natural languages divide phrases into words. Some languages (e.g., those with *clitics*) have words that other languages would regard as phrases, and some languages (the polysynthetic ones) have words that other languages would regard as sentences.

The key point is that whether a symbol-combination counts syntactically as a word or not is irrelevant to semantic composition, which constructs the meaning of a phrase out of the meanings of its constituent morphemes, whose meanings are given by the lexicon.

2. X-Like Phrases versus X-Headed Phrases

The term ‘phrase’ is often appended to a grammatical term, such as ‘noun’, to produce a new term, such as ‘noun phrase’. There are two ways to understand this compounding technique, given as follows, where ‘X’ is a category-word such as ‘noun’ or ‘verb’.

(1) an X-phrase is an X-**headed** phrase.
(2) an X-phrase is an X-**like** phrase – i.e., a phrase that behaves like an X.

By way of illustration, the phrase ‘my brown dog’ behaves like the proper-noun ‘Penny’. Semantically speaking, they both refer to a particular individual, in this case a particular dog. Syntactically speaking, ‘my brown dog’ can be properly used anywhere ‘Penny’ can. Accordingly, ‘my brown dog’ is a proper-noun-like phrase.

The notion of head in syntax is theoretically subtle, and depends upon the details of one’s syntactic theory. In categorial grammar, at least, it is natural to regard the functor in a functor-argument compound as the head of the compound. For example, ‘my’ is the head of ‘my brown dog’. Accordingly, since ‘my’ is a determiner ‘my brown dog’ is a determiner-headed phrase, which is customarily called a **determiner phrase** (DP).

The phrase ‘brown dog’ in turn behaves like the common-noun ‘dog’, so it is a common-noun-like phrase. On the other hand, its head is ‘brown’, which is an adjective, so it is an adjective-headed phrase.

As used in most syntactic theories, the term ‘verb phrase’ refers to verb-headed phrases, not to verb-like phrases. The latter include phrasal verbs – such as ‘clean up’ – and count as morphemes, appearing as separate entries in modern dictionaries. Similarly, the term ‘prepositional phrase’ refers to preposition-headed phrases – such as ‘by the house’ – which behave like adjectives and adverbs, and are accordingly adjective-like and adverb-like phrases. There are also preposition-like phrases, or

59 We also note that some morphemes look like phrases, including *phrasal-verbs* – for example, ‘clean up’, which has three entries in the *American Heritage Dictionary* [2015].

60 Pinker, *Words and Rules*.

61 Clitics blur the word-phrase boundary. The best example in English is *apostrophe-s*, which behaves phonetically like a word-suffix, but behaves semantically like phrase-suffix as in:

\[
\text{the Queen of England's husband}
\]

where *apostrophe-s* modifies the phrase ‘Queen of England’, not the word ‘England’.

62 In Linguistics, *analytic-synthetic* is a spectrum along which languages can be categorized, according to how many morphemes typically appear in words. The fewer morphemes per word, the more analytic; the more morphemes per word, the more synthetic.
phrasal-prepositions, which include ‘next to’ and ‘in front of’, which are lexical items. Finally, a noun-phrase is a noun-like phrase – more specifically, a proper-noun-like phrase.

25. Appendix 4 – Rank and Order

There are two measures of type-complexity that are worth considering. The first one, called rank, measures how “deep” a type is. It is inductively defined as follows.

\[
\begin{array}{c|c}
\text{rank}(D) &= 0 \\
\text{rank}(C) &= 0 \\
\text{rank}(S) &= 0 \\
\text{rank}(X \rightarrow Y) &= 1 + \max \{ \text{rank}(X), \text{rank}(Y) \}
\end{array}
\]

Examples:
\[
\begin{align*}
\text{rank}\{ D \rightarrow S \} &= 1 + \max \{ \text{rank}(D), \text{rank}(S) \} = 1 + \max \{ 0, 0 \} = 1 \\
\text{rank}\{ D \rightarrow (D \rightarrow S) \} &= 1 + \max \{ \text{rank}(D), \text{rank}(D \rightarrow S) \} = 1 + \max \{ 0, 1 \} = 1 + 1 = 2 \\
\text{rank}\{ (D \rightarrow S) \rightarrow S \} &= 1 + \max \{ \text{rank}(D \rightarrow S), \text{rank}(S) \} = 1 + \max \{ 1, 1 \} = 2
\end{align*}
\]

Rank is a decent first-approximation as a measure of complexity, but it has its limitations. For this reason, it is helpful to formulate another measure of complexity, which corresponds to a well-known notion in logic – the notion of order (as in first-order, second-order, etc.) – which we officially define as follows.

\[
\begin{array}{c|c}
\text{order}(D) &= 0 \\
\text{order}(C) &= 0 \\
\text{order}(S) &= 0 \\
\text{order}(X \rightarrow Y) &= \max \{ 1 + \text{order}(X), \text{order}(Y) \}
\end{array}
\]

Examples:
\[
\begin{align*}
\text{order}\{ D \rightarrow S \} &= \max \{ 1 + \max \{ \text{order}(D), \text{order}(S) \} \} \\
&= \max \{ 1 + \max \{ 0 \}, 0 \} \\
&= \max \{ 1, 0 \} \\
&= 1 \\
\text{order}\{ D \rightarrow (D \rightarrow S) \} &= \max \{ 1 + \max \{ \text{order}(D), \text{order}(D \rightarrow S) \} \} \\
&= \max \{ 1 + \max \{ 0 \}, 1 \} \\
&= \max \{ 1, 1 \} \\
&= 1 \\
\text{order}\{ (D \rightarrow S) \rightarrow S \} &= \max \{ 1 + \max \{ \text{order}(D \rightarrow S), \text{order}(S) \} \} \\
&= \max \{ 1 + \max \{ 1 \}, 0 \} \\
&= \max \{ 2, 0 \} \\
&= 2 \\
\text{order}\{ ((D \rightarrow S) \rightarrow S) \rightarrow S \} &= \max \{ 1 + \max \{ \text{order}((D \rightarrow S) \rightarrow S), \text{order}(S) \} \} \\
&= \max \{ 1 + \max \{ 2 \}, 0 \} \\
&= \max \{ 3, 0 \} \\
&= 3
\end{align*}
\]
26. **Appendix 5 – Comparison with Montague's Typology**

Richard Montague's grammatical typology is widely used in semantics, so it is useful to compare it to ours.

Montague proposes the following typology.

1. **Primitive Types**
   
   (1) \( e \) entity
   
   (2) \( t \) truth-value

2. **Types**
   
   (1) every primitive type is a type.
   
   (2) if \( a \) and \( b \) are types, then \( \langle a, b \rangle \) is a type.
   
   (3) nothing else is a type.

Entity is the broadest class of particulars. Truth-value has only two instances – 1 (True), 0 (False).

Note that these are semantic-types, not syntactic-types, although there is a natural correspondence between Montague-types and our types, illustrated as follows.

<table>
<thead>
<tr>
<th>Montague-Type</th>
<th>Syntactic-type</th>
</tr>
</thead>
<tbody>
<tr>
<td>( e )</td>
<td>D</td>
</tr>
<tr>
<td>( t )</td>
<td>S</td>
</tr>
<tr>
<td>( \langle e, t \rangle )</td>
<td>D→S</td>
</tr>
<tr>
<td>( \langle e, \langle e, t \rangle \rangle )</td>
<td>D→(D→S)</td>
</tr>
<tr>
<td>( \langle e, t \rangle )</td>
<td>(D→S)→S</td>
</tr>
</tbody>
</table>

We use upper case letters in accordance with the usual category-naming practice in syntax, and we use arrow in accordance with mathematical function notation. We do not use the letters ‘E’ and ‘T’, since ‘E’ has no obvious syntactic category, and since we use ‘T’ for the truth-value True.

---