Lecture Notes for 6.824, Artificial Intelligence
 Rev. September, 1994

Natural Language Syntax and Semantics

The treatment of natural language syntax and semantics presented here is based on an approach known as categorial grammar. The basic idea is that the syntactic categories of English and other natural languages are analogous to types in programming languages and typed logics. For example, an adjective, like "yellow" modifies a noun phrase like "bird" to give a new noun phrase "yellow bird". This can be treated formally by giving the word "bird" the type N and the word "yellow" the type $\mathrm{N} \rightarrow \mathrm{N}$. The phrase "yellow bird" is then a well typed application of the adjectival operator "yellow" to the argument "bird". The phrase "bird yellow" is not well typed under this scheme and is not a grammatical noun phrase of English. Treating syntactic categories as types and assigning types to the individual words works surprisingly well as a theory of grammaticality in English. Although the basic categorial concepts do not account for all grammatical phenomenon, they do provide an integrated approach to syntax and semantics and clearly demonstrate the intimate relationship between syntactic categories and semantic types. The categorial grammar formalism can be used to essentially derive the basic syntactic categories of natural language from simple semantic considerations.

In this section we present a very simple version of categorial grammar. This simple treatment only scratches the surface of the rich theories of natural language syntax that have evolved over the past thirty years. Pointers to more complete treatments are given at the end.

1 Categorial Grammar

There is a strong analogy between the notion of grammatical sentences in natural languages such as English and the notion of well typed expressions in computer programs. Consider a function SQUARE defined in some programming language so that SQUARE takes a floating point number as an argument and returns the square of that argument. An application SQUARE(e) is well typed provided that $e$is an expression that returns a floating point number. The expression SQUARE(3.2) is well typed but SQUARE("foo") is not. The fact that SQUARE takes and returns a floating point number can be represented by assigning the function SQUARE the type $F \rightarrow F$ where F represents the type of floating point numbers. In typed programming languages an application $f(t_1 \;\cdots\; t_n)$ is well typed if f has type $\tau_1 \times \cdots \times \tau_n \rightarrow \sigma$ and each argument t_i has type τ_i.

In English, the sentence "The yellow bird sang" is grammatical (well typed), while "yellow the sang bird" is not. In making an analogy with the types used in programming languages we parse this sentence as ((the (yellow bird)) sang) where the word "bird" has type N, the word "yellow" has type $N \rightarrow N$, the word "the" has type $N \rightarrow NP$, and the word "sang" as type $NP \rightarrow S$. Under this assignment of types, and under the assumption that operator "sang" follows its argument rather than precedes it, the sentence ((The (yellow bird)) sang) is an expression of type S.

When discussing natural language we will use the term "category" rather than the term "type". English, and every other natural language, has syntactic categories — each phrase in English is either a noun phrase, a verb phrase, an adjectival phrase, or some other standard category. The categories of English are analogous to the types of programming languages. We use the symbol S to represent the category of a sentence, the symbol N to represent the category of an indefinite noun phrase noun such as "woman", the symbol NP to represent the category of a complete noun phrase such as "every woman" or "Mary", and the symbol VP to represent the category of a verb phrase such as "runs" or "kissed John". An English sentence is an expression of the form (NP VP), such as ((Every woman) (kissed (some man))). Table 1 gives a list of English categories. Each category is named by a symbol, although many of these symbols are just abbreviations for arrow categories. In the case where a symbol is an abbreviation for an arrow category, the arrow category is shown in a separate column.

Unlike the functions of most programming languages, an English word can

Category Name	Example Phrases	Arrow Category
S	((the (yellow bird)) flies)	category symbol
N	bird, man, (yellow bird)	category symbol
NP	Mary, (some bird), (every bird)	category symbol
VP	runs, (likes Tom), (flies quickly)	$S \leftarrow NP$
Adj	big, yellow	$N \rightarrow N$
AP	(on (the table))	$N \leftarrow N$
Adv	(into (the house))	$VP \leftarrow VP$

Table 1: Some fundamental categories of English

appear either before or after its arguments. To distinguish words which precede their arguments from words which follow their arguments we allow the arrow to be written in either direction. An operator that appears before its arguments will have a category of the form $\tau_1 \times \cdots \times \tau_n \rightarrow \sigma$ while an operator that appears after its arguments will have category $\sigma \leftarrow \tau_1 \times \cdots \tau_n$. For example, consider the phrase (big (yellow bird)). In this phrase, "bird" has category N and the adjectives "big" and "yellow" each have category $N \rightarrow N$. So the overall phrase has category N. Now consider the sentences "The yellow bird ate the seed". This sentence can be parsed as a noun phrase followed by a verb phrase ((The yellow bird) (ate the seed)). In this case the verb phrase is viewed as an operator that is applied to the noun phrase. Verb phrases have category $S \leftarrow NP$.[1]

An initial theory of English grammaticality can be constructed by hypothesizing that every word of English is associated with a category expression that is either one of the basic category symbols, or a left or right arrow expression built up form other category expressions.

[1] It is also possible to give a categorial grammar in which VP is a category symbol and NP is the category $VP \rightarrow S$. However, treating VP as $S \leftarrow NP$ allows for quantifier scope ambiguity. There are also other examples of operators that follow rather than proceed their arguments. For example, consider the adjectival phrase "on the roof" in "the man on the roof".

> **Definition:** A *lexical entry* is a pair of a word and a category expression.

Intuitively, a lexical entry is an entry in the "internal dictionary" of a native English speaker. It contains the category of the word, e.g., that it is a noun. Real dictionaries, the ones on bookshelves, also contain descriptions of the meanings of words. Word meanings are discussed in section 2. Linguists call the internal dictionary hypothesized to exist for a particular English speaker the *lexicon* of that speaker.

> **Definition** A *lexicon* is a set of lexical entries.

It is important to note that a given lexicon can contain more than one lexical entry for the same word. For example, the word "house" can be used either as a noun or as an adjective as in the phrase "house boat". This can be handled by having two lexical entries for the word "house".

We can now define the set of word strings that are grammatical relative to a given lexicon. This definition is similar to the definition of the well-typed expressions in programming languages except that, in natural languages, operators are allowed to follow their arguments and individual words may be ambiguous in the sense that they have more than one lexical entry in the lexicon. The following definition formally defines a "can be parsed as" relation between word strings and categories. If W is a word string and α is a category, then we say W can be parsed as α if there exists a parenthesization of the words in W, and an assignment of types to words in accordance with the lexicon, such that the resulting parenthesized expression is well-typed and has type α.

> **Definition** The "can be parsed as" relation between word strings and category expressions is the least relation satisfying the following conditions.
>
> - If there exists a lexical entry for a word w with category expression α, then w can be parsed as α.

- If the word string W can be written as the concatenation of substrings $U; S_1; \cdots; S_n$ where U can be parsed as $\alpha_1 \times \cdots \times \alpha_n \rightarrow \beta$, and each S_i can be parsed as α_i, then W can be parsed as β.
- If the word string W can be written as the concatenation of substrings $S_1; \cdots; S_n; U$ where U can be parsed as $\beta \leftarrow \alpha_1 \times \cdots \times \alpha_n$, and each S_i can be parsed as α_i, then W can be parsed as β.

Consider the sentence "Every yellow bird flies quickly". This sentence can be parsed relative to the following assignment of category expressions to words.

- every: $N \rightarrow NP$
- yellow: $N \rightarrow N$
- bird: N
- flies: VP
- quickly: $VP \leftarrow VP$

The parenthesization resulting from the parse is

((Every (yellow bird)) (flies quickly)).

When all category expressions in the lexicon involve only right arrows (or only left arrows), and if every word has only a single lexical entry, then a given word string has at most one parse. However, even if every word has only a single lexical entry, the presence of both left and right arrow categories allow a single word string to have more than one parse. For example, consider the phrase "the old grey man's house". The possessive noun "man's" can be treated as an abbreviation for the word man followed by the "word" poss (for possessive). We treat poss as a word of category $A \leftarrow N$, i.e., it takes a noun phrase argument, which it follows, and produces a phrase of category A, i.e., of category $N \rightarrow N$. The word "the" has category $N \rightarrow NP$, the words "old" and "grey" have category A, and the words "man" and "house" have category N. The word string "the old grey man poss house" has three different parses corresponding to the three following partial parenthesizations.

the old grey ([man poss] house)

the old ([(grey man) poss] house)

the ([(old grey man) poss] house)

Another example of an ambiguous phrase is the sentence "john saw the man with the telescope". In this sentence the phrase "with the telescope" can either specify how john saw the man (with the telescope) or it can specify which man John saw (the man with the telescope). In this case the ambiguity can be attributed to two different lexical entries for the preposition "with".

- with: $NP \rightarrow Adv$
- with: $NP \rightarrow AP$

The first lexical entry specifies that "with" constructs an adverbial phrase, while the second lexical entry specifies that "with" constructs an adjectival phrase. Table 1 lists Adv is the category expression $VP \leftarrow VP$ and lists AP as the category expression $N \leftarrow N$.

The different parses of the above phrases seem to correspond to different semantic interpretations. It is possible to give a theory of the semantics of English sentences that is similar to the semantics of the expressions in typed programming languages.

2 Evaluation Semantics

There are two quite different uses for the term "semantics" which we will call "semantics by evaluation" and "semantics by translation". We first discuss semantics by evaluation and consider semantics by translation briefly in section 3. In semantics by evaluation one gives some way of determining the value of expressions. For example, in a programming language the value of `SQUARE`(2.1) is 4.31. In a programming language the value of an application $f(e_1, \ldots, e_n)$ is computed by computing the values of $e_1, \ldots, e_n$ and then applying the function f to these values. In semantics by evaluation it is important that an operator f of type $\tau \rightarrow \sigma$ be associated with a (semantic) function which maps each semantic value of type τ to a semantic value of type σ. Semantics by evaluation is often called *denotational* semantics and the value of an expression is often called its *denotation*.

For expressions in programming languages it is possible to actually compute the value of expressions. This can not be done for phrases of natural language. For example, we can not compute the referent of "the man on the roof". One obvious problem is that the referent is a man and men do not live inside computers. However, one can gain insight into the structure of natural language by *assuming* that each category of English is associated with a set, called the semantic domain of the category, and that when an English word is associated with an arrow category $\alpha \rightarrow \beta$ it is also associated with a particular semantic function which maps any semantic value in the domain of α to some semantic value in the domain of β. We will call this the "assumption of compositional semantics". The assumption of compositional semantics for natural language is an idealization — a completely accurate theory of English semantics would be more elaborate. The simple compositional theory is analogous to an ideal gas or a frictionless plain — it is a useful simplification of reality.[2]

Although semantic values of natural language expressions are not computable,

[2]The phrase "compositional" is used to signify that the value of an expression is determined by the values of its parts. In a compositional language if e_1 and e_2 have the same value then $f(e_1)$ and $f(e_2)$ must have the same value. Quantified expressions, such as universally quantified sentences or λ-expressions, are not compositional. For example, consider the logical sentence $\forall x\ x = x$ which states that for any x, x must equal itself. Suppose that x has the value 3. Then $x = x$ and $x = 3$ both have value $\mathbf{T}$. But $\forall x\ x = x$ does not have the same value as $\forall x\ x = 3$. This shows that the meaning of $\forall x\ \Phi[x]$ is not determined by the meaning of the three parts consisting of the universal quantifier $\forall$, the variable x, and the formula $\Phi[x]$. Using combinators it is possible to give a completely compositional treatment of first order logic or the λ-calculus. However, such compositional treatments seem unnatural to many people and are not widely used.

Category Name	Example Phrases	Arrow Category	Type Expression
S	((the bird) flies)	category symbol	B
N	man, (yellow bird)	category symbol	$D \rightarrow B$
NP	Mary, (every bird)	category symbol	$(D \rightarrow B) \rightarrow B$
VP	runs, (likes Tom)	$S \leftarrow NP$	
Adj	big, yellow	$N \rightarrow N$	
AP	(on (the table))	$N \leftarrow N$	
Adv	(into (the house))	$VP \leftarrow VP$	

Table 2: An association of type expressions with category symbols.

the assumption of compositional semantics accurately predicts many features of natural language syntax. To see this assume there is a semantic domain D corresponding to the "things" in the world and a domain B consisting of two values **T** and **F**. The domain B is motivated by the fact that natural language is used to express statements and statements seem to have have truth values — we often say that a given statement is true or false. In the remainder of this section we will use the term "category" to refer to the categories of natural language and the word "type" to refer to type expressions built from the base types D and B and the type constructor $\rightarrow$. Now if English has a compositional semantics then we would expect the syntactic categories of English to correspond in a natural way to type expressions constructed from D and B. Table 2 is a copy of table 1 plus an extra column. The extra column associates each of the basic category symbols with a type expression constructed from D and B. Given an interpretation of each category symbol as a type expression, any category expression can be converted to a type expression by replacing each category symbol by the corresponding type expression and replacing each left arrow by a right arrow (type expressions only have right arrows).

Although each category expression maps to a well defined type expression, it seems that category expressions, and not type expressions, determine grammaticality. For example, the direction of the arrow in a category expression is clearly important in determining grammaticality. The use of category expressions rather than type expressions in determining grammaticality is an example of Chomsky's general principle of the autonomy of syntax — syntactic grammaticality is, to some extent, independent of semantics. However, the existence of a simple mapping between the syntactic categories of English and small type expressions constructed from D and B shows that in some sense the structure of the syntactic categories can be predicted form the compositional semantics assumption and the fact that language can express statements about objects. Given that syntactic categories can be largely derived from these basic principles it is not surprising that all human languages have these basic syntactic categories.

In addition to providing insight into the structure of syntactic categories, the assumption of compositional semantics can be used to accurately predict truth values that people will assign to certain sentences. Consider the sentence ((Some bird) (ate (every seed)). This sentence can be assigned an intuitively correct semantic value by assigning an intuitively plausible value to each word and then constructing the value of the sentence by application. The word "bird" has category N which is associated with the type expression $D \rightarrow B$. Under the compositional semantics assumption the word "bird" must denote a function that maps objects to truth values. We can assume that the word "bird" denotes the function that is true of an object just in case that object is a bird. Now consider the phrase (some bird). To assign a meaning to this phrase one must assign a meaning to the word "some". The word "some" has category $N \rightarrow NP$. This maps to the type expression

$$((D \rightarrow B) \rightarrow ((D \rightarrow B) \rightarrow B))$$

This type expression can be written more clearly by introducing P (for predicate) as an abbreviation for the type expression $D \rightarrow B$. The type of "some" can now be written as

$$P \rightarrow (P \rightarrow B).$$

We take the meaning of "some" to be the operator s such that $[s(Q)](R)$ is $\mathbf{T}$ provided there is some x such that both Q and R are true of x. In other words, $[s(Q)](R)$ is $\mathbf{T}$ just in case the set on which Q is true intersects the set on which R is true. Given this semantic value for "some", the phrase (some bird) denotes an operator which maps a predicate P to $\mathbf{T}$ just in case P is true of some bird. Similarly, the semantic value of "every" is the operator a

such that $[a(Q)](R)$ is $\mathbf{T}$ if the set on which Q is true is a subset of the set on which R is true. Given this semantic value for "every", the phrase (every bird) denotes an operator that takes a predicate and returns true just in case that predicate is true of every bird.

I will use the symbol Q as an abbreviation for the type expression associated with the category symbol NP, i.e., Q is an abbreviation for the type expression $P \rightarrow B$ (where P is $D \rightarrow B$). Objects of type Q will be called *quantifiers* (hence the symbol Q). The word "ate" has category $NP \rightarrow VP$ where VP is an abbreviation for $S \leftarrow NP$. This give the word "ate" the semantic type $Q \rightarrow (Q \rightarrow B)$. To define the semantic value of the word "ate" some additional notation is needed. We use the expression $\lambda x\ B[x]$ to denote the semantic function which maps x to $B[x]$. For example, $\lambda x\ x + x$ is the function which maps a number to twice that number. Let E be the semantic predicate of type $D \times D \rightarrow B$ such that $E(x,\ y)$ is $\mathbf{T}$ if x ate y. We can take the meaning of the word "ate" to be the operator f of type $Q \rightarrow (Q \rightarrow B)$ satisfying the equation

$$[f(A)](B) = B(\lambda x\ A(\lambda y\ E(x,\ y)).$$

Now we can derive the compositional meaning of the sentence ((some bird) (ate (every seed))) using the meanings we have just assigned to "some", "bird", "ate", "every", and "seed". For any expression e I will use $V[e]$ to denote the semantic value of e. Note that $V[\text{(some bird)}]$ and $V[\text{(every seed)}]$ are both quantifiers, i.e., operators of type $P \rightarrow B$. The meaning of the expression (ate (every seed)) can be written as $f(V[\text{(every seed)}])$ where f is the function associated with "ate". We have that $f(V[\text{(every seed)}])$ is an operator of type $Q \rightarrow B$. To derive the meaning of the whole sentence we apply the operator $f(V[\text{(every seed)}])$ to $V[\text{(some bird)}]$. So the overall meaning is

$$[f(V[\text{(every seed)}])](V[\text{(some bird)}]).$$

By the definition of f this is equivalent to

$$V[\text{(some bird)}](\lambda x\ V[\text{(every seed)}](\lambda y\ E(x,\ y))).$$

This is true just in case some bird (the same bird) ate every seed.

There is another reading of ((some bird) (ate (every seed))) in which every seed was eaten but different seeds might have been eaten by different birds. This alternate meaning can be formally handled by providing a second lexical entry for ate. Rather than use the above equation for f we can take the meaning of "ate" to be the operator f satisfying the equation

$$[f(A)](B) = A(\lambda y\ B(\lambda x\ E(x,\ y)).$$

Then the meaning of ((some bird) (ate (every seed))) becomes

$$V[(\text{every seed})](\lambda y\ V[(\text{some bird})](\lambda x\ E(x,\ y))).$$

3 Semantics by Translation

Although the compositional semantics assumption in some sense predicts the structure of syntactic categories, and predicts the truth values people assign to certain sentences, it does not provide an account of the computational processes involved in human natural language understanding. People understand phrases even when they can not compute their values. For example, consider the phrase "the first prime number greater than one million". Most people would agree that they know what this phrase means, and that it denotes a unique prime number, even though they do not know what that number is. One very simple model of the computational processing can be derived from the notion of logical inference — English sentences are translated into expressions in an internal language called "mentalese" and inference mechanisms are used to derive consequences from beliefs and premises all of which are expressed as statements in mentalese.

A translation from English to mentalese can be viewed as an alternative to compositional semantics. Once the translation has been specified, and the computational inference processes on mentalese have also been specified, then it seems that one should not need any additional theory of compositional semantics. Various proposals have been given for translations to mentalese, and for processing of mentalese, but none will be explicitly discussed here.

Although translation semantics seems to have the potential of providing more insight into computational processing, it does not have the elegant predictive power of compositional semantics. Translation semantics does not predict the syntactic categories of English nor the truth values that people assign to sentences. Translation semantics requires a theory of the structure of mentalese before such predictions can be made.

It seems possible that the syntax and semantics of mentalese is actually very similar to the syntax and semantics of spoken natural language. It is difficult to see why any elaborate divergence between the external and internal representations would be useful. If mentalese is essentially a parsed version of spoken language then the syntax of natural language provides direct insight into the syntax of mentalese. Furthermore, the compositional semantics of natural language may also serve as a compositional semantics for mentalese. An understanding of a compositional semantics for mentalese would provide a predictor of (some of) the structure of the inference mechanisms underling reasoning. For example, given two inference rules one of which is semantically sound and the other of which is not, the sound rule should be more likely to be used in reasoning than the unsound rule.

4 Syntactic Complications

Empirically it seems that categorial grammar and compositional semantics do not provide a full account of the syntax and semantics of natural language. One thing that seems unaccounted for is "number agreement". The verb of a sentence must agree with the main noun phrase, if the noun phrase is plural then the verb must be in plural form and if the noun phrase is singular the verb phrase must agree. For example consider the following sentences. Sentences that people judge to be ungrammatical are labeled with $\star$.

The bird is pretty.

$\star$ The bird are pretty.

The birds are pretty.

$\star$ The birds is pretty.

Another example of noun-verb agreement is related to whether the noun is first, second, or third person. For example, consider the following sentences.

I want to go fishing.

John wants to go fishing.

$\star$ I wants to go fishing.

$\star$ John want to go fishing.

Another thing unaccounted for in categorial grammar is case. In many languages, such as Latin, word order is not very important — most operators are free to appear before, after, or between their arguments. For example, a transitive verb can be viewed as an operator that takes two noun phrases as arguments. In Latin, the arguments can appear in either order, and the verb can appear before, after, or between the arguments. Fortunately, one of the argument noun phrases must be in nominative case and the other in the accusative case. Case can be used to determine how the operator is to be applied to the arguments.

A third aspect of grammaticality not accounted for by categorial grammar concerns "traces". A trace is a theoretical device used to explain missing arguments. For example, consider the sentence

Mary liked the picture John drew.

The phrase "John drew" acts as an adjectival phrase modifying "picture". However, it seems unreasonable to conjecture lexical entries for "John" and "drew" such that "John drew" can be parsed as AP. It seems more reasonable to assume that the phrase "the picture John drew" has a hidden variable, called a trace, that is the argument of "drew". This phrase can be treated as phonological variant of (the (picture (λx (John (drew x))))) where the λ-predicate (λx (John (drew x))) is an adjectival phrase. The variable x that serves as an argument to "drew" is called a trace because it does not appear in the spoken version of the sentence. Traces account for a wide variety of sentences. For example, consider the following.

Mary liked the picture John gave to Sally.

Mary liked the picture John said was expensive.

Mary liked the picture John saw Sally give to Bill.

There is even phonological evidence that traces are actually present in the phrase structures people assign to sentences such as the above. Consider the following two phrases.

The person I want to see

The person I want to go away

Both of these sentences involve traces. They can be taken to be phonological variants of the following expressions.

The person (λx I want to see x)

The person (λx I want x to go away)

The pair of words "want to" are often pronounced as "wanna". However, it seems that if a trace appears between "want" and "to" this contraction is prevented.

The person I wanna see

$\star$ The person I wanna go to the store

Clearly, the simple version of categorial grammar given above must be modified to account for all of the syntactic features found in human languages.

5 Semantic Complications

In section 2 it was assumed that the semantic value of a sentence was a truth value. This idealized semantical picture must be modified to handle sentences such as the following.

Category Name	Example Phrases	Type Expression
S	(Mary runs)	$Sit \rightarrow B$
N	(yellow bird)	$D \rightarrow (Sit \rightarrow B)$
NP	Mary, (some bird)	$(D \rightarrow (Sit \rightarrow B)) \rightarrow (Sit \rightarrow B)$

Table 3: Situations in the type expressions.

Mary believes John stole the painting.

This sentence can be parsed provided that "believe" is assigned the category $S \rightarrow (S \leftarrow NP)$. In other words, "believe" denotes an operator that takes a sentence and a noun phrase. The difficulty is semantic rather than syntactic. We have assumed that the semantic value of any sentence is a truth value. This implies that "believe" is an operator that takes a truth value and a noun phrase. The only plausible semantic interpretation of "believe" is such that a sentence of the form "Mary believes Φ" is true if and only if Φ is true. This is clearly counter to common sense.

One way of overcoming this difficulty is to introduce "situations" as an additional semantic domain. Intuitively, a situation is a time and place in some actual or possible world. For example, 12 o'clock noon on May 3, 1989, at table four of the Cornell faculty club. Situations can also be "imaginary" or "possible but not actual". Santa's work shop on Christmas eve is a situation.

Rather than say that sentences denote truth values, we can say that sentences denote functions from situations to truth values. For example, if Tweety is a particular bird, then the sentence "Tweety can fly" is true at some times but false at times when Tweety's wings have just been clipped. To incorporate situations into compositional semantics we introduce a new type symbol Sit which denotes the set of all possible situations (actual, possible, and imaginary). We retain the two previous type symbols D and B, but we change the assignment of type expressions to category symbols. More specifically, every occurrence of B in the previous assignment is replaced by $Sit \rightarrow B$. The resulting assignment is shown in figure 3. It is useful to compare table 3 with table 2.

Under this semantics sentences denote objects of type $Sit \to B$. It is possible to assign a semantic value to the word "believes" such that the sentence "Mary believes Φ" is true of a situation s, even though Φ is false of s. Even the situation based semantics of sentences may not be adequate. The following sentences present semantic difficulties of various kinds.

Mary wants John to go away.

Mary saw John run down the street.

Mary knows John's telephone number.

6 Further Reading

An overview of modern linguistics can be found in [Sells, 1985]. An introduction to Monatgue grammar and categorial grammar can be found in [Dowty *et al.*, 1981]. A broad collection of papers on categorial grammar can be found in [Oerhle *et al.*, 1988].

References

[Dowty *et al.*, 1981] D. Dowty, R. E. Wall, and S. Peters. *Introduction to Montague Semantics.* D. Reidel, 1981.

[Oerhle *et al.*, 1988] R. T. Oerhle, E. Bach, and D. Wheeler, editors. *Categorial Grammars and Natural Language Structures.* Reidel, 1988.

[Sells, 1985] Peter Sells. *Lectures on Contemporary Syntactic Theories.* Center for the Study of Language and Information, 1985.

7 Problems

1. Fill the missing semantic types in the following table. Note that this table is different form the one given in the notes. In the table given below VP is a primitive syntactic type. Each semantic type expression should be an arrow expression built from the basic semantic types D and B.

Category Name	Example Phrases	Arrow Category	Semantic Type Expression
S	((the bird) flies)	category symbol	B
N	man, (yellow bird)	category symbol	$D \to B$
VP	runs, (likes Tom)	category symbol	$D \to B$
NP	Mary, (every bird)	$VP \to S$	
Adj	big, yellow	$N \to N$	
AP	(on (the table))	$N \leftarrow N$	
Adv	(into (the house))	$VP \leftarrow VP$	

2. Under the type system given in problem 1 the word "every" has syntactic category $N \to NP$. Give a semantic value for the word "every" that is compatible with the semantic type expressions of part a. Your value should be a λ-term of the higher order logic and should give the intuitively correct meaning for the sentence "Every bird flys".

3. Show the equivalence of categorial and context free grammars.