Generative Grammar

Gerald Gazdar

April 1986

Cognitive Science Research Paper

Serial No. CSRP 057

Cognitive Studies Programme,
The University of Sussex,
Brighton BN1 9QN

To appear in: John Lyons, Richard Coates, Margaret Deuchar, and Gerald Gazdar, eds. *New Horizons in Linguistics II*. Harmondsworth: Penguin, 1987.

SSX

OS?

Generative Grammar

Gerald Gazdar
University of Sussex

1. Introduction

Traditionally, the term "generative grammar[11] has been used to refer to an approach to linguistics which is characterized by the goal of investigating natural language through the construction of (i) mathematical models of (aspects of) particular languages (we will refer to such models as "generative grammars") and (ii) a general mathematical framework for building such models (we will refer to such frameworks as "theories of generative grammar").

By way of analogy to (i), consider a mathematical model of the behaviour of tides in the Bristol Channel, and by way of analogy to (ii), consider a general theory of the interaction of tides with estuaries, one which would allow one to construct not just a predictive model of the Bristol Channel, but also the Nile Delta, the mouth of the Rhine, and so on. This analogy is misleading in one respect — a tide model is likely to be quantitative, as are most mathematical models in the physical and social sciences, whereas generative grammar is rarely, if ever, quantitative. Generative grammar involves the mathematics of structures and strings, not the mathematics of numbers.

An early paper of Chomsky's provides us with succinct definitions of "generative grammar" and "theory of generative grammar", and it is worth quoting the passage here.

> A generative grammar is a device (or procedure) which assigns structural descriptions to sentences in a perfectly explicit manner, formulated independently of any particular language. A particular theory of generative grammar - specifies a class of potential grammars (that is a schema for grammars), and provides a procedure by which a structural description is assigned to an arbitrary sentence by arbitrary grammar of this class - that is, it explains how a grammar of the specified form provides structural information about sentences. A theory of grammar is open for discussion only insofar as it meets these conditions. To the extent that it fails to provide an explicit characterization of the class of grammars and of the manner in which grammars assign structural descriptions, it cannot be confronted by evidence, and can receive neither criticism nor support (Chomsky 1964, 995)

The basic assumption made in generative grammar is that a language can be regarded as a set whose membership is precisely specifiable by rules (we use 'set* here informally to refer to any collection of objects: there is a technical issue as to whether natural languages can be construed as sets in the conventional mathematical sense — see Langendoen & Postal 1984). In what follows, we will take the elements of such a set to be all the expressions in the language, not merely the expressions corresponding to sentences but also subsentential expressions of all categories. The set of compound linguistic expressions in a natural language is not finite, so we cannot list them all (cf. *a tall story, a very tall story, a very very tall story, ...)*. What we need are formal (i.e. mathematical) systems which define the membership of the infinite sets of linguistic expressions and assign a structure to each member of these sets. Such formal systems are generative grammars.

The notion of generative grammar that we have outlined above, and to which this chapter is devoted, is normally seen as having its origins in work that Chomsky (and others - e.g. Bar-Hillel 1953) did in the 1950's (Chomsky 1955/1975) and early 1960's. As a consequence, the term "generative grammar[11] has been, and is, widely used to refer to just whatever it is that Chomsky happens to be doing (likewise the terms "transformational grammar[11] and "transformational-generative grammar", cf NHL 1. 25). Riemsdijk (1982). for example, is particularly explicit in this usage, and even critics of Chomsky's work sometimes adopt it (Comrie 1984). The usage is especially unfortunate at the present time since Chomsky has not published any new mathematical work on language for two decades, and has explicitly repudiated the basic notions of generative grammar including that of a 'language* (1981, 18) and the notion of 'rule* (1984. 25).

2. Generative grammar in the 1970's

As noted in the introduction, generative grammar has its origin in the 1950's and 1960#s. But that period is well covered in Lyons (1970) and so we take up the story here at the point where Lyons left off.

There is a myth about theoretical syntax that has been fostered within the discipline and widely promulgated without. The myth is that nearly all work in theoretical syntax exemplifies generative grammar. But the reality is that only a rather small proportion of the theoretical syntactic work done in the last two decades is generative in the sense defined in the quotation from Chomsky given above. Consider the proceedings of the recently instituted annual 'West Coast Conference on Formal Linguistics*: the first three volumes (1982/3/4) contain a total of 73 papers, and yet of these, less than a third are 'formal* in any sense that a logician or mathematician would recognize, and most of these are about semantics, not syntax. Occasionally, informed outsiders have commented on this gulf between private reality and public relations. Thus, we find the philosophical logician Richard Montague writing as follows in 1970.

> One could also object to existing syntactical efforts by Chomsky and his associates on grounds of adequacy, mathematical precision, and elegance; but such criticism should perhaps await more definitive and intelligible expositions than are yet available In particular, - the transformational grammarians should be expected to produce a rigorous definition, complete in all details, of the set of declarative sentences of some reasonably rich fragment of English - before their work can be seriously evaluated. (Montague 1970, 223)

These expectations were not met. and, a decade later, at least some theoretical syntacticians shared Montague's doubts about their field. In the introduction to Johnson and Postal (1980). we find the following passage.

> No serious theory of grammar « can avoid the necessity of true precision, which is, in linguistics today, a value widely praised but seldom instantiated. - It turns out that what are called theories [in linguistics] are almost invariably collections of differentially vague, partially developed or undeveloped ideas. Even highly prestigious collections of ideas do not consist of explicit sets of precise assumptions, and, to the extent that they do not it is impossible to determine their consequences, and hence to study their truth or falsity. (Johnson and Postal 1980, x-xi)

Their point can be dramatically illustrated by the following statistic: of the thousands of papers published in linguistics journals over the last 15 years on the prestigious collection of ideas known as the "(Revised) Extended Standard Theory", only one (Lasnik and Kupin 1977) has attempted to reconstruct that collection of

ideas mathematically. The paper was published in a relatively obscure journal and had little impact on the framework that it addressed.

The fact is that generative grammar, in the original sense of the term, virtually died out in linguistics in the 1970's. That fact is almost certainly causally linked to what has come to be known as the "Peters-Ritchie result". At the beginning of the decade, in a classic series of mathematical papers. Peters and Ritchie (e.g. 1973) had shown that 1960Vstyle transformational grammars were equivalent to Turing machines. This meant that the claim that natural languages had transformational grammars essentially only amounted to the claim that they could be characterized mathematically. Beyond this, the claim had no content.

For the most part, theoretical syntacticians in the early 1970*s had no interest in mathematical models of language. In Newmeyer's (1980) felicitous phrase, they were engaged in the "linguistic wars" that raged between Generative Semantics and Interpretive Semantics (or the Extended Standard Theory). Despite their names, both frameworks were primarily (Generative Semantics) or entirely (Interpretive Semantics) concerned with matters of syntax. Neither transformational framework had any mathematical pretensions, nor did those working in the frameworks see any merit or interest in matters of formal detail. The notation and technical vocabulary of generative grammar, to the extent that it was applicable at all, was used only in a metaphorical sense.

The only strand of work that stands as an exception to the trend described above came, not out of linguistics, but out of philosophical logic. Montague published a series of papers around 1970 which were to prove increasingly influential in the ensuing years. These papers, with titles like "Universal grammar" (1970a), "English as a formal language" (1970b). and "The proper treatment of quantification in ordinary English" (1973) were generative grammar such as nobody had ever seen before. Montague had not merely formalized the syntax of substantial fragments of English, itself a momentous achievement, but he had also given a completely explicit semantics for each syntactic fragment. Furthermore, he had done so without paying any heed to "the attempts emanating from the Massachusetts Institute of Technology" as he contemptuously referred to the transformational framework in the introductory paragraphs of his most famous paper (1973, 247).

Montague's work was first made accessible to linguists in an important article by Partee (1975). Initially, his contribution was perceived to be almost entirely semantic: by the late 197O's all serious formal semantic work in linguistics was being done within the paradigm that Montague had initiated. His work provided a sound framework for the previously amateur logic that had become popular in linguistics in the late 1960*s (cf. NHL 1. 132-134, 138). NHL 1, unsurprisingly for a linguistics book published in 1970, contains no reference to Montague, but some sense of the gap that he filled can be inferred from Fodor's comment that "something is always better than nothing and in the area of semantics we are preciously close to having nothing" (NHL 1, 210). In addition. Montague's work was also to have significant implications for both the methodology and substance of syntactic theory. Methodologically, his work had at least the following three implications. Firstly, he showed that it was possible to do generative grammar, something which linguists had ceased attempting to do. Secondly, he maintained that it was not enough to simply define the syntax of a language, but that one had, in addition to associate an equally explicit semantic theory with that syntax. These two points are neatly encapsulated in the often-quoted opening to his "Universal Grammar":

> There is in my opinion no important theoretical difference between natural languages and the artificial languages of logicians; indeed, I consider it possible to comprehend the syntax and semantics of both kinds of languages within a single natural and mathematically precise theory. (Montague 1970a, 222)

Finally, he reintroduced the notion of a 'fragment' of a language and the goal of writing completely explicit grammars for such fragments. This was an alien idea to many syntacticians, both then and now. Syntactic argumentation, as it developed within linguistics during the 1960's and 1970's, encouraged one to invoke as evidence any aspect of the language in question, including aspects of the language for which no analysis was available and where the facts were correspondingly ill-understood. Montague's fragment methodology led to a very different way of working, at least among those that adopted it.

Montague himself died in 1970, but thanks to the work of such scholars as Bach (e.g. 1979), Dowty (e.g. 1978), Karttunen (e.g. 1977), Partee (e.g. 1979), and Thomason (e.g. 1976), the paradigm of work that he started kept generative grammar alive during the 1970's.

3. *Some simple generative grammars*

Up until now we have discussed generative grammar in the abstract without ever exhibiting an example of the beast. In this section and the next, we will look in some detail at what contemporary generative grammars look like, and how they work. All the kinds of generative grammar that have been used in linguistics have contained, in one form or another, the following three components: (i) a set of syntactic categories or "parts of speech", (ii) a lexicon, dictionary or word list, and (iii) a set of syntactic rules.

What is a syntactic category? Well, at it simplest, a syntactic category is the name we give to collection of expressions of the language that have the same distribution. So, for example, *Sandy* and *the person you spoke to* are both noun phrases: they may or may not refer to the same person, but regardless of that, they have exactly the same syntactic privileges of occurrence. Construct any English sentence you like using the proper name *Sandy* and you will be able to construct another perfectly grammatical English sentence by substituting *the person you spoke to* for *Sandy* in the sentence you first constructed. The two sentences may mean different things, but they will both be grammatical. Examples of syntactic categories that linguists commonly make use of are "noun phrase" (NP), "sentence" (S), "verb phrase" (VP), "verb" (V), and so on.

What is a lexicon? A lexicon is minimally a list of words which associates each word with its syntactic properties. The most important of these properties is its (gross) syntactic category, for example, whether the word is a verb or a noun. In addition, depending on the sophistication of the overall grammar, the lexicon will contain information as to the subcategory of the word (e.g., whether a particular verb is transitive or intransitive), other syntactic properties (e.g. the gender of a noun in a language that makes gender distinctions), and perhaps also morphological and semantic information.

What is a syntactic rule? The answer to this question varies considerably depending on the precise character of the theory of generative grammar involved, and often such theories permit the use of more than one kind of rule. One very frequently employed type of syntactic rule is the "phrase structure rule", and most current linguistic frameworks, whether generative or not, presuppose the existence of such rules. A phrase structure rule simply tells one what a particular syntactic category can be composed of. Thus, "S -• NP VP" tells us that a sentence can consist of a noun phrase followed by a verb phrase, and "VP -• V NP" tells us that a verb phrase can consist of a verb followed by a noun phrase. If you treat the arrow *-*" as a convenient abbreviation for the expression "can consist of* when looking at phrase structure rules, then you will not go far wrong.

To start with, let us consider a very straightforward phrase structure grammar (Grammar 1) having the three components outlined above.

```
Grammar1    -  <Categories!, Rulesi, Leziconl>
```

This little grammar will employ four syntactic categories, namely noun phrase, sentence, verb phrase, and verb, as follows.

```
Categoriesi -   {NP, S, VP, V}
```

And we will supply the grammar with only three rules, one telling us that a sentence can consist of a noun phrase followed by a verb phrase, another which lets a verb phrase consist of a verb and a noun phrase, and a third which lets a verb phrase consist of a verb standing on its own.

```
Rules 1       -   {S   -> NP VP,
                    VP  -  V,
                    VP  -  V NP}
```

The final component we need is a lexicon to supply us with some ready-made verbs and noun phrases. So the first line of the lexicon tells us that *Kim, Fido, children,* and *dogs* are all noun phrases (admittedly one-word 'phrases')* and the second line tells us that *barked* and *chased* are verbs.

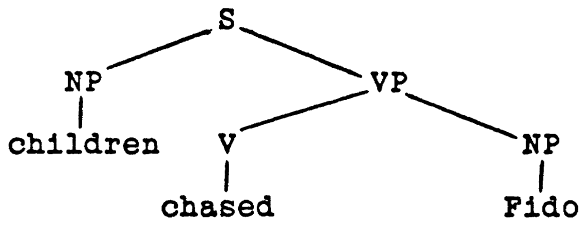
```
Lexlconl    -  {NP -> {Kim, Pido, children, dogs},
                V  -+ {barked, chased}}
```

Actually, our lexicon here is really made up of two specially notated phrase structure rule schemata (i.e. "NP -• *{Kim, Fido, children, dogs}** and "V -+ *{barked, chased}*)* that tell us that the category on the left of the arrow can consist of any one of the items found between curly brackets on the right of the arrow. In principle then, we could, if we wished, collapse our rules and our lexicon into a single component (since both components use phrase structure rules in this example), but we will keep them separate in the interests of clarity in subsequent discussion.

We have now completely defined Grammarl. But what is it good for? What can it do? Well, a generative grammar has two functions in life. The first is to define the sets of grammatical strings of words in the language under consideration (or, more realistically, part of that language). So, in the case of a natural language like English, it should define the set of grammatical sentences, the set of grammatical verb phrases, and so on. In defining the sets of grammatical strings it will, by implication, also define the ungrammatical strings (since an ungrammatical string is simply a string which is not grammatical, i.e. a string which is not 'generated* by the grammar). The second function that the grammar has is to associate with each grammatical string one or more structures. Typically, if a given string has two different structures then it will also have two different meanings, and hence be ambiguous. Part of the job of the grammar, then, is to make correct predictions about this kind of structural ambiguity.

What are these structures? The answer, in mathematical parlance, is "directed acyclic graphs": all contemporary grammatical frameworks, without exception, employ some kind of directed acyclic graph to represent structure. We will not delve into graph theory here, however. Conveniently, most, but by no means all.

contemporary linguists have elected to use one particular kind of directed acyclic graph, namely the "tree" for their structural descriptions. And a tree is what it sounds like: it has a root, it has branches, and it terminates in leaves. But linguists, like genealogists, exhibit their trees upside down, with the root poking into the sky, and the leaves on the ground. Here is such a tree.



How does Grammar1 define sets of strings and associate those strings with structures like the one shown above? Consider the top of the tree: here we have an S which has as daughters an NP and a VP (and nothing else) appearing in that order. The first rule of Grammar1 says that an S can consist of an NP followed by a VP, and that is exactly what we have here. So the grammar legitimates this part of the structure. If we turn our attention now to the VP, then again we find that the grammar contains a rule which corresponds to this part of the structure. Finally, we look in the lexicon and find that *children* and *Fido* are both NP's, as the tree claims, and that *chased* is a V. So every substructure in the tree corresponds to some rule in the grammar, and the tree is thus admitted by the grammar. Since the tree is a legitimate structure, it follows that the string of words made up of the leaves of the tree, namely *children chased Fido* is, according to Grammar1, a grammatical expression of category S (i.e., a sentence).

This is not the only string of words that Grammar1 will admit as a sentence - there are 39 more such strings including *Fido barked*, *Kim chased dogs*, etc. Unfortunately, **Kim barked dogs* and **children chased* are also among the strings that Grammar1 claims to be sentences (linguists use "*" to mark strings that are intuitively ungrammatical in the language under consideration, in this case English). The problem involves what linguists call 'subcategorization' (cf. NHL 1, 136, 280). Although *barked* and *chased* are both verbs, as Grammar1 claims, they belong in different subcategories of the class of verbs. Specifically *chased* is transitive and requires a following NP, whereas *barked* is intransitive and cannot tolerate a following NP. We could patch Grammar1 up by replacing "V" with two categories "IV" (intransitive verb) and "TV" (transitive verb) and revising the VP rules and lexicon accordingly, but such an approach rapidly proliferates a host of distinct parts of speech once a larger class of sentences is considered.

Instead of pursuing the ad hoc solution just mentioned, we will turn instead to a more principled approach, one that employs syntactic features. Most, if not all, contemporary theories of generative grammar employ features, and the extent and sophistication of their use has grown massively in the 1980's, though their roots go back to the work of Harman (1963) and Yngve in the early 1960's. In a modern feature-theoretic syntax, atomic categories such as NP, V, and so on, are replaced by sets of feature specifications. Each feature specification consists of a feature (say CASE) and a value for that feature (say ACCUSATIVE). The familiar names for categories such as "NP", "V", and so on, are then reintroduced as convenient abbreviations for sets of feature specifications.

In the light of this, what we will do now is rebuild Grammar1 using features, thus creating Grammar2 and solving the problem with subcategorization noted above. In order to make Grammar2 a little more interesting than it would otherwise be, we will also introduce a rule for coordination to exemplify the use of rule schemata in generative grammars. A rule schema is a rule containing one or more variables. Having done that, we will then extend Grammar2 into Grammar3 and in so doing illustrate the descriptive power of feature-theoretic techniques on a range of syntactic phenomena.

```
Grammar2     =  <Features2, Categories2, Rules2, Lexicon2>
```

We will employ just two features in Grammar2, namely CATegory and SUBCATegory. CAT will have as its values the labels that were used for categories themselves in Grammar1 (with one addition "C" which we discuss below). SUBCAT will, for the moment, have just two values, namely "0" (i.e. nothing) and "NP".

```
Features2    =  {<CAT,     {NP, S, VP, V, C}>,
                 <SUBCAT,  {NP, 0}>,
```

How are we to interpret this? Well, consider a category made up like this: {<CAT, V>, <SUBCAT, NP>}. This category has V as the value of its CAT feature, so it is a verb, and it has NP as the value of its SUBCAT feature. We will interpret the latter to mean that it is the kind of verb which requires a following noun phrase, that is, it is a transitive verb. A verb with 0 as its SUBCAT value will be a verb that requires nothing to follow, that is, an intransitive verb.

Since things like {<CAT, V>, <SUBCAT, NP>} are tedious to write, and confusing to read, we will now introduce some abbreviations for our feature-sets.

```
Categories2 =  {X     = {<CAT, X>, ...},
                X[Z]  = {<CAT, X>, <SUBCAT, Z>, ...}}
```

X and Z here are variables that range over the relevant set of values for the feature mentioned. Thus we can abbreviate {<CAT, V>, <SUBCAT, NP>} to "V[NP]" and use the latter as our name for the category of transitive verbs. Armed with these abbreviations, we can state rules as succinctly as we did in Grammar1.

```
Rules2       =  {S   → NP   VP,
                 VP  → V[Z]  Z,
                 X   → X  C  X}
```

The first rule looks the same as its counterpart in Grammar1, and it performs exactly the same function. But in reality it is an abbreviation for the following cumbersome object: "{<CAT, S>} → {<CAT, NP>} {<CAT, VP>}". The second rule in Grammar2 performs the function of the second and third rules in Grammar1, and in so doing, it illustrates the use of rule schemata. The schema says that verb phrase consists of a verb followed by the category whose abbreviation corresponds to the verb's SUBCAT feature. Thus a transitive verb (V[NP]) will be followed by NP, whereas an intransitive verb (V[0]) will be followed by nothing. Actually, we are trading on an equivocation between feature names and category abbreviations here, but eliminating the equivocation would make this much harder to follow. Grammar2 makes the correct claims about the grammaticality of the following examples.

```
Fido barked.
*Fido barked dogs.
*Children chased.
Children chased Fido.
```

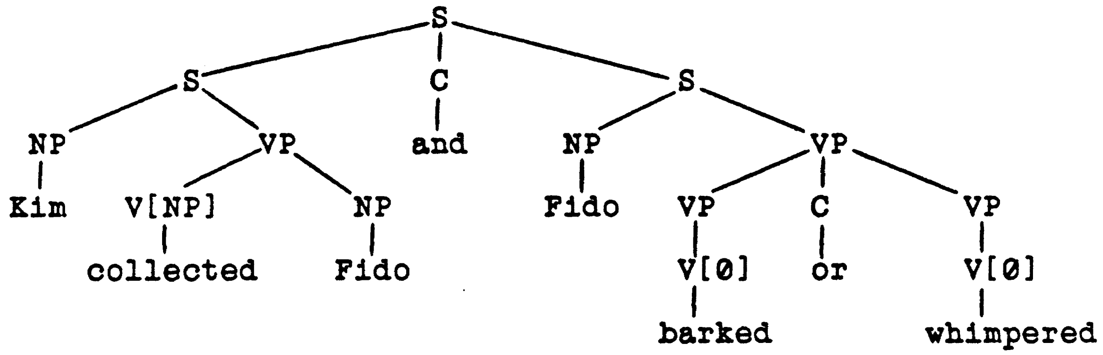
The grammar will assign structures to the first and last examples, but no structures are available for the second and third. Hence, as far as Grammar2 is concerned, these two examples are ungrammatical.

The third and final rule is a schema which takes us beyond the domain covered by Grammar1. This schema introduces the coordinate construction and says that a given category can consist of two instances of the category separated by an item of category C (which will turn out to be realized as "and" or "or").

```
Lexicon2    =  {NP    → {Kim, Fido, dogs, children},
                V[0]  → {barked, whimpered,  slept},
                V[NP] → {chased,  loved, collected},
                C     → {and, or}}
```

The lexicon for Grammar2 is much the same as that for Grammar1, except for the addition of the category C, and a few extra verbs to enhance the plausibility of the examples we shall give.

Apart from the introduction of features, which at this stage may well look to have been of much more trouble than it is worth, Grammar2 appears very little different from Grammar1. But there is one very fundamental difference between them. Grammar1 claimed that exactly 40 strings of words were grammatical instances of the category S (the reader is encouraged to work out how this figure of 40 is arrived at), whereas Grammar2 admits infinitely many strings of words as grammatical instances of S. How can this be? The answer lies in the coordination schema we have introduced into Grammar2. This allows any category to split, amoeba-like, into two instances of the same category. These new instances may themselves split, and so on. This aspect of the grammar provides an example of what is known as 'recursion' (see the examples following Grammar3, below, for a rather different example of this phenomenon). Grammar2 permits recursion, whereas Grammar1 did not. This should become clearer from an example, and so we exhibit below one of the infinitely many trees that Grammar2 admits as grammatical.



Here we have two sentences coordinated, and the second sentence itself contains a coordinate verb phrase. Thus the example illustrates how the S category can reintroduce the S category, and how the VP category can reintroduce the VP category. The grammar imposes no limit on how many times categories can get reintroduced by our coordination rule, and so there is no limit on the number of expressions that Grammar2 can admit. Hence Grammar2 admits all of the following examples.

```
Fido barked.
Fido barked and children whimpered.
Fido barked, and children whimpered, and Kim slept.
Fido barked, and children whimpered, and Kim slept,
 and dogs chased Fido.
Fido barked, and children whimpered, and Kim slept,
 and dogs chased Fido, and ...
```

Although Grammar2 allows for the grammaticality of infinitely many sentences, we would rapidly grow bored with an enumeration of them. This is because the lexicon for Grammar2 is so small. Let us therefore expand Grammar2 into Grammar3, and, in so doing enlarge our lexicon by exploiting more fully the technique for subcategorization that we employed in Grammar2. The overall structure of Grammar3 is identical to that of Grammar2:

```
Grammar3     =  ‹Features3, Categories3, Rules3, Lexicon3›
```

And we will carry over the very same set of abbreviations for categories:

```
Categories3 =  {X     = {‹CAT, X›, ...},
                X[Z]  = {‹CAT, X›, ‹SUBCAT, Z›, ...}}
```

The structure of the feature system is also exactly the same, but we will allow ourselves a greater range of values for each of our features.

```
Features3    =  {‹CAT,     {NP, PP, AP, S, VP, V, P}›,
                 ‹SUBCAT, {NP, PP, AP, S, VP, 0}›}
```

Here, P, PP, and AP are mnemonic for preposition, prepositional phrase, and adjective phrase, respectively. The rules for Grammar3 are those employed in Grammar2 with one addition, a rule for expanding prepositional phrases.

```
Rules3       =  {S   → NP  VP,
                 VP  → V[Z]  Z,
                 PP  → P[Z]  Z,
                 X   → X  C  X}
```

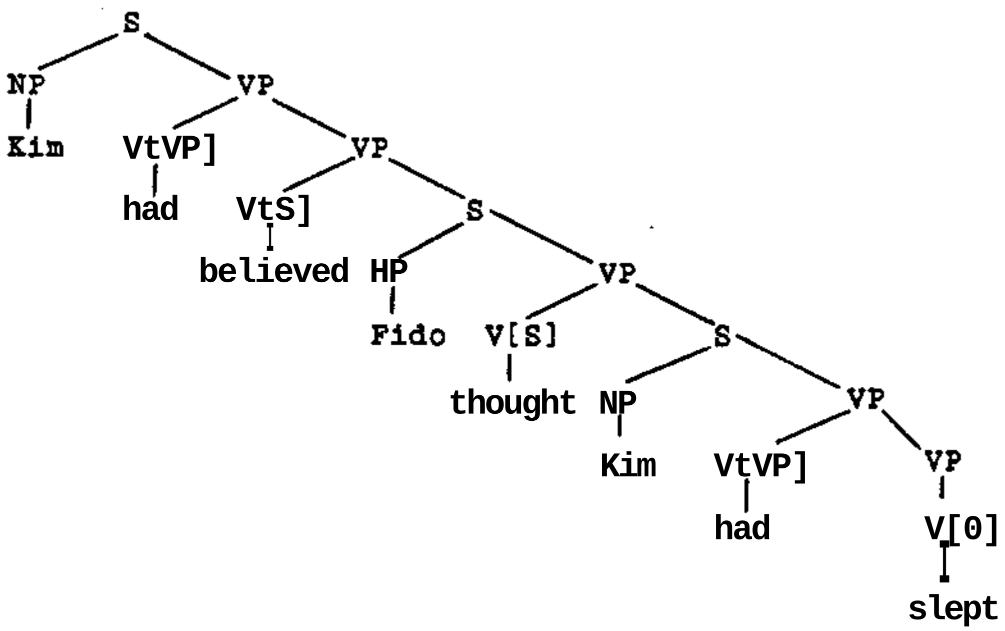
It is only when we come to the lexicon of Grammar3 that differences really become apparent. Grammar3 allows us to employ a much wider range of distinct subtypes of lexical item, including six different kinds of verb.

```
Lexicon3     =  {NP     → {Kim, Fido, dogs, children},
                 AP     → {happy, stupid, middle-aged},
                 V[0]   → {barked, whimpered,  slept},
                 V[NP]  → {chased,  loved, collected},
                 V[PP]  → {approved, disapproved},
                 V[AP]  → {appeared, seemed},
                 V[VP]  → {had},
                 V[S]   → {believed, thought},
                 P[NP]  → {of},
                 C      → {and, or}}
```

Grammar3 provides us with structures for all the following examples (and an infinity of others).

```
Kim thought Fido seemed stupid.
Fido appeared middle-aged and disapproved of children
Kim had believed Fido thought Kim had slept.
```

This last example illustrates another very common form of recursion found in natural languages, namely recursion through the sentential (i.e. clausal) or verb phrase complements of a verb. Here is the tree structure for the sentence.



Our final example grammar will maintain the overall structure employed in Grammar2 and Grammar3.

```
Grammar4    -   <Features4, Categories4, Rules4, Lexicon4>
```

And Lexicon4 will be identical in every respect to Lexicon3» so we shall not repeat it here. In fact* the only real change we will make in Grammar4 is the addition of a new feature which we will call SLASH (for a reason that will shortly emerge). However, the addition of this feature to the feature system requires certain consequential additions to the category abbreviations and rules.

```
Features4   -  {<CAT,     {NP, PP, AP, S, VP, V, P}>,
                <SUBCAT, {NP, PP, AP, S, VP, 0}>,
                <SLASH , {NP, PP, AP}>}
```

As can be seen, Features4 is just Features3 with the addition of SLASH and its permissible range of values. As in the previous grammars, we will introduce a notational abbreviation for categories that bear the SLASH feature.

```
Categories4 - {X     - {<CAT, X>, . . • },
               X[Z]  - {<CAT, X>, <SUBCAT, Z>, . •.}
               X/Y   « {<CAT, X>, <SLASH, Y>, . •.}}
```

Thus the feature SLASH turns out to be named after its conventional notational abbreviation. Expressions like "S/NP" and "VP/PP" stand for a particular kind of category, then, but what kind? What is the intuitive content of the notation? The answer is quite straightforward: an expression of category X/Y is an expression of category X from which an expression of category Y is missing. Thus an S/NP (read "S-slash-NP") is a sentence (or clause) which has got a noun phrase missing, and a VP/PP is a verb phrase that is missing a prepositional phrase.

In order to make use of, or even to make sense of, these "slash categories", we will need to make a number of additions to the rules that Grammar3 came equipped with.

```
Rules4      =  {S    → NP   VP,
                S/Y  → NP   VP/Y,
                VP   → V[Z]  Z,
                VP/Y → V[Z]  Z/Y,
                PP   → P[Z]  Z,
                PP/Y → P[Z]  Z/Y,
                X    → X  C  X,
                S    → Y  S/Y,
                X/X  → 0}
```

There are five new rules here, and we will discuss them in turn.

```
S    → Y  S/Y
```

This rule says that a sentence can consist of some category followed by a sentence which is missing an expression of that category. Given the range of values available for SLASH, this rule schematizes the following three instantiations of the rule.

```
S    → NP  S/NP
S    → PP  S/PP
S    → AP  S/AP
```

And these in turn, taken together with the rest of Grammar4, will permit such sentences as the following:

```
Children, Fido disapproved of _.
Of children, Fido disapproved _.
Middle-aged, Kim had seemed _.
```

As can readily be seen, the sentences that follow the comma in these examples all have something missing. The rule which is responsible for this missing item in Grammar4 is this one:

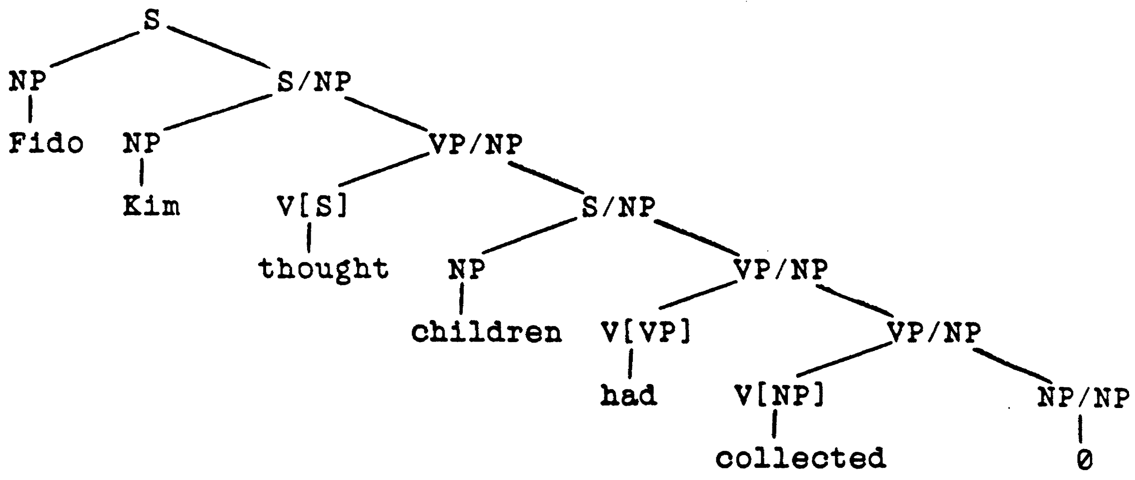
```
X/X  → 0
```

This just says that an "X missing an X" can be realized as the empty string (as previously, we are using the symbol "0" to make the empty string visible in our grammar). Thus NP/NP, PP/PP, and AP/AP are all allowed to appear as nothing in the structures defined by Grammar4.

Finally, we find the following three rules in Grammar4 but not in Grammar3:

```
S/Y  → NP   VP/Y
VP/Y → V[Z]  Z/Y
PP/Y → P[Z]  Z/Y
```

The first of these, which can stand as an exemplar for all three, says that a sentence which is missing a Y can consist of a noun phrase followed by a verb phrase which is missing a Y (where Y might be NP, say). So what these three rules do is transfer information about a missing category from mother to daughter. The way this works should become clearer if we look at a relevant tree.



There is, in principle, no limit on the amount of intervening material that can occur between the 'displaced' constituent at the front of the sentence and the empty constituent that corresponds to it and which occurs within the sentence. Such constructions manifest what are known as 'unbounded dependencies' and are surprisingly common in the world's languages. English, for example, makes extensive use of this type of construction, most notably in questions and relative clauses (cf. NHL 1, 131). Although the analysis of the construction given here is rudimentary, it does serve to illustrate the sort of feature passing technique that is very common in current generative grammar.

The grammars we have been elaborating are what linguists call "toy grammars". They serve to illustrate something of the way contemporary feature-theoretic generative grammars work, but they should not be taken too seriously as analyses of English. The fact that Grammar3 and Grammar4 work as well as they do owes a lot to the particular lexicon that they employ. The reader who has mastered the mechanics of at least Grammar3 might usefully consider the questions which follow.

What happens if you try and add the relevant form(s) of the irregular verb *go* to the lexicon? Can the analysis of *have* be maintained in the light of such additions? What fact about the existing choice of verbs makes the treatment of *have* seem superficially plausible? Does Grammar3 as it stands permit any ungrammatical sentences containing *had* to be generated?

All the verb forms given are past - what happens if you try and add the corresponding present tense forms?

The NP section of the lexicon does not contain any pronouns - what happens if you try and add *I*, *him*, etc. to the NP lexicon? Which English pronoun can be added without leading to ungrammatical sentences?

The particular featural treatment of subcategorization adopted in Grammar2 suffers from a fundamental restriction - what is it? What English verbs lie outside its scope?

In Grammar4, there is a generalization to be made about the relation between the rules which transfer SLASH information from mother to daughter and their counterparts which do not. This generalization is not, however, expressed in the form of Grammar4. What is the generalization?

Being able to find fault in a generative grammar is one thing, but being able to repair or improve a descriptively inadequate grammar is a whole lot harder. The ambitious reader may care to pursue one or more of the following projects in order to get a feel for the latter enterprise.

Modify Grammar3 so that it can distinguish between a past participle and the past tense form. Add the relevant forms of the verbs *go*, *eat*, and *know* to the lexicon. Check to see that *had* now works properly.

Add a feature (or features) to Grammar3 so that it can distinguish plural and singular. Encode this in the NP part of the lexicon in an appropriate manner. Augment the verbal part of the lexicon to include the present tense forms of the verbs. Now modify the rules so as to ensure subject-verb agreement. What problems would you encounter if you tried to add the verb *be* to your modified grammar?

Add a feature (or features) to Grammar3 so that it can distinguish between subject and non-subject noun phrases. Add *she*, *him*, *they*, and *them* to the NP part of the lexicon. Make sure that the grammar does not legitimate such ungrammatical strings as **them approved of she*.

Reconstruct Grammar2 using a different approach to subcategorization and show that your version of Grammar2 can be trivially extended to permit the inclusion of such verbs as *hand* (as in *I handed Kim Fido)*, *tell* (as in *I told Kim Fido chased children)*, and *bet* (as in *I bet Kim thousands Fido had won)*.

Add exactly one rule (and make no other changes) to Grammar4 in order to enable it to handle examples like *Fido, Kim thought seemed stupid*.

Add a rule (or rules) to Grammar4 so that it can handle two-word noun phrases like *a dog*, *the children*, and so on. Augment the lexicon accordingly. Now add one further rule to permit noun phrases containing simple relative clauses such as *a dog Fido had chased* and *the children Kim approved of*. Doing this will require you to employ the SLASH feature.

If you have some mastery of a language other than English, then try and work out a version of Grammar4 for that language using a lexicon which employs the relevant translations of the words which appear in Grammar4's English lexicon.

Typically, syntactic categories are defined as sets of syntactic feature specifications as in our last three example grammars, above (cf. NHL 1, 126,135,274). A feature specification is a pair consisting of a feature (e.g. CASE) and a feature value (e.g. ACCUSATIVE). In the toy grammars we presented in the previous section, these values have all been atomic, which is to say that they had no internal structure. This keeps things simple for pedagogic purposes, but does not do justice to the sophistication of contemporary feature theory in which the value of a feature in a category may itself be a syntactic category (i.e. features are allowed to take categories as their values). This allows features such as SLASH and SUBCAT, above, to encode not just gross categorial distinctions (as flagged by "NP", "PP", etc., in our example grammars) but also linguistically important information having to do with, for example, case, person, number, and gender. Category-valued features allow many significant grammatical generalizations to be captured rather straightforwardly. For example, a category-valued version of our SLASH feature is able to capture the generalizations underlying the class of unbounded dependency constructions (e.g. relative clauses, embedded questions, topicalization, etc.).

Principles of feature matching are standardly invoked to ensure the identity of certain features on adjacent nodes. Such principles are equations which say that certain features appearing on one node must be identical to the features appearing on another. Most current work assumes a "Head Feature Convention" which is responsible for equating one class of feature specifications as they appear on the mother category and its head daughter(s). Thus, for example, a verb phrase inherits the tense of its verb since the latter is the head of the verb phrase. Other principles match agreement features between agreeing categories (e.g. between a subject noun phrase and its verb phrase sister), or deal with the copying of category valued features between mother and daughter categories (for example, one would typically employ such a principle to handle the distribution of our SLASH feature).

The mathematical definitions of principles of feature matching, such as those mentioned above, crucially depend upon notions of extension and unification. These notions were introduced into linguistics by Kay (1979) and have been profoundly influential, finding their way into most current generative grammatical theory. Assuming, for the sake of illustration, the theory of categories sketched above, we can define extension as follows.

> A category C2 is an extension of a category C1 if and only if (i) every atom-valued feature specification in C1 is in C2, and (ii) for every category-valued feature specification in C1, the value of the feature in C2 is an extension of the value in C1.

This recursive definition (notice that the second clause invokes the notion being defined) says first of all that any specification for an atom-valued (i.e. non-category valued) feature in a category is also in all extensions of that category. It also guarantees that if a category specifies a value *v* for some category-valued feature, then any extension of that category specifies a value for that same feature that is an extension of *v*. Note that an extension of a category C may contain a specification for a category-valued feature which is unspecified in C. The relation "is an extension of" is thus a generalization of the relation "is a superset of", one which takes proper account of category-valued features, and it defines a partial order on the set of categories.

An important operation on categories is that of unification. This notion is closely analogous to the operation of union on sets except that, as in the case of extension, the resulting set must be a function. Unification is undefined for categories containing features specifications that contradict each other.

> The unification of a set of categories is the smallest category which is an extension of every member of the set, if such a category exists, otherwise the unification is undefined.

Pereira & Shieber (1984) is currently the definitive work on the underlying mathematics of unification-based linguistic formalisms but Shieber (1986) provides an excellent introduction. In discussing this approach, we have restricted ourselves to feature theory for the purposes of illustration. However, some current linguistic theories theories allow one to perform unification on structural descriptions (as in "Lexical Functional Grammar" — see below), and even on whole grammars (as in Kay's (1984) "Functional Unification Grammar").

In any feature-theoretic linguistic theory, certain feature values are the expected case, the values that ordinarily get assigned, other things being equal. Linguists call these expected values the "unmarked" or default values, and they can be handled by conditions that a category must meet if it can, but need not meet if it cannot. Thus, for example, the default value for CASE might be ACCUSATIVE, but a given noun phrase could appear in some other case if it was required to do so by a rule or by a principle of feature matching, say.

As we have presented them, grammars are things which generate languages. But a grammar is itself expressed as a series on expressions in a formal language. So we might reasonably ask ourselves whether we can write a grammar for our grammars, or, less ambitiously, whether we can write a grammar for parts of our natural language grammars. The curious reader may care to investigate the possibility of using the style of grammar employed in Grammar1 to write a grammar which will generate Grammar2, Grammar3, Grammar4 and an infinity of other grammars written in the style used for those three. The idea of using one grammar to generate another originates in computer science with the work of van Wijngaarden (1969) who used the technique to give a perspicuous syntax for ALGOL68. The same idea emerges in recent linguistic work in the guise of the "lexical rule" (Bresnan 1978) or metarule (Kay 1983, Thompson 1982, Stucky 1983). A metarule is a grammar characterization device (i.e. a clause in the definition of the grammar), one which enables one to define one set of rules in terms of another set, antecedently given. Generalizations which would be lost if the two sets of rules were merely listed are captured by the metarule.

4. Generative grammar in the 1980's

In this section we will sketch very briefly four theories of generative grammar that linguists are actively working on at the present time.

4.1 Generalized Categorial Grammar

Most work in syntax in the 1970's assumed that the description of a linguistic expression must make use of more than one level of syntactically defined structure. By contrast, much 1980's generative work adopts the view that grammars are monostratal, i.e. they refer to only a single level of representation. The ready acceptance of multistratal syntactic descriptions in earlier linguistics was not dictated by any compelling arguments based on the essential character of natural languages.

Generalized categorial grammar is a sophisticated variant of categorial grammar. It falls within the family of generative theories that have been usefully characterized as "Extended Montague Grammar" by Emmon Bach and Barbara Partee, a family to which generalized phrase structure grammar (discussed below) also belongs. These theories have much in common including the working hypothesis that a single level of structural description is sufficient for both syntactic analysis and semantic interpretation, and a metatheoretical commitment to the desirability of explicitly providing the rules necessary for such interpretation.

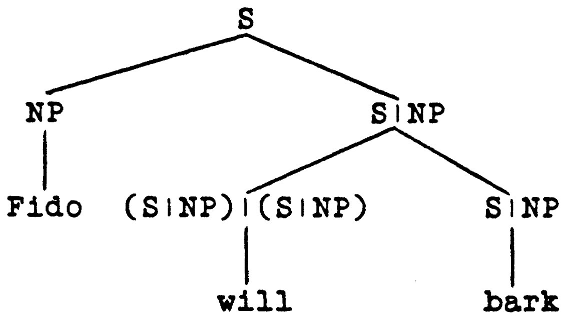
The theory of categorial grammar, which has its origins in the work of Ajdukiewicz in the 1930's and which was developed by Bar-Hillel, Curry, Lambek, and others in the 1950's and 1960's, has historically had a somewhat marginal status in linguistics. There has always been someone ready to champion it (e.g. Lyons 1968), but never enough people actually using it to turn it into a paradigm. The currency it has today is due in large measure to Montague who based his semantic work on a modified categorial grammar. However, Montague's own modifications to categorial grammar were of no great linguistic interest and most of the original "Montague Grammar" work is more concerned with semantic issues than it is with the aesthetic niceties of the underlying syntactic theory. Pure categorial grammar is essentially a variant of phrase structure grammar, one which encodes the subtance of the rules into the category system. The categories typically have the form "A|B" which stands for something which combines with something of category B to form something of category A. Thus the basic rule schemata of a categorial grammar have the following form.

$$\begin{array}{lll} A & \rightarrow & A|B \quad B \\ A & \rightarrow & B \quad A|B \end{array}$$

The variables here can themselves range over internally complex categories, thus if we set both A and B to "S|NP" (the category of constituents that combine with a noun phrase to form a sentence) then we get the following rule as a concrete instance of the first schema above.

$$S|NP \rightarrow (S|NP)|(S|NP) \quad S|NP$$

Here, by way of example, is the structure that a categorial grammar of English might assign to the sentence *Fido will bark*.



Recently some fairly principled attempts have been made, notably by Ades and Steedman (1982), Bach (1981, 1983), Flynn (1983) and Hoeksema (1984) to preserve the spirit of categorial grammar (which Montague, arguably, did not) whilst extending it to constructions that lie beyond the purview of the basic categorial apparatus. Steedman (1985), for example, defines a rule of 'partial combination' which has roughly the following form:

$$A|B \rightarrow A|C \quad C|B$$

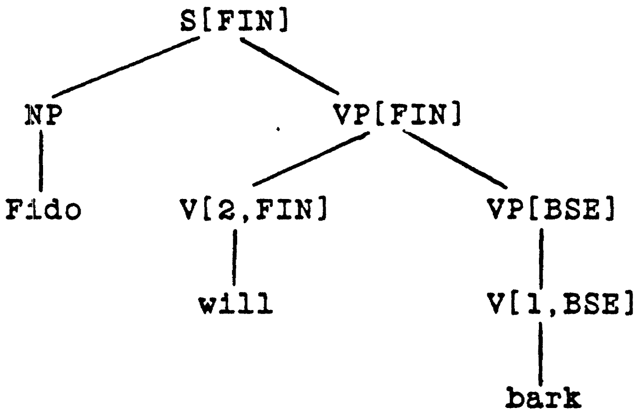
and shows how it can be used to give an insightful analysis of a range of coordination phenomena in English and Dutch.

4.2 Generalized Phrase Structure Grammar

Since 1980, following earlier suggestions by Stanley Peters, Aravind Joshi, and others, there has been a strong resurgence in the linguistics literature of the idea that phrase structure grammars could be used for the description of natural languages. PSGs had been all but abandoned in linguistics during the previous 25 years because arguments given by the proponents of transformational grammar had convinced even the few linguists who were interested in writing formal grammars that no phrase structure account of the grammar of a natural language could be adequate. But these arguments were, without exception, either fallacious or empirically ill-founded (Gazdar 1982, Pullum & Gazdar 1982).

The PSGs enjoy a wealth of literature providing them with numerous distinct but equivalent mathematical characterizations which illuminate their formal properties. They are relatively simple to write and to modify. They are associated with a successful tradition of work in computation that has provided us with a thorough understanding of how to parse, translate, and compile them (Hopcroft and Ullmann 1979). Much has been done since 1980 to establish a basis for handling the syntax and semantics of natural languages as effectively and precisely as the structures of programming languages or the artificial languages of logicians.

The major innovation that generalized phrase structure grammar (GPSG - Gazdar, Klein, Pullum and Sag 1985) makes is that the implicit PSG is not defined ostensively, but rather it is defined indirectly by various techniques which have the effect of both allowing the grammar definition to capture significant generalizations, and making the grammar definition several orders of magnitude more compact than a simple listing of rules would have been. Crucially, GPSG employs (i) categories defined in terms of syntactic features, and (ii) an inductive definition of the set of rules in the grammar via linear precedence rules, rule schemata, metarules, and principles of feature instantiation. Here is how a GPSG grammar would represent the structure of the simple English sentence used as an example in our discussion of categorial grammar, above (adapted from Gazdar, Pullum and Sag 1982).



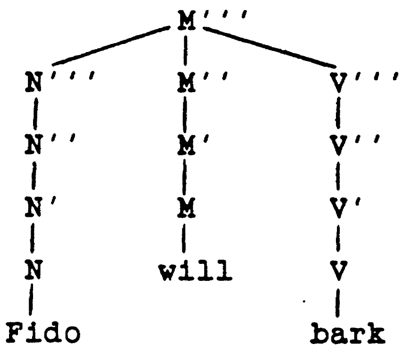
With respect to syntactic features, work in GPSG has shown how their use for subcategorization in a PSG can capture generalizations that had had to be stipulated in the classical transformational account. Furthermore, the use of features to handle the class of unbounded dependency constructions (e.g. relative clauses, wh-questions, topicalization, etc.) is able to capture as a theorem a generalization about their interaction with coordination that was never satisfactorily captured in transformational grammar (Gazdar 1981). Sells (1985) provides a useful introduction to GPSG, and to LFG, to which we turn next.

4.3 Lexical Functional Grammar

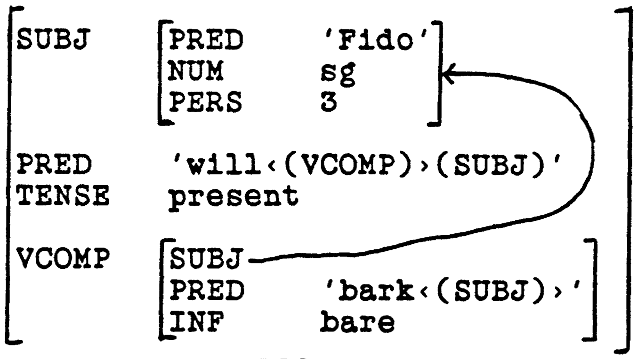
Unlike the two theories just sketched, lexical-functional grammar (LFG - Kaplan & Bresnan 1982) is similar to the classical transformational model in that it assumes two distinct levels of syntactic structure. One of these ("c-structure") is provided by a phrase structure grammar of the now familar kind. C-structures are thus trees. The other kind of structure, which is also a graph, though not a tree, is arrived at by solving various equations on the basis of the form of the c-structure. This second structural description is known as an "f-structure". Not every well-formed c-structure gives rise to a well-formed f-structure. The latter can therefore act as a kind of filter on the former. As with the "deep structures" of the classical transformational model, the f-structure is supposed to be semantically more transparent than its c-structure counterpart.

Here, by way of illustration, are the c-structure and f-structure that one LFG analysis (Falk 1984) would assign to the example sentence that we have used previously.

c-structure:



f-structure:



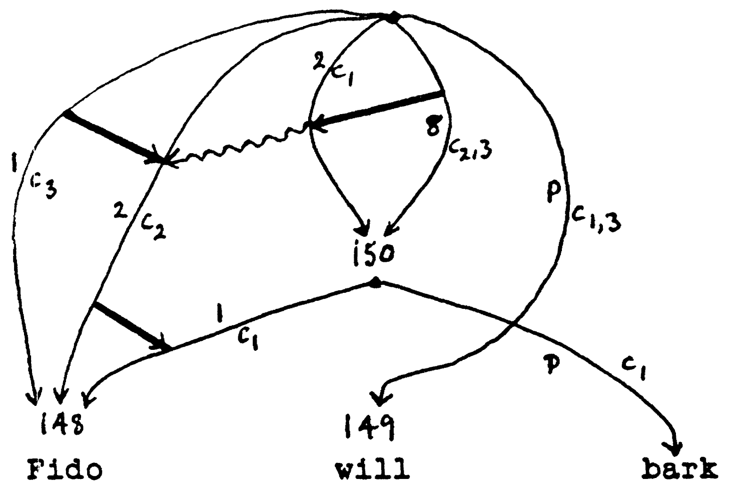
Unlike the other theories considered here, LFG is explicitly offered as a theory of the mental representation of language.

4.4 Arc Pair Grammar

One influential strand of thinking among sytacticians in the 1970's involved attaching much greater importance to traditional grammatical relations such as "subject of" and "direct object of" than had been customary in the transformational grammar of the 1960's (cf NHL 1, 128-29). A good deal of descriptive work of high quality was done, and continues to be done, within the framework known as Relational Grammar (Perlmutter 1983, Perlmutter & Rosen 1984). This framework takes the traditional grammatical relations as primitives. Arc Pair Grammar (APG) is a formal theory developed by Johnson and Postal (1980) which expresses mathematically the insights of the work done in Relational Grammar.

APG differs significantly from the theories of generative grammar considered above in that it does not define well-formedness constructively. Instead of saying that permissible structures are just those which have been explicitly permitted on a piece-by-piece basis, as a constructive definition does, it simply permits every structure (within a particular mathematical space) except those which violate one or more of its rules. On this view, universal grammar is just that subset of rules that no language can violate (these universal rules are referred to as 'laws'), and a language-particular grammar is that set of rules that you need to add to the laws in order to forbid everything that is excluded from the language in question.

Like the other theories we consider below, APG uses graph-theoretic objects to describe structure, but, in sharp contast to them, most of the work is done by the arcs in the graph and little or no work is done by the category system. Something of the flavour, though not the substance, of APG can be inferred from the structural description of the English sentence *Fido will bark* shown below.



This says, among other things, that *Fido* is the (underlying) subject of *bark* and the (superficial) subject of the whole sentence, and that the object of *will* is the clause *Fido bark[s]*.

There is more convergence between the four theories we have sketched than will be apparent from their widely divergent use of notation. Here are some examples of that convergence. (i) The traditional relational notions like "subject of" that are fundamental to APG show up also in LFG, and they have been reconstructed semantically by work in the categorial tradition (Dowty 1982, 1983), and Dowty's work is simply presupposed by GPSG. (ii) The generalized categorial grammar treatment of unbounded dependencies is conceptually very similar to that in GPSG, whereas GPSG (e.g. in the work of Pollard 1984) has been moving toward an essentially categorial treatment of subcategorization (rather like that introduced in the toy Grammar2, above). (iii) LFG's use of a PSG to generate c-structures makes it, in effect, a kind of superset of GPSG. Furthermore, both theories make extensive use of extension and unification as discussed in the preceding section.

Although still a minority interest among syntacticians, generative grammar is probably in a healthier state today than it has been for more than two decades. And the current demand from computational linguistics for compact precise lucid syntactic formalisms along with comprehensive grammars employing them is likely to ensure its continued health for some time to come.

References:

Ades, Anthony E., and Mark J. Steedman. 1982. On the order of words. *Linguistics and Philosophy* 4. 517-558.

Bach. Emmon. 1979. Control in Montague Grammar. *Linguistic Inquiry* 10, 515-531.

Bach, Emmon. 1981. Discontinuous constituents in generalized categorial grammar. In Victoria A. Burke and James Pustejovsky, eds.. *Proceedings of the 11th Annual Meeting of the North Eastern Linguistic Society,* 1-12. Department of Linguistics, University of Massachusetts at Amherst.

Bach, Emmon. 1983. Generalized categorial grammars and the English auxiliary. In Frank Heny and Barry Richards, eds. *Linguistic Categories,* Volume 2. Dordrecht: Reidel.

Bar-Hillel, Yehoshua. 1953. A quasi-arithmetical notation for syntactic description. *Language* 29. 47-58.

Bresnan, Joan W. 1978. A realistic transformational grammar. In Morris Halle, Joan W. Bresnan and George A. Miller, eds. *Linguistic Theory and Psychological Reality.* Cambridge, Ma.: MIT Press, 1-59.

Chomsky, Noam. 1955/75. *The Logical Structure of Linguistic Theory.* New York: Plenum (written c. 1955. published 1975).

Chomsky, Noam. 1964. The logical basis of linguistic theory. *Proceedings of the 9th International Congress of Linguists,* Cambridge, Massachusetts 1962.

Chomsky, Noam. 1981. Knowledge of language: its elements and origins. *Philosophical Transactions of the Royal Society of London,* Series B 295, 223-234.

Chomsky, Noam. 1985. *Changing perspectives on knowledge and use of language.* Manuscript, MIT.

Comrie, Bernard. 1984. Language universals and linguistic argumentation: a reply to Coopmans. *Journal of Linguistics* 20. 155-163.

Dowty. David. 1978. Governed transformations as lexical rules in a Montague Grammar. *Linguistic Inquiry* 9, 393-426.

Dowty, David. 1982a. Grammatical relations and Montague Grammar. In Pauline Jacobson and Geoffrey K. Pullum. eds. *77ie Nature of Syntactic Representation.* Dordrecht: Reidel, 79-130.

Dowty, David. 1982b. More on the categorial analysis of grammatical relations. In Annie Zaenen. ed. *Subjects and Other Subjects: Proceedings of the Harvard Conference on Grammatical Relations.* Bloomington: Indiana University Linguistic Club.

Falk. Yehuda M. 1984. The English auxiliary system. *Language* 60. 483-509.

Flynn, Michael 1983. A categorial theory of structure building. In Gerald Gazdar, Ewan H. Klein and Geoffrey K. Pullum. eds. *Order, Concord and Constituency.* Dordrecht: Foris, 138-174.

Gazdar, Gerald. 1981. Unbounded dependencies and coordinate structure. *Linguistic Inquiry* 12. 155-184.

Gazdar, Gerald. 1982. Phrase structure grammar. In Pauline Jacobson and Geoffrey K. Pullum, eds. *The Nature of Syntactic Representation.* Dordrecht: Reidel. 131-186.

Gazdar, Gerald. Ewan H. Klein, Geoffrey K. Pullum, and Ivan A. Sag. 1985. *Generalized Phrase Structure Grammar.* Oxford: Basil Blackwell.

Gazdar. Gerald. Geoffrey K. Pullum. and Ivan A. Sag. 1982. Auxiliaries and related phenomena in a restrictive theory of grammar. *Language* 58, 591-638.

Harman. Gilbert. 1963. Generative grammars without transformation rules: a defense of phrase structure. *Language* 39. 597-616.

Hoeksema, Jack. 1984. *Categorial Morphology.* Groningen: Van Denderen.

Hopcroft, John E., and Jeffrey D. Ullman. 1979. *Introduction to Automata Theory, Languages, and Computation.* Reading, Ma.: Addison-Wesley.

Johnson, David, and Paul M. Postal. 1980. *Arc Pair Grammar,* Princeton: Princeton University Press.

Kaplan, Ronald M., and Joan W. Bresnan. 1982. Lexical-Functional Grammar: a formal system for grammatical representation. In Joan W. Bresnan, ed. *The Mental Representation of Grammatical Relations*. Cambridge, Ma.: MIT Press, 173-281.

Karttunen, Lauri. 1977. Syntax and semantics of questions. *Linguistics and Philosophy* 1, 3-44.

Kay, Martin. 1979. Functional grammar. In Christina Chiarello et al.., eds. *Proceedings of the Fifth Annual Meeting of the Berkeley Linguistics Society*, 142-158.

Kay, Martin. 1983. When meta-rules are not meta-rules. In Karen Sparck-Jones and Yorick Wilks, eds. *Automatic Natural Language Parsing*. Chichester: Ellis Horwood, 94-116.

Kay, Martin. 1984. Functional Unification Grammar: a formalism for machine translation. In *Proceedings of Coling84*, Menlo Park: Association for Computational Linguistics, 75-78.

Langendoen, D. Terence and Paul M. Postal. 1984. *The Vastness of Natural Languages*. Oxford: Basil Blackwell.

Lasnik, Howard, and Joseph J. Kupin. 1977. A restrictive theory of transformational grammar. *Theoretical Linguistics* 4, 173-196.

Lyons, John. 1968. *Introduction to Theoretical Linguistics*. Cambridge: Cambridge University Press.

Lyons, John. 1970. Generative syntax. In John Lyons, ed. *New Horizons in Linguistics*. Harmondsworth: Penguin, 115-139.

Montague, Richard. 1970a. Universal grammar. *Theoria* 36, 373-398. Reprinted in Richard Montague, 1974. *Formal Philosophy* (edited by Richmond Thomason). New Haven: Yale University Press. 222-246.

Montague, Richard. 1970b. English as a formal language. In Bruno Visentini et al., eds. *Linguaggi nella Societa e nella Tecnica*. Milan: Edizioni di Comunita, 189-224. Reprinted in Richard Montague, 1974. *Formal Philosophy* (edited by Richmond Thomason). New Haven: Yale University Press. 188-221.

Montague, Richard. 1973. The proper treatment of quantification in ordinary English. In Jaakko Hintikka, Julius Moravcsik, and Patrick Suppes, eds. *Approaches to Natural Language: Proceedings of the 1970 Stanford Workshop on Grammar and Semantics*. Dordrecht: Reidel, 221-242. Reprinted in Richard Montague, 1974. *Formal Philosophy* (edited by Richmond Thomason). New Haven: Yale University Press. 247-270.

Newmeyer, Frederick J. 1980. *Linguistic Theory in America*. New York: Academic Press.

Partee, Barbara H. 1975. Montague grammar and transformational grammar. *Linguistic Inquiry* 6, 203-300.

Partee, Barbara H. 1979. Montague grammar and the well-formedness constraint. In Frank Heny and Helmut S. Schnelle, eds. *Syntax and Semantics 10: Selections from the Third Groningen Round Table*. New York: Academic Press, 275-313.

Pereira, Fernando C.N., and Stuart M. Shieber. 1984. The semantics of grammar formalisms seen as computer languages. In *Proceedings of Coling84*, Menlo Park: Association for Computational Linguistics, 123-129.

Perlmutter, David M., ed. 1983. *Studies in Relational Grammar 1*. Chicago: Chicago University Press.

Perlmutter, David M., and Carol G. Rosen, eds. 1984. *Studies in Relational Grammar 2*. Chicago: Chicago University Press.

Peters, P. Stanley, and Robert W. Ritchie. 1973. On the generative power of transformational grammars. *Information Science* 6, 49-83.

Pollard, Carl. 1984. *Generalized Phrase Structure Grammars, Head Grammars, and Natural Language*. PhD dissertation, Stanford University.

Pullum, Geoffrey K., and Gerald Gazdar. 1982. Natural languages and context-free languages. *Linguistics and Philosophy* 4, 471-504.

van Riemsdijk, Henk. 1982. Derivational grammmar vs. representational grammar in syntax and morphology. In Linguistic Society of Korea, ed. *Linguistics in the Morning Calm*. Seoul: Hanshin, 211-231.

Sells, Peter. 1985. *Lectures on Contemporary Syntactic Theories.* Stanford: CSLI.

Steedman, Mark. 1985. Dependency and coordination in the grammar of Dutch and English. *Language* 61, 523-568.

Stucky, Susan U. 1983. Metarules as meta-node-admissibility conditions. *Technical Note* 304. AI Center, SRI International. Menlo Park.

Thomason. Richmond. 1976. Some extensions of Montague Grammar. In Barbara H. Partee, ed. *Montague Grammar.* New York: Academic Press, **75-117.**

van Wijngaarden. Adriaan. 1969. Report on the algorithmic language ALGOL68. *Numerische Mathematik* 14, 79-218.