COMPILER GRAMMAR:

A DEPENDENCY-ORIENTED MINIMALIST APPROACH

Anat Ninio

Department of Psychology,

The Hebrew University, Jerusalem, Israel

*Theoretical Linguistics*, 1995, *21*, 159-195.

Address for correspondence: Anat Ninio, Department of Psychology, The Hebrew University, Jerusalem 91905, Israel.

E-mail address: msninio@pluto.mscc.huji.ac.il

ABSTRACT

This article presents the outlines of a procedural model of syntax. Syntax is likened to a compiler that translates computer programs into machine-interpretable language. The basic combinatory operations of the linguistic system are claimed to be three combinatory options existing for logical predicate-argument couples, namely, functor-evaluation, argument-modification, and triadic relations, equivalent both to Combinatory Logic's primitive atomic combinators and to the basic grammatical relations posited by traditional grammar. The syntactic relation of dependency is shown to represent the direction of information unification involved in the three types of combination. The grammatical devices of natural languages are likened to commands calling up this limited set of basic combinatory operations in the central language processor.

**1. Minimalist grammars**

This article presents the outlines of a procedural model of syntax. Even though it is situated in the dependency-grammatical tradition, it can be regarded as carrying out Noam Chomsky's "minimalist program for syntactic theory". In the paper bearing the same name, Chomsky calls for the elimination of previously-adopted multiple structural representations, each representing a different linguistic level, and for a reconceptualization of linguistic expressions as "a complex of instructions for [the] performance systems, providing information relevant to their functions" (Chomsky 1992:2). The aim of the "minimalist program" is to ultimately define all grammar as consisting of "interface levels" with, respectively, the articulatory-perceptual and the conceptual-intentional performance systems.

Chomsky's novel program is reminiscent of previously proposed procedural grammars, all aimed at doing away with levels of syntactic structural representations, for instance, of Lakoff and Thompson's Cognitive Grammar (1975a, 1975b) or of Crain and Steedman's (1985) proposals. According to Lakoff and Thompson, for example, rules of grammar *are* processing strategies of a certain sort. Abstract grammars are convenient fictions for representing certain aspects of linguistic processing, and have no independent reality (1975b:337). Similarly, Crain and Steedman claim that the rules of syntax do not "describe a class of structures that are *built*"; rather, they "describe what a processor *does* in assembling a semantic interpretation" (1985:323, italics in the original). In their approach, as in Chomsky's, the radical proceduralism consists of doing away with syntactic structures; in particular, semantic information is assembled directly from the sentence, without first creating intermediate syntactic representations.

The present proposal belongs to this family of 'minimalist' grammars, representing a version with a basic Dependency Grammatical orientation. Its main 'parent' is Hudson's Word Grammar (1984, 1990) whose mono-stratal, word-based, lexical approach is the foundation on which the present model rests. In addition, the model is deeply influenced by mathematical logic, both in its Lambda-calculus and Combinatory Logic versions, and thus is allied also to Categorial Grammars. Lastly, it incorporates the basic approach of Unification Grammars, viewing syntax and semantics as procedures involving the unification of partial information-bearing structures. The final mixture is the fruit of an effort to accommodate the insights of these different approaches under a common roof.

**2. Syntax as a compiler**

The starting point of the present approach is the conception of language as a medium of information transmission. In speaking, speakers convey to their addressees a complex unitary information entity, namely, sentence-meaning.[[1]](#endnote-2) Natural languages do not, as a rule, encode sentence-meanings in pre-existing fixed codes. Rather, the code carrying sentence-meaning is synthesized *ad hoc* in the form of multi-element strings of lower level information-carrying units. Syntax is the method by which a unitary piece of information is packaged in a multi-element code.

It is obvious that such an encoding method must involve some process of systematic information-splitting among the discrete elements of the code. Conversely, there has to be a process of systematic information-combination through which the complex whole of sentence-meaning is reassembled from the separate elements of the code. These two processes must be complementary or the inverse of each other. As Rommetveit (1974:55) pointed out, information is packaged into the sentence on the principle of "anticipatory decoding".

The characterization of these processes is at the core of a syntactic theory. The first question is whether the methods of information-splitting and combining employed in different natural languages are identical or language-specific. Cross-linguistic evidence is, at first glace, contradictory. On the one hand, the number of different combinatorial operations seeming signalled by the grammar of a given language is very high, and these appear to differ considerably from language to language. However, there are also indications that there exists a set of basic information-combining operations employed with all languages, universal as well as highly limited in number.

In previous solutions offered to this paradox, the presence of language universals has been a motivation for the positing of a level of deep syntactic structure doubling the surface structure of sentences. Prominent examples are the pre-minimalist Chomskian systems such as Government-Binding (Chomsky 1981) or Mel'cuk's (1988) Meaning-Text Model. In the system described in this article, a different solution is proposed. In short, the universal is attributed to the actual information-manipulating processes performed by the human language processor whereas the language-specific is attributed to variations in the terminology in which these basic processes are activated or 'called up' in different languages. In contrast to previous approaches, the duality of the system is not a motivation for positing multiple levels of structure but, on the contrary, language universals are the major means through which syntactic rules directly assessing central processing operations are established.

The central metaphor for the suggested theory of syntax is that of syntax as a compiler.[[2]](#endnote-3) A compiler, in its prototypical use, is a program that reads and translates a computer program written in a 'high-level' *programming language* such as Fortran or Pascal into an equivalent program written in the 'low-level' *machine-interpretable language* which is used to actually tell the computer what operations to carry out.[[3]](#endnote-4) For example, an assignment statement of the form Y := 5 + X may result, after compiling, in a series of zeroes and ones, something along the line of 0001 01 00000010, functionally equivalent to a series of commands for the computer to move the contents of some memory registers to other addresses, and to add them to each other. The crucial point for our metaphor is that whatever the form in which commands are written in a high-level programming language, they must be translatable to actual machine operations. Every high-level programming language is so constrained that the range of operations -- arithmetic or otherwise -- that can be expressed in its terms is within the abilities of the machine. A programmer is not allowed to request of a computer to ride a pony; allowed commands are either directly isomorphic to basic computer operations or else they are decomposable into ones. A compiler can only be written for a programming language which is actually translatable to machine language.

Similar to the situation obtaining for computers, in the present approach, instructions for information manipulation inherent in natural language sentences are seen as, ultimately, instructions for an addressee's language processor. The variability in grammatical operations apparently signalled by language-specific devices (e.g. Mel'cuk 1979) is that of the terminology of commands, not of the operations themselves. Whatever the language-specific grammatical relation, its functional significance is limited to a small number of basic information-combining operations.

This analogy suggests a strategy for the description of the grammar of natural languages as they would be in an idealized compiler or translation program. Rules of syntax are to be formulated as equivalences, identifying the basic processing operations called up by language-specific structure-imposing devices. Accordingly, syntax is seen as simultaneously language-specific and universal, using a common method of information-manipulation but accessing it in a variety of forms.

The attempt to establish syntax on the foundations of basic combinatory operations represents a bottom-up approach to linguistic connectivity. Accordingly, the starting point is the building blocks of sentences, namely, words.

**3. Words and the predicate-argument relation**

It is a truism that the meaning of sentences is assembled -- with the aid of syntax -- from the meaning of the individual words comprising the sentence. It is less obvious that words are inherent sentential components rather than independent meaning-carrying entities. Attempts by formal semanticists (e.g. Katz and Fodor 1963) to construct a theory of word meaning as a compositional entity built up from semantic primitives are judged not to have been successful (Kempson 1977). The consensus in the philosophical literature dealing with theories of meaning, unifying both sides of the truth-conditional *vs*. use-conditional debate, is that the primitive of the meaning-system is sentence or utterance-meaning, and that word meaning is to be defined as the systematic contribution of a word to the meaning of the sentences (or utterances) it is a component of (Allwood 1981, Alston 1964, Frege 1892, Quine 1960, 1967, Wiggins 1971, Ziff 1960).

Given this conception, it is possible to distinguish between two broad types of words according to how they achieve meaningfulness. Ever since Frege (1879, 1891) and Husserl (1900), it has been pointed out that some words 'make sense' on their own, independent of the rest of the sentence in which they occur, whereas others do not. The latter are felt to be 'unsaturated', of 'incomplete meaning', to possess 'an empty space' in them that requires completion by the meaning carried by other words. A prototypical example of the 'complete' class are proper names; of the latter, verbs, adjectives and adverbs. Words possessing 'complete' meaning are of a different logical type than words of the 'unsaturated' kind; whereas the former denote entities of some sort, the latter present second-level information about entities such as their properties, states or transformations, namely, they denote concepts defined in relation to other concepts. In consequence, words of the 'incomplete' type need to be supplemented by some other elements of the sentence, designating the entities they provide information on, in order to denote a 'complete' second-level concept. In informational terms, without a supplementary specification of the entities involved, the information encoded in these second-level words is not utilizable.

As an example, let us take the preposition *in*. This word denotes the positioning of an entity inside some space; it is impossible to define its meaning without mentioning both the entity and the space. The very concept of 'in-ness' is defined relative to these other two concepts, and not as an absolute notion that can stand on its own.

As the property of having a 'complete' or 'incomplete' meaning is a generalization over the relevant words' behavior in all linguistic contexts, it is to be seen as their fixed logical characteristic *qua* linguistic signs, a feature that belongs to the words as they are entered in the hypothetical mental lexicon.[[4]](#endnote-5)

The acknowledgement of logically 'incomplete' words forms part of most syntactic theories, variously recognized in concepts such as the "semantic valency" of words (Tesniere 1959, Allerton 1982); their "predicate-argument structure" (Chomsky 1981, Jackendoff 1983, 1990); "logical-functional structure" (Bresnan 1982a), and so forth. Consequently, I shall assume the existence of 'incomplete' words -- namely, *predicates*, *function words* or *functors* in standard terminology -- as consensual.

**4. Combinatory options for predicates and their logical arguments**

We have defined predicates as words carrying 'incomplete' concepts requiring to be completed in the sentence. It has been agreed ever since Frege first introduced the concept that the predicate's incomplete character stems from the fact that it is a word carrying second-level information *about* something but, by definition, it does not, by itself, denote that 'something'. In order to denote a 'complete' second-level concept, these 'incomplete' words have to be supplemented by some other word of the sentence, designating the entities they provide information on.

There is however another reason for the predicate concepts' incompleteness. Stated informally, second-order information 'about something' comes in several varieties; it may be a thing claimed to be true *of* that something, or a feature or component *of* that something; in either case the information can be said to be incomplete without knowing who or what that 'something' is. Until the predicate is used in combination with some other element of the sentence, it is undetermined in what sense the predicate-information is used to create a complete second-order concept. In formal terms, there may be several different ways in which predicates can be saturated by their logical arguments. The mapping of these options is the subject to which we turn now.

Cross-linguistic evidence suggests that there are three major alternative ways to saturate a predicate. First, the variable or indefinite semantic elements of a predicate may be *evaluated* by -- receive a constant value from -- their arguments. In this case, the combination of the functor and the argument results in a saturated functor. This combinatory option, resembling functional application and substitution, is exemplified by (1). The relevant predicate-argument pairs are underlined.

(1)a. John slept.

b. Fred hit John.

c. Fred hit John.

d. This letter is from John.

e. This letter is from John.

f. The lining of my topcoat is torn.

The examples show the prototypical syntactic patterns in English in which a tensed intransitive verb (1a), a tensed transitive verb (1b and 1c), a preposition (1d), a demonstrative pronoun (1e), and a noun (1f), receive saturation from words carrying values for the predicates' variable elements.

The second combinatory option is one in which the predicate concept becomes an *ad hoc* *component* of the argument-concept. The result of these combinations is a modified argument-concept rather than a saturated predicate-concept. (2) presents some examples.

(2)a. That tall boy is my cousin.

b. She always carries a backpack loaded with books.

c. The cat sitting on the keyboard is Persian.

d. The cat on the keyboard is Persian.

e. The temperature outside is freezing.

f. The friend I met yesterday lives in Australia.

g. The friend whom I met yesterday lives in Australia.

h. The friend I met yesterday lives in Australia.

i. She took off the wrapping carefully.

j. She took off the wrapping very carefully.

k. He was very tall.

(2a-d) involve the use of adjectives, nonfinite and finite verbs, and adverbs as the modifiers of the common nouns which are their logical arguments. (2g-k) present adverbs (which are usually considered second-order predicates) modifying verbs, adverbs and adjectives. It may be said that the predicate ends up inserted into the information-structure of the argument-word as a temporary and *ad hoc* novel feature.[[5]](#endnote-6)

The third combinatory option is a *triadic*, rather than, as the previous two, a diadic or binary pattern. The participants in the combination are two predicates and an argument-word; the two predicates possess co-referential variable elements which get set to the value carried by the argument-word.

Typically,[[6]](#endnote-7) the two predicates belong to different form-classes; one of them is a tensed verb or else one of a small set of special predicate-taking prepositions; the other one is invariably a secondary or untensed predicate, either a nonfinite verb (base, to-infinitive, participle, etc) or else an adjective, adverb, preposition or predicative nominal. For the latter class, the triadic pattern is the only option for getting the relevant argument evaluated.

(3) and (4) present some examples. The two predicates and the relevant argument are underlined. To simplify the discussion, to-infinitives are treated in these examples as a single word.

(3)a. John tried to sleep.

b. Mary promised Fred to talk to John.

c. Fred asked Mary to talk to John.

d. John seems to like Mary.

e. Fred believes John to like Mary.

f. Most people supposed him guilty.

g. John made Mary happy.

h. John saw Fred leave the party.

i. John came to respect Fred.

j. Fred was respected by John.

k. Mary was talking to John.

l. With Mary talking to John, Fred was bored.

m. It was unusual for John to talk to Mary.

(4)a. John read THE TIMES folded up.

b. John wrote the letter laughing at his own jokes.

c. John wrote the letter to please Mary.

d. John awoke to find Fred at the door.

e. John awoke sick.

Three points are worth noting. First, it is impossible to reduce these triadic patterns to two or three independent binary predicate-argument relations, and it is a mistake to analyze them in isolation. The fact that identity of reference exists between the arguments of two predicates is not a contingent or accidental feature but part of the intended logical structure of the combination.

Second, (3) and (4) represent two different types of triadic relations, even though in both the two predicates accept the 'shared' argument as an evaluator. The difference resides in the kind of relationship between the two predicates; in (3) it is evaluative, in (4), modifying. This parallels the distinction between the two binary relations defined earlier.

In (3) the triadic relation is an a priori, lexically established pattern. The 'higher' predicate has the 'lower' predicate as its own logical argument; that this lower predicate should have a coreferential argument with the higher predicate is part of its specification *as* an argument of the higher predicate. For example, in (3c):

(3)c. Fred asked Mary to talk to John.

the verb *asked*, which is a predicate reporting on a speech act, takes three logical arguments: the person(s) who performed the speech act of asking; the person(s) who were the addressee(s) of the speech act of asking; and the action asked of the addressee(s). The last argument, here having the value of "to talk", is thus explicitly defined as a predicate denoting an action whose 'actor' argument is identical-referential to the 'addressee' argument of the main predicate *asked*.

As the variety of the examples in (3) demonstrates, the pattern represented by *asked* is not the only lexically established triadic pattern possible. For example, in 'raising' patterns (3d-e) the higher predicate may by definition possess a first argument identical to the logical first argument of the lower predicate, whereas in equi-type patterns (3a-c) the logical argument of the lower predicate is by definition identical to that of the higher one (see also Pollard and Sag 1987, Steedman 1988). The particulars of these phenomena are beyond the scope of the present article. For the moment, it suffices to point out that in all these cases, the triadic logical relation and its details are part of the lexically established information content of the higher predicate.

In (4), the higher predicate does not lexically require a predicate-type completor. When a second and uncoordinated predicate appears in the sentence, it is given a modificatory role with respect to the higher predicate *precisely* because it is interpreted as providing further information on one of its arguments. Very often this concerns some action or state of the argument-entity which is temporally or causally related to the focal event, either concurrent with it (4a,b), immediately following it (4d,e), or whose attainment is the reason for it (4c). This information is construed as modificatory information on the event denoted by the main predicate when the modification concerns specifically one of the event's participant entities (see also Goldenberg 1985). Thus, it can be said that the secondary predicate's logical argument is the main predicate's logical *argument slot*, rather than the *argument word* that carries the value for both slots.

Compare, for example, the triadic 'concurrent-action' pattern (5a) with the non-triadic full clause introduced by *while* in (5b):

(5)a. John wrote the letter laughing hysterically.

b. John wrote the letter while Fred was laughing hysterically.

Whereas the predicate *laughing* of the full *while*-clause in (5b) can have any entity as its first argument, the one in (5a) must refer to one of the arguments of *wrote*, in this case to the writer "John". Which of the higher predicate's arguments is meant and what exactly is predicated of it is largely a matter of relative semantic congruity (see the ambiguous (4a)); the sole requirement is that the secondary predicate should be interpretable as relating to *one* of the higher predicate's arguments. Translated to formal terms, this means that the lower predicate will accept as the evaluator of its own first argument whatever is the current content of one of the higher predicate's argument slots.

The modifying triadic relations in (4) should not be confused with lexically specified 'controlled adjunct' patterns, whether depictive (3f) or resultative (3g). In the latter two, the higher predicate a priori defines an evaluative relation between its second argument and the lower predicate, and both are its obligatory arguments. It is possible to demonstrate that in the case of 'intransitive resultatives' such as (6a), the pattern represents the insertion of a temporary causative element into the semantics of a verb, which generates an *ad hoc* lexical triadic relation, rather than an *ad hoc* modificatory one. For instance, it is impossible to omit the lower predicate (6b), as in a lexical resultative (6c) (except if the meaning changes), but not as in an attributive one (6d). Incidentally, it is not possible to omit the temporary second argument either (6e); they can only be omitted together (6f):

(6)a. They laughed themselves sick.

b. They laughed themselves \*(sick).

c. John made Mary \*(happy).

d. John read THE TIMES (folded up).

e. They laughed \*(themselves) sick.

f. They laughed (themselves sick).

Thus intransitive resultatives pattern themselves on ordinary lexical causal verbs, generating two *ad hoc* logical arguments simultaneously (but *cf*. Carrier and Randall 1992). Such sentences provide a validation for the triadic analysis proposed above.

Lastly, it is apparent that not every argument of a (lower) predicate can be 'shared' with a higher predicate and thus form the basis of a triadic relation. For example, in (3h), here reproduced as (7a), the predicate *leave* shares with *saw* the logical argument denoting the entity performing the action of leaving. (7b) demonstrates that it is impossible to form this kind of triangular relation using the other argument of this predicate, namely, the thing that was left.

(7)a. John saw Fred leave the party.

b. \*John saw the movie Fred leave.

c. John saw the movie Fred left.

The untensed base form *leave* is blocked here, whereas if the tensed verb form *left* is used (7c), the result is a relative clause, modifying the argument *the movie* rather than the higher predicate *saw*, and thus representing not a triadic but a diadic relationship.

This regularity allows us to define formally a distinction we have been making in an intuitive and informal way, namely, between the *first argument* and the *second* (or third) *argument* of a predicate. We shall define as the first argument of a predicate as that logical argument which can be 'shared' with a higher predicate in a triadic relationship. It is immediately obvious that the active and passive forms of given predicate stem have different first arguments. For example, in (3j), repeated here as (8a), the argument of the passive participle *respected* which is 'shared' with the tensed *was* is the person/entity respected:

(8)a. Fred was respected by John.

b. Susan expected John to respect Fred.

By contrast, as (8b) demonstrates, in active forms of the stem *respect*, e.g. in its active infinitive form, the argument 'shared' with a higher predicate is the person doing the respecting, not the one respected.

The distinction between first and other arguments of a predicate allows us to differentiate -- from the bottom up -- between two subtypes of binary functional evaluation; one involving the evaluation of the 'first argument', the other, the evaluation of a 'second' or 'third', etc, argument of the same predicate. It has been suggested that this ordering represents the sequence in which the relevant arguments are evaluated; typically, a 'first argument' is evaluated *last*, after the predicate's other arguments are made definite (Dowty 1982, Hudson 1990, Pollard and Sag 1987, but *cf*. Lambek 1958, Steedman 1988). This suggests that combinatory operations involving the same predicate may be marked for relative ordering; at the very least, the evaluation operation of first arguments may be provided with a 'wait-command' according to which this argument is evaluated only when the predicate is otherwise saturated.

In principle, every predicate can combine with every logical argument of it in all three or four types of combination. In actuality, however, languages typically restrict the combinatory range of certain type of predicate words by creating lexically distinct versions of them, each covering a different subset of the complete range of combinatory possibilities. Thus, English uses several different forms for verbs -- the base, present, and past participle forms -- specializing in varieties of complementation, attribution, and triadic combinations. The English adjective class is restricted to either attribution or triadic -- secondary -- complementation; most members of this class can participate in both, except for small subgroups that can only attribute or only serve as predicative complements. Japanese predicative and attributive adjectives, by contrast, are lexically distinct and nonoverlapping. There are considerable cross-linguistic differences in the degree of specialization of predicate words; this applies also to the predicate properties of common nouns and the boundaries between them and more prototypical predicates such as verbs and adjectives (Schachter 1985). Whereas the restrictions on the combinatory options of predicates are heterogeneous and language-specific, the existence of multiple combinatory options for the same words is universal; e.g. most known languages allow the relativization of verbs by which they become the attributors of their logical arguments (Keenan 1985). As the basic phenomenon, then, we shall assume combinatory freedom, and leave the restrictions to be specified by the language-dependent component of the grammatical system.

In summary, we have identified three main types of predicate-argument combinations, two binary combinations and a triadic pattern with two subtypes corresponding to the binary ones:

A. *Functor-evaluation*, in which the predicate receives the argument as a provider of definite value for one of its variable elements. The outcome of this combination is the saturated predicate. There were also indications that functor-evaluation of the 'first argument' is a distinct subtype.

B. *Argument-modification*, in which the predicate becomes an *ad hoc* component of the argument-concept. The result of these combinations is a modified argument-concept rather than a saturated predicate-concept.

C. *Triadic relations*, in which two predicates, one 'higher', the other 'lower', possessing co-referential variable arguments, combine with a single argument-word that evaluates both variables. The overall triadic relation that results can be one of the following two:

C1. *Triadic relations with predicate evaluators*: If the higher predicate possesses a variable slot for the lower predicate, the outcome of the combination is two predicates one of whom is the argument of the other, with co-referential argument slots that get evaluated identically by the 'argument word'.

C2. *Triadic relations with predicate modifiers*: If the higher predicate does not possess a variable slot for the lower predicate, the result is a higher predicate receiving the lower predicate as its modifier, while still both predicates get their co-referential argument slots evaluated identically by the value carried by the 'argument word'.

This taxonomy of basic predicate-argument combinatory options immediately brings to mind two other suggestions regarding basic combinatory processes, one specifically linguistic, the other, abstract or logical. The first is an older grammatical tradition established by Becker (1841) and still evident in various guises in current linguistic theorizing and practice,[[7]](#endnote-8) according to which all syntactic structure is created by the recursive iteration of just three basic grammatical relations, the *predicative*, *complementative* (or *objective*) and *attributive*.[[8]](#endnote-9) The parallel to our basic diadic combinations for predicates and arguments is striking; the latter appear to describe the logical basis underlying the hypothesized grammatical relations which, it should be emphasized, are motivated by morphological and syntactic, rather than logical or semantic, regularities.

The second suggestion regarding the existence of a finite set of basic combinatory operations comes from Combinatory Logic, the branch of mathematical logic concerned with the properties of abstract combinatory systems. In its two parallel versions, Lambda-calculus (Church 1932/33, Barendregt 1985) and Combinatory Logic (Curry 1930, Curry and Feys 1958, Curry, Hindley and Seldin 1972, Schonfinkel 1924), it has identified a minimal set of three basic operations by the means of which any and all logical combinations can be generated. The first is the primitive binary operation of *application*, through which the applying term that contains a variable element receives a value for it from the argument-term to which it applies. It is evident that the combinatory operation we have identified in the linguistic system by the name of "functor-evaluation" is a particular concrete instance (or interpretation) of the abstract operation of "application".[[9]](#endnote-10)

The second basic binary operation is a variant of application in which the applying term does not contain a variable element for which the value of the argument can be substituted. In the formalization of Combinatory Logic this is an operation of the type **K**, after the constant logical combinator **K** that generates it.[[10]](#endnote-11) In parenthesis, a combinator, in this system, is the means by which combinations other than application are formed (Curry and Feys 1958:263). The combinator **K** is used when a constant *C* is to be expressed as a function of *x*, applying to some argument *A*. Formally, the result of this combination is the applying element returned as constant; the argument is 'cancelled'. However, it is possible to interpret the operation **K** as an operation transforming the constant term to which it applies into a function with a fictitious argument inserted into it. This is the interpretation of **K** suggested by the two founders of Combinatory Logic, Curry (1930) and Schonfinkel (1924; see Curry *et al*. 1972:10-11). Under this interpretation, a **K**-type combination inserts the term for which there had not been a variable slot, into the constant that applies to it, as the value for its fictitious argument. This abstract operation appears to model the linguistic operation we have defined as *argument-modification* and which may be identified with the grammatical relation of attribution.

The third type of basic combinatory operation identified by Combinatory Logic is of the type **S**, the *distributory combination*. This combinator takes three arguments *M*, *N*, and *L*, and creates a combination in which *L* is substituted for a variable of both *M* and *N*. Intuitively, **S** 'distributes' the value *L* between the two members of an applicative relation *M* and *N*, so that both *M* and *N* apply to *L*. The affinity to our triadic combinations is obvious; **S** forms a triadic relation between the three terms it applies to. The first subtype of the triadic relation represent cases when the first term, *M*, can be applied to the second, *N*, namely, cases with predicate evaluators (examples (3)). When *M* cannot be applied to *N* because *M* does not possess a variable slot for *N*, it accepts *N* in a **K**-type pattern; these are examples (4) with predicate modifiers.[[11]](#endnote-12)

In summary, it appears that the three types of basic combinatory operations on predicates and arguments -- the functional-evaluative, the argument-modifying and the triadic -- consist of possible 'interpretations', in the linguistic system, of the three primitive types of logical combinations -- application, **K**-combination and **S**-combination -- identified as constitutive of combinatory systems in the abstract.[[12]](#endnote-13)

The convergence with these two approaches -- the grammatical relations tradition and the abstract logical analysis -- is supportive of the claim that the basic combinatory operations of predicates and arguments we have identified are indeed primitives of language in its capacity as a combinatory system. As such, they are thought to underlie the generation of all syntagmatic connectivity in language.[[13]](#endnote-14) Linguistic evidence bears the centrality of the distinctions out: For example, the difference between complementation and the adjunct relation plays a central role in such diverse phenomena as binding and extraction (see Hudson 1990).

**5. Information unification and dependency**

According to the results of the last section, in principle a given predicate word can be saturated by its lexically determined logical argument in one of three different ways, the functor-evaluative, the argument-modifying, and as part of a triadic relation. As their names imply, the three operations differ as to the logical status of their results. The outcome of functor-evaluation is the saturated predicate; the outcome of argument-modification is a modified argument for which a novel feature is instantiated; the result of a triadic combination is the higher predicate either evaluated by or modified by the lower predicate, with the argument word evaluating the relevant variable element of both. If the logical status of the combination relative to that of its components is translated to actual information-manipulation terms, the possibility arises that the three combinatory operations describe three different information-unification processes.

In functor-evaluation, information combination starts with an unsaturated predicate and with an argument-word, and results in a saturated predicate. The most parsimonious description of this process is that the information content of the argument-carrying storage unit is, as it were, 'moved into' the predicate-carrying unit, where it is used to evaluate the relevant indeterminate element.[[14]](#endnote-15)

In argument-modification, information combination starts with an unsaturated predicate and with a word serving as its logical argument, and results in a modified argument word. The most parsimonious description of this process is that the information content of the predicate-carrying storage unit is 'moved into' the argument-carrying unit, where it is inserted or inscribed as an *ad hoc* novel element. The content of this novel feature of the argument-word is "argument *x* of the predicate *F*". In the notation of Lambda-calculus, the predicate's content can be represented by a lambda-expression of the form **λ***x*.*F*(*x*), namely, an expression specifying the bound variable *x* as a component of *F* with a missing value. Intuitively, this expression may be read either as "*F* of *x*" (a term *F* with a labelled variable element), or as "*x* of *F*", namely, a variable defined in terms of its being a labelled component of the term *F*. It is with the latter 'reading' that the predicate's content is inserted into the argument-word.[[15]](#endnote-16)

Analogously, in triadic relations, the content of both the argument and of the lower predicate is 'moved into' that of the higher predicate. The content of the lower predicate serves either to evaluate an existing indefinite predicate-slot of the higher predicate or to insert a novel component into the predicate. Simultaneously, the lower predicate itself gets evaluated by the same argument-unit that evaluates the higher predicate's argument slot.

According to this conception, then, the unification of information in predicate-argument combining operations is a *directional* process, where the one of the combining units serves as the recipient or *host* of the combination, and the other combining unit or units are the contributors 'unified into it'. As the result of unification, only the host, now containing the combined information, survives; the contributor units which are 'unified into' it are, functionally speaking, 'destroyed' and cease to function as independent units.

This conception of the information-unification process is somehow different from the one currently in use in the literature (see for instance Gazdar, Klein, Pullum, and Sag 1985, Pereira and Shieber 1987, Pollard and Sag 1987, Shieber 1986, Zeevart, Klein and Calder 1987). Fundamentally, unification is a hypothetical procedure (with mathematical roots in algebra) for uniting two information-carrying records in case they appear to carry information on the same object-token. The procedure consists of checking if two records are or are not contradictory; in case of compatibility, namely, if they appear to be containing information on the same object, the information in the two are pooled to a novel information frame containing the combined contents of the two.

Unification as a structure-building operation is said to be destructive. Namely, information frames or subframes used in a unification operation are not available any more for further unifications. This procedure is thought to be the sole and universal information-combining operation, and, indeed, the sole means by which syntactic structure is created by the language processor.

While the role of unification in creating syntactic structure is not disputed, our proposal diverges from the above in one detail: it is not accepted that unification is destructive for both of the original records. This particular element of the conceptualization of unification has been influenced by the adoption of a configurational linguistic framework in the field of natural language processing: Unification is seen as the means by which higher-order linguistic units, namely phrases, clauses and sentences, are formed from a combination of lower-order ones. In a typical configurational approach, a higher node on a syntax-tree is conceived as *replacing* its daughters; moreover, in the rules of syntax regulating the substitution of phrases for their constituents (i.e. phrase-structure or 'rewrite' rules), these constituents are mentioned in a symmetrical way. That there may be a basic asymmetry in phrase-formation is acknowledged by *x*-bar theory and its introduction of concepts like head-projection and head-feature (Jackendoff 1977; and see also Lyons 1968); however, the relevant concepts are not integrated into the module of the grammar dealing with substitution rules, whose symmetrical treatment of daughters is still reflected in the unification formalisms used in the relevant theories (e.g. Pollard and Sag 1987:55) as well as in the natural-language unification literature in general (e.g. Pereira and Shieber 1987). In other words, head daughters as well as non-head daughters are said to be unified into the phrasal projection of the head in an identical destructive manner.

However, in a theory-neutral approach, there is no sound motivation for assuming that the host of a unification is also, functionally, destroyed. On the contrary: the undisputed fact of the inheritance of syntactic and semantic features of the host by the 'combination' suggests that the positing of a novel record replacing the host is redundant and unmotivated.[[16]](#endnote-17) Instead of a symmetrically destructive process, therefore, a directional unification process is posited in which one of the combining objects, the host, survives to represent the combination whereas the others, united into it, cease to exist functionally. Instead of creating novel information-structures, unification is thought to temporarily modify the information-structure of the host-word, whether by exchanging a definite value for one of its originally indefinite arguments or by the insertion of a novel subordinate entry in its frame.

Returning to predicate-argument combinations, it appears that the basic difference between the three type of combinations is which storage-unit serves as host to the combination. In binary functor-evaluation, it is the predicate; in argument-modification, it is the argument; in triadic relations, it is the higher predicate which is the host for all three predicate-argument combinations involved. Being the host of a combinatory operation is equivalent to determining its logical status; the two concepts are derived from each other.

The property of playing host to a combination is allocated to one of the combining units on the basis of the type of combination in which they participate. Namely, a predicate-word does not necessarily become the host of a combination involving it and one of its logical arguments; the outcome depends on whether the relevant combination is one of functor-evaluation, of argument-modification, or a triadic relation involving a higher predicate. That, however, is a contingent fact determined by the logical-semantic goals of the particular sentence. If the speaker intends the predicate and the argument to form a functor-complementing relation, the predicate will receive the argument; if however the goals of the user are to create a modificatory relation between them, the predicate will reverse-unify into the argument, turning it into the host. If a triadic relation is aimed at, the predicate and its argument will both be moved into the home-ground of a common host in the form of a higher predicate.

Being the host of a predicate-argument combination is thus an emergent property, jointly determined by the inherent logical characteristics of the combining units and the logical goals of the combination. In other words, this is an outcome of the combination, not one of its initial givens.

Given this definition, it is somehow surprising that the property of being the host of a predicate-argument combination can be demonstrated to be identical to what is usually called the property of being the *head* (regent) of a government or dependency relation.

The relation of *government* (or its inverse, *dependency*) is an asymmetrical relation claimed to exist between pairs of words of a sentence (e.g. Mel'cuk 1979, Hudson 1984, 1990, Zwicky 1985). It is a function, namely, a many-one mapping relation which is directional (or antisymmetrical), antireflexive, and antitransitive. The application of this ordering relation to the words of a well-formed sentence creates a structure represented by a directional graph of the tree type; it is a two-dimensional abstract object, totally connected; it has a single root and no cyclicity (Tutte 1984). In other words, every word of a sentence is said to have a head (or regent), and exactly one,[[17]](#endnote-18) except for the root (the highest word) of the sentence which has none; a single regent can have a theoretically unlimited number of dependents.

According to dependency theory, the transferral from the two-dimensional tree-structure of a sentence to its one-dimensional phonetic chain-structure is regulated by a single rule, that of adjacency or projectivity (Mel'cuk 1988, Robinson 1970): a dependent must appear in a sentence immediately adjacent to its head except that the two may be separated by dependent(s) of either words. This rule is applied recursively, so that if the inserted dependent has a dependent of its own, the latter may in turn be inserted between its own head and *the head's* head. These two principles (of acyclic connectivity and projectivity) are thought to be completely general features of all syntactic systems in all languages.

The positing of the dependency relation between words is motivated by several interrelated phenomena of syntactic and semantic control of one word by another. The crucial phenomena are of two kinds: first, the governor is said to control the occurrence, features, semantic contribution, sometimes lexical form, as well as the positioning of its dependents in the sentence. Second, the head is said to determine the syntactic and semantic features of the head-dependent combination. As a rule, the combination is said to inherit the features of the head, such as its form-class classification and its semantic type. According to theories positing a dependency relation, the head-features of words are lexically given; these features go under the name of syntactic valency (see also Allerton 1982, Pollard and Sag 1987).

There are, however, problematic exceptions to the criteria defining headhood. Not surprisingly, they involve either words standing in a modificatory relation or else forming triadic structures. First, neither the occurrence nor the semantic contribution of adjuncts is determined by the head of the relevant combinations; it seems as if all adjunct relations are exceptions to the first set of criteria defining the dependency relation. In addition, except for a few not very successful efforts,[[18]](#endnote-19) it is universally recognized that heads do not subcategorize for adjuncts. In consequence, theories using the concept of lexical head-features for the establishment of syntactic structure find it problematic to deal with government relations involving adjuncts. For instance, Hudson (1987:124) equates "semantic functor" with "head", even though this is obviously not the case for adjuncts (see also Pollard and Sag 1987:157-168).

Second, the definition of syntactic valency as a lexically given, fixed characteristics of words creates grave difficulties when the relevant word participates in a modificatory or triadic relationship. For example, Hudson's Word Grammar (1990, 1992) has to posit that the finite verb of a relative clause is both the head and the dependent of the relative pronoun, or that the subject of a copular sentence is the dependent both of the copula and of the predicate complement of the copula. These solutions create mutual dependencies or multiple heads for a single dependent; positing them conflicts with the fundamental definition of the dependency relation as a function creating an acyclic tree structure over the sentence. The same multiple-relations problem exist for Lexical-Functional Grammar's treatment of *x*-complements, see Bresnan (1982b). In a yet different tradition, namely Categorial Grammar, the same problem surfaces in the form of a double categorial classification of adjectives; e.g. Moortgat (1988) categorizes adjectives as N/N when they are used attributively and as the syntactic primitive AP when used as predicate complements. What Moortgat does not touch on is that his system now needs a set of syntactic rules to determine when is the very same word in the lexicon is an AP and when it is an N/N.

According to the present proposal, the relative dependency status of two words that combine is not a lexically determined characteristics of either, but rather, a reflection of their roles in the current combinatory operation. When a predicate receives a value in an applicative operation, it is the host of the operation and it is thus the head of its logical argument; when however a predicate modifies its own logical argument in an attributive operation, the argument is the host and thus the head of the combination. Being the head of a dependency relation is thus equivalent to being the host of the relevant predicate-argument combination; the head is the designation of the object that survives the unification. The dependency relation is nothing but the *direction of the unification* of the relevant predicate-argument couples. "Head" and "dependent" are names for the two asymmetrical roles in combinatory operations.[[19]](#endnote-20)

This conceptualization of what the dependency relation consists of, accounts for the two sets of features associated with heads as well as the problematic exceptions in case of adjuncts and triadic relations. First, the head is by definition the unit determining the syntactic and semantic status of the combination; that is how it was defined. Second, the head controls the occurrence and semantic contribution of the dependent only if the head happens to be a predicate and the dependent a logical argument of it; namely, when the combination is applicative. If however the combination is attributive and the head is the logical argument of its dependent, it is the dependent predicate (e.g. an attributive adjective) that determines its own semantic contribution as well as its own occurrence in the sentence. In other words, whereas the head is always -- by definition -- the determiner of the features of the combination, it is the predicate component of the combination, whether head or dependent, that controls its own semantic features and the semantic role it allocates to the logical argument. The two sets of features defining headhood are thus seen to diverge; one set is an emergent characteristic of words as hosts of combinations, others are the inherent features of certain words by virtue of representing second-order or incomplete concepts.

Regarding triadic combinations, we have seen that the only head emerging in such combinations is the higher predicate. The lower predicate, whether evaluative or attributive, gets absorbed into the higher predicate and does not host a combinatory relation. It follows immediately that the lower predicate of triadic relations does not achieve the status of the head of a dependency relation, whatever that implies: e.g. it does not generate an independent 'arrow' to its logical argument in a dependency-tree; and it does not determine the placement of its logical argument in the sentence-string as the head of an evaluating argument would. Its relationship with its logical argument is essentially indirect, through its host the higher predicate; despite interesting differences between subtypes of the triadic patterns, the best generalization is that the lower predicate cannot receive a *word* as its logical (first) argument, but rather, its argument is by definition its co-dependent, the *argument slot* of the higher predicate.[[20]](#endnote-21) Thus, the identification of the dependency relation with the host-contributor relation solves the problem of the excess structure supposedly generated by triadic relationships.[[21]](#endnote-22)

Defining dependency as the host-contributor relation provides a clear view of its role in the generation of the unified meaning of a sentence. The dependency-tree of a sentence represents the path of the gradual accumulation of word-information from the bottom up, specifying at each stage which of the participant words serves as the host of the relevant unification. As information accumulates, as it were, 'in' the records of words serving as host, a word participating in a given unification already 'contains' all the structured information of its dependents. At the end of the process, the complete information content of the sentence is unified into the highest word, namely, the root of the sentence. Obviously, the converse of this process can be described from the perspective of sentence-generation from top to bottom; then, sentence-information will be said to be split rather than accumulated along the lines of the dependency structure.

**6. Defining syntactic rules**

The analysis presented in the previous sections of this article converges to the conclusion that the syntactic structure of sentences represents a flow-chart, encoding the movement of information from word to word in the sentence. The participants in each step of information-accumulation are a predicate and a word carrying the value for one of its logical arguments; whereas the sheer possibility of forming a bond between the relevant two words is determined by their respective a priori logical characteristics, whether the information will flow from the predicate to the argument or the converse, and the way it will be integrated into the information structure of the host, depends on the combinatory goal currently operative for that particular predicate-argument couple.

It appears that syntagmatic connectivity in sentences is founded on the basis of lexically given logical characteristics of predicate words. However, the lexicon leaves open the question how the relevant predicate functions in the present sentence, and where is it to find the value for its indeterminate elements. If syntactic rules are to be instructions for the construction of this network of connections, their role is to provide, for each predicate, the 'address' of the word carrying the value for each of its logical arguments, as well as to specify the combinatory operation to be applied to that predicate-argument couple.

Accordingly, a syntactic rule must contain three pieces of information: a logical predicate-argument couple; a combinatory goal in the form of one of the basic combinatory operations; and the language-specific codes or markers for locating the argument and calling up the relevant operation. As predicates are words, namely, entries in a mental lexicon, rules of syntax also reside in the lexicon, part of the entry of a given predicate *P*. Thus, for a predicate *P*, syntactic rules are triples of the form

<<argument **Ai**>, <combinatory operation **Oj**>, <markers **Mij(1-k)**>>,

defined for all its logical arguments and for all possible combinatory operations. The set of surface markers includes such well-known encoding devices as agreement, case marking, linear ordering, relative proximity and so forth, to be discussed below. According to the theory, the fundamental valency of words is their logical valency; their syntactic valency is in reality a *set* of 'syntactic valencies', each for a different pattern of combinatory operations with their logical arguments. In the usual conceptualization of syntactic valency, e.g. of a verb's requirement for a subject or a direct object, there is a confounding of the logical valency of the word with one of the grammatically marked options for the realization of the logical predicate-argument relation inherent in the word. Typically, the syntactic valency of verbs is defined solely in terms of their functioning in complement-taking operations, where their first argument is indeed marked with the complex of grammatical markers definitive of 'subject', but there is no parallel 'valency' definition for verbs functioning e.g. as attributors of their first arguments in relative clauses. In the present proposal, the overall syntactic valency of a predicate is defined by the complete range of options available for it to combine with its logical arguments, namely, both as their head and their dependent. Complementation of tensed verbs is not seen as in any way more basic that their use for attribution; Each grammatical pattern is 'base generated' and separately defined, depending on the grammatical relation to be achieved. As no one grammatical relation is given privileged status, there is no need to posit movements of arguments from one sentential position to another.

According to the theory, the same rule system sketched above serves both parsing and sentence generation. Given that the aim is the generation of a sentence, the input to the rules is the intention of the speaker to create a certain syntagmatic relation -- completion, attribution or some triadic relation -- between that predicate and a given logical argument of it. Then, the rules specify the language-specific devices for generating the relevant predicate-argument pair in the sentence, marked for the required combinatory operation. Conversely, if the aim is the parsing or interpretation of an existing sentence, the input to the rules is the set of grammatical devices used to realize a predicate and its logical argument in the sentence.[[22]](#endnote-23) On the basis of this information, the syntactic rule system recognizes the combinatory operation which is to be applied to the relevant predicate-argument couple. In both cases, syntactic rules directly match a semantic intent or interpretation to the formal patterning of the predicate-argument couple in the sentence. The distinction between structure and information content is eradicated.

In our model, we have allocated to language-specific syntactic devices the twin roles of locating in the sentence the word that carries the current value for the variable argument slot of the predicate, as well as the task of the specification of the combinatory operation the predicate-argument couple is to undergo. A priori, the two pieces of information are independent and it could have been the case that languages possess a separate 'pointer' system for identifying argument-carrying records and another system for signalling the type of operation to carry out on the couple. However, a review of the cross-linguistic evidence (see Note 23) discloses that languages typically signal the identity of the combinatory operation that applies to the relevant argument *vis-a-vis* the predicate, by manipulating the 'address' of the argument-carrying word. In other words, when a predicate is to unify with a logical argument by complementation, the argument will have a different record type, data type, marking and location -- or some of these -- than if the argument is to be e.g. the attributed head of the predicate. Argument-identification and combinatory-operation identification are, in natural languages, indistinguishable processes.

The details of these 'addressing' devices is the language-specific component of grammar. Overall, there is a limited number of different type of 'addressing' devices, but their precise nature depends on each specific language. It is outside the scope of this article to present a detailed exposition of the cross-linguistic variety of syntactic marker devices. As a generalization, it may be said that there are three major classes of devices: the redundant encoding of predicate or argument information on (or next to) the other member of the couple, as in cross-marking, pronoun clitics, case markers or oblique complement introducers; the linear positioning of arguments relative to functors in the sentence-string, including pre- and post-positioning and relative proximity; and the specification of form-class membership of the arguments, including, at the limit, the lexical stipulation of specific words as accepted arguments.

Cross-linguistic evidence demonstrates that these markers function as discriminant signals for the identification of predicate-argument couples and the relevant combinatory operation.[[23]](#endnote-24) For example, it is well-documented that the agreement or cross-referencing system is exclusively reserved for the insertion of core logical argument information into predicate words (Andrews 1985, Goldenberg 1985, Keenan 1984, Schacter 1977). Typically (e.g. Andrews 1985, Keenan 1985, Noonan 1985), the system is maximally elaborate in evaluative relations; it is relatively reduced in attributive uses and, as a rule, absent from the lower predicates of triadic combinations, which in any case receive the contents of some other predicate's argument slot and so do not need to directly identify the word serving as their evaluator. Thus, the existence, form, positioning, and elaboration of cross-markers on a predicate not only serve to provide a link with their logical arguments in the sentence, but also to differentiate between types of combinations the predicate and the relevant argument are to undergo.

Another example is provided by the differences in word order between evaluative, attributive, and triadic functor-argument couples, in languages that use word order to mark grammatical relations. Thus in English, an attributive adjective (or some other nonfinite predicate) typically precedes its argument, whereas an adjective in a triadic relation, serving as the predicative complement of e.g. a copula, typically follows it, as would a finite predicate its first argument. This generalization holds for all the subtypes of triadic relations, including controlled adjuncts and "small clauses"; apparently, the before/after type of word order in English mainly marks the difference between evaluation and modification, and it is indifferent to the distinction between word-type or slot-type evaluation.[[24]](#endnote-25)

In the present, dependency-type, approach, most syntactic rules except for the general principles of connectivity and projectivity reside in the lexicon. It is possible that some of the rules are defined for classes of words sharing combinatory properties, and that particular words 'inherit' them via a word-class identifier that forms part of their lexical entry (Hudson 1990:30-37). Whatever the precise arrangement for storage of rules, the crucial claim is that rules regulating binary and triadic relations between individual words are sufficient to account for the syntactic patterning of sentences. There is no need to evoke nonterminal units such as clauses or sentences in syntactic rules; these larger units can be represented as partial trees headed by the highest word in the relevant dependency structure, and their combinatory behaviour is determined by their root word, just as the case for any head-dependent combination. Even the requirement that every (English) sentence contain a tensed verb can be taken care of on the level of individual words, e.g. in allowing a finite verb to have no head (Hudson 1990:218-219). Given the general principle of acyclic connectivity between the words of a sentence and the principle of projectivity that transfers the resultant tree-structure to the linear sentence-string, lexical rules creating individual dependency couples are sufficient to generate grammatical sentences.

The model proposed above has obvious affinity with both dependency and categorial grammars, as well as with the varied group of grammars that use a unification formalism. Maybe it should be viewed as the categorial version of Word Grammar, or the dependency version of categorial grammar, with unification taken as the structure-constructing procedure. The former is the more accurate description: categorial grammars (as well as the related Montague grammar) adopt an explicit phrase-structural stance[[25]](#endnote-26) in which the basic ordering relation of sentences is not that of dependency but that of containment or part-whole relations. It makes for a much smaller modification to introduce logical predicate-argument relations as the foundation of dependency into Word Grammar that to change the basic characterization of syntactic structure in categorial grammars.

**7. The limits of a combinatory syntax**

It is obvious that the model presented above does not deal with the complete range of grammatical phenomena. A syntax of combinatory relations in natural languages is concerned with the explanation of a single phenomenon: the recoverability of the syntagmatic network expressed in a sentence, from the linear sentence-string. In the present proposal, the surface string is thought to be derived by projection from the syntagmatic or dependency structure of the sentence; the latter, in turn, is claimed to represent the path of combinatory or unification operations applied to the predicate-argument couples of the sentence.

However, grammar is not restricted to syntagmatic relations and their projection onto the sentence-string. A partially independent system of pragmatic topic-comment structuring existing to various degrees in most languages has already been touched on; at times, this system is in a considerable interaction with the predicate-argument organization forming the backbone of syntactic structure.[[26]](#endnote-27)

Even more centrally, it appears that not all vocabulary items in natural languages function as either predicates or arguments of predicates. In particular, languages typically contain a set of logical connectives such as *and*, *or*, *nor* and so forth; these appear to function as *record modifiers*, transforming a one-item predicate or argument storage record into a multi-element compound one. Even though such words have an important role in building the complete semantic reading of a sentence, it is not carried out by their constituting a component of the sentence's dependency structure.[[27]](#endnote-28) The present model is restricted to predicate-argument combining operations, but it is evident that language processors are able to perform other type of operations as well; these need to be taken into consideration in a comprehensive description of the human language processing system.

Third, it is well-accepted that not all of the linear structure of sentence-strings is arrived at by the projection of the syntactic tree-structure of the sentence. Two very prominent examples are coordination and gapping in English (see Hudson 1990:404-421). In both cases, the antecedent clause is constructed in the regular manner, by projecting the dependency structure to linear order through an adjacency-preserving transformation. However, the coordinated or the gapped clause appears to be parasitic on the already-achieved, semantically interpreted, linear order of the former, rather than being constructed afresh on the basis of the predicate-argument relations among its own constituents. Syntagmatic relations in these strings seem to be reconstructed on the basis of what is already established in their antecedent, rather than generated anew. A comprehensive grammar needs to account for surface-order copying operations like these as well as for regular combinatory operations.

On the positive side, the proposed model offers a one-step account of obtaining semantic readings of sentences from the processing of linear sentence-strings. By defining (syntactic) dependency as an aspect of the combinatory operations performed on logical predicate-argument couples, it provides a direct bridge between the formal structure of sentences and their meaning.

The human language processor emerges in this account as a relatively straightforward logical machine, with language-specific grammatical devices serving as commands for the activation of a highly limited set of basic combinatory operations. Syntax indeed appears to be like a compiler, reducing the particularities of individual languages to direct operating orders for a universal central processor. If this account withstands the test of detailed examination and the necessary extensions, it may provide a psychologically plausible working hypothesis about the characteristics of the human language processor.

REFERENCES

Ajdukiewicz, K. (1935), Die syntaktishe Konnexitat. Studia Philosophica 1, 1-27; translated as "Syntactic connection", pp.207-231 in: McCall, S. (Ed.), 1967, Polish Logic. Oxford: Oxford University Press.

Aho, A. V., Hopcroft, J. E. and Ullman, J. D. (1987), Data Structures and Algorithms. Reading, M.A.: Addison-Wesley.

Aho, A. V., Sethi, R. and Ullman, J. D. (1986), Compilers: Principles, Techniques, and Tools. Reading, M.A.: Addison-Wesley.

Allerton, D. J. (1982), Valency and the English Verb. London: Academic Press.

Allwood, J. (1981), On the distinction between semantics and pragmatics. pp.174-189 in: Klein, W. and Levelt, W. (Eds.), Crossing the Boundaries in Linguistics. Boston: D. Reidel Publishing.

Allwood, J., Andersson, L-G. and Dahl, O. (1977), Logic in Linguistics. Cambridge: Cambridge University Press.

Alston, W. P. (1964), The Philosophy of Language. Englewood Cliffs, N.J.: Prentice‑Hall.

Anderson, S. R. (1985a), Typological distinctions in word formation. pp.3-56 in: Shopen, T. (Ed.), Language Typology and Syntactic Description, vol 3, Cambridge: Cambridge University Press.

Anderson, S. R. (1985b), Inflectional morphology. pp.150-201 in: Shopen, T. (Ed.), Language Typology and Syntactic Description, vol 3, Cambridge: Cambridge University Press.

Anderson, S. R. and Keenan, E. L. (1985), Deixis. pp.259-308 in: Shopen, T. (Ed.), Language Typology and Syntactic Description, vol 3, Cambridge: Cambridge University Press.

Andrews, A. (1985), The major functions of the noun phrase. pp.62-154 in: Shopen, T. (Ed.), Language Typology and Syntactic Description, vol 1, Cambridge: Cambridge University Press.

Andrews, P. B. (1986), An Introduction to Mathematical Logic and Type Theory: To Truth through Proof. Computer Science and Applied Mathematics Series, Orlando: Academic Press.

Barendregt, H. P. (1985), The Lambda Calculus: Its Syntax and Semantics. Rev. ed. Amsterdam: North Holland.

Bar-Hillel, Y. (1953), A quasi-arithmetical notation for syntactic description. Language 29, 47-58.

Becker, K. F. (1841), Organism der Sprache. Frankfurt am Main.

Berman, R. (1980), The case of an (S)VO language: Subjectless constructions in Modern Hebrew. Language 56, 759-776.

Bolinger, D. (1975), Aspects of Language. 2nd. ed. New York: Harcourt Brace Jovanovich.

Borer, H. (1983), Parametric Syntax. Dordrecht: Foris.

Bresnan, J. (Ed.), (1982a), The Mental Representation of Grammatical Relations. Cambridge, M.A.: MIT Press.

Bresnan, J. (1982b), Control and complementation. pp. 282-390 in: Bresnan, J. (Ed.), The Mental Representation of Grammatical Relations. Cambridge, M.A.: MIT Press.

Burton-Roberts, N. (1975), Nominal apposition. Foundations of Language 13, 391-419.

Buszkowski, W. (1988). Generative power of categorial grammars. pp. 69-94 in: Oehrle, R. T., Bach, E. and Wheeler, D. (Eds.), Categorial grammars and natural language structures. Dordrecht: Reidel.

Carrier, J. and Randall, J. H. (1992), The argument structure and syntactic structure of resultatives. Linguistic Inquiry 23, 173-234.

Chomsky, N. (1981), Lectures on Government and Binding. Dordrecht: Foris.

Chomsky, N. (1992), A minimalist program for linguistic theory. MIT Occasional Papers in Linguistics 1.

Church, A. 1932/33 A set of postulates for the foundation of logic. Annals of Mathematical Studies (2)33, 346-366 and (2)34, 839-864.

Crain, S. and Steedman, M. (1985), On not being led up the garden path: the use of context by the psychological syntax processor. pp.320-358 in: Dowty, D. R., Karttunen, L. and Zwicky, A. M. (Eds.), Natural language parsing, Cambridge: Cambridge University Press.

Crystal, D. (1987), The Cambridge Encyclopedia of Language. Cambridge: Cambridge University Press.

Curry, H. B. (1930), Grundlagen der kombinatorischen Logik. American Journal of Mathematics 52, 509-536, 789-834.

Curry, H. B. and Feys, R. (1958), Combinatory Logic, vol 1, Amsterdam: North Holland.

Curry, H. B., Hindley, J. R. and Seldin, J. P. (1972), Combinatory Logic, vol 2, Amsterdam: North Holland.

Doron, E. (1993), The Discourse Function of Apposition. Paper presented at the Workshop on Discourse of the Israeli Association of Theoretical Linguistics, The Hebrew University, Jerusalem.

Dowty, D. R. (1982), Grammatical relations and Montague Grammar. pp.79-130 in: Jacobson, P. and Pollum, G. K. (Eds.), The nature of syntactic representation. Dordrecht: Reidel.

Foley, W. A. and Van Valin, R. D. (1985), Information packaging in the clause. pp.282-364 in: Shopen, T. (Ed.), Language Typology and Syntactic Description, vol 1, Cambridge: Cambridge University Press.

Frege, G. (1879), Begriffsschrift, eine der arithmetishen nachgebildete Formelssprache des reinen Denkens. Translated as "Begriffsschrift, a formula language, modelled upon that of arithmetic, for pure thought", reprinted pp.1-82 in: van Heijenoort, J. (1971), From Frege to Godel: A Source Book in Mathematical Logic, 1879-1931. Cambridge, M.A.: Harvard University Press.

Frege, G. (1891), Funktion und Begriff. Translated as "Function and concept", pp.21-41 in: Geach, P. and Black, M. (Eds.), Translations from the Philosophical Writing of Gottlob Frege. Oxford: Basil Blackwell, 1970.

Frege, G. (1892), Uber Sinn und Bedeutung, translated as "On sense and reference", pp.56-78 in: Geach, P. and Black, M. (Eds.), Translations from the Philosophical Writing of Gottlob Frege. Oxford: Basil Blackwell, 1970.

Gazdar, G., Klein, E., Pullum, G. and Sag, I. (1985), Generalized Phrase Structure Grammar. Oxford: Basil Blackwell.

Gibbs, R. W. (1984), Literal meaning and psychological reality. Cognitive Science 8, 275-304.

Givon, T. (1976), Topic, pronoun, and grammatical agreement. pp.151-188 in: Li, C. N. (Ed.), Subject and Topic. New York: Academic Press.

Goldenberg, G. (1985), Al torat hapoal vehapoal haivri [On the theory of the verb and the Hebrew verb]. pp.295-348 in: Bar-Asher, M. (Ed.), Mexkarim Balashon, vol 1, Jerusalem: The Hebrew University.

Goldenberg, G. (1989), The contribution of semitic languages to linguistic thinking. Ex Oriente Lux 30, 107-115.

Haugen, J. E. (1953), On resolving the close apposition. American Speech 28, 165-170.

Hellwig, P. (1986), Dependency Unification Grammar (DUG). Proceedings of the 11th International Conference on Computational Linguistics (COLING-1986), 195-198, Bonn.

Hindley, J. R. and Seldin, J. P. (1986), Introduction to Combinators and Lambda-Calculus. London Mathematical Society Student Texts, vol 1, Cambridge: Cambridge University Press.

Hudson, R. (1987), Zwicky on heads. Journal of Linguistics 23, 109-132.

Hudson, R. (1984), Word Grammar. Oxford: Basil Blackwell.

Hudson, R. (1990), English Word Grammar. Oxford: Basil Blackwell.

Hudson, R. (1992), Raising in syntax, semantics and cognition. pp.175-198 in: Roca, I. (Ed.), Thematic Structure: Its Role in Grammar. Berlin: Mouton de Gruyter.

Husserl, E. (1900), Logische Untersuchungen. Halle: Max Niemeyer. Translated by Findlay, J. N. as Logical Investigations. London: Routledge & Kegan Paul, 1970.

Jackendoff, R. (1977), X' Syntax: A Study of Phrase Structure. Cambridge, M.A.: MIT Press.

Jackendoff, R. (1983), Semantics and Cognition. Cambridge, M.A.: MIT Press.

Jackendoff, R. (1990), Semantic Structures. Cambridge, M.A.: MIT Press.

Jacobson, P. (1990), Raising as function composition. Linguistics and Philosophy 13, 423-475.

Jaeggli, O. A. (1986), Three issues in the theory of clitics: Case, doubled NPs, and extraction. pp.15-42 in: Borer, H. (Ed.), Syntax and Semantics 19: The Syntax of Pronominal Clitics. New York: Academic Press.

Katz, J. J. and Fodor, J. A. (1963), The structure of semantic theory. Language 39, 170-210.

Kay, M. (1985), Parsing in functional unification grammar. pp.251-278 in: Dowty, D. R., Karttunen, L. and Zwicky, A. M. (Eds.), Natural Language Parsing, Cambridge: Cambridge University Press.

Kayne, R. (1975), French Syntax: The Transformational Cycle. Cambridge, M.A.: MIT Press.

Keenan, E. L. (1984), The functional principle: generalizing the notion of 'subject of'. pp. 298-309 in: La Gally, M. W., Fox, R. and Bruck, A. (Eds.), Papers from the Tenth Regional Meeting of the Chicago Linguistic Society. Chicago: Chicago Linguistic Society.

Keenan, E. L. (1985), Relative clauses. pp.141-170 in: Shopen, T. (Ed.), Language Typology and Syntactic Description, vol 2, Cambridge: Cambridge University Press.

Keenan, E. L. and Comrie, B. (1977), NP accessibility and universal grammar. Linguistic Inquiry 8, 63-100.

Kempson, R. M. (1977), Semantic Theory. Cambridge: Cambridge University Press.

Lakoff, G. and Thompson, H. (1975a), Introducing Cognitive Grammar. Proceedings of the First Annual Meeting of the Berkeley Linguistic Society 1, 295-313.

Lakoff, G. and Thompson, H. (1975b), Dative questions in Cognitive Grammar. pp.337-350 in: Grossman, R. E., San, L.J. and Vance, T. J. (Eds.), Papers from the Parasession on Functionalism. Chicago Linguistic Society, Chicago.

Lambek, J. (1958), The mathematics of sentence structure. American Mathematics Monthly 65, 154-170.

Lee, D. W. (1952), Close apposition: An unresolved patter. American Speech 27, 268-275.

Longacre, R. E. (1985), Sentences as combinations of clauses. pp.235-286 in: Shopen, T. (Ed.), Language Typology and Syntactic Description, vol 2, Cambridge: Cambridge University Press.

Lyons, J. (1968), Introduction to Theoretical Linguistics. Cambridge: Cambridge University Press.

McCord, M., Bernth, A., Lappin, S. and Zadrozny, W. (1992), Natural language processing within a Slot Grammar framework. International Journal on Artificial Intelligence Tools 1, 229-277.

Mel'cuk, I. A. (1979), Studies in Dependency Syntax. Ann Arbor: Karoma.

Mel'cuk, I. A. (1988), Dependency Syntax: Theory and Practice. Albany: State University of New York Press.

Mittwoch, A. (1982), On the difference between Eating and Eating something. Linguistic Inquiry 13, 113-121.

Montague, R. (1974), Formal Philosophy (Edited by R. H. Thomason). London: Yale University Press.

Moortgat, M. (1988), Categorial Investigations: Logical and Linguistic Aspects of the Lambek Calculus. Dordrecht: Foris.

Nichols, J. (1986), Head-marking and dependent-marking grammar. Language 62, 56-119.

Noonan, M. (1985), Complementation. pp.42-140 in: Shopen, T. (Ed.), Language Typology and Syntactic Description, vol 2, Cambridge: Cambridge University Press.

Norwood, J. E. (1954), The loose apposition in present-day English. American Speech 24, 267-271.

Pereira, F. C. N. and Shieber, S. M. (1987), Prolog and Natural-Language Analysis. Stanford: Center for the Study of Language and Information.

Pollard, C. and Sag, I. (1987), Information-based Syntax and Semantics, vol 1, Fundamentals. Stanford: Center for the Study of Language and Information.

Quine, W. V. (1960), Word and Object. Cambridge, M.A.: MIT Press.

Quine, W. V. (1967), Russell's ontological development. pp.304-314 in: Schoenman, R. (Ed.), Bertrand Russell, philosopher of the century. London: Allen and Unwin.

Robinson, J. J. (1970), Dependency structures and transformational rules. Language 46, 259-285.

Rommetveit, R. (1974), On Message Structure. London: Wiley.

Safir, K. (1986), Subject clitics and the NOM-drop parameter. pp.333-356 in: Borer, H. (Ed.), Syntax and Semantics 19: The Syntax of Pronominal Clitics. New York: Academic Press.

Schachter, P. (1976), The subject in Philippine languages: topic, actor, actor-topic, or none of the above. pp.491-518 in: Li, C. N. (Ed.), Subject and Topic. New York: Academic Press.

Schachter, P. (1977), Reference-related and role-related properties of subjects. pp.279-306 in: Cole, P. and Sadock, J. M. (Eds.), Syntax and semantics 8: Grammatical relations. New York: Academic Press.

Schachter, P. (1985), Parts-of-speech systems. pp.3-61 in: Shopen, T. (Ed.), Language Typology and Syntactic Description, vol 1, Cambridge: Cambridge University Press.

Schonfinkel, M. (1924), Uber die Bausteine der mathematischen Logik. Mathematische Annalen 92, 305-316. Translated as "On the Building Blocks of Mathematical Logic", pp. 355-366 in: van Heijenoort, J. (1971), From Frege to Godel: A Source Book in Mathematical Logic, 1879-1931. Cambridge, M.A.: Harvard University Press.

Shieber, S. M. (1986), An Introduction to Unification-based Approaches to Grammar. Stanford: Center for the Study of Language and Information.

Sleator, D. D. and Temperley, D. (1992), Parsing English with a Link Grammar. Unpublished manuscript, School of Computer Science, Carnegie Mellon University, P.A.: Pittsburgh.

Steedman, M. (1988), Combinators and grammars. pp.417-442 in: Oehrle, R. T., Bach, E. and Wheeler, D. (Eds.), Categorial grammars and natural language structures. Dordrecht: Reidel.

Tesniere, L. (1959), Elements de Syntaxe Structurale. Paris: Klincksieck.

Tutte, W. T. (1984), Graph Theory. Menlo Park, C.A.: Addison-Wesley.

Wiggins, D. (1971), On sentence‑sense, word‑sense and difference of word-sense. pp.14‑34 in: Steinberg, D. D. and Jakobovits, L. A. (Eds.), Semantics. Cambridge: Cambridge University Press.

Zeevart, H., Klein, E. and Calder, J. (1987), Unification Categorial Grammar. Working Papers in Cognitive Science 1. pp.195-222, University of Edinburgh Centre for Cognitive Science.

Ziff, P. (1960), Semantic Analysis. Ithaca: Cornell University Press.

Zwicky, A. (1985), Heads. Journal of Linguistics 21, 1-30.

Notes

1. A more precise conceptualization is that speakers intend to convey utterance-meanings, namely, situated and illocutionary-force loaded speaker's meanings. Sentence-meaning as employed in this text is considered an abstraction, representing the meaning of a sentence in its most typical communicative use. Similarly, lexical word-meaning is considered a representation of the prototypical contribution of the relevant word to the meaning of situated utterances. For an exposition of this view, see Alston (1964), Gibbs (1984). The resulting syntax is seen as a first approximation to a use-conditional system.

2. The use of the compiler analogue in theoretical linguistics is not new; however, as far as I know, it has not been employed as a metaphor for syntax. For example, Kay (1985) evoked the image of a compiler for a transducer translating the canonical form of the grammar used for sentence generation into a different form used for sentence parsing, namely, as a simile for a translation between two declarative grammars. Closer to the present approach is Steedman (1988) who pointed out that in certain (actual) compilers a use is made of the primitive logical combinators of Combinatory Logic which Steedman claims also underlie natural language processing. Steedman mentions this fact as an indirect support for his proposing Combinatory Logic (which uses no bound variables) rather than other applicative systems as the foundation of Categorial Grammar; the simile is not developed further.

3. See Aho, Sethi and Ullman (1986:1). Actually a compiler is any program or set of programs that translates a program in one language to a program in another language of whatever nature; still, for the present purposes what is relevant is that in all computers there must be a compiler for every programming language users are writing programs in, before these programs are functional in operating the machine.

4. A complication arises when certain words have multiple semantic valencies, namely, in cases of so-called zero-derivation (Allerton 1982, Bolinger 1975). The range of solutions available for this problem is lucidly presented in Allerton (1982:65-79). At present, this source of ambiguity will be ignored.

5. The use of verbs in an attributive relation with their logical arguments is a widespread pattern, cross-linguistically. See for example Keenan (1985) on the ubiquitousness of relative clauses, or Schacter (1985) on the adjectival use of verbs in many languages. Cross-linguistic variations in the freedom of participating in combinatory relations will be discussed later in the text.

6. See Noonan (1985) for a cross-linguistic review.

7. Even though post-traditional syntactic theories do not as a rule allocate a central or even any formal role to grammatical relations in the establishment of syntactic structure, it can be demonstrated that the relevant distinctions are embedded in the theory. In configurational approaches where grammatical relations do not form part of the formal terminology of the grammar, they can be nevertheless derived from the grammar: It is possible to identify them with pairs of configurational positions. In dependency-type grammars, grammatical relations appear either as subtypes of the dependency relation (e.g. Hudson 1984, 1990) or else as labels on a dependency structure (e.g. Mel'cuk 1988).

In spite of the appearance on the linguistic scene of post-traditional formal syntactic theories, the grammatical-relations theory in its original form has by no means disappeared. For example, it underlies most current work on Semitic languages (e.g. Goldenberg 1985, 1989), as well as much of the work on language typology (e.g. Andrews 1985).

8. In several taxonomies of grammatical relations, a separate appositional relation is acknowledged. However, an examination of apposition and its subtypes reveals that apposition is either an attributive-type relation, consisting of a reduced relative clause, restrictive (in the case of close apposition, see Burton-Roberts 1975, Haugen 1953, Lee 1952) or nonrestrictive (in the case of loose apposition, see Norwood 1954), or else it is a complementation-type relation, e.g. of a title-proper name combination. See also Doron (1993).

9. Another, and prototypical, 'interpretation' of the same abstract operation is the mathematical functional application. As it is well known, the tradition of likening the linguistic combination of predicates and arguments to functional application in mathematics goes back as far as Frege (1879, 1891). In current terminology, we would say that both are separate concretizations or 'interpretations' of the abstract combinatory operation confusingly going under the same name. However, by contrast to the Fregean tradition in linguistics (e.g. Ajdukiewicz 1935, Bar-Hillel 1953, Lambek 1958, and Montague 1974), in Combinatory Logic application is considered to be only one of two different primitive binary combinations, the other being **K**. (See in text).

10. Church, the founder of Lambda-calculus, did not accept lambda-expressions abstracting a variable *x* out of an expression *M* as well-formed logical objects unless *x* occurred as a free variable in *M*. This restriction has been removed in the system of Combinatory Logic, which accepts functions with fictitious arguments defined over all objects (Curry and Feys 1958:89), and, subsequently, in Lambda-calculus, where it defines a different and more inclusive logical system than Church's original one.

11. Combinatory Logic is equivocal about the primitiveness of yet another combinator, namely, the identity combinator **I**. This combinator returns without a change whatever it is applied to. On the one hand, it can be derived from **K** and **S** (**I** = **SKK**) and therefore cannot be said to be properly speaking a primitive; on the other hand, it is said to be necessary as an atomic combinator for "strong reduction" (Curry *et al*. 1972:87, note 2). For our purposes, what is interesting is that it can be demonstrated that generalized relativizers and subordinators like the English *that* exhibit the logical characteristics of an **I**-combinator. Apparently, the use of a fully or partially saturated lambda-expression involving a tensed verb in the role of a nominal is easier when it is mediated by an identity-combinator. Nevertheless, note the optional character of *that* when it is not required for disambiguating purposes.

12. Interestingly, Steedman (1988) who suggested the extension of Categorial Grammar by the combinators of Combinatory Logic, did not think the linguistic system makes use of the "elegantly minimal" **S--K** system of primitives but of the less elegant **B, C, W** and **I** system.

13. There are serious indications that the list should be extended to include multi-element combinations of greater complexity that the triadic; there have been various suggestions made for a generalized compositional operation involving an unlimited number of components. Such a generalized **S** combinator (or a generalized version of some of its subtypes, e.g. **B**) may be needed to account for 'unbounded dependencies' constituting apparent violations of the adjacency principle such as topicalization and Dutch 'crossed dependencies'; for some discussion, see Hudson (1990) on the 'visitor' relation and Steedman (1988), Moortgat (1988), and Jacobson (1990) on function composition in the Categorial Grammar tradition. One possibility to be explored is that predicate compounds are created by a 'wait-command' that prevents the unification of an argument into a predicate until that predicate is otherwise saturated. In topicalization the fronting may trigger such a 'wait-command' relative to the highest predicate in a chain, allowing the fronted word to become the argument of the whole compound via the argument-slot open at its lowest level.

14. The information content of one storage unit being 'moved into' the content of another unit is of course a metaphorical description of what the actual process consists of; if real-life computer data structures are anything to go by, the connection between the variable slot of one unit and the value-carrying other unit is probably instantiated by something analogous to a pointer from the one to the other (see for instance Aho, Hopcroft and Ullman 1987:14-16).

15. As the cross-linguistic evidence demonstrates, any predicate whose variable components has not yet been completely saturated can serve, often without any further marking whatsoever, as an attributive predicate. In Mandarin Chinese, for example, clauses from which one of the arguments of the verb (subject, direct-object and so on) is missing, function as relative clauses 'outputting' a definition of the relevant argument, with a general marker particle *de* distinguishing these cases from compounded modifiers using the very same lambda-expression (Anderson 1985a:48-49).

16. It is instructive in this context that in the computer science literature which has adopted a Chomskian, phrase-structure based 'rewrite' approach to the characterization of the syntax of expressions, the resultant "parse tree" containing nonterminal higher nodes is immediately exchanged for a pruned "abstract syntax tree" without such nodes, where operators appear as interior nodes and the operands are the children (rather than the sisters) of the node for the operator (Aho, Sethi and Ullman 1986:287-288).

17. There are some versions of dependency grammar, e.g. Hudson's Word Grammar (1984, 1990, 1992) where multiple heads as well as bidirectional (mutual) dependencies are allowed.

18. See for example Pereira and Shieber's (1987) Definite Clause Grammars; Hellwig's (1986) Dependency Unification Grammar; McCord's Slot Grammar (e.g. McCord, Bernth, Lappin and Zadrozny 1992); Sleator and Temperley's (1992) Link Grammar. Pollard and Sag (1987:157-168) also have adjunct slots similar to complement slots in subcategorization frames, at least for nouns; that is, nouns are said by them to have an optional adjective slot and/or relative-clause slot.

19. As mentioned earlier (Note 7), the relationship of grammatical relations and the dependency relation is not quite clear in dependency-type grammars. Mel'cuk (1988) allocates to grammatical relations the role of labels on the dependency structure, a move essentially isomorphic to Hudson's (1990) who considers grammatical relations subtypes of the dependency relation. However, the dependency relation is a unitary ordering relation and it cannot be said to have subtypes; there is no variability in its characteristics correlated with the grammatical relation involved. For instance, the degree of asymmetry in the roles of the head and the dependent is constant; so is the requirement for dependents to have one and only one head; nor is there any effect of grammatical relation on the application of the Projection Principle to the relevant couple. Thus, to consider grammatical relations as subtypes rather than as creators of dependency between words is an error of logical type.

20. Cross-linguistic evidence strongly supports this analysis. According to Noonan's review of complementation phenomena (Noonan 1985:49-91), the most frequent device for marking secondary-predicate status, and one that characterizes almost all complementation phenomena across languages, is the loss of the capacity of complement-predicates to have an overt or syntactic subject. The English pattern is thus not the exception but the rule cross-linguistically, and it may be taken to reflect a universal and fundamental property of triadic unification.

21. It also solves the problem of the apparent cyclical dependency structures generated by relative clauses or wh-questions, for analogous reasons.

It should be noted that even though the present approach does not allow either multiple heads or mutual and cyclical dependencies, that is not to say that two words may not unify because of more than a single logical requirement for saturation. For example, it can be demonstrated that locative prepositions such as *on* in *John put the book on the table* are both required by the logical argument structure of the verb *put* and function as controlled adjuncts in an attributive triadic relation, taking object slot of the verb (currently occupied by *the book*) as their first argument. In general, attributive triadic predicates may or may not be part of the logical valency structure of the higher predicate, and if the former, they stand in a double relation with it.

22. Interestingly, the present approach implies that a parser has two different and in principle independent sets of constraints available to guide its search for the structure of a sentence. On the one hand, every word but the root is to be supplied with a head; on the other hand, every predicate is to be supplied with values for its arguments. Actual parsing is probably built around the simultaneous satisfaction of both sets of constraints.

23. See for example on agreement and clitic pronouns: Anderson (1985a), Anderson and Keenan (1985), Andrews (1985), Berman (1980), Borer (1983), Givon (1976), Goldenberg (1985), Jaeggli (1986), Kayne (1975), Nichols (1986), Safir (1986), Schacter (1976, 1977). On case marking: Andrews (1985), Foley and Van Valin (1985), Haegeman (1991), Pollard and Sag (1987), Schachter (1976, 1977). On lexical specialization: Keenan (1985), Keenan and Comrie (1977), Longacre (1985), Noonan (1985). On oblique introducers: Allerton (1982), Anderson (1985a, 1985b). On linear ordering: Andrews (1985), Crystal (1987). On argument hierarchy: Pollard and Sag (1987).

24. A complication is introduced into the word order system in English by topicalization; thus relative pronouns and subordinate interrogative pronouns, which are the heads of the clausal finite verb, precede it whether or not this constitutes a difference from the normal evaluative word order between the verb and the argument. In addition, all appositional type of attribution typically places the modifier nominal after the modified one, e.g. in the full relative clause *the child who is on the swing*. For this reason, it is possible that appositional attribution should be viewed as a special subtype of attribution, involving exclusively a co-referential nominal redoubling. This applies also to cases of reduced relative clauses such as *the child on the swing* or *Lawrence the writer*, where the modifier represents the predicative component of a relative construction.

25. See for instance the syntactic rules defined by Montague (1974:195-199) which are formation rules for the combination of basic expressions into well-formed complex expressions by concatenation. And, relatedly, as Buszkowski (1988:92) pointed out, "Approximately, categorial grammar theory can be viewed as a logical meta-theory for the theory of phrase-structure grammar".

26. For instance, in certain languages relativization may be restricted to topics rather than freely available for all arguments of a verb (Andrews 1985:140). In addition, the hypothetical generalized compositional operation may centrally involve topicalization.

27. For example, Hudson (1990:421) does not analyze *and* as part of the dependency structure of a sentence.

1. . A more precise conceptualization is that speakers intend to convey utterance-meanings, namely, situated and illocutionary-force loaded speaker's meanings. Sentence-meaning as employed in this text is considered an abstraction, representing the meaning of a sentence in its most typical communicative use. Similarly, lexical word-meaning is considered a representation of the prototypical contribution of the relevant word to the meaning of situated utterances. For an exposition of this view, see Alston (1964), Gibbs (1984). The resulting syntax is seen as a first approximation to a use-conditional system. [↑](#endnote-ref-2)
2. . The use of the compiler analogue in theoretical linguistics is not new; however, as far as I know, it has not been employed as a metaphor for syntax. For example, Kay (1985) evoked the image of a compiler for a transducer translating the canonical form of the grammar used for sentence generation into a different form used for sentence parsing, namely, as a simile for a translation between two declarative grammars. Closer to the present approach is Steedman (1988) who pointed out that in certain (actual) compilers a use is made of the primitive logical combinators of Combinatory Logic which Steedman claims also underlie natural language processing. Steedman mentions this fact as an indirect support for his proposing Combinatory Logic (which uses no bound variables) rather than other applicative systems as the foundation of Categorial Grammar; the simile is not developed further. [↑](#endnote-ref-3)
3. . See Aho, Sethi and Ullman (1986:1). Actually a compiler is any program or set of programs that translates a program in one language to a program in another language of whatever nature; still, for the present purposes what is relevant is that in all computers there must be a compiler for every programming language users are writing programs in, before these programs are functional in operating the machine. [↑](#endnote-ref-4)
4. . A complication arises when certain words have multiple semantic valencies, namely, in cases of so-called zero-derivation (Allerton 1982, Bolinger 1975). The range of solutions available for this problem is lucidly presented in Allerton (1982:65-79). At present, this source of ambiguity will be ignored. [↑](#endnote-ref-5)
5. . The use of verbs in an attributive relation with their logical arguments is a widespread pattern, cross-linguistically. See for example Keenan (1985) on the ubiquitousness of relative clauses, or Schacter (1985) on the adjectival use of verbs in many languages. Cross-linguistic variations in the freedom of participating in combinatory relations will be discussed later in the text. [↑](#endnote-ref-6)
6. . See Noonan (1985) for a cross-linguistic review. [↑](#endnote-ref-7)
7. . Even though post-traditional syntactic theories do not as a rule allocate a central or even any formal role to grammatical relations in the establishment of syntactic structure, it can be demonstrated that the relevant distinctions are embedded in the theory. In configurational approaches where grammatical relations do not form part of the formal terminology of the grammar, they can be nevertheless derived from the grammar: It is possible to identify them with pairs of configurational positions. In dependency-type grammars, grammatical relations appear either as subtypes of the dependency relation (e.g. Hudson 1984, 1990) or else as labels on a dependency structure (e.g. Melcuk 1988).

   In spite of the appearance on the linguistic scene of post-traditional formal syntactic theories, the grammatical-relations theory in its original form has by no means disappeared. For example, it underlies most current work on Semitic languages (e.g. Goldenberg 1985, 1989), as well as much of the work on language typology (e.g. Andrews 1985). [↑](#endnote-ref-8)
8. . In several taxonomies of grammatical relations, a separate appositional relation is acknowledged. However, an examination of apposition and its subtypes reveals that apposition is either an attributive-type relation, consisting of a reduced relative clause, restrictive (in the case of close apposition, see Burton-Roberts 1975, Haugen 1953, Lee 1952) or nonrestrictive (in the case of loose apposition, see Norwood 1954), or else it is a complementation-type relation, e.g. of a title-proper name combination. See also Doron (1993). [↑](#endnote-ref-9)
9. . Another, and prototypical, 'interpretation' of the same abstract operation is the mathematical functional application. As it is well known, the tradition of likening the linguistic combination of predicates and arguments to functional application in mathematics goes back as far as Frege (1879, 1891). In current terminology, we would say that both are separate concretizations or 'interpretations' of the abstract combinatory operation confusingly going under the same name. However, by contrast to the Fregean tradition in linguistics (e.g. Ajdukiewicz 1935, Bar-Hillel 1953, Lambek 1958, and Montague 1974), in Combinatory Logic application is considered to be only one of two different primitive binary combinations, the other being **K**. (See in text). [↑](#endnote-ref-10)
10. . Church, the founder of Lambda-calculus, did not accept lambda-expressions abstracting a variable x out of an expression M as well-formed logical objects unless x occurred as a free variable in M. This restriction has been removed in the system of Combinatory Logic, which accepts functions with fictitious arguments defined over all objects (Curry and Feys 1958:89), and, subsequently, in Lambda-calculus, where it defines a different and more inclusive logical system than Church's original one. [↑](#endnote-ref-11)
11. . Combinatory Logic is equivocal about the primitiveness of yet another combinator, namely, the identity combinator **I**. This combinator returns without a change whatever it is applied to. On the one hand, it can be derived from **K** and **S** (**I** = **SKK**) and therefore cannot be said to be properly speaking a primitive; on the other hand, it is said to be necessary as an atomic combinator for "strong reduction" (Curry et al. 1972:87, note 2). For our purposes, what is interesting is that it can be demonstrated that generalized relativizers and subordinators like the English that exhibit the logical characteristics of an **I**-combinator. Apparently, the use of a fully or partially saturated lambda-expression involving a tensed verb in the role of a nominal is easier when it is mediated by an identity-combinator. Nevertheless, note the optional character of that when it is not required for disambiguating purposes. [↑](#endnote-ref-12)
12. . Interestingly, Steedman (1988) who suggested the extension of Categorial Grammar by the combinators of Combinatory Logic, did not think the linguistic system makes use of the "elegantly minimal" **S--K** system of primitives but of the less elegant **B, C, W** and **I** system. [↑](#endnote-ref-13)
13. . There are serious indications that the list should be extended to include multi-element combinations of greater complexity that the triadic; there have been various suggestions made for a generalized compositional operation involving an unlimited number of components. Such a generalized **S** combinator (or a generalized version of some of its subtypes, e.g. **B**) may be needed to account for 'unbounded dependencies' constituting apparent violations of the adjacency principle such as topicalization and Dutch 'crossed dependencies'; for some discussion, see Hudson (1990) on the 'visitor' relation and Steedman (1988), Moortgat (1988), and Jacobson (1990) on function composition in the Categorial Grammar tradition. One possibility to be explored is that predicate compounds are created by a 'wait-command' that prevents the unification of an argument into a predicate until that predicate is otherwise saturated. In topicalization the fronting may trigger such a 'wait-command' relative to the highest predicate in a chain, allowing the fronted word to become the argument of the whole compound via the argument-slot open at its lowest level. [↑](#endnote-ref-14)
14. . The information content of one storage unit being 'moved into' the content of another unit is of course a metaphorical description of what the actual process consists of; if real-life computer data structures are anything to go by, the connection between the variable slot of one unit and the value-carrying other unit is probably instantiated by something analogous to a pointer from the one to the other (see for instance Aho, Hopcroft and Ullman 1987:14-16). [↑](#endnote-ref-15)
15. . As the cross-linguistic evidence demonstrates, any predicate whose variable components has not yet been completely saturated can serve, often without any further marking whatsoever, as an attributive predicate. In Mandarin Chinese, for example, clauses from which one of the arguments of the verb (subject, direct-object and so on) is missing, function as relative clauses 'outputting' a definition of the relevant argument, with a general marker particle de distinguishing these cases from compounded modifiers using the very same lambda-expression (Anderson 1985a:48-49). [↑](#endnote-ref-16)
16. . It is instructive in this context that in the computer science literature which has adopted a Chomskian, phrase-structure based 'rewrite' approach to the characterization of the syntax of expressions, the resultant "parse tree" containing nonterminal higher nodes is immediately exchanged for a pruned "abstract syntax tree" without such nodes, where operators appear as interior nodes and the operands are the children (rather than the sisters) of the node for the operator (Aho, Sethi and Ullman 1986:287-288). [↑](#endnote-ref-17)
17. . There are some versions of dependency grammar, e.g. Hudson's Word Grammar (1984, 1990, 1992) where multiple heads as well as bidirectional (mutual) dependencies are allowed. [↑](#endnote-ref-18)
18. . See for example Pereira and Shieber's (1987) Definite Clause Grammars; Hellwig's (1986) Dependency Unification Grammar; McCord's Slot Grammar (e.g. McCord, Bernth, Lappin and Zadrozny 1992); Sleator and Temperley's (1992) Link Grammar. Pollard and Sag (1987:157-168) also have adjunct slots similar to complement slots in subcategorization frames, at least for nouns; that is, nouns are said by them to have an optional adjective slot and/or relative-clause slot. [↑](#endnote-ref-19)
19. . As mentioned earlier (Note 7), the relationship of grammatical relations and the dependency relation is not quite clear in dependency-type grammars. Melcuk (1988) allocates to grammatical relations the role of labels on the dependency structure, a move essentially isomorphic to Hudson's (1990) who considers grammatical relations subtypes of the dependency relation. However, the dependency relation is a unitary ordering relation and it cannot be said to have subtypes; there is no variability in its characteristics correlated with the grammatical relation involved. For instance, the degree of asymmetry in the roles of the head and the dependent is constant; so is the requirement for dependents to have one and only one head; nor is there any effect of grammatical relation on the application of the Projection Principle to the relevant couple. Thus, to consider grammatical relations as subtypes rather than as creators of dependency between words is an error of logical type. [↑](#endnote-ref-20)
20. . Cross-linguistic evidence strongly supports this analysis. According to Noonan's review of complementation phenomena (Noonan 1985:49-91), the most frequent device for marking secondary-predicate status, and one that characterizes almost all complementation phenomena across languages, is the loss of the capacity of complement-predicates to have an overt or syntactic subject. The English pattern is thus not the exception but the rule cross-linguistically, and it may be taken to reflect a universal and fundamental property of triadic unification. [↑](#endnote-ref-21)
21. . It also solves the problem of the apparent cyclical dependency structures generated by relative clauses or wh-questions, for analogous reasons.

    It should be noted that even though the present approach does not allow either multiple heads or mutual and cyclical dependencies, that is not to say that two words may not unify because of more than a single logical requirement for saturation. For example, it can be demonstrated that locative prepositions such as on in John put the book on the table are both required by the logical argument structure of the verb put and function as controlled adjuncts in an attributive triadic relation, taking object sot of the verb (currently occupied by the book) as their first argument. In general, attributive triadic predicates may or may not be part of the logical valency structure of the higher predicate, and if the former, they stand in a double relation with it. [↑](#endnote-ref-22)
22. . Interestingly, the present approach implies that a parser has two different and in principle independent sets of constraints available to guide its search for the structure of a sentence. On the one hand, every word but the root is to be supplied with a head; on the other hand, every predicate is to be supplied with values for its arguments. Actual parsing is probably built around the simultaneous satisfaction of both sets of constraints. [↑](#endnote-ref-23)
23. . See for example on agreement and clitic pronouns: Anderson (1985a), Anderson and Keenan (1985), Andrews (1985), Berman (1980), Borer (1983), Givon (1976), Goldenberg (1985), Jaeggli (1986), Kayne (1975), Nichols (1986), Safir (1986), Schacter (1976, 1977). On case marking: Andrews (1985), Foley and Van Valin (1985), Haegeman (1991), Pollard and Sag (1987), Schachter (1976, 1977). On lexical specialization: Keenan (1985), Keenan and Comrie (1977), Longacre (1985), Noonan (1985). On oblique introducers: Allerton (1982), Anderson (1985a, 1985b). On linear ordering: Andrews (1985), Crystal (1987). On argument hierarchy: Pollard and Sag (1987). [↑](#endnote-ref-24)
24. . A complication is introduced into the word order system in English by topicalization; thus relative pronouns and subordinate interrogative pronouns, which are the heads of the clausal finite verb, precede it whether or not this constitutes a difference from the normal evaluative word order between the verb and the argument. In addition, all appositional type of attribution typically places the modifier nominal after the modified one, e.g. in the full relative clause the child who is on the swing. For this reason, it is possible that appositional attribution should be viewed as a special subtype of attribution, involving exclusively a co-referential nominal redoubling. This applies also to cases of reduced relative clauses such as the child on the swing or Lawrence the writer, where the modifier represents the predicative component of a relative construction. [↑](#endnote-ref-25)
25. . See for instance the syntactic rules defined by Montague (1974:195-199) which are formation rules for the combination of basic expressions into well-formed complex expressions by concatenation. And, relatedly, as Buszkowski (1988:92) pointed out, "Approximately, categorial grammar theory can be viewed as a logical meta-theory for the theory of phrase-structure grammar". [↑](#endnote-ref-26)
26. . For instance, in certain languages relativization may be restricted to topics rather than freely available for all arguments of a verb (Andrews 1985:140). In addition, the hypothetical generalized compositional operation may centrally involve topicalization. [↑](#endnote-ref-27)
27. . For example, Hudson (1990:421) does not analyze and as part of the dependency structure of a sentence. [↑](#endnote-ref-28)