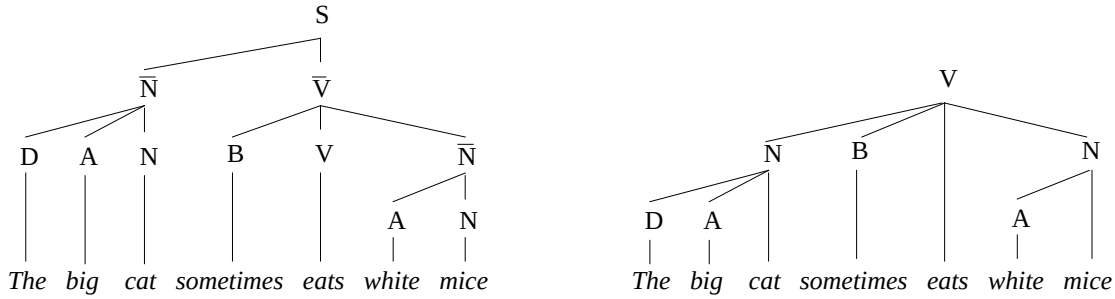
If HPSG were a dependency grammar ...

Sylvain Kahane[1]

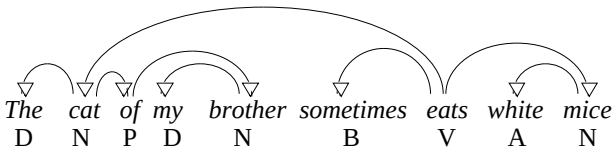
Abstract. The purpose of this paper is to show how HPSG can be favourably simulated by a dependency grammar. We will be specially interested in the translation of the SLASH feature and we will see that, providing we accept not to put the constituency structure forward, linguistic phenomena using the SLASH feature can be interpreted in various ways, including a fondamental notion which will be named, following Tesnière, nucleus.

1 HEAD MARKED PHRASE STRUCTURE GRAMMARS AND DEPENDENCY GRAMMARS

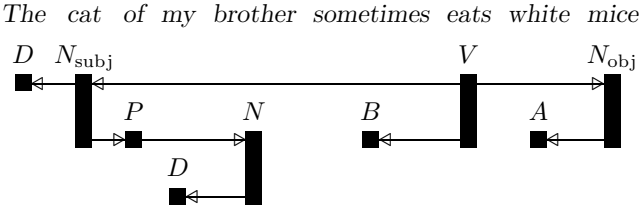
It is sometimes forgotten that a head-marked phrase-structure tree can be canonically converted into a dependency tree. Much better, Gaifman 1965 proves that a (head marked) phrase-structure grammar which verifies a special condition of finiteness (which is verified by X-bar grammar and therefore by HPSG) can be converted into a dependency grammar.



A **dependency structure** for a sentence is a one-to-one mapping between the nodes of a tree (the **dependency tree**) and the words of the sentence. In other words, it is a tree whose nodes are linearly ordered (by the speech order) and labelled by words and lexical categories. Moreover, the tree and the linear order verify a property of compatibility, the **Projectivity Property** (PP) (Lecerf 1961, Mel'čuk 1988), which is equivalent to the Constituent Connexity Property (the constituents are the projections of the subtrees of the dependency tree) and which ensures that, when the nodes are linearly ordered, (1) links (= edges of the tree) do not cut each other and (2) the governor of a given node cannot be covered by a link between this node and one of its dependent nodes.



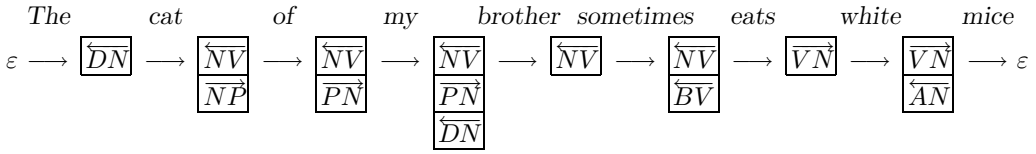
The structure can be extended by adjoining an order on the concomitant links (i.e. whose supports meet) by agreeing that the longest link is above. That gives us a new structure which will be named a **scheme** (= tree + linear order + order on the concomitant links).



1. E-mail : sk@ccr.jussieu.fr. This work is supported by Univ. Paris 10 (92001 Nanterre cedex), TALANA (UFR Ling., Univ. Paris 7, 75254 Paris 5 cedex) and Equipe d'Analyse (UFR Math., Univ. Paris 6, 75252 Paris 5 cedex). Paper revised in March 2002.

2 DEPENDENCY GRAMMARS AND PUSHDOWN AUTOMATA

The scheme of a sentence can be built incrementally by a pushdown automaton where the links are the stack symbols (a link is represented by the categories of the words it links and and an arrow pointing on the dependent). Each word accepts links which are removed from the stack and puts new links (which will be accepted by following words). A transition (associated with a given word) is a couple of strings called left and right valences. In a standard pushdown automaton (e.g. Hopcroft & Ullman 1980), only the "top" stack symbol is removed, but, in our case, it is possible to remove a (possibly empty) string (= the left valence) and the projectivity ensures that it is a "top" stack string. When the input head is between two words, the content of the stack is exactly the string of links which are present between these two words. Note that with our conventions of representation, the "top" of the stack is on the bottom. A correct parsing gives us a scheme. In particular, the order on the concomitants links is the order of the stack. The stack is empty at the beginning and must be empty again at the end (for the acceptance). The following parsing corresponds to the scheme of section 1.



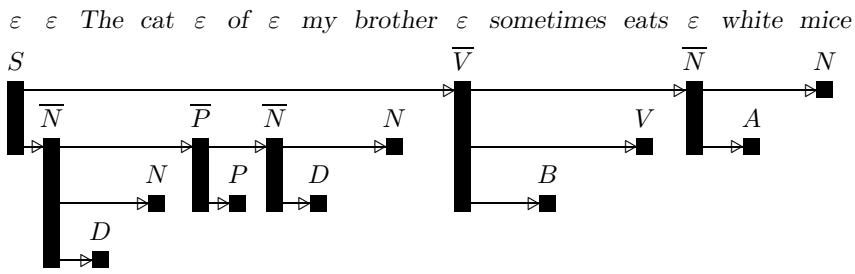
It can be easily verified that such an automaton is weakly equivalent to a standard pushdown automaton, hence to a context-free grammar.

Note that our automaton, that will be called the **dependency parser** in the following, is in the present state very near (= strongly equivalent) to a link grammar (Sleator & Temperley 1993), a diamond parser (Lombardo 1992) or a categorial grammar (Bar-Hillel 1953, Bar-Hillel *et al.* 1960).

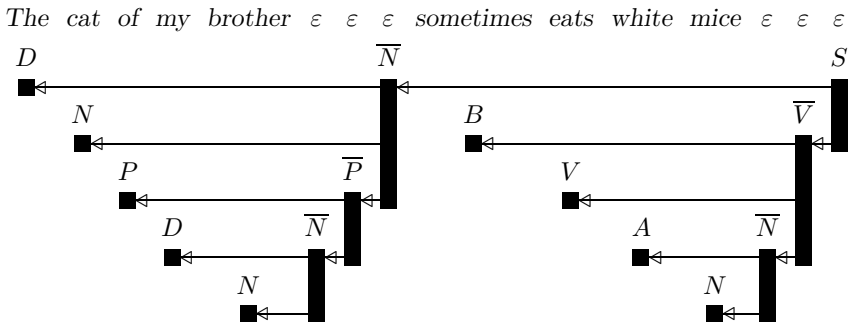
3 HEAD DRIVEN PARSERS AND DEPENDENCY PARSERS

Let us come back to a (head-marked) phrase-structure grammar G and consider a parser deriving from it. The task of the parser is to verify whether a given sentence is generated by G, and, if so, to provide the derivation tree. This tree can be constructed in as many ways as we can order its nodes (respecting the order on the leaves whoch correspond to the words of the sentence) : All these parsings can be realized by a pushdown automaton (as previously defined) (providing we allows ε-transitions corresponding to non-lexical rules). We will present the three most standard parsings on an example.

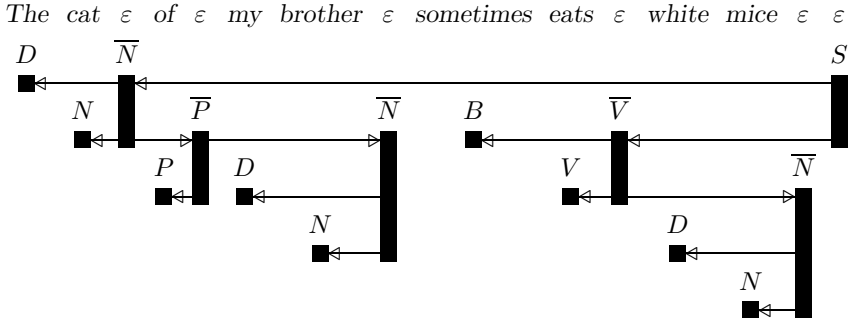
1. The top-down parser : The derivation tree is constructed by following the prefix order. The main problem of this parser is that we always have to anticipate, and thus we are frequently forced to backtrack).



2. The bottom up parser : The derivation tree is constructed by following the postfix order. The main problem of this parser is that we cannot anticipate, and so we almost have to complete the parsing before noting a failure (for example, in order to verify that the sequence *Peter is speaking a boy who* ... is ungrammatical, we must completely parse the $\overline{N}$ and therefore the relative clause !).



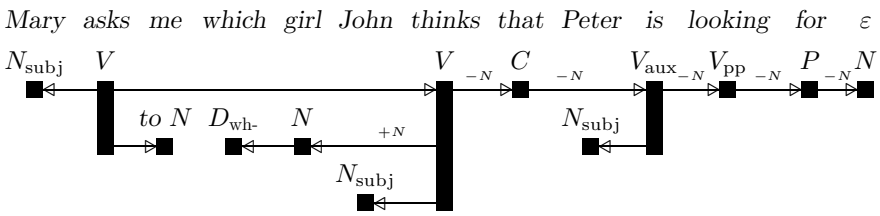
3. The head-driven parser : It is the best compromise between the two previous parsers. The idea is to trigger off a rule as soon as we have enough information (to avoid a backtracking or conversely a pointless parsing), that is, as soon as the head of the rule has been activated :



In fact, it is exactly what the dependency parser does. Each rule of the dependency grammar corresponds to a sequence of rules of the phrase-structure grammar ; the different pieces of this rule are activated by the dependency parser exactly when the corresponding pieces are activated by the head-driven parser. So what are the differences ? The dependency grammar is more concise and, as we will see in the next section, allows particular extensions. On the other hand, a certain modularity is lost : Each dependency rule is the compound of a sequence of rules of the phrase-structure grammar, which are used in the composition of other dependency rules. This idea that the rules are modular and that they are the compound of more general rules is extremely important : The set of rules has many regularities, thus it must not be defined extensively, but it must be generated by a sort of grammar that we will call the **metagrammar**. Nevertheless, the implementation of this idea need not use phrase-structure system. Moreover, using phrase-structure system does not prevent from resorting a metagrammar.

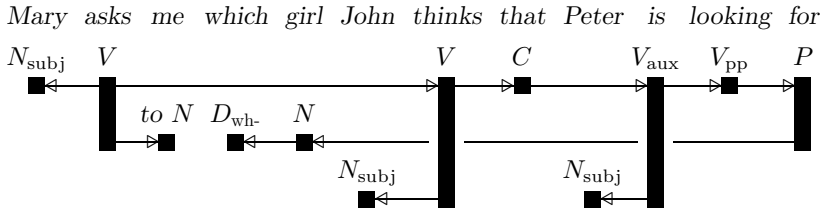
4 THE SLASH FEATURE

We are now interested in sentences with discontinuous constituents which require to use a SLASH feature in HPSG. The SLASH feature can be also translated into a dependency representation. For example, we can propose the following analysis for a sentence with an extraction :



In this analysis, the extracted phrase is governed by the main verb of the interrogative clause (here *thinks*). Trough the SLASH feature (supported by the links), this verb informes the governor of the extracted position (here *for*) that this position is empty. Note we chose to label the links, but it would have been possible, in a equivalent way, to add a SLASH feature on the category node as it is done in HPSG).

But it is not necessary to use a SLASH feature : We can link the extracted phrase directly to its governor :



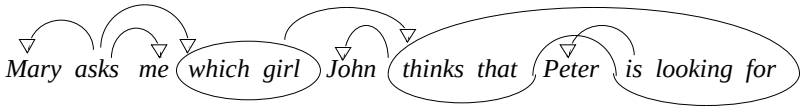
Note that we lost the PP in this new analysis : That was expected because we are dealing with discontinuous constituents. But we do not want to completely give up the PP. We are going to see how the projectivity can be controled.

5 NUCLEUS AND PROJECTIVITY

> *Le nucléus est donc en dernière analyse l'entité syntaxique élémentaire, le matériau fondamental de la charpente structurale de la phrase, et en quelque sorte la cellule constitutive qui en fait un organisme vivant. Tandis que le mot, simple segment de la chaîne parlée, est l'unité linéaire de la phrase, le nucléus en est l'unité structurale. Jetant maintenant un regard en arrière, nous comprenons pourquoi il est si difficile de saisir la notion de mot. C'est qu'elle n'a aucune réalité syntaxique. On ne peut faire de la syntaxe que dans la mesure où derrière les mots on sait voir les nucléus.*
>
> Lucien TESNIÈRE

Let us focus on the extraction phenomena (= relativization, interrogation, topicalization ...). We claim that a position can be extracted only if it is governed by a main verbal nucleus, where **nucleus** (resp. **verbal nucleus**) means a complex unit which assumes the same role as a word (resp. as a simple verb). The notion of nucleus appears in Tesnière 1959. Every language has nuclei, but each language develops its own types of nuclei. Verbal nuclei of English are verbs and complex units like auxilliary−participle (*be eating*, *have eaten*), verb−infinitive verb (*want to eat*), verb−conjunction−verb (*think that eat*)[2] , verb−preposition (*look for*) and all units built by transitivity with the previous ones (*thinks that is looking for*). In French, the nucleus verb−preposition does not exist (**Marie se demande quelle fille Pierre* ***parle à***), but there exists a nucleus verb−subject and verb−direct object (*l'homme dont la* ***fille dort***, *l'homme dont Pierre* ***aime*** *la* ***fille***) and no other nucleus verb-complement (**l'homme dont Pierre* ***parle à*** *la* ***fille***). We will also consider nominal nuclei : there are nouns (*who*) and complex units like determinant−noun (*which girl*, *whose girl*) and noun−noun complement (*the daughter of which man*). We claim that a position can be pronominalized (in an extraction) if it belongs to a nominal nucleus governed by a main verbal nucleus. The notion of nucleus is very general and intervenes in other linguistic phenomena like negation and coordination.[3]

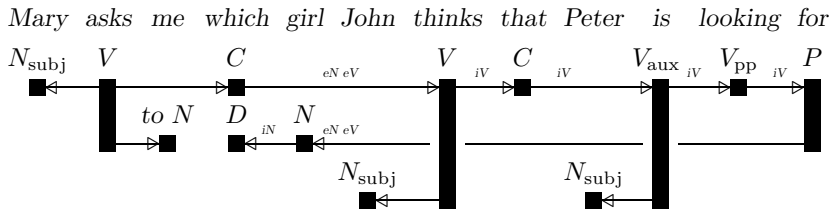
Let us return to the representation of a sentence. Nuclei are represented by bubbles ; they share with words the property to be a node of the dependency structure.



Another representation consists of retaining the usual dependency structure and pointing out the nuclei by labelling the internal and external links of a verbal (resp. nominal) nucleus with labels iV and eV (resp. iN and eN). Furthermore, I think it is better to associate two nodes to the pronoun because it assumes two functions : a pronominal function and the same function as other conjunctions, like *that*, which have the role to subordinate a verb (the subordinating category of the pronoun will be named C ; it is shared by all subordinating pronouns). Therefore the link between the verbal nucleus and the extracted nucleus gives us two links in the scheme.

2. Note the fact that the verbal nucleus, as a verb, cannot have two subjects avoid the subject of the subordinate verb to be recognized by the nucleus and so to be extracted !

3. For example, in gapping coordination, a simple verb, but also a verbal nucleus, can be erased : *Mary* ***is looking for*** *the landscape and Peter a pretty girl.*



The linear order is not controlled by the dependency structure on the words but by the dependency structure on the nucleus. And for this structure, a projectivity property must be assumed, the PPν (Projectivity Property with Nucleus). The PPν ensures that only intra- or extranuclear links can cut each other.

We can modify the pushdown automata of Section 2 in order to allow some links to cross a transition, providing certain conditions like PPν. Such an automaton is generally not context-free. If the number of concomitant crossings is bounded, it is obviously context-free. It just seems that this number is bounded by 2 for sentences of natural languages. It even seems that the scheme of any sentence of a natural language cannot have more than 7 (perhaps 8) concomitant links. Therefore, the stack of the dependency parser does never contain more than 7 links and so the parser is weakly equivalent to a finite state automaton (whose states are the possible contents of the stack, i.e. the strings of less than 7 couples of categories).

6 CONCLUSION

All the mecanisms of a head-driven phrase-structure parser (and in particular of HPSG) can be favourably simulated by a dependency parser (controlled by a metagrammar). On the other hand, the dependency structure allows to consider primordial linguistic units, the nuclei, that cannot be simply expressed in a phrase-structure system. A study in progress about coordination phonemona confirms the previous results : Dependency systems combined with nuclei and a similar tool for coordination (the string of coordinate elements, which can be words or nuclei, is represented by a bubble which have the same properties as a simple node) seem to be an appreciable model.

REFERENCES

BAR-HILLEL Yehoshua, 1953, "A quasi-arithmetical notation for syntactic description", *Language* ***29.1***, 47-58.

BAR-HILLEL Yehoshua, GAIFMAN Haïm & SHAMIR E., 1960, "On categorial and phrase-structure grammars", *Bull. Res. Counc. of Israël* ***9F***, 1-16.

GAIFMAN Haïm, 1965, "Dependency systems and phrase-structure systems", *Information and Control* ***8***, 304-337 ; Rand Corporation Techn. Report RM-2315, 1961.

HOPCROFT John E. & ULLMAN Jeffrey D., 1980, *Introduction to automata theory, languages, and computation*, Addison-Wesley, Mass.

LECERF Yves, 1961, "Une représentation algébrique de la structure des phrases dans diverses langues natuelles", *C. R. Acad. Sc. Paris* ***252***, 232-234.

LOMBARDO Vincenzo, 1992, "Incremental Dependency Parsing", *Proc. of the 30th ACL*, 291-293.

MEL'CUK Igor, 1988, *Dependency syntax : theory and practice*, State Univ. Press NY, Albany, NY.

SLEATOR Daniel & TEMPERLEY David, 1993, "Parsing english with a link grammar", *Third Int. Workshop on Parsing Technologies* ; Carnegie Mellon Univ. Comp. Sc. Techn. Report CMU-CS-91-196, 1991.

TESNIÈRE Lucien, 1959, *Éléments de syntaxe structurale*, Kliencksieck, Paris, 670p. ; *Esquisse d'une syntaxe structurale*, 1953, Kliencksieck, Paris, 30p.