A Genealogy of Phrase Structure

F.A. Grootjen, V. Kamphuis, J.J. Sarbo

Computing Science Institute/

CSI-R9814 April 1998

Computing Science Institute Nijmegen
Faculty of Mathematics and Informatics
Catholic University of Nijmegen
Toernooiveld 1
6525 ED Nijmegen
The Netherlands

A Genealogy of Phrase Structure

F. Grootjen[*] V. Kamphuis[†]
J. Sarbo[‡]
University of Nijmegen, The Netherlands

May 4, 1998

Abstract

In standard approaches to NLP phrase structure is usually specified by a grammar. The coverage of such a grammar, especially in the case of performance data, is often insufficient; the grammar is subject to frequent modifications and this can bring about maintainability problems. In this paper we describe a relational basis underlying phrase structure. Based on that, a model is defined that yields hierarchical structure as the result of more abstract principles related to the combinatorial properties of linguistic units. The model is simple in use and easy to maintain, and provides an important key to the description of non-phrase structure configurations that require greater flexibility.

1 Background

The standard paradigm in NLP is one using part-whole analysis (e.g. Haegeman, 1991; Pollard and Sag, 1994; Lambek, 1988). This approach works well for clear cut examples, but in performance data combinations of language items may occur that the grammar writer may have overlooked, or that are difficult to describe, for example, discontinuous structures and structural variations, and coordination.

Could this problem be solved by extending the grammar? Certainly, but experience shows that such phenomena occur frequently, and this can cause maintainability problems, eventually. Due to the frequent modifications, the size and the complexity of the grammar may grow beyond an effectively

[*]Supported by a grant from the Computing Science Institute
[†]Dept. of Language and Speech
[‡]Computing Science Institute

maintainable limit. This also implies that the (side) effects of a modification to the grammar may not be correctly estimated, in general.

Another solution could be found in extending the description formalism. Although this may help in some cases, there is a danger that the formalism becomes too ornate, and thereby loses some of its theoretical modelling power.

There is also a language theoretical issue involved. We exemplify this by a sample specification of NPs. In a part-whole analysis, NPs can be specified as e.g. NP: determiner, premodifiers, noun, postmodifiers. But NPs show a greater variety, as *the happy girl*; *happy girls*; *girls*; *the girl in the red dress*; and even *the happy* are well-formed NP instances.

What then makes the NP? In fact, no one of the elements mentioned, as each of them can be optional. We think that it is not a particular *element* that creates the NP, but rather, the *different relations* between the elements. In general, we consider hierarchical structure in language to be the result of a dynamic process in which the interaction of different relations reaches some form of completeness. In the next section we discuss these relations in more detail.

2 Relation schemes

Natural Language Concept Analysis (NLCA) [KS98] defines three relation schemes underlying structure in language, in particular, phrase structure. The different relation schemes reflect certain conceptual distinctions that may be expressed by means of language. For example, the distinction between an action/state and its participants differs from that between an action/state or participant on the one hand, and some property of it on the other. Also, one may distinguish between the core content of an action/state, participant or property, and some specification of it with respect to time, reference, intensity etc.

The above distinctions are incorporated in the relation schemes: major predication, minor predication and qualification, respectively. We refer to the first two as predication. An instance of a relation scheme is called a relation.

A *qualification* (Q) consists of a qualifier and a core. The qualifier has no information content independent of the core; it makes the core more specific. For example, a Q-relation can specify the referential status of NPs; tense/aspect in VPs. The qualifier needs the core, and this is modelled by introducing a placeholder, called a *Proto-item*, for the core (in the case the qualifier precedes the core). The relation between qualifier and Proto-item can be symmetric or asymmetric. When the core is realized, it replaces the Proto-item. In the relation between qualifier and core, the qualifier points at some qualification of the core, and the core fulfils the combinatorial need of the qualifier. E.g. the article–noun relation.

A *minor predication* (*mp*) consists of a (minor) predicate and an argument. The predicate has information content independent of its argument and adds new, factual information to it. The relation between minor predicate and argument is asymmetric: the predicate needs its argument, but not the other way round (modification). The predicate points at some property of its argument, and the argument fills the combinatorial need of the predicate. E.g. the adjective–noun relation.

A *major predication* (*MP*) consists of a predicate and its argument(s). Both have information content, and the relation between predicate and argument(s) is symmetric (each requires the presence of the other). The predicate introduces an argument structure, and incorporates its arguments into a single relation. E.g. the verb–argument(s) relation.

The three relation schemes (cf. Fig. 1) can be recursively applied, and their sum uniquely characterises the input.

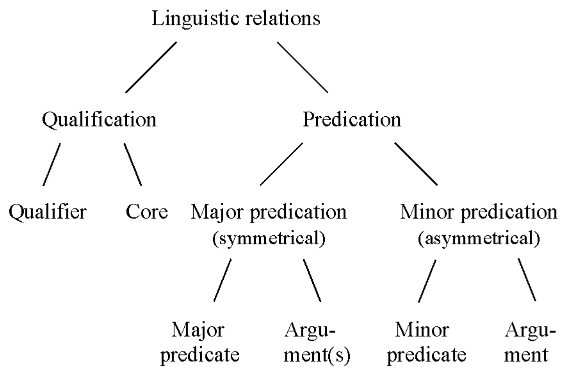

Figure 1: Linguistic relations in NLCA

Ex. 1 *The happy girl bought some flowers.*

There are four *Q*-relations: *the–girl*, *some–flowers*, PAST–*buy*, and PLURAL–*flower*. The adjective *happy* is in *mp*-relation with *girl*. There is a *MP*-relation between the verb *buy* and its arguments, '*the happy girl*' and '*some flowers*'.

The relations *Q*, *mp* and *MP* can be realized in language on different levels, e.g. on the morphological level, or on the level of syntax. For this reason NLCA has the potential to be language independent. At present we have a detailed specification of English. An NLCA application of Hungarian is under development. Preliminary results show the viability of our approach also for this language. In this paper we will restrict ourselves to English.

3 Principles

In NLCA the input is analysed from left to right, and the relations are evaluated incrementally. A relation is evaluated when qualifier and core, or predicate and argument(s) bind to each other. The evaluation, which can be initiated by either participant in the relation, is greedy, meaning that lexical items relate with the 'nearest' surrounding candidates.

This principle, called *greedy binding*, is restricted by the demand that only visible items can bind to each other. The visibility structure and any change to it, is due to the *Q*- and *MP*-relations: each introduces a new visibility range for itself. That *mp*-relations do not change visibility coincides with the optionality of modifiers (minor predicates). The creation of a new range makes an older range invisible, but when the new range ceases to exist, the older range becomes visible again (cf. Fig. 3 below). The set of such ranges defines a set of partitions of the input lexical items.

A visibility range is terminated by *closing* (and by encountering end-of-sentence). This operation applied to a combination of lexical items can yield a single new item, called a *lexical unit*. (N.B. a lexical item is a lexical unit; the principle described above extends naturally from lexical items to lexical units.) Closing is triggered by a change in the visibility structure.

The linguistic properties of a lexical unit are derived from its members. Technically, a new lexical item is only generated in case of a *MP*-relation. No new lexical unit needs to be introduced for a *Q*-relation, as the core can also represent the relation itself. We will use this feature in the examples.

Visibility can be more easily formalised by its complement, invisibility. Consider the input sentence as a sequence (s) of lexical units numbered from 1 (first unit) to n (last unit). Let $s(i)$, $s(j)$ and $s(k)$ be three lexical units, with $i < j < k$ (or $k < j < i$). Let $s(j)$ and $s(k)$ be involved in one *Q*- or *MP*-relation. All lexical units following (preceding) and including $s(j)$ are *invisible* to $s(i)$. The reader is invited to compare the dynamic visibility of NLCA with the static one known from block structured programming languages.

Finally, we say, the input is *well-formed* if the combinatorial need of each lexical unit is satisfied, meaning that the external argument positions of all items are filled (see below).

4 Representation

Both predicate–argument and qualifier–core relations are based on functor–argument relations. These are represented as pointers between lexical units, or in the case of morphological realisation, as constants.

The source of a pointer is a lexical unit; the destination is an *argument position* of the related item. There are two *internal argument positions* for each lexical unit, representing information about the item itself: one for the *Q*- and one for the *mp*-relations, denoted by _int(q) and _int(m), resp. The number of *Q*-relations a lexical unit can be involved in is finite, whereas that of *mp*-relations is unlimited.

Each lexical unit has one or more *external argument positions* _ext, representing its combinatorial properties. In the case of verbs there are as many of such positions as the number of obligatory arguments. In an *MP*-relation there is a bijection between external arguments and external argument positions. The argument positions can be labelled, e.g. in the case of verbs; the labelling is defined by the lexicon, e.g. AGENT.

We represent relations by a *Relation Matrix* (RM). There is a row allocated for each noun (a typical argument), and a column for each article, preposition, adjective, adverb and verb (typical functors). Furthermore a column is allocated for each external argument position of a verb. For Proto-items a row or a column is allocated (referred to as Proto-object and Proto-attribute), depending on the qualifier introducing it. Lexical units created by closing are represented similarly, depending on their properties. Internal and external argument positions of lexical items are given as buckets on the left and right hand side, respectively.

The pointers between rows and columns are (conceptually) stored in the cells of the RM, in the examples however, they are graphically represented. Besides the pointers, a cell contains a '+' sign if the destination of a pointer stored in the cell is an external argument position. These signs will be used as markers of the emerging phrasal structure.

5 Towards an algorithm

All information about lexical units is contained in the lexicon, such as type, number and location (pre- or post-) of argument(s) etc. In the example below, lexical information is specified on-the-fly.

Besides the lexicon, an NLCA implementation needs an interpreter that (incrementally) evaluates the relations. The interpreter is very simple, and (conceptually) could be 'moved' to each lexical entry. The resulting model of language would be one in which the input lexical units generate the phrase structure *autonomously.*

The user of an NLCA implementation only needs to know about the relation schemes and NLCA's principles, no knowledge about the algorithm or the implementation is required. This implies that in such a model of language, information about lexical units can be expressed in terms of familiar linguistic notions.

Ex. 2 *The happy girl bought some flowers.*

the generates a column. As a qualifier, it functions as the internal argument of its object. The qualifier precedes its core, therefore it creates a Proto-object and points at its qualifying internal argument.

· the → Proto-object_int(q)

happy generates a column. Its internal argument position is not filled. As an adjective, its external argument position needs to be filled with a nominal element. There is a Proto-object present, hence there is a pointer from the Proto-object to the external argument position of the adjective (greedy binding), which results in a '+' in the RM (under 'happy'). The attribute itself points at the modifying internal argument position of the Proto-object. Note that this leads to a *chain* of pointers from 'the' via the Proto-object to an external argument that has been filled; such a chain gives rise to *inheritance* of bindings to all units involved in the chain. Therefore, there will also be a '+' under 'the' and a pointer from the Proto-object to 'the_ext'. This, in fact, creates a *nominal adjective phrase* with an implied head (the Proto-object).

· Proto-object → happy_ext
· happy → Proto-object_int(m)
· Proto-object → the_ext
· '+' in RM in cell Proto-object/happy
· '+' in RM in cell Proto-object/the

There is an important difference here between the role and treatment of the article and the adjective. Note that the nominal adjective phrase would not be created without the article: it is the article that supplies the Proto-object that functions in the nominal adjective phrase. This is because articles belong to the class of qualifiers. The adjective, on the other hand, belongs to the class of modifiers that are involved in the relation of minor predication. For this reason, they can be said to have an implicit object required to fill the place of their external argument position. The Proto-object generated by the qualifier can fill this role. Both qualifier and modifier belong to the class of internal arguments; however, they do not have the same status and are treated differently. The qualifier 'the' generates a Proto-item but this Proto-item cannot fulfil its external argument need: otherwise, it would be wrongly assumed that a string consisting of the article only would be grammatical. Hence there is no pointer initially from the Proto-object to the external argument position of the qualifier. With the modifying adjective, exactly the reverse situation holds. As adjectives can be used in different types of contexts (e.g. attributively or predicatively), they do not create a Proto-object. However, since they are involved in the relation of predication, their external argument position can be filled by a Proto-object generated by a qualifier. They themselves point to the internal argument position of that Proto-object.

girl replaces the Proto-object. The noun inherits the pointer to the external argument position of 'the'. There is still a phrase, but now it is a full noun phrase rather than a nominal adjective phrase. Since there is not yet a pointer to the external argument position of the noun, we still do not have a clause, only a phrase.

· 'girl' replaces Proto-object

bought generates a column. Its qualifying internal argument is filled by the feature PAST; its external arguments are AGENT and THEME. Since 'buy' is a major predicate, it fills the external argument position of the object 'girl', and 'girl' points to the AGENT role. As a result, there is a '+' in the Relation Matrix in cells girl/AGENT and girl/buy. However, since only one of the external argument positions of the transitive verb is filled, the clause is not yet complete.

· buy $\rightarrow$ girl_ext
· girl $\rightarrow$ buy_ext (THEME)
· '+' in RM in cell girl/AGENT
· '+' in RM in cell girl/buy

some A quantifying pronoun which may function as a determiner or as an independent pronoun. We can make a unified account if we treat it as a qualifier that, like the article, introduces a Proto-object; however, unlike with articles the Proto-object now also points to the external argument of the qualifier. As a result, there is a '+' in the Relation Matrix in cell Proto-object/some. This explains the possibility of e.g. *She bought some*, which, indeed, is complete but has an implicit object.

The Proto-object also realizes the external argument THEME, causing a '+' to be placed in the appropriate cell of the Relation Matrix.

· some → Proto-object_int(q)
· Proto-object → some_ext
· Proto-object → buy_ext (THEME)
· '+' in RM in cell Proto-object/some
· '+' in RM in cell Proto-object/THEME
· '+' in RM in cell Proto-object/buy

flowers replaces the Proto-object. 's' can be regarded as part of the qualifying internal argument (qualifier). Note that this does not conflict with the fact that 'some' also is an internal argument: they are unifiable within the same domain (both can signify plural; together they are plural indefinite).

· 'flowers' replaces Proto-object

The Relation Matrix for this sentence is displayed in Fig. 2.

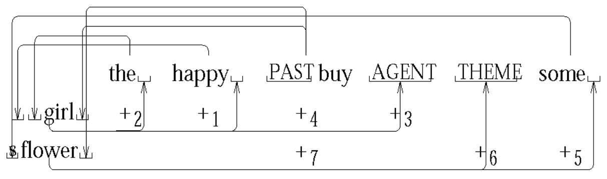

Figure 2: *The happy girl bought some flowers*

This treatment of quantifying pronouns has two important advantages. First, it does not require ambiguous lexical entries. The same can be said for demonstrative pronouns, numerals and other function words that are ambiguous between independent and adjectival use. Second, the use of Proto-objects makes it unnecessary to have a rule defining noun phrase heads as realized either by nouns, or by numerals, quantifying pronouns, demonstrative pronouns etc. In fact, this also applies to nominal adjective phrases: there is no need to define adjectives as possible realizations of noun phrase heads. The nominal adjective phrase follows naturally from the presence of the article (creating the Proto-object) and the adjective (combining with the Proto-object). Furthermore, this approach also accounts for the potential structural ambiguity of a quantifying pronoun or a nominal adjective phrase followed by a plural noun phrase, as in apposition. (Example: ‘On Monday she got a big bunch of flowers. The white, lilies, wilted after a mere few days.’)

Going through the sentence from left to right, we see the following structure emerge:

- At word ‘happy’ we obtain the nominal adjective phrase ($+_1$ and, through inheritance, $+_2$);
- At word ‘girl’ we obtain the noun phrase (‘girl’ replaces Proto-object);
- At word ‘some’ we obtain the clause with an independent pronoun ($+_5$, $+_6$ and $+_7$);
- At word ‘flowers’ we obtain the clause with ‘some’ as determiner (‘flowers’ replaces Proto-object).

The visibility ranges emerging during the analysis are depicted in Fig. 3.

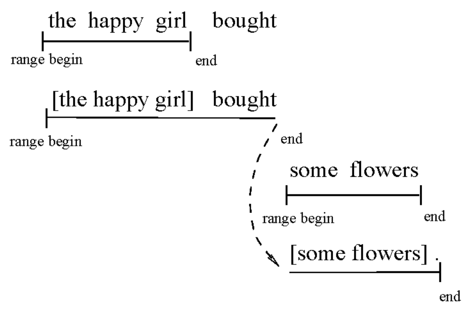

Figure 3: *Visibility ranges in Ex. 2*

6 Flexibility

In the previous sections we have discussed a model which attempts to explain the underlying nature of hierarchical structure in language. Our aim in doing so was to find a method of description which is *inherently* more flexible than one that takes hierarchical structure (in particular, phrase structure) as given. This is needed especially in order to account for non-phrase structure configurations that occur in natural language. In this section we shall illustrate the greater flexibility of our approach on the basis of an example of *coordination* (we restrict ourselves to a case of coordination with *and*).

Coordination often gives rise to disruption of regular phrase structure patterns. This occurs, especially, in cases of conjunction reduction, gapping, or other cases of non-constituent coordination. Furthermore, the relation between the individual conjuncts and their context need not necessarily be the same. These observations give rise to a view of coordination where each conjunct needs to be analysed separately in relation to the context of the coordinate structure as a whole [Kam97]. The relational basis of NLCA suits this view perfectly, as the discussion of the following examples illustrate.

Ex. 3 *The happy girl bought some flowers yesterday, and a skirt today.*

This example is problematic for a phrase structure-based account, because the substrings *some flowers yesterday* and *a skirt today* do not form a single unit at any level of hierarchical structure; rather, they contain two independent constituents, one of which functions as an adverbial at clause level, and the other as object to the verb. In standard phrase structure, such coordination cannot be described, as there is no single unit of analysis that the rule describing the coordinate structure can refer to. In NLCA, however, the relational basis makes a natural account possible, as follows.

We divide the surface string of this sentence into the following substrings:

left context + left conjunct +
coordinator +
right conjunct + right context.

This applies to all cases of continuous coordination; however, there are also cases of discontinuous coordination where the left conjunct and coordinator are not immediately adjacent. An example would be the sentence in Ex. 4; we will come back to this below.

Ex. 4 *The happy girl bought some flowers yesterday, and a skirt.*

Continuing with Ex. 3, the coordinator in the pattern is realized by *and*; the content of the other substrings needs to be identified.

Syntactically, the sequences [left context + left conjunct + right context] and [left context + right conjunct + right context] need to be *well-formed* (see definition above). This condition will be used to identify left context and right context. The content of the substrings [left conjunct] and [right conjunct] may consist of one or more lexical units (as defined in Sect. 3). Each of these units has its own combinatorial properties, and participates in some relation(s). For the left conjunct, these relations will have been determined at the point where the coordinator is found. For instance, in the current example, the substrings [left context + left conjunct] form the sequence *The happy girl bought some flowers yesterday* (cf. Ex. 2; the adverb *yesterday* forms a *mp*-relation with the verb). Note that, at this point, it is not yet clear which part of the sentence makes up the left context or the left conjunct.

Upon reaching the coordinator, we have completed a *MP*-relation, i.e. the external arguments have been found. The effect of the coordinator *and* is that the external argument positions of the major predicate can be re-used.

In general, coordination refers to the *relation(s)* invoked in the substring preceding the coordinator; in essence, it connects compatible lexical units in the left and right conjunct. Briefly, two such units are *compatible* when they may participate in the same relation scheme, in the same role (i.e. qualifier or core; predicate or argument), and relate to the same type of lexical unit. Note that the well-formedness condition mentioned above applies.

We now attempt to identify left and right conjunct by evaluating the string following the coordinator incrementally. The first lexical unit is *a*. This invokes a *Q*-relation. In the visibility range of the major predicate, there are: the predicate (*bought*), its external arguments, and its internal argument, the minor predicate *yesterday.* None of them is compatible with *a.* This means the coordination cannot be resolved, and we have to proceed with the input. The analysis of *a skirt* yields a lexical unit that can participate in a *MP*-relation as argument. *some flowers* is compatible; a first approximation of the two conjuncts is found: *some flowers* and *a skirt.* Now we have two possibilities: (a) the left conjunct also contains *yesterday* (continuous coordination), or (b) we have a case of discontinuous coordination. These options can be evaluated using look-ahead.

ad (a). There is a minor predicate (*today*) in the input that is compatible with the minor predicate *yesterday.* The left and right conjuncts are determined as: *some flowers yesterday* and *a skirt today.*

ad (b). There is no compatible minor predicate in the input (see Ex. 4).

This means that *yesterday* does not belong to the left conjunct. The left and right conjuncts are determined as: *some flowers* and *a skirt*.

We note that, from a syntactic point of view, an alternative analysis could identify the conjuncts as *the happy girl* and *a skirt* (discontinuous coordination of AGENTs). Such an analysis can only be excluded on semantic grounds. Questions regarding the preferred strategy of binding are interesting especially from a psycholinguistic point of view, but are beyond the scope of this paper.

7 Related research

One of the linguistic theories related to NLCA is the class of dependency-based models. A representant of them is Word Grammar (WG) [Hud84]. A major difference with WG lies in the fact that in NLCA a lexical unit can be involved in different types of relation at the same time. Unlike WG, there is no primary item that the structure of the sentence is based on. Furthermore NLCA identifies three relation schemes, where WG (and other dependency based models) has only one.

NLCA as a parsing method shows resemblance with the CKY algorithm, both being purely bottom-up. A closing step in NLCA is comparable to a 'deduction' step in CKY-parsing. The lexical information about the input lexical units is in the same complexity class as the size of the parsing table in CKY-parsing. This implies that the complexity of NLCA, like that of CKY, is polynomial.

8 Summary

The model presented exemplifies that structure in language can be derived from the (combinatorial) properties of lexical items and a set of general principles. The advantage of such an approach lies in its flexibility that makes it applicable to difficult syntactic phenomena like coordination, discontinuous structures and structural variations. An implementation of NLCA is currently under development.

References

[Hud84] R.A. Hudson. *Word Grammar.* Basil Blackwell Inc., Cambridge, MA, 1984.

[Kam97] V. Kamphuis. Coordination and Surface Structure. In B. Caron, editor, *Actes du Congrès International des Linguistes*, Paris, 1997. Elsevier Sciences (to appear).

[KS98] V. Kamphuis and J.J. Sarbo. Natural Language Concept Analysis. In D.M.W. Powers, editor, *Proc. of NeMLaP3/CoNLL98: International Conference on New Methods in Language Processing and Computational Natural Language Learning, ACL*, pages 205–214, 1998.