Interactive visualization of syntactic structure assembly for grammar-intensive first- and second-language instruction

Gerard Kempen
Cognitive Psychology Unit, Leiden University
& Max Planck Institute for Psycholinguistics, Nijmegen
kempen@fsw.leidenuniv.nl

Abstract

Understanding the grammatical structure of the target language is beneficial to certain categories of language learners, in particular to students learning to write in a language with many syntax-sensitive spelling rules, and to adult L1/L2 learners who prefer explicit rules. Moreover, grammar is an important body of knowledge in its own right.

Two visual-interactive grammar instruction tools are presented that graphically support students in composing and transforming sentences. The structure of sentences is displayed in the form of easily interpretable syntactic trees that the student can assemble and modify interactively. The tools check on-line the grammatical well-formedness of the structures being manipulated. The programs are based on the new Performance Grammar formalism (e.g., Kempen & Harbusch, 2002). They have been implemented in JAVA and can run under most modern operating systems.

1 Introduction

Few foreign-language teachers and learners consider grammar their favorite subject matter. Whether or not explicit grammatical knowledge helps attaining *oral* proficiency in a second language, is controversial. Many learners become fluent speakers and smooth understanders simply by practicing these skills, without having paid much attention to the grammar rules of the target language. However, this observation does not justify downplaying the importance of grammar in the acquisition of *written* language skills, neither in the mother tongue nor in a foreign language. Striking examples are provided by languages with large numbers of homophonous wordforms whose orthography is dictated by syntactic context. Writers in these languages, e.g. French and Dutch, are likely to commit numerous errors if they ignore syntax-sensitive spelling rules (Kempen & Dijkstra, 1994; Kempen, 1999).

Grammar rules underlying the syntax-sensitive aspects of verb form spelling tend to be rather abstract and difficult to understand and apply for primary and secondary schoolers. Consequently, negligible or even negative effects of explicit grammar instruction on L1 and L2 acquisition are unavoidable. This pessimistic conclusion, which is probably valid in the context of grammar instruction (if any) as practiced in many schools, has become a self-fulfilling prophecy. It has abated the interest of linguists and educationists in designing improved grammar didactics. The revolutionary developments in the cognitive sciences (linguistics, informatics, psychology) over the past decades indeed have hardly affected the teaching of grammar.

The outcomes of explicit grammar instruction in L1 and L2 curricula may be improved drastically by profiting from recent developments in the cognitive sciences. In this paper, I describe two interactive software tools for the on-line visualization and manipulation of syntactic structures. The first program is intended as an exercise tool in L1 courses on grammatical terminology ("sentence analysis"). The students can train the application of grammatical concepts by constructing easily understandable syntactic trees for sentences prepared by the teacher. The tool checks the correctness of the student responses on-line. The second program lets students compose new L2 sentences on the computer screen by assembling syntactic trees from partial trees ("treelets") associated with individual words. The tool immediately displays on-line the morphological and syntactic consequences of any assembly operation performed by the student. It refuses assembly attempts that yield ungrammatical results and can provide feedback on the reason for the refusal. It thus guides the student to the construction of grammatically correct L2 sentences.

Section 2 outlines some basic properties of the syntactic processor, the functionality of the tools, the user interfaces, and the status of the current implementations. Positive results of initial user evaluations are mentioned in Section 3.

2 Syntactic processor and user interfaces

2.1 Performance Grammar

Both tools are based on the Performance Grammar formalism (PG) developed by Kempen and Harbusch (Kempen & Harbusch, 2002; Harbusch & Kempen, 2002; Kempen & Harbusch 2003). Because PG trees have *n*-ary branching nodes, they tend to be "flat" — flatter than, e.g., the binary branching trees typical of Generative Grammar. Moreover, PG makes systematic use of functional concepts such as Subject, Direct Object, Head, etc., that are also used in traditional school grammars. Syntactic movement operations (e.g., "raising transformations") do not leave "traces" and are displayed graphically in a way that transparently brings out the source and target positions of moved constituents. Every word of the language is the head of a so-called lexical frame, that is, a treelet whose root can be merged (technically: unified) with a foot node of treelets associated with other words. This enables constructive exercises where students build sentences not by stringing words together but by assembling trees that automatically apply morpho-syntactic (hierarchical, word order) constraints. These features qualify PG as a formalism suitable for use in grammar teaching for linguistic novices.

2.2 PGW and PGT

The current computer implementation of PG is called the Performance Grammar Workbench (PGW). It covers a range of grammatical constructions of Dutch sufficiently wide to serve as the linguistic engine of a grammar teaching tool. Its graphical layer includes advanced tree drawing and manipulation functions capable of displaying "elastic" trees whose branches can be selected via mouse clicks and reordered from left to right without distorting the horizontal and vertical alignment of the nodes of the resulting tree.

Students can launch treelets by dragging words from a lexicon window to a workspace. Here, treelets and trees-under-construction can be picked up (on "mouse-down") and moved around freely. As soon as a root node hits upon a non-lexical foot node of another tree(let), the system checks whether or not the nodes are "unifiable", given the current configuration of morpho-syntactic features in the two structures. If so, the unification is realized when the student decides to drop ("mouse-up") the treelet at the current position. Otherwise, the node merger is refused and the tree(let) is dropped at a nearby position. If unification does take place, the system computes any morpho-syntactic implications for the shape of the enlarged tree and for the features of its node, and the new tree is drawn afresh. Any unification can be undone by picking up a branch of a treelet and pulling it away from the tree it belongs to. The shape of the trimmed tree and of the detached treelet as well as the features on their nodes are recomputed and set to the state that would have resulted, if the undone unification had never taken place before.

The second program, called Performance Grammar Trainer (PGT), may be viewed as a "light" version of the PGW. It does not incorporate a linguistic engine and can only display static PG trees. Teachers can define new sentence analysis ("parsing") exercises by graphically constructing syntactic trees with three types of nodes: word classes (parts of speech), word groups (phrases), and grammatical functions. Once stored, these trees serves as the criterion of correctness for students who are performing parsing exercises. Teachers can tune the PGT to the terminological preferences of their grammar curriculum. Moreover, they can adapt the trees to the proficiency level of the student by selecting which type(s) of nodes is/are displayed on the screen. Due to this flexibility, the program can be used as an add-on to many printed grammar curricula and is independent of the native language of the students.

2.3 Current implementations

Both the PGW and the PGT have been written in JAVA and run under most modern operating systems, either as autonomous programs or as applets within internet browsers. The Appendices contain sequences of screenshots that illustrate some of the functionality and the user interfaces of the PGT and the PGW.

3 Evaluation and conclusion

In each of two formal evaluation experiments, 18 undergraduate Dutch-language university students used the PGT during two 45-minute periods in order to refresh their high-school knowledge of grammatical concepts. The students were enthusiastic about the tool and produced considerable learning gains in this relatively short period of time. At the time of writing, no formal user evaluation of the PGW is available.

Despite the relatively small amount of experience with the tools, I conclude that PGW and

PGT represent a proof of concept showing that language technology can create useful tools for grammar-intensive L1 and L2 teaching. Whether such innovations can tip the balance towards higher efficiency of grammar-intensive language instruction as compared to the currently dominant communicative didactics, remains to be seen.

4 Acknowledgements

I am indebted to Camiel van Breugel and Robert Berg for implementing, respectively, the PGW and the PGT, and for their help in running the evaluation experiments. I also thank Freek Gertsen and Nard Loonen for constructing the PGT website for students of the Hogeschool Arnhem-Nijmegen (HAN) in Nijmegen.

References

Karin Harbusch & Gerard Kempen. 2002. *A quantitative model of word order and movement in English, Dutch and German complement constructions.* In "Proceedings of the 19th International Conference on Computational Linguistics (COLING-2002)", Taipei (Taiwan), pages 328-334, Morgan Kaufmann, San Francisco, California.

Gerard Kempen. 1999. *Visual Grammar: Multimedia for grammar and spelling instruction in primary education.* In "CALL: Media, design, and applications", Keith Cameron, ed., pages 223-238, Swets & Zeitlinger, Lisse, The Netherlands.

Gerard Kempen & Alice Dijkstra. 1994. *Toward an integrated system for grammar, writing and spelling instruction.* In "Computer-Assisted Language Learning. Proceedings of the Seventh Twente Workshop on Language Technology", Lisette Appelo & Francisca M.G. de Jong, eds., pages 41-46, University of Twente, Enschede, The Netherlands.

Gerard Kempen & Karin Harbusch. 2002. *Performance Grammar: A declarative definition.* In "Computational Linguistics in the Netherlands 2001", Anton Nijholt, Mariët Theune, & Hendri Hondorp, eds., pages 148-162, Rodopi, Amsterdam, The Netherlands.

Gerard Kempen & Karin Harbusch. 2003. *Dutch and German verb constructions in Performance Grammar.* In "Verb constructions in German and Dutch", Pieter Seuren & Gerard Kempen, eds., pages 185-221, Benjamins, Amsterdam.

Appendix A: Screenshots of the PGT

An exercise starts either with a "flat" presentation of the to-be-analyzed sentence, or with a tree whose node labels are missing. The latter alternative provides "scaffolding" and helps the student to become familiar with tree diagrams.

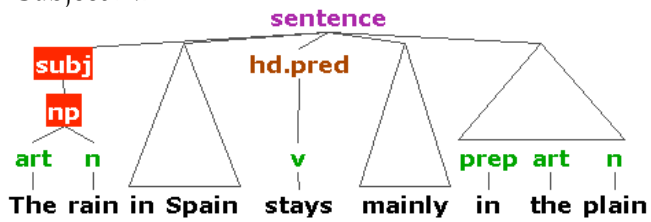
At each labeling attempt, the student is free to select one of four types of units: an individual word, a continuous string of words, an individual grammatical label, or a continuous string of labels. After that, s/he selects one option from one of three pop-up menus, one for each type of nodes. The PGT automatically positions the selected label in the tree. Errors can be restored at any time. The order in which the labels are assigned, is arbitrary.

Flat initial presentation of a sample sentence:

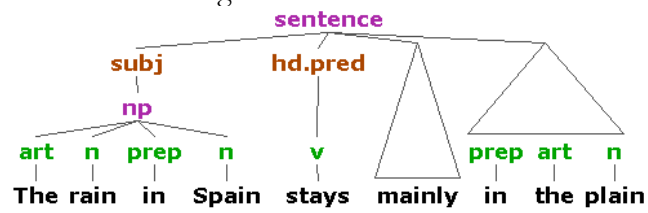
Menu options (three types of nodes):

Function	Phrase	Word class
subject	subclause	noun
direct object	noun phrase	adjective
indirect object	adjectival phrase	verb
prepositional object	prepositional phrase	article
head of predicate	adverbial phrase	preposition
nominal part of predicate	verb phrase	numeral
verbal rest of predicate		pronoun
modifier		adverb
subordinator		conjunction
particle		

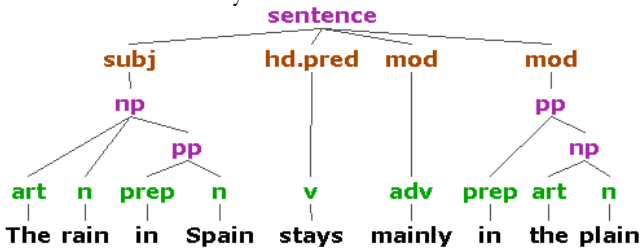
An intermediate solution stage with errors in Subject NP:



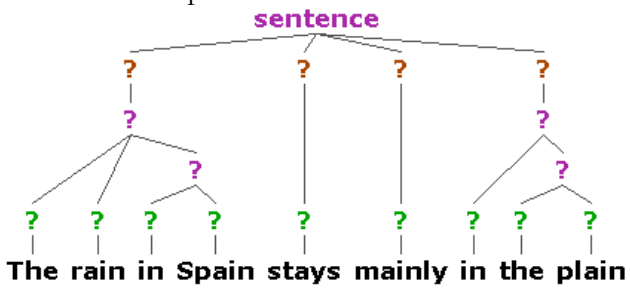
Intermediate stage with errors corrected:



Final correct analysis:



Alternative initial presentation of the exercise:

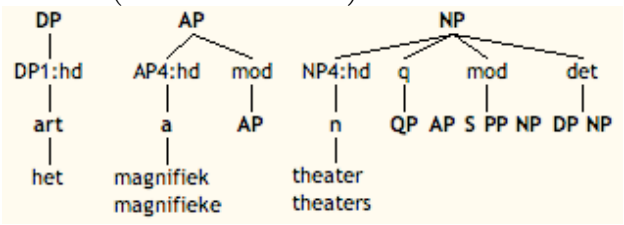


Appendix B: Screenshots of the PGW

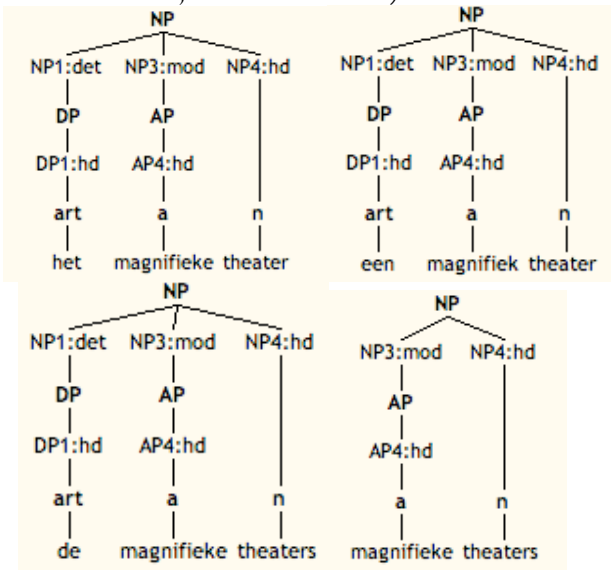
In this exercise, the PGW helps an L2 student of Dutch to compose sentence (1a) and its synonymous variant (1b) with Subject-Verb inversion.

(1) a. Hier zie je een magnifiek theater
Here see you a magnificent theater
'Here you see a magnificent theater
b. Je ziet hier een magnifiek theater

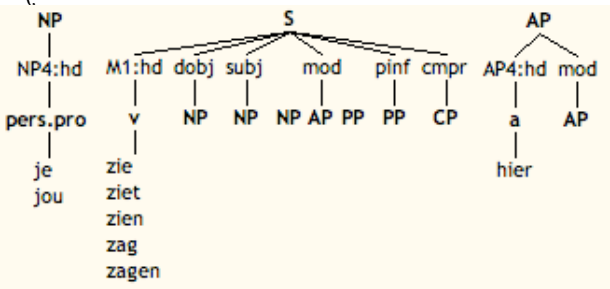
These are three treelets retrieved from the lexicon. Alternative wordforms are printed below the head branch. Numbered labels such as DP1, AP4, etc., have been computed by the linearization module (not discussed here).



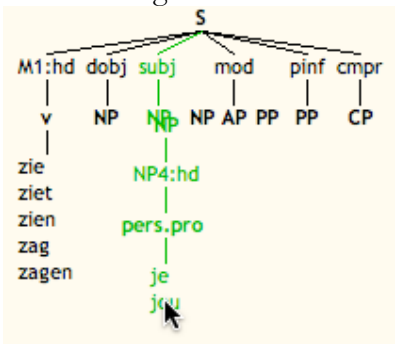
Upon unification, PGW's linguistic engine automatically selects the correct wordform(s) and word order(s). (*De*, *het* and *een* are articles. Notice inflection of adjective and noun.)



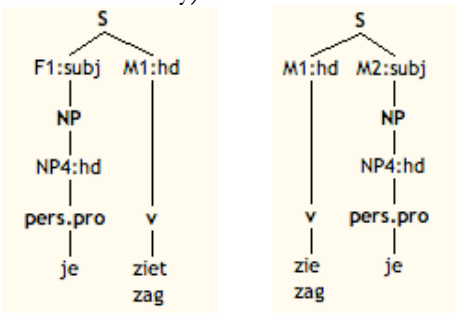
Here are the treelets for the remaining words. *Je* 'you' is nominative, *jou* is accusative. The verb *zien* 'see' has five finite forms.



In order to test whether two treelets can be combined, the student drags the root of one tree(let) over the target foot node of another:



This triggers a unification attempt by the PGW. Here unification succeeds, resulting in a reduced set of applicable wordforms and two ordering options (some branches have been pruned away for reasons of clarity):



Dragging the root NP of *een magnifiek theater* over the Direct Object foot node of the verb yields (1a) and (1b). Here is the tree for (1a), after superfluous branches have been pruned away, and without past-tense form *zag* 'saw'. (1b) is identical, except that Subject and Modifier have been been interchanged and that the verbforms differ.