A natural-language paraphrase generator for on-line monitoring and commenting incremental sentence construction by L2 learners of German

Karin Harbusch*, Gerard Kempen**, and Theo Vosse***

* *Computer Science Department, University of Koblenz-Landau, Germany*
harbusch@uni-koblenz.de
** *Max Planck Institute for Psycholinguistics, Nijmegen, The Netherlands*
gerard.kempen@mpi.nl
*** *Cognitive Psychology Unit, Leiden University, The Netherlands*
vosse@fsw.leidenuniv.nl

Certain categories of language learners need feedback on the grammatical structure of sentences they wish to produce. In contrast with the usual NLP approach to this problem—parsing student-generated texts—we propose a generation-based approach aiming at preventing errors ("scaffolding"). In our ICALL system, students construct sentences by composing syntactic trees out of lexically anchored "treelets" via a graphical drag&drop user interface. A natural-language generator computes all possible grammatically well-formed sentences entailed by the student-composed tree, and intervenes immediately when the latter tree does not belong to the set of well-formed alternatives. Feedback is based on comparisons between the student-composed tree and the well-formed set. Frequently occurring errors are handled in terms of "malrules." The system (implemented in JAVA and C++) currently focuses constituent order in German as L2.

1. Motivation

Certain categories of language learners, at varying levels of proficiency, rely on explicit L2 grammatical knowledge. This raises the question how and when ICALL (Intelligent Computer-Assisted Language Learning) systems should provide feedback on the grammatical structure of L2 sentences students wish to produce—for instance, in an essay writing exercise.

The usual NLP approach to this problem is based on *parsing*. After the student has typed a sentence, the parser evaluates it and provides feedback on the grammatical quality. The more errors a sentence contains, the less accurate the feedback is, due to the many correction options in the parser.

We propose a *generation*-based approach aiming at the prevention of errors ("*scaffolding*"). Students construct sentences incrementally, and the ICALL system intervenes immediately when they try to build an ill-formed structure. We use a natural-language sentence and paraphrase generator with a graphical drag&drop user interface.

The student drags words into a workspace where their grammatical properties are displayed in the form of syntactic treelets as defined in the lexicalized *Performance Grammar* (PG) formalism (Harbusch & Kempen, 2002; Kempen & Harbusch, 2003). In the workspace, the student can combine treelets by moving the root of one treelet to a foot of another treelet (see Section 3.2). In the generator, this triggers a unification process that evaluates the quality of the intended structure. If the latter is licensed by the generator's syntax, the tree grows and a larger phrase-structure tree is displayed. In case of licensing failure, the generator informs the student about the reason(s). This feedback follows directly from the unification requirements.

The system presented here focuses on constituent order in German as L2 and checks correctness of attempted orderings. Feedback is based on the (in)correctly applied L2 ordering rules. Additionally, typical errors due to intrusions from L1 (currently English) are handled by *malrules*. The paraphrase generator (briefly *paraphraser*) can provide the student with the correct ordering(s) on demand.

The paper is organized as follows. In the next Section, we outline the state of the art in ICALL systems for essay writing based on natural-language processing (NLP) techniques. In Section 3, we present our generation-based L2-learning system called COMPASS-II.[1] We first sketch the underlying grammatical formalism (PG). Then we describe the generator subserving the sentence construction process and show an example of feedback while the system is at work. In final Section 4, we take stock and discuss present and future work.

2. State of the art in ICALL writing tools

Computer-supported learning of how to write essays in L1 and L2 figures prominently in the ICALL literature. Due to space limitation, we cannot review systems working with canned texts. In-

[1] COMPASS is an acronym for COMbinatorial and Paraphrastic Assembly of Sentence Structure. "II" refers to an improved version, implemented in JAVA and C++, of the COMPASS system described by Harbusch *et al.* (2007).

stead, we focus the question of how the deployment of *natural-language processing (NLP) techniques*, in particular *parsing* and *generation,* can support students in writing novel sentences that are grammatically correct.

Virtually the entire literature on NLP applications to the syntactic aspects of first- and second-language teaching is based on parsing technology (Heift & Schulze, 2003). A *parser* computes the syntactic structure, possibly in combination with the semantic content of input sentences (provided that all words in the sentence are in the vocabulary, that all grammatical constructions are spelled out by grammar rules, and that the input does not contain any errors). However, all these systems struggle with ungrammatical input. They all have to take measures preventing the parsing quality from getting unacceptably poor. For example, in the *FreeText* system (L'haire & Vandeventer Faltin, 2003), the syntactic–semantic analysis is supplemented with *constraint relaxation* and *sentence comparison*. Other systems invoke matches with corpus texts (e.g. Granger, 2004). Yet another option is the addition of malrules to cover frequent errors (e.g. Fortmann & Forst, 2004). Another problem is caused by ambiguities. Hardly any sentence can be parsed unambiguously (cf. the proverbial *Time flies like an arrow,* for which Wikipedia lists no less than seven different interpretations). Hence, it is notoriously difficult to produce highly reliable feedback based on the parsing results.

To our knowledge, currently no *generator-based* software tool exists capable of evaluating the grammatical quality of student output. A generator produces a sentence or a set of paraphrases from an abstract representation of the content, often called *logical form* (see Reiter & Dale, 2000, for an authoritative overview of sentence and text generation technology). In the case of paraphrase generation, the generator delivers all possible ways of linguistically realizing the input logical form, given the lexicon and the grammar rules. However, virtually all natural language generation systems work in a best-first manner and produce only one output sentence rather than the set of all paraphrases. As it is not so easy to change the control structure of such a system, the choice of generators is very limited. Zamorano Mansilla's (2004) project is the only one that applied a sentence generator (KPML; Bateman, 1997) to the recognition and diagnosis of writing errors ("fill-in-the-blank" exercises). Zock & Quint (2004) converted an electronic dictionary into a drill tutor. Exercises were produced by a goal-driven, template-based sentence generator, with Japanese as the target language.

3. Incremental sentence production based on natural-language generation

In this Section, we first sketch the grammar formalism of COMPASS-II and its graphical user interface. Then, we describe the paraphraser and run a stepwise demo illustrating the system's feedback for a sentence under construction.

3.1 The Performance Grammar formalism

COMPASS-II is based on the *Performance Grammar (PG)* formalism, which is well suited to express fine-grained word-order rules in Dutch and German. Moreover, these rules can easily be tailored to other languages, in particular English. PG is a declarative syntax formalism where the hierarchical structure of a sentence is kept separate from its linear structure. PG's key operation is *typed feature unification*. Figure 1 illustrates an *elementary treelet* (also called *lexical frame)* for the wordform *Junge* 'boy'. The second layer represents grammatical functions (e.g., "hd" for *head*). Phrasal leave nodes (e.g. "ADJP" for *adjectival phrase* in the function of modifier) can be expanded by an appropriate treelet whose root node carries the same label (Figure 2).

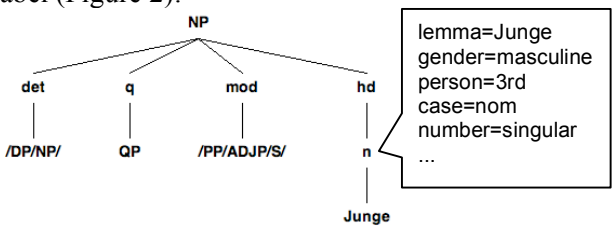


Figure 1. Elementary treelet for the lexical anchor *Junge*. The box associated with the wordclass of the head shows a subset of this node's morpho-syntactic features. Slashes represent alternative options.

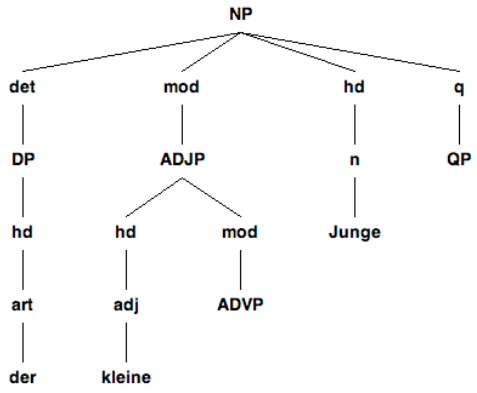
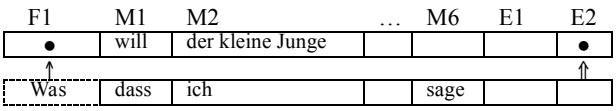


Figure 2. Well-formed tree for *der kleine Junge* 'the little boy': Appropriate DP and ADJP treelets have been unified at two leaves of the *Junge* treelet (cf. Figure 1). The Quantifier Phrase (QP) has no unification partner. Word order is not yet defined (see remainder of Section 3.1).

Associated with every treelet is a *topology*. Topologies serve to assign a left-to-right order to the branches of lexical frames. Here, we only illustrate

the topologies for verb frames (clauses).

(1) Was will der kleine Junge dass ich sage?
what wants the little boy that I say
'What does the little boy want me to say?'



The slot labeled F1 makes up the Forefield (from German Vorfeld), M1-M6 the Midfield (Mittelfeld), and E1 and E2 the Endfield (Nachfeld). Every constituent (subject, head, direct object, complement, etc.) has a small number of placement options, i.e. slots in the topology associated with its "own" clause. For instance, the finite verb of a main clause goes to M2 whereas in subordinate clauses it goes to M6.

How is the Direct Object NP *was* 'what' "extracted" from the complement clause and "promoted" into the main clause? Movement of phrases between clauses is due to *lateral topology sharing*. If a sentence contains more than one verb, each of their lexical frames instantiates its own topology. This applies to verbs of any type—main, auxiliary or copula. In such cases, the topologies are allowed to *share* identically labeled lateral (i.e. left- and/or right-peripheral) slots, conditionally upon several restrictions (not to be explained here; but see Harbusch & Kempen, 2002). *After two slots have been shared, they are no longer distinguishable; in fact, they are unified and become the same object.* In example (1), the embedded topology shares its F1 slot with the F1 slot of the matrix clause. This is indicated by the dashed borders of the bottom F1 slot. Sharing the F1 slots effectively causes the embedded Direct Object *was* to be *preposed* into the main clause (black dot in F1 above the single arrow in (1)). The dot in E2 above the double arrow marks the position selected by the finite complement clause (cf. Figure 3).

3.2 "Scaffolded" writing with COMPASS-II

The paraphrase generator of COMPASS-II can produce all linear order variants licensed by the most important word order rules of German. The generator takes as input tentative syntactic trees constructed by the student through a graphical direct-manipulation ("drag&drop") user interface.

The student drags words into a workspace where their grammatical properties are displayed in the form of PG treelets (cf. Figure 1). In the workspace, the student can combine treelets by moving the root of one treelet to a foot of another treelet. In the paraphraser, this triggers a unification process that evaluates the quality of the intended structure. If the latter is licensed by the paraphraser's syntax, the tree grows and the resulting larger tree is displayed (cf. Figure 2).

In case of licensing failure, the paraphraser informs the student about the reason(s). This feedback follows directly from the unification requirements. For instance, when a student tries to unify the genitive article *des* with the DP leaf node of the nominative noun *Junge*, the *des* treelet would refuse to be unified, thus warning the student that there is a feature mismatch.

Moreover, the system calculates on demand all possible correct sentences—in particular all word-order variations. This action is triggered when the student orders the branches from left to right—either by dragging nodes around or by editing the leaf string, which appears in a special word-order window. (This also allows, among other things, to agglutinate *zu dem* 'to the' to *zum*.) Figure 3 displays the resulting order for sentence (1) in terms of the topological slot positions defined in PG. Currently, we show such trees only to advanced students. In order to tailor the feedback to the level of a beginner, we will revise this window to give more verbose tutoring feedback.

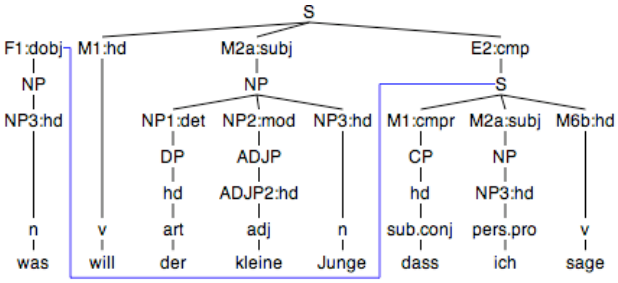


Figure 3. Ordered tree spelling out the topological array positions for sentence (1) (*was* goes to top level F1).

Moreover, COMPASS-II runs a small set of *malrules* that derive from typical errors users make. These malrules can be spelled out for unification and for word ordering. For instance, the erroneous string *der kleiner Junge* is "accepted" by the system but triggers a negative feedback message. (The correct string is *der kleine Junge*; the confusion arises from the correct *ein kleiner Junge* 'a little boy'.).

With respect to word order, malrules refer to typical differences between L1 and L2. For instance, one rule "allows" ungrammatical verb-second word-order in German subordinate clauses (most of which are clause-final rather than verb-second), but it triggers an error message if the student-produced sentence conforms to it. Figure 4 displays the overall system at work for clause (2) where the student uses an English word-order rule.

(2) *Heute Anja baut eine Rakete*
Today Anja builds a rocket

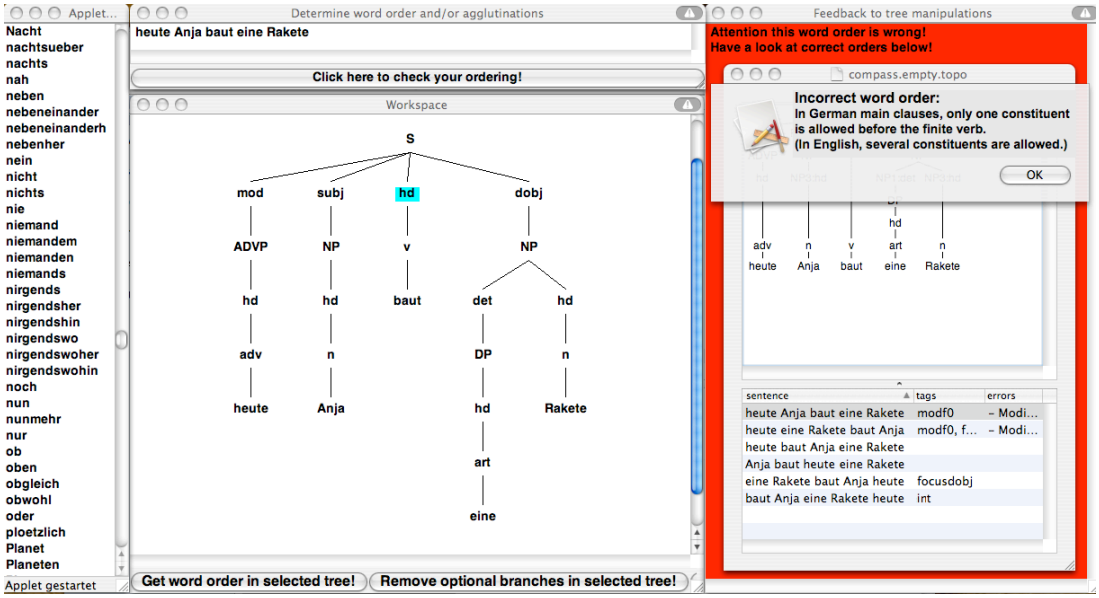


Figure 4. Snapshot of the first system response to the incorrect word order in sentence (2).

The left window shows the system's *word list*. Lexical frames (treelets) corresponding to selected words appear in the big window in the middle column. This window is the workspace where the student can combine and edit the selected treelets. The resulting leaf strings are automatically shown in the small window at the top. The student can edit these strings by typing or cut&paste. A button in this window ("Click here ...") activates paraphraser and word-order checks. The right panel immediately provides the feedback (here in red, signalling an error). The six sentences at the bottom of this panel (currently not shown to students) enumerate all orders (including those yielded by a malrule; cf. (2)).

4. Discussion

We view the current version of the COMPASS-II as the prototype of an "engine" that can drive the automatic evaluation and diagnosis of sentences produced by L2 students of German. Applying the system in the classroom will require tailoring it to the requirements imposed by specific exercises and specific student populations.

In additional future work, we may target another class of constructions that are problematic for L2 students of German: *elliptical forms of coordinate structures*. Most of the linguistic and computational groundwork for a PG treatment of these ellipsis types in German has been laid (Kempen, in press; Harbusch & Kempen, 2006, 2007) and provides a suitable starting point for an ICALL application.

References

Bateman, J.A. (1997). Enabling technology for multilingual natural language generation. *Natural Language Engineering*, 3:5-55.

Delmonte, R., Delcloque, P., &Tonelli, S. (Eds.) (2004). *Procs. of InSTIL/ICALL2004 Symposium,* Venice.

Granger, S. (2004). Computer learner corpus research: Current status and future prospects. In Connor, U., & Upton, T. (Eds.). *Applied Corpus Linguistics: A Multidimensional Perspective*. Amsterdam: Rodopi.

Fortmann, C., & Forst, M. (2004). An LFG Grammar Checker for CALL. In (Delmonte *et al.,* 2004).

Harbusch, K., & Kempen, G. (2002). A quantitative model of word order and movement in English, Dutch and German complement constructions. In *Procs. of 19th COLING*, Taipei.

Harbusch, K., & Kempen, G. (2006). ELLEIPO: A module that computes coordinative ellipsis for language generators that don't. In *Procs. of 11th EACL*, Trento.

Harbusch, K., & Kempen, G. (2007). Clausal coordinate ellipsis in German. In *Procs. of 16th NODALIDA*, Tartu.

Harbusch, K., Kempen, G., van Breugel, C., Koch, U. (2006). A generation-oriented workbench for Performance Grammar. In *Procs. of 4th INGL,* Sydney.

Harbusch, K., van Breugel, C., Koch, U., & Kempen, G. (2007). Interactive sentence combining and paraphrasing in support of integrated writing and grammar instruction. In *Procs. of the 11th ENLG,* Dagstuhl.

Heift, T., & Schulze, M. (Eds.) (2003). Error diagnosis and error correction in CALL. *CALICO Journal, 20*.

Kempen, G. (in press). Clausal coordination and coordinate ellipsis in a model of the speaker. *Linguistics*.

Kempen, G., & Harbusch, K. (2003). Dutch and German verb constructions in Performance Grammar. In Seuren, P.A.M., & Kempen, G. (Eds.), *Verb Constructions in German and Dutch*. Amsterdam: Benjamins.

L'haire, S., & Vandeventer Faltin, M. (2003). Error diagnosis in the FreeText project. In (Heift & Schulze, 2003).

Reiter, E., & Dale, R. (2000). *Building applied natural language generation systems*. New York: Cambridge University Press.

Zamorano Mansilla, J.R. (2004). Text generators, error analysis and feedback. In (Delmonte *et al.,* 2004).

Zock, M., & Quint, J. (2004). Converting an Electronic Dictionary into a Drill Tutor. In (Delmonte *et al.,* 2004).