Error Recognition and Feedback with Lexical Functional Grammar

Veit Reuer
Institute of Cognitive Science
Universität Osnabrück, Germany

ABSTRACT

This paper describes the error recognition module of an interactive ICALL system with a special focus on the underlying grammar theory. Using the system, language learners are invited to produce complete written sentences in small question-answer dialogs with the computer. This setting challenges learners to use language interactively in order to enhance the development of communicative competence. Emphasis is put on the possibility of giving adequate feedback to the learner if a syntactically ill formed sentence is encountered. It is argued that the theory of Lexical Functional Grammar (LFG) is well suited to be used in the parsing and error recognition module of the system as well as to provide intelligent feedback to learners. The concepts and structures used in LFG closely resemble the descriptive knowledge of language learners about a language, and, therefore, the results of an automatic analysis can easily be translated from a computationally tractable form to language easily understood by the learner.

KEYWORDS

Lexical Functional Grammar (LFG), Intelligent Computer-Assisted Language Learning (ICALL), Error Recognition, Parsing, Feedback

1. INTRODUCTION

There has been some successful research on recognizing errors made by learners of a foreign language with methods of computational linguistics (e.g., Schwind, 1988; Heift, 1998; Schneider & McCoy, 1998). Some advantages of Intelligent Computer-Assisted Language Learning (ICALL) systems that incorporate advanced error recognition are:

1. An intelligent system can provide precise feedback about erroneous input by learners.
2. Analyses can be used for learner modeling and individual tutoring.
3. Dialog-style exercises with almost free input can be implemented.

This paper describes the error recognition module of an interactive ICALL system with a special focus on the underlying grammatical theory. Using the system, language learners are invited to produce complete written sentences in small dialog tasks with the computer. This setting challenges learners to use language interactively in order to enhance the development of communicative competence. Emphasis is put on the possibility of giving adequate feedback to learners if a syntactically ill formed sentence is encountered by the system. As mentioned in Menzel and Schröder (1998) for example, programs usually do not support both requirements at the same time. If free formed input is allowed, the system is not able to perform a detailed analysis of the input; whereas if the error recognition capabilities are advanced in order to provide satisfying feedback, the choice of exercises may be quite limited. Thus, the main goal in developing a CALL system should be to provide a stimulating environment and to give adequate feedback. The preconditions are automatic syntactic analysis of learner input (parsing) and error detection, both of which are handled by the method described below. Error detection is based on the algorithm of the parser and not on information encoded in the grammar or the lexicon, that is, the parser does not anticipate errors.

The aim of the system described is to avoid error anticipation in the grammar and the lexicon and to use a sound grammar theory, namely Lexical Functional Grammar (LFG) (Bresnan, 1982; Dalrymple, Kaplan, Maxwell, & Zaenen, 1995), which has been used successfully to describe various languages and linguistic phenomena. Furthermore, LFG is based on concepts which can easily be implemented in an efficient parser.

The first part of this paper describes the methods used for the analysis of learner input and which types of errors can be detected. In addition, based on the functional/lexical perspective, concepts used in LFG are possibly easier to understand by learners than the ones used in other linguistic theories (e.g., see Schwarze, 1995 for a descriptive Italian grammar based on LFG). Hubbard (1994) presents arguments for considering function-based grammar theories in a language learning scenario. The second part of this paper argues for the suitability of LFG to explain syntactic concepts and errors contained in learner input.

2. ERROR RECOGNITION

Recent research in error recognition includes that of Heift (1998), Heift and Nicholson (2001), and Schneider and McCoy (1998). In these instances, the parsing process is supported by a modified or even specially designed grammar and lexicon. Certain kinds of marking are added to the grammar to allow for the detection of errors. This in turn forces the linguist designing the grammar to foresee all possible errors that may occur in any given learner input. However, the anticipation of errors in the grammar or the lexicon also has some advantages. One advantage is the possibility to use fast algorithms for obtaining a description of the sentence. A second is the possibility for such a system to distinguish between input that contains an error and input that cannot be parsed because the grammar does not cover the construction.

In a second type of approach, more or less costly parsing strategies are implemented for error identification: Mellish (1989) and an improved version in Kato (1994) or Menzel and Schröder (1998) are examples of this approach. While these systems are able to identify almost any error, the drawback in Mellish's and Kato's approach, for instance, lies in two parses of the same string in case of erroneous input. The dependency-based grammar formalism used by Menzel and Schröder only allows for binary constraints, which is rather artificial for some linguistic structures, even though the expressive power of the formalism is not limited by this restriction. Another disadvantage of this type of approach is that it is difficult for the system to distinguish between erroneous input and input not covered by the grammar. Therefore, feedback on free input has to be presented with caution.

The implementation of the parser described below uses a completely unrestricted (i.e., standard) grammar and lexicon. However, in order to make the parser more efficient, some linguistic knowledge is used for the generation of error hypotheses, as will be explained below. Based on the analysis of a learner corpus of German as a foreign language marked with error types, the parser was adapted to identify the most frequent errors. Therefore the parser presented here does contain some linguistic knowledge from this analysis to constrain the search space but does not contain buggy or mal-rules that are inserted in the grammar for describing erroneous sentences.

LFG is based on two main structure types used for describing the syntactic properties of a sentence. Phrase-structure rules (ps-rules) provide the means for describing word-order phenomena (c-structure), and feature structures (f-structures) are built up via annotations to the ps-rules for the functional description of a sentence. Syntactical errors are detected in both areas of this parallel grammar theory. In the following, these issues will be considered in turn.

Feature structures or attribute-value-matrices (AVMs) are used in LFG for the encoding of agreement, subcategorization, and other morpho-syntactic features. If the values of two features—which should undergo unification—conflict, one value is retained at the original position and the other is added to the resulting structure as a value of an error feature. Unification is here the operation of merging two AVMs into one matrix. The unification module for building up f-structures contains a list of features which restricts the error location in a given f-structure mainly to agreement-features. This corresponds to a two-valued constraint ranking: either the values of corresponding attributes may clash and the error location and values are stored in the f-structure, or the attributes are considered "hard" and the unification fails in case of an error. Note that this approach differs from ideas like those of Vogel and Cooper (1995) and Carpenter (1993) which are also able to handle so-called "clashing values" but have no mechanism to store the information which values did not actually match. However, in a language learning scenario, this is crucial to provide adequate feedback to learners. This method is explained in detail in the next section.

For the processing of ps-rules, an extended version of Earley-based chart parsing is used which is known to be relatively fast (Earley, 1970). Here the parsing is restricted to identifying an error only when there is an indication for this in the chart. In the domain of linearization (i.e., c-structure), four types of errors can be distinguished: insertion, omission, misplacement, and spurious replacement. So far methods for the recognition of misplacements and omissions have been implemented. In order to identify omissions, the chart-parsing component uses a part-of-speech (POS) list consisting mainly of "functional" categories that restricts possible insertions of apparently omitted items in the sentence. These are then inserted into the chart as "hypothetical" elements to correct a sentence.

Misplacements are handled according to the state of the chart by putting the sequence of words into a storage for later insertion at the correct position. Spurious or superfluous insertions and replacements are not yet recognized by the parser; however, insertions account only for a small percentage of all errors made.

The type of error is stored as an annotation to the chart item. Thus, when the final item providing a description of the complete sentence is generated, the annotation can be used for feedback. The recognition of omission errors is similar to the approach in Lee, Kweon, Seo, and Kim (1995). However, the concept described here also includes the frequent type of misplacement errors and is tailored towards identifying errors fast as opposed to merely robust parsing. As will be explained later, the identification of insertion should not be a problem to integrate.

Using this concept, more than one structure may result from the parsing process. Depending on the number of clashing values and the type of error, an error value is assigned to each chart item. For example, a simple agreement error will result in a lower error value than a missing word. The error value is then used to determine which structure to use for further analyses of the learner's input. In the parsing process chart, items with a lower error value are preferred, and there is an upper limit to the error value.

3. MODIFIED UNIFICATION

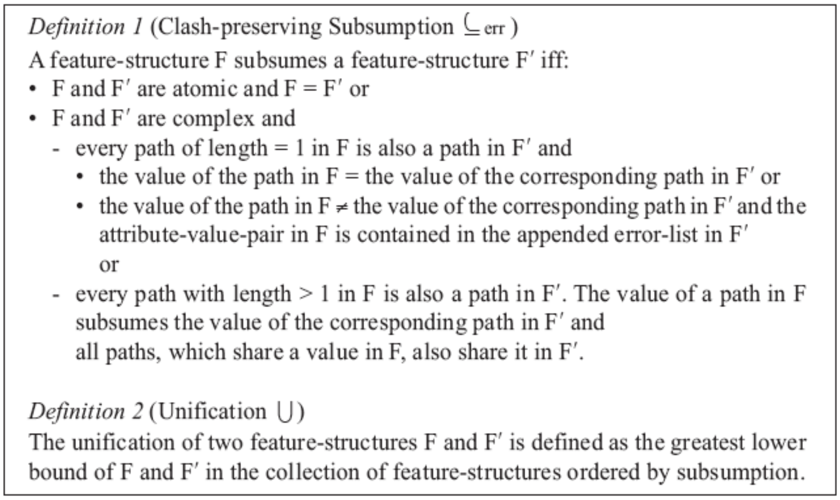
This section describes the mechanism that is employed in locating and storing errors found in the f-structure. As mentioned above, in LFG these structures contain mainly agreement information, but they may also contain functional and subcategorizational information as well. (1)b is a representation of the sentence given in example (1)a.

(1)a. I hit him.

b.
$$\begin{bmatrix} \text{modus:} & \text{ind} \\ \text{fin:} & + \\ \text{pred:} & \text{HIT(subj, obj\_dir)} \\ \text{subj:} & \begin{bmatrix} \text{case:} & \text{nom} \\ \text{num:} & \text{sg} \\ \text{pers:} & \text{I} \\ \text{ntype:} & \text{prn} \end{bmatrix} \\ \text{obj\_dir:} & \begin{bmatrix} \text{case:} & \text{acc} \\ \text{num:} & \text{sg} \\ \text{pers:} & \text{III} \\ \text{ntype:} & \text{prn} \end{bmatrix} \end{bmatrix}$$

The first two types of features, in particular, can be used to identify certain types of errors and to provide feedback to learners about these errors. The unification algorithm has been changed to incorporate information about clashing values in the f-structure in order to inform learners precisely about their error. The idea is to add a list of attribute-atomic-value pairs to every feature structure that resulted from a unification of mismatching values. This idea resembles the notion of adding a disjunction of values to those attributes, but it does not rely on it. Definition (1) extends the standard definition of subsumption of atomic and complex values (paths of length = 1 and paths of length > 1)

Definition 1
Subsumption and Unification



Definition 1 (Clash-preserving Subsumption $\sqsubseteq_{err}$)
A feature-structure F subsumes a feature-structure F′ iff:
- F and F′ are atomic and F = F′ or
- F and F′ are complex and
  - every path of length = 1 in F is also a path in F′ and
    - the value of the path in F = the value of the corresponding path in F′ or
    - the value of the path in F ≠ the value of the corresponding path in F′ and the attribute-value-pair in F is contained in the appended error-list in F′
    or
  - every path with length > 1 in F is also a path in F′. The value of a path in F subsumes the value of the corresponding path in F′ and
    all paths, which share a value in F, also share it in F′.

Definition 2 (Unification $\sqcup$)
The unification of two feature-structures F and F′ is defined as the greatest lower bound of F and F′ in the collection of feature-structures ordered by subsumption.

This definition of subsumption has some of the relevant properties of standard subsumption: reflexivity and transitivity. It is, however, not completely anti-symmetric: Two AVMs may subsume each other and may not be equal in the standard sense but only equivalent. As a result, we obtain an equivalence class of AVMs where a certain attribute-value-pair may be contained either in a corresponding f-structure or in the appended error-list and vice versa for another member of the class. The following is an example.

(2) $$[F:a] \cup [F:b] = \left\{ \left[\frac{F:a}{err:[F:b]}\right], \left[\frac{F:b}{err:[F:a]}\right] \right\}$$

Following the definition of subsumption, two f-structures result from the unification in an equivalence class. Both variants are subsumed by F and F′ and both variants subsume each other. Since the result of unification may be an equivalence class of two f-structures, that is, each f-structure subsumes the other and therefore contains (almost) the same amount of information, one of the structures is discarded. Also, the possible problem of multiple clashing attributes that generate n^2 members in the equivalence class is thus avoided. However, the information from which f-structure which attribute-value-pair originated is lost, but it can easily be recovered after the parse by traversing the resulting phrase-structure tree downwards until the node from which the error originated is reached. Note that this has only been defined for paths of length 1 and is used only for the generation of f-structures through unification. Additionally, this method of recognizing errors has the advantage of preserving the strict monotonicity of unification and of the grammatical theory.

At first glance, this is especially suited to identifying agreement errors. The head of a phrase calls for certain properties of the complement, which are possibly not met by the complement. However, in addition, other features like finiteness can fail and therefore provide information about this type of error.

In the actual implementation, even more information is stored in the appended error list. For every new clash and the addition of an element to the error list, a single attribute is added to the error list on the next higher level if it exists. If paths contain complex values with clashes, then the feature structure containing this path will also be marked for errors by putting the single path attribute into the error list.

As a consequence, the completed f-structure of a sentence contains information about the error that occurred while unifying inside a functional element or while unifying the functional element with the head. The following unification illustrates this process.

(3)

$$\left[\texttt{subj: }\left[\texttt{pers: III}\right]\right] \cup \left[\begin{array}{l}\texttt{pred: PRED(subj)}\\ \texttt{subj: }\left[\texttt{pers: I}\right]\end{array}\right] =$$

$$\left[\begin{array}{l}\texttt{pred: PRED(subj)}\\ \texttt{subj: }\left[\begin{array}{c}\texttt{pers: III}\\ \hline \textit{err}\texttt{:[pers:I]}\end{array}\right]\\ \hline \textit{err}\texttt{\{ subj]}\end{array}\right]$$

In example (3), the subject is marked for [pers:III] and the verb subcategorizes for a subject with [pers:I]. This information conveys that no error occurred inside the subject (e.g., det-noun mismatch) but that the error lies between the NP and the verb. The error list in the outer feature structure would not be present otherwise. Using this mechanism, additional measures like case filtering proposed by Schwind (1988) are unnecessary for the identification of an error.

If learners have chosen the wrong valency of a verb, the error is detected by the implementation of the completeness and coherence conditions of LFG given in definition (2).

Definition 2
Completeness and Coherence

Definition 2 (simplified Completeness and Coherence in LFG)

An f-structure is *complete* iff it contains all the governable grammatical functions that its predicate governs, and an f-structure is *coherent* iff all the governable grammatical functions that it contains are governed by a predicate.

The completeness and coherence are checked at the end of the parsing process when no more information can be added to the resulting f-structure. Depending on the value of the "pred" attribute, the f-structure is checked for the existence of functional attributes with complex values. In example (3) the verb subcategorizes only for a subject which must be present in the f-structure. Missing or additional functional attributes point to a valency error by the learner.

4. EXTENDED EARLY PARSING

Another important area where errors occur is linearization, which in LFG is coded in the c-structure. One classical approach that does not use error anticipation, and is thus very general, was developed by Mellish (1989). An improved version is presented in Kato (1994). In both cases, the sentence is initially chart-parsed bottom-up until no further items can be found. After this procedure, a top-down process is employed using mainly the chart items to find the position and the type of error. Kato's improvement concerns the structure of the chart items and, from that, the use of a bidirectional parser in the first step. For a language learning environment, this seems too general because errors tend to occur only in restricted domains. Also the use of two parsing strategies following one another seems too costly for the task at hand. This is especially relevant when using LFG, where the ps-rules are annotated with constraints which may restrict the use of a rule at a very late stage (i.e., high up in the tree). Another point of criticism is the difficulty with multiple errors at which point Mellish's approach will produce unanchored items in the chart.

The approach described here for the recognition of linearization errors is an extension of the SHIFT-predicate (or 'scanner' as sometimes referred to) of the Earley chart-parsing algorithm (Earley, 1970). Using this algorithm, hypotheses are added to the top-down-based chart on the category of the current word. An item in the chart may either be active or passive, that is, all hypotheses may be fulfilled or not (see below). If the next word cannot be connected to a bottom-up hypothesis, then the parser encountered an error. In this case, it is assumed that either a word is missing or a phrase is misplaced. Both possibilities are tested, and the chart is checked for any progress.

Three cases can be distinguished after the standard SHIFT predicate and the CLOSURE predicate have been called. An 'item' contains the common variables: _begin, _end, _lhs, _closed and _open, which mark the start position of the string, the end position of the string covered so far, the left hand side of a ps-rule, the elements already found and the elements still looked for of the right hand side of the ps-rule.

case 1
The current word is included in passive chart items and there are hypotheses for the following word.

case 2
The current word could not be connected to any active item, that is, there is no item spanning the current word and having an open element:
```
not(item(_,_end,_,_,[_open|_]))
```

case 3
Active items have been put into the chart spanning but neither the current word nor any of the hypotheses can be linked to the following word:
```
not(item(_,_end,_,_,[_open|_]),
   link(_open,_nextword))
```

These mechanisms have been adopted for the end of a sentence where no word follows. If a single passive item spanning the complete sentence is not present, the parser looks for the longest chunk starting from the end of the sentence. The beginning of this largest chunk marks the position where, probably, an error occurred.

Example 4 illustrates an instance of error detection. In this case, a determiner has been left out. The parsing process starts by adding items to the chart that contain hypotheses about the possible beginning of a sentence. Again, an item stores the information: from, to, left_side, found_element, expected_element, and error_type_position.

(4) Sentence: $_0$ Dog $_1$ barks $_2$.

PS-Rules: `S → NP VP        NP → det n       VP → v`

Chart:

```
item(0,0,S,    [],    [NP,VP],  [])
item(0,0,NP,   [],    [det,n],  [])
item(0,1,n,  [DOG],     [],       [])            (case 2)
item(0,0,NP, [det],    [n],    [ins,det,0])     (Omission)
item(0,1,NP,[det,n], [],       [ins,det,0])     (Continuation)
```

Since the grammar consists of only three ps-rules, only two hypotheses need to be added to the chart until the problematic situation is encountered. There is only one hypothesis for a determiner, and the first element in the input string is a noun (“dog”). Therefore a “virtual” determiner is added to the chart which also carries a marking for the type of correction. This allows the parsing process to continue until an item is created with S as the left_side, no expected_element, and spanning the complete sentence (a “passive” item spanning the complete sentence).

In case a phrase is placed at the wrong position, an intermediate parse has to determine the size of the phrase that needs to be moved to a different position. A common error by learners of German, for example, is the placement of the subject before the finite verb even though an adverb is the first element of the sentence (example [5]). In this case, the subject should be moved to the position immediately following the finite verb (example [6]).

(5)	**Jetzt*	*ich*	*sehe*	*dich.*
	Now	I	see	you.
(6)	*Jetzt*	*sehe*	*ich*	*dich.*
	Now	see	I	you.

Due to the restrictions of German word order, the parser determines that the NP *ich* must be removed from the sentence and inserted at a later position (i.e., directly after the verb). The previous word is “extended” in the chart to cover the gap in case the repositioning of the phrase is successful. Since the NP could also consist of more than one word, a parse has to be used to identify this phrase as the longest possible complex phrase starting at the error position.

With the help of these mechanisms a large number of errors can be detected and reported to learners. The f-structure contains information about agreement and subcategorization in addition to “verbal” features such as finiteness. Error markers for these features are integrated into the f-structure and can be used for precise feedback. This will be explained in detail in the next section.

With respect to errors in linearization, only two types of errors are recognized so far, but these types represent the most frequent linearization errors in some learner corpora (e.g., Heringer, 1995). Here the chart as the main database is changed in such a way as to correct the input string to a sentence that can be parsed successfully. The changes are stored in the corresponding chart items for further evaluation.

5. LFG AS EXPLANATORY BASIS

This section provides a rationale for using LFG in a CALL environment. A grammatical theory for the automatic analysis of linguistic structures such as LFG can be chosen for a variety of reasons: explanatory power, efficiency in computing algorithms, ease of development, and so forth.

First, there have been some implementations which are working quite efficiently even for large-coverage grammars such as the *Xerox Workbench* (Kaplan & Maxwell, 1993) or the XLFG environment (Clément & Kinyon, 2001). In both environments rather large-scale grammars have been developed. Butt, Dipper, Frank, and King (1999) report on the efforts made in the ParGram project in which parallel grammars for German, English, and French were developed.

Second, LFG has been proven to successfully describe morpho-syntactic phenomena for a large variety of languages. One of the important aspects captured by LFG is the difference between configurational languages with strict word order and non-configurational languages with more or less free word order (see Austin, 2001). The key point is that the different structures in LFG, namely f- and c-structures resemble this difference. The f-structures in different languages stay the same for sentences with the same functional structure, whereas the c-structures vary according to the rules of the single language. In the ParGram project, this idea was used for the description of the three languages. The f-structure representing a certain functional structure (and, therefore, a certain meaning) in one language is the same for all languages. However, the c-structure changes from language to language. This can be seen as one of the main reasons for the linguistic adequacy of LFG as a framework for linguistic description.

Third, and most important, LFG theory seems to be quite close to the concepts used in teaching grammar. Hubbard (1994, pp. 59-60) identifies three main implications derived from the general assumptions of LFG for language teaching:

1. Unlike transformational grammar and its direct descendants, grammatical functions are primitives. Thus, [...] it is reasonable to state pedagogical rules using them.
2. The lexicon is a central part of the grammar, not just dumping place for exceptions to rules. Lexical rules relate verb forms of active and passive

to one another [...] This binds the learning of vocabulary and grammar more closely than is usually done in grammar texts, and certainly more so than in texts inspired by structural or earlier transformational approaches.

3. Similar, in [LFG] the content of a lexical entry is much more than a phonological form and one or more meanings: It includes all of the subcategorization information for constituent and functional structure. This suggests that the learning of vocabulary, particularly verbs, should include as much of this subcategorizational information as possible. For example, it is not enough for students to memorize lists of verbs taking infinitive versus gerund complements if other important aspects of subcategorization, such as whether or not the gerund or infinitive is preceded by an NP, are ignored.

Here, two major aspects need to be emphasized. The functional perspective on language is not only present in LFG but is also widely used in grammar teaching. Notions such as "subject" or "direct object" are used in LFG where they represent central concepts as in pedagogical grammars. Also, the "topological" view on German word order with concepts such as *Vorfeld*, *Mittelfeld*, and *Nachfeld* have been realized in LFG for German (Clément & Kinyon, 2002). This type of structuring has been widely used in pedagogical grammars for German (see Kars & Häussermann, 1997; Eisenberg, 1999). Therefore, feedback derived from a LFG-based parser analysis need not be "translated" but can be passed on directly to students.

The second aspect mentioned by Hubbard is the strong reliance on the lexicon as a resource of information. This can also be found in descriptive pedagogical grammars, where word-order and subcategorizational phenomena are first explained in general, and then single lexemes are used to provide examples for the realization of the phenomena. Therefore, learners should memorize all relevant information about the syntactic behavior of lexemes instead of learning only the lexemes and their translations.

In the context of a CALL system with advanced feedback capabilities, it is important that the messages to users can be tailored to their level of understanding of linguistic terminology (see Heift, 2001). A grammatical theory adopted in a system should therefore support the generation of messages that are easy to understand. As mentioned above, the concepts used in LFG (can) have a strong resemblance to the terminology used in traditional or pedagogical grammar teaching. The following f-structure (7)b of a simple sentence demonstrates the way in which a pedagogical explanation of the structural properties can be extracted from the description. For the sake of clarity, some morpho-syntactic features, especially those of the complement, have been omitted.

(7)a. I have seen a car crash into a traffic light.

(7)b.

$$\left[\begin{array}{ll} \text{auxtype}: & \text{have} \\ \text{fin}: & + \\ \text{pred}: & \text{SEE(subj, comp)} \\ \text{subj}: & \left[\begin{array}{ll} \text{case}: & \text{nom} \\ \text{num}: & \text{sg} \\ \text{pers}: & \text{I} \\ \text{ntype}: & \text{prn} \end{array}\right] \\ \text{comp}: & \left[\begin{array}{ll} \text{fin}: & - \\ \text{pred}: & \text{CRASHINTO(subj, obl}_{\text{into}}\text{)} \\ \text{subj}: & \left[\begin{array}{l} \text{pred}:\text{CAR} \\ \text{case}:\text{nom} \end{array}\right] \\ \text{into}: & \left[\begin{array}{ll} \text{obj}: & [\text{pred}:\text{TRAF\_LIGHT}] \\ \text{pred}: & \text{INTO(obj)} \end{array}\right] \end{array}\right] \end{array}\right]$$

A pedagogical explanation which can easily be derived from this structure might have the following form: The sentence consists of a main verb "see" which is used with the auxiliary "have." The verb "see" subcategorizes for an infinitive complement that requires a subject. The verb "crash" contains the subject "a car" and a prepositional phrase with "into" as its head. The preposition "into" subcategorizes for the noun phrase "a traffic light." Additionally, the words/phrases of the sentence have morpho-syntactic features such as person, number, etc., which can be extracted.

The resulting structure is also suited to construct error messages with little need for translation. Consider example (8):

(8)a. I has seen an accident.

(8)b.

I has seen an accident.

$$\left[\begin{array}{ll} \text{auxtype}: & \text{have} \\ \text{fin}: & + \\ \text{pred}: & \text{SEE(subj, obj\_dir)} \\ \text{subj}: & \left[\begin{array}{l} \text{case}:\text{nom} \\ \text{num}:\text{sg} \\ \text{pers}:\text{I} \\ \hline \textit{err}:[\text{pers}:\text{III}] \end{array}\right] \\ \text{obj\_dir}: & \left[\begin{array}{ll} \text{case}: & \text{acc} \\ \text{num}: & \text{sg} \\ \text{pred}: & \text{ACCIDENT} \end{array}\right] \\ \hline \textit{err}:[\text{subj}] & \end{array}\right]$$

A possible feedback message to learners might be: “There is an agreement error between the subject and the verb. The subject should be third person, but you chose first person.” As mentioned above, the phrase structure tree can be used for the generation of a correction of the sentence. Moving downward in the tree tells us where the error originated, and the correct forms can be retrieved from the lexicon.

This section illustrated that LFG contains properties that are useful for the pedagogical explanation of syntactic structures. The functional perspective and the strong emphasis on the lexicon coincide with the view taken in descriptive learner grammars. Therefore, feedback to users of an intelligent CALL program can be easily generated to suit the needs of language learners.

6. DISCUSSION AND EVALUATION

The system is able to recognize a range of syntactic errors with little anticipation of the error types. There are, however, two difficulties that are worth mentioning.

The third important linearization error type, the insertion error, has not yet been implemented. However, integrating this capability into the current implementation should not prove too difficult as can be seen in Lee et al. (1995) where this has been accomplished in a somewhat similar way. If a word is encountered for which no hypothesis exists in the chart, a check on the next word should verify that this word can be accounted for. Then, the span of the previous word should be expanded to cover and exclude the unsuitable word.

The second problem is the analysis of words with multiple functions. For example, adjectives in German not only inflect for case, number, and gender, but also for the type of determiner preceding the adjective. Thus, a single inflected adjective may have more than 10 different functions, all of which have to be added to the chart. This in turn increases the time needed for sentence analysis, especially for sentences with multiple occurrences of adjectives. A possible solution might be the integration of disjunctive values for atomic features to reduce the number of lexical entries.

In an initial small-scale evaluation of the system, foreign students at the Humboldt University in Berlin were asked to answer a series of questions about three different topics: reporting an accident shown in a cartoon, introducing oneself, and describing a path on a road map. The students had an intermediate to good knowledge of German with either Slavic (Polish, Russian) or Romance (Spanish, Italian) as their L1 languages. A total of 136 sentences were collected. Fifty-three percent of the sentences were correct and correctly parsed. Sixteen percent were incomplete sentences in that they were either highly elliptical or contained “yes” and “no” answers. Thirteen percent of the sentences could not be parsed, that is, they were either not parsable or they were false positives. Eighteen percent (57% of the erroneous sentences) could be parsed, and an error was recognized in either the c-structure (8 sentences) or the f-structure (16 sentences).

In a follow-up evaluation, the corpus by Heringer (1995) was used to determine the accuracy of the parser. First, a sample of 50 sentences with syntactic errors was randomly chosen from a set of sentences with the 10 most frequent syntactic error types (1,694 sentences with agreement errors, wrong case in prepositional phrases, missing subject, wrong position of finite verb, and wrong subcategorization). The grammar and the lexicon were expanded so that the correct versions of these sentences could be parsed. Then, the erroneous versions were parsed. Of the 50 sentences, 41 (82%) could be parsed, and an error was determined. For only 3 sentences, an incorrect error was reported. Nine sentences (18%) contained errors which the parser could not detect. A subsequent analysis revealed that these errors were mainly case errors with word forms that contained the correct part of speech but highly unlikely morpho-syntactic features. For example, an imperative verb form was placed at the position of a finite indicative verb in a main clause. Also, for 3 sentences, the word order was so garbled that the correction method was not able to analyse the sentence.

In order to handle true free-formed input, the grammar and the lexicon have to be expanded further. Currently, the lexicon contains approximately 14,000 fully inflected forms.

7. CONCLUSION

In this paper, I have discussed how certain adjustments to a unification algorithm and general parsing schemes can account for a range of syntactic errors made by learners of German as a foreign language. As a result, adequate feedback can be generated for robust, but sensitive, parsing of dialog-style exercises with almost free-form input. Unlike selecting practice items from a predefined list as found in most of today's CALL exercises, the system described here contains language learning scenarios that guide learners to actively participate in a communicative situation with the computer as the dialog partner. The main advantages of LFG as the grammatical theory used in the program are the possibility of independent grammar and lexicon development and the clearly defined coding of possible error positions and types outside the grammar. Furthermore, LFG is suitable for the linguistic explanations presented to learners about their input.

REFERENCES

Austin, P. K. (2001). Lexical functional grammar. In N. J. Smelser & P. Baltes (Eds.), *International encyclopedia of social and behavioral sciences* (pp. 8748-8754). Amsterdam: Elsevier.

Bresnan, J. (Ed.). (1982). *The mental representation of grammatical relations*. Cambridge, MA: MIT Press.

Butt, M., Dipper, S., Frank, A., & King, T. H. (1999). Writing large-scale parallel grammars for English, French and German. In M. Butt & T. King (Eds.), *Proceedings of the LFG99 conference*. Palo Alto, CA: Stanford University, CSLI. Available: csli-publications.stanford.edu/LFG

Carpenter, B. (1993). Sceptical and credulous default unification with applications to templates and inheritance. In T. Briscoe, A. Copestake, & V. D. Paiva (Eds.), *Inheritance, defaults and the lexicon* (pp. 13-37). Cambridge: Cambridge University Press.

Clément L., Gerdes, K., & Kahane, S. (2002). An LFG-type grammar for german based on the topological model. In M. Butt & T. King (Eds.), *Proceedings of LFG02 conference*. Palo Alto, CA: Stanford University, CSLI. Available: csli-publications.stanford.edu/LFG

Clément, L., & Kinyon, A. (2001). XLFG—an LFG Parsing Scheme for French. In M. Butt & T. King (Eds.), *Proceedings of LFG01 conference*. Palo Alto, CA: Stanford University, CSLI. Available: csli-publications.stanford.edu/LFG

Dalrymple, M., Kaplan, R. M., Maxwell, J., & Zaenen, A. (Eds.). (1995). *Formal issues in lexical-functional grammar*. Palo Alto: CSLI, Stanford University.

Earley, J. (1970). An efficient context-free parsing algorithm. *Communications of the ACM, 13*, 94-102.

Eisenberg, P. (1999). *Grundrß der deutschen Grammatik - Der Satz* (Vol. 2). Stuttgart: Metzler.

Heift, T. (1998). *Designed intelligence: A language teacher model*. Unpublished doctoral dissertation, Simon Fraser University, Burnaby, British Columbia, Canada.

Heift, T. (2001). Error-specific and individualized feedback in a web-based language tutoring system: Do they read it? *ReCALL, 13* (2), 129-142.

Heift, T., & Nicholson, D. (2001). Web delivery of adaptive and interactive language tutoring. *International Journal of Artificial Intelligence in Education, 12* (4), 310-325.

Heringer, H. J. (1995). *Aus Fehlern lernen* [CD-ROM for Win9x/NT]. Augsburg: Universität Augsburg.

Hubbard, P. L. (1994). Non-transformational theories of grammar: Implications for language teaching. In T. Odlin (Ed.), *Perspectives on pedagogical grammar* (pp. 49-71). Cambridge: Cambridge University Press.

Kaplan, R. M., & Maxwell, J. T. (1993). *LFG grammar writer's workbench.* Palo Alto, CA: Xerox PARC. Available: www2.parc.com/istl/groups/nltt/

Kars, J., & Häussermann, U. (1997). *Grundgrammatik Deutsch*. Frankfurt a.M.: Diesterweg.

Kato, T. (1994). Yet another chart-based technique for pars-ing ill-formed input. In *Proceedings of the 4th conference on applied natural language process-ing (ANLP)* (pp. 107-112). Stuttgart: Association for Computational Linguistics.

Lee, K. Joo, Kweon, C. J., Seo, J., & Kim, G. C. (1995). A robust parser based on syntactic information. In *Proceedings of the 7th Conference of the European Chapter of the Association for Computational Linguistics (EACL)* (pp. 223-228). Dublin: Association for Computational Linguistics.

Mellish, C. S. (1989). Some chart-based techniques for parsing ill-formed input. In *Proceedings of the 27th conference of the association for computational linguistics (ACL)* (pp. 102-109). Vancouver, Canada: Association for Computational Linguistics

Menzel, W., & Schröder, I. (1998). *Constraint-based diagnosis for intelligent language tutoring systems* (Fachbereich Informatik Report Nr. FBI-HH-B-208-98), Universit Hamburg: Universität Hamburg. Available: nats-www.informatik.uni-hamburg.de/~wolfgang/#publi

Schneider, D., & McCoy, K. F. (1998). Recognizing syntactic errors in the writing of second language learners. In *Proceedings of the 17th international donference on computational linguistics (COLING)* (pp. 1198-1204). Montreal, Canada: Association for Computational Linguistics.

Schwarze, C. (1995). *Grammatik der italienischen Sprache*. Tübingen: Niemeyer.

Schwind, C. (1988). Sensitive parsing: Error analysis and explanation in an intelligent language tutoring system. In D. Vargha (Ed.), *Proceedings of the 12th COLING* (pp. 608-613). Budapest: ICCL.

Vogel, C., & Cooper, R. (1995). Robust chart parsing with mildly inconsistent feature structures. In A. Schöter & C. Vogel (Eds.), *Nonclassical feature systems* (Edinburgh Working Papers in Cognitive Science, 10, pp. 197-216). Edinburgh: Edinburgh University.

AUTHOR'S BIODATA

Veit Reuer has a Master's Degree in Computational Linguistics, Linguistics, and Philosophy from the University of Osnabrück in Germany. After receiving his degree, he worked at the Institute for German Language and Linguistics at the Humboldt University in Berlin and is now a researcher at the Institute of Cognitive Science at the University of Osnabrück. Currently he is finishing his Ph.D. thesis on error recognition in ICALL systems. His interests include ICALL, natural language processing, error recognition, German syntax, and e-learning methods.

AUTHOR'S ADDRESS

Veit Reuer
Institute of Cognitive Science
University of Osnabrück
49069 Osnabrück, Germany
Phone: +49 541-969-6219
Fax: +49 541-969-6210
Email: vreuer@uos.de