UNIFICATION-BASED PARSING APPLICATIONS FOR INTELLIGENT FOREIGN LANGUAGE TUTORING SYSTEMS

L. Kirk Hagen, University of Houston—Downtown

ABSTRACT

This paper is a description of a computer based intelligent tutoring system for French as a second language, and of its ancillary grammar editor, which is built on an object oriented, unification-based natural language processor. First I review some of the considerations that motivate using intelligent tutoring systems in second language education. Second, I discuss the details of object oriented programming and unification grammar. Third, I give some examples of how this software interacts with students and of what sort of grammatical phenomena it is capable of handling. I will show how the parser straightforwardly handles some thorny problems of complex grammatical structures (e.g. conjunctions; reflexive binding; phrase embedding; dislocated, missing, or superfluous parts of speech) which have had an inhibitory effect on CALL parsing applications in the past.

KEYWORDS

NLP, parsing, foreign language education, intelligent tutoring, French, unification, OOP, HPSG.

UNIFICATION-BASED PARSING APPLICATIONS FOR INTELLIGENT FOREIGN LANGUAGE TUTORING SYSTEMS

In this paper I describe a computer based intelligent tutoring system which I have developed for students of French as a second language (FSL). This system, called FrenchWriter, is modular in design; its components include 1) a grammar reference
driven by hypertext, 2) a bilingual dictionary with a rule based verb conjugation module, 3) a vocabulary drill with error specific feedback, and 4) a grammar checker (called GrammarMaster) built on a unification-based natural language processor. The first three of these modules are examples of software already familiar to those who have some experience with computer assisted language learning (CALL), and my discussion of them will therefore be limited. I intend instead to devote most of my paper to the grammar checker since this is where the tutorial's novelty is to be found. Following an overview of some of the problems and possibilities of parsing and intelligent tutoring, I will give a sketch of FrenchWriter’s architecture. Then I discuss the fundamentals of unification-based parsing. Finally, I give some specific examples of how the grammar checker interacts with FSL students. I will show that a number of complex and historically problematic aspects of grammatical analysis like conjunctions, reflexive binding, phrase embedding, and dislocated, missing, or superfluous parts of speech can be handled straightforwardly by unification-based parsing algorithms.

THE SCOPE OF NATURAL LANGUAGE PROCESSING (NLP) AND CALL

As a preface let me provide some background on the considerations that motivate the use of parsing technology in the foreign language classroom, as well as on some of the computing and linguistic considerations that ought to restrict its scope. The traditional view in CALL research is that NLP’s most valuable potential contribution to foreign language (FL) tutoring would be its support of open-ended writing activities, e.g. short biographies, narratives, essays, mock letters, and the like. Although assignments like these are generally ‘structured’ in that they come with guidelines about what sort of discourse is expected, it is not possible to enumerate all of the conceivable sentences which would fulfill the assignment’s requirements. Thus simple drills or other tutorials based on string matching algorithms alone are of little use because one obviously cannot enter arbitrarily many sentences into memory for purposes of comparison. Ideally one would prefer an intelligent, rule based, interactive tutorial capable of accommodating an infinite variety of input and thus offering the possibility of versatile, long term and intensive individualized tutoring.
Many of the early foreign language publications on parsing were programmatic in that they amounted to little more than wish lists about what a successful NLP based tutorial could do for FL education if ever one were built. Without a detailed account of a parser’s architecture, or of precisely what sort of grammatical constructions prototypes could deal with, there was little for subsequent research little to build on. Computer programmers, for their part, have often worked independently of linguists and language instructors, and for that have on occasion relied on a rather tenuous grasp of the complexities of languages and the special problems second languages pose for adult learners. For instance, one textbook on AI (Charniak and McDermott 1985) proposed the following phrase structure rule to handle reflexive constructions:

1) 

While this schema works for the simplest case in which a verb is immediately followed by a direct object, as in "John washed himself" which the authors cite as an illustration (191), it is patently inadequate for more complex cases in which the reflexive pronoun is at a greater distance from the subject, as in

2) John is probably going to think about washing himself.

Such difficulties in reflexive agreement have in fact been well known to both linguists and language instructors for some time.

The upshot is that it has never been clear exactly what FL pedagogy should reasonably expect of NLP. Most of the published commentary on the matter has had to do with what not to expect. The following admonition from Marty (1984), one of the pioneers in CALL, is a good example (cf. Leech 1986 as well):
As far as language teachers are concerned, I do not see any possibility that we will ever have a computer program that, for example, would judge free written expression, would perform a phonemic analysis of a student’s oral response, or would understand oral free expression and respond coherently. (160)

In fact, the field of applied linguistics is somewhat notorious for overstating the computer’s prowess, and anyone contemplating onsite development would do well to bear in mind that if AI research has taught us anything over the past few decades, it is that the success of an intelligent tutoring system is usually inversely related to breadth of its knowledge domain.

To have a realistic idea of how ambitious one can afford to be in the development of intelligent CALL software, it will help to divide NLP into three classes, following linguistic tradition. Regardless of what use it is put to, a language parser may be 1) syntactically intelligent, if it can characterize a segment of discourse as well formed or ill formed, and assign a grammatical structure to those segments that are well formed, 2) semantically intelligent, if it can characterize a segment of discourse as true or false, given some set of suppositions about the world, or 3) pragmatically intelligent, if it is able to characterize discourse as appropriate or inappropriate in some specific circumstance, given the intentions of the author, the shared world knowledge of author and audience, their conventions of rhetoric, and so forth (cf. Hagen 1992).

If by “judging free written expression” Marty means software that can pass judgment on the semantic intelligibility, or on the pragmatic appropriateness, of student input, then one would be hard pressed to disagree. To illustrate some of the problems CALL developers would have to look forward to, let us consider a wholly plausible segment of written discourse in English:

Madeleine wondered if the pharmacy was open. I told her she was crazy because it was snowing outside.

Syntactically intelligent software would tell us merely that no English grammar rules have been violated in these sentences; a modest but nonetheless worthwhile objective if one is primarily interested in grammar checking software. A semantically intelligent program, on the other hand, would know what each sentence claims about the world (that Madeleine was curious about some pharmacy’s hours of business, that white ice
crystals were falling from clouds, and so on). This in itself would be a formidable accomplishment, but it would still miss the point. To appreciate fully the implications of these two short statements, a machine would have to understand that the writer was trying to dissuade Madeleine from traveling, that travel by car is dangerous during snowstorms of a certain severity, that the pharmacy in question was not within Madeleine’s walking distance (“walking distance in general” is not a sufficiently ambiguous concept for a computer), that what she planned to buy at the pharmacy was not important enough to warrant a trip in cold weather, or whatever. Although AI is a contentious topic, it is probably safe to say that most who work under its rubric would admit, if pressed, that this is not simply a matter of encoding the relevant information into a computer’s memory. The machine would likely require both the cognitive capacities and the life experiences of a human. Clearly these are hopelessly over optimistic aspirations.

In short, creating intelligent CALL tutorials broadly construed to encompass pragmatic and semantic aspects of language use is too intimidating a venture to contemplate for the time being. There has nonetheless been some recent progress within the more restricted domain of grammatical analysis. For instance Loritz (1992) developed an instructional parsing system for Chinese, Russian, Japanese, and English (GPARS) using an augmented transition network (ATN), which, along with definite clause grammars (DCGs, cf. Labrie and Singh 1991; Norvig 1992, 690-714) has historically been the most popular approach to NLP (cf. For example Charniak and McDermott 1985; Sampson 1986; Seneff 1992; and Germain 1992). Loritz is cautious about his results. He notes that ATN parsers are “versatile and fast, and...can be applied to a wide range of natural language tasks,” but adds that

[i]n education, sophisticated interactive grammar checkers built on instructional parsing systems might eventually be used for direct student instruction, but we do not expect such use will soon be widespread. (17)

Loritz’s appraisal of instructional parsing systems is fairly typical among CALL specialists; the widely held belief is that highly complex syntactic structures such as those mentioned in the introduction simply do not allow for satisfactory parsing and error analysis beyond a very elementary level.

Combining object oriented approaches to linguistic description with unification-based grammar formalisms ... is very attractive. On one hand, we gain the advantages of the object oriented approach: abstraction and generalization through the use of inheritance. On the other hand, we gain a fully declarative framework, with all the advantages of logical formalisms... (160)

Indeed, unification grammar shows signs of unprecedented cooperation between linguists and programmers. Among other things, there is a family of linguistic theories closely aligned with unification grammar—namely Generalized Phrase Structure Grammar (GPSG, cf. Gazdar, et al. 1985; Fisher 1989; Daelemans and Gazdar 1992) and Head Driven Phrase Structure Grammar (HPSG, cf. Pollard and Sag 1987, Pollard and Sag 1994; Carpenter 1991)—which provides a sound theoretical linguistic framework in which NLP development can take place. At the very least, this more promising state of affairs suggests that machine based grammar editing for second languages is still an avenue worth pursuing.

OVERVIEW OF FRENCHWRITER AND GRAMMARMASTER

Before going into detail about the how GrammarMaster’s parser works, a quick overview of the tutorial in its entirety is in order. FrenchWriter is a modular program for developed in HyperCard environment for Macintosh computers with at least 25 MHz microprocessor. Its components are designed to interact with each other as well as with commercial word processing software like WordPerfect and ClarisWorks. Figure 1 shows its overall structure. The base component is an on-line grammar reference with 160 separate screen displays for approximately 100 French grammar rules linked by hypertext, which allows quick and easy linking of related grammar topics.

The component called GrammarMaster is the one that uses a natural language parser. To begin a session, a user calls up an exercise by selecting a button on screen. The session begins with a general topic, for example Parlez-nous de vous (Tell us about
Students can then select a START button on screen, which prompts them to enter some background information—such as their names, the names of a few friends and family members, and the name of the city where they live. Students then begin to write their sentences. For suggestions, they have access to a set of questions, e.g. *Où habitez-vous?* (Where do you live?), *Quels cours avez-vous?* (What classes do you have?), *Comment sont vos amis?* (What are your friends like?), and so on, although they are still free to write in whatever they choose, provided they use the vocabulary supplied in a lexicon on the right side of the screen (cf. Figure 2). The lexicon for the exercise illustrated in Figure 2 contains approximately 500 words.

When a user enters a sentence, the parser attempts to assign a grammatical structure to the input and flags any errors it detects. As an illustration, suppose that a user enters a description of his sister which includes an error in subject/predicate agreement, as illustrated in Figure 2. The number following the section symbol ($) is a hyperlink to the **FrenchWriter** base module. If a user clicks on that number a grammar explanation of subject/predicate agreement will appear on screen. If no errors are detected the user is prompted to save the sentence in a file field and move on to the next sentence, as Figure 3 shows.
The buttons available to the student throughout a session include SPELL, which does spell checking and recommends words in the event one entered by the student is not in the lexicon; CHECK, which provides an account of the various parts of speech of the user’s response. For instance in the example in Figure 3 CHECK returns the following:

Word 1 [soeur] is the SUBJECT of the sentence.
Word 2 [est] is the MAIN VERB of the sentence.
Word 3 [sportive] is the predicate.
FILE, which stores responses entered during a session; and PRINT, which allows all the responses in the file field to be printed at session’s end. There are also hyperlinks connecting the lexicon to an external bilingual dictionary (cf. Figure 1), so that selecting an unfamiliar word in the lexicon will display an English equivalent. The average speed of a parse on a 33 MHz machine is approximately 0.3 seconds per word in a string.

OBJECT ORIENTED PROGRAMMING AND UNIFICATION GRAMMAR

Now let us turn our discussion to the details and crucial procedures of the parser. To clarify some of the terminology used in this section, let me first point out that GrammarMaster is a particular FSL exercise that runs on a particular parser. In principle, some other type of parser could be used to drive the exercise, and since parsing takes place in memory without showing up on screen, an end user might not notice any difference if the two were equally equipped for the task at hand. In any case, GrammarMaster is driven by an object oriented, unification-based parser called HANOI, which was written in HyperTalk. Zahedi (1993) provides a good working definition of object oriented programming:

In modeling knowledge or programming systems, OOP abstracts from reality by choosing the components (objects) of the real world that matter to the problem at hand, and disregards the irrelevant aspects. In this process, OOP comes close to the way we see the world, as interacting objects. The process of abstraction in object oriented programming (OOP) is through the selection of relevant objects, and the identification of their relevant features, attributes, or variables. Objects with similar attributes are classified into a class or objects. (289)

Unification grammar has its underpinnings in the work of Gerald Gazdar and others during the late 1970s and early 1980s (Gazdar 1982; Gazdar and Pullum 1981; Pollard 1984). Their research evolved into a fairly homogeneous group of theories during the mid 1980s with the publication of Generalized Phrase Structure Grammar (Gazdar, et al. 1985) and Information Based Syntax and Semantics (Pollard and Sag 1987). The distinguishing characteristics of this type of framework—at least for purposes of this discussion—are as follows:
The grammar operates on a sets of linguistic objects.

The highest level in the HANOI hierarchy is input, which is simply an ordered set of words which a user enters into a particular field. At the next level of the hierarchy is the phrase — e.g. verb phrase, prepositional phrase, sentence, etc. A phrase is any string of words having a grammatical structure. For instance *mon ta très de vais* Paul is a string but it is not a phrase, and *aller au parc* (to go to the park) is both a string and a phrase. The next level on the hierarchy is the word, and at the most elementary level is the feature, which is simply a label for some linguistic attribute of a word (cf. Keller 1993, 14-59 for a survey of feature logics). HANOI interprets words as sets or matrices of linguistic features, for example

3 \( \{[adorer][\text{catVERB}][\text{compINF}]\} \),

that is, “a word spelled 'adorer' which is a verb in the infinitive and which takes an infinitive verb phrase as a complement.” Conceptually these are unordered sets of features, so that, for example;

\[ \{[\text{catVERB}][\text{compINF}]\} = \{[\text{compINF}][\text{catVERB}]\} \]

As a matter of convenience, however, they are ordinal features in HANOI's lexicon. Thus item(GEN) of a matrix, where GEN is a numeric variable, always designates a word's gender, if any, item(CAT) its category, and so forth. This makes it easier to refer to and modify features during a parse.

These feature specifications are never visible on screen; students merely see the word adorer listed alphabetically in a scrolling field (Figure 2). HANOI is a bottom-up parser, and as such requires a mechanism for passing syntactic information from one level of the hierarchy to the next. Its object oriented design serves just such a purpose in that it allows objects to undergo successive transformations from words to phrases in the course of a parse by preserving or altering features along the way, according to the exigencies of the grammar.

The grammar operates on partial information structures.
For illustrative purposes let us imagine for a moment a primitive grammar which treats the phrase *la table* (the table) as a combination of two features, [artDEF] and [typeCOMMON]. A parser based on such a simple description would quickly fail the mission we have sketched so far because in French articles must also carry information about NUMBER and GENDER. Therefore the lexicon must at a minimum specify the following:

4) a. ([la][artDEF][genF][numS])
   b. ([table][catNOUN][typeCOMMON][genF][numS])

In other circumstances, however, the parser should be able to “disregard irrelevant aspects” of input, to use Zahedi’s words, and to make decisions based on relevant but partial information of objects. For instance, it will recognize *manger sa soupe* (…to eat her soup) as a grammatically correct verb phrase without reference to the number and gender of the direct object because verbs in the infinitive do not show any inflectional changes that reflect those properties. A well formed object is one that is a permissible instance or extension of a general rule.

The grammar makes heavy use of the operation on sets of features called unification, which is defined in Pollard and Sag (1987) as follows:

...if A, B, and C are feature structures, we call C the unification of A and B...provided C is the least informative feature structure which is at least as informative as A and at least as informative as B. (35)

To illustrate, the definite article *les* in French is overtly marked as plural but is not marked as masculine or feminine. Conversely, the word *cours* (course/courses) is always masculine, but it is not overtly marked as singular or plural. Instead of burdening lexical searches with, say, one masculine and one feminine form of *les*, or one singular and one plural form of *cours*, HANOI’s lexicon includes one entry for each but omits unmarked features:

5) a. ([les][artDEF][numP])
   b. ([cours][catNOUN][typeCOMMON][genM])
By Pollard and Sag’s definition, (5a) and (5b) unify to form,

6) \{[les][cours][catNOUN][artDEF][numP][typeCOMMON][genM]\}

which is nothing more than an abbreviation for "a noun phrase with a common, masculine, plural noun and a definite article," which is precisely what les cours is.

HPSG, which was created to describe first languages, places an important restriction on unification, namely that two categories A and B fail to unify if they contain mutually inconsistent information. Intuitively this means, for example, that in French as a first language there is no such thing as a noun phrase that is both masculine and feminine. However, there are indeed such phrases in second languages; i.e. “errors.” A parser designed to accommodate second languages thus requires a slight qualification on the notion of unification. A very common type of error among FSL students — article/noun agreement, as in the phrase *le table — will illustrate. Strictly speaking, this is not a phrase at all, if by phrase we mean “a well formed grammatical unit,” though needless to say, a CALL program that returned a message like “not a well formed grammatical unit” as error feedback would not be of much help to an FL student. When HANOI detects contradictory features like [genM] and [genF], it preserves the features on the major part of speech (in the case of noun phrases, the noun) and inserts the contradictory feature on a minor part of speech like an article into an error stack along with a corrective message. Once the parse is finished, HANOI accesses its error stack and returns its feedback. This means that the outcome of an actual parse of *le table would be

7) \{[catNOUN][typeCOMMON][genF][numS][artDEF]\}

and the user would be informed via feedback that the definite article does not agree in gender with the noun it modifies.

- HANOI is a cyclical parser in that it analyzes strings repeatedly until it fails or succeeds in determining the overall grammatical structure of the input.
As it analyzes a string like

8)  *Paul habite à Londres.* “Paul lives in London”

it first determines that *à* and *Londres* meet the structural description of a prepositional phrase and then combines the two into a constituent:

\[
\text{put } \\
\text{delete }
\]

During a subsequent cycle it determines that *habite à Londres* is a verb phrase, and on its final pass it will analyze subject/verb agreement and assign the structure “sentence” to the entire string. The sequence of phrases below illustrate this process.

9)  

L1: [Paul][habite][à][Londres]  

L2: [Paul][habite][à Londres]  

L3: [Paul][habite à Londres]  

L4: [Paul habite à Londres]

The number of cycles required for a parse is determined by the number of words in the string. For instance, let us say that L1 in (9) is the initial “line” of a parse, i.e. each item in brackets represents a word entered by the user. Suppose that in the course of a parse line(L+1) were identical to line(L). Since the same procedures which operate on line(L) also operate on line(L+1), it follows that line(L+2) and indeed all subsequent lines would be identical. In other words, any given line must contain at least one fewer constituent than its predecessor if a cycle is to produce any novel output. This in turn means that if the number of cycles is equal to the number of words, the parser will be able to assign a structure to any string so long as it has adequate rules as part of its code. By the same token, if at any point in the course of a parse some line(L) is identical to line(L-1), then the string either cannot be or already is completely parsed, and the procedure is aborted.
ERROR ANALYSIS: CONJUNCTIONS

Now that we have laid out the fundamental operations of HANOI's parser we are in a better position to show some concrete examples of how it interacts with students of French, and of what sort of grammatical contingencies it must be prepared for. Let us begin with the analysis of subject/verb agreement in strings like (8) and (10),


and focus on the final cycle of the parse for the time being. Suppose the relevant algorithms were more or less as follows. Having determined that habite is a verb in the third person singular, the parser undertakes a search for a noun to the left of the verb. Finding “Paul,” which is also third person singular, it determines that subject and verb agree, and its work is therefore finished. This simple strategy would succeed only so long as we excluded the conjunction et (and) from the lexicon. Otherwise the parser would not detect errors in cases where a verb in the singular has a compound noun phrase as its subject, as in (10), which includes a string (in italics) identical to the one that is judged correct in the case of (8).

Excluding conjunctions from the lexicon is unacceptable on pedagogical grounds, since even elementary level FSL students should be expected to make frequent use of them. Simplifying for expository purposes, HANOI's solution to (10) is to generate during initial cycles a single part of speech (Marc et Paul) which is specified as plural. This is what the computer will use in subsequent cycles to evaluate subject/verb agreement. The output is the feedback shown in Figure 4, where the section symbol and number (§2) once again are hyperlinked to a short lesson in the grammar module.

HANOI takes a similar tack in its treatment of grammatical case assignment. For example the string moi allons au cinéma is perfectly acceptable in (11a) but quite ungrammatical in (11b):

11)  

   a. Lundi mon ami et moi allons au cinéma. “Monday my friend and I are going to the movies.”

   b.*Lundi moi allons au cinéma. “Monday me go to the movies.”
In the lexicon, the item which denotes the CASE specifications of proper nouns is empty since proper nouns are not inherently subjects or direct objects. Stressed or “tonic” pronouns, on the other hand, are specified as [caseTON], which blocks them from being subjects or direct objects by themselves (cf. 11b). When tonics and proper nouns are conjoined, HANOI considers the latter to be “more marked,” so that the CASE feature on the output is empty. By the same token, if the PERSON specifications are inconsistent in conjoined phrases, HANOI considers the lesser of the two values to be more marked. Thus in (11a) *mon ami is [per3], while moi is [per1]. The lesser overrides, the output is marked [per1], and (11a) therefore returns in the same “correct answer” feedback shown in Figure 3, while (11b) returns the feedback shown in Figure 5.

Things become even more complex when HANOI has to decide exactly which parts of speech it is supposed to conjoin. The examples in (12) illustrate an especially problematic set of conditions for natural language processors:


   b. Le professeur connaît Marie et je connais sa soeur. “The professor knows Marie and I know her sister.
Here the italicized string in (12a) illustrates a case in which the subject pronoun *je* is incorrectly used as a direct object in a conjoined phrase. However, (12a) is perfectly acceptable when it is part of a larger sentence like (12b). Under such conditions HANOI makes use of an algorithm which instructs it to postpone its conjoining operations to later cycles, after it has assigned a structure to the rest of the sentence:

\[
\text{if } \min((N-1),(\text{the number of items in line(L) of this Parse}) - 1) \geq L \text{ then conjoin}
\]

Here “\(\text{min}\)” is a function which return the minimum number from the list of specified values. Essentially this algorithm divides the string into two substrings at the point of the conjunction, and parses each independently. What holds for the relation between in words and cycles in larger strings also holds for these substrings: a four cycle parse will yield two sentences (*le professeur connaît Marie* and *je connais sa soeur*) in the case of (12b), while a single cycle parse will yield two contiguous noun phrases (*Marie* and *je*) in the case of (12a). Since HANOI will only conjoin like elements—noun phrases with noun phrases, sentence with sentences, and so on—the error message shown in Figure 6 is generated only in case of (12a). Example (12b) generates the “correct response” feedback.

![Figure 6](image)

**Figure 6.**

**REFLEXIVE BINDING**

The behavior of reflexive pronouns is one of the most intensely studied phenomena in both formal and applied linguistics (cf. Sells, et al. 1987; and especially Dalrymple 1993). It is clear that the distribution of reflexives in the grammars of human languages is subject to universal constraints (cf. Pollard and Sag 1983; Pollard and Sag 1987, 23-25). One’s inclination is probably that any principle which holds universally is not a likely source of error for second language learners, and a language tutorial need not include contingency plans for cases in which the principle is violated. If, on the other hand, a principle of universal grammar does not have force in a second language, then the tutorial must be able to diagnose violations accordingly and offer appropriate corrective
feedback. As it turns out, there is evidence from second language acquisition research (cf. Eckman 1994; Hagen 1994) that reflexive binding constraints do not hold for adults learning a new language. The sentences that follow illustrate what sort of input is of concern:

13) a. je me couche. “I put myself to bed.”

   b. Demain il va se coucher à neuf heures. “Tomorrow he is probably going to go to bed at nine o’clock.”

   c. Les étudiants que je connais se couchent à huit heures. “The students that I know put themselves to bed at eight o’clock”

   d. Avant de se coucher il va manger. “Before putting himself to bed he is going to eat.”

As an initial and approximate solution to subject/reflexive pronoun agreement, let us imagine that when the parser encounters the reflexive pronoun *se* it is supposed to check features on the noun to the immediate left for agreement. This would allow the parser to generate the appropriate feedback in an example like the first sentence in Figure 7, where the subject *je* is first person and the reflexive *se* is third person. But as we noted above, when the reflexive is in an embedded verb phrase, as in (13b), the word to the immediate left of the reflexive is not a noun at all, and the parser would therefore send error feedback even though the input is grammatically correct.

As a revised version, suppose we were to program the computer to “keep looking left until you find a subject pronoun,” so that if *se* is in item(N) of the string, the parser would search backwards to item(N-1), item(N-2), and so on, until it finds *il* in (13b). This strategy would solve one embedding problem, but it too would fail in still more complex cases like (13c). Here, because of the intervention of the embedded relative clause *que je connais*, instructions to “keep looking backwards” would mean that the computer would encounter the pronoun *je*, interpret it as the subject of the base clause, and return an error message. To complicate things further, when a reflexive pronoun is part of an adverbial verb phrase that occurs at the beginning of a sentence, then the subject it is supposed to agree with will be to the right, not the left, as in (13d).
Unless we were to sharpen our algorithms, the parser would wrongly inform the user, in each case in (13b-d), that the reflexive pronoun does not have an acceptable subject. HANOI’s approach is to exploit feature passing or “inheritance” principles common to both unification grammars and object oriented environments. Its algorithm is based to a large extent on FOOT features described in Gazdar, et al. (1985). Since the various operations in HANOI are ordered hierarchically, the parser will attempt to identify a direct object complement for a verb before reflexive pronoun/subject agreement is evaluated. In (13b), the pronoun *se* and the infinitive *coucher* (to put oneself to bed) will generate a verb phrase with a REFL notation as output. On the next cycle the main verb *va* initiates a search for an infinitive verb phrase complement and, finding *se coucher*, in turn adopts the REFL notation as its own:

```plaintext
if item(REFL) of nextConstituent ≠ ø then
    put item(REFL) of nextConstituent into item(REFL) of thisConstituent
end if
```

Subsequently the head constituent is raised to the next line of the parse:

```plaintext
put item(N) of line(L) of thisParse & the itemDelimiter after ¬ line(L+1) of thisParse
```

The process continues until the verb phrase with the REFL notation and the subject are contiguous, at which time the agreement handler is executed. The result is that HANOI finds no errors in any example in (13) but correctly identifies all of the errors shown in Figure 7.
DISPLACED, MISSING, AND SUPERFLUOUS CONSTITUENTS

The mechanism that permits HANOI to bind subject and reflexive pronoun also underlies its treatment of displaced parts of speech in unbounded dependency clauses. Here are some relevant facts about French grammar that HANOI must account for: i) past participles agree in number and gender with a preceding direct object; ii) when clitic pronouns are direct objects, they must occur in the same clause as the past participle (i.e. they are bounded); and iii) when a relative pronoun is the preceding direct object, any number of clauses can intervene between a verb and the complement (thus they are unbounded). To illustrate agreement between past participle and direct object, in example (14) below the past participle écrite (written) must be feminine because the displaced object la lettre is feminine:

14) Je n’ai pas la lettre que Marie a écrite. “I don’t have the letter that Marie has written.”

The grammar of French also allows for examples which translate as “I don't have the letter she is supposed to have written,” or “I don't have the letter that Paul said she is supposed to have written,” and so on ad infinitum. Here as in the case of reflexive agreement, the principle must have force across arbitrarily long strings of words intervening between pronoun and participle.

When a transitive verb fails to find an acceptable direct object complement it inserts a “trace word” into the string it is parsing. This trace word passes up the hierarchy until it finds a direct object relative pronoun. If NUMBER and GENDER match, then the input is correct. If not, HANOI returns error feedback, as shown in Figure 8. This trace mechanism has the added advantage of facilitating error feedback in instances of missing or superfluous parts of speech. Fisher (1989) includes an example of the sort of problem we are concerned with here. He describes a GPSG parsing system of English built for a Sun platform which uses a feature passing rule similar to what is used in HANOI, and notes that

the rule ... correctly generates 'the telephone Carol tested.' It is not possible, however, by this mechanism, to prevent '*the telephone Carol tested the telephone,' in which 'Carol tested the telephone' is correctly parsed as a S... (147).
Similar examples with superfluous constituents are frequent in second languages, particularly in cases of L2 words whose L1 homologues have a different range of complements. For instance, “leave” in English is optionally transitive, but one of its French equivalents (partir) is intransitive. Another French homologue of “leave” (quitter) is obligatorily transitive. Thus examples like


       b. *je quitte à huit heures. “I’m leaving at eight o’clock.”

are common among students of French. Labrie and Singh (1991) discuss some of the problems examples like these can cause for parsing programs.

One of the attractive features of HPSG is its eschewal of “the orthodoxies dictated by [one's] teachers' and senior colleagues' doctrinal affiliations” (Pollard and Sag 1987, 11). As the authors say,

   it will become evident to readers familiar with a range of contemporary syntactic and semantic theories of language that many of the constructs and hypotheses of HPSG—perhaps most of them—are borrowed or adapted from elsewhere. (11)

and consequently,

   the full range of current theories can be composed, decomposed, compared, recombined, and generally tinkered with, in a manner constrained only by the individual researcher’s aesthetic sense, philosophical predispositions, and responsibility to get the facts right. (11)
The advantages of such openness should not be underestimated: it allows parsers to pick and choose principles and techniques from various theories to the extent that they allow the parser to perform as desired. In its treatment of superfluous constituents, HANOI borrows the notion of thematic assignment or “θ-assignment” from Government Binding theory (cf. Chomsky 1986, 93-101) by requiring that at the end of a parse all constituents in a string be assigned a thematic or grammatical role, i.e. the item in its lexical matrix denoting THEME must not be empty. Any constituent that does not have a role is flagged, and an error message is sent to the user. Thus in an example similar to Fisher’s cited above,

\[ 16) \quad *Je n’ai pas la lettre que Marie a écrite la lettre. \quad “I don’t have the letter that Marie wrote the letter.” \]

the relative pronoun *que* is supposed to be assigned a direct object role by the missing trace word. But the parser will not insert trace word in the case of (16) because the verb *écrite* (written) will find the direct object it is looking for. The result is that when the parse is finished only *que* is without a role, and the student thus receives a query as feedback:

“What does ‘que’ refer to? It is direct object of which verb?”

which, as far as the end user is concerned, is an intuitively pleasing account of what is wrong with the example.

As for missing constituents, HANOI similarly borrows the notion of saturation from HPSG, defined in Pollard and Sag (1987) as follows:

The subcategorization (or valence, or combinatory potential) of an HPSG sign is described by a special feature called the subcategorization feature (usually abbreviated SUBCAT), whose value is simply a list of the kinds of signs with which the sign in question must combine in order to become saturated (e.g. in the case of a verb, in order to make a complete sentence). (11-12)
This is an unusually obtuse way of saying that all of the words in a string must be able to account for whatever other parts of speech need to go with them. In French the verb donnes (second person singular of donner, to give) requires a direct object, an indirect object, and a subject. When HANOI finds a direct object complement of donnes during a parse, the verb “absorbs” the object. That is, the object becomes part of a verb phrase, much as the article becomes part of a noun phrase in la table. Once subject, direct object and indirect object have all been absorbed, donnes is said to be “saturated.” If, at the end of a parse, a complement or a subject is not accounted for, the verb remains unsaturated, and unsaturated words signal error messages, as shown in Figure 9.

SUMMARY

Let us sum up by trying to establish a few general criteria for evaluating parsers like the one described in this paper. Following Allen (1995), we should stipulate that these must include minimally 1) generality, i.e. “the range of sentences the grammar analyzes correctly,” 2) selectivity, “the range of non-sentences it identifies as problematic,” and 3) understandability, “the simplicity of the grammar itself.” (44)

A parser’s performance on the first two of these involve reasonably objective standards: the parser either flags grammatical errors or it does not. Consequently it is comparatively easy to pass judgment on them. Allen’s third criterion is a bit more subjective since there is still widespread disagreement over what constitutes a “simple” grammar. There is no question that the machinery of formal linguistic theory will continue to play a central role in NLP. Yet a common—and not wholly unjustified—complaint in second language pedagogy is that modern syntactic theory has reached a level of abstraction that has rendered it all but opaque to language instructors. But this is by and large a function of the theoretician’s objectives. In the case of the most popular of modern theories-Government binding—the goal is “to depict exactly what one knows when one knows a language: that is, what has been learned, as supplemented by innate principles” (Chomsky 1986, 24). It is, in Chomsky’s words, the study of the ‘internalized’ or acquired language, and thus it properly falls under the rubric of cognitive science and learning theory. Much of the unwarranted resentment against Chomsky style linguistics frequently expressed in second language pedagogy may be due to the false assumption that Chomsky and his disciples somehow intend to exclude all other forms
of linguistic inquiry. There is, however, nothing in Chomsky's writing to support such a view, as Newmeyer (1983, 137-148) pointed out a long time ago. In any event, the grammar one uses to drive a parser is properly understood as an example of what Chomsky (1986) calls an “externalized language (E-Language),”

in the sense that the construct is understood independently of the properties of the mind/brain...the linguist is free to select the grammar one way or another as long as it correctly identifies the E-language. (20)

Expressed in these terms, “correctly identifying” a suitable grammar for a parser does not in itself entail any questions of cognition. Only practical matters ought to have a bearing on the development of a linguistic framework for FL parsing. For instance, we might add to Allen's list generalizability, or the extent to which the grammar is applicable across second languages, so that the code that drives a parser of one second language should adaptable to a parser of other second languages with minimal modifications.

These considerations, rather than any epistemological questions raised by language acquisition, were paramount in the development of the software I described above. For instance, by construing the entire Grammar in terms of linguistic objects, HANOI parsing can in principle be implemented in any other environment that accommodates object oriented programming. And OOP carries with it the added practical advantage of code reusability, which is less taxing on memory and facilitates debugging.

There remains a substantial amount of work to be done on GrammarMaster and its parser. As of now, for instance, there is a HANOI parser of French only, since the tutorial was created specifically for the users I mentioned earlier in this paper. Its lexicon, moreover, needs to be augmented to handle a greater variety of writing tasks. Nonetheless HANOI can now accommodate many of the thorny structural problems of second languages that have had an inhibitory effect on CALL parsing applications, and its design is general enough- and its expressive capacity powerful enough-that software for other second languages could be developed with reasonable speed and facility.
NOTES


2. The "major part of speech" is what is commonly referred to as the "head" in most theories of syntax. Hence the origin of “Head Driven” Phrase Structure Grammar.

REFERENCES


AUTHOR'S BIODATA

Professor L. Kirk Hagen received his Ph.D. in Second Language Acquisition and Teacher Education at the University of Illinois at Urbana-Champaign. His research interests are in intelligent tutoring systems for foreign languages and computer driven experimental design. His articles have appeared in professional journals and books, including the *ADFL Bulletin*, *La Revue canadienne des langues vivantes*, and *Research Methodology in Second Language Acquisition*. Professor Hagen is the developer of the **French Verb Tutor** (Santa Barbara: Intellimation) and has been active in Joint Europe-USA EXPERSYS Conferences on Expert Systems Applications and Artificial Intelligence, both as a member of the Program Committee and as Co-Chair. Currently Dr. Hagen is Assistant Professor of Humanities at the University of Houston-Downtown.

AUTHOR'S ADDRESS

Arts and Humanities Department  
University of Houston-Downtown  
One Main Street  
Houston, TX 77002  
Phone:(713) 221-8116  
Fax:(713) 221-8928  
E-mail: chaco@aol.com