Knowledge integration in a robust and efficient morpho-syntactic analyzer for French†

Louisette Emirkanian
Dép. de linguistique

Lorne H. Bouchard
Dép. de mathématiques et d'informatique

Université du Québec à Montréal
C.P. 8888, Succursale "A"
Montréal, QC
Canada H3C 3P8
R15320 @ UQAM . BITNET

ABSTRACT

We present a morpho-syntactic analyzer for French which is capable of automatically detecting and of correcting (automatically or with user help) spelling mistakes, agreement errors and certain frequently encountered syntactic errors. Emphasizing the specific language knowledge that is used, we describe the major sub-tasks of this analyzer: word categorization by dictionary look-up and spelling correction, construction of a parse tree or of a forest of parse trees, correction of syntactic and morphological errors by processing the parse tree. The spelling corrector module is designed to help correct the spelling mistakes of a French novice, as opposed to those of an experienced typist. The syntax analysis module is driven by an empirical grammar for French and is based on the work of Tomita. The presentation is based on the design and implementation of a prototype of the system which is written in Lisp for the Macintosh computer.

1. INTRODUCTION

Our goal is to construct a morpho-syntactic analyzer for French which is capable of automatically detecting and of correcting (automatically or with help from the user) spelling mistakes, agreement errors and the most important syntax errors. This system could be used to analyze word processor output, for example.

Since our main goal is to implement a robust and efficient analyzer for French, we have designed a system which can detect errors as opposed to one which can only process well-formed input.

A number of systems for English text analysis have been developed. The Writer's Workbench /Frase 1983/ is a collection of tools developed at AT&T's Bell Laboratories: the two most important ones address proof reading and style analysis. The EPISTLE project /Miller, Heidorn & Jensen 1981/ is a vast project undertaken at IBM's Thomas J. Watson research laboratory, the long term goal of which is to develop a system which not only supports writing, but also text understanding. WANDAH /Friedman 1984/, a system that was developed at UCLA, comprises three sub-systems: a word processor designed to support interactive composition, tools to assist composition and tools to help in the editing and the revising phases.

These systems are difficult to adapt to French since they are based on knowledge which is specific to English. Furthermore, in these systems the knowledge is rarely represented explicitly: indeed, the knowledge has most often been "compiled" for reasons of efficiency. Thus, these systems cannot easily reason about the knowledge they have.

The novel feature of our system is that it is based on an integration at different levels of the knowledge of French. This knowledge is represented explicitly in the system and the system keeps track of the decisions it has made, which will allow it not only to justify its decisions but also to reason about its reasoning.

The main problem is in the integration of knowledge of the language, knowledge which is at different levels: knowledge of orthography /Catach 1980/, of traditional grammar /Le nouveau Bescherelle 1980/ /Grevisse 1969/, of syntax /Grevisse 1969/ /Gross 1975/ /Boons, Guillet & Leclère 1976/ and also of the most frequently encountered errors /Catach, Duprez & Legris 1980/ /Class & Horguelin 1979/ /Lafontaine, Dubuisson & Emirkanian 1982/. In order to be able to use such knowledge, it must on the one hand be made operational and it must on the other hand be orchestrated.

† Research funded by the Social Sciences Research Council of Canada (SSRCC grant no. 410-85-1360).

In our system, these sources of knowledge are used as follows. Each sentence of the text is split up into words. Each word is categorized by dictionary look-up; knowledge of French orthography is represented as a collection of correction rules. An efficient parser, driven by a context-free grammar, builds a parse tree or a forest of parse trees in the case of ambiguity. This parser is deterministic in the sense that it blocks as soon as an error is detected. The parser can recall the spelling corrector, if need be. Then, knowledge of the sub-categorization of French verbs allows the system to eliminate automatically certain ambiguities and to detect and correct many errors. Finally, the user is consulted whenever the system cannot intervene.

Before presenting the system in depth, we must emphasize that the system we have designed is intended to assist at the knowledge level and not at the competence level. It is not designed as a tool to improve written communication skills.

The main sub-tasks of the system are as follows:
- word categorization by dictionary look-up and spelling correction,
- construction of a parse tree or of a forest of parse trees in cases of ambiguity,
- correction of syntax errors, detection and correction of morphological errors by processing the parse tree.

We shall now examine these three phases.

2. WORD CATEGORIZATION AND SPELLING CORRECTION

2.1 Classification of spelling mistakes

We have adopted Catach's classification /Catach, Duprez & Legris 1980/ from where we also borrow the examples. She distinguishes phonetic errors (**puplier* instead of *publier*), from phonogrammic errors (the user knows the sound without knowing the transcription) some of which can modify the phonic value of a word (**gérir* instead of *guérir* , **oisis* instead of *oasis*) whilst others do not change the phonic value (**pharmatie* instead of *pharmacie*). In addition to these two types of errors, she identifies morphogrammic errors (caused by faulty knowledge of non-phonetic orthography) in grammatical elements (number agreement, for example) or in lexical elements (**enterremant* instead of *enterrement*, **abrit* instead of *abri*, for example), confusion of lexical homophones (*vain / vin*) or grammatical homophones (*on / ont*), problems with ideograms (punctuation, for example) and finally problems with non-functional letters which are derived, for example, from the Greek origin of a word (**téatre* instead of *théatre*).

We have excluded from our area of investigation all phonetic errors, that is errors which can be caused by faulty pronunciation.

On the other hand, our system can handle all the phonogrammic errors. Morphogrammic errors in grammatical elements are detected during the later morphological analysis phase. Errors in lexical morphemes are corrected during this phase, as well as errors which are due to the existence of non-functional letters. As for problems with homophones, grammatical homophones are detected during the parsing or the syntax analysis phases, but

only some lexical homophones are detected during these phases: we can correct *vain / vin* but not *chant / champ*, since these elements, in addition to being homophones, belong to the same lexical category. The semantic knowledge available in our system is not sufficient to resolve this ambiguity.

Regarding spelling mistakes, phonogrammic errors (i.e., those due to the transcription of sounds) are the most frequent in French, mainly because of the problems caused by the phonic/graphic correspondence. For example, the sound *[o]* can be written in many ways: *au, aud* (at the end of a word), *eau,* etc. This is not the case in English /Peterson 1980/, where the main spelling mistakes seem to be due to random insertions or suppressions of letters, substitution of a letter for another or transposition of two letters. We call these errors "typographical" errors: we will not discuss them further in this paper.

2.2 The dictionary

Our system is based on two dictionaries, a dictionary of stems and a dictionary of endings. Associated with a stem, in the stem dictionary, is stored a pointer to a list of one or more endings which are stored in the endings dictionary. In this way, our system can handle all inflected forms efficiently, as well as the numerous exceptions. Based on a suggestion by Knuth /Knuth 1973/, a *trie* is used to index the stem dictionary. Diacritical signs are removed from the letters when the *trie* is constructed and also when a word is looked up in the *trie*. Indeed, the letters modified by the diacritical signs are only stored in the leaves of the *trie*. This allows our system to handle accent errors, a common spelling mistake, very efficiently.

Instead of storing "chameau", "chameaux", "chamelle" and "chamelles" in the dictionary, we only store the common form "cham-" in the stem dictionary together with its lexical category. We also store there, as pointers to the endings dictionary, the corresponding rules for constructing the number and gender endings and any additional syntactic or semantic information, as required.

2.3 The look-up algorithm

The word to be looked up is scanned from left to right: each letter, stripped of its diacritical sign if need be, controls the walking of the stem *trie* until a leaf is reached. Associated with the leaf, we find the lexical category and the ending rules for the stem. Remaining letters of the word are looked up in the list of endings associated with the stem: the entry corresponding to an ending records, for example, the number and gender of nouns and adjectives or the person, time and mood of the verbs which have this ending (the endings lists contain all possible endings of the verbs /Le nouveau Bescherelle 1980/). The most important ending errors are also recorded in the endings lists. Using this information, the system can detect and correct at this level ending errors: for example, **chevals* instead of *chevaux*, **cloux* instead of *clous*.

A block during *trie* traversal signals the detection of a spelling mistake. The context of the letter responsible for the block is used to index a large set of rewrite rules, called correction rules, which are derived mainly from the phonic/graphic transcription rules of French /Catach, Duprez & Legris 1980/. These rules characterize the knowledge of French orthography which is used to correct the spelling error.

2.4 The correction algorithm

Although the set of correction rules is mostly based on the phonic/graphic transcription rules of French, certain rules are not based on such a strict correspondence at all since the programs can also, for example, correct **enui* to *ennui* and **gérir* to *guérir*.

When a leaf is finally reached, the rule or rules which were applied to unblock the walk in the *trie* are used to correct the misspelt word.

In addition to substitution rules, we have a set of rules which are used only on the ending of a word. These rules are applied before the substitution rules. For example, for the word **blan* the system proposes *blanc*, and for the word **tros* it proposes *trots*, *trop* and *trot*, as can be seen in Fig. 1.

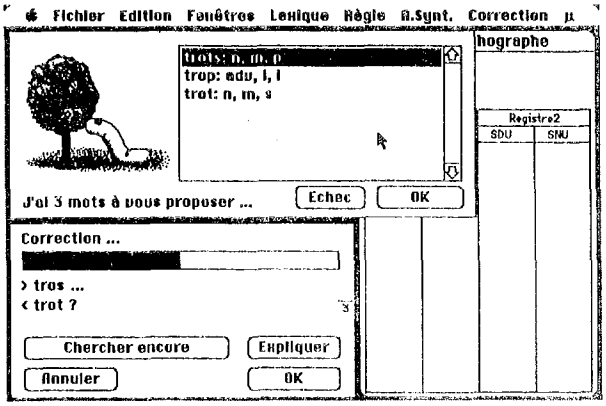

Fig. 1

If the user is not satisfied with a correction, the system can, upon request, propose another in some cases. For example, in response to the word **vi* the system proposes *vie* (the noun) and if the user requests another correction, it then proposes the two verbs *voir* and *vivre* as can be seen in Fig. 2, since the stem, or one of the stems, of these verbs matches the word **vi*. The user may conjugate these verbs using our *Conjugueur* tool, as can be seen in Fig. 3.

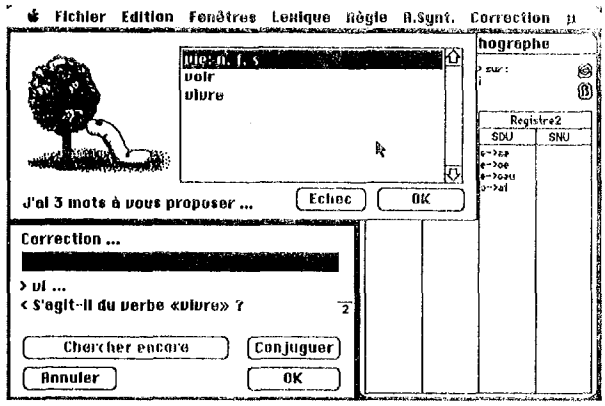

Fig. 2

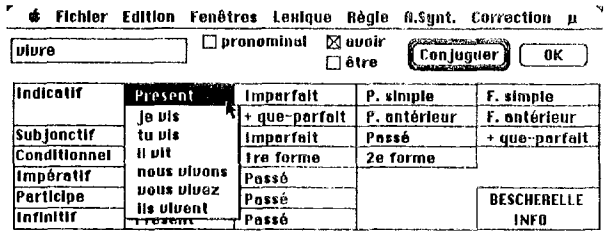

Fig. 3

In many cases however, when the error is located before the block point, the correction algorithm must move the block point back and thus performs a systematic search of the dictionary, backtracking upon failure. Indeed, for the word *entente* spelt **antente*, the first block point is just after the second *n* since *antenais* and *antenne* are in our dictionary /Robert 1967/.

The size of the dictionary and of the set of correction rules is large. The system uses simple metrics as heuristics /Romanycia & Pelletier 1985/ in order to filter the set of correction rules and reduce the search space. The selected rules are analyzed and those that do not increase *trie* penetration depth or those that do not allow the system to move forward in a word (simple metrics of progress towards the goal of accounting for all the letters in a word) are rejected. Note that the expectations of the dictionary, represented as a *trie*, also effectively constrain the search space.

2.5 Word categorization

At this point, a word can have been assigned a single lexical category, as for example *cahier* : N [F-, etc.]. The word can also be assigned a wrong category, as for example in *il *pin* : N [F-,etc.] which was written instead of *il peint*. Finally, a word can be assigned many categories (case of lexical ambiguity), as for example *il vente* : N [F+,etc.] / V [present 3rd person of indicative / subjunctive].

3. CONSTRUCTION OF A PARSE TREE OR OF A FOREST

We have compiled an empirical grammar of written French which is described by a context-free grammar. Our parser is based on the work of Tomita /Tomita 1986/ /Tomita 1987/. In a Tomita parser, a general purpose parsing procedure is driven by a parsing table which is generated mechanically from the context-free grammar of the language to be parsed. Tomita's main contribution has been to propose the use of a graph-structured stack which allows the parser to handle multiple structural ambiguities efficiently. We use YACC /Johnson 1983/, a LALR(1) parsing table generator available in UNIX to automatically generate the parsing table which drives the general parsing procedure. When generating the parsing tables, YACC detects and signals cases of structural ambiguity.

Many cases can arise in parsing French.
Consider first the case when a word has been assigned multiple categories. Some of the ambiguities can be resolved by considering the expectations of the grammar. Consider the word *court* which can be an adjective, an adverb, a noun or a verb. If *court* is found in the context *il* : [ProCl] *court* : Adj / Adv / N / V [3rd person singular, etc.], the grammar accepts only the verb at this point. Similarly the word *une* which can be a determinant, a noun or a pronoun can automatically be reduced to noun in the context *il a lu la une du journal.*

Consider now the case when the parser cannot derive a parse tree: based on the hypothesis that there may be a spelling error which caused an erroneous category to be assigned, the parser calls the spelling corrector to revise the spelling of a word and hence the category assigned to it. In the case of the previous example *il *pin*, of the spelling alternatives for *pin* , only *peint* , the verb, is retained since *pain* is no more possible in this context than *pin* . Indeed, in our grammar of the sentence only a verb or another clitic pronoun may appear after a clitic pronoun. Similarly, in the sentence *ils *on apporté le livre*, **on* will be corrected to *ont* . The parser efficiently constructs a parse tree or a forest of parse trees which account for the sentence. In a Tomita parser, the forest of parse trees is represented by a data structure analogous to a *chart* /Winograd 1983/, which allows for "local ambiguity packing".

4. ANALYSIS OF THE PARSE TREE OR FOREST

A forest of parse trees can be produced in classical cases of structural ambiguity such as in *Pierre expédie des porcelaines de Chine*. The two parse trees generated for this sentence can be seen in Fig. 4 and 5. The bracketed Lisp representation of these trees can be found in Fig. 6 and 7.

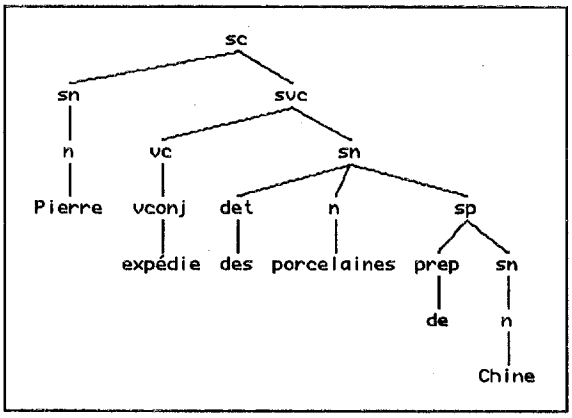

Fig. 4

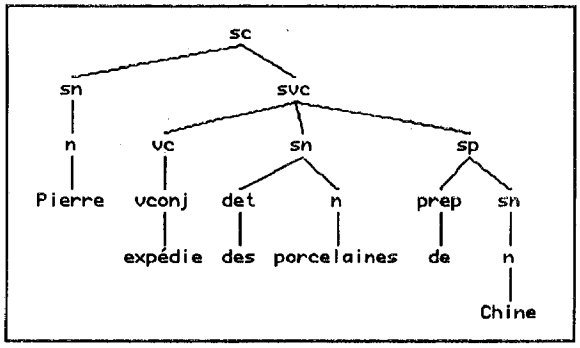

Fig. 5

```
(sc   (sn   (n     "Pierre"))
      (svc  (vc    (vconj  "expédie"))
            (sn    (det    "des")
                   (n      "porcelaines")
                   (sp     (prep   "de")
                           (sn     (n    "Chine"))))))
```

Fig. 6

```
(sc   (sn   (n     "Pierre"))
      (svc  (vc    (vconj  "expédie"))
            (sn    (det    "des")
                   (n      "porcelaines"))
            (sp    (prep   "de")
                   (sn     (n "Chine")))))
```

Fig. 7

A forest of parse trees can also be caused by cases of lexical ambiguity such as *il veut le boucher*. In many cases, only some of the trees in the forest need be retained, since the system can automatically clear the forest. For example, although two parse trees are constructed for the sentence *Jean n'a pas effectué de lancer* (*lancer* could be an infinitive verb or a noun), only the tree with *lancer* categorized as a noun is retained, as shown in Fig. 8.

```
(sc   (sn   (n    "Jean"))
      (svc  (sadv   (adv  "n"))
            (auxc   "a")
            (sadv   (adv  "pas"))
            (vc   (vpp  "effectué"))
            (sn   (det  "de")
                  (n    "lancer"))))
```

Fig. 8

At this level, the sub-categorization of the verb is of great help: this information is also stored in the dictionary of course. For example, *effectuer* does not allow an infinitive phrase as a complement. Similarly, in the sentence *il a remarqué Marie arrivant à toute allure*, *Marie arrivant à toute allure* could be an adverbial phrase, *Marie* could be the object of *remarquer* and *arrivant à toute allure* could be an adverbial phrase, finally *Marie arrivant à toute allure* could be the object of *remarquer*. The first hypothesis (tree) is rejected since *remarquer* is sub-categorized as requiring a direct complement.

Sub-categorization is used to clear the forest of trees, Fig. 9-12, resulting from the analysis of the sentence *il pense à l'envie de Paul de s'enrichir*.

```
(sc (spro  (pro-qu     "il"))
    (svc   (vc  (vconj "pense"))
           (sp  (prep "à")
                (sn (det  "l")
                    (n    "envie")
                    (sp   (prep "de")
                          (sn   (n "Paul")
                                (si (prep   "de")
                                    (svinf  (procl   "s")
                                            (vinf "enrichir")))))))))
```

Fig. 9

```
(sc   (spro (pro-qu  "il"))
      (svc  (vc   (vconj   "pense"))
            (sp   (prep    "à")
                  (sn   (det   "l")
                        (n     "envie")
                        (sp    (prep "de")
                               (sn     (n "Paul")))
                        (si    (prep   "de")
                               (svinf  (procl "s")
                                       (vinf "enrichir")))))))
```

Fig. 10

```
(sc   (spro (pro-qu "il"))
      (svc  (vc   (vconj "pense"))
            (sp   (prep  "à")
                  (sn   (det  "l")
                        (n    "envie")
                        (sp   (prep "de")
                              (sn     (n "Paul")))))
                  (si   (prep "de")
                        (svinf  (procl "s")
                                (vinf "enrichir"))))))
```

Fig. 11

```
(sc   (spro (pro-qu "il"))
      (svc  (vc   (vconj "pense"))
            (sp   (prep "à")
                  (sn   (det  "l")
                        (n    "envie")))
            (sp   (prep "de")
                  (sn   (n "Paul")
                        (si   (prep  "de")
                              (svinf (procl "s")
                                     (vinf "enrichir")))))))
```

Fig. 12

The sub-categorization information for the verb *penser* allows us to eliminate the trees of Fig. 11 and 12. Since *Paul* cannot be sub-categorized by an infinitive sentence, as *peur* can be (*la peur de s'enrichir*), the tree in Fig. 9 can also be eliminated. The only remaining analysis is the tree in Fig. 10.

Verb sub-categorization also allows the system to correct some spelling mistakes at this stage. For example, the sentence **il panse que Marie viendra* will be corrected to *il pense que Marie viendra* since *panser* does not accept a completive.

Similarly, in *il va *ou il veut*, **ou* is corrected to *où* . At this level we also correct, using information stored in the dictionary, an error of the type **quoique tu dises, je partirai* to *quoi que tu dises, je partirai* , since the sub-categorization of *dire* is not satisfied in the first case. It is also verb sub-categorization information which allows us to correct certain trees and improve others.

Consider the case of correcting a tree. For the sentence, *il punit qui ment*, initially *qui ment* is labelled as a sentence connected to the verb *punir* . Then, the sentence *qui ment* is relabelled as a noun phrase.

Consider now the case where the sub-categorization allows us to improve a tree. In the sentence *Pierre lira un livre cette nuit*, *cette nuit* initially labelled noun phrase, will be relabelled adverbial phrase since *lire* cannot be sub-categorized by two noun phrases, as *nommer* can be, for example.

5. CORRECTING SYNTAX ERRORS AND AGREEMENT ERRORS

Experience has shown that syntactic errors are relatively infrequent. For example, in a study of the syntax of primary school students /Dubuisson & Emirkanian 1982a/ /Dubuisson & Emirkanian 1982b/, out of 6580 communication units, only 79 (1.2%) were found to be ungrammatical. The unit of communication is equivalent to what the traditional grammar calls the sentence, that is the root sentence and any embedded sentences /Loban 1976/. We observed /Lafontaine, Dubuisson

and Emirkanian 1982/ that the most frequent problem is in the use of subordination (53% of the errors), the use of complex relative clauses in particular (24 cases out of 42). Children also have problems with multiple embeddings: in general when they connect an embedded sentence to another, the resulting sentence is ungrammatical, the main sentence being absent or incomplete. The other problems are related to coordination, to constituent mobility and to the use of clitic pronouns where we observed a strong influence from the oral.

As for relative clauses, we counted non-standard clauses as ungrammatical, though they follow rules as do the standard relative clauses. *La fille que je te parle* et *la fille que je parle avec* are examples of non-standard relative clauses whilst the sentence **la fille dont que je te parle* is ungrammatical.

We have chosen for now to focus our attention on two of these problems: complex relative clauses and sequences of clitics. As part of a previous research project, we developed algorithms for handling complex relative clauses /Emirkanian & Bouchard 1987/ and sequences of clitics /Emirkanian & Bouchard 1985/. For the sentence *la fille que je parle*, the syntax correction algorithm proposes *la fille de qui/dont/de laquelle/avec qui/avec laquelle/à qui/ à laquelle je parle* . On the other hand, in response to the sentence *la fille que je te parle*, the algorithm proposes *dont, de qui* and *de laquelle* as possible choices. Again it is the sub-categorization of the verb which gives us a handle on the problems with sequences of clitic pronouns. The program corrects **je lui aide* to *je l'aide* , for example. However, in most cases, only an error is reported, the system is unable to correct the error since it cannot identify precisely the referent of the clitic. **J'y donne* and **je lui donne* are examples of ungrammatical sentences; the system cannot propose with certainty the missing clitics: it will propose *la lui* , *le lui* , etc... in the first case and *le lui* , *la lui* , *lui en* , etc... in the second case.

During morphological analysis, based on the information gleaned from the dictionary, the information collected in the parse tree and the agreement rules of French, the system isolates the noun phrases and checks to see if the agreement rules for number and gender have been applied. It then checks for agreement between the subject and the verb. Note that, for example, in the case of **les belles chameaux* , the system proposes both *les beaux chameaux* and *les belles chamelles* . In response to the sentence **le professeur explique la leçon aux élève de la classes* , the system proposes *le professeur explique la leçon aux élèves de la classe* , *aux élèves des classes, à l'élève de la classe* and also *à l'élève des classes* , even if, based on our knowledge of the world, we know that the last answer is less probable.

The agreement rules which we have formalized, some of which are recorded in the dictionary, allow our system to correct the errors most frequently found in written text /Lebrun 1980/ /Pelchat 1980/. These errors are due, in particular for number agreement, to semantic interferences or to the proximity of other elements: for example, ** il veut être très riches* instead of *il veut être très riche* , **je les voient* instead of *je les vois* and ** Michel nous donnent des bonbons* instead of *Michel nous donne des bonbons* .

Finally, note that certain lexical ambiguities (there are relatively few remaining at this stage) could be resolved here: for example, this is the case for *le chouette anglais* , but *la chouette anglaise* still remains ambiguous.

6. CONCLUSION

The automatic correction of French text is a major project. Knowledge at many different levels must be integrated and coordinated in the system. Only the construction of a prototype can attest to the success of such an integration. We have developed a prototype of the correction program in LISP on a MacIntosh Plus. The behavior of the final system will be refined by weighting the rules according to their utility. Statistics gathered from many different users will help us tune the general behavior of the system whilst statistics gathered for a given user will allow us to tune the behavior of the system to the problems specific to that user.

REFERENCES

Boons, J.P., A. Guillet & Ch. Leclère (1976) La structure des phrases simples en français, Genève, Droz, 377p.

Catach, N. (1980) L'orthographe française, Paris, Nathan, 334p.

Catach, N., D. Duprez & M. Legris (1980) L'enseignement de l'orthographe, Paris, Nathan, 96p.

Clas, A. & J.P. Horguelin (1979) Le français, langue des affaires, 2e édition, Montréal, McGraw-Hill, 391p.

Dubuisson C. & L. Emirkanian (1982a), 'Complexification syntaxique de l'écrit au primaire', Revue de l'Association Québécoise de Linguistique, vol.1, n°1-2, pp. 61-73.

Dubuisson, C. & L. Emirkanian (1982b) 'Acquisition des relatives et implications pédagogiques', In: Lefebvre, Cl. (ed.): La syntaxe comparée du français standard et populaire: approches formelle et fonctionnelle, Gouvernement du Québec, Office de la langue française, pp. 367-397.

Emirkanian, L. & L.H. Bouchard (1985) 'Conception et réalisation d'un didacticiel sur les pronoms personnels', Bulletin de l'APOP, vol.III, n°3, pp. 10-13.

Emirkanian L. & L.H. Bouchard (1987) 'Conception et réalisation de logiciels: vers une plus grande intégration des connaissances de la langue', Revue Québécoise de Linguistique, vol. 16, n°2, pp.189-221.

Frase, L.T. (1983) 'The Unix Writer's Workbench Software: Rationale and Design', Bell System Technical Journal, pp. 1891-1908.

Friedman, M. (1984) 'WANDAH: Writing-aid and Author's Helper', Prospectus, University of California Los Angeles, 26p.

Grevisse, M. (1969) Le bon usage, 9e édition, Gembloux, Duculot, 1228p.

Gross, M. (1975) Méthodes en syntaxe, Paris, Hermann, 414p.

Johnson, S.C. (1983) 'YACC: Yet Another Compiler-Compiler', Unix Programmer's Manual, vol.2, New-York, Holt Rinehart and Winston, pp. 353-387.

Knuth, D.E. (1973) The Art of Computer Programming: Volume 3 / Sorting and Searching, Reading MA, Addison-Wesley, 722p

Lafontaine, L., C. Dubuisson & L. Emirkanian (1982) '"Fot s'avoir écrire": les phrases mal construites dans les textes d'enfants du primaire', Revue de l'Association Québécoise de Linguistique, vol.2, n°2, , pp. 81-90.

Le nouveau Bescherelle (1980) I-L'art de conjuguer. Dictionnaire de 12000 verbes, Montréal, Hurtubise HMH, 158p.

Lebrun, M. (1980) 'Le phénomène d'accord et les interférences sémantiques', Recherches sur l'acquisition de l'orthographe, Gouvernement du Québec, pp. 31-81.

Loban, W. (1976) 'Language development: kindergarten through grade twelve', NCTE Research report n°18.

Miller, L.A., G.E. Heidorn and K. Jensen (1981) 'Text-Critiquing with the EPISTLE System: an author's aid to better syntax', AFIPS Conference Proceedings, pp. 649-655.

Pelchat, R. (1980) 'Un cas particulier d'accord par proximité', Recherches sur l'acquisition de l'orthographe, Gouvernement du Québec, pp. 99-114.

Peterson, J.L. (1980) 'Computer Programs for Detecting and Correcting Spelling Errors', Comm. of the ACM, pp. 676-687.

Robert, P. (1967) Dictionnaire, Paris, Le Robert, 1970p.

Romanycia, M.H. & F.J. Pelletier (1985) 'What is a heuristic?', Computational Intelligence, vol.1, pp. 47-58.

Tomita, M. (1986) Efficient Parsing for Natural Language, Boston, Kluwer, 201p.

Tomita, M. (1987) 'An Efficient Augmented-Context-Free Parsing Algorithm', Computational Linguistics, vol.13, n°1-2, pp. 31-46.

Winograd, T. (1983) Language as a Cognitive Process, Volume I: Syntax, Reading MA, Addison-Wesley, 640p.