Implementation of a Principle-Based Parser based on On-Line Locality Principle

Massimo Paolucci

January 1994

Report No. CMU-LCL-94-1

Laboratory for Computational Linguistics

139 Baker Hall
Department of Philosophy
Carnegie Mellon University
Pittsburgh, PA 15213

Implementation of a Principle-Based Parser

based on On-Line Locality Principle

Massimo Paolucci - Master Paper

Committee

B. L. Pritchett, D. A. Evans, T. Gibson

Abstract

In this paper I present a principle-based parser based on Pritchett's theory of human sentence processing. The parser proceeds bottom-up and at each step it attempts to maximally satisfy the principles of the grammar; specifically, the parser satisfies the maximal number of theta and case requirements. This strategy does not guarantee that the best local choice will be the best choice globally; therefore the parser is obliged to backtrack. The reanalysis process is constrained by the **On-Line Locality Principle,** *which is responsible for the selection of the solution in points of ambiguity. As a consequence the* **On-Line Locality Principle** *is responsible for failures of the revision process in the sense that the revision process may fail to recognize the grammatical structure of some sentences because of this constraint. The theory underlying the implementation guarantees that the revision process fails with all and only garden path sentences. When the failure of the revision process is caused by the* **On-Line Locality Principle** *the parser attempts to build a different structure that violates the* **On-Line Locality Principle,** *but nevertheless satisfies the grammar. If such a structure is found, then the sentence is potentially a garden path and an appropriate message is given.*

With an extension of the revision process to NP-traces, some island violations can be seen as a side effect of the parsing process. Following this idea, island violations are a special kind of garden path involving traces instead of open NPs.

Contents

1. Introduction

One of the bases of modern theories of grammar is the distinction between linguistic competence and linguistic performance. Linguistic competence is the native speakers' knowledge about the grammar of a language. Linguistic performance is the procedure followed by the Human Sentence Processor to parse a sentence. The distinction between competence and performance is more evident when we encounter sentences that are grammatical but nevertheless are recognized as unacceptable by native speakers; this phenomenon is the so called *garden path*. The Human Sentence Processor uses the competence of the grammar, but, because of performance problems, fails to recognize garden path sentences as grammatical.

An example of a garden path sentence is the following:

1 ¿ *The boat floated down the river sank*

Sentence (1) is normally recognized as unacceptable by native speakers, but this sentence is nevertheless grammatical, as can be shown by comparing sentence (1) with the following sentence

2 *The boat that floated down the river sank*

The question that arises, as a consequence, is why the listener fails to recognize that *"floated down the river"* is a reduced relative phrase adjunct to the subject NP *"the boat"* and the verb of the matrix clause is not *"floated"* but *"sank"*. One factor is clearly that the verb *"floated"* is ambiguous; under one interpretation it is a past tense verb of a matrix clause, under the other it is a participial verb of the relative clause. For some reason, whose explanation is the kernel of the theory underlying the work presented here, the verb *"floated"* is at first recognized as a past tense verb, then the listener is not able to reanalyze it as a participle. Understanding garden path phenomena is equivalent to understanding the constraints that induce these errors in the parsing process.

The parser presented here fails to recognize garden path sentences as acceptable; it is driven by the principles of the grammar and by processing constraints that are responsible for the selection of the solution at points of ambiguity. As a consequence, they are also responsible for the failure of the revision process when the parser realizes that a wrong selection was made before. The objective of my work is to build a parser that fails to analyze a sentence when a native speaker fails too, and that realizes the failure in the same position (roughly the same word) in which a native speaker that listens or reads the same sentence would realize it.

1.1. *Importance for Computational Linguistics*

There are various reasons why this work is important for computational linguistics. First, parsers are usually driven only by the grammar, and the algorithm that the parser follows is responsible only for the selection of the rules that should be applied in a particular situation. Usually there is no difference between the set of sentences that are predicted to be acceptable by the grammar and the set of sentences that are acceptable by the parser. The parser I present, on the other hand, may fail on sentences that are predicted to be acceptable by the grammar. At first glance, to have a program that fails where other programs succeed seems to be interesting only from a linguistic point of view; what I claim is that there are problems in computational linguistics that can be solved using the constraints listed later.

The main problem computational linguistics must deal with is the large number of ambiguities in natural language sentences. This problem has two main aspects. On one side, the high number of ambiguities produces a computational overhead that decreases the efficiency of the parsing process; on the other side, given an ambiguity, we want to choose exactly the same solution chosen by a native speaker. This is a

very big problem because often the linguistic analysis of a sentence leaves open so many ambiguities that it seems impossible that people can understand natural language sentences. Nevertheless we usually do not have problems in understanding ambiguous sentences, and often we are not even aware of all of them. Understanding garden path phenomena is one step towards understanding constraints that the Human Sentence Processor follows in the analysis of ambiguous sentences. Potentially, these constraints might be used to reduce the number of ambiguities and to improve the parsing process.

Also, as computational linguists, we are expected to design systems which automatically generate natural language sentences. The generation of sentences that are unprocessable for the listener will not increase the quality of our work. Being able to detect when a sentence is a garden path can help to avoid the generation of sentences that, in fact, are not processable.

1.2. *Structure of the Paper*

In this paper I present principle based parsing as a framework for parsing within Government and Binding Theory. Then I present the theory of garden path sentences that is implemented in the program I built. Finally, a description of the program and details of the implementation of the revision process will be presented. The last section of the paper **will** describe future potential directions of development.

2. **Principle-Based Parsing**

There is a wide attempt across various linguistic theories to move from a rule based-approach to a principle-based approach. A rule-based theory describes linguistic phenomena with a set of rules that are construction-specific and possibly language-specific. In a rule-based approach, it is very difficult to generalize from language-specific rules to cross-linguistic regularities and to capture the fundamental notion of Universal Grammar. To address these problems, linguists have moved from a rule-based approach to a principle-based approach. Principle-based theories use a set of general principles that describe the behavior of linguistic phenomena in all languages and a set of parameters, or a set of language-specific rules, that specialize principles of the grammar for specific languages. As a consequence, in a principle-based approach, there are no construction-specific rules. Each construction is the result of the interaction of principles in the grammar.

As an example, in a rule-based system the rule for passive in English is probably something like the following:

3 *S -• NP BE-aux V-past-p*

In a principle-based approach, the passive construction is a consequence of the interaction of principles. Following Government and Binding Theory (Chomsky 1986,1), an inflection assigns case in subject position, and a verb in participial form assigns a role in complement position only. The object noun phrase requires case and role to satisfy the principles of the grammar; it has a role in its base position and moves to subject position to have case. In this analysis Theta Theory, Case Theory, Bounding Theory and Empty Category Principle, the basic principles of the theory, are involved and their interactions produce the description of the passive construction. The same principles are responsible for all linguistic phenomena.

Principle-based Parsing is a consequence of this generalization. Traditionally, parsers are driven by a predefined strategy and a set of rules that describe the grammar. The strategy (eg. instance top-down vs bottom-up, shift-reduce vs. chart-parser) is responsible for the selection of the rule that should be applied in a specific state of the parsing process. The set of rules is responsible for the description of the grammar, with each of the rules describing a specific construction. For example, referring back to the example in the previous paragraph, there should be a specific rule for the passive construction that is selected when the input sentence is passive.

Principle-based parsing tries to shift from a rule-based approach to a more general approach in which the parser directly or indirectly uses principles of the grammar as stated in the underlying linguistic theory. As in the case of linguistic theories, the parsing process is the result of the interaction of principles. The great hope of principle-based parsing is that by following generalizations that are linguistically motivated, it is possible to encode the Universal Grammar in the parser; then a (possibly small) set of parameters should specialize the universal grammar to a language-specific grammar.

It is often wrongly assumed in the literature that principle-based parsing is equivalent to parsing with Government and Binding Theory. In general this point of view is wrong; other linguistic theories like GPSG (Gadzar et al 1987), HPSG (Pollard, Sag 1987) and LFG (Kaplan, Bresnan 1982) are principle-based and in these frameworks it is possible to base parsing on principles. Interestingly, even though a general theory of principle-based parsing is still missing and it is difficult to generalize from different experiences, some general problems, related to overgeneration and principle ordering in the parsing process, are common to principle-based parsing in different linguistic theories.

The major problem of principle-based parsing is that principles are a kind of formalization that is more abstract, but as a consequence more difficult to use, than rules. How to apply rules is well known: a rule is called from another rule following some well-defined order; but a general and efficient way of applying principles is unknown. Principles are independent, but they encode constraints that can interact; the incorrect application of principles can produce an incorrect failure or success of the parsing process or it can decrease its efficiency.

The parser presented here uses a grammar based on Government and Binding Theory, so the discussion about principle-based parsing developed in this paper is restricted to the GB framework. It is beyond the scope of this paper to try to present a generalization of principle-based parsing for different theories.

2.1. *Principle-Based Parsing for Government and Binding*

Government and Binding is based on five main principles:

X-bar Theory: Describes the general rules governing the form of the parse tree.

Case filter: States that NPs require a case in order to be licensed.

Theta Criterion: States that NPs require a role in order to be licensed.

Binding Theory: Describes how pronouns are related to their antecedents.

Empty Category Principle: Describes the behavior of empty categories.

Bounding Theory: Restricts the possibilities of movement.

From a principle-based parsing perspective, principles play two different roles: some principles can be seen as generators in the sense that they generate possible structures; while others are used as filters that weed out structures that do not satisfy certain requirements. Among the principles listed before X-bar Theory, Empty Category Principle, and Binding Theory can be used as generators; Case filter, Theta Criterion and Bounding Theory can be used as filters. A trivial algorithm can follow the strategy of applying generators that generate possible structures, then apply filters that weed them out when at least one filter is not satisfied. This is an instance of a well-known algorithm in Artificial Intelligence and usually called "Generate and Test". This algorithm is extremely simple, but unfortunately extremely inefficient; it generates potential solutions that are likely to fail the test step; also the termination of the process is not guaranteed. In the case of principle-based parsers, candidate structures are generated and they pass more then one filter before the correct filter rules them out. From these observations it follows that the main problem with principle-based parsing is overgeneration: generators create many possible candidates for the final structure, but only few of them can pass through all the filters of the grammar. As an example consider the following sentences.

4 * *Johrii thought that Billj saw himselfi*

5 *Johrii thought that Billj saw himselfj*

6 * *Johrii thought that Billj saw himselft*

The coreference indexings in the above examples are the results of applying a generator of coindexing relations. In this case there is an overgeneration of possible structures and the filtering process is applied three times with two inevitable failures.

Two main heuristics can increase the efficiency of the parsing process: generate only structures that are likely to pass through all filters, and apply filters that constrain structures as soon as possible. Referring to the previous example, binding theory constrains indexing, hence it should be applied as soon as possible to weed out structures that lead to an ungrammatically.

Another heuristic that can improve the efficiency of the parsing process is to use linguistic information available at a particular state in the computation and derive from it the sequence of operations that should be applied at a specific point. One may look for an optimal order of principles that produces the most efficient parsing process, valid for all sentences, but unfortunately, though not unsurprisingly, this optimal order does not exist. This result, shown by Berwick and Fong (1990), suggests an alternative approach. The optimal computation can only be the result of a flexible order that makes use of local information on the current linguistic requirements of that sentence. Berwick and Fong (1990) suggest a two-step parser. In the first step, the parser decides the optimal sequence of operations the parser should perform; in the second phase, the real parsing process is performed following the sequence of operations established in the previous phase. Attempting to build an optimal ordering should not introduce a flow in this process. Therefore only inexpensive tests can be performed in this phase. In this case there is a trade-off between the validity of fit of the test and the reliability of the generated sequence of parsing steps.

Berwick and Fong (1990) use mainly lexical clues to derive the optimal order of operation. Examples of the clues they use are the following.

Trace apply ECP and Case filter

Passive apply Theta criterion, Case filter

+Anaphoric apply Binding condition A

+Pronominal apply Binding condition B

Following the previous clues, the parser will not apply the ECP when no trace is produced, or at least the application of the ECP is delayed as long as possible.

The optimal order determined by the clues suggested above usually should be modified to give an account of the logical relations between principles. The Case filter, for instance, cannot be applied before the application of the case assignment operations (Inherent Case Assignment and Structural Case Assignment).

The final result of their work is a parser that can efficiently parse different linguistic phenomena using only principles defined in the grammar. The resulting parser is a good computational device, but because of its essential use of the two-stage parsing process it cannot be used as a valid psycholinguistic model.

2.2. *Parsing as Deduction Hypothesis*

A formal approach to principle-based parsing was developed by Johnson (1992) in an attempt of formalize Government and Binding theory in a logical framework. He regards principles of Universal Grammar as

axioms of a formal logic theory. Lexicon and parameters are used to specialize principles into the description of a language specific grammar. An automatic theorem prover for the logic in which the grammar is described is associated with the set of axioms. Parsing a sentence becames equivalent to proving a theorem.

Johnson defines the parsing of a sentence as the computation of a relation from the Phonetical Form to the Logical Form. Following his axiomatization a sentence is well formed if and only if there exists a quadruple (Deep Structure, Surface Structure, Phonetical Form, and Logical Form) that satisfies the relations encoded in the axioms of the theory.

Universal Grammar is expressed as a set of axioms in the form of Horn clauses as in the following example.

7

```
parse(String, IF) ;- xBar(infl2_f DS),
                     theta(infl2, DS),
                     moveAlfa(Ds, SS),
                     caseFilter(infl2, SS),
                     phonology (String,SS),
                     lfMovement(SS,LF).
```

Example (7) shows that the parsing process is a relation between the input string and the Logical Form. This relation is valid if and only if all six principles of the grammar are satisfied.

Johnson was interested in proving that the computational process underlying the language use can be seen as a sequence of deductive operations. He wants to show that the whole grammar can be encoded within a logic axiomatization and that all deductive steps are meaningful. In particular this implies that he should verify whether the resulting logic is sound and complete in the sense that it can cover all linguistic phenomena.

The *parse* relation described in the previous example implements the idea that we want to capture, but it is very inefficient. This formulation is equivalent to the "Generate and Test" algorithm. Johnson suggested that the parsing function is more efficient if the evaluation of each principle is delayed until the input structures of the sentence are all instantiated. This feature implemented with the *freeze* operation of Prolog2, introduces an automatic reordering of the principles that avoids vacuous applications. Johnson also improves the efficiency of the parsing process as a consequence of the use of partial evaluation techniques that automatically modify the program. The final result of the partial evaluation is a parser that computes the relation between the input string and the logical form, avoiding the computation of the intermediate forms. Unfortunately the resulting parser is not described in his paper.

2.3. *Indirect Application of Principles*

The approaches shown before are based on an explicit use of Government and Binding principles. The problem is that in order to explicitly use principles of the grammar we must build a new set of heuristics that selects principles. There is another possibility open: instead of using principles directly, it is possible to use principles in an indirect way. In this case, principles are compiled and transformed in such a way that they can be used directly. In general, this is a process of partial evaluation that produces a rule-based grammar that can be used by parsers which use the traditional algorithms.

This approach is taken by Abney (1991). His parser is divided into two different stages: the chunker and the attacher. The chunker splits the input string into chunks of 2 or 3 words. He defines chunks in the following way:

The root of the Chunk headed by a major head H *is the highest node in the parse tree for which* H *is the* s-head.

Major heads *are all content words except those that appear between a function word and the content word that it selects.*

If the syntactic head H *of a phrase P is a content word,* H is *also the* s-head *of P. If* H is *a function word, the* s-head *of P is the* s-head *of the phrase selected by* H .

The rules described above describe the form of the chunk and are used by the chunker to split input sentences into sequences of chunks. Chunks are partially instantiated trees that should be attached together to form a tree. The attacher links together all chunks. Both the chunker and the attacher are driven by specific requirements of lexical items. For each lexical item, what it licenses and from what it is licensed is specified; this information is used at parsing time when the chunker attaches different items or when the attacher attaches two chunks.

2.4. *An Example of use of Parameters*

Dorr (1992) described a principle-based approach to machine translation. The goal of her work was the construction of a translation system that uses a parser and a generator driven by a principles and parameters approach. Following a standard approach to principle-based parsing, she used a set of principles that are common for all languages and a set of language-specific parameters. She was able to parse and generate sentences in different languages, using variations of parameter-settings. In this framework, to switch from one language to another does not require a complete redefinition of the grammar, but only a resetting a set of parameters.

Dorr uses the six principles of Government and Binding: X-bar, Bounding Theory, Theta and Case Theory, Binding Theory and the Empty Category Principle. Each principle is sensitive to a definite set of parameters that changes the application of the principles in different languages.

What follows is an example of the parameters used by Dorr in her parser.

X-bar Defines the form of the syntactic structure.

- Constituent Order: svo/sov languages;
- Choice of Specifier: define the specifier of a head.

Theta theory Defines the thematic assignment of the argument of the verb.

- Clitic Doubling: Spanish, French, Italian and romance languages in general allow cliticization of object NP when the object NP is a pronoun that refers to an NP in the context.

Trace theory - Null subject: Italian and Spanish allow pro-drop;

- Choice of traces: NP traces or other types of traces.

The algorithm for the parsing process is based on the use of principles as constrains on possible syntactic structures.

To parse sentences, Dorr used a Earley parser with coroutined principles that constrains the structures that the parser built. As seen above, Dorr assumed a parametrized X-bar theory. From the parameters and principle for the X-bar theory, Dorr derived a set of context-free rules that forms the grammar that the Earley parser follows.

During parsing time the Earley parser follows the rules that are produced by the compilation of the parameters and the X-bar theory. When there is a situation in which the parser has no more moves possible, the parser applies principles of the grammar. At this point the Case Theory and the Theta Theory are applied and empty categories are predicted.

This coroutine of structure generation using the X-bar theory and structure-constraint principles guarantee a control of the overgeneration of the X-bar principle with constant adherence to the constraining principles.

3. The Psycholinguistic Data and the Underlying Theory

The theory underlying my work is presented in Pritchett (1992). The basic assumption of this theory is that the parsing process locally maximizes the satisfaction of grammatical principles. Sentence (1) above is repeated here as (8).

8 & *The boat floated down the river sank*

The local maximization of the satisfaction of grammatical principles in this case predicts that the verb *"floated"* is interpreted as past tense and not as participle of the verb *"to float";* the past tense verb can license the role of subject *"The boat",* whereas the participle fails to do that. The ambiguity is locally solved in this case. The problem is to explain why the maximization process fails when the verb *"sank"* is reached. The same maximization process suggests that if *"sank"* is seen as the predicate of the matrix clause then the verb is reanalyzed and the clause *"floated down the river"* is attached as a relative clause in an adjunct position of [$_{np}$ the boat]. The maximization of grammatical principles can produce the correct choice in the local environment, but it cannot explain why the reanalysis process fails or why the sentence is recognized as unacceptable.

The consequence of this failure is the introduction of another constraint to the reanalysis process. This constraint should be able to explain why *"floated"* in sentence (8) cannot be reanalyzed as a participle. In Pritchett (1992) the constraint is presented as the following **Qn-Line** Locality **Principle:**

On-Line Locality Principle. The target position assumed by a constituent must be *governed* or *dominated* by its source position (if any), otherwise attachment is impossible for the automatic Human Sentence Processor.

Here 'source* and 'target' refer to the position of the revised structure in the parsing tree before and after the revision process.

Assuming the On-Line **Locality** Principle (hereafter OLLC), it is possible to explain why sentence (8) is judged as ungrammatical.

The boat floated down the river The reader tends to maximize the satisfaction of the principles and this is an easy task. The whole thing is analyzed as a matrix sentence. The verb *"float"* is analyzed as the verb of the matrix clause, the NP *"the boat"* is analyzed as subject of the verb, *"down the river"* is analyzed as a PP attached in adjunct position to the VP.

sank There is only one interpretation for this word: it can be only the past tense of the verb *"to sink"*. It must license an NP in subject position, but there are no unlicensed NPs available. A reanalysis is therefore required: the NP subject of the matrix clause should become the subject of the new matrix clause; the VP *"floated down the river"* should be reanalyzed as a relative clause and attached as a CP in adjunct position to the NP *"the boat"*. In this case, OLLC is violated, because the NP *"the boat"* is no longer dominated or governed by the subject position of the VP *"floated down the river"*. There is an upward movement of the NP and this movement violates OLLC.

The previous example predicts that sentence (8) can be saved if we allow two matrix clauses in the same sentence; this can be done with a conjunction.

9 *The boat floated down the river and sank*

In the former case the garden path is predicted by OLLC. Consider the following sentence and how OLLC predicts the garden path in the case of subject-object ambiguity:

10 *¿ Without her donations failed to appear*

Sentence (10) is a garden path because the reanalysis process violates OLLC. First, notice that the pronoun *"her"* has ambiguous case: accusative and the genitive.

Without her is interpreted as a PP where the pronoun *"her"* is interpreted as an accusative pronoun object of the PP.

donations requires a reanalysis of the PP: the pronoun *"her"* is reanalyzed as a genitive pronoun and attached in [Spec,NP] position; *"without"* licenses the NP *"her donations"* and all roles are licensed. It is important to see that OLLC is not violated in this case because the pronoun *"her"* moves from the object position of the PP to the subject position of the NP governed by *"donations"*. The new position is dominated by the object position of the PP; therefore OLLC is satisfied.

failed can be either past tense of the verb *"fail"* as in *"contributions failed to appear"*, or a participle of the same verb resolved as a reduced relative as in *"contributions (that) failed to appear"*; finally it can be an adjective as in *"failed factories"*.

If *"failed"* is recognized as a past tense verb, it projects the IP and it must license an NP in subject position. There is no NP available that can fill the subject role, so the reanalysis process reinterprets the structure of the sentence and checks whether there exists an alternative structure that can license the subject role of the verb and the object role of the preposition. In this case the reanalysis process should analyze the NP *"donation"* as subject of the verb and the PP *"without her"* as an adjunct in SPEC position. This reanalysis process requires a second reanalysis of the preposition *"her"* and reinterprets it as an accusative pronoun object of the preposition *"without"*. The second revision of the pronoun violates OLLC, because in this case the pronoun moves from the [Spec, NP] position to the [Obj, PP] position, but this target position is no longer dominated or governed by the source position. It is not possible to perceive this path, therefore *"failed"* should be analyzed in another way.

If *"failed"* is recognized as a participle then it can be attached as an adjunct of the noun phrase.

Finally if *"failed"* is recognized as an adjective then it is left unattached waiting for a noun to modify.

"Failed" cannot be recognized as a past tense verb, therefore this way is closed. We can assume that at this point of the computation the word is either left ambiguous between the participle interpretation and the adjectival interpretation, or we can assume that one of the interpretations is chosen. From the point of view of the theory both these hypotheses are equivalent.

to appear is an infinitival IP; it can be licensed by *"failed"* if it is recognized as a verb, but not if it is recognized as an adjective. The unacceptability of the sentence is fully predicted: *failed* is recognized as a participle and *"to appear"* is attached as a complement. The sentence is still missing the matrix clause therefore it is ungrammatical.

Sentence (10) has the opposite problem of sentence (8). In sentence (10) a past tense verb is interpreted as a participle to avoid a violation of OLLC, but then the sentence is missing a matrix clause. In sentence (8) the verb of the matrix clause cannot be reanalyzed as a participle in adjunct position otherwise OLLC is violated.

Interestingly in a sentence that allows *"failed"* to be an adjective the sentence can be saved, as can be shown by the following sentence.

11 WrtAoirf Aer *donations failed factories would be destroyed*

Another kind of garden path sentence is the following.

12 *$ After Todd drank the water proved to be poisoned*

After Todd projects a PP headed by *"after"*, with the NP *"Todd"* as complement. The structure at this point is $\backslash p_P$ *[pi* $\backslash p$ after] $[_{np}$ Todd]]].

drank projects an IP and a VP attached to the IP in complement position. The IP needs to license an agent role in specifier position. The only NP available for that position is $[_{np}$ Todd] currently bound in object position of the PP. The entire structure is reanalyzed; the NP moves from [comp, PP] to [spec, IP], and the preposition licenses the IP complement position. This revision does not violate OLLC therefore the parsing process can continue. The structure at this point is *[p_P [pi [p* after] $[_{p}$ $[_{np}$ Todd] [a *[i past)* $[_{vp}$ drink]]]]]

the water is licensed as object of the VP headed by 'drink'.

proved presents the typical ambiguity between the past tense reading and the participle reading.

If the past tense reading is allowed, *"proved"* projects an IP and a VP and binds the VP in object position of the IP. The IP should license a theta role in specifier position, but there is no NP available that can license that role. A reanalysis should remove $[_{np}$ the water] from the complement position of $[_{vp}$ drink the water] and attach it to the specifier position of the IP projected by *"proved"*. The reanalysis fails because the target position of $[_{np}$ the water] is no longer dominated by its source position.

If *"proved"* is recognized as a participle, it is attached to $[_{np}$ the water] in adjunct position.

to be poisoned is attached in complement position of the verb as infinitival phrase and the sentence is unacceptable because it lacks a matrix clause.

The analysis developed predicts two important things. The sentence can be saved if an intransitive verb like *"to arrive"* is used instead of a verb that is optionally transitive like *"to drink"*, or it can be saved if a matrix clause is added to the sentence.

As predicted above the following sentence (13) is not a garden path.

13 *After Todd arrived the water proved to be poisoned*

After Todd arrived The parsing process proceeds as above.

the water cannot be licensed by $[_{vp}$ arrived], therefore it is left unlicensed waiting for future licensing.

proved to be poisoned is recognized as the IP of the matrix clause and it licenses $[_{np}$ the water] in subject position. The whole sentence is recognized as perfectly grammatical.

As predicted above, the following sentence (14) is not a garden path, because a matrix clause is added at the end of the sentence.

14 *After Todd drank the water proved to be poisoned he fainted*

After Todd drank the water proved to be poisoned is recognized as a prepositional phrase following the same analysis developed above.

he fainted is recognized as a matrix clause, licensing the PP as an adjunct and the sentence is acceptable.

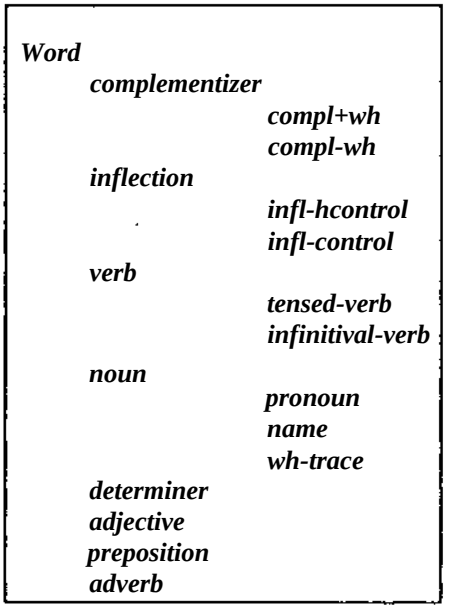
4. PLOP (Principle-Based Lisp-implemented Object-oriented Parser)

In this section the implementation of the parser is presented. Two main features of the program are: the construction of the lexicon which is fundamental for the encoding of the grammar and the mechanism of argument licensing and its extension to adjuncts which drives the parser. Also in this section the implementation of the revision process and the prediction of the garden path will be presented.

4.1. The structure of the lexicon

The lexicon is specified by the user in a file that is linked to the parser. This description specifies the grammar. The file contains the description of the lexical hierarchy and the description of each lexical entry. The following figure shows the lexical hierarchy that is actually implemented in PLOP,

15



The lexical hierarchy is defined as a class hierarchy. At the top of the hierarchy is the class *word* that defines a common structure for all the lexical entries in the lexicon. Subclasses of *word* are the nine main categories of lexical entries.

Words are then attached as subclasses to the class which describes their category. In this way each word inherits its grammatical requirements from its superclasses. At parsing time, an instance of the word is created and it inherits all grammatical requirements from its class. The prototypical lexical entry contains the information described in the following figure:

16

word *contains the literal word*

category *category of the word*

agreement *agreement information*

tense *tense information for verbs, undefined otherwise*

case *case information for the categories which need it, otherwise undefined*

role *role information for the categories which need it, otherwise undefined*

t-grid *theta grid of the word*

c-grid *case* ***grid*** *of the word*

1-adjunct *adjuncts licensed on the left side*

r-adjunct *adjuncts licensed on the right side*

t-need *requirement of the theta role*

c-need *requirement of case*

The user specifies only the features that are idiosyncratic to the lexical entry, leaving the definition of other features to the class inheritance. The category information is then used to link the lexical item to its class, and, as a consequence, it fills all the entries that are left unfilled by the user.

For example of the determiner *"the"* is defined as follows:

word the

category determiner

In the case of a pronoun more information should be specified, such as agreement and case. Here too the theta and case requirements are controlled by the class inheritance.

word his

category pronoun

agreement '(I^s 2s 3s)

case (genitive)

Finally, the definition of *"believes"* is given below. In addition to the information specified above there are requirements for case and role assignment either to an accusative noun phrase or to an accusative complementizer phrase with flag +wh, or to an infinitival inflection phrase with flag +control.

```
word believes

category verb

agreement (3s)

tense (present)

t-grid ((complement (((phrase (np))
                      (case (accusative)))
                     ((phrase (cp))
                      (category complementizer-fwh)
                      (case (accusative)))
                     ((phrase (ip))
                      (category inflection-fcontrol)
                      (tense (infinite))))))

c-grid ((complement (((phrase (np))
                      (case (accusative)))
                      (phrase (cp))
                      (category complementizer-fwh)
                      (case (accusative)))
                     ((phrase (ip))
                      (category inflection+control)
                      (tense (infinite)))))
```

Here too note that the transitivity is an idiosyncratic property of the specific verb.

4.1.1. *Complementizer*

Complementizers license an inflection in complement position and, depending on the class a complementizer belongs to, license a wh-word in specifier position. Complementizers require an optional licensement as arguments or as modifiers. The optionality is due to the fact that complementizers can be heads of sentences, and therefore can be left unattached.

Complementizers are split into two classes described by the features -f/- wh. Complementizers with the feature -wh can allow wh-words as specifiers. In English these complementizers are inflections that are moved in complementizer position in question formation. Following this analysis the structure of the following sentence is given by the structure shown in (18).

17 ***Does the man love the girl?***

18 $[_{cp}$ *Doesi* $[i_p$ $[_{np}$ *the man]* infl,- $[_{vp}$ *love the girl\]]*

Note that under this analysis auxiliaries like *do, have* and *can* are ambiguous words; they can either be complementizers in questions or inflections in declarative sentences. The feature I want to stress here is that auxiliars are complementizers which allow wh-words in the specifier position, as in the following example.

19 ***Who does the man love?***

20 *UP* $[n_P$ *Whoj]* *doesi* $[_{cp}$ $[_{np}$ *the man]* $infl_{\ell}$ $[_{vp}$ *love tj]]]*

Complementizers with the feature +wh never appear related to a wh-word because they incorporate some features of wh-words. One of these complementizers is the word *"that"*. In English, a sentence like the following is not grammatical:

21 **** the man who thai...***

But both sentence (22) and sentence (23) are grammatical..

22 ***the man who...***

23 ***the man thai...***

Following the analysis that was briefly developed for questions, sentence (21) is ill-formed because *"that"* does not allow any free position for *"who"*. As a consequence the structures of sentence (22) and sentence (23) are the following.

24 $[_{np}$ ***the man*** $[_{cp}$ $[_{np}$ ***who]*** ***comp ...]]***

25 $[_{np}$ ***the man*** $[_{cp}$ ***that ...]]***

Note that the hypothesis that both *"that"* and *"who"* occupy the same position is ruled out by the ungrammaticality of the following example.

26 **** That the man does love?***

Compare sentence (26) with sentence (19). *"Who"* and *"that"* occupy the same position in both sentences, but sentence (19) is grammatical, and sentence (26) is not; therefore the distribution of *"that"* and *"who"* is similar but their structural position is different.

Summarizing, complementizers license an obligatory inflection in complement position and require an optional licensement as argument or adjuncts. Also, complementizers with the feature -wh can license an optional wh-word in specifier position.

4.1.2. *Inflection*

The behavior of the inflection depends on the tense associated with it. If the inflection is tensed, it licenses an obligatory case to a noun phrase in specifier position and an obligatory role to a verb phrase in complement position.

The infinitival *"to"* is an inflection whose tense is set to non-finite. In this case, the "// is ambiguous between a -fcontrol reading and a -control reading. To see the difference between the two readings, consider the following sentences:

27 ***The man believes the girl to be intelligent***

28 ***The man fails to be intelligent***

The major difference between the two sentences is the open NP that occupies the subject position of the infinitival clause. In sentence (27), *Hhe girl"* has case from *Ho believe"* and a role from the verb *Ho be".* In sentence (28) there is no open NP in subject position of the infinitival clause; this suggests that the verb *"fails"* cannot transmit case to that position. *"To believe"* selects an inflection that has the feature +control, because it provides control for the subject of the infinitival clause; *Ho fail"* selects an inflection with the feature -control because it cannot transmit case to the subject of the infinitival clause. The feature +/- control is also responsible for the projection of the PRO. An inflection with -control requires the projection of a PRO whereas an inflection with H-control does not have the same requirement. The difference between feature +/-control correctly predicts the ungrammaticality of the following sentences:

29 * *The man believes to be intelligent*

30 * *The man fails the girl to be intelligent*

4.1.3. *Verb*

The *verb* class licenses an obligatory role to a noun phrase in subject position and requires an obligatory licensing of itself as argument. The transitivity or intransitivity of the verb depends on what the verb selects as argument in complement position. A verb is transitive if it selects a noun phrase in complement position; it is ditransitive if it selects two noun phrases; otherwise it is intransitive. There is no need for a specific class for each types of verbs. Furthermore there is a straightforward solution for the verbs that are optionally transitive like *Ho drink",* since for these verbs the requirement of a noun phrase in complement position is optional. There is no need to stipulate any kind of ambiguity. Also verbs can license PPs both in argument and in modifier position.

Verb has two subclasses, *tensed-verb* and *infinitival-verb.* The two classes are not distinguished by their theta requirements but by their requirements of the X-bar projection. *Tensed-verb* requires the projection of a tensed inflection, while *infinitival verb* does not. For the licensing procedure, the projection of an inflection guarantees that the verb is licensed. In the case of *infinitival-verb,* the inflection should be present at the time of the projection of the VP.

4.1.4. *Noun*

The class *noun* requires that all its instances are licensed as arguments by a case assigner and by a role assigner. Optionally or obligatorily, a *noun* can license a determiner in specifier position or a genitive noun phrase. Also idiosyncratically, nouns can license arguments in complement position and all nouns can license adjuncts. The argument licensement depends on the particular subclass of *noun. Pronoun, name* and *wh-trace* are implemented as subclasses of *noun,* they inherit case and role requests of the superclass, but none of these subclasses allows arguments or modifiers.

4.1.5. *Determiner*

The class *determiner* requires licensing of itself as an argument, but does not license any argument or modifier.

4.1.6. *Adjective*

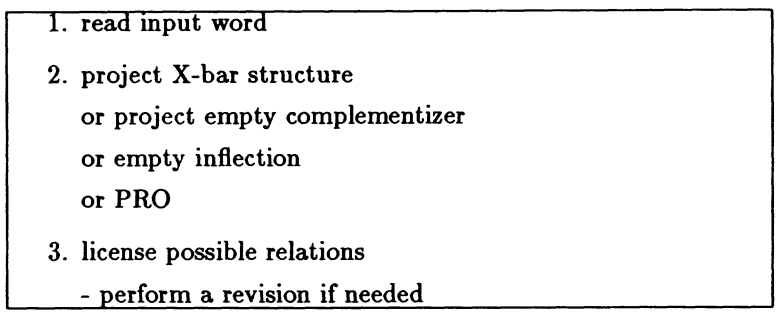
The class *adjective* requires licensing of itself as an argument or as an adjunct, but in the grammar it is always licensed as an adjunct.

4.1.7. *Preposition*

The class *preposition* requires a licensing of itself as an argument or as an adjunct. Also it obligatorily licenses a noun phrase in complement position.

4.2. Parse Time

The parser follows a shift-reduce algorithm. The control-flow of the parser is described in the following picture



4.2.1. *The projection of the X-bar structure*

The projection of the X-bar structure performs the shift operation of the parser. The parser builds the structure related to the lexical item and transfers it to the modulo that performs the licensement. The situation becames more complicated when the parser detects that some empty word should be projected. In that case the licensement of the current X-bar structure is postponed until the empty word is licensed. This means that the parser builds the X-bar structure for the empty word and licenses it; only after all the operations are done is the parser ready to license the current X-bar structure.

The parser follows very simple clues to decide whether an empty category should be projected.

1. The input word is a tensed verb. The parser projects an *empty inflection* with the same tense and person as the verb. This projection guarantees that a sentence like the following is grammatical.

   31 *John went to school*

   But the same mechanism guarantees that a sentence like the following is ungrammatical

   32 * *John is went to school*

   When the parser meets the word *"went"* it projects an empty inflection. At that point the parser has two inflections available for licensing the VP. The failure of the parsing process comes as a consequence; there is no way to license two inflections in that situation.

2. The parser should project an inflection, either an empty one or an open one. The parser checks whether in that environment an *empty complementizer* is needed. An empty complementizer is projected if there is a requirement for a complementizer as an argument of some licensing relation.

   A problem associated with this projection is the difference between the position of the open complementizer and the the position in which empty complementizers are projected. Consider the following sentences:

33 *Mary believes that John will come*

34 *Mary believes John will come*

In sentence (33) the complementizer *"that"* is projected immediately after the verb; in sentence (34) the empty complementizer is projected after the name *"John"*. But in the final structure of both sentences, the complementizer should license the same arguments, therefore in the final structure of both sentences the *[np* John] should be licensed in [SPEC, IP] position.

There is a fundamental difference between sentence (33) and sentence (34). In sentence (33) the complementizer is directly attached as argument of the verb; in sentence (34) [np John] is attached as complement of the verb and only the revision process frees that position for the complementizer. The difference in the point of projection is solved with a difference in processing.

3. The projection of PRO: As explained above PRO is projected only when the parser recognizes that the input word is an infinitival inflection with feature -control.

4.2.2. *Licensing Operation*

The licensing module performs the reduce operation of the parser. The result of the licensing of a relation is the attachment of two trees that encode the structure of two different parts of the sentence.

After the X-bar structure is projected the parser tries to license it. The licensing mechanism follows the shapes of unattached trees looking for nodes that have requirements that can be licensed by the new structure or that can license it.

There are two possibilities at this point.

1. Either some node has a free relation that can license the incoming node, or some node examined in the tree requires a licensement that can be satisfied by the incoming node. In the latter case the two nodes are immediately attached.
2. The relations are not free. In that case the attachment is possible if and only if a revision of the previous links is possible without violating OLLC.

As an example of the parsing algorithm consider how sentence (33) repeated here as (35) is parsed.

35 *Mary believes that John will come*

Mary is recognized as a noun. Nouns need both case and role licensing. The X-bar structure of the noun is projected: *[np* [ni [n Mary]]].

believes is recognized as a tensed verb, it requires the projection of an empty inflection. The inflection is projected and it tries to license the case of a noun phrase in specifier position. It licenses the structure of the noun *"Mary"*. The current structure in the parse buffer is [ip [np Mary] [i *infl\]]*. The inflection also requires a VP in complement position.

The structure of the verb is projected and attached in complement position of the inflection. The resulting structure is the following: *[ip [np* Mary] *[a* fc *infl\ [vp [vl [v* believes]]]]]. *"Believes"* licenses either a complementizer or a noun phrase in complement position.

that is recognized as a complementizer and is licensed by the verb. The structure at this point is the following *[ip [np* Mary] *[n [i infl\ [vp [vi [v* believes] *[cp [cl [c* that]]]]]]]. The complementizer licenses an inflection in complement position.

John is recognized as a noun phrase and is left temporarily unattached because there are no available positions for the attachment. There are two unattached trees, one is [ip [np Mary] [i' [infl\ [vp [vi [v believes] [cp [cl [c that]]]]]]], the other [np [nl [n John]]].

will is recognized as an inflection. Following the nearest tree it can license [np John] in specifier position. The inflection can also be licensed as complement of the complementizer. The current structure is: [cp [np Mary] [a [infl\ [vp [vi [v believes] [ep [el [e that] [ip [np John] [a [infl will]]]]]]]]]. The inflection license an infinitival VP within complement position.

come is recognized as an infinitival verb phrase and is licensed by the inflection *"will"*. The final structure is [ip [np Mary] [tl [infl [vp [vl [v believes] [cp [cl [c that] [tp [np John] [n [infl will][up [vl [v come]]]]]]]]]]]]

4.2.3. *The Revision Process*

The implementation of the revision process deals with the implementation of OLLC that is the basic condition of Pritchett's theory. Recall here that OLLC requires the satisfaction of a disjunctive condition: the target position should be *governed* or *dominated* by the source position.

The implementation of this principle in a parser which implicitly uses fundamental notions of the grammar requires the definition of the notions of *domination* and *government.* Domination can be easily defined as a structural relation in the tree, but government creates more problems.

The revision process is implemented following a different approach, OLLC leaves open only some possible structural revisions of the built structure. The program follows two fixed templates for revision.

1. The first template requires that the argument that fills the source position move to an argument position of another structure which fills the source position. It is a downward movement, the target position is structurally at a lower level then the source position in the parse tree.

   As an example consider sentence (34) repeated here as (36)

   36 *Mary believes John will come*

   Mary believes The parser follows the same path described for sentence (35). The structure at this point is [ip [np Mary] [n [infl\ [vp [v\ [v believes]]]]]. Remember that *"believes"* can license in complement position either a noun phrase or a complementizer.

   John is recognized as a noun phrase and is licensed as complement of the verb *"believes"*. The structure at this point is [ip [np Mary] [a [infl\ [vp [vi [v believes] [np [ni [n John]]]]]]].

   will is recognized as an inflection. In order to be licensed it needs the projection of an empty complementizer. Therefore a complementizer is projected and left temporarily unattached. Notice that the licensing of the complementizer is possible only in the complement position of the VP, but that position is occupied by [np John] so the revision is impossible, because of the violation of OLLC. If there were a revision [np John] would be left unattached and as a consequence the target position would be not longer dominated by the source position.

   After the licensing of the empty complementizer, the inflection is projected and licensed. First the inflection is attached to the complementizer, then the inflection tries to license [np John]. This time, the revision process succeeds: [np John] can be attached in specifier position of the inflection and the complementizer can be licensed by the verb. The final result is that the target position of [np John] is dominated by the source position and that there is a path from the source position to the target position that crosses only links bearing argument relations.

   The structure at this point is the same structure obtained by parsing the parallel sentence (33) in the same state of the parsing. [cp [np Mary] [{1 fc infl\ [vp [vi [v believes] [cp [ci [c that] [ip [np [m [n John]]] [a & will]]]]]]]]

come is recognized as an infinitival VP and licensed by the inflection *"will"*. The final structure is *[iP [np* Mary] *[ix* [- *infl*\ *[vp [vl [v* believes] *[cp* [ci [c that] *[ip [np [ni* [„ John]]] [;i [- will]^ *[vl [v* come]]]]]]]]]]

2. The second template implements the second disjunctive condition of OLLC. This condition requires that the revision is successful if the target position is governed by the source position. This condition holds for double object verbs. Consider the following sentence:

37 *John gave her letter to Mary*

Because the parser tries to maximally satisfy the grammar, the word *"letter"* is licensed as the second object of the verb. As a consequence when the parser tries to license the preposition *"to"* as the second argument of the verb, the parser must revise the structure. The dominance condition is not enough in this case and in fact it will predict that the parser will never recognize *pP* to Mary] as an argument of the VP, because there is no way the PP can license *[np* letter] as an argument.

The government condition solves this problem. *[np* Letter] licenses [np her] in specifier position and occupies the position of first complement. Note that the revision of the position of the *[np* her] is consistent with the dominance constraint so there are not problems on that side. The new position occupied by [np letter] is no longer dominated by the source position, but it is still governed by the same node as above.

The template for this revision is that the second argument of a double object verb can move to the position of the first argument if it is able to license it.

As an example consider how sentence (37) is parsed.

John gave The parsing proceeds as usual. The structure at this point is the following: [;p *[np* John] [;i *[i infl*\ *[vp* [„! [„ gave]]]]].

her is licensed as the first argument of the verb. The current structure is *[ip [np* John] [;i [- *inft*\ *[vp [vi [v* gave] *[np* her]]]]].

letter is licensed as the second argument of the verb. The current structure is [;p *[np* John] *[n* [• *infl*\ *[vP [vi [v* gave] *[np* her] *[np* letter]]]]].

to The only way to license this preposition is to move *[np* letter] to another position and free the second argument position for the preposition. The revision succeeds by using the second template. *[np* letter] moves to the first argument position and licenses *[np* her] in specifier position. The structure is the following: *[ip [np* John] *[n* [- *infl*\ *[vp [vl [v* gave] *[np [np* her] letter]]^ [pi [p to]]]]].

Mary is licensed in the complement position of the preposition. The final structure is the following: *[iP [nP* John] *[n [i infl*\ *[vp [vl [v* gave] *[np [np* her] letter]][pP [pi *[p* to] [np Mary]]]]]

4.2.4. *Lexical Disambiguation*

Often lexical items can have more than one syntactic interpretation. The parser I built can bundle these ambiguous lexical items. Consider sentence (10) presented here as (38):

38 *Without her contributions failed...*

As shown above *"failed"* is triply ambiguous: it can be an adjective as in *"failed factories"*, a participle that is resolved as a reduced relative as in *"contributions (that) failed to appear"*, or a past tense verb as in *"contributions failed to appear"*.

When an ambiguous lexical item is reached by the parser, the lexical item is first disambiguated, then projected and eventually attached to the current X-bar structure the parser is building. The disambiguation

phase of the process is driven by the request to maximally satisfy the principles of the grammar in the current state of the computation, such that the item which licenses the greatest number of licensing relations is selected. The revision process should also be able to deal with lexical disambiguation, often the revision of the current structure requires the revision of the interpretation given to a lexical items.

The function to evaluate the lexical disambiguation is computed on the basis of the following criteria.

- add one point to each argument that is licensed using the current lexical item;
- subtract ten points to each relation that should be licensed but is not;
- add 0.1 point to each adjunct that is licensed using the current lexical item.

When the score of each lexical item is computed, the parser selects the lexical item with the higher score. In case two or more lexical items have the same score, the parser randomly selects one of them, leaving to the revision process the responsibility to find the correct item for that situation.

As an example of how the parser selects lexical items consider what happens when the parser meets *Jailed"* in sentence (38).

- *'Failed'as* an adjective is neither licensed nor licenses anything in the current state of the computation. It scores 0 points.
- *'Failed*[9] as a participle can be licensed as a reduced relative by the noun in adjunct position, but points are assigned only to arguments. It scores 0.1 points.
- *'Failed'* as a past tense verb need to license a role in subject position. This licensing is impossible in the current state because it violates OLLC. There is not any possibility of license that requirement in the current state of the computation or in future states. It scores -10 points.

The parser selects the participial reading because that interpretation has the highest score.

The motivation for the scoring system is very simple. The objective is to select the lexical item that licenses as many relations as possible; also, it is important to avoid interference between argument licensing, adjunct licensing and arguments that cannot be licensed either in the current state or in future states and that will lead to an inevitable ungrammatical structure. The score of +1 point for each argument licensed allows an easy computation of the number of arguments that are licensed; the score of +0.1 for each adjunct licensed allows disambiguation between two lexical items that are distinguished only by the licensing of an adjunct, but is not enough to compete with licensing of arguments; the score of-10 points for unlicensable relations is a way of avoiding the selection of lexical items which lead to a failure of the parsing process.

4.2.5. *Prediction of potential garden path*

The failure of the revision process can be due to grammatical reasons; in this case the application of the revision process would lead to an ungrammatical structure, or to a violation of the OLLC. If the failure is grammatical there are no ways of save the sentence; if the failure is due to a violation of the OLLC the sentence may be grammatical but the parser is not able to recognize it. In this case the parser tries to fix the structure of the sentence using only grammatical information and not the OLLC; if the parser succeeds then the sentence is a *potential garden path.* It is important to stress here that this check is performed using only local information, and it is impossible to make predictions on the grammatical structure of the sentence. A sentence is a *potential garden path* when there exists a grammatical structure for the sentence that is hidden by the OLLC.

The difference between a *potential garden path* and a real *garden path* is due to the grammatical failure of the sentence. An important thing about the OLLC is that it introduces a processing constraint, which from a computational point of view, can be seen as a prune on a search tree. A sentence is a *potential garden path* when a grammatical structure can be found in the part of the search sub-tree that is pruned by OLLC; it becames a real *garden path* when there is no solution outside the pruned part. The violation of OLLC is impossible for the human sentence processor, but there is still open the possibility that the sentence can be saved by alternative attachments or by some new material that is coming.

The following sentences are examples of potential garden path:

39 ***l After Todd drank the water proved to be poisoned***

40 ***After Todd drank the water proved to be poisoned he fainted***

Sentence (39) is a real garden path because it is not acceptable for native speakers. Sentence (40) is only a potential garden path. When the parser reaches the word *"poisoned"*, in both cases it should recognize that up to that point the sentence is a *potential garden path,* because there does exist a structure of the sentence that is grammatical, but it violates OLLC. Sentence (39) is a real garden path, because the only grammatical structure is not reachable because of OLLC; whereas in sentence (40) the garden path is solved when a new matrix clause is found.

For the detection of potential garden paths the parser follows fixed templates too. These templates are based on the ambiguities that generate garden paths. As a general strategy, these tests are applied when the parser detects a violation of OLLC.

In the case of subject-object ambiguity the parser tries to break the potential subject in two arguments and to fill the subject position and the object position with one part. For example in the sentence:

41 ***l Without her contributions failed to appear***

The parser breaks *[$_{np}$* her contributions] into the two NPs *[$_{np}$* her] and [$_{np}$ contributions]. The first is licensed as an object by the prepositional phrase; the second, as a subject by the matrix clause.

If the sentence presents a 'fronted clause'-'matrix clause' ambiguity the parser tries to transfer the object of the fronted clause to the subject position of the matrix clause. For example in the following sentence:

42 ***After Todd drank the water proved to be poisoned***

***[$_{np}$* the water] moves from the object position of [t_p drink the water] to the subject position of the matrix clause *[i_p* the water proved to be poisoned].**

In the case of double object ambiguity, the parser tries to license the second argument as an adjunct of the first argument or conversely the first argument as an adjunct of the second argument that moves in first argument position. For example in the following sentence

43 ***The doctor convinced the patient he was having trouble with to leave***

The two arguments of the verb, *"convinced"*, are *[$_{np}$* the patient] and *[cp* he was having trouble with]. The second argument can be licensed as an adjunct of the first one, and then there is room for a new second argument.

4.2.6. *Adjuncts licensing*

Up to now I have explained how the parser deals with argument licensing, but the parser is also able to attach adjuncts. Adjunct licensing uses exactly the same mechanism of argument licensing so I will skip the details. In the grammar it is specified if a lexical item licenses adjuncts. This specification is similar to the specification of argument licensing. Also some lexical items like adjectives do not require licensing as arguments, but only licensing as adjuncts. When there is a request for licensing for an adjunct and a lexical item is available that matches the grammatical conditions for licensing, then the parser attaches the two structures.

The difference between adjunct licensing and argument licensing is the time of the attachment. For reasons that will be evident in the section relating to the subjacency effect, adjunct licensing is delayed as long as possible. Adjuncts are attached only when there is a link that crosses their position. There is an implicit condition that only adjacent structures can be attached together; and links cannot cross unattached structures. Also adjunct licensing adds nodes to the X-bar structure consistently with the X-bar theory.

As an example consider how the following sentence is parsed.

44 *The girl in the park loves the boy*

The **girl** is recognized as a noun phrase. The structure is the following: $[_{np}\ [_{d_p}$ the] $[_{n}$i $[_{n}$ girl]]]. Only one X-bar level is projected at this point.

in the park is recognized as a PP. The structure is the following: [pp [pi |p in] $[_{np}$ the park]]]. The structure is not attached yet to the structure of the subject NP.

loves is recognized as a verb. As described above the inflection is projected and it licenses an NP in subject position. The link which connects the inflection and $[_{np}$ the girl] crosses the position of the PP that cannot be left unattached. The parser, therefore, attaches the PP to the subject NP. The structure of the NP is $[_{np}\ [dp$ the] $[_{n}$i $[_{n}$i $[_{n}$ girl]] $[_{p_P}$ in the park]]]. Notice that is added a n-bar level to the structure to make room for the adjunct. The resulting NP is attached is specifier position of the inflection and the parsing process proceed as usual. If there are more then one PP available the parsing process adds more then one level; one for each adjunct.

As shown by the previous example, following Radford (1988) adjuncts are attached in X-bar position, but following other linguists (Haegeman (1991) among others) adjuncts are attached to the X-double-bar level. The parser presented here is not committed with any of this two choices. The selection of the attachment point is arbitrary and it can be changed without the introduction of any major change in the structure of the parser.

4.2.7. *Detection of ungrammaticality*

The detection of ungrammaticality of a sentence happens in three different places.

The first one is on line during the parsing of a sentence. If some obligatory requirement in specifier position is not locally satisfied then the sentence cannot be grammatical, and so is recognized as ungrammaticaL

As an example, consider the ungrammatical sentence

45 * *goes to Hollywood*

The verb *"goes"* requires the projection of an empty inflection that requires an obligatory licensing of an NP in subject position, but it cannot find one and there is no way that other words in the input can solve this problem, so the sentence is judged ungrammatical.

The second grammaticality test is also on line. This test is performed when the parser must license adjuncts. If there is no way to find a place for adjunct licensing the sentence is ungrammatical.

46 ** the girl goes beautiful to Hollywood*

The structure of the adjective *"beautiful"* is left unattached as explained above. When the verb wants to license [$_{pp}$ to Hollywood], it should attach the adjective as an adjunct, but this is impossible so the sentence is recognized as ungrammatical.

The last check is performed as last operation of the parsing process. In this test the parser checks if there is any obligatory requirement that is not satisfied. If this is the case, then the sentence is ungrammatical.

The only problem with the last test is that the parser in some cases cannot recognize as a sentence to be unacceptable in the same point, in the parsing process, in which a native speaker recognizes the failure of the grammaticality of the sentence.

5. Subjacency as a side effect of parsing constrains

In this section I want to show how the parser predicts some subjacency violations using OLLC and delaying adjunct attachment. This approach follows Pritchett (1992) which claims that OLLC is responsible for subjacency and that subjacency violation is a garden path. The theory is not fully developed and there are still problems and counterexamples that I will show later.

Subjacency is responsible for wh-movement and in general for long distance movement. Consider the two following sentences:

47 *What$_i$ do you believe John burned e$_i$*

48 ** What$_i$ do you eat after John burned e$_i$*

Sentence (47) is grammatical, while sentence (48) is not. The explanation of this phenomenon developed by Chomsky (1986,2) is that the maximal projection of *"after"* is a barrier that blocks extraction of the wh-word from its base position to its landing site. This idea of barrier correctly predicts that the following sentence is grammatical:

49 *What$_i$ do you eat e$_i$ after John burned the toast*

There is wide psycholinguistic evidence that gaps are postulated as soon as grammatically possible. This seems consistent with the fundamental claim that there is a maximal local satisfaction of the grammatical relations. If the claim is correct, then in the sentences (47), (48), and (49) presented above, the gap is postulated as soon as the parser recognizes that there is a request for a noun phrase. For example, in sentence (47), the parser can postulate a trace as a complement of the verb *"believe"*, while in sentences (48) and (49), the parser can postulate the trace as a complement of *"eat"*. When a trace is postulated, it behaves as any word would and it should satisfy OLLC.

Starting from these assumptions it is easy to see why sentence (48) is unacceptable and sentences (47) and (49) are fine. The parsing process of sentence (48) is described as follows.

What,- do you eat e,- As explained above, the trace is postulated as object of the verb *"eat"*.

after John burned The prepositional phrase, *"after John burned"*, cannot be licensed as an argument, so it is not licensed. Still, there is a requirement that should be filled, the object position of the verb *"burned"*. One possibility is to move the trace from the object position of the verb, *"eat"*, to the object position of the verb, *"burned*[39]. This revision is valid from the grammatical point of view, because the verb *"eat"* is optionally intransitive, hence, it can leave its object position free. It nevertheless violates the OLLP, because the new landing site of the trace is not longer dominated nor governed by the source position; in fact the target position is not even connected with the source position.

Sentences (47) and (49) do not violate OLLC. For sentence (49) this is trivial because there is not need of a revision. In sentence (47) things are slightly more complicated.

What,- do you believe e» The parser postulates the trace in the complement position of the verb *"believe"*.

John is not attached because there is no licensor for it. The complement position of the verb, *"believe"*, is occupied by the trace.

burned projects an inflection that can license *"John"* in specifier position and can occupy the complement position of *"believes"*. The trace moves from the complement position of *"believes"* to the complement position of *"burned"*. The target position of the movement is dominated by the source position, so OLLC is not violated and the sentence is recognized as grammatical.

The same mechanism predicts complex NP island violations. Consider for example sentence (50).

50 * *Who does Phineas know a boy who hates*

The parsing of this sentence, follows the usual way.

Who; does Phineas know e,- *"know"* is recognized as a transitive verb, and the trace is licensed in complement position.

a **boy** is recognized as a NP, but not attached because there is no place free where it can be attached.

who hates can be licensed as an adjunct by $[_{np}$ A boy], but it is left unlicensed.

hates can license the trace in complement position, but if the trace moves into that position it violates the OLLP. The sentence is ungrammatical because there is no way to license $[_{np}$ A boy].

For the examples presented above, the stipulation that adjuncts are not licensed until a crossing link forces their licensing can be seen as a useless stipulation. The violation can be produced also by a straight licensing of the prepositional phrase as adjunct of the verb, with the weaker assumption that the path that connects the source and the target position of the trace must cross only argument links.

Subject island violations are detected in the same way. Consider the following sentence.

51 * *Who do pictures of disturb Bill*

The important fact to see in these sentences is that the subject is not attached to the tree headed by the complementizer, so the trace cannot be licensed and the sentence result unacceptable.

who do As usual, *"do"* is recognized as a complementizer, it licenses *"who"* in specifier position.

pictures is recognized as a noun, and is left unlicensed because there is no available licensor for it.

of is recognized as a preposition, and it is licensed as complement of the noun, *"of"* is looking for a noun phrase that can fill the complement position, but the trace cannot attach here because this position is not connected to the complementizer tree.

disturb is a verb; it projects the inflection that licenses $[_{np}$ pictures of] in specifier position. There is no way to satisfy the requirements of the preposition so the sentence is recognized as unacceptable.

The explanation for the subject island is interesting from two points of view. First it suggests that the same conditions that the revision process should satisfy are the conditions that a wh-trace should satisfy: a trace should be *"dominated"* or *"governed"* by the wh-word that generates it. This condition is trivially violated in the case of subject islands because there is no connection between the wh-word and the trace.

Second subject islands suggest the idea that island violations are due to the lack of a connection between the wh-word and the trace. The hypothesis that adjuncts are not attached seems to suggest that the parsing of a sentence is mainly a process of licensing arguments, and that adjuncts are licensed only as consequence of argument licensing. I adopted this hypothesis and delayed adjunct attachment until it was forced by the licensing of some argument, but I believe that this hypothesis requires more independent support. The possibility that island violations are a consequence of the interaction of different phenomena is still open. In this case, subject island violations can be explained as a consequence of a lack of connection between the wh-word and the possible landing site for the trace; adjunct islands and complex NP islands are due to a direct violation of OLLC.

There are counterexamples in the theory that are not solved yet. For instance consider the following sentence:

52 * *What do you believe the claim that Otto saw?*

This sentence is an island violation, but the theory and the parser presented here predict it to be fine. The parsing process is as described below:

What do **you** believe The parser follows the usual way. It projects the CP and the IP and licenses the trace in the complement position of the verb *"believe"*.

the claim is licensed as a NP; it can license a sentential complement in the object position.

that Otto saw is licensed as a sentential complement in object position of $[_{np}$ the claim]. The revision of moving the trace from the complement position of the verb *"believe"*, to the complement position of the verb *"saw"*, does not violates the OLLC, and the sentence is recognized as grammatical.

The solution that Pritchett proposes for this violation, is the hypothesis that an intermediate landing site for the trace is lacking. Note that the revision is impossible when the parser recognizes $[_{np}$ the claim], because it cannot license a noun phrase as a complement. This is an additional condition that is not in OLLC.

The implementation of subjacency does not require any relevant additional functionality to the program. The only thing that is still missing is the store of the trace between its projection due to a wh-word, and the first attachment. When the parser recognizes a wh- word or a complementizer +wh, it puts a trace in a buffer. There are two types of traces:

1. IP traces, that are generated by the movement of the inflection in complementizer position. This kind of trace is used when the parser must project an inflection.
2. NP traces, that are generated by wh-words and complementizers with feature -fwh that are subject to subjacency. They are erased as soon as the parser needs a noun-phrase.

6. Conclusion

As is evident from the above description of my parser, I followed an implicit way to principle-based parsing. Principles are not directly implemented in the sense that there is no function that is responsible for the effective work of a particular principle, but they are used indirectly and are codified in the requirements of the lexical items. In this way the parser tries to satisfy all lexical requirements and does not have a problem with handling the coordination between principles, and the implementation of an efficient parser is much easier.

The work I did can be developed in different directions. I want to present some of them here.

There is an entire class of garden path sentences related to lexical ambiguities that the parser is not able to recognize. One of these sentences is the following

53 *i The old train the young*

In sentence (53) *"old"* is interpreted as an adjective and as a consequence *"train"* is interpreted as a noun instead of as a verb. The sentence seems unacceptable because the verb is missing in the matrix clause. The solution presented in Pritchett's theory requires the definition of a lexical rule that with an empty morphological transformation, transforms the adjective *"old"* into a noun that is morphologically identical. Therefore the implementation of this of garden path, requires the implementation of a morphological module that handle the redefinition of the lexical disambiguation.

The lexical selection in the program is computed by a function that is separated by the licensing part. This solution is not the best possible, and was mainly due to the lack of time and to avoid dealing with a deep revision of the architecture of the parser. An alternative that I think is more plausible from the psychological point of view and that will probably improve the efficiency of the computation is to combine the lexical selection with a parallel processing of unlicensed structures. For example consider sentence (38) repeated here as (54).

54 *Without her contributions failed to appear*

When the parser reaches the word, *"failed"*, it should decide which lexical item to project and with which category. Above, I presented the idea of counting the number of lexical relations licensed by each lexical item and of projecting the lexical item that has the best score. Following the new idea of a parallel implementation of lexical licensing I want to show how I think the parser could handle sentence (54).

After reaching *"failed"*, the parser would project all three lexical items and try to build three alternative structures, one for each of the alternative lexical items. The structure in which *"failed"* is interpreted as a past tense verb would be immediately discharged because the verb is lacking a subject. The remaining two structures would be left open waiting for the next word. It is easy to see that the parser will fail in both of cases because it discharged the only interpretation of *"failed"* that can license an infinitival clause and project the inflection for the matrix clause. Even though I think that this way is feasible, I have no idea of the complexity involved in implementing the parallel structures. I think that, in case this solution is selected, the parser should be able to avoid useless duplications of structures in memory every time an ambiguous lexical item is encountered in memory. An interesting possible way to follow is to implement a chart parser and avoid the production of links that violate OLLC. Also I have no idea yet if there are consequences for the correct prediction of the garden path.

Another direction of possible development is to change the hypothesis of maximal satisfaction of the grammatical constraints. There is wide psycholinguistic evidence from Crain and Steedman (1985) among others, that the Human Sentence Processor uses all the linguistic information available. For instance, at the beginning of the paper I referred to sentence (1), repeated here as (55), as a garden path.

55 *I The boat floated down the river sank*

Now consider what happens when in the context there are two or more boats and, in particular, one of them *"floated down the river"*. In that context, sentence (55) is no longer a garden path, and it is easy for the listener to recognize that *"floated"* is not the verb of the matrix clause, but a participle that is attached as a modifier in adjunct position to the subject noun phrase. In this case, the parser can recognize *"sank"* as the verb of the matrix clause. Evidently, in this case the context plays a fundamental role, and the listener must disambiguate the reference of the sentence and select which boat the sentence is referring to. It seems that the maximization hypothesis should be extended to pragmatic and semantic information. This change has major effects on OLLC, namely, that it should be sensitive to semantic and pragmatic information, too.

The prediction of island violations has counterexamples that should be solved, but there is no way to predict parasitic gaps like^n the following sentence.

56 *Mary kissea\ Bob and Louise, e- John*

Following Chomsky (1986,2) subjacency is responsible for the ungrammaticality of the following sentence

57 ** Mary kissea\ Bob and I believe Louise, e- John*

OLLC works with argument filling; but I don't see a natural way of explaining this phenomenon using OLLC.

7. Bibliography

Berwick, R. C. (1992); "Principle-based parsing." In R. Berwick, S. Abney and C. Tenny *Principle-Based Parsing: Computation and Psycholinguistics]* Dordrecht Kluwer Academy

Chomsky, N. (1986,1); *Knowledge of Language, Its Nature, Origin and Use]* New York Praeger

Chomsky, N. (1986,2); *Barriers]* MIT press

Crain, S and Steedman, M.(1985); "On not being led up the garden path: the use of context by the psychological syntax processor." In D. Dowty, L.Karttunen, and A. Zewicky (eds.), *Natural Language Parsing: Psychological, Computational and Theoretical Perspectives.]* Cambridge: Cambridge University Press.

Fong, S. 1992; "The computational implementation of principle-based parsers." In R. Berwick, S. Abney and C.Tenny *Principle-Based Parsing: Computation and Psycholinguistics]* Dordrecht:Kluwer Academy

Gadzar, G., Klein E., Pollum, G. K. and Sag, I. A. (1985); *Generalized Phrase Structure Grammar.]* Cambridge Mass.: Harvard University Press.

Haegeman, L. 1991; *"Introduction to Government and Binding Theory"]* Cambridge: Blackwell

Kaplan, J. A. and Bresnan, J. (1982); "Lexical Functional Grammar: A formal System for Grammatical Representations". In Bresnan (ed) —it The Mental Representation of Grammatical Relations.; Cambridge Mass.: MIT Press.

Pollard, C. Sag, I. A. (1987); *Information Based Syntax and Semantics]* CSLI Lecture Notes 13

Pritchett, B. 1992; *"Grammatical Competence and Parsing Performance"]* University of Chicago Press, Chicago

Radford, A. 1988; *"Transformational Grammars"]* Cambridge: Cambridge University Press