

For an Incremental Computation of Intrasentential Coreference

Paola Merlo

Université de Genève

LATL-Département de Linguistique

2 rue de Candolle

CH-1204 Genève Suisse

Abstract

In this paper, we present a set of algorithms to compute coreference, according to Binding theory, as presented in [Chomsky, 1986]. The novelty of the computational solution is two-fold: it is incremental and it computes both possible and impossible antecedents dynamically. Incrementality is crucial for the efficiency of a system, because the search space of possible analyses for a correct input sentence can be pruned earlier, and possible sources of ungrammaticality for incorrect input are discovered early in the parse. The local domain of coreference is computed dynamically, thus handling cases which previous proposals could not solve correctly.

1 Introduction

Intrasentential coreference is the grammatical phenomenon of two nominal expressions, which occur in the same sentence, referring to the same entity in the discourse. Thus, in example (1)a *John* and *him* can be, but might also not be, coreferent, while in example (1)b *John* and *himself* must be coreferent.

- (1) a. John's mother likes him
- b. John likes himself

A set of algorithms is presented to compute coreference, according to Binding theory, as presented in [Chomsky, 1986]. The novelty of the computational solution is two-fold: it is incremental and it computes both possible and impossible antecedents (such as *John* above) dynamically. By incrementality we mean that the algorithms are applied while building the syntactic representation for the input string. Other proposals ([Latecki, 1991; Giorgi *et al.*, 1990]) require that the entire tree be built, before the coreference algorithms can apply. In our proposal, we interleave the computation of coreference at every structure-building step of the parser, which is the interactive parser described in [Wehrli, 1992].¹

Incrementality is crucial for the efficiency of a system, because the search space of possible analyses for a

correct input sentence can be pruned earlier, and possible sources of ungrammaticality for incorrect input are discovered early in the parse.² Secondly, consider, for example, the sentence in (2).

- (2) They thought that [Bill liked them]

For sentences such as (2), Binding theory, and algorithms that implement it directly, [Giorgi *et al.*, 1990], check that *them*, a pronoun, is not coreferent with a noun phrase in its local domain, here indicated by the brackets: as *Bill* and *them* cannot be coreferent, the sentence is accepted. For practical applications, however, beside checking grammaticality, it is also important to be able to recover the possible antecedents of a pronominal expression, in (2) *they* is a possible antecedent of *them*. Thus we associate each noun phrase with two sets: one of possible and one of impossible antecedents, following [Ingria and Stallard, 1989]. The update of the two lists is incremental, as it is based on the structure-building algorithm, and it has wider empirical coverage than previous proposals, because the *local domain* of coreference is computed dynamically. More specifically, our algorithm can handle cases of coreference inside NPs correctly (cases 6 and 7 below), while [Ingria and Stallard, 1989] could not.

The paper is structured as follows: first, we present all the definitions that constitute the core of binding theory and related concepts, then we introduce the linguistic examples, and finally we present and discuss the algorithms.

2 C-command and Binding

According to binding theory, two notions are crucial to define the distribution of coreferring elements in a sentence: a topological relation between the two coreferring phrases in the tree (c-command), and a notion of domain, which determines how far away from each other the elements can be (Complete Functional Complex). The two algorithms that we propose deal with these two aspects of the computation.

[Latecki, 1991] defines a general P-command relation.

²We refer to [Fong, 1991] for experimental results on efficiency gains in parsing systems when filtering operations, such as computation of coreference, are applied as early as possible.

¹[Wehrli, 1992] is essentially a licensing parser, in the sense of [Abney, 1989], to which this algorithm is also applicable.

Definition 1 α P-commands β iff the first node with property P that properly dominates α also dominates β .

In this definition, the first node that dominates α means the parent of α .³ Given the general P-command relation in Definition 1, Barker and Pullum 1990 also argue that the notion of c-command can be expressed as a special instance of it (Def.3), where property P is the property of being a branching node, i.e. having more than one child (Def.2).

Definition 2 M , the mother relation, is given by
 $M = \{(a, b) | a >^D b \wedge \neg \exists x [a >^D x >^D b]\}$

The notation $a >^D b$ means that a dominates b .

Definition 3 C-command is the (P-)command relation with property P_1 , where P_1 is given by
 $P_1 = \{a | \exists x, y [x \neq y \wedge a M x \wedge a M y]\}$

We now proceed to define other notions used for Binding theory.

Definition 4 α and β are coindexed iff α has index i and β has index j , and $i=j$.

Definition 5 α binds β iff α c-commands β and α and β are coindexed.

The indexing I and the pair (α, β) , α a node and β a subtree, are compatible with respect to binding theory (BT-compatible) if α satisfies the binding theory in the local domain β under the indexing I . The following definitions are adapted from [Chomsky, 1986]:171-72.⁴

Definition 6 I is BT-compatible with (α, β) if

- A α is an anaphor and α is bound in β under I .
- B α is a pronominal and α is not bound in β under I .

Definition 6 states that the antecedent of an anaphor α must be inside a certain domain β (to be defined), while the antecedent (if any) of α a pronominal must be outside β . The complementary distribution of anaphors and pronominals is thus captured.

Definition 7 Licensing condition for a category α by a lexical governing category γ in the expression E with indexing I :

For some β I is BT-compatible with (α, β) , for α an anaphor or pronominal, and β the minimal Complete Functional Complex containing γ for which there is an indexing J , BT-compatible with (α, β) .

³[Latecki, 1991] shows that Definition 1 is equivalent to the definition of P-command in [Barker and Pullum, 1990], but it is an improvement from a computational point of view, as, according to this definition, only one node beside the currently processed node needs to be checked to determine whether the P-command relation holds between α and β , while in [Barker and Pullum, 1990] all nodes dominating α need to be checked.

⁴For the moment, we can assume that anaphors and pronominals are given by exhaustive listing. In the discussion that follows anaphors are words such as *himself*, *themselves*, *each other*, while pronouns are *them*, *they*, *theirs*.

Definition 8 (Giorgi 1987) γ is a Complete Functional Complex (CFC) iff one of the following holds:

- (i) γ is the minimal domain in which all the thematic roles pertaining to a lexical head are assigned;⁵
- (ii) γ is the minimal domain in which all the grammatical functions pertaining to that head are realized.

3 The Linguistic Evidence

The definitions given above correctly capture the following cases, where α is the referring nominal expression.⁶

Case 1 α is inside a VP and the local domain is the clause dominating the VP. If α is an anaphor, it can be coindexed with a c-commanding element, while if α is a pronominal it must be free.

- (3) a. I told *them_i* about *each other_i*.
 b. * I told *them_i* about *them_i*.
 c. * *They_i* thought that Bill liked *each other_i*.
 d. *They_i* thought that Bill liked *them_{i,j}*.

Case 2 α is the subject of a tensed clause. The local domain is the minimal clause containing α , since tensed inflection governs the subject position (as shown by Nominative Case (*they*).) If α is an anaphor the sentence will be bad, as there is no possible c-commanding antecedent in the minimal domain, while if α is a pronoun it will be free, or coindexed with an element in a dominating clause.

- (4) a. **They_i* expected that [_{IP} *each other_i* would win]
 b. *They_i* expected that [_{IP} *they_i* would win]

Case 3 α is the subject of a tenseless clause. The local domain is the clause dominating the clause containing α . This result descends from the licensing condition, which requires that the local domain contains a lexical governor for α . The subject of a tenseless clause is governed either by the verb or by the complementizer (*for*) of the dominating clause, as shown by accusative Case (*them*).

- (5) a. *They_i* expected [_{IP} *each other_i* to win]
 b. * *They_i* expected [_{IP} *them_i* to win]

Case 4 α is inside an NP, the NP has a subject and α is in complement position. The local domain is the NP itself, the minimal CFC.

- (6) a. They heard [_{NP} *my_i* stories about *ourselves_i*]
 b. * *They_i* heard [_{NP} *my* stories about *each other_i*]
 c. * They heard [_{NP} *John's_i* stories about *him_i*]
 d. *Bill_i* heard [_{NP} *John's* stories about *him_i*]

Case 5 α is inside an NP, α is in complement position, but the NP has no subject. The local domain is

⁵The relations between a lexical head, such as a noun or a verb, and its satellites are called thematic roles, for instance a transitive verb assigns the thematic relations of agent and patient to its arguments.

⁶We use the symbol IP, for Inflection Phrase, to define a clause in the bracketed representation.

the CFC containing the NP, namely the complete clause.⁷

- (8) a. *They_i* heard [_{NP} stories about *each other_i*]
 b. **They_i* told [_{NP} stories about *him_i*]

So far, all the cases have shown a complementary distribution of anaphors and pronouns, but in some cases pronouns and anaphors can occur in the same context, as shown by cases 6 and 7.

Case 6 In this instance, α is the subject of an NP, and α is pronominal. The local domain is the NP.

- (9) *The children_i* like [_{NP} *their_i* friends]

Case 7 α is the subject of the NP and α is anaphoric. The local domain is the clause containing the NP, because there is no BT-compatible indexation inside the NP itself.

- (10) *The children_i* like [_{NP} *each other's_i* friends]

4 The Algorithms

Other algorithms for the computational treatment of coreference have been proposed. We present a set of algorithms which capitalizes on previous work ([Giorgi *et al.*, 1990; Ingria and Stallard, 1989; Barker and Pullum, 1990; Latecki, 1991]), but it is different both in intent and practical features. Although [Ingria and Stallard, 1989; Giorgi *et al.*, 1990] present interesting treatments of coreference in the GB framework, both algorithms operate on a tree which is already completely constructed. Neither algorithm then is compositional and incremental. [Fong, 1991] has shown that the point at which filtering principles of the grammar are applied can make order-of-magnitude differences in parsing time. An incremental approach is to be preferred as it supports early applications of grammatical filters, that reduce the computational space.

4.1 The C-command Algorithm

The proposal by [Latecki, 1991], to compute c-command incrementally is modified. [Latecki, 1991] claims that the proposed indexing technique can be computed while building the tree. The algorithm, however, relies on a strict preorder traversal of the tree. This restriction cannot be met easily by many parsing algorithms, thus the incrementality of the indexing technique is not sufficiently general.

By using the definition of P-command relation from [Barker and Pullum, 1990], [Latecki, 1991] defines an indexing procedure to encode P-command relations in a tree, and proves Theorem 1, reported below for clarity. (They correspond to Def.2.1 and Theorem 2.4 respectively in [Latecki, 1991, p.53].)

⁷In this instance, an empty pronominal element can optionally be inserted, thus inducing a smaller local domain, as case 5 reduces to case 4.

- (7) a. *They_i* heard [_{NP} *PRO_i* stories about *them_i*]
 b. **They_i* heard [_{NP} *PRO_i* stories about *them_i*]

The sentence is grammatical only with the interpretation that the stories about them are not *their stories*, but *someone else's*.

Definition 9 Let P be any property. The function $\text{indp}: N \rightarrow F(N) = \{\sigma \subseteq N : \sigma \text{ is a finite subset of } N\}$ is defined recursively as follows:

- 1⁰ $\text{indp}(\text{root}(T)) = \{1\}$, where $\text{root}(T)$ denotes the root node.
 2⁰ if α immediately dominates β , then $\text{indp}(\beta) = \text{indp}(\alpha) \cup \text{fp}(\beta)$

where fp is a function $\text{fp}: N \rightarrow F(N)$, which fulfills the following conditions:

- 1¹ If $\alpha \notin P$, then $\text{fp}(\alpha) = \emptyset$.
 2¹ If $\alpha \in P$, then $\text{fp}(\alpha) = \{\tau\}$, for some unique index $\tau \in N$, $\tau \notin \bigcup \{\text{fp}(\gamma) : \gamma \in N \text{ and } \gamma \neq \alpha\}$.

Theorem 1 Node α P-commands node β iff $\text{indp}(\alpha) - \text{fp}(\alpha) \subseteq \text{indp}(\beta)$.

The procedural interpretation of this definition is a two-step algorithm: one function, corresponding to fp , which assigns a new index to each node with property P , and another function which percolates these indices down the tree. This procedure requires that the mother is always built before its children.⁸

We show how to interpret these definition with an incremental and compositional algorithm, which is interleaved in the structure-building operations of a licensing parser. A licensing parser can perform the following three operations: project, attach as complement ("right and down"), attach as specifier ("left and up"). We present the pseudo-code for the algorithm, we show that it computes the function in Definition 9, (so that Theorem 1, proved in [Latecki, 1991], also holds), and then we sketch an analysis of its complexity.

Algorithm 1 This procedure applies every time a projection is attached to the already built portion of the tree. Each node has two stores: one for the unique indexing assigned to the node (index), the other for the set of indices inherited by the parent node (index-set).

```
c-command-indexation(CurrentPtr:pointer)
begin
    update-root-of-phrase(CurrentPtr)
end

(* Step 1*)
update-root-of-phrase(PhrasePtr:pointer)
begin
    if PhrasePtr.branching-flag = false then
        PhrasePtr.index <- get-newindex
    endif
    PhrasePtr.branching-flag <- true
```

⁸Latecki notices that Theorem 1 does not directly apply to c-command, as defined in [Reinhart, 1983]. This is because [Reinhart, 1983] includes two special cases: the relation between the two sentential projections CP and IP, and the case of so-called "Chomsky-adjunction" where a maximal projection is the daughter of a maximal projection of the same category. [Latecki, 1991] shows, however, that the indexing procedure can be extended. These two cases, are treated as special cases, in which the index assigned to the node by fp is not added to the indp set. This is not an unjustified solution as these two cases are indeed violation of the well-formedness conditions on phrase structure as expressed in $\bar{X}$ theory.

```

percolate-index-to-children(PhrasePtr)
end

get-newindex
(* generating function that outputs
   unique numerical indices *)

(*Step 2*)
percolate-index-to-children(Root:pointer)
(* any type of complete tree-traversal algorithm,
   which makes a copy of the newindex for each
   traversed node, and assigns it to the indexset
   store, with the exception of CP and
   Chomsky-adjunction
*)

```

In practice, *PhrasePtr* can have two values in English. If the attachment is a complement attachment, it corresponds to the parent node of *CurrentPtr*, while if the attachment is as specifier, it is *CurrentPtr* itself.⁹

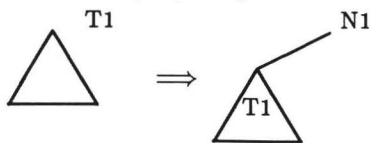
This algorithm performs an indexation like the one defined in Definition 9. On inspection of the algorithm, we note that a node obtains a new index only if it has property P (since the algorithm is called only upon attachment, and we are testing for property P of node α , α a branching node, which is always true when the algorithm is called.) *Branching flag* is an overloaded boolean variable which keeps track of whether the algorithm has already applied to a given node or not. Thus Step 1 is function *fp* in Definition 9.

A node inherits a distinct index from all of its ancestors with property P, as a consequence of *percolate-index-to-children*, which is by definition a complete tree traversal of the subtree rooted at the parent of the node which has just been attached to the tree. Thus Step 2 of the algorithm is function *indp* in Definition 9.

Consider what happens at each type of attachment. T1 is the tree before the attachment, and N1 is the newly created projection. There are two cases.

Case 1: T1 attached as Specifier to N1

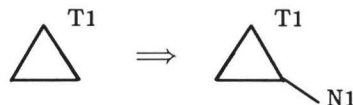
In this instance the new index of N1 is percolated to all the nodes in $T1 \cup N1$, as N1 becomes the root of the new tree. Hence, *indp* and *fp* hold for the *attach as specifier* operation.



Case 2: N1 attached as complement to T1

⁹In general, this value depends on the interaction of incremental tree-building (necessarily from left to right) with the word order of the language. For example, in a language where the complement precedes its head, like German, *CurrentPtr* and *PhrasePtr* are equivalent. These cases are predictable. It is indeed not unexpected that it should be so, as interleaving indexation and structure-building could not abstract from word order. Of course, the generalisation is that the head of the phrase, for instance a verb in a VP, is always the root (*PhrasePtr*).

N1 is attached as complement at the right frontier of T1. A new index is percolated to the subtree headed by the parent of N1, the node to which N1 has become complement. Since nodes are attached one by one, a new index is assigned to all the nodes rooted in a branching node, hence *indp* and *fp* hold for the *attach as complement* operation. Hence, Definition 9 holds, and consequently, Theorem 1 holds as well.



To calculate the complexity of this operation of indexation, we take into consideration the number of times that all nodes are visited by the algorithm. We observe that a new index is generated and percolated down at each attachment which gives rise to a branching node. Thus, the worst case arises if a complete binary tree is traversed, because it has the greatest number of internal nodes. The total number of internal nodes of a complete binary tree is $(n-1)/2$, for n leaves. Thus, the total number of updates is $(n-1)/2 \times (n-1)/2 = O(n^2)$ in the worst case.¹⁰

4.2 The Local Domain

The computation of coreference involves nodes in a tree for which the c-command relation holds in a certain domain. Antecedents of coreferring expressions are not explicitly identified. [Giorgi *et al.*, 1989] argue that this is the correct way of interpreting the purpose of a binding algorithm.

On the other hand, [Ingria and Stallard, 1989] claim that an algorithm that computes coreference must return as output both possible and impossible antecedents, to be useful for practical purposes. Following a procedural statement of binding theory [Chomsky, 1980], they present an algorithm where each nominal node in a tree is associated to two sets: the set of *internal* nodes to the local binding domain and the set of nodes *external* to it. The extension of the binding domain is computed statically: each NP with a subject and each IP define a new domain. The algorithm performs a preorder traversal of the tree and updates the two sets of possible antecedents.

This algorithm is empirically insufficient in two cases: case 7 above, where the NP with a subject is not the local domain, and case 3, where IP is not the local domain either.¹¹ Nonetheless, the idea of computing sets

¹⁰This measure is not affected by the fact that some nodes cover the empty string (which is a necessary property of these trees, since we are assuming GB theory). Each lexical token (i.e. a word in the string) can license at most k empty categories, for (k) very small, usually 3 or 4, which are in turn preterminal nodes, hence they do not license any other nodes. Thus, empty categories can increase the number of attachments by only a (small) constant factor. (See [Correa, 1991], who establishes this upper bound.)

¹¹[Ingria and Stallard, 1989] acknowledge case 7 and do not

of internal and external nodes is interesting, as it eliminates the need to traverse the tree upward each time an element that requires an antecedent is encountered (differently from [Giorgi *et al.*, 1989].)

We propose an extension of [Ingria and Stallard, 1989], which deals correctly with the problematic cases mentioned above. On observation of the actual linguistic instances, illustrated in section 2 (derived from Defs. 6-8), we notice that the current sentential node (IP) is a CFC in most cases. Thus it is the domain β in which Def. 6 applies, with the exception of cases 3 and 7. In the former case, IP does not count because it is an infinitival, in the latter case the NP, contained within the IP, is the local domain. This is possible only if the corefering expression is a pronoun, which can be not bound.¹² Computationally, the extent of the binding domain is, in part, a function of values which are known only when the input is seen (*i.e.* whether the verb is $\pm$ Tense or whether the NP is $\pm$ pronominal).

However, if the computation of the local domain, and the collection of possible/impossible antecedents is performed on-line, then an upward traversal of the tree is necessary. In order to avoid this traversal, we propose an "off-line" approach. In the spirit of [Ingria and Stallard, 1989], all the possible cases derivable from Defs. 7 and 8 are calculated, reaching a 3-way distinction. A number of global variables is set, which keeps track of all the potential local domains. For each of these local domains, the set of possible and impossible antecedents is calculated. The relevant local domain, with related antecedents, will be chosen when the actual referring expression is seen. Coreference can be established without traversing the tree.

Algorithm 2 *This algorithm applies at each attachment operation, if the node that is being attached is an NP.*

update-coreference

```
(* this procedure manipulates some global variables:
external-current-ip, internal-current-ip: nodes;
external-higher-ip, internal-higher-ip : nodes;
external-current-np, internal-current-np: nodes;
np-has-subject, np-has-no-subject : boolean;
*)
```

```
(* case 3 *)
```

```
if attachment=specifier and head=non-tensed-i then
```

```
(* local-domain<- higher-ip *)
```

```
if np = anaphor then
```

```
possible-antecedents <- internal-higher-ip
```

```
impossible-antecedents <- external-higher-ip
```

```
else
```

```
possible-antecedents <- external-higher-ip
```

```
impossible-antecedents <- internal-higher-ip
```

```
(* case 4,7 *)
```

```
else if
```

```
(attachment=complement and head=n and np-has-subject)
```

```
or (attachment = specifier and head = n ) then
```

```
treat case 3.
```

¹²This descends from Def.7: for pronouns there always is a BT-compatible indexation, because they can be free, *i.e.* not coindexed.

```
(* local-domain<- current-np *)
if np = anaphor then
possible-antecedents <- internal-current-np
impossible-antecedents <- external-current-np
else
possible-antecedents <- external-current-np
impossible-antecedents <- internal-current-np
```

```
(* all other cases *)
```

```
else
```

```
(* local-domain<- current-ip *)
```

```
if np = anaphor then
```

```
possible-antecedents <- internal-current-ip
```

```
impossible-antecedents <- external-current-ip
```

```
else
```

```
possible-antecedents <- external-current-ip
```

```
impossible-antecedents <- internal-current-ip
```

Finally, consider how the sets of internal and external nodes are updated.

Algorithm 3 *This algorithm applies at each attachment of NP.*

```
(* global pointers to local domains
```

```
current-ip, higher-ip, current-np: pointer; *)
```

```
For all the global pointers of local domain do
```

```
if ~pointer dominates new node then
```

```
add new node to set of internal nodes
```

```
else
```

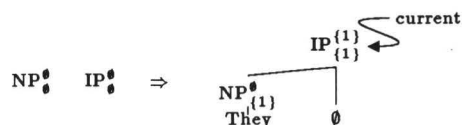
```
add new node to external nodes
```

On updating the pointers, the set of internal and external nodes are inherited by computing dominance.¹³

5 Example

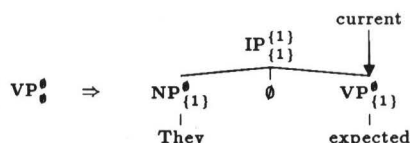
In this section we show how the percolation of the indices is done, and, for each step, what is the local domain defined by the algorithm. Sentence (5) is illustrated.

For each word in the input, and also for null strings postulated by the grammar, the parser projects a node. Clearly no attachment can occur until two nodes are projected. After an NP and an IP are projected, an attachment as specifier is performed. Due to the attachment, IP becomes a branching node, hence the index 1 is percolated, as shown. The local domain is the current IP, according to the third clause of algorithm 2. The NP is the only internal node, and there are no external nodes.

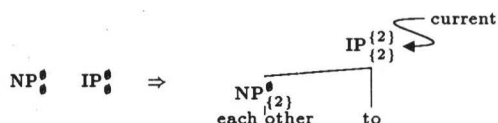


¹³Dominance is computed by two different mechanisms, which rely on the fact that nodes built by this parser are of two kinds: either projections of a head to its maximal level, for instance, a verb to a VP, or binary nodes, built by attachment. In the former case, a maximal projection is a single data structure, hence the computation of dominance is trivial. In the latter case, we observe that each node x of index y dominates all the nodes that contain y in their indexation set. Thus dominance can be computed by using the same indexation of nodes that encodes c-command.

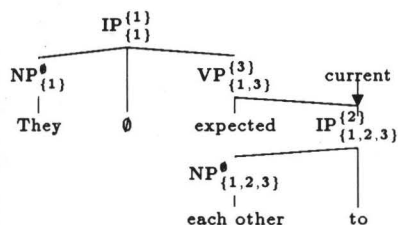
The verb *expected* is encountered, projected to a VP, and attached to the IP. No new indices are percolated, as IP was already a branching node. (This is computed by *branching flag*.) No changes occur to the local domain, as no NP nodes have been parsed. The partially built tree is shown below.



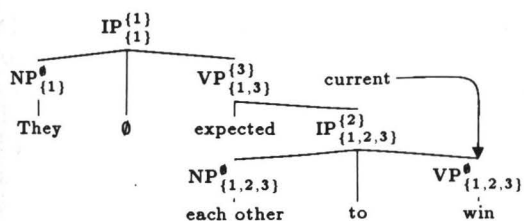
A new NP and a new inflectional, non-tensed node are parsed, projected to the maximal projection, and attached in a specifier configuration. Percolation of indices is performed as in the first step. This configuration is shown here.



The two-node configuration built in the preceding step is attached as complement, and indices are percolated, since now VP is a branching node. The value of the local domain is the higher IP. *They* is an internal node, and there are no external nodes. The indexed tree is shown below.



Finally, the verb *win* is projected and attached, no new indices are generated. Binding of *each other* is possible by the higher *they*, since principle A of the binding theory applies, and the binding is within the local domain, i.e. the anaphor is bound in the higher IP.



Conclusion In this paper we have illustrated two algorithms to compute c-command and binding. We have shown that they work incrementally and can handle a few cases that previous proposals could not treat. Future work is going to extend the algorithms beyond the core cases presented here, such as split antecedents and backward anaphora.

References

- [Abney, 1989] Abney Steven. A Computational Model of Human Parsing. *Journal of Psycholinguistic Research*, 18:129-144, 1989.
- [Barker and Pullum, 1990] Barker Chris and Geoffrey Pullum. A Theory of Command Relations. *Linguistics and Philosophy*, 13:1-34, 1990.
- [Chomsky, 1980] Chomsky Noam. On Binding. *Linguistic Inquiry*, 11(1):1-46, 1980.
- [Chomsky, 1986] Chomsky Noam. *Knowledge of Language: Its Nature, Origin and Use*. Praeger, New York, 1986.
- [Correa, 1991] Correa Nelson. Empty Categories, Chain Binding and Parsing. In Berwick, Abney and Tenny (eds.), *Principle-based Parsing*, Kluwer, Dordrecht, pp.83-122, 1991.
- [Fong, 1991] Fong Sandiway. *Computational Properties of Principle-based Grammatical Theories*. Ph.D Dissertation, MIT, Cambridge, MA, 1991.
- [Giorgi, 1987] Giorgi Alessandra. The Notion of Complete Functional Complex: Some Evidence from Italian. *Linguistic Inquiry*, 18(3):511-518, 1987.
- [Giorgi et al., 1989] Giorgi Alessandra, Fabio Pianesi and Giorgio Satta. Towards a Complexity Analysis of Binding Theory, TR 8911-06, IRST, Trento, 1989.
- [Giorgi et al., 1990] Giorgi Alessandra, Fabio Pianesi and Giorgio Satta. A Computational Approach to Binding Theory. *Proceedings of COLING*, pp. 120-125, Helsinki, 1990.
- [Ingria and Stallard, 1989] Ingria Robert and David Stallard. A Computational Mechanism for Pronominal Reference. In *Proceedings of the Association for Computational Linguistics*, pp.262-271, Vancouver, 1989.
- [Latecki, 1991] Latecki Longin. An Indexing Technique for Implementing Command Relations. In *Proceedings of the European Chapter of the Association for Computational Linguistics*, pp.51-56, Berlin, 1991.
- [Reinhart, 1983] Reinhart Tanya. *Anaphora and Semantic Interpretation*. University of Chicago Press, Chicago, 1991.
- [Wehrli, 1992] Eric Wehrli The Interactive Parsing System. *Proceedings of COLING*, pp.870-874, Nantes, 1992.