Contextual discontinuous grammars.Theoretical aspects and implementation

Patrick Saint-Dizier

INRIA

UNITÉ DE RECHERCHE
INRIA-RENNES

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
BP 105
78153 Le Chesnay Cedex
France

Tél. (1) 39 63 55 11

Rapports de Recherche

N° 611

CONTEXTUAL DISCONTINUOUS GRAMMARS

THEORETICAL ASPECTS AND IMPLEMENTATION

Patrick SAINT-DIZIER

Février 1987

IRISA

INSTITUT DE RECHERCHE EN INFORMATIQUE
ET SYSTÈMES ALÉATOIRES

Campus Universitaire de Beaulieu
Avenue du Général Leclerc
35042 - RENNES CÉDEX
FRANCE
Tél. : (99) 36.20.00
Télex : UNIRISA 95 0473 F

GRAMMAIRES CONTEXTUELLES DISCONTINUES : ASPECTS THEORIQUE ET IMPLEMENTATION

Patrick SAINT-DIZIER
IRISA
Campus de Beaulieu
35042 RENNES CEDEX

Dept. of Computing Science
SIMON FRASER University
BURNABY, B.C., V5A 156
CANADA

Publication Interne n° 332
Décembre 1986

12 pages

Résumé :

Nous présentons un formalisme basé sur la logique - les grammaires contextuelles discontinues - qui permettent de traiter des dépendances lointaines dans les langages naturels et formels. Par dessus ce formalisme, nous définissons deux schémas de métarègles pour exprimer les mouvements par substitution ou par adjonction dans le cadre de la théorie chromokienne de "Government and Binding". Plusieurs contraintes, telle la subjacente, sont intégrées au formalisme. Un traducteur de ce formalisme en PROLOG est décrit pour terminer.

.../...

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE
(L.A. 227)
UNIVERSITÉ DE RENNES 1 I.N.S.A. DE RENNES

INSTITUT NATIONAL DE RECHERCHE
EN INFORMATIQUE ET EN AUTOMATIQUE
(LABORATOIRE DE RENNES)

CONTEXTUAL DISCONTINUOUS GRAMMARS: Theoretical Aspects and Implementation.

Patrick SAINT-DIZIER
(INRIA-France)
Dept. of Computing Science
SIMON FRASER University
BURNABY, B.C. V5A 1S6
Canada

ABSTRACT

In this paper, we present a logic-based formalism - Contextual Discontinuous Grammars - that can deal with long distance dependencies. On top of this formalism, we define two metarule formats to express movements by substitution and adjunction within the framework of Government and Binding theory. Several constraints such as subjacency are also integrated into this system. Most aspects of Government and Binding theory are automatically integrated by a processor that transforms grammar rules into PROLOG clauses, making the system modular and easier to use than most current systems dealing with Government and Binding.

electronic mail: ... !ubc-vision!fornax!sfulccr!dizier

type of paper: Short Paper,

topic : Natural Language Understanding,

track : Science,

Keywords : syntactic analysis, logic-based grammars, Government and Binding theory, meta-rules, Prolog.

INTRODUCTION

Discontinuous Grammars (Dahl and Saint-Dizier 86) are a linguistically motivated tool for processing natural as well as formal languages. However, their type-0 form make them quite difficult to write, to control the power and to implement efficiently. We propose in this paper a more restricted class of discontinuous grammars we call Contextual Discontinuous Grammars (CDG for short). This restricted rule format seems to be well suited to express Government and Binding (Chomsky 84), (Dahl 86).

We restrict the form of discontinuous grammars to rules of the form:
a --> a1 / b --> b1.
This rule can be paraphrased by: *"a is rewritten into a1 iff a symbol b is rewritten into b1 somewhere else in the parsing tree with the same substitutions used. "*. In a more declarative way, we could say, in terms of node admissibility conditions in a tree, that a is an admissible node immediately dominating a1 iff, in the same tree, b is an admissible node immediately dominating b1.

This type of rule introduces a relation between two derivations: b --> b1, the right hand part of the relation is accessible to the parser only if a --> a1 has been executed. We call: / the accessibility relation.

A brief example is the grammar that recognizes strings of the language:
$L(G) = \{ a^n b^m c^n d^m, n, m \in N \}$.

s --> as, bs, cs, ds.
as --> [a], as / cs --> [c], cs.
as --> [a] / cs --> [c].
bs --> [b], bs / ds --> [d], ds.
bs --> [b] / ds --> [d].

1_ FORMAL ASPECTS OF THE ACCESSIBILITY RELATION

The general form of a CDG rule is A / B where A is of the form: a --> a1 and B : b --> b1, a,b $\in V_n$ and a1, b1 $\in V_n^*$. A and B are context-free rules, a CDG rule is a pair of context-free rules, the same set of substitutions being applied to both A and B.

1.1_ Free or restricted accessibility

The accessibility relation is *free* if B can be executed without any restriction as soon as A has been executed.

The accessibility relation is *restricted* if B can be executed as soon as A has been executed only if additional conditions are met. These conditions are, for instance, expressed in terms of derivations that may occur between A and B. For example, in Government and Binding, a condition is the subjacency constraint: A and B must not be separated by more than one bounding node.

1.2_ Modalities of accessibility

Accessibility can be viewed in two ways. After A is executed, there are two cases:
(a) B has to be executed once and only once; this introduces a operator of *necessity*, we note it: *A / L B.*
(b) B can be executed any number of times, this introduces the operator of *possibility*, noted: *A / M B.*
In the remainder of this paper, only modality M will be indicated.

1.3_ Accessibility structure

We consider a set GC of classical context-free rules (for instance, the generative component of Chomsky's linguistic description) and a set, noted CDGR, of CDG rules. Each CDG rule is composed of two parts: a left contextual rule (a --> a1) and a right contextual rule (b --> b1). Let LCR be the set of all the left contextual rules of CDGR, and RCR the set of all the right contextual rules.

An accessibility structure is a quadruple < GC, CDGR, W, P > where:
- GC is a set of context-free rules,
- CDGR is a set of contextual discontinuous rules,
- W is a non-empty set of sets w (the possible sets of applicable rules),
- P is a construction process of the elements of W.

W is constructed by P from GC and GDCR. Each time a rule is activated, the parsing state moves from state i to state i+1. To each state corresponds a set w_i which is the set of valid rules in this state. Each w_i is determined by P.

In our case, P is defined as follows:

(1) The starting element w_1 is equal to GC U LCR.
(2) Let w_i be the set of applicable rules in state i. Depending on the kind of rule activated in the step i, we have different results at step i+1 for w_{i+1}:

(a) A rule of GC is activated: $w_{i+1} = w_i$.

(b) A rule li of LCR is activated, li being the left hand part of a rule li / ri. Then: $w_{i+1} = w_i \cup \{ ri \}$.

(c) A rule rj of RCR is activated. Depending on the modality associated to this rule, we have two cases:
(i) rj is of the form L(b --> b1) then $w_{i+1} = w_i - \{ rj \}$.
(ii) if rj is of the form M(b --> b1) then $w_{i+1} = w_i$.

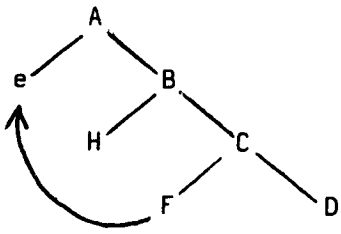
2_ SUBSTITUTIONS AND ADJUNCTIONS

In the theory of movement of Government and Binding, there is a single type of rule called move-α. This universal movement meta-rule says simply: *"Move any category α anywhere, α being a category variable. "*
Several general principles which act as filters are associated to this very general rule. This approach results in a greater economy in number of rules and in a higher level of generality, modularity and explanatory power.

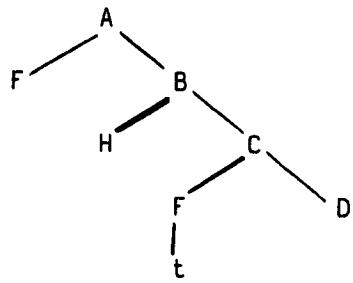
Move-α subsumes two types of movements: substitution and adjunction. We now define two types of rules to express substitution and adjunction and show how they can be translated into the CDG rule format.

2.1_ Substitutions

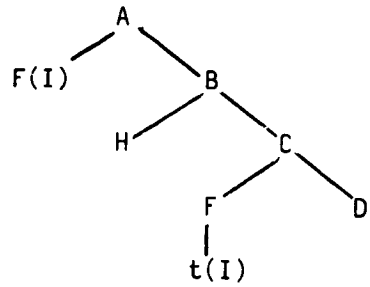
Substitution, roughly speaking, is an operation that replaces a node by a subtree. The replaced node in Government and Binding is an empty node. In a tree like:



e can be substituted for F, F leaving a trace t in its original position:



In order to establish a link between a moved constituent and its original position, the trace principle states that a moved constituent leaves behind a co-indexed empty node of itself. The above tree becomes:



Substitutions are restricted by two principles:
(a) *The Structure Preserving Principle:* a category can only be substituted for a category of the same type,
(b) *The Empty Node Principle:* a moved constituent can only be substituted for a category with an empty realization.
These two constraints are directly expressed by the way CDG rules are written, i.e. which symbols are substituted for others.

A substitution can be expressed by a meta-rule of the form:

e subs F >> F --> t. for a substitution with a movement to the left and

F --> t >> e subs F. for a substitution with a movement to the right.

The first meta-rule means that if F is substituted for e then the derivation F --> t is expected in the future. Meta-rules also have modalities L or M.

This substitution meta-rule originates the production of CDG rules. Let GC be the generative component of the grammar. Let a substitution meta-rule be of the form:
E subs A >> C --> D.

then $\forall$ r $\in$ GC, r: α --> β_1 β_2 .... β_n , with α, β_i $\in$ V_n, if $\exists$ i $\in$ [1,n], such that β_i unifies with E, then the following CDG rule is produced:

α --> β_1, β_2 ... β_{i-1}, A(I), β_{i+1}, ... ,β_n / C --> D(I).

An extra-argument, I, is automatically added to express co-indexation. To enforce the structure preserving principle, E has a category feature.

All meta-rules are independent from each other and can be executed in any order. Thus, meta-rules are applied only once on GC and the transitive closure of the system is straightforward. A quite small number of CDG rules is produced by each meta-rule.

If we follow the last developments of X-bar syntax, GC is composed of rules with at most 2 symbols in the right hand part, which results in a much simpler translation system.

In PROLOG, CDG rules are produced from substitution meta-rules by declarative clauses such as:

```
substitution((A subs B >> C --> D), (G --> A, S),
             ((G --> B coind I, S) / (C --> D coind I))).
```

Where, in substitution(MR,R,C), MR stands for a meta-rule, R is an element of GC and C is the resulting CDG rule. The symbols >>, subs, coind and / are operators. Co-indexation is automatically introduced by the PROLOG processor into the CDG rule, without the need of an explicit specification by the grammar writer.

An example is the definition of a relative clause, where the following metarule:
metarule((rel_marqueur(X) subs pronom_rel >> np(obj,X) --> trace(X))).

together with the rule of GC:

relative(X) --> rel_marker(X), sentence.

originate the production of the following CDG rule:

```
relative(X) --> pronom_rel coind I, sentence /
                    np(obj,X) --> trace(X) coind I.
```

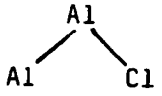
The following meta-rules are used to build direct interrogative forms:

metarule((int_marker(X) subs sn(pro,X) >> sn(object,X) --> trace(X))).

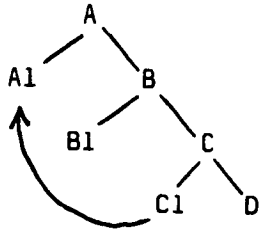
metarule((int_marker(X) subs pp(pro,X,Pronoun) >> pp(explicit,X,Pronoun) --> trace(X))).

2.2_ Adjunction

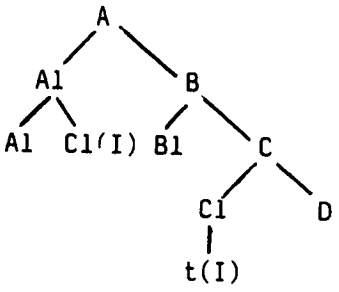
We consider here Chomsky-adjunction. To adjoin C1 to A1, entails the production of a tree of the form:



Consider the following tree:



After adjunction of C1 to A1, the tree becomes:



C1 in its new position is co-indexed with its trace by I. Adjunction is expressed by a meta-rule of the form:
A1 adj C1 >> C1 --> t. for an adjunction with a movement to the left and
A1 --> t >> C1 adj A1. for an adjunction with a movement to the right.
The first meta-rule means that if C1 is adjoined to A1, then the derivation C1 --> t is expected in the future. Adjunction originates the production of a CDG rule. For instance, the first adjunction rule above becomes:

A1 --> A1, C1(I) / C1 --> t(I).

The transformation of an adjunction rule into a CDG rule, including co-indexation, is straightforward. In PROLOG, we have the following clauses, respectively for left and right adjunction:

```
adjunction((A1 adj C1 >> (C1 --> T)),
          ((A1 --> A1, C1 coind I) / (C1 --> T coind I))).
adjunction(((A1 --> T) >> C1 adj A1),
          ((A1 --> T coind I) / (C1 --> C1, A1 coind I))).
```

The first argument of the adjunction clause represents the adjunction meta-rule and the second, the resulting CDG rule. Adjunction is used, for instance, to describe right extraposition of a relative clause as in:
"The man is here that John met. "
by the meta-rule:

```
metarule(((relative(X) --> empty) >> vp(Y) adj relative(X))).
```

adj is an operator.

3_ A PROLOG IMPLEMENTATION OF CDG RULES FOR GOVERNMENT AND BINDING

We have implemented a grammar processor in PROLOG that transforms rules given by a grammar writer into Prolog clauses. Rules given by the grammar writer have a very simple syntax: only a few syntactic features have to be specified. The remaining arguments, representing for instance the input and output list, control lists, subjacency, are automatically added by the processor. The processor is composed of several modules, each one corresponding to a specific task: meta-rule interpretation, subjacency, etc... Depending on the complexity of the sublanguage considered, some modules may not be taken into account by the processor for the translation.

Rules of GC and CDG are translated into PROLOG using additional arguments, in the same spirit than Extraposition Grammars. However, we do not use Extraposition Grammars as in (Stabler 86) to translate our formalism. Two arguments are used to deal with the input and the output string and two other arguments are used to respectively represent the input list of right parts of CDG rules (RCR) which are valid and the output list of valid rules after execution of the clause. These two lists are called the input and the output RCR valid rule lists.

A rule of GC of the form: a --> b, c. becomes:

```
a(X,Y,LE,LS) :-  b(X,Z,LE,LS1),
                 c(Z,Y,LS1,LS).
```

For CDG rules, the left hand part (a --> a1) originates the adjunction of an element to the output RCR valid rule list which corresponds to a new valid rule. The right hand part of a CDG rule (b --> b1) translation begins with a control of the applicability of the rule and, possibly, with a modification of the output RCR valid rule list. Each element of the RCR valid rule list is a structure of the form:
d(< list of functors of the terms of the rule>).
Thus,for instance, cs --> c, cs originates the construction of the element:
d(cs,c,cs).
Finally, the rule:

as --> a, as / L(cs --> c, cs).
is translated into the following PROLOG clauses:

```
as(X,Y,LE,[d(cs,c,cs)|LS2]) :-
   a(X,Z,LE,LS1),
   as(Z,Y,LS1,LS2).
cs(X,Y,LE,LS) :-
   check_withdraw(d(cs,c,cs),LE,LE1),
   c(X,Y,LE1,LS1),
   cs(Z,Y,LS1,LS).
```

The PROLOG call check_withdraw(E,L1,L2) checks whether E is an element of the RCR valid rule list L1. If the answer is positive, then the rule is applicable and E is withdrown from L1 which becomes L2. If the right hand part of the CDG rule had the modality M instead of L, then, only check(E,L1) is used that checks whether E is in L1.

Co-indexation is also expressed via the RCR valid rule list. For that purpose, an additional argument is added to each element of the list which is the co-indexation variable. This variable is instanciated to an integer number during the parsing process and it automatically percolates in the parsing tree, via unification. Thus, co-indexation in our system is straightforward and never ambiguous. Chains of coindexation can also be generated when a constituent is moved several times.

Finally, the subjacency condition expresses constraints on the accessibility of the right hand part of a CDG rule: in the parsing tree, a co-routine controls that the derivation corresponding to the left hand part of a CDG rule is not be separated by more than one bounding node from the derivation of the right hand part of this CDG rule.

DISCUSSION

CDG formalism appears to be well suited to express principles of Government and Binding theory. One of the main advantages of this formalism is that it reduces the type-0 power of Discontinuous Grammars to pairs of context-free rules. Principles of Government and Binding theory are expressed and implemented in a modular fashion: co-indexation, subjacency. Government with barriers which is still under study will also originate a separate module. Representation and evolution of features, including possible modifications of features for moved constituents (for instance in French), are presented in (Saint-Dizier 86).

A processor transforms grammar rules into PROLOG clauses. It also automatically int grates in the same time most principles of Government and Binding theory, making the original Government and Binding rules much easier to write and to read. Substitution and adjunction meta-rules originate the production of a small number of rules, without the need of extra-symbols.

This system is now fully implemented and has been tested on a quite comprehensive subset of French (including relativization, passivization, coordination and construction of completive and interrogative forms). The

overall efficiency is comparable to that of extraposition grammars (Pereira 81).

This formalism can be generalized to tuples of context-free rules (this idea was suggested to us by V. Dahl) and can also be used in parallel architectures to express dependencies or synchronizations. Research about formal properties of adjunction and substitution metarules is under study and would probably fulfill some requirements stated in (Uszkoreit and Peters, 86).

ACKNOWLEDGEMENTS We are very gratefull to Dr. Veronica DAHL for insightfull comments on a draft version of this work.

R E F E R E N C E S

Chomsky, N., *Lectures on Government and Binding*, Foris Pub. Dordrecht, 1984.

Dahl, V., Saint-Dizier, P., *Constrained Discontinuous Grammars: a linguistically motivated tool for processing language*, under submission, 1986.

Dahl, V., *Gramaticas Discontinuas, una hierramenta computacional con applicaciones en la teoria de Gobierna y Nexo*, Revista Argentina de Linguistica, 1986.

Pereira, F., *Extraposition Grammars* Computational Linguistics, Vol. 7, 1981.

Saint-Dizier, P., *Expression of features in logic-based grammars*, Computational Intelligence, Vol. 2 no 1, 1986.

Stabler, E., *Restricting Logic Grammars*, Quintus Computer systems report, 1986.

Uszkoreit, H., Peters, S., *On some Formal Properties of Metarules*, Linguistics and Philosophy, pp. 477-494, 1986.

Imprimé en France

par

l'Institut National de Recherche en Informatique et en Automatique