Unit Coordination and Gapping in Dependency Theory

Vincenzo Lombardo and Leonardo Lesmo
Dipartimento di Informatica and Centro di Scienza Cognitiva
Universita' di Torino
c.so Svizzera 185 - 10149 Torino - Italy
{vincenzo, lesmo}@di.unito.it

Abstract

The paper investigates the problem of representing coordination constructs in a formal system for the dependency approach to syntax. The distinctive aspect of the formalism is the presence of non-lexical (trace - ε) nodes in the representation. We illustrate the basic mechanisms, and how they can account for syntactic phenomena involving long-distance dependencies. Then, we see how the same mechanism can account for the gaps and ellipses of coordination constructs.

1. Introduction

Coordination is a long standing problem for linguistic theories, because of its particular aspects which do not fit well with the dominance-based character of the vast majority of paradigms. The accounts of coordination which have been devised in constituency theories all rely on the notion of non traditional constituents, because of the variety of word strings that can play the role of conjuncts (see, for example, (Gazdar et al. 1985)). Maybe the most natural description has been provided by Steedman in a series of papers on Combinatory Categorial Grammar (Steedman 1985, 1990, 1996). He generalizes the notion of constituent through the operators of functional composition and type raising, the so-called *combinators*. This uniformity of analysis suffers from the generation of spurious ambiguities, even if this proliferation of structures seems to be justified (in some cases[1]) by the requirement of characterizing intonation patterns. Sarkar and Joshi (1997) forward CCG analysis in the LTAG framework, by interpreting combinators in terms of argument sharing and traces handled by TAG operators (*conjoin* or *adjunction* itself). In this way, they are able to keep unaltered the notion of fixed constituency due to LTAG lexicon.

Dependency paradigms exhibit obvious difficulties with coordination because, differently from most linguistic structures, it is not possible to characterize the coordination construct with a general schema involving a head and some modifiers of it. The conjunction itself, has distributional properties that have nothing to do with the whole coordination. Hudson (1990, following Tesniere 1959) gives up the idea of providing a dependency structure for the coordination, and characterizes conjuncts as word strings. Conjuncts are internally organized as (possibly disconnected) dependency structures and each conjunct root is dependency related to some element of the sentence which is external to the coordination.

Mel'cuk (1988), on the other hand, privileges one of the two conjuncts as the head of the coordination, and claims that coordination symmetry is such only at the semantic level. This approach solves the problem of providing a head with the same distributional properties of the whole coordination. When applied to English, it allows to represent the prepositional character of conjunctions observed by Ross (1970).

In this work we provide a dependency account of coordination following Mel'cuk's hint: one of the two conjuncts is the head of the construction, and the conjunction itself is the head of the other conjunct. We extend the lexicalized dependency formalism presented in (Lombardo, Lesmo 1998): each rule constrains the form and the number of dependents of a word in a dependency tree (see fig. 1).

The formalism is projective, and accounts for long-distance dependency phenomena

[1] For example, it is hard to analyze "Keats steals apples" as both (Keats (steals apples)) and ((Keats steals) apples) (Sarkar, Joshi 1997). This is likely to be a spurious ambiguity.

through the introduction of non lexical categories, which can be viewed as a notational variant of some graph structures already present in some formalisms (see, e.g., Word Grammar (Hudson 1990)). The presence of non lexical categories keeps unaltered the condition of projectivity, encoded in a notion of derivation. The core of the grammar consists of *primitive* and *non primitive* dependency rules, representing predicate-argument structures associated with lexical items. Non primitive dependency rules result from the application of *metarules* to primitive rules. The dependency rules that license coordination are non primitive rules for the head conjunct. They can be distinguished in rules for *unit* coordination, where conjuncts are complete predicate-argument structures, and *gapped* coordinations, where conjuncts have gaps inside.

The paper is organized as follows. The next section presents the basic dependency system, along with the notion of derivation. Section 3 presents the treatment of coordination, together with examples of metarules that account for several forms of unit and gapped constructions. Section 4 concludes the paper.

2. A dependency formalism

The basic idea of dependency is that the syntactic structure of a sentence is described in terms of binary relations (*dependency relations*) on pairs of words, a *head* (or parent), and a *dependent* (daughter), respectively; these relations form a tree, the *dependency tree*. In this section we describe the dependency system in (Lombardo, Lesmo 1998). The syntactic knowledge is expressed by dependency rules which describe a node of and the parse structures generated by a grammar of this form are defined through a derivation procedure.

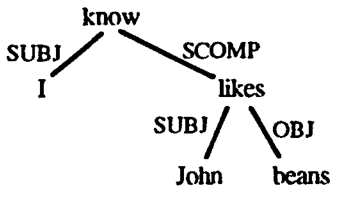


Figure 1. A dependency tree for the sentence "I know John likes beans". The leftward or rightward orientation of the edges represents the order constraints: the dependents that precede (respectively, follow) the head stand on its left (resp. right).

The grammar and the lexicon coincide, since the rules are lexicalized: the head of the rule is a word of a certain category, i.e. the lexical anchor, and the dependents are categories linked by edges labelled with dependency relations. From the linguistic point of view we can recognize two types of dependency rules: *primitive* dependency rules, which represent linearly-ordered predicate-argument structures (possibly with adjuncts), and *non-primitive* dependency rules, which describe syntactic phenomena like argument sharings and extractions. *Metarules* (see section 3), which produce non primitive rules from primitive rules (by means of one or several applications), obey linguistic principles (cf. GPSG approach (Gazdar et al. 1985)).

A *dependency grammar* is a six-tuple <W, C, S, D, I, H>, where
W is a finite set of words;
C is a set of syntactic categories;
S is a non-empty set of root categories ($C \supseteq S$);
D is the set of *dependency relations* (e.g. SUBJ, OBJ, XCOMP, P-OBJ, PRED[2]);
I is a finite set of symbols (among which the special symbol ◊), called *u-indices*;
H is a set of *dependency rules* of the form

$$x{:}X\ (<r_1 Y_1 u_1 \tau_1> \dots <r_{i-1} Y_{i-1} u_{i-1} \tau_{i-1}>\ \#\ <r_{i+1} Y_{i+1} u_{i+1} \tau_{i+1}> \dots <r_m Y_m u_m \tau_m>)$$

1) $x \in W$, is the *head* of the rule;
2) $X \in C$, is its syntactic category;
3) an element $<r_j\ Y_j\ u_j\ \tau_j>$ is a *d-quadruple* (which describes a dependent); the sequence of d-quads, including the special symbol # (representing the linear position of the head), is called the *d-quad sequence*. We have that
3a) $r_j \in D, j \in \{1, ..., i\text{-}1, i+1, ..., m\}$;
3b) $Y_j \in C, j \in \{1, ..., i\text{-}1, i+1, ..., m\}$;
3c) $u_j \in I, j \in \{1, ..., i\text{-}1, i+1, ..., m\}$;
3d) τ_j is a (possibly empty) set of triples <u, r, Y>, called *u-triples*, where $u \in I$, $r \in D$, $Y \in C$.

Principle of *u-triple satisfiability*:
For each dependency rule $\delta \in H$ and $u \in I$, there

[2] The special relation VISITOR (Hudson 1990) accounts for displaced elements and, differently from the other relations, is not semantically interpreted.

exists a u-triple $<u_j, r, Y> \in \tau_j$ in a d-quad $<r_j Y_j u_j \tau_j>$ of δ iff there exists one d-quad $<r_i Y_i u_i \tau_i>$ of δ such that $u_j = u_i$, $i \neq j$.

Intuitively, a dependency rule constrains one node (head) and its dependents in a dependency tree[3]: the d-quad sequence states the order of elements, both the head (# position) and the dependents (d-quads). The grammar is lexicalized, because each dependency rule has a lexical anchor in its head (x:X). A d-quad $<r_i Y_i u_i \tau_i>$ identifies a dependent of category Y_i, connected with the head via a dependency relation r_i. Each element of the d-quad sequence is possibly associated with a u-index (u_i) and a set of u-triples (τ_i). The τ-component can be viewed as a slash feature in GPSG terms. Both u_i and τ_i can be null elements, i.e. $\Diamond$ and $\emptyset$, respectively. A u-triple (τ-component of the d-quad) $<u, R, Y>$ bounds the area of the dependency tree where the trace can occur. The principle of u-triple satisfiability states a one-to-one correspondence between the u-indices and the u-triples of the d-quads. The dependent indexed by u_k in the rule is coindexed with a trace node in the subtree rooted by the dependent containing the u-triple $<u_k, R, Y>$.

The derivation procedure requires some preprocessing (u-index annotation) on the dependency rules and some terminology.

As one dependency rule can be used more than once in a derivation process, it is necessary to annotate the u-indices with unique symbols (progressive integers) before the actual use. The annotation must be consistent in the u and the τ components. A dependency rule (as well as a u-triple) with u-indices all annotated is said to be *instantiated*: I^+ refers to the set of annotated u-indices (including the special symbol $\Diamond$).

A word w ($\in$ W) or the trace symbol ε ($\notin$ W) and two annotated u-indices μ and ν is a *word object* of the grammar. Given a grammar G, the set of word objects of G is

$$W_x(G)=\{{}_{\mu}x_{\nu} \,/\, \mu,\nu \in I^+, x \in W \cup \{\varepsilon\}\}.$$

A pair consisting of a category X ($\in$ C) and a string of instantiated u-triples γ is a *category object* of the grammar ($X(\gamma)$).

A 4-tuple $<r, Y(\gamma_1), u, \gamma_2>$ consisting of a dependency relation r ($\in$ D), a category object $X(\gamma_1)$, an annotated u-index u, a set of instantiated u-triples γ_2 is a *derivation object* of the grammar. Given a grammar G, the set of derivation objects of G is

$$C_X(G) = \{<r, Y(\gamma_1), u, \gamma_2> :$$
$$r \in D, Y \in C, u \in I^+, \gamma_1, \gamma_2 \text{ are sequences of instantiated u-triples}\}.$$

Given a grammar G, L'(G) is the language of sequences of word objects:

$$L'(G)=\{\alpha \in W_x(G)^* \,/$$
$$<TOP, Q(\emptyset), \Diamond, \emptyset> \Rightarrow^* \alpha \text{ and } Q \in S(G)\}$$

where TOP is a dummy dependency relation for the root of the tree and $\Rightarrow$ is the derivation relation (defined below). The language generated by the grammar G, L(G), is defined through the function t:

$$L(G)=\{w \in W_x(G)^* \,/\, w=t(\alpha) \text{ and } \alpha \in L'(G)\},$$

where t is defined recursively as

$$t(-) = -,\ t({}_{\mu}w_{\nu}\ \alpha) = w\, t(\alpha),\ t({}_{\mu}\varepsilon_{\nu}\ \alpha) = t(\alpha).$$

and - is the empty sequence.

The derivation procedure performs a leftmost generation of a sentence of L'(G). The derivation relation consists of two expressions. Let $\alpha \in (W_x(G))^*$ and $\psi \in (W_x(G) \cup C_X(G))^*$.

$$\begin{array}{ll}
\text{DR1:}\ \alpha <r, X(\gamma_p), u, \gamma_x>\ \psi \Rightarrow & \\
 & \alpha \\
 & <r_1, Y_1(\rho_1), u_1, \tau_1> \\
 & <r_2, Y_2(\rho_2), u_2, \tau_2> \\
 & \ldots \\
 & <r_{i-1}, Y_{i-1}(\rho_{i-1}), u_{i-1}, \tau_{i-1}> \\
 & {}_{u}x_{\Diamond} \\
 & <r_{i+1}, Y_{i+1}(\rho_{i+1}), u_{i+1}, \tau_{i+1}> \\
 & \ldots \\
 & <r_m, Y_m(\rho_m), u_m, \tau_m> \\
 & \psi
\end{array}$$

where $x:X\ (<r_1 Y_1 u_1 \tau_1> \ldots <r_{i-1} Y_{i-1} u_{i-1} \tau_{i-1}>\ \#\ <r_{i+1} Y_{i+1} u_{i+1} \tau_{i+1}> \ldots <r_m Y_m u_m \tau_m>)$ is a dependency rule, and $\rho_1 \cup \ldots \cup \rho_m = \gamma_p \cup \gamma_x$.

[3] As stated above, the dependency rules express the subcategorization frames, and possibly include adjuncts. Here are two examples of dependency rules:

yawned:V(<SUBJ, N, $\Diamond$, $\emptyset$> #),

is the licensing rule of "John yawned";

yawned:V(<SUBJ, N,$\Diamond$, $\emptyset$> # <MANNER, ADV,$\Diamond$, $\emptyset$>)

is the licensing rule of "John yawned loudly". In the first case the dependency rule represents the subcategorization frame; in the second case the rule includes an adjunct.

DR2: $\alpha\ \langle r, X(\langle v, r, X\rangle), u, ()\rangle\ \psi \Rightarrow \alpha\ {}_{u}\varepsilon_{v}\ \psi$

$\Rightarrow^*$ is the reflexive, transitive closure of $\Rightarrow$.

DR1 distributes over the dependents of a dependency rule the u-triples still to be satisfied (γ_p) plus the new u-indices introduced by the rule itself (γ_x). The u-index u, occurring in the current derivation object, is associated (as a μ-index) with the lexical head of the rule x. The whole subtree headed by x will be co-indexed with a trace node which will satisfy a u-triple of u in one of the ρ sets (DR2). If u=◊ no coindexing occurs. DR2 implements the u-triple satisfaction: the u-triple <v, r, X> represents an expectation for a trace of category X attached to its head through a relation r. When DR2 applies, the trace (word object ${}_{\mu}\varepsilon_{v}$) is in fact inserted in the derivation with v-index v. Notice that the trace has u (the u-index of the derivation object) as μ-index: this means that the trace itself can be the reference for another trace.

As an example, consider the grammar
G_1= <
  W_1: {I, John, beans, know, likes}
  C_1: {V, V+EX, N}
  S_1: {V, V+EX}
  D_1: {SUBJ, OBJ, VISITOR, TOP}
  I_1: {◊, u}
  T_1>,
where T_1 includes the following dependency rules:

1. I: N (#);
2. John: N (#);
3. beans: N (#);
4. likes: V (<SUBJ, N, ◊, ∅> # <OBJ, N, ◊, ∅>);
5. know: V+EX (<VISITOR, N, u, ∅>
   <SUBJ, N, ◊, ∅>
   #
   <SCOMP, V, ◊, {<u,OBJ,N>}>).

A derivation for the sentence "Beans I know John likes" is the following (at each step, the leftmost derivation object is underlined, and the derivation relation to be applied marks the symbol $\Rightarrow$):

$\underline{\langle \text{TOP}, \text{V}_{+\text{EX}}(\emptyset), \Diamond, \emptyset\rangle} \Rightarrow^{\text{DR1}}$
$\underline{\langle \text{VISITOR}, \text{N}(\emptyset), \text{u1}, \emptyset\rangle}\ \langle \text{SUBJ}, \text{N}(\emptyset), \Diamond, \emptyset\rangle$ know
  $\langle \text{SCOMP}, \text{V}(\emptyset), \Diamond, \{\langle \text{u1}, \text{OBJ}, \text{N}\rangle\}\rangle \Rightarrow^{\text{DR1}}$
${}_{\text{u1}}$beans $\underline{\langle \text{SUBJ}, \text{N}(\emptyset), \Diamond, \emptyset\rangle}$ know
  $\langle \text{SCOMP}, \text{V}(\emptyset), \Diamond, \{\langle \text{u1}, \text{OBJ}, \text{N}\rangle\}\rangle \Rightarrow^{\text{DR1}}$
${}_{\text{u1}}$beans I know $\underline{\langle \text{SCOMP}, \text{V}(\emptyset), \Diamond, \{\langle \text{u1}, \text{OBJ}, \text{N}\rangle\}\rangle}$
$\Rightarrow^{\text{DR1}}$
${}_{\text{u1}}$beans I know $\underline{\langle \text{SUBJ}, \text{N}(\emptyset), \Diamond, \emptyset\rangle}$ likes
  $\langle \text{OBJ}, \text{N}(\langle \text{u1}, \text{OBJ}, \text{N}\rangle), \Diamond, \emptyset\rangle \Rightarrow^{\text{DR1}}$
${}_{\text{u1}}$beans I know John likes
  $\underline{\langle \text{OBJ}, \text{N}(\langle \text{u1}, \text{OBJ}, \text{N}\rangle), \Diamond, \emptyset\rangle} \Rightarrow^{\text{DR2}}$
${}_{\text{u1}}$**beans I know John likes** ε_{u1}

The dependency tree corresponding to this derivation is in fig. 2.

Given DR1 and DR2, a trace node represents a gap for the whole subtree headed by the lexical node coindexed with it. So, in fig. 2, if we replace "beans" with a complex noun phrase ("black beans that come from Guatemala"), u_1 again annotates only "beans", but the trace node represents the gap for the whole subtree.

3. Coordination in English

In section 2 we have described a lexicalized dependency formalism and a notion of derivation that yields the legal sentences and the corresponding dependency trees. In (Lombardo, Lesmo 1998) we have defined an Earley-type parsing algorithm for this formalism. In this section we address the problem of providing a dependency analysis of coordination constructs. The (non primitive) dependency rules that account for coordination are the result of the application of metarules to basic dependency rules. Metarules and primitive dependency rules express the linguistic knowledge. The iterated application of metarules produces all the non primitive dependency rules, that together with the primitive rules, form the grammar. The application of the derivation rules to the complete set of rules produces the sentences of the language. This section addresses the linguistic issue of providing a number of

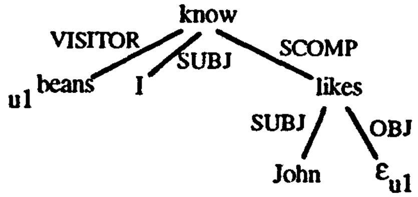


Figure 2. Dependency tree of the sentence "Beans I know John likes", given the grammar G_1.

metarules that deal with coordination, and the formal issue of extending the derivation procedure to gapped conjuncts. First, we introduce the general notation for metarules; then, we consider a number of coordination phenomena in English, starting from the simple to the complex ones, which require some extensions to the derivation rules.

3.1 Metarules

The general schema of a metarule[4] is

$$\text{SOURCE} \overset{\text{meta-id}}{\Longrightarrow} \text{TARGET}$$

where "meta-id" is an identifier of the metarule, and SOURCE and TARGET are PATTERNS of dependency rules. A PATTERN is an abstraction over a dependency rule, where the head can (possibly) reduce to the syntactic category (from x:X to X), and some subsequences of d-quads can be (possibly) replaced by some variable symbol. The abstraction allows to generalize the description of phenomena, where possible.[5]

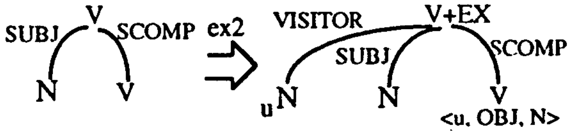
An example of metarule for object extraction is the following:

$$V\,(\langle \text{SUBJ}, N, \diamond, \emptyset\rangle \,\#\, \langle \text{SCOMP}, V, \diamond, \emptyset\rangle)$$

$$\longrightarrow^{ex2}$$

$$V_{+EX}\ \begin{array}{l}(\ \langle \text{VISITOR}, N, u, \emptyset\rangle \\ \ \ \langle \text{SUBJ}, N, \diamond, \emptyset\rangle \\ \ \ \# \\ \ \ \langle \text{SCOMP}, V, \diamond, \{\langle u, \text{OBJ}, N\rangle\}\rangle\ )\end{array}$$

which can be represented graphically as



The metarule ex2 takes as SOURCE a PATTERN that represents the subcategorization frame of a verb with a sentential complement (*know*, *believe*, *admit*, ...), and produces as TARGET the corresponding PATTERN that accounts for extracting the object from the sentential complement. In the previous section, we have seen how a dependency rule abstracted by this PATTERN can derive "Beans, I know John likes".

Some subsequences of d-quads do not affect the application of metarules, and pass unaltered from the SOURCE to the TARGET. PATTERNs avoid redundancy by introducing variables. The following metarule accounts for *preposition stranding* (for example, "This place, he comes to", or "This place, he often comes to with reluctance"):

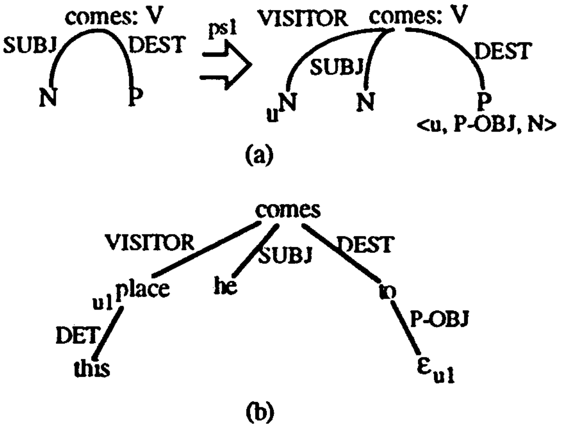
$$V\,(\langle \text{SUBJ}, N, \diamond, \emptyset\rangle \,\#\, \sigma_1 \,\langle p, P, \diamond, \emptyset\rangle\, \sigma_2)$$

$$\longrightarrow^{ps1}$$

$$V_{+PS}\ \begin{array}{l}(\ \langle \text{VISITOR}, N, u, \emptyset\rangle \\ \ \ \langle \text{SUBJ}, N, \diamond, \emptyset\rangle \\ \ \ \# \\ \ \ \sigma_1 \\ \ \ \langle p, P, \diamond, \{\langle u, \text{P-OBJ}, N\rangle\}\rangle \\ \ \ \sigma_2\ )\end{array}$$

A unification procedure accounts for matching PATTERNs (SOURCE and TARGET) and dependency rules, taking care of variable sustitutions from SOURCE to TARGET. The SOURCE dependency rule for "comes" in fig. 3a represents its subcategorization frame. The metarule ps1 matches this dependency rule through its SOURCE PATTERN, and produces the TARGET dependency rule, which licenses the dependency tree in fig. 3b.

In the remainder of this section, we describe the metarules that apply to dependency rules to generate the coordination constructs.



(a)

(b)

Figure 3. Metarule ps1 for preposition stranding (a), and dependency tree of the sentence "This place, he comes to" (b).

[4] This section presents an intuitive sketch of metarules, without any claim about the full generality of the examples of metarules reported herein.

[5] Recently, there has been a trend in linguistics to view most syntactic phenomena as lexicon-dependent rather than category-dependent (sometimes, e.g. passivization, the shift even continues to sentence-dependency). In our work, the final goal is to associate metarules to subcategories arranged in a hierarchy (on this topic, see (Barbero et al., this volume)). In this paper, we neglect this point, and all our examples of metarules assume category-headed patterns.

In the derivation process, we assume the grammar to be fully instantiated, that is all the metarules have been applied to generate all the non primitive dependency rules.

3.2 Metarules for coordination

The treatment of coordination relies on the application of proper metarules to dependency rules. Gapped constituent coordination results from the same mechanism that accounts for long-distance dependencies in section 2. The treatment of coordination illustrated here pursues the goal of providing a dominance-based representation. As we will see, this can lead sometimes to less standard analyses of coordination phenomena. Most of the theories in the literature assume a parallel constituency-based (Steedman 1996) (Sarkar, Joshi 1997) or string-based (Hudson 1990) analyses of the conjuncts. This prevents a fully incremental processing of the input, since before processing the infix conjunction (*and, or, ...*) we need to analyze the second conjunct. The theory illustrated here naturally favors the incremental analysis.

As the formalism is lexicalized, coordinating conjunctions are treated as lexically ambiguous elements (polymorphic functors in categorial terms): they are assigned the categories of the form CONJ-X, for each syntactic category X. So, we have CONJ-V for verbs, CONJ-N for nouns, and so on.

The first metarule we take into account is the one that works for *unit coordination* (coord-unit)[6]:

$X(\sigma) \rightarrow^{\text{coord-unit}} X(\sigma <\text{COORD, CONJ-X}, \Diamond, \emptyset>)$

For each dependency rule with a head of category X (variable), coord-unit produces a dependency rule having exactly the same d-quad sequence (σ), but with the added rightmost dependent CONJ-X, whose relation with the head is COORD (fig. 4a). The dependency rule

and: CONJ-V (# <2nd, V, $\Diamond$, $\emptyset$>)

licenses the second conjunct of a coordination of finite verbs. The relation "2nd" links the second conjunct as a dependent of the conjunction *and.* The TARGET dependency rule in fig. 4b results from the application of the metarule "coord-unit" to the primitive dependency rule for the predicate-argument structure of *laughed* (the same for *sneezed* and other intransitive verbs). According to these dependency rules, the sentence "John laughed and Mary sneezed" can be represented as shown in fig. 4c.

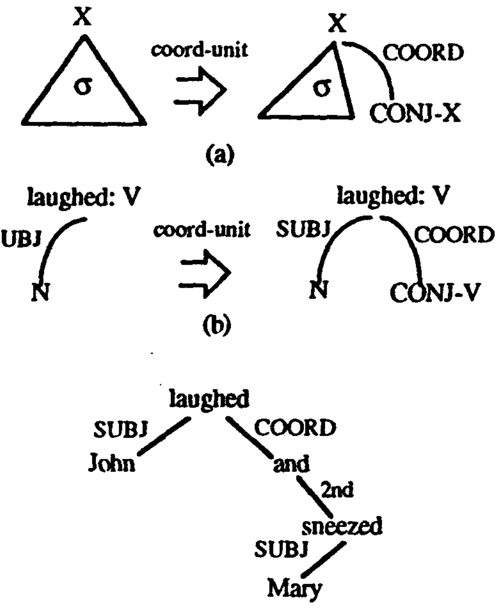


Figure 4. Graphical representation of metarule for unit coordinations (a), application to a dependency rule for *laughed* (b), and dependency tree of "John laughed and Mary sneezed".

The metarule "coord-unit" accounts for full constituent (unit) coordinations, where the two conjuncts are headed by a word of the same category. Note that "full constituents" means that conjuncts have no gaps inside. Other examples are the following sentences:

She eats apples and pears (OBJ, coordination),
Elizabeth likes to go to the zoos and to the museums (DEST coordination),
Mary thought ostrichs fly and kangaroos swim (SCOMP coordination)[7]

Difficulties with coordination arise with gapped conjuncts. Literature usually distinguishes between non-traditional-

[6] The term "unit coordination" for the full constituent coordination comes from (Huang 1984).

[7] The latter examples consider constituents that are full arguments of predicate-argument structures. Note that VP coordination, which is a full constituent coordination in phrase structure terms, is not an example of unit coordination, because the dependency rules represent the whole predicate-argument structure (including subject).

constituent coordination and gapping (see, however, (Steedman 1990), for an exception). As stated above, we deal uniformly with the two types of coordination: the common key is to classify coordination phenomena according to the elements that are missing in the conjuncts. Consider the following case of a VP-coordination:

Lucy saw a butterfly and laughed.

In fig. 5 we can see a dependency representation of this sentence. *Lucy* is co-indexed by u_1 with the trace ε_{u1} in the second conjunct.

We assume that any coordination where one of the two conjuncts misses some part of the predicate-argument structure (including adjuncts) is a *gapped coordination*. The interpretation of the gapped coordination can be summarized as follows: a coordination has a structure headed by one of the conjuncts; the missing elements are handled by traces (non lexical nodes) and u-indices, that are controlled through u-triples in non-primitive dependency rules (accounting for coordination). The dependency rule that licenses the tree in fig. 5 is the following:

saw: V(<SUBJ, N, u, ∅>
#
<OBJ, DET, ◊, ∅>
<COORD, CONJ-V, ◊, {<u, SUBJ, N>}>)

The metarule that produces this dependency rule is in fig. 7a: it states that it is possible to produce a dependency rule for SUBJ-gapped coordinations from a rule having a verb as its head and an initial d-quad which represents a SUBJ relation. In the subtree rooted by the conjunction there must be a SUBJ trace node. U-triple specifications allow to produce a uniform treatment of many kinds of argument gaps, among which the following (single and multiple) ones:

Mary cooked and John ate beans
(OBJ gap, fig. 7b)

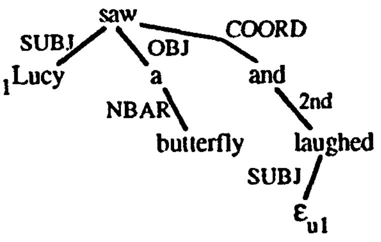


Figure 5. Dependency tree of the sentence *Lucy saw a butterfly and laughed.*

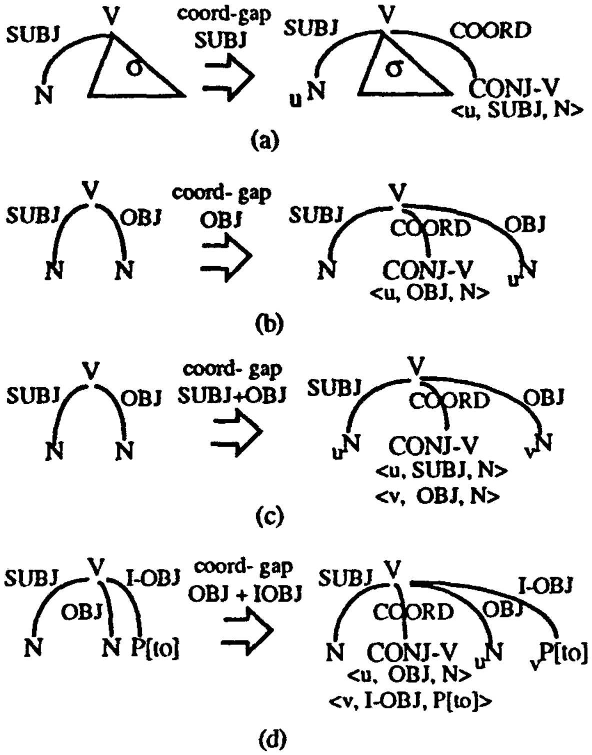


Figure 7. Metarules for several types of gapped coordinations.

Mary cooked and ate beans
(SUBJ+OBJ gap, or V coordination, fig. 7c)

John offered, and Mary actually gave, a golden Cadillac to Billy Schwartz
(OBJ+I-OBJ gap, also Right Node Raising, fig. 7d)

These metarules enforce the high attachment of actual dependents and the low attachment of gapped dependents (trace nodes). This implies that right dependents attach to the farther head of the two, always respecting the condition the projectivity. Even if the second conjunct could attach more immediately to the lower head, we think that, intonationally, the second conjunct forms a unit per se, without the right dependents. This also seems to be an empirical support of the constraint that u-triple satisfaction follows the top-down direction.

3.3 Head gapping

The formalism and the derivation procedure introduced in section 2 assume a trace node represents the gap for the whole subtree headed by the lexical node coindexed with it (*full subtree gapping*). This prevents the representation of coordination constructs where the missing structure in the second conjunct is not a whole subtree, but only a part of it, namely the head and, possibly, some of its dependents. This phenomenon occurs in the cases known as *gapping* in the literature. Here are some examples:

I saw a unicorn and Carol a tyrannosaurus (V gapping)
I gave the books to Mary and the records to Sue (V + SUBJ gapping)
John wants to give the books to Mary and Bill to Sue (complex V + OBJ gapping)

In this cases we speak of *head gapping*. In this section we extend the formalism of section 2 to account for head gapping.

The steps which are necessary for the extension are the following. (1) We must be able to associate a u-index with the head position (#); in the formalism of section 2 only d-quads (of dependents) had u-symbols associated. (2) We must introduce and keep apart in the word objects μ-indices concerning a whole subtree and μ-indices concerning a single node. (3) We must introduce and keep apart in the u-triples the u-indices concerning a whole subtree and the u-indices concerning a single node. (4) When both head and (some) dependents are gapped, we must constrain the satisfaction of the u-triples of the dependents to occur below the satisfaction of the u-triple of the head.[8] (5) We must modify the derivation relation in order to account for the changes in the formalism.

(1)
To associate a u-index with the head position in the dependency rules, we need to modify its definition:

A *dependency grammar* is a six-tuple <W, C, S, D, I, H>, where W, C, S, D, I are as defined in section 2, and H is a set of *dependency rules*, of the form

$$x:X\,(<r_1 Y_1 u_1 \tau_1> \ldots <r_{i-1} Y_{i-1} u_{i-1} \tau_{i-1}> <\#, u_i> <r_{i+1} Y_{i+1} u_{i+1} \tau_{i+1}> \ldots <r_m Y_m u_m \tau_m>)$$

where all symbols are the same as in section 2, except for the head position (#), which has been associated an index u_i (possibly $\Diamond$), such that $u_i \in I$

(2)
To introduce and keep apart the u-indices for a subtree and the ones for a single node in the derivation process, we must modify the *word objects* in 4-tuples consisting of a word w ($\in W$) or the trace symbol ε ($\notin W$) and three annotated indices η, μ and ν.
Given a grammar G, the set of word objects of G is

$$W_x(G)=\{{}_{\eta,\mu}x_{\nu} \,/\, \eta, \mu, \nu \in I^+, x \in W \cup \{\varepsilon\}\}.$$

where η concerns x as a single node, and μ concerns x as head of a subtree.

(3)
To introduce and keep apart the u-indices concerning a full subtree gapping or a head gapping in the u-triples, we introduce two forms:

- *simple* (full subtree gapping): <u, r, X>
- *complex* (head gapping): <!.u, r, x:X>

where $u \in I$, $r \in D$, $x \in W$, $X \in C$, and ! is a special symbol that specifies that the complex index concerns the single node linked by the dependency relation r.

(4)
To constrain the attachment of the trace nodes of gapped dependents to the trace node of the head, we generalize the complex form of the u-triple in

<q.u, r, x:X>

where u, r, and X are as before, and $q \in I \cup \{!\}$. Now, when the index is !.u we have a head gap indexed u (no indication on its parent, as in standard u-triple satisfaction); when the index is q.u, we have a head gap u whose parent is indexed by q.

(5)
The new derivation relation is the following:

$$\begin{aligned}
\mathrm{DR1}: \alpha <r, X(\gamma_p), u, \gamma_x>\ \psi \Rightarrow\ & \alpha \\
& <r_1, Y_1(\rho_1), u_1', \tau_1> \\
& <r_2, Y_2(\rho_2), u_2', \tau_2> \\
& \ldots \\
& <r_{i-1}, Y_{i-1}(\rho_{i-1}), u_{i-1}', \tau_{i-1}> \\
& {}_{u_i,u}x_{\Diamond} \\
& <r_{i+1}, Y_{i+1}(\rho_{i+1}), u_{i+1}', \tau_{i+1}> \\
& \ldots \\
& <r_m, Y_m(\rho_m), u_m', \tau_m> \\
& \psi
\end{aligned}$$

where

a) $x:X\,(<r_1 Y_1 u_1 \tau_1> \ldots <r_{i-1} Y_{i-1} u_{i-1} \tau_{i-1}> <\#\ u_i>$

[8] The derivation process defined in section 2 loosely constrains the satisfaction of u-triples to occur in a specific subtree. Head gapping requires a stronger constraint to satisfy the u-triples of the gapped heads and dependents in the same respective tree position.

$<r_{i+1} Y_{i+1} u_{i+1} \tau_{i+1}> \ldots <r_m Y_m u_m \tau_m>)$ is a dependency rule;

b) $\rho_1 \cup \ldots \cup \rho_m = \gamma_p \cup \gamma_x$;

c) if u is of the form q.s (where $q,s \in I$) then:

c') build a u-triple

$$<q.s,\ r,\ x{:}X>$$

and insert it in the ρ set in α or ψ containing the u-triple $<t.q, \ldots>$ ($t \in I$);

c") for each u_k ($k=1, \ldots, i-1, i+1, \ldots, m$) in the (head gapped) dependency rule such that $u_k = !.w$ ($w \in I$), replace it with $u_k' = s.w$.

The new DR1 extends the version of section 2 by accounting for the treatment of complex u-triples (part c). If the u-index (u) of the derivation object to expand ($<r, X(\gamma_p), u, \gamma_x>$) is a pair of indices of the form q.s, this means that the derivation object results from the application of a head gapped dependency rule. Head gapping can possibly include some dependents: DR1 collects all the word objects that participate to the gapping phenomenon. This process has two phases: c' builds a new u-triple for the current derivation object, and inserts it in the ρ set that already contains the head of the current gapped element (of the form <t.q, ...>); in other words, c' builds a "subtree" of gapped elements; c" obtains the actual (instantiated) index by referencing the current element as head of the next dependent.

DR2: $\alpha <r, X(<v, r, X>), u, ()> \psi \Rightarrow \alpha\ {}_{\Diamond,u}\varepsilon_v\ \psi$

DR2 is still the rule for full subtree gapping: it only applies to the satisfaction of simple u-triples.

DR3: $\alpha <r, X(\gamma_p), u, \gamma_x> \ \psi \Rightarrow$

$$\begin{array}{l}
\alpha \\
<r_1, Y_1(\rho_1), u_1, \tau_1> \\
<r_2, Y_2(\rho_2), u_2, \tau_2> \\
\ldots \\
<r_{i-1}, Y_{i-1}(\rho_{i-1}), u_{i-1}, \tau_{i-1}> \\
{}_{u_i,u}\varepsilon_w \\
<r_{i+1}, Y_{i+1}(\rho_{i+1}), u_{i+1}, \tau_{i+1}> \\
\ldots \\
<r_m, Y_m(\rho_m), u_m, \tau_m> \\
\psi
\end{array}$$

where

a) $x{:}X\ (<r_1 Y_1 u_1 \tau_1> \ldots <r_{i-1} Y_{i-1} u_{i-1} \tau_{i-1}> <\#\ u_i> <r_{i+1} Y_{i+1} u_{i+1} \tau_{i+1}> \ldots <r_m Y_m u_m \tau_m>)$ is a dependency rule;

b) γ_p includes a complex u-triple $<v.w, r_X, x{:}X>$, $v \in I \cup \{!\}$, $w \in I$;

c) for each complex u-triple $<w.t, r_Y, y{:}Y>$ in γ_p, there exists a d-quad $<r_{k_i} Y_{k_i} u_{k_i} \tau_{k_i}>$ in the dependency rule such that $r_{k_i} = r_Y$ and $Y_{k_i} = Y$, and $\rho_{k_i} \supseteq \{<t, r_{k_i} Y_{k_i}>\}$;

d) $\rho_1 \cup \ldots \cup \rho_m = \gamma_x \cup \gamma_p - \{<v.w, r_X, x{:}X>\}$

DR3 is totally new, and accounts for the satisfaction of complex u-triples. The complex u-triples that are ready for satisfaction are those of the form $<v.w, r_X, x{:}X>$. The satisfaction mechanism looks for a dependency rule headed by x:X that contains a d-quad for each u-triple that represents a gapped dependent of the head η-indexed by v. The result is a trace node where the v-index is w, and the η- and the μ-indices are computed as usual. Finally, DR3 modifies the set of u-triples γ_p by removing the u-triple which has been satisfied.

For the (V-gapped) sentence

I saw a unicorn and Carol a tyrannosaurus

we need the metarule in fig. 7a, which produces the TARGET dependency rule

saw: V (<SUBJ, N, ◊, ∅> <#, !.u> <OBJ, N, ◊, ∅> <COORD, CONJ-V, ◊, {<!.u, 2nd, V>}>)

which licenses the dependency tree in fig. 7b.

For the (V+SUBJ gapped) sentence

I gave the books to Mary and the records to Sue (V + SUBJ gapping)

the (non primitive) dependency rule is:

gave: V (<SUBJ, N, u.v, ∅> <#, !.u> <OBJ, N, ◊, ∅> <I-OBJ, P[to], ◊, ∅> <COORD,CONJ-V, ◊, {<!.u, 2nd, V>, <u.v, SUBJ, N>})

which is built by the metarule in fig. 8a.

The dependency tree which results from the derivation of this sentence is in fig. 9.

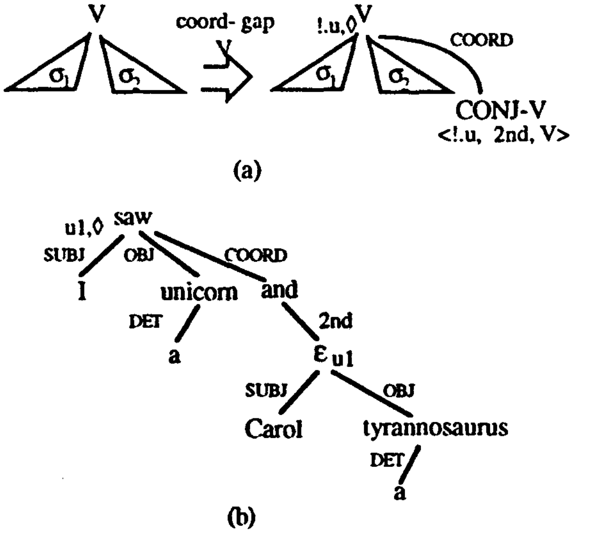


Figure 7. The metarule for verb gapping (a) and the dependency tree of the sentence "I saw a unicorn and Carol a tyrannosaurus" (b).

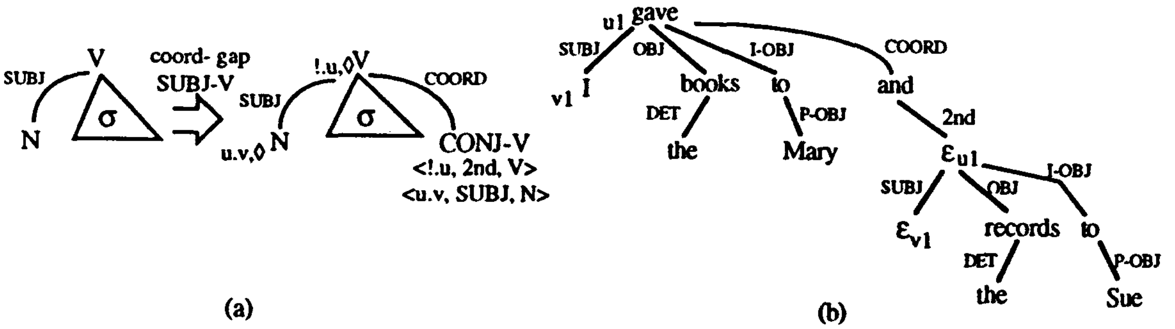


Figure 8. The dependency tree associated with the sentence "I gave the books to Mary and the records to Sue".

4.Conclusions

The paper has described a lexicalized dependency formalism and an asymmetric dominance-based treatment of coordination. The linguistic knowledge is formalized in primitive dependency rules and a number of metarules, which produce non primitive rules from primitive rules. Long-distance dependencies are represented by coindexing lexical and non lexical nodes in the dependency tree. Metarules can generate the non primitive rules that license long distance dependencies and, in particular, those connected with coordination gapped constructs.

The paper gives an account of pair coordinations, but the formalism seems to be generalizable to any number of conjuncts. The basic conjunct unit is the predicate-argument structure including adjuncts. This assumption leads to the broad classification in unit and gapped coordinations. As we have seen, this can caused some non standard analyses of Right Node Raising and verb coordination.

The formalism presented in this paper is only a first attempt of producing a dependency formalism that can account for coordination without resurging to phrase structure augmentations or string-based representations. The formalism is thus suitable to left-to-right incremental processing, because of the dominance-based character of the representation. We are aware that the formalism is weak on the linguistic competence, because it probably overgenerates with respect to legal sentences; however, we believe that the formal tools introduced (constraints on the u-triple satisfaction) can be successfully used to eliminate this problem.

Acknowledgements

This paper has benefited from comments of the people at a seminar held by one of the authors at IRCS, UPenn, Philadelphia.

References

Barbero C., Lesmo L., Lombardo V., Merlo P., *Integration of syntactic and lexical information in a hierarchical dependency grammar*, this volume.

Gazdar G., Klein E., Pullum G., Sag I., *Generalized Phrase Structure Grammar*, Basil Blackwell, Oxford, 1985.

Huang X., *Dealing with conjunctions in a machine-translation environment*, Proc. of COLING 84, pp. 243-246.

Hudson R., *English Word Grammar*, Basil Blackwell, Oxford, 1990.

Lombardo V., Lesmo L., *Formal aspects and parsing issues of dependency theory*, in Proc. of ACL/COLING 98.

Mel'cuk I., *Dependency Syntax: Theory and Practice*, SUNY Press, Albany, 1988.

Neuhaus P., Broker N., *The Complexity of Recognition of Linguistically Adequate Dependency Grammars*, Proc. ACL/EACL97, Madrid, 1997, pp.337-343.

Ross J. R., *Gapping and the order of constituents*, in Bierwisch M. and Heidolph M. (eds.), *Progress in Linguistics*, Mouton, The Hague, 1970, pp. 249-259.

Sarkar A., Joshi A.K., *Handling Coordinatin in a Tree Adjoining Grammar*, Unpublished manuscript, Department of Computer and Information Science, University of Pennsylvania, Philadelphia (PA), February 1997.

Steedman M., *Dependency and Coordination in the grammar of Dutch and English*, **Language 61**, 1985, pp. 523-568.

Steedman M., *Gapping as constituent coordination*, **Linguistics and philosophy 13**, 1990, pp. 207-264.

Steedman M., *Surface structure and interpretation* , MIT Press, 1996.

Tesniere L., *Elements de syntaxe structurale*, Klincksieck, Paris 1959