PRUNING SEARCH SPACE FOR PARSING FREE COORDINATION IN CATEGORIAL GRAMMAR

Crit Cremers
Maarten Hijzelendoorn
Department of General Linguistics, Leiden University, The Netherlands
{cremers,hijzelendrn}@rullet.leidenuniv.nl

Abstract

The standard resource sensitive invariants of categorial grammar are not suited to prune search space in the presence of coordination. We propose a weaker variant of count invariancy in order to prune the search space for parsing coordinated sentences at a stage prior to proper parsing. This Coordinative Count Invariant is argued to be the strongest possible instrument to prune search space for parsing coordination in categorial grammar. Its mode of operation is explained, and its effect at pruning search space is exemplified.[1]

1 Lexical Ambiguity and Natural Language Parsing

Lexical ambiguity is known to be a major threat to efficient parsing of natural language. It is easy to see why. Let G_{NL} be a grammar for a language NL, and L a lexicon with initial assignment A of nonterminals to the words of NL. Let S = $w_1 \dots w_n$ be a sentence over L. Under a parsing-as-deduction-approach, then, the parsing problem for S is whether for some sequence C = $c_1 \dots c_n$, with $c_i \in \mathrm{A}(w_i)$, C is derivable under G_{NL}. The solution to the problem may require checking the derivability of many such Cs. Basically, the number of sequences of which the derivability must be checked for a certain S is $\prod_1^n|\mathrm{A}(w_i)|$, exponentially dependent on *n*. $\prod_1^n|\mathrm{A}(w_i)|$ measures the search space for parsing S. There is a trade-off between lexical ambiguity and properties of the grammar. In particular, a grammar may assign nonterminals to certain lexically assigned nonterminals, by having monadic rules or theorems of type $c \rightarrow c'$. In that case, the search space for parsing S is partially erected by the grammar itself. Again, the question whether G_{NL} derives S, explodes to a multiplicity of queries as to whether G_{NL} derives a certain C, thus simulating the effect of structural variation in parsing phrase structure grammar. But categorial grammar tends to spell out (or compile) structural variation in the lexicon. A certain degree of lexical ambiguity, or rather: polymorphism *per* lexical atom, seems inevitable and inherent to the expressive power of natural language. In categorial grammar, for example, differences in subcategorization (a verb may select an infinitival as well as a tensed complement), word order (a finite verb may have its complements to the left or to the right) or functionality (a word may be a preposition or a particle) must lead, in some stage of the parsing process, to branching possibilities of assignments and an increase of search space.

To a large extent, the art of parsing consists in finding secure means to restrict the number of possible assignments. Given the function $\prod_1^n|\mathrm{A}(w_i)|$ over strings of words, efficiency requires serious pruning

[1] We are greatly indebted to three anonymous referees for their valuable comments on an earlier version of this paper, and to Michael Redford and Jeroen van de Weijer.

of the search space. Optimally, the means for pruning are anchored in the grammar that is to be applied. Occurrence-sensitive or linear categorial grammar offers some options for pre-checking assignments. In this article, we will discuss a particular instrument that overcomes a flaw in the effectiveness of the standard pre-checks in the presence of coordination. This instrument is developed and tested as a part of a parsing system for Dutch - Delilah - in which the lexicon is the main generator of search space.[2] Some test results are presented in section 4.

2 Categorial Count Invariance and Parsing Coordination

Certain categorial calculi exhibit a property that is known as *count invariance* [Van Benthem 1986]. For these logics, it is true that if a proposition $Y \longrightarrow z$ is derivable, the string to the left of → and the type to its right share the results of a particular way of counting occurrences of basic types. This count protocol discriminates between positive and negative occurrences of basic types; for each basic type x and for each string of types S it yields an integer representing the occurrences of x in S. Here and below, slash categories are invariantly written in the format *(head)-direction-argument*, where *direction* is indicated by the slash; the bracketing of the head is often suppressed.

(1) **Count Protocol**
for each basic type x,
$count_x(x) = 1$
$count_x(y) = 0$ if y is a basic type and $y \neq x$
$count_x(y/z) = count_x(y\backslash z) = count_x(y) - count_x(z)$
$count_x(y_1, ..., y_n) = count_x(y_1) + ... + count_x(y_n)$.

Because of this way of counting, a complex type x/y will be said to contain a positive occurrence x of type x and a negative occurrence $/y$ of type y. The main application of type count is stated in (2):

(2) **Count Invariance for grammar *G***
if $Y \longrightarrow z$ is derivable in some resource sensitive categorial grammar G, then for all basic types x, $count_x(Y) = count_x(z)$. By consequence, if z is a basic type, then for all basic types $x \neq z$, $count_x(Y) = 0$.

Count Invariance can be proved for resource sensitive calculi like Lambek, Lambek+Permutation and Ajdukiewicz/Bar-Hillel, but does, of course, not hold in systems that perform Contraction or Expansion - with $y\ y \longrightarrow y$ - and/or Monotonicity - if $X \longrightarrow y$ then $X\ x \longrightarrow y$; see [Van Benthem 1991]. Count Invariance underlies the notion of balance in proof nets [Roorda 1992]: for a sequent $Y \longrightarrow z$ (in Roorda's notation: $y_1...y_n\, z$) to be balanced implies that for every basic type x, $count_x(Y) - count_x(z) = 0$. Consequently, if for some (sub)sequent S $count_x(S) \neq 0$, S is unbalanced in x.
Count Invariance can be used to delimit the search space unfolded by lexical ambiguity [Moortgat 1988]. By contraposition of (2), a proposition $Y \longrightarrow z$ cannot be proved in a count invariant system G if for some basic type x, $count_x(Y) \neq count_x(z)$.

[2] This parsing system is available athttp://fonetiek-6.leidenuniv.nl/hijzlndr/delilah.html. Among the phenomena it deals with are unbounded coordination, verb clustering including cross-serial dependencies, wh-movement, topicalization and adjunction.

Almost by definition, coordination in natural language involves the multiplication of types: a certain subsequence of types to the left of the coordination point is doubled or mirrored at the right of the coordination point. The relative unrestrictedness of coordination is reflected in the proposal to categorize coordinating elements like *and* as *(X\X)/X*, i.e. by means of essential variables over types [Wittenburg 1986, Moortgat 1988, Steedman 1990, Emms 1993]. Thus they are assigned type frames (polymorphic types) rather than types. We may assume that these frames do not occur negatively in any other type, as coordinations - or free coordinations, at least - are not selected by other elements in a natural language [Grootveld 1993].

This categorization as a frame of type variables also accounts for the doubling of types induced by coordination. This doubling would interfere with Count Invariance: unless the repeated sequences are themselves fully balanced, the count of certain types will be unbalanced by the repetition of subsequences. The variables in the coordinator's type frame *(X\X)/X* are supposed to be instantiated by a type to which both the repeated sequences can be reduced. Since, according to the Count Protocol, the type frame for *and* halves the counts for whatever type substitutes *X*, the effect of doubling is neutralized and Count Invariance is still a property of a coordinated sentence.

Unfortunately, this is not sufficient to keep Count Invariance available as a method to prune the search space for coordination. In fact, Count Invariance (2) needs more information than is available prior to proper parsing, to do its pruning job. It is not difficult to see what prevents it from being effective in the presence of coordination. Consider the following sequent as one of the possible hypotheses emenating from a lexicon concerning the assignment of categories to the words of the coordinated sentence w_1 ... *and* ... w_n, according to which the category c_i is assigned to w_i.

(3) $$c_1 \ldots c_i \ldots c_k\ (X\backslash X)/X\ c_{k+2} \ldots c_m \ldots c_n \rightarrow s$$

In order to know whether it is sensible to look for a possible proof of this sequent, we would like to check Count Invariance, i.e. the property that for every basic type x, $count_x(c_1 \ldots c_n) = count_x(s)$. Since the sequent contains a coordinator, we know that for the sequent to be derivable, for some i and some m the segment $c_i \ldots c_k$ and the segment $c_{k+2} \ldots c_m$ must have identical count characteristics, which accumulate in the overall count. According to one's theory of coordination and the scope of the intended grammar, one can relax the conditions on the two count characteristics, but the categorial structure of the two coordinates will always be related in one way or another, and will always be dependent on the non-coordinated context. In the spirit of [Sag *et al.* 1985], one could try to define a more general or liberal relation between the categorial characteristics of the coordinates. This would only complicate but not alter the problem of finding the coordinates. For ease of exposure, however, we choose here the most restrictive form of coordination, and require the coordinates to be categorially equal.

The doubling of type occurrences disturbs the count balances. As a matter of fact, the relevant instances of Count Invariance are $count_x(c_1 \ldots c_n) - count_x(c_{k+2} \ldots c_m) = count_x(s)$; here the part of the string counted twice is substracted to restore the balance. In order to check these instances at (3), however, we have to know which is the proper value of m, and of i, for that matter. Since we have not yet parsed the sequent - we are just testing whether it should be parsed - we do not have information as to the borders of the coordinates. It is well known that neither coordinates nor their borders are locally marked - which is a problem for parsing itself anyway [Cremers 1993]. Consequently, we can only guess i and m without having any clue, and apply Count Invariance with respect to each pair. This is easily recognized as a procedure which is as complex as parsing: for every assignment of lexical categories to the sentence, we have to check every possible pair <*i*, *m*>, and if we find a pair for which Count Invariance can be established, this pair marks the borders of the coordination. But this implies that we

(partially) parse that particular assignment while we are pre-checking its derivability. It follows that we cannot use Count Invariance to prune search space, since applying Count Invariance presumes the exploration of search space. In the presence of coordination, Count Invariance is a blunt knife.
It is worth noting that replacing the type frame for coordinators by a set of constant types does not solve the problem. It would just lead to the additional hypothesis that the coordinator type which was (randomly) selected in a certain assignment, complies with the counts for c_i .. c_k and c_{k+2} .. c_m. Again, by lack of a proper parse we can only guess the values i and m. In either case, therefore, Count Invariance (2) faces a comparison of two unknown elements.

This indeterminacy of coordination at the level where Count Invariance is supposed to come in, is a property of natural language coordination as such. It is not caused or provoked by a particular way of categorizing a language. Since almost everything can almost always be coordinated and coordinations are generally neither selected nor selecting, a string of words or categories can contain no immediately accessible hints as to what is coordinated in that particular case.
In the presence of coordination, Count Invariance (2) thus turns out to be of no help at selecting viable sequences prior to parsing. Therefore, we developed a weaker alternative which can be effectively exploited in delimiting the search space for coordinated sentences. It operates at least on the basis of a nonassociative, context-sensitive and bracket-free categorial grammar, as designed in [Cremers 1993]. Its basic properties, however, are definable for any grammar that is sensitive to directionality and occurrence.

3 A Count Invariant for Coordinated Sequences

Coordination can be constructed so as to imply that certain elements outside the scope of the coordination have a double task with respect to elements inside the scope of the coordination: they have to serve elements in both coordinates. Under this approach a coordinator is treated syncategorematically, and thus as combinatorially inactive. For example, in a sequent

(4) $a/b\ b\ [\&]\ b\ d\backslash a \rightarrow d$

the single negative occurrence */b* in *a/b* has to deal with the two positive occurrences of *b* to the left and the right of the coordinator & which itself does not contribute to balancing the sequent. In fact, we see that *b* is coordinated in that string, and being so it overcharges, in terms of count balance, its non-coordinated context. By contrast, the negative occurrence *\a* in *d\a* is cancelled on a one-to-one basis by the positive occurrence *a* in *a/b*; neither *a* nor *\a* occurs in the scope of coordination, and they are balanced. This phenomenon is general: in a well-formed coordinated string, being in the scope of coordination determines whether or not the occurrences of a type are balanced (for some relevant lemmas in this respect, see [Cremers 1993, ch. 3.3]. Types occurring in a coordinate and not balanced inside that coordinate, will require certain other types to take care of more than one of them. The outside types, like */b* in (4), can be said to have double functions.
By just counting the positive and negative occurrences of primitive types to the left and the right of the coordinator, we can check whether enough suitable double function categories are available. Their presence in the sequent is a necessary condition for grammaticality. For example, if we change (4) into (5) by adding a positive occurrence *b* to the left of the coordinator which cannot be part of the coordination, the intended double function type */b* is no longer available for the coordinated *b*'s, and the sequent must be rejected.

(5) $a/b\ b\ b\ [\&]\ b\ d\backslash a \rightarrow d$

In order to bring about this rejection in an early stage, we only have to know the count values of the intended coordinated substring, since these values will specify the nature of the need for double function primitives. The question is, then, how to determine these 'coordinated' count values prior to proving the corresponding propositions, i.e. prior to deriving the proper coordinates.

For that purpose, we need a particular counting device which keeps track of detailed information on the categorial architecture of an assignment. Suppose we have a string of words of the form $w_1 \ldots w_i$ *and* $w_{i+2} \ldots w_n$. Let L be an assignment of types $c_1 \ldots c_i$ to $w_1 \ldots w_i$; let R be an assignment of types $c_{i+2} \ldots c_n$ to $w_{i+2} \ldots w_n$. Assume that a negative and a positive occurrence of a certain type cancel each other on a one-to-one basis. In a sequent $\ldots x \ldots x \ldots y\backslash x \ldots$ exactly one of the x-occurrences and the $\backslash x$-occurrence cancel each other; the other x is free if there is no other occurrence of $\backslash x$ or $/x$ in the sequent. In this stage of parsing we cannot and do not need to decide which particular occurrence is cancelled against which other occurrence. We are just interested in the numbers of cancellation candidates.

Accordingly, with L can be associated a register reg_L of quadruples of integers <*bound_head, bound_arg, free_head, free_arg*>, such that for each primitive type x there is an x-quadruple specifying:

$bound\_head^x_L$: the number of (positive) occurrences of x in L that are cancelled by (negative) occurrences of x under a \-slash ($\backslash x$) in L;

$bound\_arg^x_L$: the number of (negative) occurrences of x under a /-slash ($/x$) in L that are cancelled by (positive occurrences) of x in L;

$free\_head^x_L$: the number of (positive) occurrences of x in L that are not cancelled by (negative) occurrences of $\backslash x$ or $/x$ in L;

$free\_arg^x_L$: the number of (negative) occurrences of $/x$ in L that are not cancelled by (positive) occurrences of x in L.

Each occurrence (positive or negative) of a type is counted once in its register. The procedure for this way of counting differs from the count protocol (1) underlying Count Invariance (2) in being sensitive to the direction of the negative occurrences. A head (or positive occurrence of) x is counted as free in L if there is no $/x$ to its left or $\backslash x$ to its right in L by which it could be cancelled. Similarly, an argument (or negative occurrence of) $/x$ is free in L if no head to its right can possibly match it. There is no need to count free occurrences of $\backslash x$. Arguments $\backslash x$ cannot be free in L if L is to be the prefix of a derivable sequent, by the directionality invariant of [Steedman 1990]: no rule can change the directionality of an argument type; if an occurrence of $\backslash x$ is free in a prefix, it can never be cancelled by a positive occurrence to its right, and thus it is doomed to remain free. A free argument $\backslash x$ will obstruct any derivation of a sequent in which it occurs. As soon as a free argument $\backslash x$ shows up in L, L can be rejected as a prefix of the sequence of assignments to the sentence.

For R, a similar register reg_R is supposed to be available, though directional parameters are reversed.

Here is an example of a full register reg_L for basic types a, b, c, and of the register for the first three types in L.

(6) L = $a/c\ b/c/a\ b\backslash b\ c\backslash b\ a\ a$

$reg_L = \{\ a{:}{<}0,1,2,0{>},\ b{:}{<}1,0,0,0{>},\ c{:}{<}0,1,0,1{>}\ \}.$

$L_3 = a/c\ b/c/a\ b\backslash b$ (a prefix of L)

$reg_{L3} = \{\ a{:}{<}0,0,1,1{>},\ b{:}{<}1,0,1,0{>},\ c{:}{<}0,0,0,2{>}\ \}$

Computing the register is a linear task. It is computed and updated incrementally and accumulatively whenever a new type is added to the prefix. In (6), reg_L reflects the changes in reg_{L3} after adding $c\backslash b$, a and a to L_3.

What can the registers reg_L and reg_R tell us about combining L and R into a hypothesis $L\ [\&]\ R \rightarrow s$? The values for *free_head*^{x} and *free_arg*^{x} in each quadruple provide the number of positive and negative occurrences, respectively, that are not cancelled at their side of the coordinator. These occurrences may or may not be in the domain of coordination. We can determine the occurrence patterns that are characteristic for grammatical, i.e. derivable sequents as follows (for some more formal elaboration on this topic, see [Cremers 1993, ch. 3]. Suppose a free (positive) occurrence a in L is in the scope of coordination. By the nature of coordination, this occurrence has a matching occurrence in R that is necessarily cancelled at its side. For a free occurrence of x in L to be inside the scope of coordination, a potentially cancelling type must be available in the balanced part of R -this type will have a double function. By the same line of reasoning, if the occurrence of x which is free in L, is to be outside the scope of coordination, it has to cancel against some free occurrence in R. We can illustrate the range of possibilities with a simple example; it is assumed that the substrings V, W, X, Y and Z do not contain occurrences of type a.

(7) $X\ a\ Y\ [\&]\ W\ a\ V\ b\backslash a\ Z \rightarrow s$
$\mathrm{L} = X\ a\ Y$
$\mathrm{R} = W\ a\ V\ b\backslash a\ Z$
a-quadruple in reg_L = <0, 0, 1, 0>
a-quadruple in reg_R = <0, 1, 0, 0>

The a-quadruple in reg_L tells us that some positive occurrence of a is free in L: *free_head*$^a_L = 1$. This occurrence may be part of the coordinated substring. If it is, there must be some negative occurrence in R that takes care of both a in L and its matching counterpart in R. This negative occurrence, however, is cancelled in R, and must be accounted for in the number of bound arguments $\backslash a$ in reg_R, *bound_arg*^a_R; this number happens to be 1, due to the composition of R . Now suppose the positive free occurrence of a in L is not inside the scope of coordination. Then there must be a negative occurrence $\backslash a$ free in R. This negative occurrence should be counted in *free_arg*^a_R. Since, in example (7), *free_arg*^a_R has the value 0, the latter assumption is rejected; all the occurrences counted in *free_head*^a_L must be in the scope of coordination.

This type of reasoning can be generalized in order to decide, under the hypothesis that the coordinated sequent is derivable, how many balanced negative and positive occurrences of basic types in L and R must be in the scope of coordination. Here is the argument.

Let λ_x be the difference *free_head*^x_L – *free_arg*^x_R, for some basic type x. Clearly, if $\lambda_x > 0$, there are λ_x positive occurrences of x in L which are not cancelled by negative occurrences $\backslash x$ free in R. These λ_x occurrences must therefore be in the scope of coordination and be co-covered by already bound negative occurrences $\backslash x$ in R. In that case, the value of *bound_arg*^x_R must be at least as large as λ_x. On the other hand, if $\lambda_x < 0$, there must be negative occurrences $\backslash x$ in R that, for the string to be grammatical, must be dealt with by already cancelled occurrences x in L, accounted for in the number *bound_head*^x_L; this number must be large enough to accommodate the $|\lambda_x|$ negative occurrences $\backslash x$ in R. If $\lambda_x = 0$, all or none of the *free_head*^x_L positive occurrences and *free_arg*^x_R negative occurrences are in the scope of coordination, depending on other parameters. An equivalent reasoning can be built around the value ρ_x, this being the difference *free_head*^x_R – *free_arg*^x_L. (As a corollary, $\lambda_x + \rho_x$ = *count*$_x$*(C)* for that proper affix C of L and of R that happens to be in coordination.)

The above reasoning gives us the following inequalities for two assignments L and R:

(8) **Coordinative Count Invariant**

if $L\ [\&]\ R \rightarrow s$ is derivable, then

for every basic type $x \neq s$ such that

$<bound\_head^x_L,\ bound\_arg^x_L,\ free\_head^x_L,\ free\_arg^x_L>$ is in reg_L and

$<bound\_head^x_R,\ bound\_arg^x_R,\ free\_head^x_R,\ free\_arg^x_R>$ is in reg_R and

$\lambda_x = free\_head^x_L - free\_arg^x_R$ and

$\rho_x = free\_head^x_R - free\_arg^x_L$,

it is true that

$\lambda_x \leq bound\_arg^x_R$ and

$\rho_x \leq bound\_arg^x_L$ and

$-\lambda_x \leq bound\_head^x_L$ and

$-\rho_x \leq bound\_head^x_R$.

By contraposition, a sequence of types with a coordinator is not reducible to s if for some primitive type x one or more of the inequalities in (8) does not hold. This justifies the following statement:

(9) **Conjoinability**

An assignment L of types to the words to the left of a coordinator and an assignment R of types to the words to its right are *conjoinable with respect to a basic type x* iff the quadruples for x in reg_L and reg_R satisfy the inequalities of (8).

Two strings of types L and R are *conjoinable* as $L\ [\&]\ R$ iff L and R are conjoinable with respect to every basic type x, $x \neq s$.

This is the invariant that can do the job of selecting coordinated type sequences prior to proper parsing. It requires neither information nor guesses as to the nature and the extent of the coordination in the sentence which is to be parsed.

Given a sentence W_l *and* W_r, a set LL of assignments to W_l and a set RR of assignments to W_r, with registers associated to each member of each set, it is straightforward to select those pairs <L, R> in LL × RR that are conjoinable according to (9). Only these pairs have to be checked for derivability. The pairs rejected as not conjoinable cannot represent a well-formed coordination, by *modus tollens* applied to (8).

Here is a simple example involving just two basic types where both LL and RR contain only two assignments; the designated type s is neglected, as its count deserves special treatment. (10) specifies the four subsequences of assignments to a coordinated sentence of length 7, including the coordinator, and the related registers. (11) - (14) specify the relevant data for a particular member of the product LL × RR, according to the registers in (10), and apply the Coordinative Count Invariant (8) in a *modus tollens* mode. If one of the queries fails, the hypothesis that that particular member of LL × RR gives rise to a derivable sequent, is rejected, by Conjoinability (9).

(10)	LL = {L_1, L_2}	
	L_1 = a b/b a\b	reg_{L1} = {a:<0,0,2,0>, b:<1,0,0,1>}
	L_2 = a b a\b	reg_{L2} = {a:<0,0,2,0>, b:<1,0,0,0>}
	RR = {R_1, R_2}	
	R_1 = a\b s\a\a	reg_{R1} = {a:<0,1,0,1>, b:<0,0,0,1>}
	R_2 = a\b s\b\a	reg_{R2} = {a:<0,1,0,0>, b:<0,0,0,2>}

(11) hypothesis 1: $L_1\ [\&]\ R_1 \longrightarrow s$ (L_1 and R_1 are conjoinable)

$\lambda_a = 1$; $\rho_a = 0$; $\lambda_b = -1$; $\rho_b = -1$; $\text{bound\_arg}^{a}_{L} = 0$; $\text{bound\_arg}^{a}_{R} = 1$; $\text{bound\_head}^{a}_{L} = 0$; $\text{bound\_head}^{a}_{R} = 0$; $\text{bound\_arg}^{b}_{L} = 0$; $\text{bound\_arg}^{b}_{R} = 0$; $\text{bound\_head}^{b}_{L} = 1$; $\text{bound\_head}^{b}_{R} = 0$

$\lambda_a \leq \text{bound\_arg}^{a}_{R}$? *true* ; $\rho_a \leq \text{bound\_arg}^{a}_{L}$? *true*; $-\lambda_a \leq \text{bound\_head}^{a}_{L}$? *true*; $-\rho_a \leq \text{bound\_head}^{a}_{R}$? *true*; $\lambda_b \leq \text{bound\_arg}^{b}_{R}$? *true*; $\rho_b \leq \text{bound\_arg}^{b}_{L}$? *true* ; $-\lambda_b \leq \text{bound\_head}^{b}_{L}$? *true*; $-\rho_b \leq \text{bound\_head}^{b}_{R}$? *false*

▲ hypothesis 1 rejected by (9)

(12) hypothesis 2: $L_1\ [\&]\ R_2 \longrightarrow s$ (L_1 and R_2 are conjoinable)

$\lambda_a = 2$; $\rho_a = 0$; $\lambda_b = -2$; $\rho_b = -1$; $\text{bound\_arg}^{a}_{L} = 0$; $\text{bound\_arg}^{a}_{R} = 1$; $\text{bound\_head}^{a}_{L} = 0$; $\text{bound\_head}^{a}_{R} = 0$; $\text{bound\_arg}^{b}_{L} = 0$; $\text{bound\_arg}^{b}_{R} = 0$; $\text{bound\_head}^{b}_{L} = 1$; $\text{bound\_head}^{b}_{R} = 0$

$\lambda_a \leq \text{bound\_arg}^{a}_{R}$? *false*

▲ hypothesis 2 rejected by (9)

(13) hypothesis 3: $L_2\ [\&]\ R_1 \longrightarrow s$ (L_2 and R_1 are conjoinable)

$\lambda_a = 1$; $\rho_a = 0$; $\lambda_b = -1$; $\rho_b = 0$; $\text{bound\_arg}^{a}_{L} = 0$; $\text{bound\_arg}^{a}_{R} = 1$; $\text{bound\_head}^{a}_{L} = 0$; $\text{bound\_head}^{a}_{R} = 0$; $\text{bound\_arg}^{b}_{L} = 0$; $\text{bound\_arg}^{b}_{R} = 0$; $\text{bound\_head}^{b}_{L} = 1$; $\text{bound\_head}^{b}_{R} = 0$

$\lambda_a \leq \text{bound\_arg}^{a}_{R}$? *true*; $\rho_a \leq \text{bound\_arg}^{a}_{L}$? *true* ; $-\lambda_a \leq \text{bound\_head}^{a}_{L}$? *true*; $-\rho_a \leq \text{bound\_head}^{a}_{R}$? *true*; $\lambda_b \leq \text{bound\_arg}^{b}_{R}$? *true*; $\rho_b \leq \text{bound\_arg}^{b}_{L}$? *true* ; $-\lambda_b \leq \text{bound\_head}^{b}_{L}$? *true*; $-\rho_b \leq \text{bound\_head}^{a}_{R}$? *true*

▲ hypothesis 3 can not be rejected by (9).

(14) hypothesis 4: $L_2\ [\&]\ R_2 \longrightarrow s$ (L_2 and R_2 are conjoinable)

$\lambda_a = 2$; $\rho_a = 0$; $\lambda_b = -2$; $\rho_b = 0$; $\text{bound\_arg}^{a}_{L} = 0$; $\text{bound\_arg}^{a}_{R} = 1$; $\text{bound\_head}^{a}_{L} = 0$; $\text{bound\_head}^{a}_{R} = 0$; $\text{bound\_arg}^{b}_{L} = 0$; $\text{bound\_arg}^{b}_{R} = 0$; $\text{bound\_head}^{b}_{L} = 1$; $\text{bound\_head}^{b}_{R} = 0$

$\lambda_a \leq \text{bound\_arg}^{a}_{R}$? *false*

▲ hypothesis 4 is rejected by (9).

Only one of the four possible hypotheses concerning the derivability of the sentence underlying the assignments in LL × RR is submitted to the proper parsing procedure. One can easily check that this sequent, *a b a\b [&] a\b s\a\a → s*, is the only one that induces a proper coordination. The type *a\b* is coordinated. If one occurrence of this type and the inert coordinator type *&* are neglected, the resulting sequent *a b a\b s\a\a → s* complies with the general Count Invariance (2) and will turn out to be derivable under standard assumptions of categorial grammar. The other three hypotheses can be rejected on the basis of the checklist defined in the consequent of (8); these checks are performed in constant time, not dependent on the length of the sentence cq. the sequents of types. This rejection is correct. None of the hypothesis checked in (11), (12) and (14) looks fit for derivability. In particular, they do not appear to contain two coordinates in such a fashion that the non-coordinated context with one of the coordinates added makes sense as a derivable sequent. For example, the hypothesis tested in (12), *a b/b a\b [&] a\b s\a\a → s*, may seem to contain a coordination *a\b [&] a\b*, but the 'de-coordinated' sequent *a b/b a\b s\a\a → s* from which the coordination is removed, does not even comply with (2) and is thus inderivable. Therefore, the check in (12) correctly rejects the hypothesis.

4 Effectiveness of Conjoinability

Conjoinability (9) has been implemented in a categorial parsing system called Delilah, which embodies among other things the algorithm for parsing unbounded coordination of [Cremers 1993]. In this system the selection of conjoinable pairs of strings is completely deterministic. During the construction of an assignment, a register is maintained and associated with it. If the assignment passes some other occurrence tests, it is admitted to LL or RR; its register is fixed. The procedure for comparing assignments in the product LL × RR checks pairs of quadruples from the registers in the spirit of the contraposition to the Coordinative Count Invariant (8). This requires only a fixed number of steps per pair of quadruples and per pair of registers. Furthermore, the check based on the Coordinative Count Invariant (8) is applied in addition to and in accordance with occurrence checks living on the more general Count Invariance (2); for these checks, see [Cremers 1989]. Here is an example of the effect of conjoinability in Delilah.

Sentence (15) contains 51 words from a restricted but highly polymorphic lexicon of Dutch. In this lexicon, all kinds of combinatorial options for each word are spelled out as categorial types. Because of this lexical polymorphism, the number of possible different assignments of lexical categories to the sentence is 5.3e16. The coordinator *en* is handled syncategorematically.

(15) Omdat ik niet had willen zeggen dat ik door de man met
Because I not had want say that I by the man with
de auto werd gedwongen te proberen Henk met de pop **en**
the car was forced to try Henk with the doll and
Agnes met een boek te laten spelen, werd de man door de
Agnes with a book to let play, was the man by the
vrouw gedwongen te zeggen dat hij mij niet met de pop
woman forced to tell that he me not with the doll
wilde proberen te laten spelen.
wanted try to let play.

First, some general occurrence checks that are not related to coordination keep the number of 5.6e16 possibilities just virtual by dynamically reducing the set of viable assignments of categories to this string to 2.9e6, or 6e–9 % of the search space. This number is the product of 136 assignments to the left of the coordinator - the set LL - and 21576 assignments to the right of the coordinator - the set RR. Then, applying Conjoinability to LL × RR leaves 652 combinations of a left and a right assignment as viable, i.e. 0,02 % of |LL × RR|. Only these 652 assignments are analyzed by the parser, which in this case finds a derivation for exactly one of them.
By the joined forces of independent occurrence checks and Conjoinability the number of assignments admitted to full parsing, is thus reduced to 1.2e–12 % of the original 5.3e16.

It is very difficult, if not impossible, to find a general metric for the effect of Conjoinability (9) on the set of admitted sequences: its effect - the ratio of pairs of left and right assignments that are rejected - is intrinsically dependent on type instantiations. Therefore, we can only give some more data to illustrate its effect. Table (16) contains, for some grammatical coordinated sentences over the same lexicon as was used for sentence (15), their length in words L, the number of possible assignments PA (= $\prod_1^n|A(w_i)|$), the product CP = |LL × RR|, the ratio CP/PA as a percentage, the number AA of assignments admitted to parsing, and the ratio AA/CP as a percentage. The latter ratio measures the effectiveness of applying Conjoinability. The ratio AA/PA gives the percentage of the total space of

possibilities that is transmitted to the parsing module; it stands for the effectiveness of the count preselection as a whole.
A comparable overview for some ungrammatical sentences is given in (17). The zero values mean that the system, in a stage prior to proper parsing, could not find any potentially derivable sequence; in the case of ungrammatical sentences this is, of course, a desirable result.

(16) *Pruning search space for grammatical coordinated sentences*

L	PA	CP	CP/PA %	AA	AA/CP %	AA/PA %
16	6.0e3	2e1	3.3e-1	2	1.0e1	3.3e–2
22	7.7e5	9.6e1	1.2e-2	4	4.2e0	5.2e–4
33	5.0e7	1.2e3	2.3e-3	8	6.9e–1	1.6e–5
39	7.4e7	5.4e3	7.3e-3	72	1.3e0	9.7e–5
44	4.4e8	2.2e4	5.1e-3	2	8.9e–3	4.5e–7

(17) *Pruning search space for ungrammatical coordinated sentences*

L	PA	CP	CP/PA %	AA	AA/CP %	AA/PA %
15	2.0e3	2.4e1	1.2e0	0	0	0
22	9.6e2	2.2e2	2.3e1	4	1.9e0	4.2e–1
33	1.5e6	2.1e3	1.4e–1	0	0	0
38	3.7e7	2.7e3	7.3e–3	24	8.8e–1	6.5e–5
47	2.6e8	6.7e4	2.6e–2	242	3.6e–1	9.2e–5

L: sentence length in words, including the coordinator. *PA*: the number of possible different assignments of lexical categories to the sentence of length L. *CP*: the cardinality of the cartesian product over the set of pre-checked assignments to the left hand side of the coordinator and the set of pre-checked assignments to the right of the coordinator, previously referred to as LL × RR. *AA*: the number of sequents admitted to the proper parsing procedure

From these figures one can conclude that the fraction AA/PA, which measures the effectiveness of the complete battery of count checks prior to parsing, including Conjoinability (9), tends to *decrease* as the search space PA, i.e. the number of lexically possible assignments, *increases.* This is also the tendency of AA/CP in figure (16): as CP increases - with PA - this ratio gets smaller, though not monotonically. The flaws in the tendency may be due to the ratio's being dependent on each and every detail in the type string. Nevertheless, we feel the tendency suggests that applying Conjoinability (9) is not only effective but also efficient: the time needed to bring about the reduction of CP to AA is only linearly dependent on CP, and the time for checking (8) and (9) at an individual sequent - demonstrated in (11) to (14) - is constant and given with the fixed number of basic types in a grammar.

5 Inevitability of Inequalities in Coordinative Count

In many cases, Conjoinability admits more assignments to the parser module than is necessary or desirable from a parsing point of view. The remaining redundancy is due to the fact that the Coordinative Count Invariant (8) is stated in terms of inequalities, rather than in terms of equalities. Count invariants in terms of inequalities, however, are the best we can get for a pruning instrument prior to parsing in the presence of coordination. To see why, consider the family of conjunctions

(18) Henk zei dat ik Agnes een boek **en**
Henk said that I Agnes a book and
(a) een tijdschrift had gegeven
a magazine had given
(b) de vrouw een tijdschrift had gegeven
the woman a magazine had given
(c) jij de vrouw een tijdschrift had gegeven
you the woman a magazine had given

Each of the right hand sides (18)a - c is a legitimate continuation of the given left hand side. Now take some assignment L to the left environment of the coordinator, i.e. to *Henk zei dat ik Agnes een boek*. From L alone one cannot make any predictions as to the nature of assignments R that are conjoinable. The only grammaticality condition imposed by L on R is that some proper prefix of R should reflect some proper suffix of L. Because coordination does not express itself functionally to the left nor to the right of the coordinator, we cannot tell which suffixes of L are available. In general, then, there will be more than one conceivable way of grammatically extending the string to the right of the coordinator. Consequently, several essentially different sequences R for (18)a - c have to be conjoinable with L. Nothing in L or its register imposes *a priori* occurrence conditions on assignments to possible extensions of the string to the right of a coordinator. The fact is that an assignment L with a fixed register reg_L may be conjoinable with many different Rs. Since these Rs all bear different registers, it is impossible to derive an nontrivial equality-condition on reg_R from reg_L - nor can it be done the other way around. One would have to look for a two-place function *f* such that *f(l,r)* is constant for some *l* while *r* varies: these functions will hardly be dependent on *r* in an interesting way. Thus, the Coordinative Count Invariant defines the margins for the candidate Rs as narrowly as possible, but has to leave room for variation caused by the functional indeterminacy of coordination.

6 Discussion

The above result does not offer a general method of reducing parsing complexity in the presence of coordination. Rather it shows that in parsing certain types of lexicon-driven categorial grammars the harmful mix of natural language's intrinsic ambiguity and the intrinsic indeterminacy of coordination can be tamed. Although every grammar will confront its parsers with this mix in one way or another, it is by no means clear that there is a general strategy to handle this source of computational complexity. This paper argues that there is an approach for certain categorial grammars, namely those for which the parameters of the Coordinative Count Invariant (8) can be set in a meaningful way. These grammars are certainly not structurally complete in the sense of [Buszkowski 1988], but may have mildly context-sensitive capacity [Joshi *et al.* 1991]. To the categorial grammars in the scope of the present approach belong at least those dubbed 'parenthesis-free' in [Friedman *et al.* 1986]. For other (categorial) grammars, there may be no, or a completely different, solution to the explosion of search space in the presence of coordination.

Moreover, up to now the Coordinative Count Invariant (8) is only consolidated for sentences containing a single coordinator. Handling multiple coordination in the same spirit will be even more tedious, but there is no reason to believe that (8) could not be generalized. The discussion in 4 suggests that the set of conditions resulting from generalizing (8) will be weaker than the present set of inequalities.

References

[Van Benthem 1986] Van Benthem, J. (1986), *Essays in Logical Semantics*, Reidel Pub Cy

[Van Benthem 1991] Van Benthem, J. (1991), *Language in Action*, SLFM 130, North-Holland

[Buszkowski 1988] Buszkowski, W. (1988), 'Generative Power of Categorial Grammars', in R. Oehrle, E. Bach and D. Wheeler (eds), *Categorial Grammars and Natural Language Structures*, Reidel, p. 69-94

[Cremers 1989] Cremers, C. (1989), 'Over een lineaire kategoriale ontleder', *TABU 19:2*, p. 76-86

[Cremers 1993] Cremers, C. (1993), *On Parsing Coordination Categorially*, HIL diss 5, Leiden University, also available as http://fonetiek-4.leidenuniv.nl/pub/cremers/dissi.ps

[Emms 1993] Emms, M. (1993), 'Parsing with Polymorphism', *Proceedings Sixth Conference of the European Chapter of the ACL*, ACL, p. 120 - 129

[Grootveld 1994] Grootveld, M. (1994), *Parsing Coordination Generatively*, HIL diss 7, Leiden University

[Friedman *et al.* 1986] Friedman, J., D. Dai, W. Wang (1986), 'The weak generative capacity of parenthesis-free categorial grammars', *Proceedings of Coling 86*, ACL, pp. 199 - 201

[Joshi *et al.* 1991] Joshi, A., K. Vijai-Shanker and D. Weir (1991), 'The Convergence of Mildly Context-Sensitive Grammar Formalisms', in P. Sells, S. Shieber and T. Wasow (eds), *Foundational Issues in Natural Language Processing*, MIT Press, p. 31-82

[Moortgat 1988] Moortgat, M. (1988), *Categorial Investigations*, GRASS 9, Foris

[Roorda 1992] Roorda, D. (1992), 'Proof Nets for Lambek calculus', *in*: A. Lecomte (ed), *Word Order in Categorial Grammar*, Ed. Adosa, p. 149-171

[Sag *et al.* 1985] Sag, I.A., G. Gazdar, T. Wasow and S. Weichler (1985), 'Coordination and How to Distinguish Categories', *Natural Language and Linguistic Theory 3:2*, p. 117-171

[Steedman 1990] Steedman, M. (1990), 'Gapping as Constituent Coordination', *Linguistics and Philosophy 13:2*, p. 207-263

[Wittenburg 1986] Wittenburg, K.B. (1986), *Natural Language Parsing with Combinatory Categorial Grammar in a Graph-Unification Based Formalism*, Ph.D. diss., University of Texas at Austin