

An introduction to mildly context-sensitive grammar formalisms

Gerhard Jäger and Jens Michaelis
`{jaeger,michael}@ling.uni-potsdam.de`

Reader

Contents

- Huybregts, Riny (1985), The weak inadequacy of Context Free Phrase Structure Grammars. In: Ger Jan de Haan, Mieke Trommelen and Wim Zonneveld (eds.), *Van Periferie naar Kern*, Dordrecht: Foris Publications, 81-99.
reproduced with kind permission of the author
- Joshi, Aravind, K. Vijay-Shanker and David Weir (1991), The Convergence of Mildly Context-Sensitive Grammar Formalisms, In: Peter Sells, Stuart Shieber and Tom Wasow (eds.), *Processing of Linguistic Structure*, MIT Press, Cambridge (Mass.), 31-81.
reproduced with kind permission of The MIT Press, this electronic version is only for personal use and not for distribution without the consent of the Publisher
- Steedman, Mark and Jason Baldridge (2003), Combinatory Categorical Grammar, manuscript, University of Edinburgh
reproduced with kind permission of the authors
- Joshi, Aravind (2004), Starting with Complex Primitives Pays Off: Complicate Locally, Simplify Globally, manuscript, University of Pennsylvania, to appear in *Cognitive Science*, 2004.
reproduced with kind permission of the author

THE WEAK INADEQUACY OF CONTEXT-FREE PHRASE STRUCTURE GRAMMARS

Riny Huybregts

From:

Ger J. de Haan, Mieke Trommelen
and Wim Zonneveld (eds), *Van
Periferie naar Kern*, pp. 81-99. Foris,
Dordrecht.

1. *Natural Languages*

Ever since the beginning of generative grammar the question of what exactly constitutes a natural language (*NL*) has been a vexed one. At present, there is considerable controversy over (and misunderstanding of) the issue of what constitutes the appropriate dimension along which to evaluate the class of formal devices capable of generating all and only *NL*.¹ Even restriction to aspects of weak generative capacity has produced various contrary claims on the nature of *NL* conceived of as string sets. Two of these are particularly interesting: (i) the claim that so far *NL* have not been shown to exceed the power of context-free grammar (*cf*-grammar), expounded by Pullum and Gazdar (1982), and (ii) the claim due to Langendoen and Postal (1983) that *NL* are infinitely beyond the capacity of *unrestricted rewriting systems*. These mutually inconsistent claims have each been argued to have theoretical (the non-necessity vs. non-sufficiency of *TGG*) as well as philosophical (*Performancist* vs. *Platonist* approaches to language) implications for the study of grammar.²

The sufficiency of the theory of context-free generalized phrase-structure grammar (*CF-GPS-Grammar*) to adequately characterize both the string sets and the structure sets of *NL* has been construed as an argument for the non-necessity of the more powerful theory of transformational generative grammar (*TG-Grammar*). In general, *TG-Grammar* would suffer from excessive descriptive latitude. Even the context-free based locally filtering transformational languages of Peters and Ritchie (1973) include some non-recursive languages. Since the theory of *GPS-Grammar* permits the generation of only *cf*-languages, and since any *cf*-language has a low polynomial recognition time complexity, *GPS*-grammars provide a first approximation to what seems to be an explanation of how humans parse sentences.³ Apparently, the theory of *GPS-Grammar* is not only a highly restrictive theory of *NL* with well-studied mathematical properties and notational devices capable of capturing significant syntactic generalizations, but in addition has some relevance to the study of natural language processing by virtue of its computational tractability as well as to learnability theory by virtue of the restriction it imposes on the class

of learnable languages.⁴

The claim that *NL* are not even recursively enumerable points to a non-constructive approach to *NL*.⁵ In general, *NL* would be too large to be specified by the constructive, derivational, approach of any generative grammar. Instead, the nondenumerably infinite string sets of *NL* can only be dealt with in the nonconstructive, non-proof-theoretic, manner of nonfinitistic systems. The nonconstructive grammar (*NCG*) approach has been linked, as a matter of fact, to a *Platonic* conception of linguistics but must be so linked as a matter of logic if *NCG* turns out not to be finitizable.

Both claims can be seriously questioned. *NL* badly need extra *cf*-power: they provably incorporate infinite string sets with no upper bound on the number of cross-serial dependencies. Efficient parsability can therefore not be the privilege of *cf*-languages, a point that has been argued independently in a slightly different context by Berwick and Weinberg (1982). Also, *NL* need not be nondenumerably infinite. In this article we will be concerned exclusively with the putative non-necessity of *TGG* and the relevance of *GPSG* for *NL*. In Huybregts (in preparation) we also take up the question of the putative insufficiency of *TGG* for the proper characterization of *NL*. There we argue that Langendoen and Postal's Closure Principle begs the question of transfinite sentence length. As a result, there remains no reason to question the superiority of the derivational conception of generative grammar as the more restrictive, i.e. the more falsifiable and hence better, theory of language. At the present state of our knowledge, there seems to be no ground for counter-arguing the position that the issue of *Platonic Realism vs. Cartesian Rationalism* is independent of that of *Constructive vs. Nonconstructive Grammar*.⁶ Still, in view of the possibly non-finite axiomatizability of *NCG* Chomsky's linguistic conceptualism seems much harder to reconcile with Langendoen and Postal's more "homogeneous" but less restrictive, hence worse, theory of *NCG* than is Katz's linguistic Platonism. A result that hopefully turns out not to be inconsequential for the empirical study of *NL*-grammars.

2. Hierarchies of languages

Let us take as a point of departure the *Chomsky-hierarchy* of formal languages.

- (1) $RE-lgs \supset CS-lgs \supset CF-lgs \supset R-lgs$

According to this hierarchy the class of context-sensitive (*cs*) languages properly includes the class of context-free languages and is properly contained

in the class of recursively enumerable (*re*) languages. Any *cs*-language will be recursive as may be some, but not all, recursively enumerable non-*cs* languages. The regular (*r*) languages properly contain the class of finite languages (not represented in (1)), and are a proper subset of the *cf*-languages themselves. These proper inclusions can be represented graphically as a series of concentric circles with the class of finite languages as the most interior circle and the *re*-languages as the most inclusive one. *NL* are obviously nonfinite and can therefore not be characterized by finite sets. If we adopt the *Chomsky-hierarchy* as a possible hierarchy on which to locate the class of *NL*, it will be evident that finite sets must be excluded from the succession of concentric circles. (Irrespective of the 'epiphenomenal' status of natural languages, any generative grammar will specify a class of objects, called "language", and we may legitimately investigate the properties of this 'derivative' notion.)⁷ As a result we derive the concentric doughnut-hierarchy with the class of finite languages constituting the hole of the doughnut.⁸

The hole in the doughnut has not been explained so far. We offer the following explanation. Assume some version of the $\bar{X}$ -theory as a proper partial characterization of the phrase system of *NL*. In particular, let us assume that every maximal projection is the projection of a unique lexical head. Thus we have (2) yielding (3) but not (4).

- (2) $\bar{X} \longrightarrow \phi \ X \ \psi$
 (3) $\dots X \dots \bar{Y} \dots$
 (4) $\ast \dots X \dots Y \dots$

If lexical heads take complements, these complements must themselves be maximal projections, since uniqueness of head will be contradicted otherwise. Now recursion results if we make the further simplifying assumption that $\bar{X}$ -theory is maximally weak, i.e. does not restrict the choice of complements with respect to heads (this being a lexical rather than structural property). This situation is represented in (5).

- (5) (a) $\bar{X} \longrightarrow \dots X \dots \bar{Y}$
 (b) $\bar{Y} \longrightarrow \dots Y \dots \bar{X} \dots$

Apparently, the only stipulation we need to explain nonfiniteness of *NL* is the simplifying assumption of an optimally impoverished $\bar{X}$ -theory.

We now have the following possibilities. The class of *NL* may be a concentric doughnut properly containing the *cf*-languages and properly contained in the *cs*-languages. Or the class of *NL* may cut across the doughnut-hierarchy as does the class of languages generated by Peters and Ritchie's (1973) *cf*-based local-

ly filtering transformational grammars. The latter class of languages is incomparable with any class of languages that is reconstructible as a concentric circle in the doughnut-hierarchy since it includes some *cs*-languages (e.g. $L = \{ a^n \mid n = 2^m, m \geq 0 \}$) as well as some nonrecursive languages but it also excludes other *cs*-languages (e.g. $L' = \{ a^n \mid n = 2^{2^m}, m \geq 0 \}$). Likewise the class of *NL* may turn out to cut across (or even intersect with) the classes of languages on the *Chomsky-hierarchy*.

NL are clearly not regular (see Chomsky, 1956) but can they be context-free? Significantly, only a few arguments have been advanced in support of the non-context-freeness of some *NL*. Equally significantly, Pullum and Gazdar (1982), in a general survey of these arguments, claim to have shown that all of these attempts (particularly Postal (1964), Huybregts (1976), and Langendoen (1977)) to establish non-context-freeness of *NL* are formally or empirically invalid. Consequently, *NL* may be context-free after all and *CF-PS*-Grammar or some limited generalization thereof, *CF-GPS*-Grammar, may still be capable of weakly characterizing *NL*. (In addition, Gazdar (1981) argues that *GPS*-Grammar provides an adequate characterization of the structural properties of *NL* and therefore *NL* are also strongly generable by *CF-GPS*-grammars.) The claim, represented in (6), that the thesis of the context-freeness of *NL* has not been counterexemplified will be our main concern.

(6) *NL* can be *cf*-languages

This proposition is provably false. See the argument presented in section 3. As a result *CF-PS*-Grammar is too restrictive in weak generative capacity and *a fortiori* fails in strong generative capacity. No *CF-PS*-grammar can either weakly or strongly characterize *NL*. (A 'generalized' version of some *cf*-language inducing theory of grammar must at least have the generative capacity of the class of *Indexed Grammars* that characterizes the class of *indexed languages* properly containing the *cf*-languages and properly contained in the *cs*-languages.)⁹

There are also languages that fall outside the *Chomsky-hierarchy*. Any language on this hierarchy, i.e. any generative language, is countable : either finite or denumerably infinite. In the latter case the cardinality of the language is of the same order of magnitude as the set of integers and its sentences can therefore be effectively listed. However, there are also non-denumerably infinite languages with a cardinality of the order of magnitude of that of the set of reals or even more. These can not be effectively listed. As an immediate consequence of Gödel's (1931) Incompleteness Theorem, these

non-recursively enumerable language systems can not be *consistent* without being *incomplete*.¹⁰ We will, therefore, call these languages "incomplete languages" (*ic*-languages), since being nonrecursively enumerable they can only be dealt with incompletely if at all. A strictly *ic*-language is for example a language that contains some subset of sentences of infinite length (*transfinite* sentences). No doubt, infinite sentence length is not likely to empirically manifest itself. It might plausibly be argued that transfinite objects are no part of the physical world.¹¹ Therefore the claim that there is no size law on the length of *NL*-sentences must be based on considerations of theoretical simplicity, itself an empirical hypothesis. Chomsky (1956) rejected any finite size restriction, arguing that any such size law would "prohibitively" complicate the statement of the grammar.¹² Any rejection of recursion will force an increase of the nonterminal vocabulary proportional to the upper sentence length. Generative grammars thus characterize infinite languages with sentences of only finite length. There is no upper bound on sentence length, hence the class of generable sentences is countably infinite, but each sentence is a finite formal object.

Langendoen and Postal (1983) argue, analogously, that any size law on sentence length entails loss of generality. In fact, any argument against a specific finite size law (like Chomsky, 1956) is equally an argument against any size law at all, in particular against the transfinite size law imposed on sentences specified by generative grammars. Once it is accepted that there is no size law on *NL*-sentences, an analogue of Cantor's paradox may be developed in the domain of sentence coordination on the assumption that *NL* are closed under coordinate compounding. Apparently none too innocuous an assumption. In fact Langendoen and Postal (1983) argue on the basis of a Cantorian analogy that *NL* have at least the magnitude of the class of reals and consequently *NL* are not countable. What is even more, *NL* have no determinable magnitude. The claim, due to Langendoen and Postal, that *NL* have an indeterminable magnitude is embodied in (7).¹³

(7) *NL* are *megacollections*

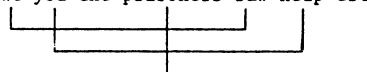
In Huybregts (in preparation) we argue that this proposition is provably arbitrary. Consequently, the concept of generative grammar can be maintained, and the spectre of infinitistic systems that deal with nonrecursively enumerable collections in an incomplete manner can be aborted. Proof-theoretic systems such as generative grammars suffice to specify adequately the class of *NL* in a constructive, complete and consistent way.

In the next section we will question the content of the claim embodied in (6). We will argue that *NL* are too rich to be even weakly generable by any *cf*-grammar. In contrast cross-serial dependencies pose no serious problems to the theory of *TGG*.

3. The non-sufficiency of *cf*-grammars

In Huybregts (1976) it was argued on the basis of verb-raising constructions in Dutch like (8) that *NL* must be at least context-sensitive.

- (8) ... dat we jou de gevangenen zagen helpen ontkomen
that we you the prisoners saw help escape



We argued that there could be no upper bound on the number of overlapping dependencies an arbitrary sentence can take. Dutch, then, contains an infinite subset of grammatical sentences with unbounded cross-serial dependencies. We next presented a formal proof that Dutch is non-context-free by a provably non-context-free intersection with a regular set. In stating the formal argument we made use of Bar-Hillel *et al*'s (1961) pumping lemma on the string sets of *cf*-languages (9) as well as of some well-known closure properties of the class of *cf*-languages (10) and (11).¹⁴

(9) Characterization Theorem of *CF*-Languages

Let *L* be any *cf*-language. Then there exist constants *p* and *q* depending on *L* only, such that if *z* ∈ *L* and |*z*| > *p*, then *z* may be written as *z* = *uvwx*^{*i*}*y*, where |*vwx*| ≤ *q* and *vx* ≠ *ε*, and for all *i* ≥ 0, *uv*^{*i*}*w**x*^{*i*}*y* is in *L*.

(10) Closure under Intersection

The class of *cf*-languages is closed under intersection with a regular set.

(11) Closure under Homomorphism

The classes of *r*-languages and *cf*-languages are closed under finite substitution and homomorphism.

Intersection of an appropriately chosen infinite regular set *D_r* with the full Dutch language *D* will yield some infinite set *D_{cs}* of sentences with infinitely many cross-serial dependencies, i.e. *D_r* ∩ *D* = *D_{cs}*. This set *D_{cs}* can be mapped under homomorphism into the language $\Omega = \{ \omega \omega \mid \omega \in \{a, b\}^* \}$, i.e. $h(D_{cs}) = \Omega$. Since properties of *cf*-languages are preserved under homomorphisms if Ω can not be context-free, then neither can *D_{cs}* (theorem 11). But Ω is provably non-context-free: intersecting Ω with the *r*-language $L_r = \{ a^n b^m a^p b^q \mid n, m, p, q > 0 \}$ gives the language $L_{cs} = \{ a^n b^m a^n b^m \mid n, m > 0 \}$, i.e.

$L_r \cap \Omega = L_{cs}$. But *L_{cs}* is provably non-context-free.¹⁵ An immediate consequence of applying theorem (9) to *L_{cs}*. But if *L_{cs}* is not context-free, then neither is Ω (theorem 10). Therefore we derive the result that that *D_{cs}* can not be context-free since Ω can not be context-free (theorem 11). But if *D_{cs}* is not context-free, then application of theorem (10) to the intersection of *D_r* with *D* yields the result that *D* can not be context-free since neither can *D_{cs}*. Hence Dutch is provably non-context-free. Δ

However, Pullum and Gazdar (1982) argue that "no argument of the sort Huybregts advances can in fact succeed." They base their claim on the observation that the putative crossing dependencies were not formally encoded in the string set as observed incidentally in Huybregts (1976, fn. 37). For some appropriately selected set *D_{cs}*, apparently, grammaticality is sufficiently guaranteed by any sequence of NPs followed by a matching number of Vs. Clearly, any such string set can be weakly generated by some *cf*-grammar as noted in Huybregts (1976). The argument presented there can be interpreted as establishing that cross-serial dependencies in Dutch are properties not of the string set but, instead, of the structure set of Dutch. Consequently, even though Dutch is a *cf*-language, the structure set of Dutch poses only obvious problems for a context-free compositional semantics as noted in Pullum and Gazdar. In fact it remains a widely open question whether some *cf*-language inducing theory of grammar can in principle be capable of yielding an adequate grammar of Dutch. E.g. Bresnan *et al* (1982), while conceding that Dutch may be context-free stringwise, argue that Dutch can be strongly characterized by no *cf*-grammar. To derive this result, they first apply Thatcher's (1967, 1973) pumping lemma on the structure sets of *cf*-languages to Dutch.

(12) Characterization Theorem of *CF*-Languages

For any *cf*-grammar *G*, the derivation trees *T*(*G*) are recognizable by some frontier-to-root finite state tree automaton.

Application of Thatcher's characterization theorem to the set of Dutch verb-raising (*VR*) constructions show these unrecognizable.¹⁶ Next, these *VR*-constructions are shown to be adequately characterizable by Bresnan's theory of Lexical-Functional Phrase-Structure Grammar (*LF-PS*-Grammar). We will not take issue with the conclusion arrived at in Bresnan *et al* (although their argument is seriously invalidated) to the effect that Dutch can not be strongly context-free.

Instead we will argue that Dutch can not even be weakly context-free. *A fortiori* Dutch can not be strongly context-free. Hence we derive Bresnan *et al*'s result as a corollary of a stronger theorem. By virtue of this theorem, which will be completed below, we must accept their conclusion just as we must accept the conclusion of Huybregts (1976) without having to accept, however, their argument.¹⁷

The argument from the existence of infinitely many overlapping dependencies to the non-context-freeness of Dutch depended, as noted, on the explicitly stated assumption that selectional restrictions were syntactic in nature. However, since Pullum and Gazdar authoritatively state that selectional restrictions are "uncontroversially" non-syntactic, we propose to simply skip the discussion about the true status of these restrictions.¹⁸ Instead we ask if these restrictions, whatever their nature, can be 'morphologized'. Do they possess the property of being overtly representable in the string set of Dutch? If this question can be answered in the positive, we have a decisive argument against Pullum and Gazdar as well as Bresnan *et al.*

To start, let us consider the *VR*-triggers *zien* ('see'), *helpen* ('help') and *laten* ('let') as well as the intransitive verb *drijven* ('float'). *Zien* takes animate subjects, *helpen* selects both animate and inanimate subjects just as *laten* en *drijven*, but *helpen* selects animate objects only. What is of special interest is that there is a class of pronominals that is differently morphologized on the animate-inanimate dimension. The pronominals of (13a) are inanimate and their gender is neuter (e.g. *dat alles* ('that all'), *het niets* ('the nothing'), *iets dat* ('something which')). In contrast, the pronominals of (13b) are all animate and have nonneuter gender (e.g. *die iemand* ('that someone'), *de velen* ('the many'), *niemand die* ('nobody who')).

- (13) (a) *iets, niets, alles, veel, weinig*
something, nothing, everything, much, little
(b) *iemand, niemand, allen, velen, weinigen*
someone, no-one, everyone, many, few

Clearly it makes no sense to say that (14) is just perfect under the assumption that there happen to be a lot of people, *velen*, called 'veel', the same way Pullum and Gazdar explain the grammaticality of (15) on the assumption that there are a language and a person called 'Pieter' and 'Arabisch' respectively.

- (14) * *Jan hielp veel nauwelijks*
Jan helped much hardly
('Jan hardly helped much')
(15) ...*dat Jan Marie Arabisch Pieter wil laten zien schrijven*
...that Jan Marie Arabic Pieter will let see write
('...that Jan will let Marie see Arabic write Pieter')

Substitution of *velen* ('many') for *veel* in (14) yields a perfect sentence *Jan hielp velen nauwelijks*.

Let us now develop the argument. *VR* is obviously a recursive process in

the language as shown by (16). These sentences can be analyzed to offer a classic A-B-A proof of the cyclic character of *VR*.¹⁹

- (16) (a) ...*dat we jou de gevangenen helpen laten ontsnappen*
...that we you the prisoners help let escape
('that we help you let the prisoners escape')
(b) ...*dat we jou de gevangenen laten helpen ontsnappen*
...that we you the prisoners let help escape
('that we let you help the prisoners escape')

Given unbounded *VR*, the existence of selectional properties that are morphologically representable in the string set of *VR*-sentences entails that Dutch can not even be *weakly* context-free. Consider (17).

- (17) ...*dat velen iets iemand alles zagen helpen laten ontploffen*
...that many something someone everything saw help let explode
('that many saw something help someone let everything explode')
(Cf. ...*dat vele mensen verscheidene ouders hun kinderen oude*
vrouwjes hebben zien helpen laten oversteken
...that many people have seen several parents help their
children let little old women cross (the street))

While hardly felicitous, sentences like (17) must count grammatical. However, any pairwise permutation of the animate and inanimate pronominals of (17) produces an ungrammatical outcome. If accepted, the argument suffices to establish the conclusion that Dutch is beyond the weak generative capacity of context-free grammars.

Still one might feel some reservations although we feel there is no rational basis for them. Therefore it would be intrinsically interesting e.g. if we were to find other instantiations of the morphological expression of cross-serial dependencies like the availability of overt morphological case. Unfortunately, Modern Dutch only has a nominative vs. non-nominative case marking. Interestingly, Middle Dutch still had a productive dative vs. accusative distinction. It also had *VR*. Since *VR*-trigger *doen* ('make') assigns accusative case to the subject NP of its complement clause under conditions of case-government, and *VR*-trigger *helpen* ('help') assigns dative case to its object, we have the beginnings of a new argument.²⁰ We will not flesh out this argument here. Instead we will look for a modern language with the arbitrarily deep cross-serial dependencies of Modern Dutch *VR*-constructions but with the morphological case properties of Middle Dutch. Does any such language actually exist? It most certainly does. Swiss German is precisely such a language.

First Swiss German or *Züritütsch* (ZT) will be shown to possess the relevant case properties. Next we will discuss the very interesting *VR*-phenomena in the sentential complement system of Swiss German.²¹ Consider then the

following sentences of *Züritütsch* with their Standard German counterparts.

- (18)(a)(i) *De Joggel hät em vatter (em chind) ghulffen*
Jockel has the father (the child) helped
('Jockel has helped the father (the child)')
- (ii) *De Joggel hät de vatter (s chind) gsee*
Jockel has the father (the child) seen
('Jockel has seen the father (the child)')
- (b)(i) *Jockel hat dem Vater (dem Kind) geholfen*
(ii) *Jockel hat den Vater (das Kind) gesehen*

These sentences unequivocally show case-assignment to nouns whose grammatical gender is masculine (*vatter, Vater*) and neuter (*chind, Kind*): *hälffe* (*helfen*) assigns dative case and *gsee* (*sehen*) assigns accusative case which is isomorphic with nominative case in the examples above.

Züritütsch also exhibits the second relevant property of VR. This is clear from (19). As before the German equivalents are given for comparison.

- (19)(a)(i) ...*wil de Fritz em vatter s bschteck hilft abtröchne*
...because Fritz the father the forks and spoons helps dry
('because Fritz helps father dry the forks and spoons')
- (ii) ...*wil de vatter syni chind medizyn hät wele laa studiere*
because the father his kids medicine has wanted let study
('because the father wanted to let his kids read medicine')
- (b)(i) ...*weil Fritz dem Vater das Besteck abtrocknen hilft*
(ii) ...*weil der Vater seine Kinder Medizin studieren lassen wollte*

Observe the asymmetry in the operation of VR in *Züritütsch* and German: the order of verbs in the *Verbalfeld* of Standard German is the inverse of that of Swiss German, which is identical to the Dutch order of verbs.²² We conclude then that *Züritütsch* has all relevant properties to develop the argument to the non-context-freeness of NL.

Now compare (20a) with the German equivalent (20b).

- (20)(a)(i) ...*wil mer de maa lönd schwüme*
because we the man let swim
('because we let the man swim')
- (ii) ...*wil mer em chind hälfed schwüme*
...because we the child help swim
('because we help the child swim')
- (b)(i) ...*weil wir den Mann schwimmen lassen*
(ii) ...*weil wir dem Kind schwimmen helfen*

These sentences clearly illustrate case dependencies formally encoded in the string set of *Züritütsch*: *laa* (*lassen*) is an accusative case assigning verb, while *hälffe* (*helfen*) assigns dative case. Since VR is a cyclic process, we obtain increasingly complex sentences with crossing dependencies, viz. case

dependencies, formally encoded in the surface strings. See (21).

- (21)(a)(i) ...*wil mer de maa em chind lönd hälffe schwüme*
because we the man the child let help swim

('because we let the man help the child swim')

- (ii) ...*wil mer em maa s chind hälfed laa schwüme*
because we the man the child help let swim

('because we help the man let the child swim')

- (b)(i) ...*weil wir den Mann dem Kind schwimmen helfen lassen*

- (ii) ...*weil wir dem Mann das Kind schwimmen lassen helfen*

Here the crossing lines stand for case dependencies rather than the grammatical relations involved in the selectional dependencies of Dutch. As in Dutch we obtain infinitely many cross-serial dependencies by virtue of the recursive properties of VR. This will be clear from (21), where (i) differs from (ii) only in the ordering of VR-triggers, which means that VR must by all means be cyclic. Swiss German differs from Dutch in that not only the verb, but also some or all of its complement structure can be raised into the next higher clause, a property it shares with *Flemish Dutch*. Apparently VR must be parametrized with respect to the constituent size of the affected terms: V only in German and increasingly large "small clauses" in Dutch *Particle-V* and *Flemish Dutch* and *Züritütsch* (VP or parts thereof).²³ Consequently, (19a) and (21a) have alternative realizations in *Züritütsch*. These are given in (22) through (25) where the "Dutch only" variants of (19) and (21) are repeated.

- (22)(a) *wil de Fritz em vatter s bschteck hilft abtröchne*
(b) *wil de Fritz em vatter hilft s bschteck abtröchne*
(c) **wil de Fritz hilft em vatter s bschteck abtröchne*
- (23)(a) *wil de vatter syni chind medizyn hät wele laa studiere*
(b) *wil de vatter syni chind hät medizyn wele laa studiere*
(c) *wil de vatter syni chind hät wele medizyn laa studiere*
(d) *wil de vatter syni chind hät wele laa medizyn studiere*
(e) *wil de vatter hät syni chind medizyn wele laa studiere*
(f) *wil de vatter hät syni chind wele medizyn laa studiere*
(g) *wil de vatter hät syni chind wele laa medizyn studiere*
(h) *wil de vatter hät wele syni chind medizyn laa studiere*
(i) *wil de vatter hät wele syni chind laa medizyn studiere*
(j) **wil de vatter hät wele laa syni chind medizyn studiere*

The ungrammaticality of (22c) and (23j) is due to the illicit raising of the whole sentential complement. Recall that only VPs or "small VPs" can be legi-

timately raised under VR.

- (24) (a) *wil mer de maa em chind lönd hälffe schwüme*
 (b) *wil mer de maa lönd em chind hälffe schwüme*
 (c) **wil mer de maa lönd hälffe em chind schwüme*
 (d) **wil mer lönd de maa em chind hälffe schwüme*
 (e) **wil mer lönd de maa hälffe em chind schwüme*
 (f) **wil mer lönd hälffe de maa em chind schwüme*
- (25) (a) *wil mer em maa s chind hälfed laa schwüme*
 (b) *wil mer em maa hälfed s chind laa schwüme*
 (c) **wil mer em maa hälfed laa s chind schwüme*
 (d) **wil mer hälfed em maa s chind laa schwüme*
 (e) **wil mer hälfed em maa laa s chind schwüme*
 (f) **wil mer hälfed laa em maa s chind schwüme*

Again the ungrammaticality of (24c-f) and (25c-f) can be explained as the result of illicit raising of *S*, and ultimately as a violation of Case Theory.²⁴ For a detailed discussion of VR in Germanic languages and its non-trivial implications we refer the reader to Huybregts (forthcoming).

Although the variable realizations of VR-processes neutralize an (infinite) number of cross-serial dependencies, this in no way affects our conclusion that *Zürichdeutsch* can not be a cf-language.

Let us finally state the formal proof. We first define the regular language ZT_r on some restricted subalphabets of *Zürichdeutsch*. Let ZT_r specifically be the set product of two regular sets, $ZT_{r\phi}$ · $ZT_{r\psi}$, defined in (26).

$$(26) \quad ZT_r = \{ \phi^* \psi^* \mid \phi \in ZT_{r\phi} \text{ \& } \psi \in ZT_{r\psi} \},$$

where $ZT_{r\phi} = \{ \text{ich, de/em maa, s/em chind} \}^*$
 $ZT_{r\psi} = \{ \text{ha, laa, hälffe, schwüme} \}^*$

Since regular sets form a Boolean algebra, closed under product, ZT_r as the concatenation of the regular sets $ZT_{r\phi}$ and $ZT_{r\psi}$ must itself be regular. ZT_r thus defines a set of strings each of which consists of a sequence of (pro)-nominals followed by a sequence of verbs. Intersecting ZT_r with the complete Swiss German language *ZT* yields an infinite set of strings with no upper bound on the number of cross-serial morphological case dependencies. Note that by appropriate selection of ZT_r we bypass the difficulty noted with respect to (22) through (25). Specifically, the intersection language ZT_{cs} of (27) contains nothing but strings with crossing dependencies.

$$(27) \quad ZT_{cs} = \{ a\omega b'c \mid \omega \in \{ \text{de/em maa, s/em chind} \}^* \text{ \& } \\ \omega'c \in \{ \text{laa, hälffe} \}^* \text{ \& } a = \text{ich} \text{ \& } \\ b = \text{ha} \text{ \& } c = \text{schwüme} \}$$

Since ZT_{cs} can be mapped under homomorphism $h(ZT_{cs})$ into the provably non-context-free language $\Omega = \{ \omega\omega \mid \omega \in \{ a, b \}^* \}$, it can not be context-free.²⁵ Note that in ZT_{cs} every i^{th} term is case-dependent on the $(n+i)^{\text{th}}$ term, where $m = 2n+1$ and $|a\omega b'c| = m$.

$$(28) \quad h(ZT_{cs}) = \Omega, \text{ where}$$

$h(\text{ich}) = e,$	$h(\text{de maa}) = a,$
$h(\text{ha}) = e,$	$h(\text{s chind}) = a,$
$h(\text{schwüme}) = e,$	$h(\text{em maa}) = b,$
$h(\text{laa}) = a,$	$h(\text{em chind}) = b.$
$h(\text{hälffe}) = b,$	

Recall that the closure properties of cf-languages are preserved under the operation of homomorphism (theorem 11). Applying this theorem to the value of the homomorphism that takes ZT_{cs} into Ω gives the result that since Ω is not context-free, neither can ZT_{cs} be context-free. Finally, by applying theorem (10) to the intersection of ZT_r with *ZT* we derive the result that *ZT* is context-free only if its intersection with ZT_r is context-free. But ZT_{cs} is not context-free. Therefore *ZT* is not context-free. Δ

Note that the logic of the argument to the non-context-freeness of the *Zürichdeutsch* language is identical to that we gave earlier for Dutch. We only have to substitute *ZT* for *D* in argument following (9)-(11). Since *ZT* is just an arbitrary NL, we can generalize this result to a stronger theorem.

(29) *Theorem of Non-Context-Freeness of NL*
 NL can not be context-free

Now since the class of cf-grammars is provably too weak to even generate the string set of *ZT*, hence the string sets of NL in general, it immediately follows that cf-grammars fail to strongly characterize NL. In other words we trivially derive Bresnan *et al*'s result as a corollary of the stronger theorem (29).

4. Conclusion

In the present article two propositions have been offered:

- (i) NL must be beyond the weak (hence strong) generative capacity of context-free grammar,
- (ii) NL need not be beyond the capacity of generative grammar.

We presented a demonstrable proof that NL can not be context-free as a rule. Proposition (i) therefore has the status of a *theorem* and assuming infinite-cardinality for NL, it is an impregnable mathematical truth about NL. Context-freeness is thus a valid lower bound on the Chomsky-hierarchy for the class of

NL. Consequently, the theory of CF-PS-Grammar can not be a sufficient system for dealing with NL since it must fail to even recursively enumerate the string sets of NL. Its failure in strong generative capacity immediately follows as a corollary of the stronger theorem (29): no cf-grammar can strongly characterize the relevant structure sets of NL. We suggested that in contrast TGG can deal effectively with VR-effects in a linguistically significant manner.²⁶

Although CF-PS-Grammar is provably inadequate as a theory of NL (unless "generalized" beyond cf-capacity, in which case the term GPS-Grammar is a misnomer since any such extension must incorporate the full derivational possibilities of some hyperclass of cf-grammars that no longer induces only context-free languages), there do not seem to be necessary grounds for extending the theory of grammar beyond generative, constructive, systems. The availability of non-constructive, incomplete, systems is not provably necessary.

We did not discuss these nonfinitistic, non-constructive, approaches to NL (now viewed as megacollections), but suggested that Langendoen and Postal's closure principle for coordinate compounding of sentential structures is only one of several ways of dealing with coordination and not necessarily the most highly valued one by standards of empirical significance for linguistic theories.²⁷ Since there is no other "evidence" for transfinite sentence length other than general considerations of simplicity (*Ockham's razor*) which so far as I see do not bear on questions of linguistic simplicity construed as an empirical hypothesis, we propose that non-constructive grammars are not necessary systems for dealing with NL.

One conclusion to draw from the previous discussion is that PS-Grammar if it is to survive as a descriptively adequate system for NL must be extended beyond context-free capacity. Precisely to what extent I do not know. However, we know that languages such as $\{\omega\omega\}$ and $\{\omega\omega\}$, which figured in the proof-theoretic derivation of theorem (29) are recognizable by one way nondeterministic stack automata (INSA).²⁸ Since the nondeterministic one way stack automata are properly included in the class of *Indexed Grammars* there must be indexed grammars that generate these languages.²⁹ Consequently, there remains the possibility that NL can be weakly specified by systems whose generative capacity is that of indexed grammars (or maybe intermediate between that of indexed and context-free grammars) with no need to invoke the full power of context-sensitive grammars, which are too rich for dealing with NL. Nevertheless it would be surprising if any such "generalization" of the class of CF-PS-Grammars to that of indexed grammars turned out to be linguistically significant. Indexed grammars while demonstrably sufficient to weakly character-

ize Dutch and Swiss German are likely to be too powerful to capture the structural subtleties of these language systems in a linguistically significant way. This suspicion is based on the fact that indexed grammars not only handle the "reasonable" context-sensitive Languages $L = \{\omega\omega\}$ or $L' = \{a^n b^n c^n\}$ but are rich enough to generate "unreasonable" context-sensitive languages such as $L = a^{2^n}$ or $L' = a^{n^2}$, which possess arithmetical properties of a polynomial or exponential kind that are absent from NL languages. There are at present no intuitive or formal grounds for accepting the view that phrase structure systems ("generalized" or not) will be successful in strongly characterizing cross-serial dependencies of the type discussed here.

Notes

* I am grateful to Steven Krauwer and Jan Odijk for helpful discussion. To Henk van Riemsdijk I am especially indebted for sharing generously with me his vast knowledge of *Zurritütsch* as well as numerous other objects abstract and otherwise.

1. For discussion of controversial issues see Langendoen (1977), Postal (1964), Huybregts (1976) and especially Pullum and Gazdar (1982). For discussion of the sometimes confused issue of linguistic simplicity and learnability see Chomsky (1965, chapter 1, esp. pp.61,62), (1977, note 61), and particularly (1980). For different and opposing views see Gazdar (1981, 1983) and Langendoen and Postal (1983).

2. As to the non-necessity of TGG see Gazdar (1981,1982,1983) and Gazdar and Pullum (1982). As to the non-sufficiency of TGG see Langendoen and Postal (1983). For discussion of philosophical issues and matters relating to real-time processing of language, see Katz (1981) and Gazdar (1983), Pullum (1983).

3. In the worst possible case context-free grammars parse in time proportional to the cube of the length of the sentence. See Gazdar (1981), Berwick and Weinberg (1982) and Pullum (1983).

4. See Gazdar (1981,1983) and Pullum (1983).

5. Langendoen and Postal (1983) give a summary of existing theories of grammar. It appears that all theories except the Arc-Pair Model are of the constructive type.

6. For an interesting in-depth discussion of Platonic Realism and relating matters see especially the work of Katz (1981), where a version of a Platonic linguistic theory is defended.

7. Suppose someone were to argue that since languages are derivative and the notion NL is obscure, the concept of a CF-PS-Grammar must be on the wrong track. For a suggestion to that effect see Koster (1983). Note, however, that this reasoning is false. One could accept the view that grammar is the more primitive notion and still claim that CF-Grammar is a fully explanatory theory of language. This is exactly what the controversy between Williams (1978) and Gazdar (1981) is about. Koster's metaphysical argument is quite independent of the issue of what is the best theory of language, CF-Grammar or TG-Grammar.

8. The term 'doughnut-hierarchy' is due to an unpublished paper by Lasnik, "Learnability, restrictiveness and the evaluation metric." The Chomsky-Hierarchy is developed in Chomsky (1959). See also Chomsky (1963).

9. Clearly, any such "generalization" is no longer CFL-inducing, resulting only in unfortunate terminology. Note that any "generalization" or "extension" of categorial and context-free grammar as suggested e.g. in Ades and Steedman (1982), Culy (1983), and Thompson (1983) to 'handle' cross-seriality can not deal with the Dutch and Swiss German cases unless these systems are "generalized" beyond CFL-inducing systems like indexed grammars. For a discussion of indexed grammars see Hopcroft and Ullmann (1979) and Aho and Ullmann (1972). It appears that the 'copying languages' in the main text are indexed languages. See note 29.

10. See Davis (1958) for an intelligible discussion of Gödel (1931).

11. See Gödel's discussion cited in Katz (1981, p. 200). Interestingly, in this paper ("What is Cantor's Continuum Problem") Gödel defends a Platonist view of the foundations of mathematics. It should not surprise us, then, that the Platonist view of language adopted by Langendoen and Postal (1983) coincides with their acceptance of number-theoretic aspects of NL, viz. transfiniteness of NL-sentences and nondenumerability of NL-collections.

12. See Chomsky (1956, 1963).

13. Megacollections or 'proper classes' are collections with the size of the collection of all sets. Megacollections are therefore no sets. According to Langendoen and Postal, NL are megacollections to be identified with no fixed set of sentences "no matter how great its cardinality". Note that if NL can not be finitely specified humans can not have complete knowledge of NL, a position that is perfectly consistent with Platonism but hardly if at all with Competence. See Langendoen and Postal's discussion of these matters. For an early argument that 'internalized' grammars need not be generative see Postal (1972).

14. For proofs of these theorems see Bar-Hillel (1961), Hopcroft and Ullmann (1979), and Aho and Ullmann (1972).

15. This proof is stated in Huybregts (1976, pp. 36-38). There we applied theorem (9) to the strings of $L_{cs} = \{a^n b^m a^n\}$. Informally, by theorem (9) there are two constants p and q such that every string z_i in L_{cs} with length greater than p has a factorization $z_i = uv^q wx^q y$ (v & x not both null, and the length of vwx less than or equal to q). Suppose we take q to be smaller than $n+m$, and p to be the length of the longest terminal string of all S-rooted trees generated by grammar $G(L_{cs})$ all of whose paths are of length n at most, where n is the cardinal number of the nonterminal vocabulary of $G(L_{cs})$. Then for all $a^n b^m a^n$ to be represented as $uv^q wx^q y$, v and x must consist entirely of as or bs . Moreover, since q is less than $n+m$, if v is in the first run of as , then x can not be in the second run of as . Similarly if v and x both consist entirely of bs . Consequently, in every factorization consistent with theorem (9), recursion of v and x will produce strings in which the number of as or bs or both in the first run is not matched by the respective number of as or bs or both in the second run. The assumption of context-freeness of L_{cs} therefore leads to contradictions only, proving the falseness of the underlying assumption. Hence L_{cs} can not be context-free.

16. Thatcher's theorem is linguistically interesting in that it applies finite automata theory to prove theorems about sets of derivation trees, giving a characterization of strong generative capacity of context-free grammars. Looking again at theorem (9) and letting $fr(t_i) = uSy$, $fr(t_i) = vSx$, and $fr(t_i) = w$ (where fr stands for 'frontier' of trees as well as S), then if some automaton accepts the derivation trees $T(G)$ of some context-free grammar G , $t_i(t_j(t_k))$ is in $T(G)$ for all n (where $t_i(t_j(t_k))$ is the result of substituting t_k for the occurrence of S in t_j). Hence, for all i , $uv^q wx^q y$ is in $L(G)$.

17. Their claim that the relevant trees "contain two right-branching subtrees of matching heights", crucial to derive strong non-context-freeness is invalidated by perfectly natural sentences of Dutch like (i) with instances of illicit (in their terms) coordination. Note that in (i) neither *ik heb* nor *een hond onder deze* nor *oude stoel zien liggen* is a constituent. Therefore, their

argument is nullified.

(i) *ik heb een hond onder deze - en (hij heeft) een kat onder die - oude stoel zien liggen*

(I have a dog under this - and (he has) a cat under that - old chair see lie)

Also note that their "explanation" entails that in Dutch maximal projections are the projections of no head rendering X-bar theory completely vacuous.

18. The relevant notion in the cross-serial dependences in Dutch is the syntactic concept of GR bearing a theta-role in the sense of Chomsky (1981, 1982), viz. theta-position. Naturally, theta-positions are subject to selectional conditions imposed by their predicates, heads of X-bar theory.

19. See Evers (1975).

20. For some recent discussion see van den Berg (1971). Apparently, *laten* and *zien* also assign dative instead of accusative to their complement subjects. Since *doen* assigns accusative only we still have an argument.

21. See Lötscher (1978) for a discussion of the Swiss German facts. We are indebted to Henk van Riemsdijk for checking the Swiss German data in the discussion that follows.

22. See Bech (1955) for a discussion of the German data. For a classic TG-account of VR-phenomena see especially the work of Evers (1975).

23. For a useful general taxonomy of VReffects in Germanic languages see den Besten and Edmondson (1981). Also see Vanacker (1970) for Flemish Dutch. For a nonstandard approach to these matters see Huybregts (forthcoming) where VReffects are analyzed as implicating a totally different conception of the form of syntactic representation (a preliminary view was presented on the UFSAL Brussels conference on Sentential Complementation, June 1983, in a talk "Cross-serial dependencies and the form of syntactic structure").

24. Cf. Chomsky (1981, 1982).

25. See note 15.

26. See Huybregts (forthcoming) for an explanation of VR-effects that has some nontrivial consequences for the nature of syntactic representation.

27. For detailed discussion of this claim see Huybregts (in preparation), where a different formulation of the relevant closure properties of NL under coordinate compounding is given, which does not necessitate the extension of NL-grammars to non-proof-theoretic systems.

28. See Hopcroft and Ullmann (1979). In fact, even the more restricted non-erasing varieties of one-way stack automata are rich enough to recognize these languages deterministically ($\omega\omega$) or nondeterministically ($\omega\omega$). The deterministic variants are properly included in the non-deterministic variants just as is the case with push-down storage automata.

29. Consider the following indexed grammar. Let $G = (\{S, C, L, R\}, \{a, b\}, \{i, j\}, p, q, P, S)$, where P consists of the following production rules:

$S \rightarrow Cp / Cq,$	$Li \rightarrow al,$	$Ri \rightarrow aR,$
$C \rightarrow Ci / Cj,$	$Lj \rightarrow bL,$	$Rj \rightarrow bR,$
$C \rightarrow LR,$	$Lp \rightarrow a,$	$Rp \rightarrow a,$
	$Lq \rightarrow b,$	$Rq \rightarrow b.$

A sample derivation for *baabaa* would be $S \Rightarrow Cp \Rightarrow Cip \Rightarrow Cjip \Rightarrow LjipRjip \Rightarrow bLipRjip \Rightarrow baLpRjip \Rightarrow baarjip \Rightarrow baabRip \Rightarrow baabaRip \Rightarrow baabaa$. In general, $S \Rightarrow Ckr \Rightarrow LkrRkr \Rightarrow \omega\omega$, where $k \in \{i, j\}^*$, $r \in \{p, q\}$, and $\omega\omega = \{a, b\}^*$. It is clear that $L(G) = \{\omega\omega \mid \omega \in \{a, b\}^*\}$.

References

- Ades, A., and M. Steedman (1982) "On the order of words," *Linguistics and Philosophy* 4, 517-558.
- Aho, A.V., and J.D. Ullmann (1972) *The Theory of Parsing, Translation and Compiling*, Vol. I: *Parsing*. Prentice-Hall, Englewood Cliffs, New Jersey.
- Bar-Hillel, Y., M. Perles, and E. Shamir (1961) "On formal properties of simple phrase structure grammars," in R.D. Luce, R. Bush, and E. Galanter (eds.), *Readings in Mathematical Psychology* (1965), Vol. II. John Wiley, New York, 75-104.
- Bech, G. (1955) *Studien über das Deutsche Verbum Infinitum*, Band I, Munksgaard, Copenhagen.
- van den Berg, B. (1971) *Inleiding tot de Middelnederlandse Syntaxis*, Wolters-Noordhoff, Groningen.
- Berwick, R.C., and A.S. Weinberg (1982) "Parsing efficiency, computational complexity, and the evaluation of grammatical theories," *Linguistic Inquiry* 13.2, 165-191.
- den Besten, H., and J. Edmondson (1981) "The verbal complex in continental West Germanic," *Groninger Arbeiten zur Germanistischen Linguistik* 19, 11-61.
- Bresnan, J., R.M. Kaplan, S. Peters, and A. Zaenen (1982) "Cross-serial dependencies in Dutch," *Linguistic Inquiry* 13.4, 613-635.
- Chomsky, N. (1956) "Three models for the description of language," in R.D. Luce, R. Bush, and E. Galanter (eds.), *Readings in Mathematical Psychology* (1965), Vol. II, John Wiley, New York, 105-124.
- Chomsky, N. (1959) "On certain formal properties of grammar," *Information and Control* 2, 133-167.
- Chomsky, N. (1963) "Formal properties of grammar," in R.D. Luce, R. Bush, and E. Galanter (eds.) *Handbook of Mathematical Psychology* (1963), Vol. II. John Wiley, New York, 323-418.
- Chomsky, N. (1965) *Aspects of the Theory of Syntax*. MIT Press, Cambridge, Mass.
- Chomsky, N. (1977) "On *wh*-movement," in P. Culicover, T. Wasow, and A. Akmajian (eds.) *Formal Syntax*. Academic Press, New York, 71-132.
- Chomsky, N. (1980) *Rules and Representations*. Columbia University Press, New York.
- Chomsky, N. (1981) *Lectures on Government and Binding*. Foris, Dordrecht.
- Chomsky, N. (1982) *Some Concepts and Consequences of the Theory of Government and Binding*. *Linguistic Inquiry Monograph* 6, MIT Press, Cambridge, Mass.
- Culy, C.D. (1983) "An extension of phrase structure rules and its application to natural language," MA thesis, Stanford University.
- Davis, M. (1958) *Computability and Unsolvability*. McGraw-Hill, New York.
- Evers, A. (1975) *The Transformational Cycle in Dutch and German*. Utrecht diss., Indiana University Linguistics Club, Bloomington.
- Gazdar, G. (1981) "Unbounded dependencies and coordinate structure," *Linguistic Inquiry* 12, 155-184.
- Gazdar, G. (1982) "Phrase structure grammar," in P. Jacobson and G.K. Pullum (eds.) *The Nature of Syntactic Representation*. D. Reidel, Dordrecht, 131-186.
- Gazdar, G. (1983) "Phrase structure grammars and natural languages," in A. Bundy (ed.) *Proceedings of the 8th International Joint Conference on Artificial Intelligence* (8-12 August 1983, Karlsruhe, W-Germany), Vol. I. William Kaufmann, Los Altos, Cal., 556-565.
- Gazdar, G., and G.K. Pullum (1982) "Generalized phrase structure grammar: a theoretical synopsis." Indiana University Linguistics Club, Bloomington.
- Gödel, K. (1931) "Ueber formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I," *Monatshefte für Mathematik und Physik* 38, 173-198.

- Huybregts, M.A.C. (1976) "Overlapping dependencies in Dutch," *Utrecht Working Papers in Linguistics* 1, 24-65.
- Huybregts, M.A.C. (in preparation) "Bounds and dimensions of natural language".
- Huybregts, M.A.C. (forthcoming) *A Formal Theory of Binding and Bounding*.
- Hopcroft, J.E., and J.D. Ullmann (1979) *Introduction to Automata Theory, Language and Computation*. Addison-Wesley, Reading, Mass.
- Katz, J.J. (1981) *Language and other Abstract Objects*. Rowman and Littlefield, Totowa, New Jersey.
- Koster, J. (1983) "De ontsemitisering van het wereldbeeld," *Tilburg Papers in Language and Literature* 40.
- Langendoen, D.T. (1977) "On the inadequacy of type-3 and type-2 grammars for human languages," in P.J. Hopper (ed.), *Studies in Descriptive and Historical Linguistics. Festschrift for Winfred P. Lehmann*. John Benjamin, Amsterdam, 159-171.
- Langendoen, D.T., and P.M. Postal (1983) "The vastness of natural languages," Mimeo, CUNY and IBM T.J. Watson Research Center.
- Lötscher, A. (1978) "Zur Verbstellung im Zürichdeutschen und in anderen Varianten des Deutschen," *Zeitschrift für Dialektologie und Linguistik* 45.1, 1-29.
- Postal, P.M. (1964) "Limitations of phrase structure grammars," in J. Fodor and J.J. Katz (eds.), *The Structure of Language*. Prentice-Hall, Englewood Cliffs, New Jersey, 131-170.
- Postal, P.M. (1972) "The best theory," in S. Peters (ed.), *Goals of Linguistic Theory*. Prentice-Hall, Englewood Cliffs, New Jersey, 131-170.
- Peters, S., and R. Ritchie (1973) "Nonfiltering and local-filtering transformational grammars," in J. Hintikka, J. Moravcsik, and P. Suppes (eds.), *Approaches to Natural Language*. D. Reidel, Dordrecht, 180-194.
- Pullum, G.K. (1983) "Context-freeness and the computer processing of human languages," *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, 1-6.
- Pullum, G.K., and G. Gazdar (1982) "Natural languages and context-free languages," *Linguistics and Philosophy* 4, 471-504.
- Thatcher, J.W. (1967) "Characterizing derivation trees of context-free grammars through a generalization of finite automata theory," *Journal of Computer and System Science* 1, 317-322.
- Thatcher, J.W. (1973) "Tree automata: an informal survey," in A.V. Aho (ed.), *Currents in the Theory of Computing*. Prentice-Hall, Englewood Cliffs, New Jersey, 143-172.
- Thompson, H. (1983) "Crossed serial dependencies: a low-power parseable extension to GPSG," *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, 16-21.
- Vanacker, V. (1970) "Een 'Zuidnederlandse' konstruktie in een paar Zuidnederlandse dialecten," *De Nieuwe Taalgids (van Haeringen-nummer)*, 140-157.
- Williams, E.S. (1978) "Across-the-board rule application," *Linguistic Inquiry* 9, 31-43.

Chapter 2

The Convergence of Mildly Context-Sensitive Grammar Formalisms

Aravind K. Joshi, K. Vijay-Shanker, and David Weir

1 Introduction

Since the late 1970s there has been vigorous activity in constructing highly constrained grammatical systems by eliminating the transformational component either totally or partially. This was caused by increasing recognition that the entire range of dependencies that transformational grammars in their various incarnations have tried to account for can be captured satisfactorily by classes of grammars that are *nontransformational* and at the same time highly constrained in terms of the classes of grammars and languages they define. Peters and Ritchie (1972) showed that context-sensitive grammars (CSGs)—if used for analysis (and not for generation), thus providing more descriptive power than context-free grammars (CFGs)—have the same weak generative capacity as CFGs. This result was generalized by Joshi and Levy (1977) to Boolean combinations of contextual predicates and domination predicates.

In the early 1980s Gazdar (1982) proposed a grammatical formalism (which later became the Generalized Phrase Structure Grammar (GPSG) formalism; see Gazdar et al. 1985) whose weak generative capacity is the same as that of the CFGs, but which is adequate to describe various syntactic phenomena previously described in transformational terms. Gazdar was careful to note that his results did not mean that syntactic phenomena that required formal power beyond CFG did not exist. Rather, his claim was that, with respect to the range of phenomena known at that time, CFGs seemed to be quite adequate. Later, some clear examples of natural language phenomena were discovered that required formal power beyond CFG (see, for example, Shieber 1985 and Culy 1985, for an argument from

This chapter is a revised version of a paper presented by the first author at the conference on "The Processing of Linguistic Structure" sponsored by the System Development Foundation held at Santa Cruz, CA, January 1987. We want to thank Jean Gallier, Tony Kroch, Mitch Marcus, Remo Pareschi, Yves Schabes, Mark Steedman, Ramesh Subrahmanyam, and Bonnie Webber for valuable discussion about several aspects of this chapter. This work was partially supported by NSF grant IRI84-10413 A02, US Army grant DAA6-29-84K-0061, DARPA grant N0014-85-K0018, and Advanced Technology Center (PA) grant #309.

weak generative capacity, and Bresnan et al. 1982, for an argument from strong generative capacity). Hence, the question of how much power beyond CFG is necessary to describe these phenomena became important.

Pollard (1984) proposed an extension of CFG, called head grammars (HGs), which introduced some wrapping operations beyond the concatenation operation in CFG. Roach (1987) investigated some formal properties of HGs. HGs, like CFGs, are string-generating systems.

Joshi, Levy, and Takahashi (1975) introduced a grammatical formalism called tree-adjoining grammars (TAGs), which are tree-generating systems, and investigated some of their formal properties. Joshi (1985) showed how TAGs factor recursion and the domain of dependencies in a novel way, leading to "localization" of dependencies, their long-distance behavior following from the operation of composition, called "adjunction." TAGs have more power than CFGs, and this extra power is a corollary of factorization of recursion and the domain of dependencies. This extra power appeared to be adequate for the various phenomena requiring more formal power than CFG. The linguistic significance of TAGs has been discussed by Joshi (1985), Kroch and Joshi (1985, 1986), Kroch (1986), and Kroch and Santorini (to appear). Based on the formal properties of TAGs, Joshi (1985) proposed that the class of grammars that is necessary for describing natural languages might be characterized as the class of *mildly context-sensitive grammars* (MCSG; MCSL for the corresponding languages) possessing at least the following properties: (1) context-free languages (CFLs) are properly contained in MCSL; (2) languages in MCSL can be parsed in polynomial time; (3) MCSGs capture only certain kinds of dependencies, such as nested dependencies and certain limited kinds of crossing dependencies (for example, in subordinate clause constructions in Dutch or some variations of them, but perhaps not in the so-called MIX (or Bach) language, which consists of equal numbers of *a*'s, *b*'s, and *c*'s in any order); and (4) languages in MCSL have the constant growth property. This last property means that if the strings of a language are arranged in increasing order of length, then two consecutive lengths do not differ by arbitrarily large amounts. In fact, any given length can be described as a linear combination of a finite set of fixed lengths. This property is slightly weaker than the property of semilinearity. It is intended to be an approximate characterization of the linguistic intuition that sentences of a natural language are built from a finite set of clauses of bounded structure using certain simple linear operations. The characterization of this intuition by the constant growth property is approximate because it refers to the growth of strings and not to the growth of structures.

It should be noted that these properties do not precisely define MCSG but rather give only a rough characterization, because the properties are

only necessary conditions, and because some of the properties are properties of structural descriptions rather than of the languages and therefore are difficult to characterize precisely. This characterization of MCSG, obviously motivated by the formal properties of TAGs, would have remained only a passing remark were it not for some subsequent developments.

In response to a talk by Geoffrey Pullum at COLING 84, Joshi pointed out that all the known formal properties of HGs appeared to be exactly the same as those of TAGs. Later it was shown that with a slight modification of HGs (which is necessary because the wrapping operations are undefined for null strings), HGs are equivalent to TAGs (Vijay-Shanker, Weir, and Joshi 1986; Weir, Vijay-Shanker, and Joshi 1986).

Since then, two other formalisms have also been shown to be equivalent to TAGs. These are linear indexed grammars (LIGs) (Gazdar 1985) and combinatorial categorial grammars (CCGs) (as developed in Steedman 1987, 1990). Thus, four quite different formalisms have been shown to be equivalent and thus belong to the class of MCSGs. These formalisms are different from each other in the sense that the formal objects and operations they employ are quite distinct and in the sense that they are motivated by attempts to capture different aspects of language structure. Each of these formalisms has a domain of locality that is larger than that specifiable in a CFG. By a domain of locality we mean the elementary structures of a formalism over which dependencies such as agreement, subcategorization, and filler-gap relations can be specified. However, it is not the case that each one of these formalisms extends the domain of locality to the same extent. TAGs extend the domain of locality far enough that recursion is factored away from the domain of dependencies.

When two formalisms based on apparently completely different ideas turn out to be equivalent, there is a possibility that we are getting a handle on some fundamental properties of the objects that these formalisms were designed to describe. When more than two distinct formalisms turn out to be equivalent, the possibility is even greater. In fact, a deeper understanding of the relationships between these formalisms is obtained if we look at the derivation structures (related to structural descriptions) provided by each formalism. A first attempt to capture the closeness of some of these formalisms at the level of derivation structures resulted in the linear context-free rewriting systems (LCFRSs) described in section 6.

The rest of the chapter is organized as follows. In section 2 we present an introduction to TAGs, including some simple examples. We discuss TAGs in more detail than the other formalisms because the theory of TAGs has played a key role in our investigations of the relationships between different grammatical formalisms. In fact, most of the equivalences described in this chapter have been established via TAGs. In section

2 we also describe an extension of TAGs, called multicomponent TAGs (MCTAGs), first discussed by Joshi, Levy, and Takahashi (1975) and later defined precisely by Weir (1988). MCTAGs also belong to the class of MCSGs and are in fact equivalent to LCFRSs.

In section 3 we briefly describe HGs and show their equivalence to TAGs. In section 4 we show the equivalence of LIGs to TAGs, and in section 5 the equivalence of CCGs to LIGs and thereby to TAGs and HGs. In section 6 we present LCFRSs.

In section 7 we briefly present feature-structure-based TAGs (FTAGs), where adjunction becomes function application and unification. FTAGs, in general, are unconstrained; however, if the feature structures associated with each node in an elementary tree are bounded, then this restricted FTAG (RFTAG) is equivalent to TAG. This restriction on feature structures is similar to that in GPSG. However, since TAGs have an extended domain of locality, RFTAGs (equivalent to TAGs) are more powerful than GPSGs (equivalent to CFGs).

In section 8 we consider a variant of TAGs, the lexicalized TAGs (LTAGs). Although adjunction can simulate substitution, by adding the operation of substitution explicitly, we obtain LTAGs. Such a lexicalization is implicit in TAGs in the sense that all the elementary trees need not be explicitly stated. However, the framework of LTAGs brings this out explicitly. LTAGs are equivalent to TAGs. The relationship between TAGs and CCGs also becomes clear in the framework of LTAGs.

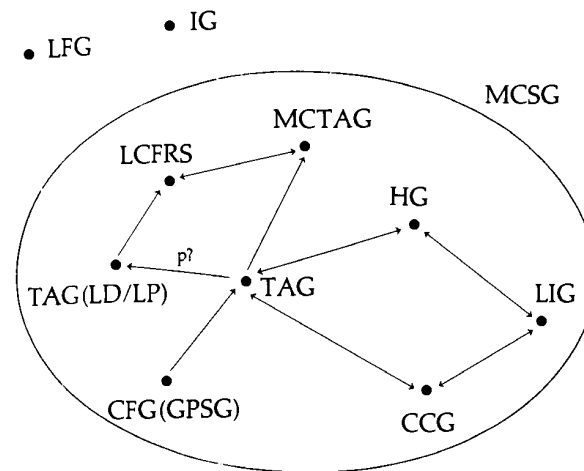
Finally, in section 9 we consider a generalization of TAGs, called TAG(LD/LP), which decouples (local) domination from linear precedence and allows a treatment of complex word-order patterns including those that exhibit long-distance behavior similar to the filler-gap dependencies, but different in the sense that "movement" is not to a grammatically defined position. The languages of TAGs are obviously contained in TAGs(LD/LP). It is not known yet whether the containment is proper.

Figure 2.1 summarizes all these relationships. The nodes are labeled by the grammar formalisms. The containments shown are, of course, with respect to the corresponding languages. Two formalisms outside of MCSG are also shown in the figure.

We present our results here informally, giving examples to illustrate the ideas in the proofs. We leave more detailed proofs with full technical details to future work.

2 Tree-Adjoining Grammar Formalism

A *tree-adjoining grammar* (TAG) $G = (I, A)$, where I and A are finite sets of *elementary trees*. The trees in I will be called the *initial trees* and the trees in A , the *auxiliary trees*. A tree α is an initial tree if it is of the form in (1):



[The containments are with respect to the languages.]

CCG: Combinatory categorial grammars

CFG(GPSG): Context-free grammars (generalized phrase structure grammars)

HG: Head grammars

IG: Indexed grammars

LCFRS: Linear context-free rewriting systems

LFG: Lexical-functional grammars

LIG: Linear indexed grammars

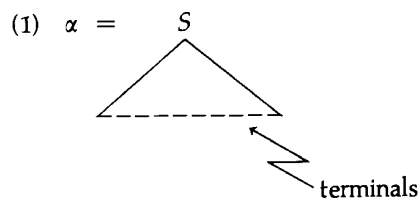
MCTAG: Multicomponent TAGs

TAG: Tree-adjoining grammars

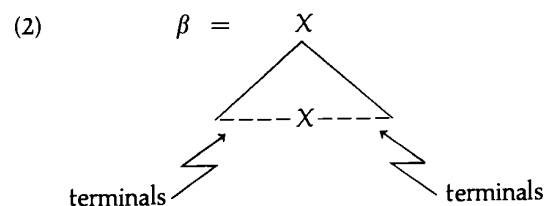
TAG(LD/LP): TAGs (local domination/linear precedence)

Figure 2.1

Mildly context-sensitive grammar formalisms (MCSG)



That is, the root node of α is labeled S and the frontier nodes are all terminals. A tree β is an auxiliary tree if it is of the form in (2):



That is, the root node of β is labeled X , where X is a nonterminal, and the frontier nodes are all terminals except one, which is also labeled X . The frontier node labeled X will be called the *foot node* of β . The internal nodes are nonterminals. The initial and the auxiliary trees are not constrained in any other manner. The idea, however, is that both the initial and the auxiliary trees will be *minimal* in some sense. An initial tree will correspond to a minimal sentential tree (that is, without recursing on any nonterminal), and an auxiliary tree, with root and foot node labeled X , will correspond to a minimal recursive structure that must be brought into the derivation, if one recurses on X .

We will now define a composition operation called *adjunction*, which composes an auxiliary tree β with a tree γ . Let γ be a tree containing a node n bearing the label X and let β be an auxiliary tree whose root node is also labeled X . (Note that β must have, by definition, one (and only one) frontier node labeled X .) Then the adjunction of β to γ at node n will be the tree γ' that results when the following operations are carried out:

1. The subtree of γ dominated by n , call it t , is excised.
2. The auxiliary tree β is attached at n and its root node is identified with n .
3. The subtree t is attached to the foot node of β and the root node n of t is identified with the foot node of β .

Figure 2.2 illustrates this operation.

The intuition underlying the adjunction operation is a simple one, but the operation is distinct from other operations on trees that have been discussed in the literature. In particular, we want to emphasize that adjunction is not a substitution operation in the usual sense.

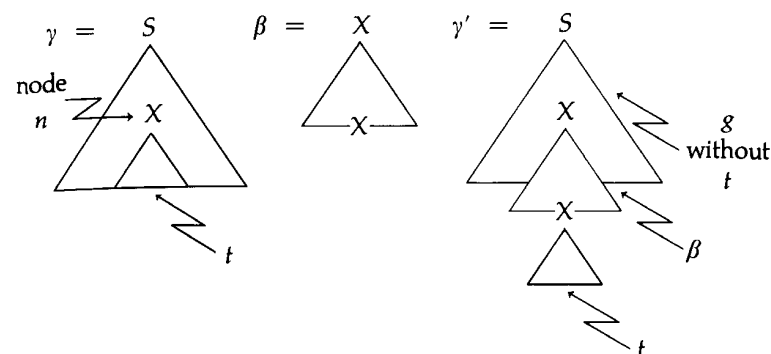


Figure 2.2
Adjunction

Adjunction can, however, simulate substitution. A variant of TAG (called lexicalized TAG) that explicitly uses both adjunction and substitution is discussed in section 8. Lexicalized TAGs are equivalent to TAGs. For the rest of the chapter we will only consider adjunction as defined above.

The definition of adjunction allows more complex constraints to be placed on adjoining. Associated with each node is a *selective adjunction* (SA) constraint specifying the subset of auxiliary trees that can be adjoined at this node. Trees can be included in the SA constraint associated with a particular node only if their root and foot are labeled with the same nonterminal that labels the node. A mechanism is provided for ensuring that adjunction is performed at a node. This is done by associating an *obligatory adjunction* (OA) constraint with that node.

We should note that the SA and OA constraints are more than mere notational convenience, since they increase the generative power of the formalism. If the SA constraint specifies an empty subset of trees, then adjunction cannot be performed at this node—we call this constraint the *null adjunction* (NA) constraint.

(In section 7 we will describe a feature-structure-based TAG (FTAG). In this framework, adjunction becomes function application and unification, and the constraints described above are implicit in the feature structures and the success or failure of unification during composition. A restricted version of FTAG, RFTAG, also described in section 7, is equivalent to TAG. For the rest of the chapter we will consider the constraints as described above.)

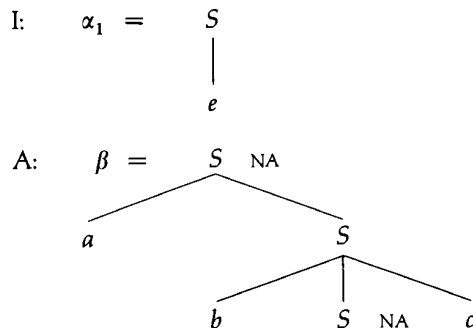
Let us now define two auxiliary notions, the tree set of a TAG and the string language of a TAG. Suppose $G = (I, A)$ is a TAG with a finite set of initial trees, a finite set of auxiliary trees, and the adjunction operation, as

above. Then we define the *tree set* of a TAG G , $T(G)$, to be the set of all trees derived in G starting from initial trees in I . We further define the *string language* (or *language*) of G to be the set of all terminal strings of the trees in $T(G)$. The relationship between TAGs, CFGs, and the corresponding string languages can then be summarized in the following theorems (Joshi, Levy, and Takahashi 1975; Joshi 1985):

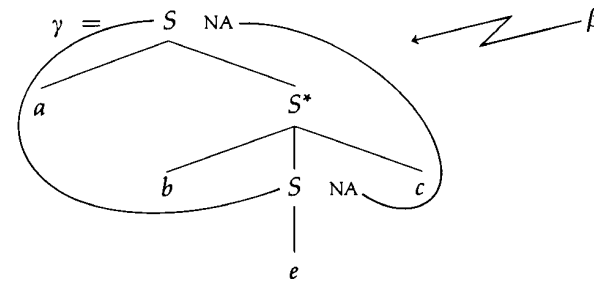
1. For every CFG, G' , there is a TAG, G , which is both weakly and strongly equivalent to G' . In other words, $L(G) = L(G')$ and $T(G) = T(G')$.
2. There exists a nonempty set of TAGs G_1 such that for every $G \in G_1$, $L(G)$ is context free but there is no CFG G' such that $T(G') = T(G)$. That is, TAGs are capable of providing structural descriptions for context-free languages that are not obtainable by a CFG.
3. There exists a nonempty set of TAGs G_2 such that for every $G \in G_2$, $L(G)$ is strictly context sensitive. That is, there is no CFG G' such that $L(G) = L(G')$ (or, in other words, TAGs are strictly more powerful than CFGs).
4. There exist context-sensitive languages for which there are no equivalent TAGs. That is, TAGs are properly contained in context-sensitive languages.
5. TAGs are semilinear and hence have the constant growth property.
6. TAGs can capture only certain limited kinds of crossed dependencies. This follows from the nature of the automaton that corresponds to a TAG, called an *embedded pushdown automaton (EPDA)*, which is a generalization of the *pushdown automaton (PDA)*.
7. TAGs can be parsed in polynomial time—in fact, with a time based Kn^6 , where n is the length of the string and K is a constant depending on the grammar.

2.1 Some Examples of Formal Languages

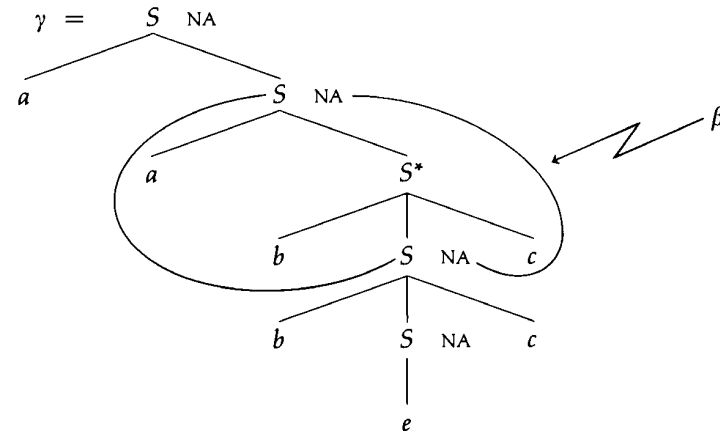
EXAMPLE 1 Let $G = (I, A)$ be a TAG with local constraints where



There are no constraints in α_1 . In β no auxiliary trees can be adjoined at either the root node or the foot node; the center S node has no constraints. Starting with α_1 and adjoining β to α_1 at the root node, we obtain



Adjoining β to the center S node (the only node at which adjunction can occur), we have



It is easy to see that G generates the string language

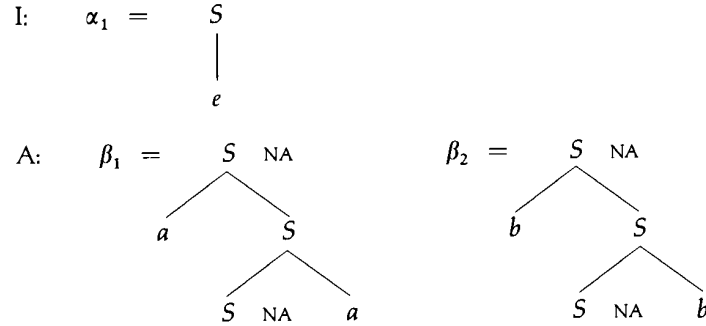
$$L = \{a^n b^n e c^n | n \geq 0\}.$$

EXAMPLE 2 Let G' be a TAG similar to G in example 1, except that in G' there are no constraints in β . G' generates

$$L = \{wec^n | n \geq 0, \#a\text{'s in } w = \#b\text{'s in } w = n, \\ \text{and for any proper initial string } u \\ \text{of } w, \#a\text{'s in } u \geq \#b\text{'s in } u\}.$$

This language is closely related to the context-sensitive language discussed by Higginbotham (1984), which can also be shown to be a TAG language.

EXAMPLE 3 Let $G = (I, A)$ be a TAG with local constraints, where



G generates the language

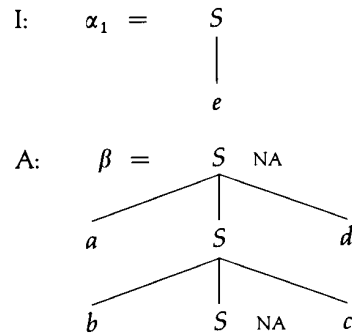
$$L = \{wew \mid w \in \{a, b\}^*\}.$$

EXAMPLE 4 Let G' be a TAG that is the same as G in example 3 but without any local constraints. The corresponding language is

$$L = \{wew' \mid w, w' \in \{a, b\}^*, |w| = |w'|, \\ \#a\text{'s in } w = \#a\text{'s in } w', \#b\text{'s} \\ \text{in } w = \#b\text{'s in } w' = n\}.$$

This language is related to the Swiss-German example discussed by Shieber (1985).

EXAMPLE 5 Let $G = (I, A)$ be a TAG with local constraints, where



G generates

$$L = \{a^n b^n e c^n d^n \mid n \geq 1\}.$$

Note that it can be shown that languages

$$L^1 = \{a^n b^n c^n d^n e^n \mid n \geq 1\}$$

and

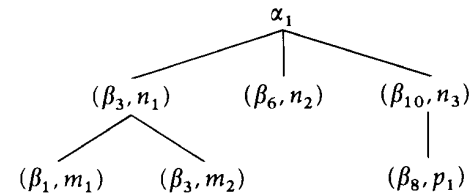
$$L^2 = \{www \mid w \in \{a, b\}^*\}$$

cannot be generated by TAGs either with or without local constraints (Joshi 1985). Other languages such as $L' = \{a^{n^2} \mid n \geq 1\}$ also cannot be generated by TAGs. This is because the strings of a TAG have the constant growth property.

2.2 Derivation in a TAG

We will not describe formally the notion of derivation in a TAG; however, the informal discussion to follow will make the notion of derivation in TAG precise enough for our purpose. Adjunction is an operation defined on an elementary tree, say γ , an auxiliary tree, say β , and a node (that is, an address) in γ , say n . Thus, every instance of adjunction is of the form " β is adjoined to γ at n ," and this adjunction is always and only subject to the local constraints associated with n . Although we very often speak of adjoining a tree to a node in a complex structure, we do so only for convenience. Strictly speaking, adjunction is always at a node in an elementary tree; therefore, it is more precise to talk about adjoining at an address in an elementary tree. More than one auxiliary tree can be adjoined to an elementary tree as long as each tree is adjoined at a distinct node. After these auxiliary trees are adjoined to the elementary tree, only nodes in the auxiliary trees are available for further adjunction. This precision in the definition of adjunction will be necessary when we define multi-component adjunction in section 2.3.

Now suppose that α is an initial tree and that $\beta_1, \beta_2, \dots$ are auxiliary trees in a TAG, G . Then the derivation structure corresponding to the generation of a particular string in $L(G)$ might look as follows:



α_1 is an initial tree. β_3, β_6 , and β_{10} are adjoined at nodes n_1, n_2 , and n_3 respectively in α_1 , where n_1, n_2 , and n_3 are all distinct nodes. β_1 and β_3 are adjoined to β_3 at nodes m_1 and m_2 respectively. Again, m_1 and m_2 are distinct. β_6 has no further adjunctions, but β_8 is adjoined to β_{10} at node p_1 . Note that the derivation structure implicitly characterizes the "surface" tree that is generated by it. (See section 7 for the relationship of TAG and the unification formalism.) In this way the derivation structure can be seen as the basic formal object constructed in the course of deriving a sentence.

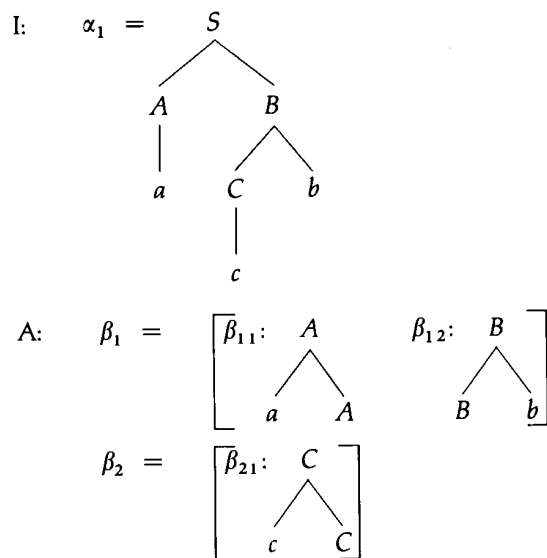
Associated with it will be two mappings, one to a surface syntactic tree and the other to a semantic interpretation, as below. (We are not concerned with semantic interpretation in this chapter.)

surface tree $\leftarrow$ derivation structure $\rightarrow$ semantic interpretation

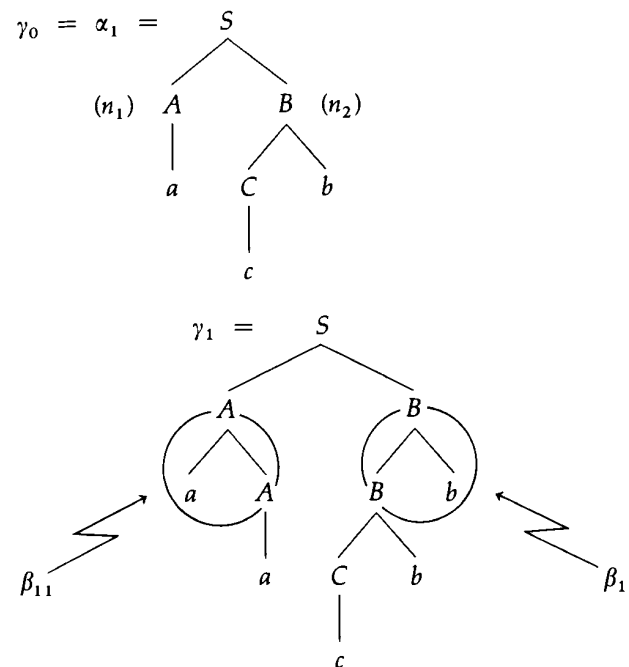
In a CFG the derivation structure is the same as the surface structure. In a TAG this is not the case. Several of the formalisms that we have shown to be equivalent are comparable at the level of the derivation structures. Based on this observation, in section 6 we discuss a framework called the linear context-free rewriting system (LCFRS), which captures the commonality at the level of derivation structures.

2.3 Multicomponent TAGs

In Joshi, Levy, and Takahashi 1975 a version of the adjunction operation is introduced under which, instead of a single auxiliary tree, a set of such trees is adjoined to a given elementary tree. This extension of TAG is known as *multicomponent TAG (MCTAG)*. We define the adjunction of such a set as the simultaneous adjunction of each of its component trees to a distinct node (address) in an elementary tree. This adjunction can, of course, take place only if the local constraints associated with each affected node of the elementary tree are satisfied. Consider, for example, the following grammar $G = (I, A)$:



β_1 is an auxiliary set consisting of the two trees β_{11} and β_{12} . Here is a sample derivation in G :

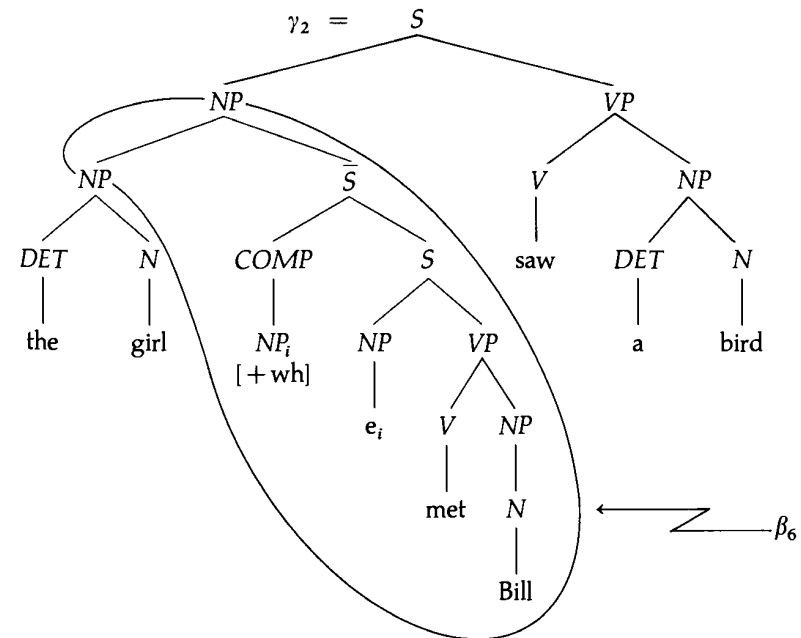
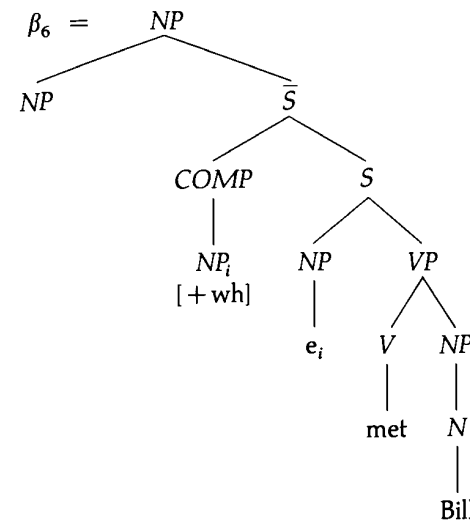
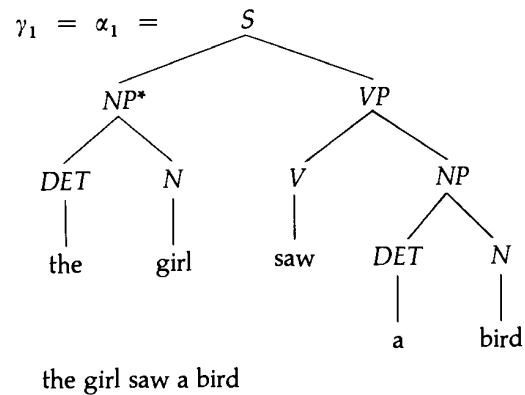


γ_1 , it should be clear, is obtained by adjunction of the components β_{11} and β_{12} of the auxiliary set β_1 to γ_0 at the nodes n_1 and n_2 respectively. In the current example, the set β_1 has two component trees and β_2 has only one component. If every auxiliary tree set of a TAG has only one component, we have a TAG as defined earlier. It can be shown that the number of components in the auxiliary sets does not make any difference to the generative capacity; in other words, both the weak and strong (with respect to tree sets generated and not, of course, with respect to the derivation structures) generative capacities of MCTAGs are the same as those for TAGs where each auxiliary set has exactly one component. On the other hand, derived auxiliary sets can be defined by adjoining an auxiliary set, say β_1 , to another auxiliary set, say β_2 , as follows. Each component of β_1 is adjoined to one (and exactly one) component of β_2 , and all adjunctions are at distinct nodes. Note that since it is not required that each component of β_1 adjoin to the same component of β_2 , one component may adjoin to one component and another component to a different component of β_2 ; that is, adjunctions of components are not to the same component (elementary tree) of β_1 , but they are all adjunctions to the *same auxiliary set*. Thus, *locality* of adjunction can be defined in two ways: (1) by requiring that all components of an auxiliary set adjoin to the *same elementary tree*, (2) by requiring that all components of an auxiliary set adjoin to the *same auxiliary*

set, not necessarily to the same elementary tree. The first type of locality does not add to the generative capacity of the MCTAG. The second type of locality does add to the weak generative capacity of the MCTAG; however, the resulting class of languages still falls within the class of "mildly context-sensitive" languages. With the second type of locality an MCTAG can be defined for the language $L' = \{a^n b^n | n \geq 1\}$ such that the a 's all hang from one path from the root node S and the b 's all hang from another path from the root node. Such a structural description cannot be provided by TAGs, where each auxiliary set has exactly one component (see also Joshi 1985). For further details of MCTAGs, see Weir 1988. Weir (1988) has also shown that MCTAGs (with the second definition of locality) are equivalent to LCFRSs (see section 6).

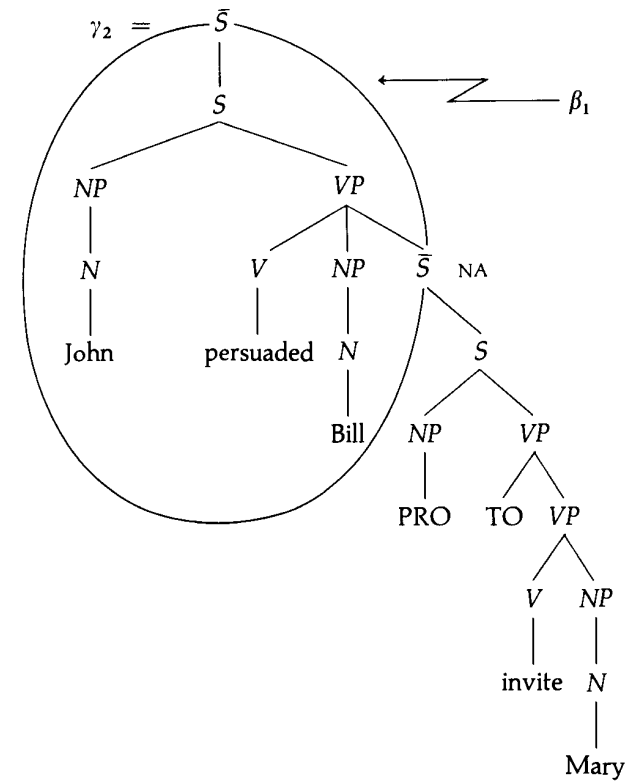
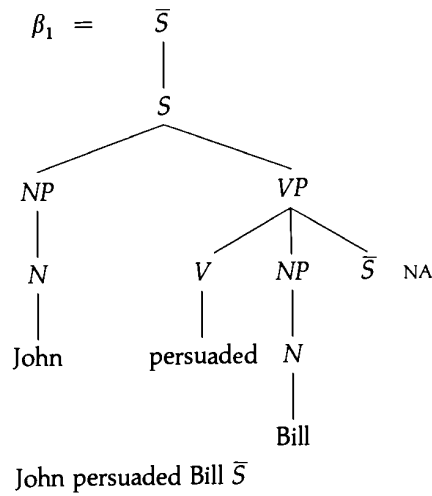
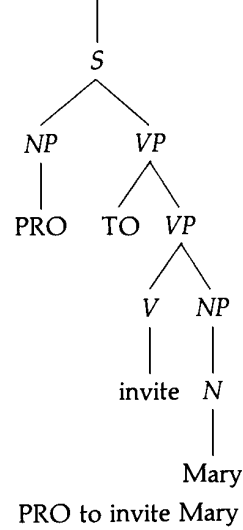
2.4 Some Linguistic Examples

EXAMPLE 6 Starting with the initial tree $\gamma_1 = \alpha_1$ and then adjoining β_6 at the indicated node (marked by *) in α_1 , we obtain γ_2 .



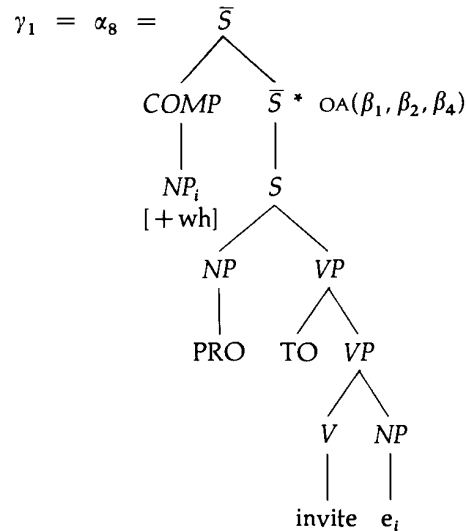
EXAMPLE 7 Starting with the initial tree $\gamma_1 = \alpha_6$ and adjoining β_1 at the indicated node in α_6 , we obtain γ_2 .

$$\gamma_1 = \alpha_6 = \bar{S} * \text{OA}(\beta_1, \beta_2, \beta_4)$$

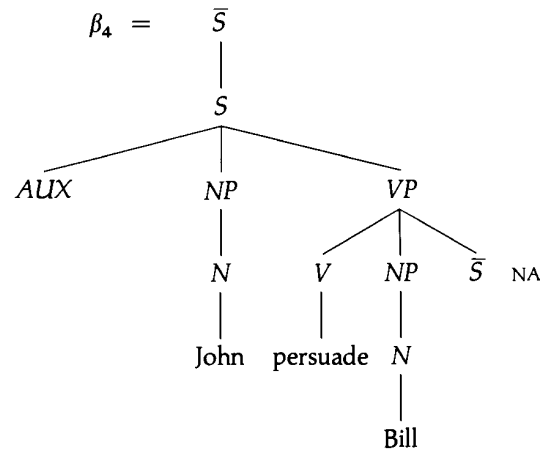


Since the initial tree α_6 is not a root sentence, it must undergo an adjunction at its root node, for example, by the auxiliary tree β_1 as shown above. Thus, for α_6 we have specified a local constraint $\text{OA}(\beta_1, \beta_2, \beta_4)$ for the root node, indicating that α_6 must undergo an adjunction at the root node by an auxiliary tree β_1 .

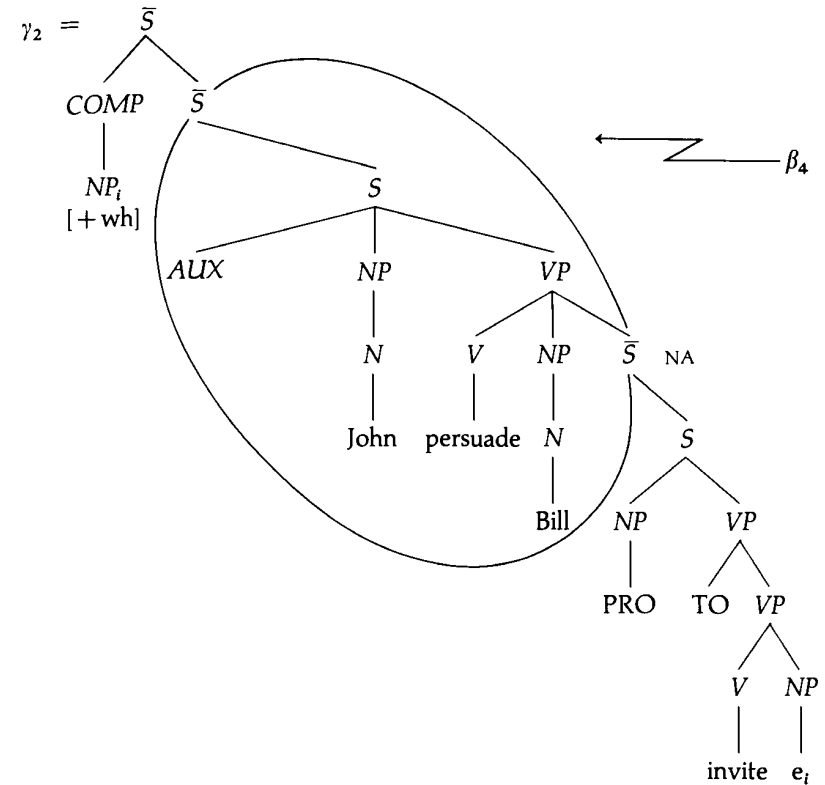
EXAMPLE 8 Starting with the initial tree $\gamma_1 = \alpha_8$ and adjoining β_4 to α_8 at the indicated node in α_8 , we obtain γ_2 .



who PRO to invite



did John persuade Bill $\bar{S}$



Who did John persuade Bill PRO to invite?

Note that the link in γ_1 is *preserved* in γ_2 ; it is *stretched*, resulting in a so-called unbounded dependency. Also note that, as in previous examples, α_8 is an initial tree that cannot serve as a root sentence and the obligatory adjunction possibilities are as indicated. Again the local constraint (NA) at the foot node of β_4 prevents further adjunction at this node in γ_2 .

3 Head Grammars: Head Wrapping and Tree Adjunction

In this section we will briefly describe *head grammars* (HGs) (Pollard 1984) and relate them to TAGs. (For further details, see Weir, Vijay-Shanker, and Joshi 1986; Vijay-Shanker, Weir, and Joshi 1986). HGs are not to be confused with head-driven phrase structure grammars (HPSGs) (Pollard 1985).

3.1 Head Grammars

Head grammars are string-rewriting systems like CFGs, but differ from them in that each HG string has a distinguished symbol corresponding to the

head of the string. These are therefore called *headed strings*. The formalism allows not only concatenation of headed strings but also so-called *head-wrapping* operations, which split a string on one side of the head and place another string between the two substrings. When we wish to explicitly mention the head, we use the notation $w_1 \bar{a} w_2$; alternatively, we simply denote a headed string by $\bar{w}$. Productions in an HG are of the form $A \rightarrow f(\alpha_1, \dots, \alpha_n)$ or $A \rightarrow \alpha_1$, where A is a nonterminal, α_i is either a nonterminal or a headed string, and f is either a concatenation or a head-wrapping operation. Roach (1987) has shown that there is a normal form for HGs that uses only the following operations:

$$LC1(u_1 \bar{a}_1 u_2, v_1 \bar{a}_2 v_2) = u_1 \bar{a}_1 u_2 v_1 \bar{a}_2 v_2$$

$$LC2(u_1 \bar{a}_1 u_2, v_1 \bar{a}_2 v_2) = u_1 \bar{a}_1 u_2 v_1 \bar{a}_2 v_2$$

$$LL1(u_1 \bar{a}_1 u_2, v_1 \bar{a}_2 v_2) = u_1 \bar{a}_1 v_1 \bar{a}_2 v_2 u_2$$

$$LL2(u_1 \bar{a}_1 u_2, v_1 \bar{a}_2 v_2) = u_1 \bar{a}_1 v_1 \bar{a}_2 v_2 u_2$$

$$LR1(u_1 \bar{a}_1 u_2, v_1 \bar{a}_2 v_2) = u_1 v_1 \bar{a}_2 v_2 \bar{a}_1 u_2$$

$$LR2(u_1 \bar{a}_1 u_2, v_1 \bar{a}_2 v_2) = u_1 v_1 \bar{a}_2 v_2 \bar{a}_1 u_2.$$

LC1 concatenates the two strings; the head of the resulting string comes from the first string. Similarly for LC2. LL1 inserts the second string into the first string to the right of the head of the first string; that is, the head of the first string is to the left. The head of the resultant string is the head of the first string. Similarly for LL2, LR1, and LR2.

Pollard's definition of headed strings includes the headed empty string $\bar{\lambda}$. However, the term $f(\bar{w}_1, \dots, \bar{w}_i, \dots, \bar{w}_n)$ is undefined when $\bar{w}_i = \bar{\lambda}$. This nonuniformity has led to difficulties in proving certain formal properties of HGs (Roach 1987). This difficulty can be removed by formulating HGs as follows.

Instead of headed strings, we will use so-called *split strings*. Unlike a headed string, which has a distinguished symbol, a split string has a distinguished *position* about which it may be split. There are three operations on split strings: W , $C1$, and $C2$. The operations $C1$ and $C2$ correspond to the operations $LC1$ and $LC2$ in HGs. They are defined as follows:

$$C1(w_1 \uparrow w_2, u_1 \uparrow u_2) = w_1 \uparrow w_2 u_1 \uparrow u_2$$

$$C2(w_1 \uparrow w_2, u_1 \uparrow u_2) = w_1 w_2 u_1 \uparrow u_2.$$

Since the split point is not a symbol (which can be split either to its left or to its right) but a position between strings, separate left and right wrapping operations are not needed. The wrapping operation, W , is defined as follows:

$$W(w_1 \uparrow w_2, u_1 \uparrow u_2) = w_1 u_1 \uparrow u_2 w_2.$$

It can be shown that this reformulation is equivalent to HG. We will use this reformulation in our further discussion.

3.2 Wrapping and Adjunction

The weak equivalence of HGs and TAGs is a consequence of the similarities between the operations of wrapping and adjunction. It is the roles played by the split point and the foot node that underlie this relationship. When a tree is used for adjunction, its foot node determines where the excised subtree is reinserted. The strings in the frontier to the left and right of the foot node appear on the left and right of the frontier of the excised subtree. As shown in figure 2.3, the foot node can be thought of as a position in the frontier of a tree, determining how the string in the frontier is split.

Adjunction, in this case, corresponds to wrapping $w_1 \uparrow w_2$ around the split string $v_1 \uparrow v_2$. Thus, the split point and the foot node perform the same role. The proof showing the equivalence of TAGs and HGs is based on this correspondence.

3.3 Inclusion of TAL in HL

We will briefly present a scheme for transforming a given TAG to an equivalent HG. We associate with each auxiliary tree a set of productions such that each tree generated from this elementary tree with frontier $w_1 X w_2$ has an associated derivation in the HG, using these productions, of the split string $w_1 \uparrow w_2$. The use of this tree for adjunction at some node labeled X can be mimicked with a single additional production that uses the wrapping operation.

For each elementary tree we return a sequence of productions capturing the structure of the tree in the following way. We use nonterminals that are named by the nodes of elementary trees rather than the labels of the nodes.

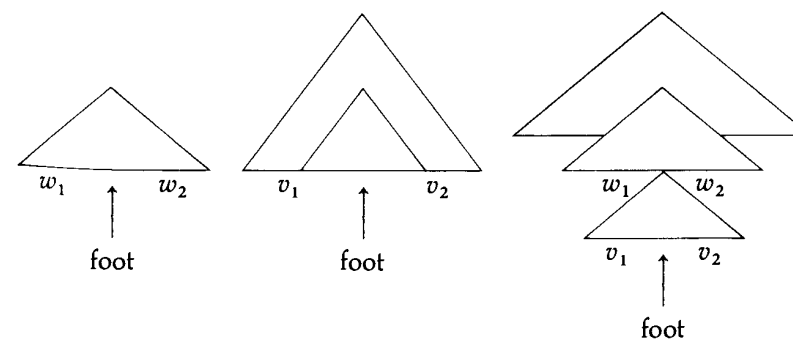


Figure 2.3
Wrapping and adjunction

For each node η in an elementary tree, we have two nonterminal nodes X_η and Y_η allowing for the possibility that an adjunction occurs at η : X_η derives the strings appearing on the frontier of trees derived from the subtree rooted at η ; Y_η derives the concatenation of the strings derived under each daughter of η . If η has daughters $\eta_1, \dots, \eta_k$, then we have the production

$$Y_\eta \rightarrow Ci(X_{\eta_1}, \dots, X_{\eta_k}),$$

where the node η_i dominates the foot node (by convention, we let $i = 1$ if η does not dominate the foot node). Adjunction at η is simulated by use of the production

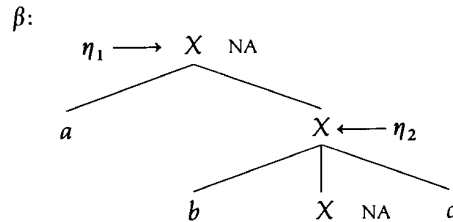
$$X_\eta \rightarrow W(X_\mu, Y_\eta),$$

where μ is the root of some auxiliary tree that can be adjoined at η . If adjunction is optional at η , then we include the production

$$X_\eta \rightarrow Y_\eta.$$

Notice that when η has an NA or OA constraint, we omit the second or third of the above productions, respectively.

We illustrate the construction with an example showing a single auxiliary tree and the corresponding HG productions. In this example, $\mu_1, \mu_2, \dots, \mu_n$ are the root nodes of the trees that can be adjoined at η_2 in β .



HG productions corresponding to β are

$$X_{\eta_1} \rightarrow Y_{\eta_1},$$

$$Y_{\eta_1} \rightarrow C2(a, X_{\eta_2}),$$

$$X_{\eta_2} \rightarrow W(X_{\mu_1}, Y_{\eta_2}),$$

$$X_{\eta_2} \rightarrow W(X_{\mu_n}, Y_{\eta_2}),$$

$$X_{\eta_2} \rightarrow Y_{\eta_2},$$

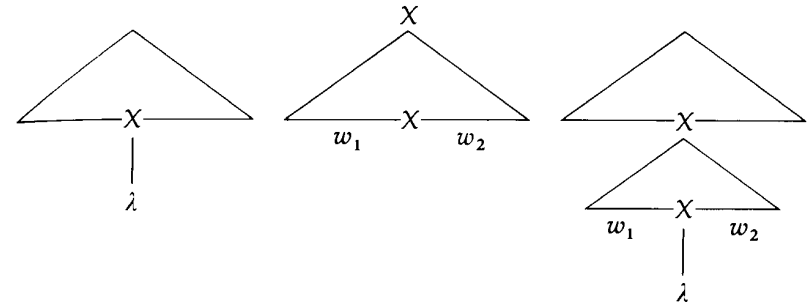
$$Y_{\eta_1} \rightarrow C2(b, X_{\eta_3}c),$$

$$X_{\eta_3} \rightarrow Y_{\eta_3},$$

$$Y_{\eta_3} \rightarrow \lambda.$$

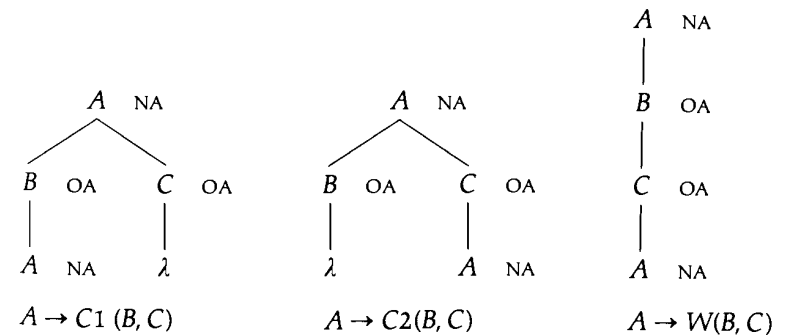
3.4 Inclusion of HL in TAL

In this construction, we use elementary trees to simulate directly the use of productions in HG to rewrite nonterminals. Generation of a derivation tree in string-rewriting systems involves the substitution of nonterminal nodes, appearing in the frontier of the unfinished derivation tree, by trees corresponding to productions for that nonterminal. From the point of view of the string languages obtained, tree adjunction can be used to simulate substitution, as illustrated in the following example.



Notice that although the node where adjunction occurs does not appear in the frontier of the tree, the presence of the node labeled by the empty string does not affect the string language.

For each production in the HG we have an auxiliary tree. A production in an HG can use one of the three operations C1, C2, and W. Correspondingly, we have the following three types of trees.



Drawing the analogy with string-rewriting systems: NA constraints at each root node ensure that a nonterminal is rewritten only once; NA constraints at the foot node ensure that, like the nodes labeled by λ , they do not contribute to the strings derived; and OA constraints ensure that every nonterminal introduced is rewritten at least once.

The two trees mimicking the concatenation operations differ only in the position of their foot node. This node is positioned in order to satisfy the

following requirement: for every derivation in the HG there must be a derived tree in the TAG for the same string, in which the foot is positioned at the split point.

The tree associated with the wrapping operation is quite different. The foot node appears below the two nodes to be expanded because the wrapping operation of HGs corresponds to the *LL1* operation of TAGs in which the head (split point) of the second argument becomes the new head (split point). Placement of the nonterminal that is to be wrapped above the other nonterminal achieves the desired effect, as described earlier.

Though straightforward, this construction does not capture the linguistic motivation underlying TAGs. The auxiliary trees directly reflect the use of the concatenation and the wrapping operations. Elementary trees for natural languages are constrained to capture meaningful linguistic structures. In the TAGs generated in the above construction, the elementary trees are incomplete in this respect, as reflected by the extensive use of the *OA* constraints. Since HGs do not explicitly give minimal linguistic structure in the sense of TAG, it is not surprising that such a direct mapping from HGs to TAGs does not recover this information.

3.5 Notational Differences between TAGs and HGs

TAGs and HGs are notationally very different, and this has a number of consequences that influence the way in which the formalisms can be used to express various aspects of language structure. The principal differences derive from the fact that TAGs are a tree-rewriting system, whereas HGs manipulate strings or pairs of strings.

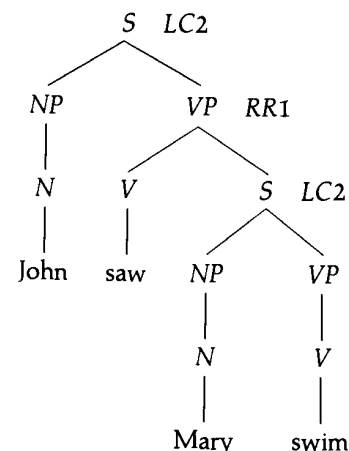
In order to be linguistically meaningful, the elementary trees in a TAG must conform to certain constraints that are not explicitly specified in the definition of the formalism. In particular, each elementary tree must constitute a *minimal linguistic structure* elaborated up to preterminal (terminal symbols and containing a head and all its complements or a modifier. Initial trees have essentially the structure of simple sentences; auxiliary trees correspond to minimal recursive constructions and generally constitute structures that act as modifiers of the category appearing at their root and foot nodes.

A hypothesis that underlies the linguistic intuitions of TAGs is that dependencies are captured within elementary trees. This is based on the assumption that elementary trees are the appropriate domain upon which to define dependencies, rather than, for example, productions in a CFG. Since in string-rewriting systems dependent lexical items cannot always appear in the same production, the formalism does not prevent the possibility that it may be necessary to perform an unbounded amount of computation in order to check that two dependent lexical items agree on certain features. However, since in TAGs dependencies are captured by

bounded structures, we expect that the complexity of this computation does not depend on the derivation. Features such as agreement may be checked within the elementary trees (instantiated up to lexical items) without need to percolate information up the derivation tree in an unbounded way. Some checking is necessary between an elementary tree and an auxiliary tree adjoined to it at some node, but this checking is still local and bounded. Similarly, elementary trees, being minimal linguistic structures, capture all of the subcategorization information.

TAGs have only one operation of composition, namely, adjunction. HGs have concatenation and a variety of types of wrapping. Further, TAGs differ from HGs in that they generate phrase structure trees. As a result, the elementary trees must conform to certain constraints such as left-to-right ordering and dominance relations. Unlike other string-rewriting systems that use only the operation of concatenation, HGs do not associate a phrase structure tree with a derivation: wrapping, unlike concatenation, does not preserve the word order of its arguments.

It is still possible to associate a phrase structure with a derivation in HGs that indicates the constituents, and we use this structure when comparing the analyses made by the two systems. These trees are not really phrase structure trees but rather trees with annotations that indicate how the constituents will be wrapped (or concatenated). It is thus a derivation structure, recording the history of the derivation. The following example illustrates how a constituent analysis is produced by a derivation in an HG, corresponding to the order *John Mary saw swim* as required in a Dutch subordinate clause.



Although a TAG generates trees (phrase structure trees), these trees are not the derivation trees. For a tree γ generated in a TAG, G , there is a

derivation structure associated with γ . Because TAG and HG are different systems, it is not possible to directly compare them with respect to their "strong" generative capacities. (In Weir, Vijay-Shanker, and Joshi 1986 we discuss a few linguistic examples comparing the structural descriptions provided by HG and TAG.) HG and TAG are comparable at the level of the derivation structures they produce. This aspect will be discussed in section 6.

4 Linear Indexed Grammars

Indexed grammars (IGs) were introduced by Aho (1968) as a generalization of CFG, and their mathematical properties have been investigated extensively. The class of indexed languages (ILs) is properly contained in the class of context-sensitive languages (CSLs) and properly contains CFLs. IGs were not introduced as grammatical formalisms for natural language; however, because IGs are more powerful than CFGs and because recent investigations have shown that additional power beyond that of CFGs is required, IGs have received attention from linguists. Gazdar (1985) discusses the relevance of IGs to natural language. The class of ILs as a whole is clearly larger than the class of so-called mildly context-sensitive languages (MCSLs), simply because ILs, in general, do not have the constant growth property, because not all ILs are semilinear. For example, $L = \{a^{n^2} | n > 1\}$, $L = \{a^{2^n} | n \geq 1\}$ are ILs, but are not semilinear.

IGs are defined as follows. (We will adopt the notation used in Gazdar 1985, which is essentially the same notation used in Hopcroft and Ullman 1979.) Let $A, B, C, \dots$ be the nonterminals; $a, b, c, \dots$ the terminals; $W, W_1, W_2, \dots$ strings of terminals and nonterminals; $i, j, k, \dots$ indices; $[]$, $[..]$, $[i..]$ stacks of indices, where $[]$ denotes an empty stack, $[..]$ a possibly empty stack, and $[i..]$ a stack whose topmost index is i . The productions are as follows:

1. $A[.] \rightarrow W[.]$
2. $A[.] \rightarrow B[i..]$
3. $A[i..] \rightarrow W[.]$

$W[.]$ is a shorthand for, for example, $A_1[.] \dots A_n[.]$; in other words, it stands for a right-hand side in which each nonterminal in W has $[.]$ associated with it.

Rule 1 copies the stack on A to all the nonterminal daughters. It is assumed by convention that no stacks are associated with the terminals. Rule 2 pushes an index i on the stack passed from A to a unique nonterminal daughter. Rule 3 pops an index i and then copies the stack on A to all the nonterminal daughters. Gazdar (1985) includes some additional rules, albeit redundantly. These are

4. $A[.] \rightarrow W_1[] B[.] W_2[]$
5. $A[.] \rightarrow W_1[] B[i..] W_2[]$
6. $A[i..] \rightarrow W_1[] B[.] W_2[]$

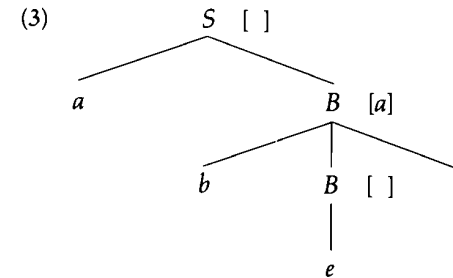
Rule 4 copies the stack on A to exactly one nonterminal daughter. Rules 5 and 6 push and pop an index on the stack of a designated daughter.

An IG that has only rules of the form 4, 5, and 6 will be called a *linear indexed grammar* (LIG). It can be shown that LIGs and TAGs, and therefore HGs, are weakly equivalent (Vijay-Shanker 1987). We will briefly illustrate the relationship between LIG and TAG by means of an example.

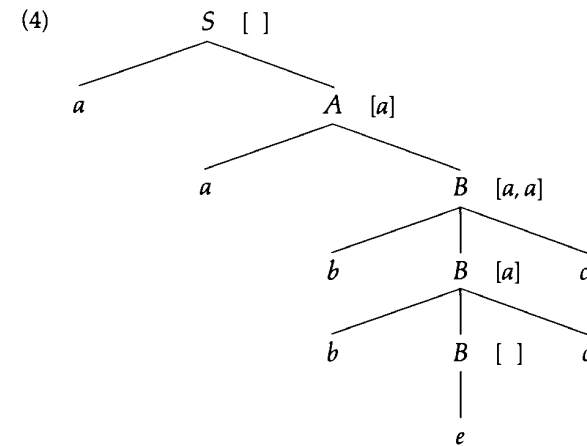
Let G be an LIG as follows:

1. $S[.] \rightarrow aA[a..]$
2. $S[.] \rightarrow aB[a..]$
3. $A[.] \rightarrow aA[a..]$
4. $A[.] \rightarrow aB[a..]$
5. $B[a..] \rightarrow bB[.]c$
6. $B[] \rightarrow e$

Let $w_1 = abec$. The derivation of w_1 is given in (3).

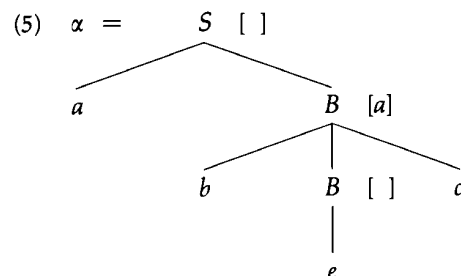


Let $w_2 = aabbec$. The derivation of w_2 is given in (4).

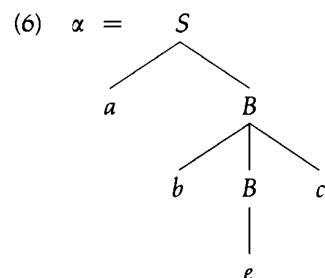


Comparing (3) and (4), one can see that $(n - 1)$ applications of rule 3, one application of rule 4, then n applications of rule 5 allows us to add n a 's, n b 's, and n c 's in the right order.

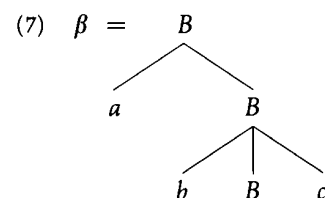
Note that (3) is a minimal derivation in G . We now take (3) to be an elementary tree (initial tree), say α , of a TAG G' , as shown in (5).



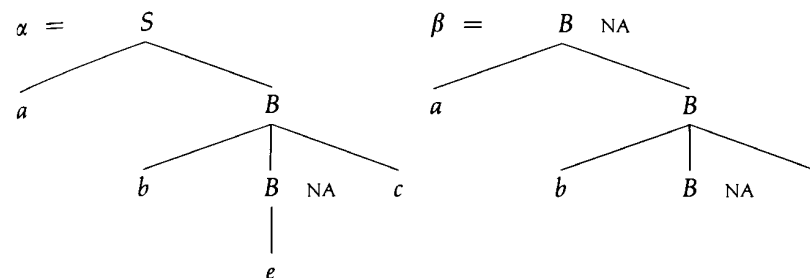
There is no need to keep the stacks that are associated with the non-terminals in α . Thus, we can have (6).



If we now introduce an auxiliary tree, say β , as in (7),



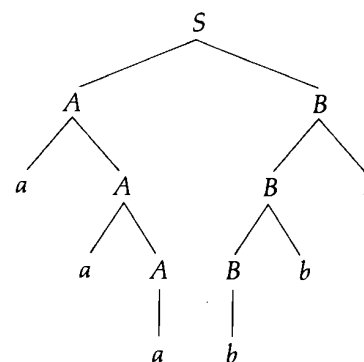
which introduces one a , one b , and one c , in the right order, we can simulate the effect of applying rules 3, 4, and 5 in LIG, assuming β is adjoined only to the node B that is the sibling of a . β should not be adjoined to the node B that dominates e in α . Similarly in β , since β can be adjoined to β itself, we want this adjunction to take place only to the interior B node of β , and not to the root and foot nodes of β . We can achieve this by placing the NA constraint at the appropriate nodes. Hence, the equivalent TAG, G' , is



The operation of adjunction indirectly pairs the pushes and pops of indices in just the right way. Also, this information is implicitly communicated from the root of an auxiliary tree to its foot, along its spine (that is, along the path from the root to the foot). This corresponds roughly to the LIG constraint that the stack passes on from the left-hand side of a rule only to a designated daughter.

The equivalence of LIG and TAG thus explains a conjecture made by Bill Marsh (referred to in Gazdar 1985) that the languages $L_1 = \{a^n b^n c^n d^n e^n | n \geq 1\}$ and $L_2 = \{www | w \in \{0, 1\}^*\}$ cannot be generated by LIG.

One variant of MCTAGs is more powerful than TAGs both weakly and strongly (as described in section 2.3). For example, MCTAGs can generate, for each $k \geq 1$, $L_3 = \{a_1^n, a_2^n, \dots, a_k^n | n \geq 1\}$ and can also generate the double-spined structural descriptions for $L_4 = \{a^n b^n | n \geq 1\}$, that is, structural descriptions of the form



LIGs are inadequate for this purpose, because they do not permit branching stacks. MCTAGs are capable of achieving some of the effects permitted by branching stacks; however, since they still maintain the constant growth property, they are more constrained. Whether there is any appropriate extension of LIG that permits branching stacks without leading to the full

power of IG is not known at present. Finding a subset of IG that is exactly or nearly equivalent to MCTAG is a challenging open problem.

Note that although the equivalence of LIG and TAG has been established, there are some key differences between them. If we consider the example described above, it is immediately clear that the "dependent" a , b , and c are always in the same elementary tree. This is only implicit in LIG. The dependency can only be inferred by examining the state of the stacks at the various stages of the derivation. This is due to the fact that TAGs factor recursion and dependency, whereas string-rewriting systems, of which LIG is an example, do not do so.

5 Combinatory Categorical Grammars

In this section we examine *combinatory categorical grammars* (CCGs), an extension of classical categorical grammars (Ajdukiewicz 1935) developed by Steedman and his collaborators (Ades and Steedman 1982; Steedman 1985, 1987, 1990). Classical categorical grammars are known to be weakly equivalent to CFGs (Bar-Hillel, Gaifman, and Shamir 1964), and the main result in this section is that under a certain definition (which corresponds to Steedman's recent work) CCGs are weakly equivalent to TAGs, HGs, and LIGs. We show this by showing that combinatory categorical languages (CCLs) are included in linear indexed languages (LILs) and that tree-adjoining languages (TALs) are included in CCLs (Weir and Joshi 1988).

On the basis of their weak equivalence with TAGs and HGs, it appears that CCGs should be classified as a mildly context-sensitive grammar formalism. The derivation tree sets traditionally associated with CCGs have context-free path sets; they are similar to those of LIGs and therefore differ from those of LCFRSs. This does not, however, rule out the possibility that there may be alternative ways of representing the derivation of CCGs that will allow them to be classified as LCFRSs.

The complexity of TAL recognition is $O(n^6)$. Thus, a corollary of our result is that this is also a property of CCLs. Although previous work has been done on the parsing of CCGs (Pareschi and Steedman 1987; Wittenburg 1986), no specific upper bound on recognition has been previously suggested.

5.1 Definition of CCGs

As defined here, CCG is the most recent version of a system that has evolved in a number of papers by Steedman (1985, 1987, 1990). In this section we first define CCGs and then show that the class of string languages generated by CCGs is equal to the languages generated by TAGs (and LIGs).

DEFINITION 1 A CCG, G , is denoted by (V_T, V_N, S, f, R) where

V_T is a finite set of terminals (lexical items),

V_N is a finite set of nonterminals (atomic categories),

S is a distinguished member of V_N ,

f is a function that maps elements of $V_T \cup \{\epsilon\}$ to finite subsets of $C(V_N)$ is the set of categories,¹ where

$$V_N \subseteq C(V_N) \text{ and if } c_1, c_2 \in C(V_N) \text{ then } (c_1/c_2) \in C(V_N) \text{ and } (c_1 \backslash c_2) \in C(V_N),$$

R is a finite set of combinatory rules as defined below.

There are four types of combinatory rules, which involve variables x , y , z over $C(V_N)$, and each $|_i \in \{\backslash, / \}$:

1. Forward application:

$$(x/y) \quad y \rightarrow x$$

2. Backward application:

$$y \quad (x \backslash y) \rightarrow x$$

3. Generalized forward composition for some $n \geq 1$:

$$(x/y) \quad (\dots(y|_1 z_1)|_2 \dots |_n z_n) \rightarrow (\dots(x|_1 z_1)|_2 \dots |_n z_n)$$

4. Generalized backward composition for some $n \geq 1$:

$$(\dots(y|_1 z_1)|_2 \dots |_n z_n) \quad (x \backslash y) \rightarrow (\dots(x|_1 z_1)|_2 \dots |_n z_n)$$

Restrictions can be associated with the use of the combinatory rules in R . These restrictions take the form of constraints on the instantiations of variables in the rules. These can be constrained in two ways:

1. The initial nonterminal of the category to which x is instantiated can be restricted.

2. The entire category to which y is instantiated can be restricted.

Derivations in a CCG involve the use of the combinatory rules in R . Let the *derives* relation be defined as follows:

$$\alpha c \beta \xRightarrow[G]{*} \alpha c_1 c_2 \beta,$$

if R contains a combinatory rule that has $c_1 c_2 \rightarrow c$ as an instance, and α and β are (possibly empty) strings of categories. The string language, $L(G)$, generated by a CCG, G , is defined as follows:

$$\{a_1 \dots a_n \mid S \xRightarrow[G]{*} c_1 \dots c_n, c_i \in f(a_i), a_i \in V_T \cup \{\epsilon\}, 1 \leq i \leq n\}.$$

Although there is no type-raising rule, its effect can be achieved to a limited extent since f can assign type-raised categories to lexical items. This is the scheme employed in Steedman's recent work.

5.2 Weak Generative Capacity

In this section we show that CCGs are weakly equivalent to TAGs, HGs, and LIGs. We do this by showing the inclusion of CCLs in LILs and the inclusion of TALs in CCLs. We have already seen that TAG and LIG are equivalent (and TAG and HG are equivalent) (Weir, Vijay-Shanker, and Joshi 1986). Thus, the two inclusions shown below imply the weak equivalence of all four systems (TAG, HG, LIG, and CCG).

5.2.1 CCLs $\subseteq$ LILs We describe how to construct an LIG, G' , from an arbitrary CCG, G , such that G and G' are equivalent. Let us assume that categories are written without parentheses, unless they are needed to override the left associativity of the slashes.

A category c is *minimally parenthesized* if and only if one of the following holds:

- $c = A$ for $A \in V_N$;
- $c = (A|_1c_1|_2 \dots |_nc_n)$, for $n \geq 1$, where $A \in V_N$ and each c_i is minimally parenthesized.

It will be useful to be able to refer to the *components* of a category c . We first define the immediate components of c :

When $c = A$, the immediate component is A ,

When $c = (A|_1c_1|_2 \dots |_nc_n)$, the immediate components are $A, c_1, \dots, c_n$.

The components of a category c are its immediate components together with the components of its immediate components. The immediate components are the category's arguments. Thus, $c = (A|_1c_1|_2 \dots |_nc_n)$ is a category that takes n arguments of category $c_1, \dots, c_n$ to give the *target* category A .

Although in CCGs there is no bound on the number of categories that are derivable during a derivation (categories resulting from the use of a combinatory rule), there is a bound on the number of *components* that derivable categories may have. This would no longer hold if unrestricted type raising were allowed during a derivation.

Let the set $D_C(G)$ be defined as follows:

- $c \in D_C(G)$ if c is a component of c' where
- $c' \in f(a)$ for some $a \in V_T \cup \{\epsilon\}$.

Clearly for any CCG, G , $D_C(G)$ is a finite set. $D_C(G)$ contains the set of all *derivable* components; that is, for every category c that can appear in a sentential form of a derivation in some CCG, G , each component of c is in $D_C(G)$. This can be shown, since, for each combinatory rule, if it holds of the categories on the left of the rule, then it will hold of the category on the right. The number of derivable *categories* is unbounded because they can have an unbounded number of immediate components.

Each of the combinatory rules in a CCG can be viewed as a statement about how a pair of categories can be combined. For the sake of this discussion, let us name the members of the pair according to their role in the rule:

The first of the pair in forward rules and the second of the pair in backward rules will be named the *primary* category. The second of the pair in forward rules and the first of the pair in backward rules will be named the *secondary* category.

As a result of the form that combinatory rules can take in a CCG, they have the following property. When a combinatory rule is used, there is a bound on the number of immediate components that the secondary categories of that rule may have. Thus, because immediate constituents must belong to $D_C(G)$ (a finite set), there is a bound on the number of categories that can fill the role of secondary categories in the use of a combinatory rule. Thus, there is a bound on the number of instantiations of the variables y and z_i in the combinatory rules in section 5.1. The only variable that can be instantiated to an unbounded number of categories is x . Thus, by enumerating each of the finite number of variable bindings for y and each z_i , the number of combinatory rules in R (while remaining finite) can be increased in such a way that only x is needed. Notice that x will appear only once on each side of the rules (in other words, they are linear).

We are now in a position to describe how to represent each of the combinatory rules by a production in the LIGs, G' . In the combinatory rules, categories can be viewed as stacks since symbols need only be added and removed from the right. The secondary category of each rule will be a ground category: either A or $(A|_1c_1|_2 \dots |_nc_n)$, for some $n \geq 1$. These can be represented in an LIG as $A[]$ and $A[|_1c_1|_2 \dots |_nc_n]$ respectively. The primary category in a combinatory rule will be unspecified except for the identity of its leftmost and rightmost immediate components. If its leftmost component is a nonterminal, A , and its rightmost component is a member of $D_C(G)$, c , this can be represented in an LIG by $A[\dots c]$.

In addition to mapping combinatory rules onto productions, we must include productions in G' for the mappings from lexical items:

- If $A \in f(a)$, where $a \in V_T \cup \{\epsilon\}$, then $A[] \rightarrow a \in P$.
- If $(A|_1c_1|_2 \dots |_nc_n) \in f(a)$, where $a \in V_T \cup \{\epsilon\}$, then $A[|_1c_1|_2 \dots |_nc_n] \rightarrow a \in P$.

We now illustrate this construction by giving an LIG from a CCG that generates the language

$$\{a^n b^n c^n d^n | n \geq 0\}.$$

EXAMPLE 9 Let a CCG be defined as follows, where we have omitted unnecessary parentheses:

$$\begin{aligned} A \in f(a) \quad B \in f(b) \quad C \in f(c) \quad D \in f(d) \\ S/S_1 \in f(\epsilon) \quad S_1 \in f(\epsilon) \quad S_1 \backslash A/D/S_1 \backslash B/C \in f(\epsilon) \end{aligned}$$

The following combinatory rules are permitted:

- Forward application involving x/y and y either when x begins with S_1 and y is C , or when x begins with S and y is S_1 or D .
- Backward application involving y and x/y when x begins with S and y is A or B .
- Forward composition involving x/y and $y \backslash z_1/z_2/z_3 \backslash z_4$ when x begins with S and y is S_1 .

The productions of the LIG that would be constructed from this CCG are as follows. The first seven rules result from the definition of f . (In the notation for stacks below, the top of the stack appears to the right. In section 4, in the notation for stacks, the top of the stack appears to the left. We have changed the notation from section 4 for convenience because the "top" of a category in CCGs is on the right also.)

$$\begin{aligned} A[] \rightarrow a \quad B[] \rightarrow b \quad C[] \rightarrow c \quad D[] \rightarrow d \\ S_1[] \rightarrow \epsilon \quad S_1 \backslash A/D/S_1 \backslash B/C \rightarrow \epsilon \quad S_1[.] \rightarrow S_1[./C]C[] \\ S[./S_1] \rightarrow \epsilon \quad S[.] \rightarrow S[./S_1]S_1[] \quad S[.] \rightarrow S[./D]D[] \\ S[.] \rightarrow A[]S[..\backslash A] \quad S[.] \rightarrow B[]S[..\backslash B] \\ S[..\backslash X_1/X_2/X_3 \backslash X_4] \rightarrow S[./S_1]S_1[\backslash X_1/X_2/X_3 \backslash X_4] \end{aligned}$$

for all $X_1, \dots, X_4 \in V_N$.

5.2.2 $TALs \subseteq CCLs$ We will just give the main idea of the construction of a CCG, G' , from a TAG, G , such that G and G' are equivalent. It is important to appreciate that the order in which categories are combined is crucial in a CCG derivation, since the same categories combined in different orders give different strings.

Each of the auxiliary trees will result in certain assignments of categories by f to a terminal or the empty string. Each occurrence of adjunction will be mimicked by the use of a combinatory rule.

Adjunction into nodes to the right (left) of the foot node (which corresponds to concatenation) will be simulated by backward (forward) application. Adjunction into nodes dominating the foot node of a tree (which corresponds to wrapping) will be simulated in the CCG by composition. It is necessary to ensure that the subsidiary category in every occurrence of

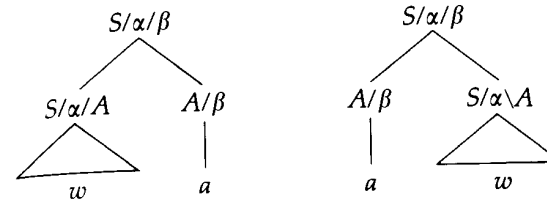


Figure 2.4
Context of composition

composition has just been introduced into the derivation by an assignment of f (see figure 2.4). This will correspond to the adjunction of an auxiliary tree that has not had any trees adjoined into it. It can be shown that composition is guaranteed only in this context.

Forward and backward application are restricted to cases where the secondary category is some X^a , and the left immediate component of the primary category is some Y^a .

Forward and backward composition are restricted to cases where the secondary category has the form $((X^c|_1 c_1)|_2 c_2)$, or $(X^c|_1 c_1)$, and the left immediate component of the primary category is some Y^a .

An effect of the restrictions on the use of combinatory rules is that the only categories that can fill the secondary role during composition are categories assigned to terminals by f . Notice that the combinatory rules of G' depend only on the terminal and nonterminal alphabet of the TAG and are independent of the elementary trees.

The construction depends on a particular normal form for TAGs. We will omit all the details here. The tree in figure 2.5 is encoded by the category

$$A \backslash A_1^a/A_2^c/A_3^a \backslash A_4^a.$$

EXAMPLE 10 Figure 2.6 shows an example of a TAG for the language $L_2 = \{a^n b^n | n \geq 0\}$ with crossing dependencies.

We give the CCG that would be produced according to this construction.

$$\begin{aligned} S^a \backslash S_1^a/S_2^c \in f(\epsilon) \quad S^c \backslash S_1^a/S_2^c \in f(\epsilon) \\ S_2^a/S^c/S_3^c \in f(\epsilon) \quad S_2^c/S^c/S_3^c \in f(\epsilon) \\ S_1^a \backslash A \in f(\epsilon) \quad S_1^c \backslash A \in f(\epsilon) \\ S_3^a/B \in f(\epsilon) \quad S_3^c/B \in f(\epsilon) \end{aligned}$$

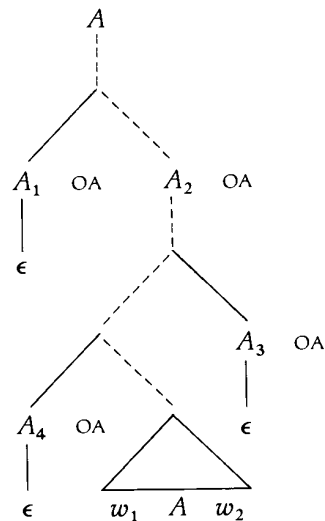


Figure 2.5
Tree encoding $A \setminus A_1^a / A_2^a / A_3^a \setminus A_4^a$

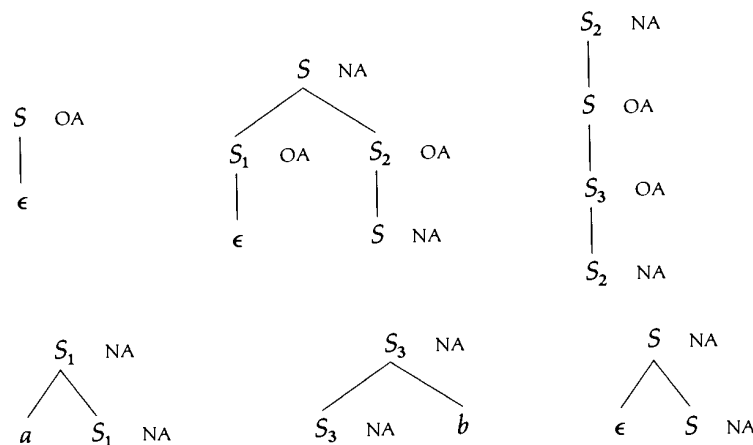


Figure 2.6
TAG for L_2

$$A \in f(a) \quad B \in f(b)$$

$$S^a \setminus S_\epsilon \in f(\epsilon) \quad S^\epsilon \setminus S_\epsilon \in f(\epsilon)$$

$$S_\epsilon \in f(\epsilon)$$

The CCGs produced according to the construction given here have the property that parentheses are redundant. It can be shown in general that the use of parentheses in CCGs does not increase the generative power of the formalism. Parenthesis-free CGs differ from CCGs (Friedman, Dai, and Wang 1986; Friedman and Venkatesan 1986). It can be shown that parenthesis-free CGs generate languages not generable by CCGs (Weir 1988).

For further details of this construction and a discussion of derivation trees, see Weir 1988.

6 Linear Context-Free Rewriting Systems

So far we have discussed a number of formalisms and shown equivalences among them. All these systems share certain properties that make them members of the so-called mildly context-sensitive grammar formalisms. All these systems involve some type of context-free rewriting. Vijay-Shanker, Weir, and Joshi (1987) and Weir (1988) have described a system, called *linear context-free rewriting system* (LCFRS), that attempts to capture the common properties shared by these formalisms.

6.1 Generalized Context-Free Grammars

We define *generalized context-free grammars* (GCFGs), first discussed, though with a somewhat different notation, by Pollard (1984).

DEFINITION 2 A GCFG, G , is written as $G = (V, S, F, P)$, where

- V is a finite set of variables,
- S is a distinguished member of V ,
- F is a finite set of function symbols, and
- P is a finite set of productions of the form

$$A \rightarrow f(A_1, \dots, A_n),$$

where $n \geq 0$, $f \in F$, and $A, A_1, \dots, A_n \in V$.

The set of terms (trees), $T(G)$, derived from a GCFG, G , is the set of all t such that $S \xrightarrow{*}_G t$, where the *derives* relation is defined as follows.

- $A \xrightarrow{*}_G f()$ if $A \rightarrow f()$ is a production.
- $A \xrightarrow{*}_G f(t_1, \dots, t_n)$ if $A \rightarrow f(A_1, \dots, A_n)$ is a production, and $A_i \xrightarrow{*}_G t_i$ for $1 \leq i \leq n$.

Notice that in GCFGs, rewriting choices during the derivation are independent of context. GCFG will generate a set of trees that can be interpreted as derivation trees in various grammar formalisms. Based on GCFG, we can now define a formalism called *linear context-free rewriting systems* (LCFRSs), which captures the common properties shared by the formalisms discussed earlier. By giving an interpretation for each of the functions in F , each term (tree) in $T(G)$ can be seen as encoding the derivation of some derived structure. It can be shown that CFGs, HGs, TAGs, and MCTAGs are example of LCFRSs.²

In each of the LCFRSs that we have given, the functions (combining strings, trees, or sequences of strings and trees) share certain constrained properties. It is difficult to completely characterize the entire class of such functions that will be so constrained because we are considering formalisms with arbitrary structures. Instead, we will give two restrictions on the functions. We would like these restrictions to ensure that the functions do not "copy," "erase," or "restructure" unbounded components of their arguments. The result of composing any two structures should be a structure whose "size" is the sum of its constituents plus some constant. Every intermediate structure that a grammar derives contributes some terminals to the string that is yielded by the structure that is finally derived. However, the symbols in the yield of an intermediate structure do not necessarily form a continuous substring of the final string. In general, though, we can write the yield of an intermediate structure as a finite sequence of substrings of the final string. The composition operations are "size" preserving. Thus, with respect to the yield of the structures being manipulated, the composition operations do no more than reorder their arguments and insert a bounded number of additional terminals. It can be shown that LCFRSs are semilinear (and hence obey the constant growth property) and are parsable in polynomial time. (For further details, see Vijay-Shanker, Weir, and Joshi 1987 and Weir 1988.) Weir (1988) has also shown that languages generated by MCTAGs are equal to the languages generated by LCFRSs.

Kasami, Seki, and Fujii (1988) have studied a system called multiple context-free grammars, which is the same as LCFRSs. They have obtained some additional properties of the classes of languages generated by their system; in particular, they have shown that the "nonerasing" property does not change the power.

7 Feature-Structure-Based Tree-Adjoining Grammars and Restricted Feature-Structure-Based Tree-Adjoining Grammars

In this section we briefly present *feature-structure-based TAGs* (FTAGs), where adjunction becomes function application and unification, and re-

stricted FTAGs (RFTAGs), where the feature structures associated with each node in an elementary tree are bounded.

7.1 FTAGs

The linguistic theory underlying TAGs is centered around the factorization of recursion and localization of dependencies into the elementary trees. The "dependent" items usually belong to the same elementary tree.³ Thus, for example, the predicate and its arguments will be in the same tree, as will the filler and the gap. Our main goal in embedding TAGs in a unificational framework is to capture this localization of dependencies. Therefore, we would like to associate feature structures with the elementary trees (rather than breaking these trees into a CFG-like rule-based system and then using some mechanism to ensure that only the trees produced by the TAG itself are generated).⁴ In the feature structures associated with the elementary trees, we can state the constraints among the dependent nodes directly. Hence, in an initial tree corresponding to a simple sentence, we can state that the main verb and the subject NP (which are part of the same initial tree) share the agreement feature.

In unification grammars, a feature structure is associated with a node in a derivation tree in order to describe that node and its relation to features of other nodes in the derivation tree. In a TAG, any node in an elementary tree is related to the other nodes in that tree in two ways. Feature structures written in FTAG using the standard matrix notation, describing a node, η , can be made on the basis of

1. The relation of η to its supertree (that is, the view of the node from the top). Let us call this feature structure t_η .
2. The relation of η to its descendants (that is, the view from below). This feature structure is called b_η .

Note that both the t_η and b_η feature structures are associated with the node η . In a derivation tree of a CFG-based unification system we associate one feature structure with a node (the unification of these two structures) since both the statements, t and b , together hold for the node, and no further nodes are introduced between the node's supertree and subtree. This property is not true in a TAG. On adjunction, at a node there is no longer a single node; rather, an auxiliary tree replaces the node. We believe that this approach of associating two statements with a node in the auxiliary tree is consistent with the spirit of TAGs. A node with OA constraints *cannot* be viewed as a single node and must be considered as something that has to be replaced by an auxiliary tree. t_η and b_η place restrictions on the auxiliary tree that must be adjoined at η . Note that if the node does not have an OA constraint, then we should expect t_η and b_η to be compatible (that is,

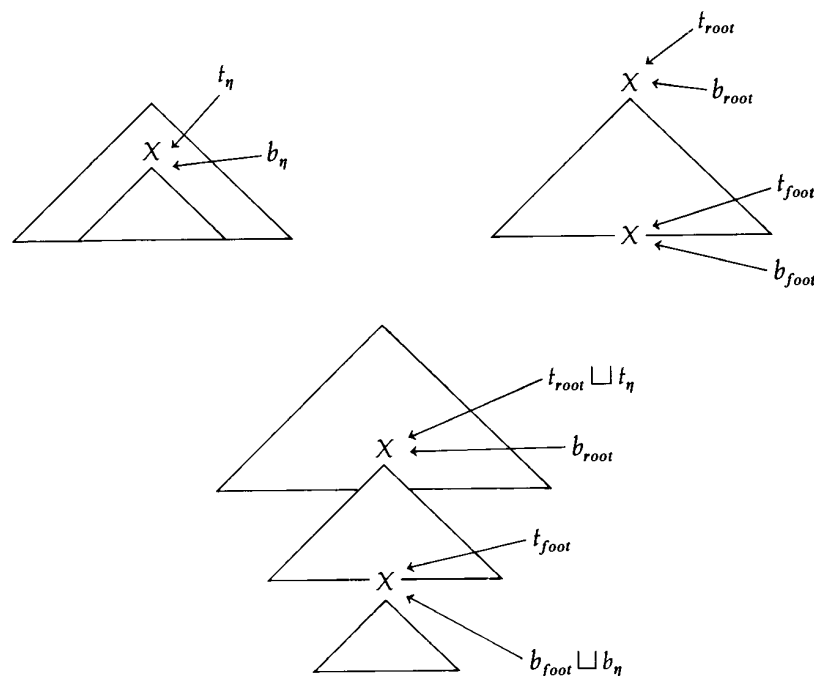


Figure 2.7
Feature structures and adjunction

unifiable). For example, in the final sentential tree, this node will be viewed as a single entity.

Thus, in general, with every internal node, η , at which adjunction can take place, we associate two structures, t_η and b_η . With each terminal node, we associate only one structure.⁵

Let us now consider the case when adjunction takes place as shown in figure 2.7. The notation we use is to write the t and b statements alongside each node, with the t statement written above the b statement. Let us say that t_{root} , b_{root} and t_{foot} , b_{foot} are the t and b statements of the root and foot nodes of the auxiliary tree used for adjunction at the node η . Based on what t and b stand for, it is obvious that on adjunction the statements t_η and t_{root} hold of the node corresponding to the root of the auxiliary tree. Similarly, the statements b_η and b_{foot} hold of the node corresponding to the foot of the auxiliary tree. Thus, on adjunction, we unify t_η with t_{root} , and b_η with b_{foot} . In fact, this adjunction is permissible only if t_{root} and t_η are compatible and b_{foot} and b_η are compatible as well. At the end of a derivation, the tree generated must not have any nodes with OA constraints. We check that by unifying the t and b feature structures of every node. More details of the

definition of FTAG may be found in Vijay-Shanker 1987, Vijay-Shanker and Joshi 1988, and Joshi and Vijay-Shanker, to appear.

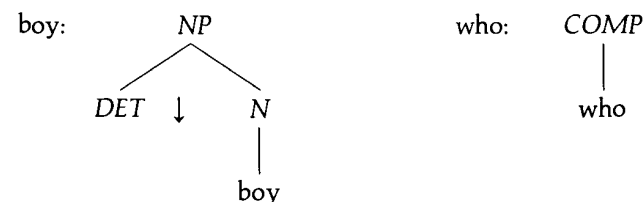
7.2 Restricted FTAGs

FTAGs as defined above are not constrained, just as CFG-based unification grammars are not constrained. However, if the feature structures associated with each node of an elementary tree are bounded, then RFTAGs can be shown to be equivalent to TAGs (Vijay-Shanker 1987). This restriction is the same as in GPSG; however, because of the larger domain of locality of TAGs and the operation of adjunction, RFTAGs are more powerful than GPSGs.

8 Lexicalized Tree-Adjoining Grammar

We call a grammar *lexicalized* if it consists of (1) a finite set of structures associated with each lexical item, which is intended to be the "head" of these structures, and (2) an operation or operations for composing these structures. The finite set of structures define the domain of locality over which constraints are specified, and these are local with respect to their "heads." It can be shown that, in general, a CFG cannot be lexicalized. Even if a CFG can be lexicalized, it is not always the case that we can guarantee that the lexical item associated with a structure is the linguistically appropriate "head." Both these results hold even if the domain of locality is extended to trees. This is so because the only operation available in CFG is substitution. If, however, we add adjunction as an operation, along with substitution, then we can appropriately lexicalize a CFG (see Schabes, Abeille, and Joshi 1988 for further details about lexicalized grammars, in particular, *lexicalized TAGs (LTAGs)*).

TAGs are "naturally" lexicalized because of their extended domain of locality and the operation of adjunction. In an LTAG we allow substitution in addition to adjunction. Adjunction can simulate substitution; however, in an LTAG we allow substitutions explicitly. The definitions of elementary trees are as before except that at the frontiers we can have nodes that are substitution nodes, in the sense that we have to substitute elementary or derived structures at the substitution nodes. Adjunction is defined as before. In the following example, the substitution nodes are marked by $\downarrow$.



the: DET
|
the

young:

N
/ \
ADJ N
|
young

saw:

S
/ \
NP ↓ VP
| \
V NP ↓
|
saw

S
/ \
NP ↓ VP
| \
V S
|
saw

S
/ \
COMP ↓ S
| \
NP ↓ VP
| \
V NP
| |
saw e

It can be shown that LTAGs are equivalent to TAGs. An Earley-type parser has been described in Schabes and Joshi 1988. For further details about LTAGs, see Schabes, Abeille, and Joshi 1988.

9 Tree-Adjoining Grammars—Local Dominance and Linear Precedence

9.1 TAG (LD/LP)

The extended domain of locality of TAGs has implications for how domination and precedence can be factored. We will now take the elementary trees of TAG as *elementary structures* (initial and auxiliary) specifying only domination structures over which linear precedences (LPs) can be defined. In fact, from now on we will define an *elementary structure* (ES) as consisting of the *domination structure plus a set of linear precedences*. Thus, α below is the domination structure of an ES.

$\alpha =$

S 0
/ \
NP 1 VP 2
/ \
V 2.1 NP 2.2

The addresses for nodes serve to identify the nodes. They are not to be taken as defining the tree ordering. They are just labels for the nodes.

Let LP_1^α be a set of linear precedence statements associated with α ,

$$LP_1^\alpha = \left[\begin{array}{l} 1 < 2 \\ 2.1 < 2.2 \end{array} \right],$$

where $x < y$ (x precedes y) if x and y are nondominating nodes (that is, x does not dominate y and y does not dominate x) and if x dominates z_1 and y dominates z_2 , then $z_1 < z_2$.

Note that LP_1^α corresponds exactly to the standard tree ordering. Given LP_1^α , (8) is the only terminal string that is possible with the ES (α, LP_1^α) , where α is the domination structure and LP_1^α is the linear precedence statement:

$$(8) \quad NP_1 V NP_2.$$

Suppose that instead of LP_1^α , we have

$$LP_2^\alpha = \left[\begin{array}{l} 1 < 2.1 \\ 2.1 < 2.2 \end{array} \right].$$

First note that in $1 < 2.1$, 2.1 is not a sister of 1. We can define precedences between nonsisters because the precedences are defined over α , the domain of locality.

Once again, the only terminal string that is possible with the ES (α, LP_2^α) is

$$(9) \quad NP_1 V NP_2,$$

but there is an important difference between (α, LP_1^α) and (α, LP_2^α) that will become clear when we examine what happens when an auxiliary tree is adjoined to α . Before we discuss this point, let us consider

$$LP_3^\alpha = \emptyset,$$

that is, the case where there are no precedence constraints. In this case we will get all six possible orderings:

$$NP_1 V NP_2, NP_1 NP_2 V, V NP_2 NP_1, NP_2 V NP_1, V NP_1 NP_2,$$

$$NP_2 NP_1 V.$$

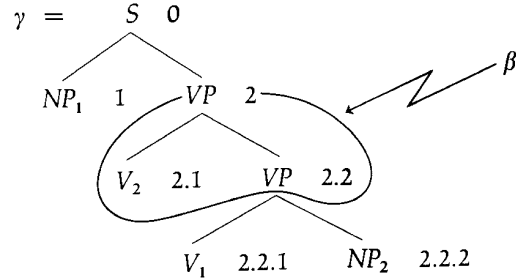
Let us return to (α, LP_1^α) and (α, LP_2^α) . As we have seen, both ESs give the same terminal string. Now let us consider an ES that is an auxiliary structure β (analogous to an auxiliary tree) with an associated LP, LP^β :

$$LP^\beta = [1 < 2].$$

We have

$$\beta = \begin{array}{c} VP \quad 0 \\ \swarrow \quad \searrow \\ V \quad 1 \quad VP \quad 2 \end{array} \quad LP^\beta = [1 < 2].$$

When β is adjoined to α at the VP node in α , we have



We have put indices on NP and V for easy identification. NP_1 , V_1 , NP_2 belong to α and V_2 belongs to β . If we have LP_1^α associated with α and LP^β with β , after adjunction the LPs are updated in the obvious manner:

$$LP_1^\alpha = \begin{bmatrix} 1 < 2 \\ 2.2.1 < 2.2.2 \end{bmatrix},$$

$$LP^\beta = [2.1 < 2.2].$$

The resulting LP for γ is

$$LP^\gamma = LP_1^\alpha \cup LP^\beta = \begin{bmatrix} 1 < 2 \\ 2.1 < 2.2 \\ 2.2.1 < 2.2.2 \end{bmatrix}.$$

Thus, γ with LP^γ gives the terminal string

$$(10) \quad NP_1 V_2 V_1 NP_2.$$

If we associate LP_2^α , rather than LP_1^α , with α , then after adjoining β to α as before, the updated LPs are

$$LP_2^\alpha = \begin{bmatrix} 1 < 2.1 \\ 2.2.1 < 2.2.2 \end{bmatrix},$$

$$LP^\beta = [2.1 < 2.2].$$

The resulting LP for γ is

$$LP^\gamma = LP_2^\alpha \cup LP^\beta = \begin{bmatrix} 1 < 2.2.1 \\ 2.1 < 2.2 \\ 2.2.1 < 2.2.2 \end{bmatrix}.$$

Thus, (γ, LP^γ) gives the terminal strings

$$(11) \quad NP_1 V_2 V_1 NP_2$$

and

$$(12) \quad V_2 NP_1 V_1 NP_2.$$

(11) is the same as (10), but in (12) NP_1 has "moved" past V_2 . If we adjoin β once more to γ at the node VP at 2, then with LP_1^α associated with α , we will get

$$(13) \quad NP_1 V_3 V_2 V_1 NP_2,$$

and with LP_2^α associated with α , we will get

$$(14) \quad NP_1 V_3 V_2 V_1 NP_2,$$

$$(15) \quad V_2 NP_1 V_3 V_1 NP_2,$$

and

$$(16) \quad V_3 V_2 NP_1 V_1 NP_2.$$

Let us consider another LP for α , say LP_3^α :

$$LP_3^\alpha = [1 < 2.1].$$

Then we have the following terminal strings for α (among others):

$$(17) \quad NP_1 V_1 NP_2,$$

$$(18) \quad NP_1 NP_2 V.$$

It can easily be seen that given LP_3^α associated with α and LP^β associated with β as before, after two adjunctions with β_1 , we will get

$$(19) \quad NP_1 V_3 V_2 V_1 NP_2,$$

$$(20) \quad NP_1 V_3 V_2 NP_2 V_1,$$

$$(21) \quad NP_1 V_3 NP_2 V_2 V_1,$$

$$(22) \quad NP_1 NP_2 V_3 V_2 V_1.$$

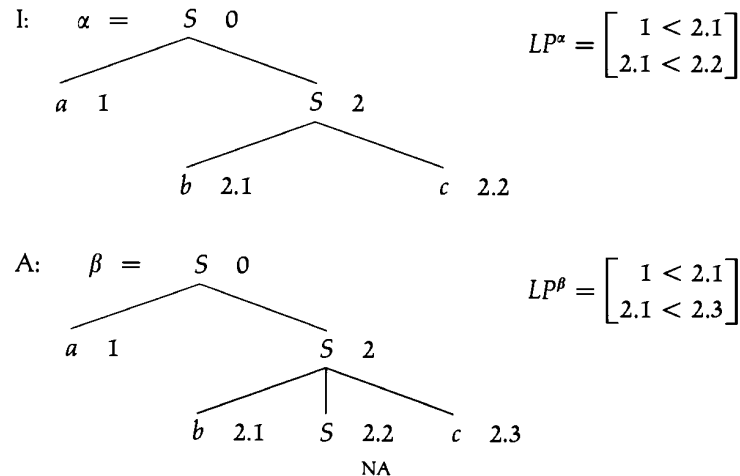
and, of course, several others. In (20), (21), and (22), NP_2 , the complement of V_1 in α , has "moved" past V_1 , V_2 , and V_3 respectively.

The ESs with their domination structure and the LP statements factor the constituency (domination) relationships from the linear order. The complex patterns arise due to the nature of the LP and the operation of adjunction. The main point here is that both the constituency relationships (including the filler-gap relationship) and the linear precedence relationship are defined on the ESs. *Adjunction preserves these relationships*. We have already seen in section 2 how the constituency relationships are preserved by adjunction. Now we have seen how the linear precedence relationships are preserved by adjunction. Thus, we have a uniform treatment of these two kinds of dependencies; however, the crucial difference between them clearly shows up in our framework. Joshi (1987b) has shown how this type of TAG can be used to account for several word order variations in Finnish, first described by Karttunen (1986).

The idea of factoring constituency (domination) relationships and linear order is basically similar to the immediate domination/linear precedence (ID/LP) format of GPSG. However, there are important differences: the domain of locality is the ES (and not the rewrite rules or local trees), and we have defined the LP for each ES. Of course, a compact description of LP over a set of ESs can easily be defined; but when it is compiled out, it will be in the form we have given here. We will call a TAG in which the (local) domination (LD) relationships and linear precedence (LP) relationships are factored out, a TAG in the LD/LP representation, or a *TAG(LD/LP)*.

In order to give further insight into the degree of word order variation possible in TAG(LD/LP), we will give an example.

Let $G = (I, A)$ be a TAG (LD/LP) where



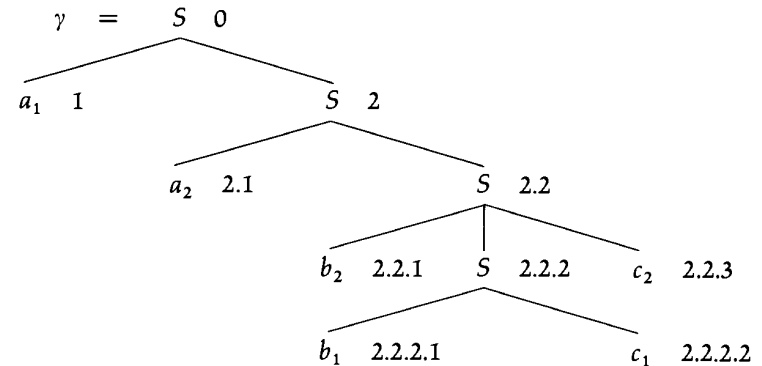
If G were a regular TAG (that is, if α and β were trees), then clearly the language generated by G , $L(G)$, is (as we have shown in section 2)

$$L(G) = \{a^n b^n c^n | n \geq 1\}.$$

However, G is a TAG(LD/LP). It is clear that the only string that corresponds to α is

abc .

If we adjoin β to α at the node 2 in α , we get



(We have indexed a 's, b 's, and c 's for convenience.) The updated LP^α and LP^β are

$$LP^\alpha = \begin{bmatrix} 1 < 2.2.2.1 \\ 2.2.2.1 < 2.2.2.2 \end{bmatrix},$$

$$LP^\beta = \begin{bmatrix} 2.1 < 2.2.1 \\ 2.2.1 < 2.2.3 \end{bmatrix}.$$

Hence,

$$LP^\gamma = LP^\alpha \cup LP^\beta$$

$$= \begin{bmatrix} 1 < 2.2.2.1 \\ 2.2.2.1 < 2.2.2.2 \\ 2.1 < 2.2.1 \\ 2.2.1 < 2.2.3 \end{bmatrix}.$$

Thus, some of the possible strings corresponding to γ are

- (23) $a_1 a_2 b_2 b_1 c_1 c_2$,
- (24) $a_1 a_2 b_2 b_1 c_2 c_1$,
- (25) $a_1 a_2 b_1 b_2 c_1 c_2$,

(26) $a_1 a_2 b_2 b_1 c_2 c_1$,

(27) $a_1 b_1 c_1 a_1 b_1 c_1$,

(28) $a_2 b_2 c_2 a_1 b_1 c_1$,

(29) $a_1 a_2 b_1 c_1 b_1 c_2$.

There are 20 strings in all. Each string contains 2 a 's, 2 b 's, and 2 c 's. The corresponding a 's, b 's, and c 's appear in the order a, b, c . This example shows that the elements of an auxiliary structure can "scatter" in a fairly complex manner over the elements of the ES to which it is adjoined, the complexity arising out of the specifications for LP and the adjunction operation itself.

If in the TAG(LD/LP) above both LP^a and LP^b are empty (that is, there are no LP statements), then the language generated by G is the so-called MIX language (or Bach language), which consists of strings of equal numbers of a 's, b 's, and c 's in any order. MIX can be regarded as the extreme case of free word order. It is not known yet whether TAG, HG, CCG, and LIG can generate MIX. This has turned out to be a very difficult problem. In fact, it is not even known whether an IG can generate MIX.

9.2 Languages of TAG(LD/LP)

It is clear that the languages of standard TAG (that is, when the elementary structures are trees with the standard tree ordering) are contained in the class of languages of TAG(LD/LP). Whether the containment is proper or not is not known at present. Languages of TAG(LD/LP) continue to satisfy the constant growth property, hence cannot generate languages of the form $L = \{a^n | n \geq 1\}$, $L = \{a^{2^n} | n \geq 1\}$, and so on.

Notes

1. Note that f can assign categories to the empty string, ϵ , though, to our knowledge, this feature has not been employed in the linguistic applications of CCG.
2. Weir (1988) has examined derivation trees associated with CCGs. Although the derivation trees traditionally associated with CCGs differ from those of LCFRSs, this does not preclude the possibility that there may be an alternative way of representing derivations. Weir (1988) gives a normal form for CCGs that allows them to be classified as LCFRSs.
3. It is sometimes possible for "dependent" items to belong to an elementary tree and the auxiliary tree that is immediately adjoined in it.
4. Such a scheme would be an alternative way of embedding TAGs in a unificational framework. However, it does not capture the linguistic intuitions underlying TAGs, and it loses the attractive feature of localizing dependencies.
5. It is possible to allow adjunctions at nodes corresponding to prelexical items. For example, we may wish to obtain verb clusters by adjunctions at nodes that are labeled as verbs. In such a case we will have to associate two feature structures with prelexical nodes as well.

References

- Ades, A. E., and M. J. Steedman. 1982. On the order of words. *Linguistics and Philosophy*, 3:517-558.
- Aho, A. V. 1968. Indexed grammars: An extension of the context-free grammars. *Journal of the ACM* 15:641-671.
- Ajdukiewicz, K. 1935. Die syntaktische Konnexität. *Studia Philosophica* 1:1-27. English translation in S. McCall, editor, *Polish Logic 1920-1939*. Oxford University Press, Oxford.
- Bar-Hillel, Y., C. Gaifman, and E. Shamir. 1964. On categorial and phrase structure grammars. In Y. Bar-Hillel, editor, *Language and information*. Addison-Wesley, Reading, MA.
- Berwick, R., and A. Weinberg. 1984. *The grammatical basis of linguistic performance*. MIT Press, Cambridge, MA.
- Bresnan, J. W., R. M. Kaplan, P. S. Peters, and A. Zaenen. 1982. Cross-serial dependencies in Dutch. *Linguistic Inquiry* 13:613-635.
- Culy, C. 1985. The complexity of the vocabulary of Bambara. *Linguistics and Philosophy* 8:345-351.
- Friedman, J., D. Dai, and W. Wang. 1986. The weak generative capacity of parenthesis-free categorial grammars. In *11th Intern. Conf. on Comput. Ling.*
- Friedman, J., and R. Venkatesan. 1986. Categorial and non-categorial languages. In *24th Meeting Assoc. Comput. Ling.*
- Gazdar, G. 1982. Phrase structure grammars. In P. Jacobson and G. Pullum, editors, *The nature of syntactic recognition*. D. Reidel, Dordrecht, Holland.
- Gazdar, G. 1985. Applicability of indexed grammars to natural languages. Technical report CSLI-85-34, Center for the Study of Language and Information, Stanford University.
- Gazdar, G., E. Klein, G. K. Pullum, and I. A. Sag. 1985. *Generalized Phrase Structure Grammar*. Basil Blackwell, Oxford. Also published by Harvard University Press, Cambridge, MA.
- Higginbotham, J. 1984. English is not a context-free language. *Linguistic Inquiry* 15:225-234.
- Hopcroft, J. E., and J. D. Ullman. 1979. *Introduction to automata theory, languages, and computation*. Addison-Wesley, Reading, MA.
- Joshi, A. K. 1985. How much context-sensitivity is necessary for characterizing structural descriptions: Tree adjoining grammars. In D. Dowty, L. Karttunen, and A. Zwicky, editors, *Natural language parsing: Psychological, computational and theoretical perspectives*. Cambridge University Press, New York. Originally presented in May 1983 at the Workshop on Natural Language Parsing at the Ohio State University.
- Joshi, A. K. 1987a. An introduction to tree adjoining grammars. In A. Manaster-Ramer, editor, *Mathematics of language*. John Benjamins, Amsterdam.
- Joshi, A. K. 1987b. Word-order variation in natural language generation. In *Proceedings of the Annual Conference of the American Association for Artificial Intelligence (AAAI-87)*, Seattle, July 1987.
- Joshi, A. K., and L. S. Levy. 1977. Constraints on structural descriptions: Local transformations. *SIAM Journal of Computing* 6:272-284.
- Joshi, A. K., L. S. Levy, and M. Takahashi. 1975. Tree adjunct grammars. *Journal of Computer and System Sciences* 10:136-163.
- Joshi, A. K., and K. Vijay-Shanker. To appear. Unification-based tree adjoining grammars. In J. Wedekind, editor, *Unification-based grammars*. MIT Press, Cambridge, MA.
- Kasami, T., H. Seki, and M. Fujii. 1988. *Generalized context-free grammars, multiple context-free grammars and head grammars*. Technical report, Department of Information and Computer Science, Osaka University, Osaka, Japan.

- Kroch, A. S. 1986. Asymmetries in long distance extraction in a tag grammar. In M. Baltin and A. S. Kroch, editors, *New conceptions of phrase structure*. University of Chicago Press, Chicago, IL.
- Kroch, A. S. 1987. Subacency in a tree adjoining grammar. In A. Manaster-Ramer, editor, *Mathematics of language*. John Benjamins, Amsterdam.
- Kroch, A. S., and A. K. Joshi. 1985. Linguistic relevance of tree adjoining grammars. Technical report MS-CIS-85-18, Department of Computer and Information Science, University of Pennsylvania, Philadelphia.
- Kroch, A. S., and A. K. Joshi. 1986. Analyzing extraposition in a tree adjoining grammar. In G. Huck and A. Ojeda, editors, *Syntax and semantics: Discontinuous constituents*. Academic Press, New York.
- Kroch, A. S., and B. Santorini. To appear. The derived constituent structure of the West Germanic verb-raising construction. In R. Freidin, editor, *Principles and parameters in comparative grammar*. MIT Press, Cambridge, MA.
- Pareschi, R., and M. J. Steedman. 1987. A lazy way to chart-parse with categorial grammars. In *25th Meeting Assoc. Comput. Ling.*
- Peters, S., and R. W. Ritchie. 1972. Context-sensitive immediate constituent analysis: Context-free languages revisited. *Mathematical Systems Theory* 6:324-333.
- Pollard, C. 1984. Generalized phrase structure grammars, head grammars and natural language. Doctoral dissertation, Stanford University.
- Pollard, C. 1985. Lecture notes on head-driven phrase-structure grammar. Center for the Study of Language and Information, Stanford University.
- Pullum, G. K. 1982. Free word order and phrase structure rules. In J. Pustejovsky and P. Sells, editors, *Proceedings of NELS 12*. GLSA, University of Massachusetts, Amherst.
- Roach, K. 1987. Formal properties of head grammars. In A. Manaster-Ramer, editor, *Mathematics of language*. John Benjamins, Amsterdam.
- Rounds, W. C. 1988. LFP: A logic for linguistic descriptions and an analysis of its complexity. *Computational Linguistics* 14.4:1-9.
- Schabes, Y., A. Abeille, and A. K. Joshi. 1988. Parsing strategies with "lexicalized" grammars: Applications to tree adjoining grammars. In *12th Intern. Conf. on Comput. Ling.*
- Schabes, Y., and A. K. Joshi. 1988. An Earley-type parsing algorithm for tree adjoining grammars. In *26th Meeting Assoc. Comput. Ling.*
- Shieber, S. M. 1985. Evidence against the context-freeness of natural language. *Linguistics and Philosophy* 8:333-343.
- Steedman, M. 1985. Dependency and coordination in the grammar of Dutch and English. *Language* 61:523-568.
- Steedman, M. 1986. Combinators and grammars. In R. Oehrle, E. Bach, and D. Wheeler, editors, *Categorial grammars and natural language structures*. Foris, Dordrecht, Holland.
- Steedman, M. 1987. Combinatory grammars and parasitic gaps. *Natural Language and Linguistic Theory* 5:403-439.
- Steedman, M. 1990. Gapping as constituent coordination. *Linguistics and Philosophy* 13:207-263.
- Vijay-Shanker, K. 1987. A study of tree adjoining grammars. Doctoral dissertation, University of Pennsylvania.
- Vijay-Shanker, K. and A. K. Joshi. 1985. Some computational properties of tree adjoining grammars. In *23rd Meeting Assoc. Comput. Ling.*
- Vijay-Shanker, K., and A. K. Joshi. 1988. Feature structure based tree adjoining grammars. In *Proceedings of the 12th International Conference on Computational Linguistics*. Budapest, Hungary.
- Vijay-Shanker, K., D. J. Weir, and A. K. Joshi. 1986. Tree adjoining and head wrapping. In *11th Intern. Conf. on Comput. Ling.*

- Vijay-Shanker, K., D. J. Weir, and A. K. Joshi. 1987. Characterizing structural descriptions produced by various grammatical formalisms. In *25th Meeting Assoc. Comput. Ling.*
- Weir, D. 1988. Characterizing mildly context-sensitive grammar formalisms. Doctoral dissertation, University of Pennsylvania. Also published as Technical Report MS-CIS-88-74, Department of Computer and Information Science, University of Pennsylvania.
- Weir, D. J., and A. K. Joshi. 1988. Combinatory categorial grammars: Generative power and relationship to linear context-free rewriting systems. In *26th Meeting Assoc. Comput. Ling.*
- Weir, D. J., K. Vijay-Shanker, and A. K. Joshi. 1986. The relationship between tree adjoining grammars and head grammars. In *24th Meeting Assoc. Comput. Ling.*
- Wittenburg, K. B. 1986. Natural language parsing with combinatory categorial grammar in a graph-unification based formalism. Doctoral dissertation, University of Texas at Austin.

Mark Steedman and Jason Baldridge

5.1 Introduction

Categorial Grammar (CG, Ajdukiewicz 1935; Bar-Hillel 1953) is one of the oldest lexicalized grammar formalisms, in which all grammatical expressions are distinguished by a syntactic type identifying them as either a function from arguments of one type to results of another, or as an argument. Such types, or *categories*, are closely related to the semantic type of the linguistic expression itself, differing mainly in the inclusion of information about language-specific linear order.

The earliest forms of CG were immediately recognized as being context-free and weakly equivalent to context-free phrase-structure grammars (CFPSG, Bar-Hillel, Gaifman and Shamir 1964). More or less contemporaneously with their development by Bar-Hillel, Lambek (1958) cast CG as a logical calculus, which was also widely (and correctly) assumed to be context-free, although the actual proof—due to Pentus (1993)—was much harder to discover.

The early evidence of weak equivalence to CFPSG led to a partial eclipse of CG in the 1960's. However, syntactician's interest in CG began to revive in the late 1970's and early 1980's. One reason for revived interest came from contemporary developments in formalizing a type-driven semantics for natural language in the work of Richard Montague (1974) and his followers (see Partee 1976), which made the syntactic/semantic type-transparency of CG attractive. Another reason was the realization that transformational generative grammar was overly expressive (Peters and Ritchie 1973), leading to a search for more minimal extensions of context free core grammars of various kinds (cf. Gazdar 1981), including CG.

The first extensions to CG were “combinatory” in nature, extending the core CG with functional operations on adjacent categories, such as “wrap” (Bach 1979; Dowty 1979) and functional composition (Ades and Steedman 1982). These developments in turn led to a revived interest in the type-logical al-

ternative stemming from Lambek's work in the late 1950's (see Oehrle, this volume).

The separation between combinatory and type-logical approaches has remained fairly sharp since these early developments. On the one hand, Combinatory Categorical Grammar (CCG) of the kind presented in this chapter has retained an active concern with keeping expressive power to the “mildly context sensitive” level identified by Joshi and his students (Joshi, Vijay-Shanker and Weir 1991), and has been actively involved with issues of linguistic explanation and practical computational linguistics, including wide-coverage parsing using statistical models. On the other hand, the Lambek tradition of type-logical grammars has been more concerned with theoretical issues and relations to logic, theorem proving, and complexity.

This chapter presents a formulation of CCG that goes some way toward reconciling this difference. While we retain the combinatory apparatus and low expressive power, we also incorporate the multi-modal approach developed for type-logical grammar as the sole means of constraining derivation in CCG. This move allows the rules of the system to be stratified and selectively used in lexically specified contexts, thereby removing the need for the category-based restrictions on combinatory rules used for this purpose in previous formulations of CCG.

We begin by motivating CCG in terms of the current state of linguistic theory and then outline the modalized version of the formalism. CCG is then applied to the bounded constructions (binding, reflexivization, heavy NP shift, dative shift, raising, object and subject control, and passive). Next, we give analyses for the unbounded constructions (including extraction, scrambling, and coordination) in a number of languages, including English, Dutch, Japanese, Turkish, and Irish Gaelic. Finally, we briefly consider intonation structure and parentheticalization in English and end with some remarks on implications for the theory of performance.

5.2 The Crisis in Syntactic Theory

The continuing need for volumes like the present one raises an obvious question: why are there so many theories of grammar around these days?¹ It is usual in science to react to the existence of multiple theories by devising a cru-

cial experiment that will eliminate all but one of them. However, this tactic does not seem to be applicable to these proliferating syntactic theories. For one thing, in some respects they are all rather similar. Sometimes the similarities are disguised by the level of detail at which the grammar is presented—for example, Tree-Adjoining Grammar (TAG, Joshi 1988) and CCG can be regarded as precompiling into lexical categories some of the feature-unification that goes on during derivations in Lexical-Functional Grammar (LFG, Bresnan 1982), Head-driven Phrase Structure Grammar (HPSG, Pollard and Sag 1994) and other attribute-value grammars. Nevertheless, (thanks to Reinhart and Reuland 1991, 1993 and Pollard and Sag 1992, 1994, who clarified the descriptive account considerably), all of the theories under discussion including CCG and at least some varieties of Government-Binding or Principles and Parameters grammar (GB) have essentially the same binding theory, with a lexically defined domain of locality corresponding to the tensed clause, and a command or scope relation defined at some level representing predicate argument structure, such as logical form. The mechanisms involved, even when couched in terms of transformations like “NP movement,” seem to be of rather low expressive power—essentially context-free (CF) and “base generable,” to use Brame's (1978) term. Many phenomena involving dependencies bounded by the tensed verbal domain, such as raising, control, passivization, reflexivization, and the like, have this character. While some deep problems remain—in particular, the question of what the primitive components of linguistic categories themselves are—the theories are all in formal terms pretty much alike in their analysis of these constructions.

It is really only when we consider the unbounded dependencies that cross the bounds of the tensed clause in constructions such as the relative clause, various kinds of “reduced” or “gapped” coordinate structures, and other “stylistic” constructions, that the theories differ in important ways. However, in most cases, the apparatus that is added to the CF core is sufficiently powerful and expressive that it is impossible to falsify or to distinguish any of the alternatives on grounds of expressive power. “Wh-movement” or equivalent coindexing of traces in GB, “functional uncertainty” or the ability to define dependencies in terms of paths defined as regular expressions in LFG, set-valued “SLASH features” in HPSG, certain classes of structural rules in Type Logical Grammar (TLG, Oehrle, this volume), are all examples of powerful mechanisms of this

kind. It is a measure of their expressive power that they have to be attended by seemingly arbitrary constraints on their operation which are strikingly similar to one or another of the constraints that limited classical transformational rules, such as the Coordinate Structure Constraint (Ross 1967) and the Fixed Subject Condition or “*that-trace*” filter, first identified by Bresnan (1972).

Constraints on rules are not necessarily in themselves a sign of anything wrong with a theory of grammar. They can arise from all kinds of extragrammatical sources, such as the requirements of semantics, the parser, or the language learner. (Island constraints like the Complex Noun Phrase Constraint of Ross 1967 provide an example of a group of constraints that should probably be explained in terms of probabilistically or semantically guided parsing rather than in terms of grammar as such.)

However, when a constraint is observed to hold cross-linguistically, as in the case of certain restrictions discussed below which relate coordinate constructions to primary word-order, that fact calls for *some* kind of explanation. One way to provide that explanation is to show that the constraints stem from limitations in automata-theoretic power of the grammar itself. A theory that is incapable in the first place of expressing grammars for languages that violate the condition provides a very convincing explanation for why they hold. Such a theory of grammar may also bring beneficial complexity and learnability results (although such theoretical results do not necessarily tell us much about the actual difficulty of practical processing and language learning for realistic grammars).

The project of explaining constraints on observed grammars as arising in part from grammar formalisms of low expressive power was the impulse behind Generalized Phrase Structure Grammar (GPSG, Gazdar 1981; Gazdar et al. 1985), which tried to capture as much as possible within a strictly context-free formalism. While it was clear from the start that phenomena existed that were unlikely to be capturable in this way, the effects of seeing just how many linguistic generalizations *could* be captured in context-free terms, supporting a fully compositional semantics, was extremely salutary. Most of all, it focused attention on multiple long range dependencies, since these required generalization of the mediating SLASH feature to be either a stack- or set-valued feature. In particular the fact that multiple dependencies in English show a tendency to nest rather than cross, as evidenced by the following minimal pair,

suggested that SLASH features should be stacks.

- (1) a. a violin which_i [this sonata]_j is hard to play_j upon_i
 b. *a sonata which_i [this violin]_j is hard to play_i upon_j

The two dependencies in (1a) must nest, rather than intercalate, as they would have to for (1b) to have a meaning to do with playing sonatas on violins (the asterisk here means “not allowed with the intended reading”).

However, the tendency to nest multiple dependencies is by no means universal. In certain Dutch constructions, multiple dependencies obligatorily intercalate (Huybregts 1976, 1984; Shieber 1985), as in the following example:²

- (2) ... omdat ik Cecilia Henk de nijlpaarden zag helpen voeren.
 ... because I Cecilia Henk the hippopotamuses saw help feed
-

‘... because I saw Cecilia help Henk feed the hippopotamuses.’

GPSG itself does not seem to have been particularly amenable to any restricted kind of generalization (although such a generalization is implicit in Pollard 1984 and Gazdar 1988), and constraining automata-theoretic power ceased to be a major focus of concern during its evolution into HPSG. However, a number of other formalisms, including TAG and CCG, continued to explore the possibility of capturing human grammars using low-power formalisms. In particular, Ades and Steedman (1982, 522) suggested that the *same* stack might be implicated both in the push-down automaton (PDA) characteristic of context-free grammar and in mediating multiple unbounded dependencies. Vijay-Shanker and Weir (1990, 1993, 1994) subsequently showed that all three formalisms were weakly equivalent to Linear Indexed Grammar, and delineated a new level in the Chomsky Hierarchy characterized by a generalization of the PDA, called an Extended Push Down Automaton (EPDA), which utilized a single stack of stack-valued features. Subsequent explorations with the TAG and CCG frameworks suggest that this level may be the lowest at which all syntactic phenomena of natural grammar can be captured.³

5.3 The Problem of Natural Language Grammar

In artificial languages like arithmetic, logic, and LISP, grammar is almost trivial. Syntax is essentially context-free. The only important exception is variable binding (Floyd 1962), which motivated the introduction of the indexed languages to the Chomsky hierarchy (Aho 1968, 1969 and see Gazdar 1988), a level or series of levels more expressive than context-free grammars, but less expressive than context-sensitive ones. Rules and constituents of syntax and semantics are essentially related one-to-one in these artificial languages. What little ambiguity they permit is strictly local (usually resolved by the adjacent symbol).

The fact that we might reasonably expect natural grammars to be the same is enshrined in “The Constituent Condition on Rules,” a principle which has always constituted a primary assumption of every linguistic theory:

- (3) The inputs and outputs of all rules of syntax should be constituents.

Since constituents are merely the units to which the grammar assigns meaning, this amounts to saying that rules of grammar should have a compositional semantics.

On this view, it is important to know what it is legitimate to regard as a constituent. Traditional linguistic tests for constituency are equivocal on this point. Chomsky (1955/1975, p.210-211) offers the following criteria:

- (4) a. The rule for conjunction
 b. Intrusion of parenthetical expressions
 c. Ability to enter into transformations
 d. Certain intonational features

The implication is that rules designed to capture these phenomena should all conform to the constituent condition.

There has been some success in keeping rules within the constituent condition with the bounded constructions and the “*wh*-movement” family of unbounded constructions under heading (c). Such constructions (in English at least) generally concern relations between quite uncontroversial constituent types, such as NPs, PPs and verbs. It has been much harder to adhere strictly to the constituent condition in analyses of the remaining three phenomena, which appear to involve relations between fragmentary entities of various kinds that

do not correspond straightforwardly to any traditional type of constituent.

For example, at first glance coordination looks very akin to *wh*-movement, as Gazdar (1981) pointed out. Thus, “right node raising” is unbounded (*contra* Ross 1967):

- (5) a. a book which *I hope I will write*, and *I fear that most people will burn without reading*
 b. *I hope I will write*, and *I fear that most people will burn without reading*, my book about the Theory of Everything.

It is also subject to similar, though not identical, “island conditions:”

- (6) a. *How many mathematicians do you think in ten derive a lemma?
 b. *Three mathematicians in ten derive a lemma and in a hundred prove completeness.

However, the residues of coordination in general are much more diverse than those of *wh*-movement:

- (7) a. I want to try to write, and *hope to see published*, a book about the Theory of Everything.
 b. Give Walt the salt, and *Malcolm the talcum*.
 c. I want to try to write a novel, and *you, a screenplay*.

All of these constructions appear (at least at first glance) to violate the constituent condition on rules.

The reaction of the early transformational grammarians, including Chomsky (1957) and Ross (1967), was to regard such conjunctions as arising from underlying or “deep structural” coordination of sentential constituents of type *S*, followed by processes of “deletion under coordination,” delivering surface form. While this expedient restored the constituent condition on rules as far as the inputs to rules of coordination were concerned, and ensured interpretability at the underlying level of representation, the outputs of deletion processes remained in violation of the constituent condition. Such rules were also of a disconcertingly arbitrary and unmotivated character. While they generated intense interest in early transformational studies, they proved extremely resistant to explanatory formal analysis, and in fact coordination has received comparatively little attention in transformational studies since the mid-’70s.⁴

Although there is much less work on parentheticals (but see Emonds 1979

and Taglicht 1998), the intrusion of parentheticals seem to isolate similar non-constituent fragments:

- (8) a. “*Are you now or have you ever been*,” demanded Harriet, “a member of the American Automobile Association?”
 b. “*You could give*,” soliloquized Harvey, “a policeman a flower.”

They also obey similar restrictions to coordinates:

- (9) *Three mathematicians, I conjecture, in ten derive a lemma.

Intonational phrases (for example, as defined by “continuation rises” here marked informally by right parentheses) can correspond to non-constituents (small caps indicate nuclear stress):

- (10) a. (The absent minded PROFESSOR)(was avidly READING)(about the latest BIOGRAPHY)(of Marcel PROUST).
 b. (Marcel PROVED) (COMPLETENESS).

Phonological processes like American English flapping and the Rhythm Rule are sensitive to such intonational constituency. (In both (10a) and (10b), the Rhythm Rule operates to move lexical stress from its normal position on the second syllable of *Marcel* to the first syllable, under the influence of the pitch accents on *PROUST* and *PROVED*.)

Intonational phrase boundaries obey similar restrictions to coordination, parentheticalization, and other “stylistic” constructions:

- (11) *(Three MATHEMATICIANS)(in ten derive a LEMMA).

This observation gave rise to the “Sense Unit Condition” on prosodic phrases of Halliday (1967), cf. Selkirk (1984), according to which such phrases have to be in some sense semantically interpretable.

In view of the similarities noted above between the restrictions on coordination and those on intonation structure, it would have been possible to regard the latter as stemming from the same grammatical processes (of deletion, movement, or whatever) as the former.⁵ However, Selkirk instead proposed incorporating an explicit level of Intonation Structure into a then-standard GB/P&P model of generative grammar, as shown in Figure 5.3 (adapted from Selkirk 1984—cf. p.205).

In according first-class structural citizenship to the intonational phrase, this is an appealing move. However, the resultant theory has some worrying as-

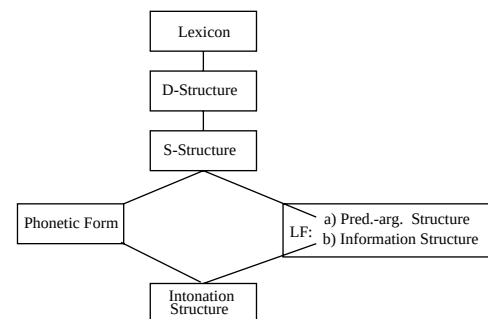


Figure 5.1
Architecture of a GB/P&P Generative Grammar

pects. The existence of a Sense Unit Condition on intonational phrases implies that they are syntactic in nature, for what is a theory of syntax if not a theory of sense units? Selkirk (1984) calls them “honorary constituents of the syntax,” and offers a definition in terms of headedness. In later work (Selkirk 1986, 1990), this condition is defined in “end-based” structural terms.

However, this means that the theory now has two parallel “tiers” of structure mediating between sound and meaning. Since such structural levels are what we call syntax, the grammar in effect has two orthogonal syntaxes. Moreover, since it deals with coordination and parentheticalization, phonological form (PF) also does significant strictly syntactic work in GB (Chomsky 1986 calls PF “Surface Structure”), and hence duplicates much of the work of Intonation Structure.

It would be in keeping with other contemporary developments to reduce the number of modules in the theory. In view of the similarity in constraints upon relativization, coordination, parentheticalization, and intonation structure, it seems possible that one could do this by combining PF, S-structure, and Intonation Structure in a single module. To do this we need a theory that stands traditional generative syntax on its head.

5.4 Combinatory Categorial Grammar

Combinatory Categorial Grammar (CCG), like other varieties of categorial grammar discussed by Oehrle, this volume, is a form of lexicalized grammar in which the application of syntactic rules is entirely conditioned on the syntactic type, or *category*, of their inputs. No rule is structure- or derivation-dependent.

Categories identify constituents as either *primitive categories* or *functions*. Primitive categories, such as N, NP, PP, S, and so on, may be regarded as further distinguished by features, such as number, case, inflection, and the like. Functions (such as verbs) bear categories identifying the type of their result (such as VP) and that of their argument(s)/complements(s) (both may themselves be either functions or primitive categories). Function categories also define the order(s) in which the arguments must combine, and whether they must occur to the right or the left of the functor. Each syntactic category is associated with a logical form whose semantic type is entirely determined by the syntactic category.

Pure CG (Ajdukiewicz 1935, Bar-Hillel 1953) limits syntactic combination to rules of functional *application* of functions to arguments to the right or left. This restriction limits expressivity to the level of context-free grammar, and CCG generalizes the context-free core by introducing further rules for combining categories. Because of their strictly type-driven character and their semantic correspondence to the simplest of the combinators identified by Curry and Feys (1958), these rules are called *combinatory* rules and are the distinctive ingredient of CCG, giving it its name. They are strictly limited to certain directionally specialized instantiations of a very few basic operations, of which the most important are *type-raising* and functional *composition*.⁶

Though early work in CCG focused primarily on phenomena in English and Dutch, grammar fragments capturing significant cross-linguistic generalizations have been constructed more recently in the framework (e.g., Turkish, Hoffman 1995; Japanese, Komagata 1999; Tzotzil, Trechsel 2000; Tagalog and Toba Batak, Baldridge 2002). In this chapter, we present basic aspects of analyses of English, Dutch, Japanese and Turkish, with a particular focus on a generalization for free word-order that leaves expressive power at the same mildly context-sensitive level as standard CCG. Finally, the problem of

parsing in the face of so-called spurious ambiguity is not only easily solvable with standard parsing methodologies, yielding processors which are of polynomial worst-case complexity and practicable average case complexity, as well as compatible with state-of-the-art probabilistic optimization (Hockenmaier and Steedman 2002b; Clark, Hockenmaier and Steedman 2002; Hockenmaier 2003a), but also directly compatible under the most restrictive assumptions possible with what is known about human sentence processing, as discussed in the chapter by Tanenhaus in the present volume.

5.4.1 Categorial Grammar

In CCG, as in other varieties of Categorial Grammar reviewed by Wood (1993) and exemplified in the bibliography below, syntactic information of the kind that can be captured for English in familiar context-free production rules like (12) is transferred to lexical entries like (13):

- (12) $S \rightarrow NP\ VP$
 $VP \rightarrow TV\ NP$
 $TV \rightarrow \{proved, finds, \dots\}$

- (13) $proved := (S \backslash NP) / NP$

This syntactic “category” identifies the transitive verb as a function, and specifies the type and directionality of its arguments and the type of its result. We here use the “result leftmost” notation in which a rightward-combining functor over a domain β into a range α are written α/β , while the corresponding leftward-combining functor is written $\alpha \backslash \beta$.⁷ α and β may themselves be function categories.

We follow Jacobson (1990, 1992), Hepple (1990) and Baldridge (2002); Baldridge and Kruijff (2003) (and depart from SP) in assuming that rules and function categories are “modalized,” as indicated by a subscript on slashes. Specifically, we assume that function categories may be restricted as to the rules that allow them to combine with other categories, via slashes typed with four basic modalities: $\star$, $\times$, $\diamond$, and $\cdot$. The effect of each of these modalities will be explicated as we introduce each of the combinatory rules and define their interaction with the modalities. The basic intent behind the modalities is as follows: the $\star$ modality is the most restricted and allows only the most basic applicative rules; $\diamond$ permits order-preserving associativity in derivations;

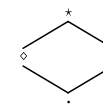


Figure 5.2: CCG type hierarchy for slash modalities.

$\times$ allows limited permutation; and $\cdot$ is the most permissive, allowing all rules to apply. The relation of these modalities to each other can be compactly represented via the hierarchy given in Figure 5.2.⁸

The essential effect of these modalities is to permit lexical control over CCG’s combinatory rules by defining the ability of functional categories to serve as input to only a subset of the available rules. Without modally typed slashes, language-specific restrictions or even bans on some combinatory rules are necessary in order to block certain ungrammatical word orders. With them, the combinatory rules are truly universal: the grammar of every language utilizes exactly the same set of rules, without modification, thereby leaving all cross-linguistic variation in the lexicon. As such, CCG is a *fully* lexicalized grammar formalism. See Baldridge (2002) and Baldridge and Kruijff (2003) for further discussion of the implications of the modal formulation of CCG.⁹

The most freely-combining types of slash $/$ and $\backslash$ allow a category to combine by any combinatory rule. The slashes in (13) are of this type. It will be convenient to abbreviate this type as a plain forward or backward slash, continuing to write such categories as before.

In order to allow functors such as (13) to combine with their arguments, we need combinatory rules, of which the two simplest are the following functional application rules:

- (14) *The functional application rules*

- a. $X / \star Y \Rightarrow X$ ($>$)
 b. $Y \backslash \star X \Rightarrow X$ ($<$)

Because $\star$ is the supertype of all other modalities, the $/\star$ and $\backslash\star$ slashes on these rules mean that *all* categories can combine by these rules.¹⁰

These rules have the form of very general binary phrase structure rule schemata. In fact, “pure” categorial grammar limited to these two rules alone is essentially context-free grammar written in the accepting, rather than

the producing, direction, with a consequent transfer of the major burden of specifying particular grammars from the PS rules to the lexicon. While it is now convenient to write derivations as in (15a), they are equivalent to conventional phrase structure derivations (15b):

(15) a. $\frac{\text{Marcel}}{NP} \frac{\text{proved}}{(S \setminus NP)/NP} \frac{\text{completeness}}{NP}$
 $\xrightarrow{S \setminus NP}$
 $\xleftarrow{S}$

b. $\frac{\text{Marcel}}{NP} \text{proved } \frac{\text{completeness}}{NP}$
 $\swarrow \quad \searrow$
 $\quad \quad VP$
 $\quad \quad \downarrow$
 $\quad \quad S$

It is important to note that such tree-structures are simply a representation of the process of derivation. They are not structures that need to be built by a processor, nor do they provide the input to any rules of grammar.

Despite this close correspondence, the categories labeling the nodes of the derivation in (15a) are much more informative than the atomic symbols in the tree (15b). Subcategorization is directly encoded in functor categories rather than through the use of new symbols such as $V_{intrans}$, V_{trans} and $V_{ditrans}$. Furthermore, there is a systematic correspondence between notions such as *intransitive* and *transitive* — after the transitive category $(S \backslash NP) / NP$ consumes its object argument, the resulting category $S \backslash NP$ is exactly that of an intransitive verb. This is a result of the way lexical categories are defined in combination with the universal rules of functional application.¹¹

Categories can be regarded as encoding the semantic type of their translation. This translation can be made explicit in the following expanded notation, which associates a logical form with the entire syntactic category, via the colon operator, which is assumed to have lower precedence than the categorial slash operators. (Agreement features are also included in the syntactic category, represented as subscripts, much as in Bach 1983. The feature 3s is “underspecified” for gender and can combine with the more specified 3*sm* by a standard unification mechanism that we will pass over here—see Shieber 1986,¹²

(16) $\text{proved} := (S \setminus NP_{3s}) / NP : \lambda x \lambda y. \text{prove}' xy$

We must also expand the rules of functional application in the same way:

(17) *Functional application*

- $X/*Y:f \quad Y:a \Rightarrow X:fa$ ($>$)
- $Y:a \quad X\backslash_*Y:f \Rightarrow X:fa$ ($<$)

They yield derivations like the following:

$$(18) \frac{\frac{\text{Marcel}}{NP_{3sm} : \text{marcel}'}}{\frac{\frac{\text{proved}}{S \backslash NP_{3s} / NP : \lambda x \lambda y. \text{prove}'xy} \quad \frac{\text{completeness}}{NP : \text{completeness}'}}{\frac{S \backslash NP_{3s} : \lambda y. \text{prove}' \text{completeness}'y}}{S : \text{prove}' \text{completeness}' \text{marcel}'}$$

The derivation yields the category S with a compositional interpretation, equivalent under a convention of left associativity to (19a):

(19) a. (prove'completeness')marcel' b.  b. prove' completeness' marcel'

Thus, the traditional subject-predicate structure reflecting c-command relations exhibited in (19b) is expressed at the level of propositional logical form or LF-structure.

5.4.2 Coordination

Coordination is captured in the present version of CCG via the following category (actually, category schema) for conjunctions like *and*, allowing constituents of like type to conjoin to yield a single constituent of the same type:¹³

(20) *The Conjunction Category*
and $:= (X \setminus_{\star} X) /_{\star} X$

The $\star$ modality on the slashes of this category restrict it to combine *only* by the application rules (14). It gives rise to derivations like the following:

$$\begin{array}{c}
 \text{(21) Marcel} \quad \text{conjectured} \quad \text{and} \quad \text{proved} \quad \text{completeness} \\
 \hline
 NP \quad (S \setminus NP) / NP \quad (X \setminus_* X) / {}_* X \quad (S \setminus NP) / NP \quad NP \\
 \hline
 \hline
 ((S \setminus NP) / NP) \setminus_* ((S \setminus NP) / NP) \\
 \hline
 (S \setminus NP) / NP \\
 \hline
 S \setminus NP \\
 \hline
 S
 \end{array}$$

5.4.3 Composition

In order to allow coordination of contiguous strings that are not standardly assumed to constitute constituents, CCG allows certain further operations on

functions related to Curry's combinators (Curry and Feys 1958). For example, functions may nondeterministically *compose*, as well as *apply*, under the following rules:¹⁴

(22) *The harmonic functional composition rules*

- $X/_\circ Y \quad Y/_\circ Z \Rightarrow X/_\circ Z \quad (>\mathbf{B})$
- $Y \backslash_\circ Z \quad X \backslash_\circ Y \Rightarrow X \backslash_\circ Z \quad (<\mathbf{B})$

The operation of these rules in derivations is indicated by an underline indexed $\triangleright \mathbf{B}$ or $\triangleleft \mathbf{B}$ respectively (because Curry called his composition combinator $\mathbf{B}$). The $\diamond$ modality means that only categories bearing that modality or the most general $\cdot$ modality (here abbreviated as plain slash) may compose. Categories bearing the incompatible $\times$ modality or the least general $\star$ modality (such as the conjunction category (20)) cannot combine by these rules.

Without the use of the hierarchy given in Figure 5.2 relating the various modalities, the forward composition rule would be stated with the following four instantiations:

(23) a. $X/\circ Y \quad Y/\circ Z \Rightarrow X/\circ Z$
 b. $X/\circ Y \quad Y/Z \Rightarrow X/Z$
 c. $X/Y \quad Y/\circ Z \Rightarrow X/\circ Z$
 d. $X/Y \quad Y/Z \Rightarrow X/Z$

We explain why only these four mixtures are utilized for $\mathbf{B}$ in section 5.5.

The effect of (22a) can be seen in the derivation of sentences like (24), which crucially involves the composition of two verbs to yield a composite of the same category as a transitive verb. It is important to observe that composition also yields an appropriate interpretation for the composite verb *might prove*, as $\lambda x \lambda y. \text{might}'(\text{prove}'x)y$, an object which if applied to an object *completeness* and a subject *Marcel* yields the proposition *might(prove'completeness')marcel'*. The coordination will therefore yield an appropriate semantic interpretation.¹⁵

tleties to be specified in meaning postulates that we do not specify.

5.6.1 Binding Theory

We define a structural notion of LF-command over logical forms like the one built in derivation (18), along lines set out more fully in Steedman 1996, (hereafter, *SSI*), and Steedman 2000b, (hereafter, *SP*).

(35) *LF-command*: a node α in a logical form Λ LF-commands a node β in Λ if the node immediately dominating α dominates β and α does not dominate β .

(The relation “dominates” is defined as the recursive transitive closure of the parent relation. The relation “immediately dominates” is defined as holding between the first branching node that dominates a node and that node.)

If the LF interpretations of bound pronouns and reflexive/reciprocal anaphors bound in a logical form Λ are defined as (non-branching) “pro-terms” of the form *pro*' x , *ana*' x , in which x is identical to some other node in Λ , then a binding theory much like that of Chomsky (1981) can be defined.¹⁷ For example, Condition C of the binding theory, which rules out *He_i likes John_i*, *He_i thinks Mary likes John_i*, etc., can be defined as follows:

(36) *Condition C*: No node except the argument in a pro-term may be LF-commanded by itself.

We shall see directly that Condition A of the binding theory follows immediately from Condition C and the assumption that reflexivization is lexicalized. Condition B, which says that pronouns must not be bound in their local tensed domain) is claimed in *SSI* to arise because pronominal anaphora is *not* lexicalized, but mediated by contextual update.

5.6.2 Reflexivization

Condition A says that reflexives etc. must be bound in their local tensed domain, excluding **Himself_i likes Harry_i*, **Harry_i thinks Sally likes himself_i*, and the like. This is naturally captured in CCG if we follow Reinhart and Reuland (1991, 1993) in assuming that reflexivization is lexicalized. (We also follow those authors and Pollard and Sag 1992 in assuming that English reflexives have logophoric homonyms that are pronominal and not subject to Condition A). That is to say that the pronoun *itself*, like all noun phrases, is type-raised.

Unlike most arguments, it is a clitic, like French *se*, which means that it is specialized to apply only to *lexical* verbal categories. Its category is as follows:¹⁸

(37) $\text{itself} := (S \backslash NP_{3sn}) \backslash \{(S \backslash NP_{3sn}) / NP\}_{LEX} : \lambda p \lambda y. p(\text{ana}'y)y$

It gives rise to derivations like the following

$$(38) \frac{\text{Theorem 1} \quad \text{proved} \quad \text{itself}}{\frac{S / (S \backslash NP_{3sn}) : \text{theorem1}' \quad \frac{S \backslash NP_{agr} / NP : \lambda x \lambda y. \text{prove}'xy \quad S \backslash NP_{3sn} \backslash \{(S \backslash NP_{3sn}) / NP\}_{LEX} : \lambda p \lambda y. p(\text{ana}'y)y}{S \backslash NP_{3sn} : \lambda y. \text{prove}'(\text{ana}'y)y}}{S : \text{prove}'(\text{ana}'\text{theorem1}')\text{theorem1}'}} \leftarrow$$

The logical form yielded by above the example conforms to Condition C (36). However, it should be observed that a category parallel to (37) that would license **Itself proved theorem 1* would be in violation of Condition C (36), and cannot therefore exist in any language. Moreover the binding captured in (19) is by definition limited to the domain of a lexical verb. Condition A is therefore captured without further stipulation.

5.6.3 Heavy Noun Phrase Shift

The availability of backward crossed composition (34b) allows us to account for the fact that most adjuncts and second arguments can invert order with the first argument of the verb, via derivations like the following:

$$(39) \frac{\begin{array}{c} \text{I} \quad \text{introduced} \quad \text{to Marcel} \quad \text{some very heavy friends} \\ S / (S \backslash NP) : \lambda p. p \text{ me}' \quad (S \backslash NP) / PP_{TO} / NP : \lambda x \lambda y \lambda z. \text{introduce}'yxz \quad S / (S / PP_{TO}) : \lambda q. q \text{ marcel}' \quad S \backslash (S / NP) : \lambda r. r \text{ friends}' \end{array}}{\frac{(S / PP_{TO}) / NP : \lambda x \lambda y. \text{introduce}'yx \text{ me}'}{S / NP : \lambda x. \text{introduce}'\text{marcel}'x \text{ me}'}} \leftarrow \text{B}^2 \quad \leftarrow \text{B} \times$$

Such derivations preserve the binding condition C at the level of logical form as required by the following:

(40) I introduced to each other some very heavy friends.

The crucial involvement of the type-raised PP category also predicts that preposition stranding will be incompatible with the HNPS construction, as in (a), despite the possibility of stranding *in situ* propositions as in (b), which involves the unraised *PP/NP* (traces are included to indicate intended readings):

- (41) a. *Who did you introduce to *t* your very heavy friends?
b. Who did you introduce your very heavy friends to *t*?

A derivation similar to (39), suggested as an exercise, allows adjuncts of type $S \backslash S$, $VP \backslash VP$, etc. to take part in the construction, as in the following example:

(42) I shall meet tomorrow some very heavy friends from Hoboken.

5.6.4 Dative Shift

The ditransitive category for a verb like *give* is as follows:¹⁹

(43) $\text{gave} := ((S \backslash NP) / NP) /_o NP : \lambda x \lambda y \lambda z. \text{give}'yxz$

The $\diamond$ modality on the first slash is incompatible with combination via the backward crossed composition rule (34b). Ditransitives therefore cannot undergo HNPS, ruling out sentences like the following, despite the fact that the type-raised accusative would otherwise permit a derivation analogous to (39):

(44) *I gave a book my very heavy friend from Hoboken.

The category is otherwise unremarkable, apart from the fact that the command relation of the two rightward arguments is *reversed* between the surface derivation (which is suggested as an exercise) and the logical form that results from category (43), in which x LF-commands y . This property (which, as we shall see in section 5.7.2, is universal in SVOX, VSOX, XOSV, and XOVX constructions) captures the following binding asymmetry between indirect and direct object for ditransitive verbs, via Condition C (36):

- (45) a. Marcel showed me myself_{S:show' (ana'me')me'marcel'}
b. *Marcel showed myself me_{S:show'me' (ana'me')marcel'}

The type-logical varieties of CG discussed by Oehrle in this volume typically eschew such “intrinsic” use of logical form (and indeed the entire account of binding offered here). Instead, such grammars typically reverse the order of indirect and direct object in the syntactic category of ditransitives,

- (62) a. proven := $(S_{EN} \setminus NP) / PP_{BY} : \lambda x \lambda y. prove' yx$
 b. given := $((S_{EN} \setminus NP) / PP_{BY} / NP : \lambda x \lambda y \lambda z. give' xzy)$
 c. persuaded := $((S_{EN} \setminus NP) / PP_{BY}) / (S_{TO} \setminus NP) : \lambda p \lambda y \lambda z. persuade' (p \text{ and } y) z$
 d. promised := $((S_{EN} \setminus NP) / PP_{BY}) / (S_{TO} \setminus NP) : \lambda p \lambda y \lambda z. promise' (p \text{ and } y) z$

The latter category licenses both *John was promised to be left alone by Marcel*, and **John was promised to leave by Marcel*. On the assumption that *to be left alone* has an agentless passive category $S_{EN}/NP : \lambda x. leave-alone'(and'x') someone'$, the respective logical forms are as follows:

- (63) a. *promise'*(*leave-alone'*(*ana'* *john'*)*someone'*) *john'* *marcel'*
 b. *promise'*(*leave'*(*ana'* *marcel'*)) *john'* *marcel'*

Both are impeccable in terms of the binding theory, so the anomaly of the latter must stem from aspects of semantic representation that are not addressed here.

5.7 The Unbounded Constructions

The effect of including rules corresponding to the combinators **B** and **T** is to induce a rebracketing and reordering calculus over the strings and derivations that the lexicon and application alone determine, in which every type-driven derivational step is guaranteed to project function argument relations correctly. The fact that the syntactic combinators can be lexically restricted by modally typed-slashes means that languages like English and Dutch, in which reordering is quite limited, can be captured without generating undesired scrambled word orders. A number of linguistic predictions follow.

5.7.1 Unbounded Extraction and the Across-the-Board Condition

Since complement-taking verbs like *think*, *VP*/_S, can in turn compose via rule (22a) with fragments like *Marcel proved*, *S*/_{NP}, derived as in (28), we correctly predict the fact that right-node raising is unbounded, as in (64a), and also provide the basis for an analysis of the similarly unbounded character of leftward extraction, as in (64b) (see *SSI* and Baldrige 2002 for details, including discussion of “*that*-trace” effects and other extraction asymmetries):

- (64) a. [I conjectured]_{S/NP} and [you think Marcel proved]_{S/NP} completeness.
 b. The result [which]_{(N_{KN})_k(S/NP)} [I conjectured]_{S/NP} and [you think Marcel proved]_{S/NP}.

The $\star$ modality on the conjunction category (20) means that it can combine like types *only* by the application rules (14). Hence, as in GPSG (Gazdar 1981), this type-dependent account of extraction and coordination, as opposed to the standard account using structure-dependent rules, makes the across-the-board condition (ATB) on extractions from coordinate structures a prediction or theorem, rather than a stipulation, as consideration of the types involved in the following examples will reveal:

- (65) a. The result [that_{(N\setminus_{\setminus N})_h(S/NP)} [[Harry conjectured]_{S/NP} and [Marcel proved]_{S/NP}]_{S/NP}]_{N\setminus_{\setminus N}}
 b. The result that_{(N\setminus_{\setminus N})_h(S/NP)} *[[Harry conjectured]_{S/NP} and [Marcel proved it]_S]
 c. The result that_{(N\setminus_{\setminus N})_h(S/NP)} *[[Harry conjectured it]_S and [Marcel proved]_{S/NP}]
 d. The result that_{(N\setminus_{\setminus N})_h(S/NP)} *[[amazed me]_{S\setminus NP} and [Marcel proved]_{S/NP}]

Lakoff (1986) suggested on the basis of examples first noticed by Ross 1967 and Goldsmith 1985 like *What did you go to the store and buy, How much beer can you drink and not get sick?, This is the stuff that those guys in the Caucasus drink every day and live to be a hundred*, that the coordinate structure constraint and the ATB exception are an illusion. This argument has recently been revived by Kehler (2002) and Asudeh and Crouch (2002). However, it has always also been argued (by Ross and Goldsmith, among others including Lakoff himself in an earlier incarnation) that these extractions involve another, non-coordinate, subordinating lexical category for “and”, and as such do not constitute counterexamples to the coordinate structure and ATB constraints after all. Among the arguments in support of this view are the presuppositional and volitional semantics of the sentences in question (and the absence of such overtones from true coordinates), and the fact that (as Postal 1998 points out), no other conjunctions support such extractions—cf. **What did you go to the store or buy, *How much beer can you drink or not get sick?, *This is the stuff that those guys in the Caucasus drink every day or live to be a hundred*.

It follows that the problematic extractions can naturally be handled in CCG by assigning to “and” additional independent categories supporting extraction from left and right conjunct respectively, and with an appropriate voli-

tional/causal semantics (omitted here) of the kind discussed by Lakoff, Kehler, and Asudeh and Crouch, here written as follows:²⁴

- (66) a. $\text{and} := ((VP/NP_{+ANT}) \backslash_{*}(VP/NP)) /_{*} VP$
 b. $\text{and} := ((VP/NP) /_{*}(VP/NP)) \backslash_{*} VP$

The possibility of such exceptional extractions therefore does not controvert the CCG or GPSG claim that the coordinate structure constraint and its exceptions arise from the “same types” requirement on coordinands, contrary to claims by Lakoff and Asudeh and Crouch.

5.7.2 Argument Cluster Coordination

This apparatus has also been applied to a wide variety of coordination phenomena, including English “argument-cluster coordination”, “backward gapping” and verb-raising constructions in Germanic languages, and English gapping. The first of these is illustrated by the following analysis, from Dowty (1988), in which the ditransitive verb category $(VP/NP)/_o NP$ is abbreviated as DTV , and the transitive verb category VP/NP is abbreviated as TV :²⁵

- [illegible]

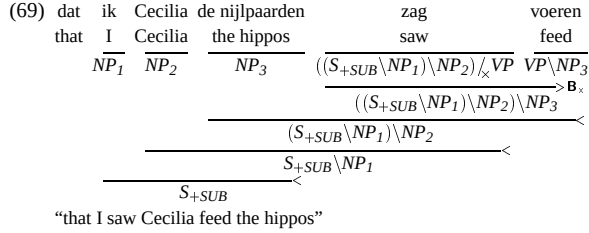
Since we have assumed the previously discussed rules of forward type-raising ($>\mathbf{T}$) and forward composition ($>\mathbf{B}$), this construction is correctly predicted to exist in English by arguments of symmetry, which imply that their backward varieties, $<\mathbf{T}$ and $<\mathbf{B}$ must also be assumed.

Given independently motivated limitations on type-raising, examples like the following are still disallowed:²⁶

- (68) *Three mathematicians [[in ten]_{PP} [derive a lemma]_{S\NP}] and [in a hundred prove completeness.]

5.7.3 “Mild Context Sensitivity”

The availability of *crossed* composition (34) to the grammar of Dutch under the Principles of Consistency and Inheritance allows crossed dependencies:



The $\times$ modality on the verb *zag* permits the forward crossed composition rule (34a) to apply (since the leftward slash of the verb *voeren* bears the $\cdot$ modality that is a subtype of $\times$), and also prevents harmonic composition of type-raised arguments with the Dutch main verb category. We pass over here the somewhat technical question of how *non-crossing* orders like (70) are excluded, merely noting that they depend on treating categories like *VP* as bundles of features or attribute-value pairs, and that such fine tuning is justified by the fact that other verbs, such as *proberen* (to try) do permit analogous orders (see *SP*, 145–153, and Baldridge 2002, 117).

(70) *dat ik Cecilia zag de nijlpaarden voeren.

The possibility of crossing dependencies immediately implies that CCG may be trans-context-free in terms of generative capacity. CCG is in fact provably weakly equivalent to TAG, Head Grammar (Pollard 1984), and Linear Indexed Grammar (Aho 1968), a group constituting the least expressive natural generalization of CFG that has so far been identified. This equivalence gives rise to a polynomial time worst-case complexity result (Vijay-Shanker and Weir 1990, 1993, 1994).

5.8 Scrambling

Many languages, such as Turkish and Japanese, permit more freedom in word order than languages like English and Dutch. The most basic expression of this is local scrambling, in which the arguments of a verb appear in permuted orders

within its clausal domain. This can be seen in the Turkish transitive sentence (71a) and its scrambled counterpart (71b), adapted from Hoffman (1995):

- (71) a. Ayse kitabi okuyor
 Ayse-NOM book-ACC read-PROG
 b. Kitabi Ayse okuyor
 book-ACC Ayse-NOM read-PROG
 ‘Ayse reads the book.’

Long distance scrambling, on the other hand, describes the appearance of an argument of a lower clause intermixed with the arguments of a higher clause. For example, the argument *kitabi* ‘book’ of the lower verb *okudugunu* ‘read’ scrambles out of its “base” position in (72a) into the matrix clause (72b) (from Hoffman 1995):

- (72) a. Fatma [Esra’nın kitabi okudugunu] biliyor.
 Fatma [Esra-GEN book-ACC read-GER-ACC] know-PROG
 ‘Fatma knows that Esra read the book.’
 b. *Kitabi*_i Fatma [Esra’nın _{t_i} okudugunu] biliyor.
book-ACC_i Fatma [Esra-GEN _{t_i} read-GER-ACC] know-PROG
 ‘As for the book, Fatma knows that Esra read it.’

The essential tension which arises in providing an analysis of *local* scrambling is that between utilizing base generation or devising a sufficiently liberal syntactic system. In CCG, base generation amounts to lexical ambiguity for verbs that allow scrambling. For example, if we assume the Turkish lexicon contains the two categories in (73) for *okuyor* ‘read’, both of the word orders in (71) are captured, as shown in derivations (74) and (75).

- (73) a. $okuyor := (S \backslash NP_{nom}) \backslash NP_{acc}$
 b. $okuyor := (S \backslash NP_{acc}) \backslash NP_{nom}$

- (74) Ayse kitabi okuyor
 $\overline{NP_{nom}} \quad \overline{NP_{acc}} \quad (S \backslash NP_{nom}) \backslash NP_{acc}$
 $\hline S \backslash NP_{nom} \xleftarrow{<}$
 $\hline S \xleftarrow{<}$

- (75) Kitabi Ayse okuyor
 $\overline{NP_{acc}} \quad \overline{NP_{nom}} \quad (S \backslash NP_{acc}) \backslash NP_{nom}$
 $\hline S \backslash NP_{acc} \xleftarrow{<}$
 $\hline S \xleftarrow{<}$

It may appear that using multiple categories as such fails to recognize the connection between the two orders; however, they can actually be generated from the specification of a single category, given a suitable theory of the lexicon. For example, one could assume that the category (73a) is the kernel category and use a lexical rule to generate (73b) from it. A more involved strategy is that advocated by Foster (1990), where unordered categories in the lexicon potentially project multiple ordered categories for use by the grammar. The difference between Foster’s strategy and one which uses lexical rules is that his approach does not require any language specific rules in order to create ordered categories from an unordered kernel category. This retains a tight connection between the different orders in a principled manner.

An alternative to multiple categories is to relax the definitions of categories and combinatory rules to allow a single category to project multiple word orders directly in syntactic combination. This is the strategy advocated by Hoffman (1995) to deal with scrambling in Turkish. She allows categories to contain multi-set arguments, as in (76), and redefines the combinatory rules to be sensitive to multi-sets, as shown for backward application in (77).²⁷ With this application rule, the category (76) can consume its arguments in either order.

- (76) $okuyor := S \{ \backslash NP_{nom}, \backslash NP_{acc} \}$

- (77) $Y \quad X(\alpha \uplus \{ \backslash Y \}) \Rightarrow X\alpha \quad (<)$

While this modified CCG apparatus suffices for local scrambling, it does not directly handle long distance scrambling. Accordingly, Hoffman redefines the type-raising and composition rules to work with such categories, but does so in a manner which increases the generative capacity of the system. In order to retain mild context-sensitivity while using multi-sets in categories, Baldridge (2002) provides more conservative definitions of the rules that permit the CCG system to use a flexible category in the same way as if it had access to an entire set of ordered categories that collectively capture the scrambled word orders. Then, the permutative powers already inherent in the crossed composition rules

can be utilized for long distance scrambling. For example, to derive (72b), the subject *Fatma* of the matrix verb must type-raise and forward cross compose into the verbal cluster in order for the derivation to proceed, as shown in (78).²⁸

$$(78) \begin{array}{c} \text{Kitabi} \quad \text{Fatma} \quad \text{Esra'nın} \quad \text{okudugunu} \quad \text{biliyor} \\ \hline \text{NP}_{acc} \quad \text{NP}_{nom} \quad \text{NP}_{gen} \quad S_{acc}\{\backslash \text{NP}_{gen}, \backslash \text{NP}_{acc}\} \quad S\{\backslash \text{NP}_{nom}, \backslash S_{acc}\} \\ \hline S / ((S \backslash \text{NP}_{nom}) \xrightarrow{T} S_{acc} \backslash \text{NP}_{acc}) \quad \xrightarrow{<} \text{B} \\ \hline (S \backslash \text{NP}_{nom}) \backslash \text{NP}_{acc} \xrightarrow{<} \text{B}_{\times} \\ \hline S \end{array}$$

Under this account, local scrambling is viewed as a clause-bounded phenomena, while long distance scrambling takes a form similar to other “extraction”-type phenomena, such as relativization.

The word order of Turkish is of course not entirely free. While verbal arguments can scramble around, the elements of some noun phrases are more fixed, as can be seen in (79):

- (79) a. [Siyah kedi] geldi
[black cat] come-PAST
'The black cat came in.'
- b. *Kedi siyah geldi
cat black come-PAST
- c. *Siyah geldi kedi
black come-PAST cat

Under the assumption that the category of *siyah* ‘black’ has a rightward slash, (79b) is obviously blocked. Modal control is crucial in the case of (79c); the slash must be non-permuting slash in order to bar backward crossed composition from applying:

$$(80) \begin{array}{c} \text{siyah} \quad \text{geldi} \quad \text{kedi} \\ \hline \text{NP}_{\times/\backslash} \text{NP}_{\times} \quad S \backslash \text{NP}_{nom} \quad \text{NP}_{nom} \\ \hline \end{array}$$

Turkish thus demonstrates the need for liberal access to the crossed composition rules at the clausal level while retaining tighter control over them at the phrasal level.²⁹ This type of control is needed for harmonic composition rules as well: for example, Baldridge’s (2002) analysis of syntactic extraction asym-

metries in Tagalog maintains tight control over forward harmonic composition while allowing local scrambling, and Trechsel (2000) utilizes restrictions with similar effects in Tzotzil.

With its universal rule set and lexical control over it via modally typed slashes, CCG supports these competing tensions straightforwardly and without recourse to powerful syntactic rules, transformations, or other devices.

5.9 Gapping and the Order of Constituents

The phenomenon of “argument cluster coordination” illustrated in (67) is an example of a much broader cross-linguistic generalization due to Ross (1970), concerning a relation between basic word-order parameters such as verb finality and constraints on deletion or gapping under coordination. While Ross originally framed his generalization in terms of a then-orthodox deep-structural word order, and on that basis it has been challenged, when reformulated in terms of surface constituent order, it appears to hold.

Ross (1970) noticed that direction of gapping (leftward or rightward) depends on basic constituent order:

- (81) a. SOV: *SOV and SO, SO and SOV
b. VSO: VSO and SO, *SO and VSO
c. SVO: SVO and SO, *SO and SVO

Apparent exceptions such as Zapotec (traditionally VSO) and Dutch (traditionally SOV) which allow both leftward and rightward gapping also have suspiciously mixed word-order, rendering the identification of “basic” word-order moot in those languages.

5.9.1 Gapping and SOV Word Order

On the assumption that type-raising is order-preserving, and defined over the Japanese SOV lexical type for transitive verbs like (82), the subject and object NP can not only combine with the verb by forward application, but also by forward composition, as in (83).

$$(82) \text{ tazuneta} := (S \backslash \text{NP}_{nom}) \backslash \text{NP}_{acc} : \lambda x \lambda y. \text{visit}'xy$$

$$(83) \begin{array}{c} \text{a. Ken-ga Naomi-o tazuneta.} \\ \text{Ken-NOM Naomi-ACC visit-PAST.CONCL} \\ \text{'Ken visited Naomi.'} \\ \\ \text{b.} \quad \begin{array}{c} \text{Ken-ga} \quad \text{Naomi-o} \quad \text{tazuneta} \\ \hline S / ((S \backslash \text{NP}_{nom}) \xrightarrow{T} (S \backslash \text{NP}_{nom}) / ((S \backslash \text{NP}_{nom}) \backslash \text{NP}_{acc}) \xrightarrow{T} (S \backslash \text{NP}_{nom}) \backslash \text{NP}_{acc}) \\ \hline S / ((S \backslash \text{NP}_{nom}) \backslash \text{NP}_{acc}) \xrightarrow{>} \text{B} \\ \hline S \end{array} \end{array}$$

The resulting nonstandard constituent *Ken-ga Naomi-o* can therefore conjoin, in a mirror image of the English derivation (67):

$$(84) \begin{array}{c} [\text{Ken-ga Naomi-o}], [\text{Erika-ga Sara-o}] \quad \text{tazuneta.} \\ S / ((S \backslash \text{NP}_{nom}) \backslash \text{NP}_{acc}) \quad S / ((S \backslash \text{NP}_{nom}) \backslash \text{NP}_{acc}) \quad (S \backslash \text{NP}_{nom}) \backslash \text{NP}_{acc} \\ \text{Ken-NOM Naomi-ACC Erika-NOM Sara-ACC visit-PAST.CONCL} \\ \text{'Ken visited Naomi, and Erika, Sara.'} \end{array}$$

Ditransitives similarly allow larger argument clusters (to save space the intransitive category is abbreviated *VP*, the transitive *TV*, and the ditransitive *DTV*):

$$(85) \begin{array}{c} \text{a. Kyooju-ga komonjo-o gakusee-ni kasita.} \\ \text{Professor-NOM manuscript-ACC student-DAT lent-PAST.CONCL} \\ \text{'The professor lent the manuscript to the student.'} \\ \\ \text{b.} \quad \begin{array}{c} \text{Kyooju-ga} \quad \text{komonjo-o} \quad \text{gakusee-ni} \quad \text{kasita.} \\ \hline S / \text{VP} \xrightarrow{T} \quad \text{VP} / \text{TV} \xrightarrow{T} \quad \text{TV} / \text{DTV} \xrightarrow{T} \quad \text{DTV} \\ \hline S / \text{TV} \xrightarrow{>} \text{B} \\ \hline S / \text{DTV} \xrightarrow{>} \text{B} \\ \hline S \end{array} \end{array}$$

In this case there is another derivation for the argument cluster:

$$(86) \begin{array}{c} \text{Kyooju-ga komonjo-o gakusee-ni kasita.} \\ \hline S / \text{VP} \xrightarrow{T} \quad \text{VP} / \text{TV} \xrightarrow{T} \quad \text{TV} / \text{DTV} \xrightarrow{T} \quad \text{DTV} \\ \hline \text{VP} / \text{DTV} \xrightarrow{>} \text{B} \\ \hline S / \text{DTV} \xrightarrow{>} \text{B} \\ \hline S \end{array}$$

Again, both derivations are guaranteed to yield identical logical forms, and all non-standard argument cluster constituents formed in both derivations can coordinate.

Ross's generalization that SOV verbs gap on the left conjunct is therefore captured: the Principles of Adjacency, Consistency, and Inheritance, together with the order-preserving constraint on type-raising that is by definition in an order-dependent language, permit any raised categories or rules of composition that would produce a *leftward*-looking function. “Forward Gapping” is therefore disallowed in any language with a pure verb-final lexicon:

- (87) *Ken-ga Naomi-o tazuneta, Erika-ga Sara-o
Ken-NOM Naomi-ACC visit-PAST.ADV Erika-NOM Sara-ACC
'Ken visited Naomi, and Erika, Sara.'

Dutch, which is often regarded as an SOV language, *does* allow coordinations on the above pattern in subordinate-clause conjunctions, in apparent exception to Ross' generalization:

- (88) ...dat Maaïke aardappels eet en Piet bonen
... that Maaïke potatoes eats and Piet beans
'... that Maaïke eats potatoes and Piet beans.'

However, this exception is clearly related to the fact that Dutch has VSO/SVO word order as well as SOV. In CCG terms, this corresponds to the fact that its lexicon is not *purely* SOV, and that main verbs must be assumed to bear VSO categories. Ross' generalization should therefore be rephrased, as in *SP*, in terms of surface order, i.e. available lexical type(s), not a single underlying order. Indeed, CCG rejects the very notion of “underlying word order.”

This observation is relevant to the fact that Japanese also allows OSV word order, as in (89):

- (89) Naomi-o Ken-ga tazuneta.
Naomi-ACC Ken-NOM visit-PAST.CONCL
'Ken visited Naomi.'

OS order can also give rise to constituent cluster coordination parallel to (84), as in (90):³⁰

- (90) [Naomi-o Ken-ga,] [Sara-o Erika-ga] tazuneta
 $S/((S \backslash NP_{acc}) \backslash NP_{nom})$ $S/((S \backslash NP_{acc}) \backslash NP_{nom})$ $(S \backslash NP_{acc}) \backslash NP_{nom}$
Naomi-ACC Ken-NOM, Sara-ACC Erika-NOM visit-PAST.CONCL
'Ken visited Naomi and Erika, Sara.'

As discussed in section 5.8, we can regard these variant constituent orders in Japanese as lexically specified, either via multiple verb categories or via explicitly unordered leftward verb categories such as (76), in keeping with the observation that local scrambling (as distinct from true extraction) is clause-bounded.

Unlike Hoffman's (1995) extension of CCG for scrambling, the formulation discussed in section 5.8 does not immediately allow the following pattern of coordination to be captured:

- (91) ?[Naomi-o Ken-ga,] [Erika-ga Sara-o] tazuneta
 $S/((S \backslash NP_{acc}) \backslash NP_{nom})$ $S/((S \backslash NP_{nom}) \backslash NP_{acc})$ $(S \backslash NP_{nom}) \backslash NP_{acc}$
Naomi-ACC Ken-NOM, Erika-NOM Sara-ACC visit-PAST.CONCL
'Ken visited Naomi and Erika, Sara.'

The status of such coordinations in Japanese is not entirely clear. They tend to be judged less acceptable than the earlier alternatives, and seem to be pragmatically marked. (Interestingly, the similarly problematic SO&OSV pattern seems to be even less acceptable than the OS&SOV pattern in (91).) But the consensus seems to be that they are grammatical.

We will return briefly to this problem in section 5.9.3.

5.9.2 Gapping and VSO Word Order

It is obvious on arguments from symmetry that the VSO subject and object must be assumed to raise over VSO verbs like the Irish Gaelic transitive verb *chonaic* ‘saw’, (92), and to compose with each other and with adjuncts in an order-preserving way to yield a single function, as in (93).

- (92) $chonaic := (S/NP) /_o NP : \lambda x \lambda y. see'yx$

- (93) a. Chonaic Eoghan Siobhán.
saw Eoghan Siobhán
'Eoghan saw Siobhán.'
- b. Chonaic Eoghan Siobhán
 $(S/NP) /_o NP$ $(S/NP) \backslash ((S/NP)/NP)$ $S \backslash ((S/NP)/NP)$
 $S \backslash ((S/NP)/NP)$ $S \backslash ((S/NP)/NP)$ $S \backslash ((S/NP)/NP)$
 S

The diamond modality on the rigid category (92) prevents Irish from exhibiting scrambling in the mirror-image of Japanese (89), and prevents reordering analogous to Heavy NP shift (39) (which is otherwise allowed in Irish as in English), just as that modality does for English ditransitives (cf. (45b)). Nevertheless the category allows the formation of relative clauses in the usual way, which may as in Dutch be ambiguous (McCloskey 1978):³¹

- (94) an file a mhol na mic léinn
the poet that praised the students
 NP/N N $(N \backslash N)/(S/NP)$ $(S/NP) /_o NP$ NP
 $(N \backslash N) /_o NP$ $(N \backslash N) \backslash_o ((N \backslash N) /_o NP)$
 $N \backslash N$
 N
 NP
'the poet that praised the students'

- (95) an file a mhol na mic léinn
the poet that the students praised
 NP/N N $(N \backslash N)/(S/NP)$ $(S/NP) /_o NP$ NP
 S/NP
 $N \backslash N$
 N
 NP
'the poet that the students praised'

Since such clusters necessarily bear leftward functor categories, when they coordinate they give the appearance of rightward gapping, in line with Ross' generalization.

- (96) Chonaic [Eoghan Siobhán] agus [Eoghnai Ciarán].
 $(S/NP) /_o NP$ $S \backslash ((S/NP)/NP)$ $(X \backslash_* X) /_* X$ $S \backslash ((S/NP)/NP)$
saw Eoghan Siobhán and Eoghnai Ciarán
'Eoghan saw Siobhán, and Eoghnai, Ciarán.'

Again the three principles correctly exclude the “backward gapping” construction that Ross (1970) held to be generally disallowed in strictly verb-initial languages:

(97) *[Eoghan Siobhán] agus chonaic [Eoghnáí Ciarán].
 $S \backslash ((S/ NP)/ NP) (X \backslash_* X)/_X (S/ NP))/_o NP S \backslash ((S/ NP)/ NP)$
 Eoghan Siobhán and saw Eoghnáí Ciarán

As in the case of SOV languages, allegedly VSO languages exist that allow leftward gapping as well as rightward. Zapotec (Rosenbaum 1977) is a standard example. However, like Dutch, Zapotec has mixed word order. It is therefore again consistent with a version of Ross' generalization formulated as in *SP* in terms of surface orders and available lexical verb categories rather than “deep” or “underlying” word-order.

5.9.3 Gapping and SVO Word Order

The fact that gapping in English and other verb-medial languages is rightward (as in the VSO pattern) needs further apparatus since the NPs in the ungapped conjunct are separated by the SVO verb.

(98) Marcel proved completeness, and Gilbert, soundness.

Comparison with example (67) and the fact that the English lexicon contains a limited class of VSO verbs already suggests that an explanation is not far away.

SP, following Steedman 1990, proposes a class of “decomposition” rules which map the left conjunct onto a virtual VSO verb and a virtual SO argument cluster with the same category as the adjacent gapped right coordinate. The latter can coordinate with the virtual cluster and apply to the verb (whose interpretation has to be obtained contextually, possibly by processes akin to VP anaphora). This augmentation of CCG makes a number of correct predictions about the possibility of “Stripping” constructions in English. Karamanis (2001) and Bozsahin (2000) have used this apparatus to capture the kind of mixed-order gapping illustrated for Japanese in (91) in Greek and Japanese, respectively. White and Baldrige (2003) compile the effect of decomposition into a coordinating category to permit gaps to be parsed and realized in a computational implementation of CCG.³² However, the decomposition analysis itself remains controversial and is passed over here.

5.10 Intonation Structure and Parentheticals

We also have seen that, in order to capture coordination with rules adhering to the constituent condition, CCG generalizes surface constituency to give substrings like *Marcel proved* and even *a policeman a flower* the full status of constituents.

But if they are constituents of coordinate constructions, they are predicted to be possible constituents of ordinary non-coordinate sentences as well. The characteristics of intonation structure and the related phenomenon of parentheticalization show that this prediction is correct. (Because of restrictions of space, this part of the account is sketched in less detail than that on coordination. For details the reader is referred to Prevost (1995) and Steedman (1991, 2000a).)

5.10.1 English Intonation and Information Structure

Consider the following minimal pair of dialogs, in which intonational tunes are indicated both informally via parentheses and small capitals as before, and in the standard notation of Pierrehumbert (1980) and Pierrehumbert and Beckman (1988), in which prosodic phrases are specified solely in terms of two kinds of elements, the pitch accent(s) and the boundary:

(99) Q: I know who proved soundness. But who proved COMPLETENESS?

A: (MARCEL) (proved COMPLETENESS).
 H* L L+H* LH%

(100) Q: I know which result Marcel PREDICTED. But which result did Marcel PROVE?

A: (Marcel PROVED) (COMPLETENESS).
 L+H*LH% H* LL%

In (99A), there is a prosodic phrase on MARCEL including the sharply rising pitch accent that Pierrehumbert calls H*, immediately followed by an L boundary, perceived as a rapid fall to low pitch. There is another prosodic phrase having the somewhat later-rising and (more importantly) lower-rising pitch accent called L+H* on COMPLETENESS, preceded by null tone (and therefore interpolated low pitch) on the word *proved* and immediately followed by an utterance-final rising boundary, written LH%.

In (100A) above, the order of the two tunes is reversed: this time, the tune with pitch accent L+H* and boundary LH% occurs on the word *PROVED* in one prosodic phrase, *Marcel PROVED*, and the other tune with pitch accent H* and boundary LL% is carried by a second prosodic phrase *COMPLETENESS*.

The intuition that these tunes strongly convey systematic distinctions in discourse meaning is inescapable. For example, exchanging the answer tunes between the two contexts in (99) and (100) yields complete incoherence. Prevost and Steedman (1994) claim that the tunes L+H* LH% and H* L (or H* LL%) are respectively associated with the “theme” and “rheme” of the sentence, where these terms are used in the sense of Mathesius (1929), Firbas (1964, 1966), and Bolinger (1989), and correspond roughly to a generalization of the more familiar terms “topic” and “comment”, which however are generally restricted by definition to traditional constituents.

Informally the theme can be thought of as corresponding to the content of a contextually available *wh*-question, which may be explicit, as in (99) and (100), or implicit in other discourse content. The position on the pitch accent, if any, in the theme, distinguishes words corresponding to “focused” elements of the content which distinguish this theme from other contextually available alternatives. The rheme can then be thought of as providing the answer to the implicit *wh*-question, with the pitch accent again marking focused words which distinguish this answer semantically from other potential answers. The system comprising the oppositions of theme/rheme and focus/background is known as information structure. Steedman (2000a) provides a more formal definition in terms of the “alternative semantics” of Rooth (1985, 1992), and the related “structured meanings” of Cresswell (1973, 1985), von Stechow (1991), and others.³³

Since alternatives like the following are equally valid surface derivations in CCG, it will be obvious that CCG provides a framework for bringing intonation structure and its interpretation – information structure – into the same syntactic system as everything else:

- (101)
$$\frac{\frac{\text{Marcel}}{NP : \text{marcel}'}}{\frac{S/(S \setminus NP) : \lambda f.f \text{ marcel}'}{S : \text{prove}' \text{ completeness}' \text{ marcel}'}} \xrightarrow{T} \frac{\frac{\text{proved}}{(S \setminus NP)/NP : \text{prove}'}}{\frac{S \setminus NP : \lambda y.\text{prove}' \text{ completeness}' y}{S : \text{prove}' \text{ completeness}' \text{ marcel}'}} \frac{\frac{\text{completeness}}{S \setminus (S \setminus NP) : \lambda p.p \text{ completeness}'}}{S \setminus (S \setminus NP) : \lambda p.p \text{ completeness}'}} \xrightarrow{<}$$
- (102)
$$\frac{\frac{\text{Marcel}}{NP : \text{marcel}'}}{\frac{S/(S \setminus NP) : \lambda f.f \text{ marcel}'}{S/NP : \lambda x.\text{prove}' x \text{ marcel}'}} \xrightarrow{T} \frac{\frac{\text{proved}}{(S \setminus NP)/NP : \text{prove}'}}{\frac{S \setminus (S \setminus NP) : \lambda p.p \text{ completeness}'}} \frac{\frac{\text{completeness}}{S \setminus (S \setminus NP) : \lambda p.p \text{ completeness}'}}{S \setminus (S \setminus NP) : \lambda p.p \text{ completeness}'}} \xrightarrow{B}$$

Crucially, these alternative derivations are guaranteed to yield the same predicate argument relations, as exemplified by the logical form that results from (101) and (102), $\text{prove}' \text{ completeness}' \text{ marcel}'$. It follows that c-command-dependent phenomena such as binding and control can be captured at the level of logical form (Steedman 1991). However, the derivations build this logical form via different routes that construct lambda terms corresponding semantically to the theme and rheme. In particular the derivation (101) corresponds to the information structure associated with the intonation contour in (99), while derivation (102) corresponds to that in (100).

This observation can be captured by making pitch accents mark both arguments and results of CCG lexical categories with theme/rheme markers θ/ρ , as in the following category for a verb bearing an L+H* accent:

- (103) $\text{proved} := (S_\theta \setminus NP_\theta) / NP_\theta : \lambda x \lambda y. * \text{prove}' xy$

The predicate is marked as focused or contrasted by the * marker in the logical form. θ/ρ marking is projected onto the arguments and result of constituents by combinatory derivation. The boundary tones like LH% have the effect of completing information structural constituents, and transferring theme/rheme marking to θ'/ρ' marking to constituent interpretations at logical form, as indicated in outline in Figure 5.12. We will pass over further details of exactly how this works, referring the reader to Prevost (1995) and to Steedman (2000a). The latter paper generalizes this approach to the full range of tunes identified by Pierrehumbert, including those with multiple pitch accents and multiple or

disjoint themes and rhemes.

5.10.2 Parentheticals

While we will not discuss parentheticals in any detail here, it seems likely that they too should be defined in terms of information structural units. In most cases, the parenthetical intrusion itself appears at the boundary between theme and rheme, hence it is subject to the same constraints as intonational phrase boundaries:

- (104) a. Marcel proved, so he claimed, the crucial property of completeness.
b. *Three mathematicians, expostulated Harry, in ten derive a lemma.

5.11 Implications for Performance: the Strict Competence Hypothesis

Psycholinguistic evidence concerning performance can never prove or disprove a competence theory. However, all things being equal, standard assumptions about the desirability of parsimony in scientific theories would lead us to expect the human sentence processing mechanism to require the minimum of specialized apparatus over and above the competence grammar that supports (and is presumably determined by) the semantics. To that extent, it is relevant to consider the implications for processing of the present theory of grammar, despite the fact that like any competence theory, it stands or falls on linguistic grounds.

The minimum apparatus besides competence grammar that is required for processing consists of the characteristic automaton for the relevant class of grammars (including its possibly limited working memories), a minimal algorithm for applying the rules, and some memory for building interpretable structure. Any extra apparatus such as rule-orderings or “strategies,” covering grammars, and the like, is otiose. To the extent that such extra stipulations are cross-linguistically universal, they complicate the problem of explaining language evolution. To the extent that they are language-specific, they do the same disservice to the problem of explaining child language acquisition.

The most restrictive hypothesis of all is that the processor involves no resources at all beyond the minimum specified above. Such processors are incapable of building intermediate structures other than those corresponding to the constituents defined by the competence grammar, and for this reason the

hypothesis that the human processor has this character is called the “strict competence” hypothesis (SCH).

One very simple processor adhering to this principle is based on the left-to-right version of the Cocke-Kasami-Young (CKY) parser (see Harrison 1978), a bottom-up parser which fills the cells of an $n \times n$ table or half-matrix t representing all spans between positions (i, j) in a string of n words.

The associativity of functional composition in interaction with type-raising potentially creates exponentially many multiple derivations for any given constituent for a given span with a given sense or interpretation (the so-called “spurious ambiguity” problem). It follows that such a parser will have exponential computational costs *unless* we include a check that a newly-derived category spanning (i, j) including its normalized logical form is not already on the list in $t(i, j)$ before appending it, a suggestion first made by Karttunen (1989).

Such parsers have been shown by Komagata (1999) to be of roughly cubic observed time complexity in the length of the sentence for reasonable-sized hand-built grammars, and Hockenmaier, Bierner and Baldridge (2002) demonstrate their practicality as a basis for large-scale grammars induced from corpora.

Cubic time costs are still prohibitive for really large volume parsing and unrealistic as a model of the human parser, which appears to be linear time or better. For large volume parsing of text corpora, statistical optimization techniques integrating probabilistic head-dependencies with competence-based grammar of the kind proposed by Collins (1999) and Charniak, Goldwater and Johnson (1998) are particularly well-adapted to CCG parsing. Clark (2002), Clark, Hockenmaier and Steedman (2002), Hockenmaier and Steedman (2002a), Hockenmaier and Steedman (2002b), Hockenmaier (2003a,b), and Gildea and Hockenmaier (2003) show that statistically optimized CCG parsers give rates of dependency recovery that are as good overall as state-of-the-art treebank parsers, and do better on recovering long-range dependencies.

For modeling human parsing, there is every indication that something even more restrictive is needed. Bever’s (1970) observation that naive subjects typically fail to find any grammatical analysis at all for “garden path” sentences like (105a) shows that the human processor is “incomplete”:

- (105) a. The doctor sent for the patient arrived.
 b. The flowers sent for the patient arrived.

The fact that (as Bever also noticed) the same subjects typically judge the isomorphic sentence (105b) grammatical suggests that the human sentence processor prunes the search space on the basis either of the relative likelihood of noun phrases like *the doctor* or *the flower* being dependent in relations like subject or object on verbs like *send for*, or the relative likelihood of the various logical forms corresponding to entire prefixes such as *the flowers/doctor sent for* in a particular context. In the case of (105a) this will cause the only analysis compatible with the rest of the sentence to be rejected, causing the garden path. Crain and Steedman (1985) and Altmann and Steedman (1988) showed that manipulating the context for related sentences in such a way as to pragmatically support the modifier reading eliminates the classic garden path effect. This fact suggests that the latter alternative is at work, rather than (or perhaps as well as) the former purely statistical mechanisms.

These authors proposed a modification of the basic parser according to which each word was processed in a left-to-right traversal of the sentence and rival analyses developed in parallel could more or less immediately be pruned under a “weak interaction” with an incrementally assembled semantic interpretation according to the following principle:³⁴

(106) *The Principle of A Priori Plausibility*

The analysis whose interpretation is most plausible in the current context will be preferred.

Since plausibility could depend on reference to context among other factors, this was expanded as including the following special case.

(107) *The Principle of Parsimony*

The analysis whose interpretation carries fewest unsatisfied but accommodatable presuppositions will be preferred.

(Since restrictive reduced relative NP interpretations for phrases like *the doctor sent for the patient* carry presuppositions such as that there is more than one doctor in the discourse context, in the null or unspecified context typical of early experiments they will be dispreferred, leading to garden paths like the one in (105a).)

There is a misleading tendency in the literature to refer to the above the-

ory as the “referential” theory of disambiguation, and to claim that evidence of other incremental semantic effects on parsing contradicts this theory (Sedivy and Spivey-Knowlton 1993; Spivey-Knowlton and Sedivy 1995; Tanenhaus and Trueswell 1995). However, the incremental semantic interaction that Crain and Steedman (1985) and Altmann and Steedman (1988) propose under these principles is clearly with all aspects of meaning that contribute to semantic plausibility — referential, sense-semantic, and knowledge-based. It should also be noted that probability as reflected in statistical models used in computational linguistics represents a mixture of semantic and knowledge-based relations bearing on plausibility, of very much the kind that we call for.

Since this theory was proposed, there has been considerable experimental evidence produced that this “weak semantic interactive” processing is indeed the way that human sentence processors actually work, using experimental measures of the time course of processing load as diverse as self-paced reading time (Altmann 1985), eye-movements (Trueswell, Tanenhaus and Kello 1993; Trueswell, Tanenhaus and Garnsey 1994; Spivey-Knowlton, Trueswell and Tanenhaus 1993; Sedivy and Spivey-Knowlton 1993; Spivey-Knowlton and Sedivy 1995; Tanenhaus and Trueswell 1995) and ERP recording from the surface of the brain itself (van Berkum, Brown and Hagoort 1999).

However, in terms of traditional grammar, both probabilistic and weak semantically interactive interpretations of the plausibility effect on garden paths present a problem for SCH. If the parser is to take account of the incompatibility of *flowers* and the subject slot of the tensed verb reading of *sent for*, this information must become available before *the patient* is integrated. (Otherwise the processor would be able to “see” the incompatible verb *arrived*, and avoid the garden path in (105a).)

This means that the parser must implicitly or explicitly have access to the interpretation or partial structure corresponding to the prefix *The flowers sent for* But this substrings is not a legal constituent according to standard grammars. So SCH appears to be breached: the parser has built or thought about building a relation that the grammar does not recognize via constituency.

This may not seem to be a very serious problem in English, where the subject and verb are immediately adjacent and could be related by other means, albeit in violation of SCH. However in verb final languages characterized by constructions like the Dutch example (2), in which arbitrarily many arguments

can be separated from their verbs by long-distance dependencies similar effects would be much more problematic, in effect requiring the parser to have sophisticated predictive mechanisms and to build explicit or implicit partial structures corresponding to non-constituent fragments.

Dutch, German and Japanese speakers greet with hilarity the suggestion that their languages prohibit any analysis until the verb group (in the Dutch bare infinitival construction, the *entire* verb group) has been processed. Moreover, there are a number of experimental results which are claimed to show effects of early syntactic commitment. In particular, Gorrell (1995b,a); Inoue and Fodor (1995); Mazuko and Itoh (1995); Sturt and Crocker (1996); Kamide and Mitchell (1999) show that Japanese speakers are committed to one analysis of an ambiguity arising from the possibility of null anaphora in complex argument sequences, as revealed by garden path effects when a verb incompatible with the preferred analysis is encountered. Konieczny et al. (1997) claim to show a similar early commitment for German. All authors relate these effects to availability of *case* information in these languages, a phenomenon whose resemblance to type-raising has already been noted.

For example, Kamide and Mitchell (1999) note that in ambiguous sentences like the following, in which the dative object of either *kasita* ‘lent’ or *miseta* ‘showed’ could be a null anaphor, there is a bias (once the intrinsic preferences of the verbs are controlled for) in favor of the former reading—that is, for the explicit NP meaning ‘the student’ to be attached to the higher verb (HA) rather than the lower (LA):

- (108) Kyooju-ga gakusee-ni toshokansisho-ga kasita mezasarii komonjo-o miseta
 [Professor-NOM {student-DAT} librarian-NOM] lent_↑ unusual manuscript-ACC showed_↑
 ‘the professor showed [HA: the student] the unusual manuscript which the librarian had lent [LA: the student].’

Introducing disambiguating transitives at ↑ shows early commitment to [] over { } causing increased processing load as measured by self-paced reading time at the disambiguation point *tore* in (110):

- (109) Kyooju-ga gakusee-ni toshokansisho-ga yabutta mezasarii komonjo-o miseta
 [Professor-NOM student-DAT] librarian-NOM tore_↑ unusual manuscript-ACC showed_↑
 ‘the professor showed the student the unusual manuscript which the librarian had torn.’

- (110) Kyooju-ga gakusee-ni toshokansisho-ga kasita mezurasii komonjo-o yabutta
 Professor-NOM {student-DAT librarian-NOM} lent unusual manuscript-ACC tore;
 ‘the professor tore the unusual manuscript which the librarian had lent the student.’

Kamide and Mitchell claim that these results show that the processor is committed to $[]$ over $\{\}$ *before* encountering the subordinate clause verb, since the verbs alone are on average unbiased. Under standard grammatical assumptions this would imply that the parser is in violation of SCH (since neither [Professor-NOM student-DAT] nor {student-DAT librarian-NOM} are constituents under these assumptions). However, both are available as constituents complete with interpretations under the various alternative derivations that CCG allows (cf. (85)), and can therefore be compared without violating SCH. (Such a comparison is likely to favor the former if only on probabilistic grounds, since the order DAT-NOM is non-canonical in Japanese.)

It would be pleasant to be able to take this result at face value, and claim it in support of a model of human sentence processing using CCG as the competence grammar under the strict competence hypothesis. We have already seen that both [Professor-NOM student-DAT] and {student-DAT librarian-NOM} are available as argument cluster constituents complete with logical forms under alternative CCG derivations illustrated in (85) and (86). CCG therefore provides everything that is needed for the parser to compare the analyses either in probabilistic or semantic/pragmatic terms under the principle of a priori plausibility and its variants.

However, such a claim would be premature at this point. Kamide and Mitchell’s interpretation of their result is based on a somewhat restrictive view of the possible parsing architectures. Under more liberal assumptions, a processor using a standard right branching grammar could produce these effects without making any commitment until the subordinate verb is actually encountered. If the processor can at that point without violating SCH temporarily build interpretable structure corresponding to both null dative anaphor and lexicalised dative argument and compare the corresponding propositions—“librarian lent *pro* trace” vs. “librarian lent student trace”—choosing the most likely, then we cannot predict the preference from the off-line comparison of “librarian lent *pro* manuscript” vs. “librarian lent student manuscript,” which Kamide and Mitchell show to be balanced overall.

The reason we cannot predict it is that (109), unlike the offline compari-

son of simplex clauses in isolation, provides a really salient antecedent for the pronominal null anaphor *pro* at the level of logical form, namely the professor.³⁵ Moreover, (109) does not offer *any* explicit possible antecedent for the matrix clause null anaphor. The preference for the interpretation of the relative clause under these conditions might well swing towards “librarian lent *pro* trace”—or in CCG terms the logical form $\lambda x.lend'(pro'professor')x librarian'$.

The only obvious way to decide whether there really is the even earlier commitment hypothesised by Kamide and Mitchell is to do an experiment in the style of Crain and Altmann, controlling the contextual appropriateness of the [Professor-NOM student-DAT] and {student-DAT librarian-NOM} cluster constituents via contextual narratives or the Japanese equivalent of questions such as *What manuscript did the professor show the student?* vs. *What manuscript did the professor tear?*.

If such an early commitment by the processor is observed, it will be of interest that CCG allows such processors to adhere rigorously to the strict competence hypothesis, even for verb-final languages.

5.12 Conclusion

Because of its very literal-minded adherence to the constituent condition on rules, and the consequent introduction of composition and type-raising, which project directionality specifications and other information from the lexicon subject to principles of slash inheritance and consistency, Combinatory Categorical Grammar abandons traditional notions of surface constituency in favor of “flexible” surface structure, in which most contiguous substrings of a grammatical sentence are potential constituents, complete with a compositional semantic interpretation, for the purposes of the application of grammatical rules.

The benefits of this move are the following.

1. Coordination, Parentheticalization, and Intonation Structure can all be handled with the same apparatus that is required for “*wh*-movement” constructions such as relativization, using purely type-driven syntactic rules that strictly adhere to the Constituent Condition on Rules.
2. The rules of syntax are universal and invariant; lexical control over their applicability allows the more powerful rules to be used in the contexts where they are needed while keeping them from causing overgeneration

elsewhere.

3. Efficient processing including weakly semantically interactive incremental parsing remains possible and is compatible with rigorous observation of the strict competence hypothesis, even for head-final languages.
4. Everything that depends on relations of “c-command” (e.g. binding and control, quantifier scope) must be dealt with at the level of logical form (cf. Bach 1980; Lasnik and Saito 1984), with a consequent transfer of responsibility for the grammar of bounded constructions to the lexicon.
5. The modules of Phonological Form, S-Structure, and Intonational Structure are unified into a single surface derivational module.

In respect of the last point, in eliminating all intervening modules between phonetic or phonological form, CCG is in broad accord with the principles of the Minimalist Program, advocated by Chomsky (1993, 1995) in recent years, and in particular the version proposed by Epstein et al. (1998) (cf. Kitahara 1995), in which it is proposed to equate Chomsky’s operations Merge and Move as a single operation. To the extent that both relativization (and other so-called movements) and in-situ argument reduction are effected in CCG by the same type-driven operation of functional application, it can be seen as formalizing this idea. However it should be noted that in other respects the frameworks are quite different. In particular, the meaning of the term “derivation” as used by Epstein et al. is quite different from the sense of that term used here and in *SP*.

The architecture of the theory that ensues can be summarized in the version of the standard T- or Y- diagram shown in figure 5.12, in which the markers θ and p denote features marking elementary categories which permit the combinatory rules to project theme- and rheme- marking from accented words onto phrasal constituents. (The markers θ' and p' in the logical forms denote the corresponding semantic functions at the level of information structure, while asterisks in logical forms mark focused or contrastive semantic elements deriving from words bearing pitch-accents.)

Although information structure further reduces to a fairly tradition (albeit unordered) predicate-argument structure by β -normalization (essentially, functional application of the λ terms), such a reduction is unnecessary both for the

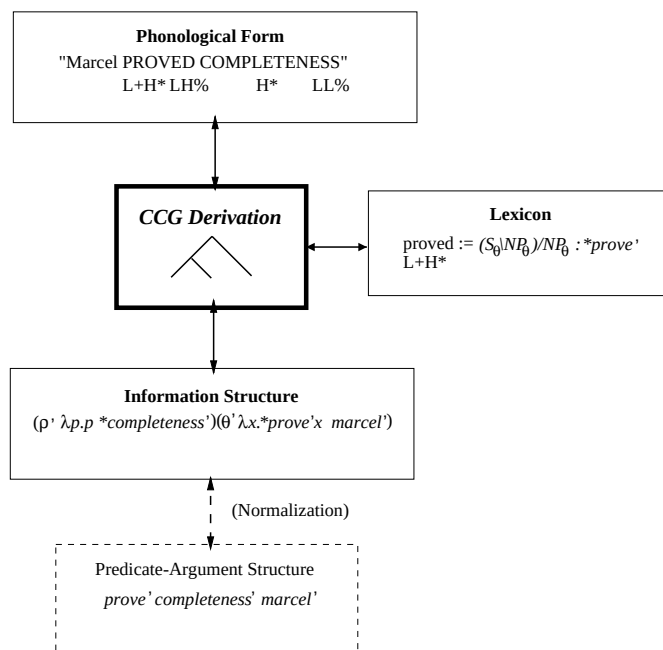


Figure 5.3
Architecture of CCG

theory of grammar and for the purposes of comprehension and production. It follows that information structure is the only level of logical form that is required and the only true representational level in CCG: since all rules are purely type-driven and monotonic, and no rule depends upon derivational history, surface structures need not be built as explicit representations. CCG is therefore a strictly monostratal theory that fully integrates coordination and intonation structure into surface grammar.

Chapter 5

1. Besides those discussed in this volume, others in active use include Tree-Adjoining Grammar (TAG, Joshi 1988 and Government-Binding theory itself (GB, a.k.a. Principles and Parameters, the Minimalist Program, etc., Chomsky 1981, 1995).
2. The indicated dependencies are those between semantically related arguments and predicates, rather than surface dependencies between verbs and NP arguments that would be attributed on a VP analysis of the construction. However, in either case the Dutch dependencies cross.
3. This conjecture has been challenged by Rambow (1994) and subsequently defended by Joshi, Rambow and Becker (2000).
4. Notable exceptions are Goodall (1983, 1987) and Moltmann (1992).
5. We believe that this was actually proposed by Mark Liberman at one stage.
6. A third class of combinatory rules related to *Substitution*, Curry and Feys' **S** combinator, are ignored here.
7. There is an alternative "result on top" notation due to Lambek (1958), according to which the latter category is written $\beta \backslash \alpha$.
8. The use of a hierarchy such as this as a formal device is optional, and instead could be replaced by multiple declarations of the combinatory rules.
9. The fact that restrictions are not required under this formulation of CCG answers a common and long-standing criticism of the theory from researchers in the TLG community. However, there is an obvious duality between restricting rules as to the categories that they may apply to, and restricting the categories themselves by distinguishing different modal varieties of slash — see Baldridge and Kruijff (2003) for an embedding of the modal formulation of CCG within a version of CCG which permits rule restrictions. Furthermore, while it is possible to define a TLG system that acts on the modalities and categories described here (see Baldridge 2002), we do not here assume that the modally typed slashes are true implicational operators as they are in TLG.
10. This accords with the fact in TLG that, under the residuation laws, all modalities have access to the base logic (in which the elimination rules correspond to CCG's application rules). Note, however, that it would be entirely possible to devise modal settings in multi-modal CCG in which some modalities can be used with the composition rules but not with application.
11. See Oehrle, this volume, for a deductive explanation within the Lambek framework of the relationship between categories and phrase structure labels.
12. It is possible to bind arguments in semantic representations using mechanisms other than those of the λ -calculus. For example, Steedman (1990), Zeevat (1988) and Hoffman (1995) employ unification for this purpose. The use of the λ -calculus as the representation framework is also optional since interpretations can instead be encoded with other representation languages such as Indexed Languages (Zeevat 1988), Hybrid Logic Dependency Semantics (Kruijff 2001) or Minimal Recursion Semantics

(Copestake, Lascarides and Flickinger 2001). See Baldridge and Kruijff (2002) for an approach which integrates CCG with Hybrid Logic Dependency Semantics, and Villavicencio (2002) for one which uses Minimal Recursion Semantics within the context of Unification-Based Generalized Categorical Grammar.

13. The semantics of this category, or rather category schema, is somewhat complex, and is omitted here.

14. Combinatory rules like functional composition resemble a highly restricted (because they are type-driven rather than structure-dependent) class of "generalized" or "double-based" transformations of the kind proposed in Chomsky 1957.

15. The analysis begs some syntactic and semantic questions about the coordination. See SSI for a more complete account.

16. To the extent that both type-raising and case associate entities with roles in actions, they are both akin to the psychologists' notion of the "affordance" of a percept, as that word is used by Gibson (1966) and his followers—see Steedman 2002.

17. For two nodes to be identical, one must be a pointer to the other. Mere equality of content is not identity.

18. This category can be thought of as suggestive of a more involved strategy using the unary modalities of TLG. For example, we could assume, similar to Morrill (1989) and Hepple (1990), that all categories from the lexicon are encapsulated in unary modalities. For example, a transitive verb from the lexicon would appear as $\square_{lex}^+ \triangleright_{lex} ((S \backslash NP_{agr}) / NP)$, and the category for *itself* would then be defined as $\square_{lex}^+ \triangleright_{lex} ((S \backslash NP_{3sn}) \backslash (\square_{lex}^+ \triangleright_{lex} ((S \backslash NP_{3sn}) / NP)))$ instead of (37). For a derivation without the reflexive, the unary modalities on the transitive category can be dropped (via TLG's residuation laws for unary modalities, which in CCG would be enacted using the unary rule $\square_i^+ \triangleright_i X \Rightarrow X$) to allow the category to be used in the usual manner. Using unary modalities in this manner would ensure that the category of *thinks that Mary likes* would be $(S \backslash NP_{agr}) / NP$ (lacking lexical unary modalities) and thus not be an acceptable argument for the category given above for *itself*, thereby blocking ungrammatical sentences such as **The dog_i thinks that Mary likes itself_i*.

19. For reasons discussed by Oehrle (1975), we should not assume that the predicate *give'* is identical to that of the verb in *I gave the flowers to Marcel*. In fact, the binding facts force the assumption that the underlying predicate in *Marcel showed me/*myself to *me/myself* reverses the command relations between object and showee.

20. The possibility of both extraction out of, and HNPS over, infinitival complements means that they must have both unraised and raised categories.

21. In fact, a search on the internet turns up positive examples of such sentences, such as the following utterance by a native speaker of (Australian) English: *I was writing it because I was wanted to write, but I didn't know what I wanted to write*.

22. Again, similar to what is noted in footnote 21, a search on internet comes up with plenty of positive examples, such as *Morgan was promised to receive \$52 million for the project in the next fiscal year*.

23. Bare passives are assumed to be derived by a similar rule which omits the *by-PP*

and makes the subject at If the equivalent of *someone*.

24. The feature $+ANT$ can only be bound by the relative pronoun category, and excludes right node raising as in **I eat and will live to be a hundred those apricots they grow in the Caucasus*. The precised mechanism is discussed in *SP* and need not detain us here. Baldridge 2002, 109–113, shows how the same “antecedent government” feature can more elegantly be brought under the control of a slightly richer set of slash modalities, and thes would allow such conjunctions to be simplified as versions of $(VP \backslash_{\times} VP) /_{\wedge} VP$ and $(VP /_{\circ} VP) \backslash_{*} VP$. These categories allow extraction to take place via the usual composition mechanism. They have the advantage over those in (66) of also supporting a variety of ordinary non extracted “subordinating” VP coordinations, with the same volitional causal semantics, via simple application, as in *Eat apricots and live to be a hundred!*, and *Go to the store and buy apricots!*.

25. In more recent work, Dowty has disowned CCG in favour of TLG, because of “intrinsic” use of logical form to account for binding phenomena that it entails, as discussed above. See *SSI* for further discussion.

26. This appears to offer an advantage over non-type-raising accounts using the product operator $\bullet$ of Lambek (Pickering and Barry 1993; Dowty 1996).

27. The α is a variable for a set of categories.

28. In this derivation, we suppress the $\{\}$ brackets around singleton sets to improve legibility.

29. It would indeed be possible to give adjectives a permutative slash, and this is indeed a necessary degree of freedom: possessive noun phrases in Turkish can be discontinuous (Hoffman 1995), allowing orders akin to (79c) in addition to (79a).

30. This fact precludes any attempt to account for (89) in terms of forward crossed composition, as in the following derivation:

$$(i) \quad \frac{\frac{\text{Naomi-o} \quad \text{Ken-ga} \quad \text{tazuneta}}{S / (S \backslash NP_{acc}) \xrightarrow{>T} T} \quad \frac{S / (S \backslash NP_{nom}) \quad (S \backslash NP_{nom}) \backslash NP_{acc}}{S \backslash NP_{acc} \xrightarrow{>B_x} B_x}}{S \xrightarrow{>} S}$$

31. See the discussion in Baldridge (2002) regarding extraction of the indirect and direct object arguments of English ditransitives for an alternative which uses a more complex set of slash modalities to allow extraction without scrambling.

32. The implementation is an open-source Java-based system and can be downloaded from <http://openccg.sourceforge.net>.

33. The much-abused term “focus” is used in CCG strictly in the “narrow” or phonological sense of the term, to refer to the effects of contrast or emphasis on a word that ensues from the presence of a pitch-accent.

34. This form of incrementality is weaker than those proposed by Hausser (1986) and Phillips (1996, 2001), since it is limited by “islands” such as right-adjuncts, into which composition cannot take place. Hausser’s and Phillips notion of incrementality is by contrast strictly word-by-word. (Phillips’ method of incremental structure-building is in addition nonmonotonic.)

35. Another is the student, but the accessibility hierarchy and extrapolation from English suggests that the matrix subject will be the default antecedent for the null anaphor.

References

- Ades, Anthony, and Mark Steedman. 1982. "On the Order of Words." *Linguistics and Philosophy*, 4, 517–558.
- Aho, Alfred. 1968. "Indexed Grammars—An Extension of Context-Free Grammars." *Communications of the Association for Computing Machinery*, 15, 647–671.
- Aho, Alfred. 1969. "Nested Stack Automata." *Communications of the Association for Computing Machinery*, 16, 383–406.
- Ajdukiewicz, Kazimierz. 1935. "Die syntaktische Konnexität." In Storrs McCall, ed., *Polish Logic 1920-1939*, 207–231. Oxford: Oxford University Press. Translated from *Studia Philosophica*, 1, 1-27.
- Altmann, Gerry. 1985. *Reference and the Resolution of Local Syntactic Ambiguity*. Ph.D. thesis, University of Edinburgh.
- Altmann, Gerry, and Mark Steedman. 1988. "Interaction with Context During Human Sentence Processing." *Cognition*, 30, 191–238.
- Asudeh, Ash, and Richard Crouch. 2002. "Coordination and parallelism in Glue Semantics: Integrating discourse cohesion and the Element Constraint." In Miriam Butt and Tracy Holloway King, eds., *Proceedings of the LFG02 Conference*. Stanford CA: CSLI Publications.
- Bach, Emmon. 1979. "Control in Montague Grammar." *Linguistic Inquiry*, 10, 513–531.
- Bach, Emmon. 1980. "In Defense of Passive." *Linguistics and Philosophy*, 3, 297–341.
- Bach, Emmon. 1983. "Generalized Categorical Grammars and the English Auxiliary." In Frank Heny and Barry Richards, eds., *Linguistic Categories: Auxiliaries and Related Puzzles, II*, 101–120. Dordrecht: Reidel.
- Baldridge, Jason. 2002. *Lexically Specified Derivational Control in Combinatory Categorical Grammar*. Ph.D. thesis, University of Edinburgh.
- Baldridge, Jason, and Geert-Jan Kruijff. 2002. "Coupling CCG and Hybrid Logic Dependency Semantics." In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, 319–326. Philadelphia, Pennsylvania.

- Baldridge, Jason, and Geert-Jan Kruijff. 2003. "Multi-Modal Combinatory Categorical Grammar." In *Proceedings of 11th Annual Meeting of the European Association for Computational Linguistics*. Budapest, Hungary.
- Bar-Hillel, Yehoshua. 1953. "A Quasi-Arithmetical Notation for Syntactic Description." *Language*, 29, 47–58.
- Bar-Hillel, Yehoshua, Chaim Gaifman, and Eliyahu Shamir. 1964. "On Categorical and Phrase Structure Grammars." In Yehoshua Bar-Hillel, ed., *Language and Information*, 99–115. Reading, MA: Addison-Wesley.
- van Berkum, Jos, Colin Brown, and Peter Hagoort. 1999. "Early referential context effects in sentence processing: Evidence from event-related brain potentials." *Journal of Memory and Language*, 41, 147–182.
- Bever, Thomas. 1970. "The Cognitive Basis for Linguistic Structures." In John Hayes, ed., *Cognition and the Development of Language*, 279–362. New York: Wiley.
- Bolinger, Dwight. 1989. *Intonation and Its Uses*. Stanford, CA: Stanford University Press.
- Bozsahin, Cem. 2000. "Ordered Set-CCG." Ms. Middle East Technical University, Ankara.
- Brame, Michael. 1978. *Base Generated Syntax*. Seattle, WA: Noit Amrofer.
- Bresnan, Joan. 1972. *Theory of Complementation in English Syntax*. Ph.D. thesis, MIT.
- Bresnan, Joan, ed. 1982. *The Mental Representation of Grammatical Relations*. Cambridge, MA: MIT Press.
- Charniak, Eugene, Sharon Goldwater, and Mark Johnson. 1998. "Edge-Based Best-First Chart Parsing." In *Proceedings of the 6th Workshop on Very Large Corpora, Montreal, August*, 127–133.
- Chomsky, Noam. 1957. *Syntactic Structures*. The Hague: Mouton.
- Chomsky, Noam. 1975. *The Logical Structure of Linguistic Theory*. Chicago: University of Chicago Press.
- Chomsky, Noam. 1981. *Lectures on Government and Binding*. Dordrecht: Foris.
- Chomsky, Noam. 1986. *Knowledge of Language*. New York: Praeger.
- Chomsky, Noam. 1993. "A Minimalist Program for Linguistic Theory." In Kenneth Hale and Samuel Jay Keyser, eds., *The View from Building 20*, 1–52. Cambridge, MA: MIT Press.
- Chomsky, Noam. 1995. *The Minimalist Program*. Cambridge, MA: MIT Press.
- Clark, Stephen. 1997. *Binding and Control in Categorical Grammar*. Master's thesis, University of Manchester.
- Clark, Stephen. 2002. "A Supertagger for Combinatory Categorical Grammar." In *Proceedings of the TAG+ Workshop*, 19–24. Venice.

- Clark, Stephen, Julia Hockenmaier, and Mark Steedman. 2002. "Building Deep Dependency Structures with a Wide-Coverage CCG Parser." In *Proceedings of the 40th Meeting of the ACL*, 327–334. Philadelphia, PA.
- Collins, Michael. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.
- Copestake, Ann, Alex Lascarides, and Dan Flickinger. 2001. "An Algebra for Semantic Construction in Constraint-based Grammars." In *Proceedings of the 39th Annual Meeting of the Association of Computational Linguistics*, 132–139. Toulouse, France.
- Crain, Stephen, and Mark Steedman. 1985. "On Not Being Led up the Garden Path: the Use of Context by the Psychological Parser." In Lauri Karttunen David Dowty and Arnold Zwicky, eds., *Natural Language Parsing: Psychological, Computational and Theoretical Perspectives*, 320–358. Cambridge: Cambridge University Press.
- Cresswell, M.J. 1973. *Logics and Languages*. London: Methuen.
- Cresswell, M.J. 1985. *Structured Meanings*. Cambridge, MA: MIT Press.
- Curry, Haskell B., and Robert Feys. 1958. *Combinatory Logic: Vol. I*. Amsterdam: North Holland.
- Dowty, David. 1979. "Dative Movement and Thomason's Extensions of Montague Grammar." In Steven Davis and Marianne Mithun, eds., *Linguistics, Philosophy, and Montague Grammar*, 153–222. Austin, TX: University of Texas Press.
- Dowty, David. 1988. "Type-Raising, Functional Composition, and Nonconstituent Coordination." In Richard T. Oehrle, Emmon Bach, and Deirdre Wheeler, eds., *Categorical Grammars and Natural Language Structures*, 153–198. Dordrecht: Reidel.
- Dowty, David. 1996. "Nonconstituent Coordination, Wrapping, and Multimodal Categorical Grammars: Syntactic Form as Logical Form." In Kees Doets, ed., *Proceedings of the Tenth International Congress of Logic, Methodology, and Philosophy of Science, 1995*. Amsterdam: North Holland.
- Emonds, Joseph. 1979. "Appositive Relatives Have No Properties." *Linguistic Inquiry*, 10, 211–243.
- Epstein, Samuel D., Erich Groat, Ruriko Kawashima, and Hisatsugu Kitahara. 1998. *A Derivational Approach to Syntactic Relations*. Oxford: Oxford University Press.
- Firbas, Jan. 1964. "On Defining the Theme in Functional Sentence Analysis." *Travaux Linguistiques de Prague*, 1, 267–280.
- Firbas, Jan. 1966. "Non-thematic Subjects in Contemporary English." *Travaux Linguistiques de Prague*, 2, 229–236.
- Floyd, Robert. 1962. "On the Nonexistence of a Phrase-Structure Grammar for ALGOL 60." *Communications of the Association for Computing Machinery*, 5, 483–484.
- Foster, John. 1990. *A Theory of Word-Order in Categorical Grammar, with Special Reference to Spanish*. Ph.D. thesis, University of York.

- Gazdar, Gerald. 1981. "Unbounded Dependencies and Coordinate Structure." *Linguistic Inquiry*, 12, 155–184.
- Gazdar, Gerald. 1988. "Applicability of Indexed Grammars to Natural Languages." In Uwe Reyle and Christian Rohrer, eds., *Natural Language Parsing and Linguistic Theories*, 69–94. Dordrecht: Reidel.
- Gazdar, Gerald, Ewan Klein, Geoffrey K. Pullum, and Ivan Sag. 1985. *Generalized Phrase Structure Grammar*. Oxford: Blackwell.
- Gibson, James. 1966. *The Senses Considered as Perceptual Systems*. Boston, MA: Houghton-Mifflin Co.
- Gildea, Dan, and Julia Hockenmaier. 2003. "Identifying Semantic Roles Using Combinatory Categorical Grammar." In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*. Sapporo, Japan. To appear.
- Goldsmith, John. 1985. "A Principled Exception to the Coordinate Structure Constraint." In *Proceedings of the 21st Regional Meeting of the Chicago Linguistic Society*. Chicago: CLS, University of Chicago.
- Goodall, Grant. 1983. "A Three-dimensional Analysis of Coordination." In *Proceedings of the 19th meeting of the Chicago Linguistic Society*. Chicago: University of Chicago, Chicago Linguistic Society.
- Goodall, Grant. 1987. *Parallel Structures in Syntax*. Cambridge: Cambridge University Press.
- Gorrell, Paul. 1995a. "Japanese Trees and the Garden Path." In Reiko Mazuka and Noriko Nagai, eds., *Japanese Sentence Processing*, 331–350. Hillsdale, NJ: Erlbaum.
- Gorrell, Paul. 1995b. *Syntax and Parsing*. Cambridge: Cambridge University Press.
- Halliday, Michael. 1967. *Intonation and Grammar in British English*. The Hague: Mouton.
- Harrison, Michael. 1978. *Introduction to Formal Language Theory*. Reading MA: Addison-Wesley.
- Hausser, Roland. 1986. *NEWCAT*. Berlin: Springer-Verlag.
- Hepple, Mark. 1990. *The Grammar and Processing of Order and Dependency: A Categorical Approach*. Ph.D. thesis, University of Edinburgh.
- Hockenmaier, Julia. 2003a. *Data and models for statistical parsing with CCG*. Ph.D. thesis, School of Informatics, University of Edinburgh.
- Hockenmaier, Julia. 2003b. "Parsing with generative models of predicate-argument structure." In *Proceedings of the 41st Meeting of the ACL, Sapporo*. San Francisco: Morgan-Kaufmann. To appear.
- Hockenmaier, Julia, Gann Bierner, and Jason Baldridge. 2002. "Extending the Coverage of a CCG System." *Journal of Logic and Computation*. (forthcoming).

- Hockenmaier, Julia, and Mark Steedman. 2002a. "Acquiring Compact Lexicalized Grammars from a Cleaner Treebank." In *Proceedings of the Third International Conference on Language Resources and Evaluation*. Las Palmas, Spain.
- Hockenmaier, Julia, and Mark Steedman. 2002b. "Generative Models for Statistical Parsing with Combinatory Categorical Grammar." In *Proceedings of the 40th Meeting of the ACL*, 335–342. Philadelphia, PA.
- Hoffman, Beryl. 1995. *Computational Analysis of the Syntax and Interpretation of "Free" Word-Order in Turkish*. Ph.D. thesis, University of Pennsylvania. IRCS Report 95-17.
- Huybregts, Riny. 1976. "Overlapping Dependencies in Dutch." *Utrecht Working Papers in Linguistics*, 1, 24–65.
- Huybregts, Riny. 1984. "The Weak Inadequacy of Context-free Phrase-structure Grammars." In Ger de Haan, Mieke Trommelen, and Wim Zonneveld, eds., *Van Periferie naar Kern*. Dordrecht: Foris.
- Inoue, A., and Janet Dean Fodor. 1995. "Information-paced parsing of Japanese." In R. Masuka and N. Nagai, eds., *Japanese Sentence Processing*. Hillsdale, NJ: Erlbaum.
- Jackendoff, Ray. 1972. *Semantic Interpretation in Generative Grammar*. Cambridge, MA: MIT Press.
- Jacobson, Pauline. 1990. "Raising as Function Composition." *Linguistics and Philosophy*, 13, 423–476.
- Jacobson, Pauline. 1992. "Flexible Categorical Grammars: Questions and Prospects." In Robert Levine, ed., *Formal Grammar*, 129–167. Oxford: Oxford University Press.
- Joshi, Aravind. 1988. "Tree Adjoining Grammars." In David Dowty, Lauri Karttunen, and Arnold Zwicky, eds., *Natural Language Parsing*, 206–250. Cambridge: Cambridge University Press.
- Joshi, Aravind, Owen Rambow, and Tilman Becker. 2000. "Complexity of Scrambling: A New Twist to the Competence-Performance Distinction." In Anne Abeillé and Owen Rambow, eds., *Tree Adjoining Grammars: Formalisms, Linguistic Analysis and Processing*, 167–182. Stanford, CA: CSLI Publications.
- Joshi, Aravind, K. Vijay-Shanker, and David Weir. 1991. "The Convergence of Mildly Context-Sensitive Formalisms." In Peter Sells, Stuart Shieber, and Tom Wasow, eds., *Processing of Linguistic Structure*, 31–81. Cambridge, MA: MIT Press.
- Kamide, Yuki, and Don Mitchell. 1999. "Incremental Pre-head Attachment in Japanese Parsing." *Language and Cognitive Processes*, 11, 631–662.
- Karamanis, Nikiforos. 2001. "A Categorical Grammar for Greek." In *Proceedings of the 15th International Symposium on Theoretical and Applied Linguistics*. Aristotle University of Thessaloniki.
- Karttunen, Lauri. 1989. "Radical Lexicalism." In Mark Baltin and Anthony Kroch, eds., *Alternative Conceptions of Phrase Structure*. Chicago: University of Chicago Press.

- Kehler, Andrew. 2002. *Coherence, Reference, and the Theory of Grammar*. Stanford CA: CSLI Publications.
- Kitahara, Hisatsugu. 1995. "Target α : Deducing Strict Cyclicity from Principles of Derivational Economy." *Linguistic Inquiry*, 26, 47–77.
- Komagata, Nobo. 1999. *Information Structure in Texts: A Computational Analysis of Contextual Appropriateness in English and Japanese*. Ph.D. thesis, University of Pennsylvania.
- Konieczny, Lars, Barbara Hemforth, Christoph Scheepers, and Gerhard Strube. 1997. "The Role of Lexical Heads in Parsing: Evidence from German." *Language and Cognitive Processes*, 12, 307–348.
- Kruijff, Geert-Jan. 2001. *Categorical-Modal Architecture of Informativity: Dependency Grammar Logic & Information Structure*. Ph.D. thesis, Charles University, Prague.
- Lakoff, George. 1986. "Frame Semantic Control of the Coordinate Structure Constraint." In Anne Farley, Peter Farley, and Karl-Eric McCullough, eds., *Papers from the Parasession on Pragmatics and Grammatical Theory at the 22nd Regional Meeting of the Chicago Linguistics Society*, 152–167. Chicago: CLS.
- Lambek, Joachim. 1958. "The Mathematics of Sentence Structure." *American Mathematical Monthly*, 65, 154–170.
- Lasnik, Howard, and Mamoru Saito. 1984. "On the Nature of Proper Government." *Linguistic Inquiry*, 15, 235–289.
- Mathesius, Vilem. 1929. "Functional Linguistics." In Josef Vachek, ed., *Praguiana: Some Basic and Less Well-known Aspects of the Prague Linguistics School*, 121–142. Amsterdam: John Benjamins. Publ. 1983.
- Mazuko, Reika, and Kenji Itoh. 1995. "Japanese Trees and the Garden Path." In Reiko Mazuka and Noriko Nagai, eds., *Can Japanese Speakers be Led Down the Garden Path*, 295–329. Hillsdale, NJ: Erlbaum.
- McCloskey, James. 1978. *A Fragment of a Grammar of Modern Irish*. Ph.D. thesis, University of Texas at Austin. Publ. as Texas Linguistic Forum, 12.
- Moltmann, Friederike. 1992. *Coordination and Comparatives*. Ph.D. thesis, MIT.
- Montague, Richard. 1974. *Formal Philosophy: Papers of Richard Montague*. New Haven, CT: Yale University Press. Richmond H. Thomason, ed.
- Moortgat, Michael. 1988. *Categorical Investigations*. Ph.D. thesis, Universiteit van Amsterdam. Published by Foris, Dordrecht, 1989.
- Morrill, Glyn. 1989. "Intensionality, boundedness, and modal logic." Tech. Rep. EUCCS/RP-32, Centre for Cognitive Science, University of Edinburgh.
- Morrill, Glyn. 1994. *Type-Logical Grammar*. Dordrecht: Kluwer.
- Oehrle, Richard. 1975. *The Grammatical Status of the English Dative Alternation*. Ph.D. thesis, MIT.

- Oehrle, Richard T. 2000. "Multi-modal Type Logical Grammar." In Robert Borsley and Kersti Borjars, eds., *Non-Transformational Syntax: A Guide To Current Models*, in preparation. Oxford: Blackwell.
- Partee, Barbara, ed. 1976. *Montague Grammar*. New York: Academic Press.
- Pentus, Mati. 1993. "Lambek Grammars are Context-free." In *Proceedings of the IEEE Symposium on Logic in Computer Science, Montreal*, 429–433.
- Peters, Stanley, and Robert Ritchie. 1973. "On the Generative Power of Transformational Grammars." *Information Science*, 6, 49–83.
- Phillips, Colin. 1996. *Order and Structure*. Ph.D. thesis, MIT.
- Phillips, Colin. 2001. "Linear Order and Constituency." *Linguistic Inquiry*. To appear.
- Pickering, Martin, and Guy Barry. 1993. "Dependency Categorical Grammar and Coordination." *Linguistics*, 31, 855–902.
- Pierrehumbert, Janet. 1980. *The Phonology and Phonetics of English Intonation*. Ph.D. thesis, MIT. Distributed by Indiana University Linguistics Club, Bloomington.
- Pierrehumbert, Janet, and Mary Beckman. 1988. *Japanese Tone Structure*. Cambridge, MA: MIT Press.
- Pollard, Carl. 1984. *Generalized Phrase Structure Grammars, Head Grammars, and Natural Languages*. Ph.D. thesis, Stanford University.
- Pollard, Carl, and Ivan Sag. 1991. "An Integrated Theory of Control." *Language*, 67, 63–113.
- Pollard, Carl, and Ivan Sag. 1992. "Anaphors in English and the Scope of Binding Theory." *Linguistic Inquiry*, 23, 261–303.
- Pollard, Carl, and Ivan Sag. 1994. *Head Driven Phrase Structure Grammar*. Stanford, CA: CSLI Publications.
- Postal, Paul. 1998. *Three Investigations of Extraction*. Cambridge, MA: MIT Press.
- Prevost, Scott. 1995. *A Semantics of Contrast and Information Structure for Specifying Intonation in Spoken Language Generation*. Ph.D. thesis, University of Pennsylvania.
- Prevost, Scott, and Mark Steedman. 1994. "Specifying Intonation from Context for Speech Synthesis." *Speech Communication*, 15, 139–153.
- Rambow, Owen. 1994. *Formal and Computational Aspects of Natural Language Syntax*. Ph.D. thesis, University of Pennsylvania.
- Reinhart, Tanya, and Eric Reuland. 1991. "Anaphors and Logophors: An Argument Structure Perspective." In Jan Koster and Eric Reuland, eds., *Long Distance Anaphora*, 283–322. Cambridge: Cambridge University Press.
- Reinhart, Tanya, and Eric Reuland. 1993. "Reflexivity." *Linguistic Inquiry*, 24, 657–720.
- Rooth, Mats. 1985. *Association with Focus*. Ph.D. thesis, University of Massachusetts, Amherst.

- Rooth, Mats. 1992. "A Theory of Focus Interpretation." *Natural Language Semantics*, 1, 75–116.
- Rosenbaum, Harvey. 1977. "Zapotec Gapping as Counterevidence to Some Universal Proposals." *Linguistic Inquiry*, 8, 379–395.
- Ross, John Robert. 1967. *Constraints on Variables in Syntax*. Ph.D. thesis, MIT. Published as *Infinite Syntax!*, Ablex, Norton, NJ, 1986.
- Ross, John Robert. 1970. "Gapping and the Order of Constituents." In Manfred Bierwisch and Karl Heidolph, eds., *Progress in Linguistics*, 249–259. The Hague: Mouton.
- Sedivy, Julie, and Michael Spivey-Knowlton. 1993. "The Effect of NP Definiteness on Parsing Attachment Ambiguity." In *NELS 23*. Amherst: University of Massachusetts, GLSA.
- Selkirk, Elisabeth. 1984. *Phonology and Syntax*. Cambridge, MA: MIT Press.
- Selkirk, Elisabeth. 1986. "Derived Domains in Sentence Phonology." *Phonology*, 3, 371–405.
- Selkirk, Elisabeth. 1990. "On the Nature of Prosodic Constituency." *Papers in Laboratory Phonology*, 1, 179–200.
- Shieber, Stuart. 1985. "Evidence against the Context-Freeness of Natural Language." *Linguistics and Philosophy*, 8, 333–343.
- Shieber, Stuart. 1986. *An Introduction to Unification-Based Approaches to Grammar*. Stanford, CA: CSLI Publications.
- Spivey-Knowlton, Michael, and Julie Sedivy. 1995. "Resolving Attachment Ambiguities with Multiple Constraints." *Cognition*, 55, 227–267.
- Spivey-Knowlton, Michael, John Trueswell, and Michael Tanenhaus. 1993. "Context Effects in Syntactic Ambiguity Resolution: Parsing Reduced Relative Clauses." *Canadian Journal of Psychology*, 47, 276–309.
- von Stechow, Arnim. 1991. "Focusing and Backgrounding Operators." In Werner Abraham, ed., *Discourse Particles: Descriptive and Theoretical Investigations on the Logical, Syntactic and Pragmatic Properties of Discourse Particles in German*, 37–84. Amsterdam: John Benjamins.
- Steedman, Mark. 1990. "Gapping as Constituent Coordination." *Linguistics and Philosophy*, 13, 207–263.
- Steedman, Mark. 1991. "Structure and Intonation." *Language*, 67, 262–296.
- Steedman, Mark. 1996. *Surface Structure and Interpretation*. Cambridge, MA: MIT Press.
- Steedman, Mark. 2000a. "Information Structure and the Syntax-Phonology Interface." *Linguistic Inquiry*, 34, 649–689.
- Steedman, Mark. 2000b. *The Syntactic Process*. Cambridge, MA: MIT Press.
- Steedman, Mark. 2002. "Plans, Affordances, and Combinatory Grammar." *Linguistics and Philosophy*, 25, 723–753.

- Sturt, Patrick, and Matthew Crocker. 1996. "Monotonic Syntactic Processing: a Cross-Linguistic Study." *Language and Cognitive Processes*, 11, 449–494.
- Taglicht, Josef. 1998. "Constraints on Intonational Phrasing in English." *Journal of Linguistics*, 34, 181–211.
- Tanenhaus, Michael, and John Trueswell. 1995. "Sentence comprehension." In Jim Miller and Peter Eimas, eds., *Speech, Language and Communication*, vol. 11 of *Handbook of Perception and Cognition*, 217–262. New York: Academic Press.
- Trechsel, Frank. 2000. "A CCG Account of Tzotzil Pied Piping." *Natural Language and Linguistic Theory*, 18, 611–663.
- Trueswell, John, Michael Tanenhaus, and Susan Garnsey. 1994. "Semantic Influences on Parsing: Use of Thematic Role Information in Syntactic Ambiguity Resolution." *Journal of Memory and Language*, 33, 285–318.
- Trueswell, John, Michael Tanenhaus, and Christopher Kello. 1993. "Verb-Specific Constraints in Sentence Processing: Separating Effects of Lexical Preference from Garden-Paths." *Journal of Experimental Psychology: Learning, Memory and Cognition*, 19, 528–553.
- Vijay-Shanker, K., and David Weir. 1990. "Polynomial Time Parsing of Combinatory Categorical Grammars." In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*, Pittsburgh, 1–8. San Francisco, CA: Morgan Kaufmann.
- Vijay-Shanker, K., and David Weir. 1993. "Parsing Some Constrained Grammar Formalisms." *Computational Linguistics*, 19, 591–636.
- Vijay-Shanker, K., and David Weir. 1994. "The Equivalence of Four Extensions of Context-Free Grammar." *Mathematical Systems Theory*, 27, 511–546.
- Villavicencio, Aline. 2002. *The Acquisition of a Unification-Based Generalised Categorical Grammar*. Ph.D. thesis, University of Cambridge.
- White, Michael, and Jason Baldrige. 2003. "Adapting Chart Realization to CCG." In *Proceedings of the 9th European Workshop on Natural Language Generation, held in conjunction with the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL), Budapest, April 2003*. San Francisco, CA: Morgan Kaufmann. To appear.
- Wood, Mary McGee. 1993. *Categorical Grammar*. London: Routledge.
- Zeevat, Henk. 1988. "Combining Categorical Grammar and Unification." In Uwe Reyle and Christian Rohrer, eds., *Natural Language Parsing and Linguistic Theories*, 202–229. Dordrecht: Reidel.

Starting with Complex Primitives Pays Off: Complicate Locally, Simplify Globally

Aravind K. Joshi

*Department of Computer and Information Science, and Institute for Research in
Cognitive Science, University of Pennsylvania, Philadelphia, PA 19104, U.S.A.*

Abstract

In setting up a formal system to specify a grammar formalism, the conventional (mathematical) wisdom is to start with primitives (basic primitive structures) as simple as possible, and then introduce various operations for constructing more complex structures. An alternate approach is to start with complex (more complicated) primitives, which directly capture some crucial linguistic properties and then introduce some general operations for composing these complex structures. These two approaches provide different domains of locality, i.e., domains over which various types of linguistic dependencies can be specified. The latter approach, characterized as *complicate locally, simplify globally* (CLSG), pushes non-local dependencies to become local, i.e., they arise in the basic primitive structures to start with.

The CLSG approach has led to some new insights into syntactic description, semantic composition, language generation, statistical processing, and psycholinguistic phenomena, all these with possible relevance to the cognitive architecture of language. In this paper, we will describe these results in an introductory manner making use of the framework of Lexicalized Tree-Adjoining Grammar (LTAG), a key example of the CLSG approach, thereby describing the interplay between formal analysis on the one hand and linguistic and processing issues on the other hand.

Key words: domain of locality, non-local dependencies, lexicalized tree adjoining grammar, nested and crossed dependencies, word order variation and scope ambiguities, grammar architecture and psycholinguistic properties

1 Introduction

The conventional (mathematical) wisdom in specifying a grammar formalism is to start with basic primitive structures as simple as possible and then in-

Email address: joshi@linc.cis.upenn.edu (Aravind K. Joshi).

roduce various operations for constructing more complex structures. These operations can be simple or complex and the number of operations (although finite) need not be limited. New operations (simple or complex) can be introduced in order to describe more complex structures.

An alternate approach is to start with complex (more complicated) primitives, which capture directly some crucial linguistic properties and then introduce some general operations for composing these complex structures (primitive or derived). What is the nature of these complex primitives? In the conventional approach the primitive structures (or rules) are kept as simple as possible. This has the consequence that information (e.g., syntactic and semantic) about a lexical item (word) is distributed over more than one primitive structure. Therefore, the information associated with a lexical item is not captured locally, i.e., within the domain of a primitive structure. We will illustrate this in Section 1.1 in terms of the well-known context-free grammar (CFG) framework.

In contrast, in the alternate approach described in this paper, we allow the primitive structures to be as complex as necessary to capture all the relevant information about a lexical item in the local domain of a primitive structure. The kinds of information that need to be localized are as follows: (a) a lexical item taken as a predicate has zero or more arguments, (b) the arguments need to satisfy certain syntactic and semantic constraints, which are determined by the lexical item, and (c) the arguments will occupy different positions relative to the position of the lexical item. Hence, in the alternate approach, all the pieces of information associated with a lexical item have to be represented in the local domains of the primitive structures of the formal system. In Section 1.3, we will illustrate these ideas in terms of the lexicalized tree-adjoining grammar (LTAG), a class of grammars that illustrates this alternate approach by adopting it in its extreme form.

In this alternate approach, although the primitives are complex and there may be more than one primitive structure associated with a lexical item, the number of primitives is finite. Further, the combining operations are kept to the minimum and they are language independent (i.e., *universal*). It will be shown later that the alternate approach, which starts with complex primitives and can be characterized as *complicate locally, simplify globally* (CLSG), pushes non-local dependencies to become local, i.e., they are located in the basic primitive structures to start with. In the conventional approach they are spread out over one or more primitive structures. The architecture resulting from the CLSG approach has important implications for linguistics, computational linguistics, and psycholinguistics, including generation and acquisition. Some of these implications will be illustrated later by examples in Section 4.

We will discuss the various issues arising out of the CLSG approach in the

context of the formal system known as the Lexicalized Tree-Adjoining Grammar (LTAG)¹. LTAG and some of its extensions have been investigated both formally and computationally for over twenty five years². LTAG represents a class of grammars that illustrates the CLSG approach by adopting it in an extreme form and thus serves to bring out the payoffs of the CLSG strategy. These payoffs are at least in the following key dimensions³.

- An extension of the notion of lexical ambiguity making it possible to unify some ambiguities, conventionally treated as structural ambiguities, with the usual lexical ambiguities, making them both lexical and leaving the other ambiguities as the only structural ambiguities (pure attachment ambiguities).
- Representation of scope ambiguities as attachment ambiguities in the syntax itself, resulting in syntactic derivations that are underspecified by just the right amount.
- Unification of even putative non-lexical information (which, in the conventional approaches, is scattered across different primitive structures, structure transformation rules, and logical forms) in terms of the finite set of primitive lexicalized structures of LTAG and the two combining operations of substitution and adjoining.

This unification is the result of a notion of lexicalization. In the LTAG conception of lexicalization, each primitive structure of LTAG is associated with a lexical item and the structure not only encapsulates all and only the arguments of the lexical anchor but crucially, also provides a structural slot for each one of the arguments. It is this notion of lexicalization that

¹ There is another dimension in which formal systems can be characterized. One could start with an unconstrained formal system (Turing machine equivalent, for example) and then add linguistic constraints, which become in a sense, all stipulative. Alternatively, one could start with a formal system that is constrained and just adequate for describing language. The formal constraints then become universal, in a sense. All other linguistic constraints become stipulative and language specific. Now it turns out that the CLSG approach leads to constrained formal systems, LTAG in particular. This convergence is of interest in its own right. However, in this paper, we will not discuss constrained systems in their own right. Our focus will be on the CLSG approach and its implications for the architecture of the grammars and their processors.

² See Joshi, Levy, & Takahashi (1975); Joshi (1985); Kroch & Joshi (1985); Vijay-Shanker (1987); Weir (1988); Kroch (1989); Kroch & Santorini (1991); Schabes (1992); Rambow (1994); Resnik (1992); Chiang (2000); Sarkar (2002); Abeillé (2002); Prolo (2003).

³ Using the CLSG approach it is also possible to study directly many aspects of strong generative capacity, in terms of the set of structural descriptions (i.e., sentence structures), provided by a formal system and not just the strings (i.e., sentences. The former are more relevant to the linguistic descriptions (Miller, 1999; Joshi, 2003). This aspect is not pursued in this paper.

takes LTAG beyond other lexicalized grammars⁴ and provides a notion of locality that results in these payoffs.

By using simple examples to illustrate the formal aspects that are needed to discuss various issues, I will try to communicate the interplay between formal analysis on the one hand and linguistic and processing issues, on the other hand, something that, I hope, is consistent with the criteria for the David E. Rumelhart Prize. The work reported here is by no means a survey of all the formal work related to language, computation, and processing; it is only a slice from my particular perspective⁵.

The plan of the paper is as follows: in the remainder of Section 1 we will introduce the notions of domain of locality and lexicalization in the context of the well-known context-free grammars (CFG) and then show how lexicalized tree-adjoining grammars (LTAG) arise in the process of lexicalizing CFGs and extending the domain of locality. In Section 2, we will also show how the architecture of the building blocks of LTAG directly predicts many complex dependency patterns and then summarize some important properties of LTAG. In Section 3, we will introduce two alternate perspectives for LTAG: (a) supertagging and (b) flexible composition, and discuss their implications for language description and language processing, and more importantly, their psycholinguistic relevance in, Section 4. In this Section, we will also briefly mention some other implications of the LTAG architecture from the linguistic, computational, and psycholinguistic perspectives. Some of these issues are

⁴ For example, in a categorial grammar (CG) each lexical item is also associated with an expression that encodes the arguments of that lexical item. However, in this representation, the different structural positions occupied by the different arguments are not shown in the primitive expression associated with the lexical item. These positions implicitly arise in the course of the derivation. In this sense LTAG can be viewed as a class of grammars emerging from the full use of the CLSG approach.

⁵ There are several formal systems that are clearly related to LTAG. Some examples are Combinatory Categorical Grammars (CCG)(Steedman, 1996), Stabler's version of minimalist grammars (Stabler, 1997), Lexical Functional Grammars (LFG)(Kaplan & Bresnan, 1983), Head Driven Phrase Structure Grammars (HPSG)(Pollard & Sag, 1994), for a constrained version of HPSG, see Kasper, Kiefer, Netter, & Vijay-Shanker (1995). Linear Indexed Grammars (LIG) by Gazdar, and Head Grammars (HG) by Pollard, CCG, and LTAG have all been shown to be weakly equivalent, i.e., in terms of the string sets they generate but not in terms of the structural descriptions. These relationships have been discussed extensively in Joshi, Vijay-Shanker, K., & Weir, D. (1991). CCGs are close to LTAGs, however, LTAGs are *fully lexicalized*, in the sense that each elementary tree of LTAG not only encodes the argument structure of the lexical anchor but also the structural positions the arguments would occupy in minimal clauses. In this sense, the elementary trees of LTAG can be considered as structured categories (see Section 3 for supertags), see also Joshi & Kulick (1997).

discussed by other authors in this issue of the journal. Finally, in Section 5, we will summarize the main issues concerning the CLSG approach.

1.1 Domain of locality of CFGs

In a context-free grammar (CFG) the domain of locality is the one level tree corresponding to a rule in a CFG (Fig. 1). It is easily seen that the arguments of a predicate (for example, the two arguments of *likes*) are not in the same local domain. The two arguments are distributed over the two rules (two domains of locality)– $S \rightarrow NP VP$ and $VP \rightarrow V NP$. They can be brought together by introducing a rule $S \rightarrow NP V NP$. However, then the structure provided by the VP node is lost. We should also note here that not every rule (domain) in the CFG in (Fig. 1) is lexicalized.

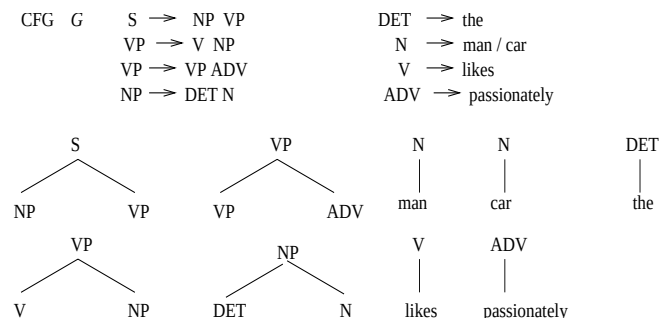


Fig. 1. Domain of locality of a context-free grammar

The five rules on the right are lexicalized, i.e., they have a lexical anchor. The rules on the left are not lexicalized. The second, the third and the fourth rule on the left are almost lexicalized, in the sense that they each have at least one preterminal category, (V in the second rule, ADV in the third rule, and DET and N in the fourth rule), i.e., by replacing V by *likes*, ADV by *passionately*, and either DET by *the* or N by *man*, these three rules will become lexicalized. However, the first rule on the left ($S \rightarrow NP VP$) cannot be lexicalized, not certainly by *man*.

Can a CFG be lexicalized, i.e., given a CFG, G , can we construct another CFG, G' , such that every rule in G' is lexicalized and $T(G)$, the set of (sentential) trees (i.e., the tree language of G) is the same as the tree language $T(G')$ of G' ? Of course, if we require that only the string languages of G and G' be the same (i.e., they are weakly equivalent) then any CFG can be lexicalized. This follows from the fact that any CFG can be put in the Greibach normal form (see (Linz, 2001)) where each rule is of the form $A \rightarrow w B_1 B_2 \dots B_n$ where w is a lexical

item and the B 's are nonterminals⁶. We call this *weak* lexicalization. The lexicalization we are interested in requires the tree languages (i.e., the set of structural descriptions) to be the same (i.e., strong equivalence). We call this *strong* lexicalization. It is easily seen, even from the example in (Fig. 1), that a non-lexicalized CFG cannot be necessarily strongly lexicalized by another CFG. Basically this follows from the fact that the domain of locality of CFG is a one level tree corresponding to a rule in the grammar (for detail see Joshi & Schabes (1997)). In Section 1.2 we will consider lexicalization of CFG by larger (extended) domain of locality.

Before proceeding further, it would be helpful to review certain definitions. The primitive structures of a formalism (also called elementary structures or elementary trees, as special cases) provide a *local* domain for specifying linguistic constraints (pieces of linguistic theory) in the sense that if the constraints are specifiable by referring to just the structures that are associated with the elementary structures then it is specifiable over the domain of these elementary structures. Therefore, we refer to the domains corresponding to the elementary structures as *domains of locality*. Formalism A is said to provide an *extended domain of locality* as compared to a formalism B if there is a linguistic constraint which is not specifiable in the local domains associated with B but which is specifiable in the local domains associated with A. The goal of the CLSG approach is to look for a formalism which provides local domains large enough so that, in principle, *all* linguistic constraints (pieces of linguistic theory) can be specified over these local domains. In the conventional approach (e.g., CFG based) the specification of a constraint is often spread out over more than one local domain and thus the specification of a constraint is intertwined with the how the local domains are composed by the grammar, in other words, specification of a constraint will require specification of recursion, resulting in an effectively unbounded domain. In contrast, in the CLSG approach, we seek a system with extended (but still finite) domains of locality capable of specifying the linguistic constraints over these extended domains. Thus, recursion does not enter into the specification of the constraints. We call this property as *factoring recursion away from the domains of locality*.

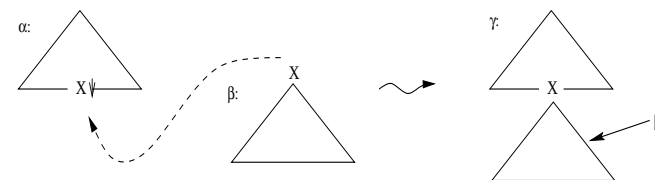


Fig. 2. Substitution

⁶ The Greibach form of the rule is related to the categories in a categorial grammar.

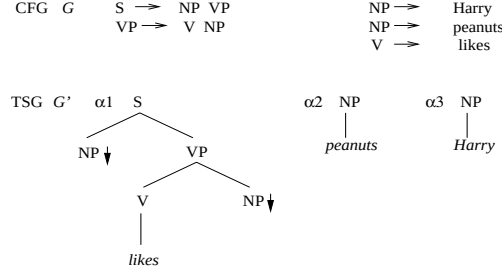


Fig. 3. Tree substitution grammar

1.2 Lexicalization of CFGs by grammars with larger domains of locality

Now we can ask the following question. Can we strongly lexicalize a CFG by a grammar with a larger domain of locality? Fig. 2 and Fig. 3 show a tree substitution grammar where the elementary objects (building blocks) are the three trees in Fig. 3 and the combining operation is the *tree substitution* operation shown in Fig. 2. The down arrows in Fig. 2 and Fig. 3 denote the substitution sites. Note that each tree in the tree substitution grammar (TSG), G' is lexicalized, i.e., it has a *lexical anchor*. It is easily seen that G' indeed strongly lexicalizes G . However, TSGs fail to strongly lexicalize CFGs in general. We show this by an example. Consider the CFG, G , in Fig. 4 and a proposed TSG, G' . It is easily seen that although G and G' are weakly equivalent they are not strongly equivalent. In G' , suppose we start with the tree α_1 then by repeated substitutions of trees in G' (a node marked with a vertical arrow denotes a substitution site) we can grow the right side of α_1 as much as we want but we cannot grow the left side. Similarly for α_2 we can grow the left side as much as we want but not the right side. However, trees in G can grow on both sides. In order for a tree to grow on both sides, the distance between the lexical anchor of a tree, a , and the root of the tree, S , must become arbitrarily large. Substitution makes a tree grow only at the leaves of the tree and cannot make it grow internally. Hence, the TSG, G' , cannot strongly lexicalize the CFG, G (Joshi & Schabes, 1997). Thus, even with the extended domain of locality of TSGs we cannot strongly lexicalize CFGs as long as substitution is the only operation for putting trees together.

We now introduce a new operation called *adjoining* as shown in Fig. 5. Adjoining involves splicing (inserting) one tree into another. More specifically, a tree β as shown in Fig. 5 is inserted (adjoined) into the tree α at the node X resulting in the tree γ . The tree β , called an *auxiliary tree*, has a special form. The root node is labeled with a nonterminal, say X and on the frontier there is also a node labeled X called the foot node (marked with *). There could be other nodes (terminal or nonterminal) on the frontier of β , the nonterminal

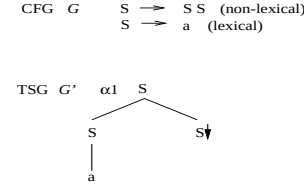


Fig. 4. A tree substitution grammar

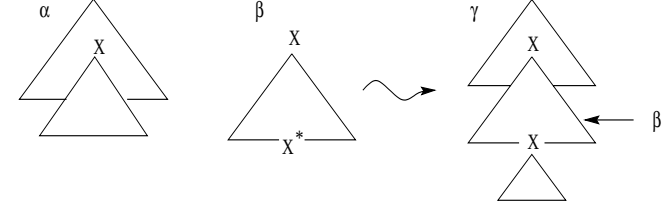


Fig. 5. Adjoining

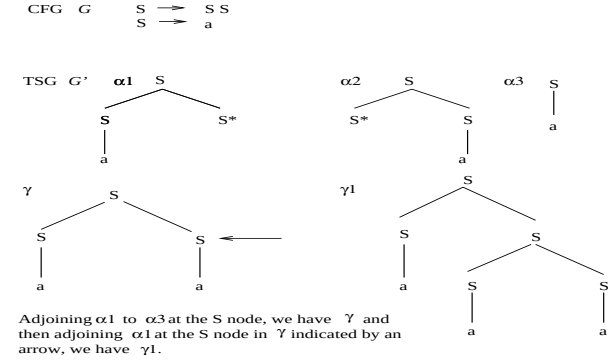


Fig. 6. Adjoining arises out of lexicalization

nodes marked as substitution sites (with a vertical arrow). Thus, if there is another occurrence of X (other than the foot node marked with *) on the frontier of β it will be marked with the vertical arrow and that will be a substitution site. Given this specification, adjoining β to α at the node X is uniquely defined. Adjoining can also be seen as a pair of substitutions as follows: The subtree at X in α is detached, β is substituted at X and the detached subtree is then substituted at the foot node of β . A tree substitution grammar when augmented with the adjoining operation is called a *tree-adjoining grammar* (lexicalized tree-adjoining grammar because each elementary tree is lexically anchored). In short, LTAG consists of a finite set of *elementary trees*, each lexicalized with at least one lexical anchor. The elementary trees are either initial or auxiliary trees. Auxiliary trees have been defined already. *Initial* trees are

those for which all nonterminal nodes on the frontier are substitution nodes. It can be shown that any CFG can be strongly lexicalized by an LTAG (Joshi & Schabes, 1997).

In Fig. 6, we show a TSG, G' , augmented by the operation of adjoining, which strongly lexicalizes the CFG, G . Note that the LTAG looks the same as the TSG considered in Fig. 4. However, now trees α_1 and α_2 are auxiliary trees (foot node marked with *) that can participate in adjoining. Since adjoining can insert a tree in the interior of another tree it is possible to grow both sides of the tree α_1 and tree α_2 , which was not possible earlier with substitution alone. In summary, we have shown that by increasing the domain of locality we have achieved the following: (1) lexicalized each elementary domain, (2) introduced an operation of adjoining, which would not be possible without the increased domain of locality (note that with one level trees as elementary domains, adjoining becomes the same as substitution since there are no interior nodes to be operated upon), and (3) achieved strong lexicalization of CFGs.

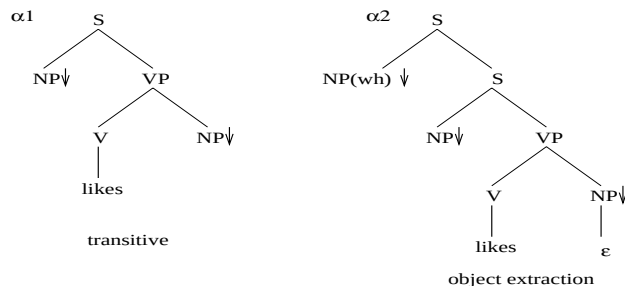


Fig. 7. LTAG: Elementary trees for *likes*

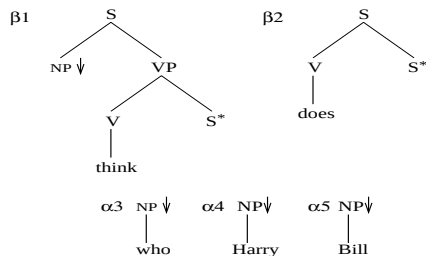


Fig. 8. LTAG: Sample elementary trees

1.3 Lexicalized tree-adjoining grammars

Rather than giving formal definitions for LTAG and derivations in LTAG, we will give a simple example to illustrate some key aspects of LTAG⁷. We show some elementary trees of a toy LTAG grammar for English. Fig. 7 shows two elementary trees for a verb such as *likes*. The tree α_1 is anchored on *likes* and encapsulates the two arguments of the verb. The tree α_2 corresponds to the object extraction construction. Since we need to encapsulate all the arguments of the verb in each elementary tree for *likes*, for the object extraction construction, for example, we need to make the elementary tree associated with *likes* large enough so that the extracted argument is in the same elementary domain. Thus, in α_2 , the node for $NP(wh)$ (the extracted argument) has to be in the tree for *likes*. Further, there is a dependency between the $NP(wh)$ node and the NP node which is the complement of *likes* (i.e., to the right of V dominating *likes*) and this dependency is local to α_2 . The tree α_2 not only shows that $NP(wh)$ is an argument of *likes* but also that it is large enough to indicate a specific structural position for that argument.

Therefore, in principle, for each ‘minimal’ construction in which *likes* can appear (for example, subject extraction, topicalization, subject relative, object relative, passive, etc.) there will be an elementary tree associated with that construction. By *minimal* we mean that all recursion has been factored away. This factoring of recursion away from the domain over which the dependencies have to be specified is a crucial aspect of LTAGs, as they are used in linguistic descriptions. This factoring allows all dependencies to be localized in the elementary domains. In this sense, there will, therefore, be no long distance dependencies as such. They will all be local and will become long distance on account of the composition operations, especially adjoining. This will become clear as soon as we describe the derivation in Fig. 9.

Fig. 8 shows some additional elementary trees – trees α_3 , α_4 , and α_5 and trees β_1 and β_2 . The β trees with foot nodes marked with * will enter a derivation by the operation of adjoining. The α trees enter a derivation by the operation of substitution⁸.

A derivation using the trees α_2 , α_3 , α_4 , α_5 , β_1 , and β_2 is shown in Fig. 9. The trees for *who* and *Harry* are substituted in the tree for *likes* at the respective

⁷ In the actual LTAG grammar each node in an elementary tree is decorated with attribute value structures (feature structures) which encode various linguistic constraints specified over the domain of an elementary tree. There is no recursion in these feature structures. We omit these details here as they are not essential for our immediate purpose.

⁸ This distinction between the two types of elementary trees is characterized in LTAG in terms of *initial trees* (the α trees) and the *auxiliary trees* (the β trees).

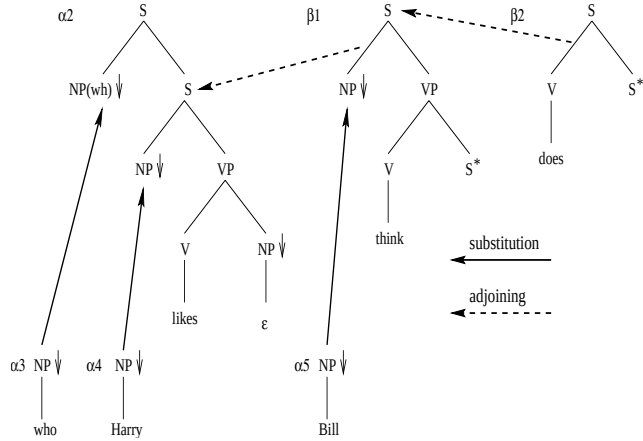


Fig. 9. LTAG derivation for *who does Bill think Harry likes*

NP nodes, at node addresses 1 and 2.1 in α_2 . The tree for *Bill* is substituted in the tree for *think* at the *NP* node at the node address 1 in β_1 . the tree for *does* is adjoined to the root node (address 0) of the tree for *think* tree (adjoining at the root node is a special case of adjoining), see Fig. 10 for this intermediate derived tree.

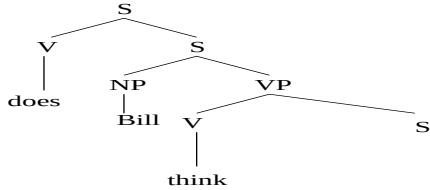


Fig. 10. Intermediate derived tree for β_2 adjoined to β_1

Finally, the derived auxiliary tree (after adjoining β_2 to β_1) is adjoined to the indicated interior *S* node of the tree α_2 at the address 2 in α_2 . This derivation results in the *derived tree* for

Who does Bill think Harry likes

as shown in Fig. 11. Note that the dependency between *who* and the complement *NP* in α_2 (local to that tree) has been stretched in the derived tree in Fig. 11. It has become *long distance*. However, it started out as a local dependency. A key property of LTAGs is that all dependencies are local, i.e., they are specified in the elementary trees. They can become long distance as a result of the composition operations. Fig. 11 is the conventional tree associated with the sentence.

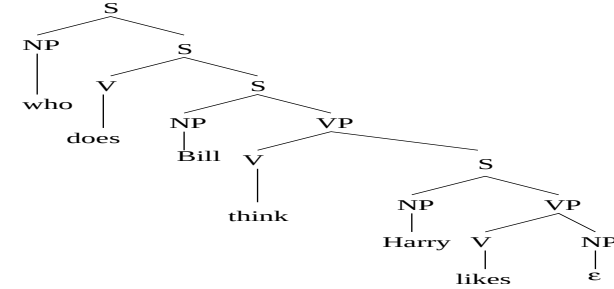


Fig. 11. LTAG derived tree for *who does Bill think Harry likes*

However, in LTAG, there is also a *derivation tree*, the tree that records the history of composition of the elementary trees associated with the lexical items in the sentence. This derivation tree is shown in Fig. 12. The nodes of the tree are labeled by the tree labels such as α_2 together with its lexical anchor *likes*⁹. The number on an edge of a derivation tree refers to the node address in a tree into which either a substitution or adjoining has been made. Thus, for example, in Fig. 12 the α_3 (*who*) tree is substituted at the node with address 1 in the tree α_2 (*likes*), the tree β_1 (*thinks*) is adjoined at the address 2 in the tree α_2 (*likes*), etc. Solid edges denote substitution and dotted edges denote adjoining.

The derivation tree is the crucial derivation structure for LTAG. It records the history of composition in terms of the elementary trees (primitive building blocks) of LTAG. The derived tree in Fig. 11 does not indicate what the component elementary trees are for the final derived tree. It should be clear that from the derivation tree we can always obtain the derived tree by performing the substitutions and adjoinings indicated in the derivation tree. So, in this sense, the derived tree is redundant.

Further, for semantic computation, the derivation tree (and not the derived tree) is the crucial object. Compositional semantics is defined on the derivation tree. The idea is that for each elementary tree there is a semantic representation associated with it and these representations are composed using the derivation tree. Since the semantic representation for each elementary tree is directly associated with the tree, there is no need to reproduce necessarily the internal hierarchy in the elementary tree in the semantic representation (Joshi & Vijay-Shanker, 1999; Kallmeyer & Joshi, 1999; Joshi, Kallmeyer, & Romero, 2003). This means that the hierarchical structure internal to each elementary tree need not be reproduced in the semantic representation. This leads to the

⁹ The derivation trees of LTAG have a close relationship to dependency trees, although there are some crucial differences. The semantic dependencies are the same, however.

so-called *flat* semantic representation. i.e., the semantic expression associated with the sentence is essentially a conjunction of semantic expressions associated with each elementary tree¹⁰. Of course, relevant machinery has to be provided for scope information (for details see Kallmeyer & Joshi (1999)). The semantics need not be compositional at the level of the elementary trees. It is, however, compositional at the level of the derivation tree, i.e., at the level at which the elementary trees are assembled. This aspect of the architecture is also helpful in dealing with some of the non-compositional aspects, as in the case of rigid and flexible idioms (see Abeillé (2002); Stone & Doran (1999)).

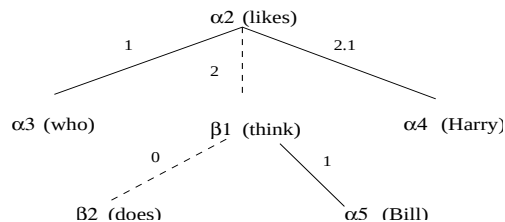


Fig. 12. LTAG derivation tree

2 Some important properties of LTAG

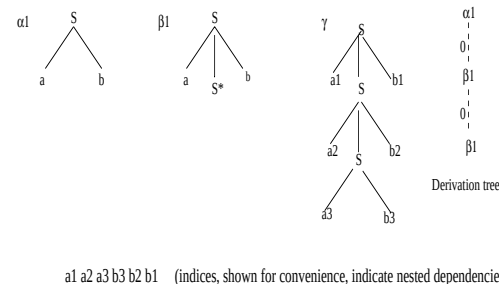
2.1 Nested and crossed dependencies

The two key properties of LTAG are (1) extended domain of locality (EDL) (for example, as compared to CFG), which allows (2) factoring recursion from the domain of dependencies (FRD), thus making all dependencies local. All other properties of LTAG (mathematical, linguistic, and even psycholinguistic) follow from EDL and FRD. Roughly speaking, this follows from the fact that since all dependent elements are encapsulated in each elementary tree, all *interesting* properties of LTAG follow directly from the architecture of the elementary trees. We will illustrate this by considering nested and crossed dependencies. Fig. 13 shows an LTAG for nested dependencies. This is, of course, a CFG represented as an LTAG. The elementary trees $\alpha1$ and $\beta1$ are both one level (depth one) trees and each tree encapsulates the two dependent elements a and b , which are at the same level (depth). The tree γ is obtained by

¹⁰ The general notion of flat semantics is related to the notion of minimal recursion semantics (MRS)(Copestake, Flickinger, Sag, & Pollard, 1999). MRS has been used in the semantic computation in the HPSG framework. In the LTAG framework the notion of elementary trees and the derivation tree which specifies the composition in terms of the elementary trees directly provides a representation for computing a *flat* semantics.

starting with $\alpha1$, then adjoining the tree $\beta1$ at the root node of $\alpha1$, and then adjoining another instance of $\beta1$ at the root node of the previously derived tree. The indices on the dependent elements show the nested dependencies in the linear string associated with γ . The rightmost tree in Fig. 13 is the *derivation tree*. The nodes are labeled by the names of the elementary trees, $\alpha1$ and $\beta1$. Going bottom up on the derivation tree, $\beta1$ adjoins to another instance of $\beta1$ at the address 0 in $\beta1$. Then, the derived $\beta1$ adjoins to $\alpha1$ at the address 0 in $\alpha1$. Once we have the derivation tree, the derived tree is redundant, because it is the result of carrying out the compositions (in our case, by adjoining) as depicted in the derivation tree.

Fig. 14, on the other hand, shows an LTAG for crossed dependencies. Here the architecture of the elementary trees is different from the trees in Fig. 13. Each elementary tree is of depth two and the two dependent elements a and b in each elementary tree are not at the same depth, b is one level below a , although both are in the same tree. The derived tree $\gamma1$ is obtained as follows. Starting with $\alpha1$ and then adjoining $\beta1$ to the interior S node (as indicated by an arrow) of $\alpha1$, we get γ . Then, adjoining another instance of $\beta1$ at the interior S node (as indicated by an arrow) of γ , we get $\gamma1$. The indices on the dependent elements show the crossed dependencies in the linear string corresponding to $\gamma1$. The rightmost tree in Fig. 14 is the *derivation tree*. This derivation tree is identical to the derivation tree in Fig. 13, except that $\beta1$ in Fig. 14 is different from $\beta1$ in Fig. 13. Further, going bottom up on the derivation tree, $\beta1$ is adjoined to another instance of $\beta1$ at the address 2 in $\beta1$ and then the derived $\beta1$ is adjoined to $\alpha1$ at the address 2 in $\alpha1$.



a1 a2 a3 b3 b2 b1 (indices, shown for convenience, indicate nested dependencies)

Fig. 13. An LTAG for nested dependencies

Note that the composition operation of adjoining is the same in both the examples. Whether we get nested dependencies or crossed dependencies is directly predicted from the architecture of the elementary trees. Nested dependencies arise due to center embedding in English as in (1) and due to complement embedding in German as in (2), (2') shows the corresponding English word order).

- (1) *The rat₁ the cat₂ chased₂ ate₁ the cheese*

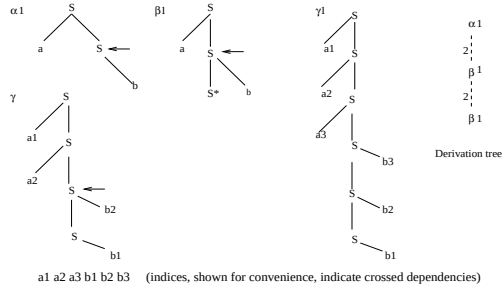


Fig. 14. An LTAG for crossed dependencies

(2) *Hans₁ Peter₂ Marie₃ schwimmen₃ lassen₂ sah₁*

(2') *Hans saw Peter make Marie swim*

Both (1) and (2) are examples of nested dependencies. However, they arise for different reasons as we have stated above. This difference can be articulated in the architecture of the respective elementary trees for (1) and (2). We will not discuss this detail here but will comment on this issue in the context of some processing issues discussed in Section 4.

Crossing dependencies arise due to complement embedding in Dutch (under certain conditions) as in (3), (3' shows the corresponding English word order).

(3) *Jan₁ Piet₂ Marie₃ zag₁ laten₂ zwemmen₃*

(3') *Jan saw Piet make Marie swim*

It is possible to obtain a wide range of complex dependencies by manipulating the architecture of the elementary trees of LTAG, which result in combining nested and crossed dependencies in complex manners. Such dependencies arise due to complex word order phenomena such as *scrambling* and *clitic movement* and also due to scope ambiguities. We will briefly describe these issues in Section 4.

2.2 Some formal properties

We will list some important properties and describe them very briefly.

1. TAGs are more powerful than CFGs, both weakly and strongly, i.e., TAGs not only generate more languages (strings) than CFGs (weak sense) but also generate more structural descriptions (trees) than CFGs (strong sense). It is

the strong power that is linguistically more relevant. Even if a language is context-free, a TAG for that language can assign structural descriptions not assignable by a CFG. Many complex word orders can be described as CFGs if we are concerned with strings only. However, if we want to make sure that for each string we have the correct structural description (i.e., correct semantics) then we may need more power, in the strong sense, than that provided by CFGs. We will pursue this issue in Section 4 in the context of a discussion of competence and performance properties.

2. TAGs carry over all formal properties of CFGs, modified in an appropriate manner. Thus, for example, the (worst case) parsing complexity of CFGs is $O(n^3)$, polynomial in n , where n is the length of the sentence, while it is $O(n^6)$ for TAGs.

3. CFGs are associated with pushdown automata (PDA) in the sense that for every CFG there is an associated PDA which recognizes precisely the same language which the corresponding CFG generates. Similarly TAGs are associated with the so-called Embedded Pushdown Automata (EPDA) (Vijay-Shanker, 1987).

EPDAs are a generalization of PDAs. In a PDA a move of the automaton is determined by (a) the input symbol, (b) the current state of the automaton, and (c) the topmost symbol on the stack. The result of the move takes the automaton to a new state, pushing a new sequence of symbols on the top of the stack or popping the topmost symbol from the stack.

In an EPDA, a move may add a specified number of stacks to the left or right of the current stack. More specifically, a move of an EPDA is determined by (a) the input symbol, (b) the current state of the automaton, and (c) the topmost symbol on the (current) stack. The result of the move takes the automaton to a new state, pushing a new sequence of symbols on the (current) stack or popping the topmost symbol from the (current) stack, adding a specified (by the move) number of stacks to the left and right of the (current) stack, and pushing specified information on the newly introduced stacks. At the end of the move, crucially, the stack pointer points to the topmost symbol of the rightmost stack. It is this last requirement that makes an EPDA behave much like a PDA. The rightmost stack after a move becomes the current stack. During the moves of the automaton, if a stack becomes empty, we assume that it is eliminated from the sequence of stacks. In Section 4, we will explore some implications of the TAG-EPDA correspondence and the EPDA architecture for certain complex dependencies for human sentence processing.

4. TAGs (more precisely, the languages of TAGs) belong to a class of languages called *mildly context-sensitive languages* (MCSL). This class was proposed by (Joshi, 1985) with a suggestion that natural languages lie in this class. This

class was characterized by the following properties: (a) worst case parsing complexity of a language in this class is polynomial, i.e., it is proportional to n^k , for some integer k and n is the length of the sentence. For CFGs $k = 3$ and for TAGs, $k = 6$, (b) grammars for languages in this class can characterize a limited set of patterns of nested and crossed dependencies and their combinations, (c) languages in this class have the constant growth property, i.e., if the sentences of language are arranged in increasing order of length then the difference between the lengths of successive sentences is bounded by a constant, and (d) context-free languages are properly included in this class.

Property (a) just says that languages in MCSL are not too hard to parse. Property (b) says that complex dependency patterns are possible but completely arbitrary patterns of dependencies are not permitted. Property (c) captures the intuition that clauses have minimal lengths and a clause can be built further by adding a bounded amount of material to it. A sentence is made up of clauses and can be lengthened by adding a clause at a time. Property (d) says that MCSL is a proper extension of CFLs. CFGs have already been extensively used in formulating linguistic and computational theories, as well as psycholinguistic processing accounts.

Over the past 20 years or so the MCSL hypothesis has generated very fruitful research in comparing different linguistic and formal proposals, and discovering some equivalences among formalisms, leading to a proper interplay of formal frameworks, substantive linguistic theories, and computational and processing paradigms.

5. Large scale wide coverage grammars have been built using LTAG, the XTAG system (LTAG grammar and lexicon for English and a parser) being the largest so far (for further details see (XTAG, 2002))¹¹. In the XTAG system, each node in each LTAG tree is decorated with two feature structures (top and bottom feature structures), in contrast to the CFG based feature structure grammars. This is necessary because adjoining can augment a tree internally, while in a CFG based grammar a tree can be augmented only at the frontier. It is possible to define adjoining and substitution (as it is done in the XTAG system) in terms of appropriate unifications of the top and bottom feature structures. Because of factoring recursion from the domain of dependencies, there is no recursion in the feature structures. Therefore, in principle, feature structures can be eliminated. However, they are crucial for linguistic descriptions. Constraints on substitution and adjoining are modeled via these feature structures (Vijay-Shanker, 1987). This method of manipulating feature structures is a direct consequence of the extended domain of locality of

¹¹ A large French grammar, FTAG is described in (Abeillé & Candito, 2002; Abeillé, 2002). Other sizable LTAG grammars exist for German, Hindi, and Korean. There may be some others that I am not aware of at this time.

LTAG.

3 Two alternate perspectives on LTAG

3.1 Supertagging

We will now describe a completely different perspective on LTAG. The elementary trees associated with a lexical item can be treated as if they are super parts-of-speech (super POS (parts-of-speech) or supertags), in contrast to the standard POS such as V (verb), N (noun) etc. Now, it is well known that local statistical techniques can lead to remarkably successful disambiguation of standard POS. Can we apply these techniques for disambiguating supertags, which are very rich descriptions of the lexical items? If we can, then, indeed, this will lead to *almost* parsing. This approach is called supertagging (Joshi & Srinivas, 1994; Srinivas & Joshi, 1998).

In Fig. 15, two elementary trees associated with the lexical item *likes* are shown. These are the same trees we have seen before. However, now we are going to regard these trees as super part-of-speech (supertags) associated with *likes*. Given a corpus parsed by an LTAG grammar, we can compute the statistics of supertags, statistics such as unigram, bigram, and trigram frequencies. Interestingly, these statistics combine not only lexical statistics but the statistics of constructions (as represented by the elementary trees) in which the items appear, thus combining lexical statistics with the statistics of the linguistic environments in which the lexical items appear.

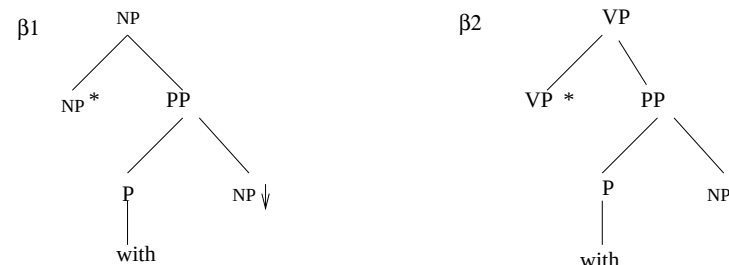


Fig. 15. Two supertags for *likes*

Thus, for example, consider the string

The purchase price includes two ancillary companies

as shown in Fig. 16. The supertags associated with each word appear on top

of that word. Some words have only one supertag associated with them and others have more than one. In the current system there are about 15-20 supertags per word on the average, so there is a very high level of local ambiguity. In Fig. 17, the same supertags are shown for each word; however, for each word one supertag has been identified (in a box). This is the *correct* supertag for this word, in the sense that this is the supertag associated with this word in the correct parse of this sentence. Suppose we are able to find the correct supertag for each word in this sentence by applying local statistical disambiguation techniques, then for all practical purposes we will have parsed the sentence. It is not a complete parse because we have not put the supertags together, hence we call it an *almost* parse.

	$\alpha 1$	$\alpha 2$	$\alpha 3$		$\alpha 4$	$\alpha 5$
	$\beta 2$	$\alpha 6$	$\alpha 7$		$\beta 4$	$\alpha 8$
$\beta 1$	$\alpha 9$	$\alpha 10$	$\alpha 11$	$\beta 3$	$\alpha 12$	$\alpha 13$
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
the	purchase	price	includes	two	ancillary	companies

Fig. 16. A sentence with supertags for each word

	$\alpha 1$	$\alpha 2$	$\alpha 3$		$\alpha 4$	$\alpha 5$
	$\beta 2$	$\alpha 6$	$\alpha 7$		$\beta 4$	$\alpha 8$
$\beta 1$	$\alpha 9$	$\alpha 10$	$\alpha 11$	$\beta 3$	$\alpha 12$	$\alpha 13$
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
the	purchase	price	includes	two	ancillary	companies

- Select the correct supertag for each word -- shown boxed
- Correct supertag for a word means the supertag that corresponds to that word in the correct parse of the sentence

Fig. 17. A sentence with the correct supertag for each word

A supertagging experiment was carried out using trigrams of supertags¹² and techniques similar to the standard POS disambiguation techniques (Joshi & Srinivas, 1994; Srinivas & Joshi, 1998). The corpus used was the Wall Street Journal Corpus (WSJ). With a training corpus of 1 million words and a test corpus of 47,000 words, the baseline performance was 75% (i.e., 75% of the words received the correct supertag). The baseline corresponds to the case when the supertag chosen for a word is just the most frequent supertag for this word. We know from the performance of disambiguators for the standard POS

¹² These are trigrams of supertags and not just trigrams of words or even words with standard POS. These trigrams are trigrams of words, together with the supertags associated with these words.

that the baseline performance is 90% or better. The low baseline performance for supertagging is due to the fact that the local ambiguity is very high (about 15 to 20 on the average) in contrast to the local ambiguity of standard POS, which is about 1.5 for English. The performance of the trigram supertagger, on the other hand, is 92%. The improvement from 75% to 92% is indeed very remarkable. This means that 92% of the words received the correct supertag. More recent experiments based on other machine learning techniques have pushed the performance to about 93% (Chen & Vijay-Shanker, 2000; Shen & Joshi, 2003).

Of course, more can be said about this supertagging approach. There are techniques to improve the performance and to make the output look more like a complete parse. We will not discuss these aspects; rather, we will talk about the abstract nature of supertagging and its relevance to the use of the CLSG approach. In supertagging we are working with complex (richer) descriptions of primitives (lexical items in our case). The descriptions of primitives are complex because we try to associate with each primitive all information relevant to that primitive. Making descriptions more complex has two consequences: (1) local ambiguity is increased, i.e., there are many more descriptions for each primitive, however, (2) these richer descriptions of primitives *locally* constrain each other. There is an analogy here to a jigsaw puzzle—the richer the description of each piece the better, in the sense that there are stronger constraints on what other pieces can go together with a given piece. Making the descriptions of primitives more complex allows us to compute statistics over these complex descriptions but, more importantly, these statistics are more meaningful because they capture the relevant dependencies directly (for example, word-to-word dependencies, where each word is the lexical anchor of some supertag, and word-to-construction dependencies). Local statistical computations over these complex descriptions lead to robust and efficient processing. Supertagging by itself is not full parsing. However, parsing a sentence already supertagged is far more efficient (faster), on the average, as compared to parsing without supertagging. Supertagging is thus an example of a local computation on complex descriptions. Psycholinguistic relevance of supertagging is described in Section 4.

These considerations are directly relevant to other domains, such as AI. We can illustrate this by pointing out interesting relationships to the well known algorithm in (Waltz, 1975) for interpreting line drawings. What Waltz did was to make the descriptions of vertices more complex by adding information about the number and types of edges incident on a vertex. Again there is an analogy here to a jigsaw puzzle: the richer the description of a piece the better. By making the descriptions of vertices more complex, the local ambiguity was increased, for example, an L junction (a particular kind of junction in the taxonomy of junctions of edges) has about 92 physically possible labelings. However, local computations on these complex descriptions are adequate to

rapidly disambiguate these descriptions, leading to efficient computation.

3.2 Flexible composition

In this Section, we will take a different perspective on the operation of adjoining and explore its implications for linguistic description. Psycholinguistic implications are described in Section 4. In adjoining, we insert an auxiliary tree, with the root and foot nodes labeled with X , into a tree at a node with label X . In Fig. 18 and Fig. 19, we present an alternate perspective on adjoining, leading to the notions of multi-component LTAGs and flexible composition.

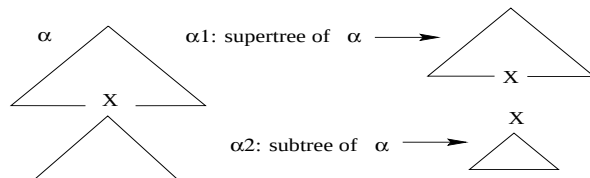


Fig. 18. Adjoining as Wrapping 1

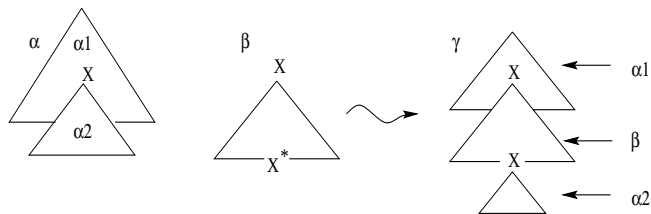


Fig. 19. Adjoining as Wrapping 2

The tree α which receives adjunction at X can be viewed as made up of two trees, the supertree at X and the subtree at X as shown in Fig. 18. Now, instead of the auxiliary tree β adjoined to the tree α at X , we can view this composition as a wrapping operation—the supertree of α and the subtree of α are wrapped around the auxiliary tree β , as shown in Fig. 19. The resulting tree γ is the same as before. Wrapping of the supertree at the root node of β is like adjoining at the root (a special case of adjoining) and the wrapping of the subtree at the foot node of β is like substitution. Hence, this wrapping operation can be described in terms of substitution and adjoining. This is clearly seen in the linguistic example in Fig. 20 and Fig. 21. The auxiliary tree β can be adjoined to the tree α at the indicated node in α as shown in Fig. 20. Alternatively, we can view this composition as adjoining the supertree α_1 (the *wh* tree) at the root node of β and substitution of the subtree α_2 (the *likes* tree) at the foot node of β as shown in Fig. 21. The two ways of composing α and β are semantically coherent, in the sense that both ways of composing

lead to the same semantics. The difference is that, in one case, β serves as the function and α as the argument and, in the other case, the two components of α (the supertree and the subtree of α at X) serve as the function and β serves as the argument¹³.

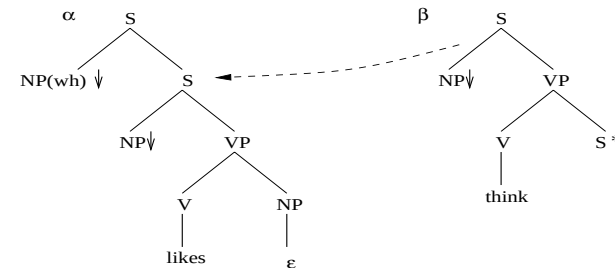


Fig. 20. Wrapping as substitution and adjunction 1

The wrapping perspective¹⁴ can be formalized in terms of the so-called *multi-component LTAGs* (MC-LTAGs). They are called multi-component because the elementary objects can be sets of trees; in our examples, we have two components (in which α was split). When we deal with multi-components, we can violate the locality of the composition very quickly because the different components may be attached¹⁵ (by adjoining or substitution) to different

¹³ Such flexibility is also available in a CFG based system. Thus, in a CFG rule $A \rightarrow BC$, B can be treated as a function and C as the argument, resulting in A , or alternatively, C as the function and B as the argument, also resulting in A . However, it is easily seen that this flexibility does not lead to any increase in the strong generative capacity (no additional structural descriptions). This is so because a CFG rule is essentially a string rewriting rule and we are composing essentially flat structures. In an LTAG, we are composing larger structures (extended domains of locality) and therefore, flexibility of composition can provide greater strong generative capacity. In a categorial grammar (CG), A , B , and C are structured categories. Flexible composition (which is achieved by the operation of type raising) can give additional structural descriptions but not to the extent that is possible in LTAG. This is because, although both LTAG and CG encode the arguments of a lexical item, in the LTAG elementary structures, the different positions occupied by each argument are also encoded. It is this property of LTAG and the fact that in LTAG we compose trees and not strings as in a CFG or CG that leads to the increased strong generative power.

¹⁴ Readers familiar with categorial grammars will notice some similarity to the wrapping rule in (Bach, 1988), which he introduced to relate the structure of *persuaded John to leave* to the structure of *persuade to leave John*, the former derived from the latter by an operation of wrapping. The wrapping operation we have defined is much more general and involves structures far more articulated than those in CFG, CG, and Bach's wrapping rule.

¹⁵ We will use the term *attachment* as a cover term for both substitution and ad-

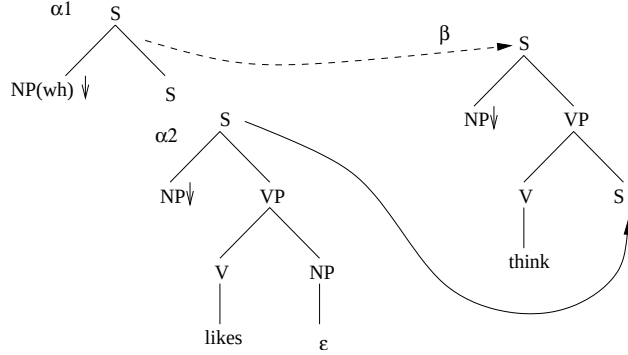


Fig. 21. Wrapping as substitution and adjunction 2

nodes of a tree and these nodes may or may not be part of an elementary tree, depending on whether the tree receiving the multi-component attachments is an elementary or a derived tree. The obvious and natural way to preserve locality is to require that when the components of a multi-component tree (say, α_{11} and α_{12}) attach to a tree (say, β) then β must be an elementary tree, i.e., a local domain. This requirement leads to what are known as tree-local MC-LTAGs. It is known that *tree-local MC-LTAGs* are weakly equivalent to LTAGs, i.e., we do not get increased weak generative power beyond LTAG. However, they can give rise to structural descriptions which are not obtainable by LTAGs, i.e., they are more powerful than LTAGs, in the sense of strong generative capacity (Weir, 1988). Thus, this alternate perspective leads to greater strong generative capacity without increasing the weak generative capacity.

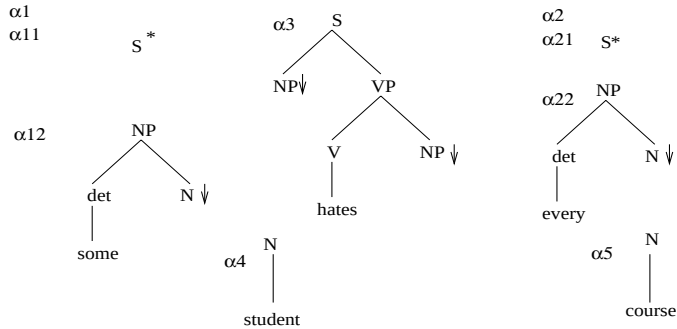


Fig. 22. Scope ambiguity: An example

We will now present an example illustrating the use of this alternate perspective joining.

tive in characterizing the *scope ambiguity* in:

Some student hates every course

This ambiguity arises because in the above example, either *some* scopes over *every* or vice versa, i.e., either ‘there is some student (say, s) who hates every course’ or ‘for each course (say, c) there is some student (say, t) who hates that course’; two different courses need not be hated by the very same student’. This is illustrated in Fig. 22, Fig. 23, and Fig. 24 (Kallmeyer & Joshi, 1999). In Fig. 22, we show a tree-local MC-LTAG for our example¹⁶. The trees for *hates*, *student*, and *course* are standard LTAG trees. The trees for *some* and *every* are multi-component trees. For example, the tree α_1 for *every* has two components, α_{11} and α_{12} , one of the components α_{11} is a degenerate tree in this special case. The multi-component tree, α_1 , is lexically anchored by *some*. Similarly, for the tree α_2 for *every*. The main idea here is that the α_{12} component corresponds to the contribution of *some* to the predicate-argument structure of the tree for *hates* and the α_{11} component contributes to the scope structure. Similarly for the two components of α_2 .

Fig. 23 shows the derivation. The main point to note here is that the two components of α_1 are attached (by substitution or adjoining) to α_3 at the appropriate nodes simultaneously. This composition is tree local as α_3 is an elementary tree. Similarly for the tree α_2 . In this example, the two top components α_{11} and α_{21} are attached to the same node (the root node) of α_3 .¹⁷ This may give the impression that the composition is non-local because once α_1 is attached to α_3 we have a derived tree to which α_2 is attached. However, the two components, α_{11} and α_{21} are degenerate and it can be shown that in this case the composition of α_2 with α_3 (after α_1 has been composed with α_3) is still effectively tree-local (Kallmeyer & Joshi, 1999).

It is clear in this example that α_2 could have been attached to α_3 first and then α_1 attached to α_3 . Fig. 24 shows the derivation tree for the derivation in Fig. 23. The numbers on the edges of the tree refer to the addresses for the attachments. Note that both α_{11} and α_{21} , the scope information carrying components, are attached to α_3 at the same node. Thus, they could be attached in any order (strictly speaking, α_1 and α_2 could be attached to α_3 in any order).

¹⁶ The component α_{11} is an auxiliary tree with just one node. Thus, in this case, the root node and the foot node of this auxiliary tree are the same. The * on the S node has the same meaning as the * on the foot node of an auxiliary tree.

¹⁷ In general, in a multi-component LTAG, multiple adjunctions to the same node are not allowed as this violates the tree-locality and also takes the system beyond LTAG. However, we are dealing here with a special case. The components entering multiple adjunctions are degenerate. It can be shown that in this case the weak generative capacity is still the same as for LTAGs.

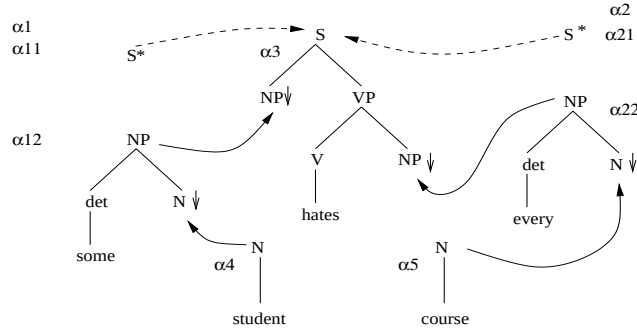


Fig. 23. Derivation with scope information

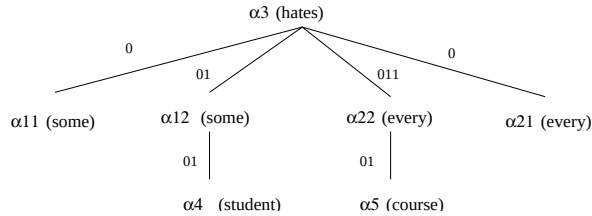


Fig. 24. Derivation tree with scope underspecification

Hence, α_{11} will outscope α_{21} if α_{21} is attached first and then α_{11} and vice versa. The scope ambiguity is thus directly reflected in the derivation tree for

Some student hates every course

This is in contrast to all other approaches (which are essentially CFG based) where the scope ambiguity is represented at another level of representation. It is possible to represent in LTAG the scope ambiguity at the level of the derivation tree itself because of the alternate perspective on adjoining, which in turn is due to the extended domain of locality, a consequence of the CLSG approach.

More recently, similar ideas have been explored in the context of other linguistic phenomena such as scrambling and clitic climbing, both with respect to linguistic coverage and certain psycholinguistic implications. A particularly interesting result is that all word order variations up to two levels of embedding (i.e., three clauses in all) can be correctly described by tree-local MC-LTAGs, correctly in the sense of providing the appropriate structural descriptions (correct semantics). Beyond two levels of embedding, not all patterns of word order variation will be correctly described (see (Joshi, Becker, & Rambow, 2002) for details). Here, we will describe the main idea informally. Let us assume that

we have three clauses, say, $C1$, $C2$, and $C3$, each clause can be a single LTAG elementary tree or a multi-component TAG (MC-LTAG) tree set with two components. We assume that the tree which involves a verb in $C1$ takes as an argument the tree containing a verb in $C2$, i.e., $C1$ embeds $C2$. Similarly, we assume that $C2$ embeds $C3$. Thus, we have here a situation of two levels of center embeddings of complements. where $C1$ embeds $C2$ and $C2$ embeds $C3$. Now, it is possible to show (essentially by exhaustive enumeration) that, using flexible composition, all word orders can be generated preserving the embedding relationships, i.e., with correct semantics. Flexible composition allows us to compose the clauses in three ways, preserving semantics, (1) $C3$ with $C2$ and then the result with $C1$, (2) $C1$ with $C2$ and then the result with $C3$, or (3) both $C1$ and $C3$ into $C1$. This third mode of composition is crucial to complete the proof. However, if we take four clauses, i.e., we have three levels of center embeddings of complements, then it can be shown (again by exhaustive enumeration) that there is at least one word order that cannot be generated without violating the semantics. The "trick" of using the third mode of composition described above does not work beyond two levels of center embeddings of complement clauses. Its use leads to violation of semantics for some cases. Some psycholinguistic implications of this result are discussed in Section 4 (see also Joshi, Becker, & Rambow (2002); Kulick (2000)).

4 Processing issues

In this Section, we will discuss the implications of the TAG architecture for certain processing issues. These pertain to (1) using supertags to make fine grained distinctions between lexical and structural ambiguities and their relevance to processing, (2) the relative processing complexities of certain embedding constructions, and (3) a different perspective on competence performance distinction.

4.1 Supertags in psycholinguistic models

Recently, there has been increasing convergence of perspectives in the fields of linguistics, computational linguistics, and psycholinguistics, especially with respect to the representation and processing of lexical and grammatical information. More specifically, this convergence is due to a shift to lexical and statistical approaches to sentence parsing. The particular integration of lexical and statistical information proposed in (Kim, Srinivas, & Trueswell, 2002) is highly relevant from the perspective of the LTAG architecture. As we have seen before in Section 3.1, LTAG associates with each lexical item one or more elementary structures (supertags), which encapsulate the syntactic and

associated semantic dependencies. The computational results in supertagging as described earlier (Section 3.1) show that much of the computational work of linguistic analysis, which is traditionally viewed as the result of structure building operations, can be viewed as lexical disambiguation, in the sense of supertag disambiguation. If the supertagging model is integrated in a psycholinguistics framework then one would predict that many of the initial processing commitments of syntactic analysis are made at the lexical level, in the sense of supertagging. The model proposed in Kim, Srinivas, & Trueswell (2002) is an integration of the Constraint-Based Lexicalist Theory (CBL) (MacDonald, Pearlmutter, & Seidenberg, 1994), where the lexicon is represented as supertags with their distributions estimated from corpora as in the supertagging experiments described earlier (Section 3.1).

For example, in this model, there is a distinction between the prepositional phrase attachment ambiguity (PP ambiguity) as in (1) below

- (1) *I saw the man in the park with a telescope*

and the PP attachment ambiguity as in (2) below

- (2) *The secretary of the general with red hair*

In the first case, the PP (*with a telescope*) either modifies a noun phrase, *the man* or a verb phrase VP, headed by *saw*. There are two supertags associated with the preposition *with*, as in Fig. 25, one with the foot and root nodes being NP (supertag β_1) or both being VP (supertag β_2). That is, the ambiguity is resolved if we pick the correct supertag for *with* anchored on the preposition *with*. Thus, this PP attachment ambiguity will be resolved at the lexical level. However, in the second case, in both readings of (2) the supertag associated with *with* is the one whose root and foot nodes are both NP. Thus, in this case, the ambiguity will not be resolved at the lexical level. It can only be resolved at the level when the attachment is computed.

The first PP attachment ambiguity is not really an attachment ambiguity. It should be resolved at an earlier stage of processing. In the second case, it will be resolved at a later stage. Similarly, the ambiguity associated with a verb such as *forgot*, because it can take either an NP complement as in (3) below

- (3) *The student forgot her name*

or a VP complement as in (4) below

- (4) *The student forgot that the homework was due today*

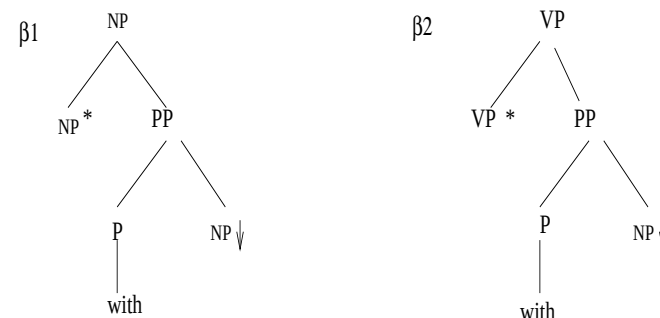


Fig. 25. Two supertags for *with*

is a lexical (supertag) ambiguity and need not be viewed as a structural ambiguity. Kim, Srinivas, & Trueswell (2002) present a neural net based architecture using supertags and confirm these and other related results.

4.2 Processing of crossed and nested dependencies

Context-free grammars (CFG) and the associated automata, pushdown automata (PDA) have been extensively used in modeling several aspects of sentence processing. LTAGs are more powerful than CFGs both weakly, i.e., in terms of strings they produce and strongly, i.e., in terms of structures they produce. Further, LTAGs are associated with embedded pushdown automata (EPDA) (see Vijay-Shanker (1987)), an extension of pushdown automata, i.e., for every LTAG, G , there is an EPDA, M that recognizes the language of G and vice versa¹⁸ TAGs and EPDAs provide a new perspective on the relative ease or difficulty of processing crossed and nested dependencies which arise in center embedding of complement constructions. We repeat here the examples of these constructions as discussed in Section 3.

- (4) *Hans₁ Peter₂ Marie₃ schwimmen₃ lassen₂ sah₁*

- (5) *Jan₁ Piet₂ Marie₃ zag₁ laten₂ zwemmen₃*

- (6) *Jan saw Piet make Marie swim*

In (4) we have the nouns and the corresponding verbs in a nested order in a German construction and in (5) we have a crossed order in a related construction in Dutch. The indices on the nouns and verbs show these dependencies.

¹⁸ EPDAs are thus a precise computational model for the languages of LTAGs. For a brief description of EPDA, see item 3 in Section 2.2. Lack of space prevents us from giving full details of this processing model.

The English word order is shown in (6). These are called complement embedding constructions because each verb is embedded in a higher verb of which it is a complement, except, of course, the matrix (top level and tensed) verb, i.e., *make* is embedded in *saw* and *swim* is embedded in *make*, *saw* is the matrix verb. In (4) and (5) we have center embedding of the complements. In the corresponding English construction as in (6) we do not have center embedding, the complements are just iterated.

The main observation for our purpose is that Standard German prefers the nested order and Standard Dutch prefers the crossed order¹⁹. In the well-known study by Bach, Brown, & Marslen-Wilson (1986) the authors investigated the consequences of these differences between German and Dutch for the processing complexity of sentences, containing either nested or crossed dependencies. Stated very simply, their results show that Dutch is *easier* than German. More specifically, in their study “German and Dutch subjects performed two tasks—ratings of comprehensibility and a test of successful comprehension—on matched sets of sentences which varied in complexity from a simple sentence to one containing three levels of embedding”, three levels means three verbs and three nouns, as in our examples above. Their results show “no difference between Dutch and German for sentences within the normal range (up to one level), but with a significant preference emerging for the Dutch crossed order. These results in Bach, Brown, & Marslen-Wilson (1986) show that pushdown automaton (PDA) cannot be the universal basis for human parsing mechanism (as the authors themselves point out). They offer an explanation for the inadequacy of PDA, based on the kinds of partial interpretations that the nested and crossed dependencies allow the listener to construct. Their main suggestion was “that the most important variable in successful parsing and interpretation is not simply *when* information becomes available, but also *what* you can do with that information when you get it.” Thus in (4) (German example), when the deepest noun (noun phrase) and verb are reached (*Marie schwimmen*), we have a verb and its argument, however, we do not know at this stage where this structure belongs, i.e., we do not have a higher structure into which we can integrate this information. Hence, we must hold this information until a higher structure becomes available. The same consideration holds for the second noun and the second verb (*Peter lassen*).

In contrast, in (5) (Dutch example), we begin to build the matrix of higher verbs as soon as the verb cluster begins, i.e., beginning with the matrix verb *zag* and the corresponding noun argument which can be integrated, similarly for the second noun and the second verb, and for the third noun and the third verb. In this case, we do not create intermediate structures that do not have a place to fit into. Thus, we first build the structure for the matrix verb into

which the next verb (with its structure) can be integrated, and similarly for the last verb.

The nested dependencies in German permit integration of structures (innermost to outermost) in a context-free manner (hence processed by PDA) but it is not possible to decide what to do with that information until the higher level verb(s) become available. Therefore, PDA will not serve as a processing model for the German case (even though the dependencies are nested) and, of course, it will not serve for the Dutch case, because PDAs cannot model crossed dependencies in the first place. Now, it turns out that EPDA can model both crossed and, of course, nested dependencies and the interpretations can be carried out in the way suggested in Bach, Brown, & Marslen-Wilson (1986), i.e., when a noun verb structure is integrated, the structure to which it belongs has already been created before (Joshi, 1990). We now introduce the following measure²⁰ of complexity—maximum number of items (say, m) from the input that have to be held back before the sentence parsing (interpretation) is complete. In Joshi (1990) it is further shown that on this measure Dutch is easier (smaller m) than German, in fact, approximately in the same proportion as in the experimental data in Bach, Brown, & Marslen-Wilson (1986), m for German is approximately twice that for Dutch. Further, linguistically relevant TAG grammars for (4) and (5) correspond (in the formal sense of the association between EPDAs and TAGs) to the EPDAs that correctly model the processing of (4) and (5).

4.3 A new twist to the competence performance distinction

How do we decide whether a certain property of language is a competence property or a performance property? This is an old question. Our main claim in this Section is that the answer to this question is not given a priori. It depends on the formal devices available to us for describing language. If a formal device corresponding to the property is available then we are presented with a choice. We can claim that the property is either a competence property or a performance property. When there is no formal device available then we are left with no choice but to treat the property as a performance property.

In the context of the LTAG framework, especially, the multi-component LTAG (MC-LTAG), discussed in Section 3, this question has an interesting answer when we look at a variety of word order phenomena, such as scrambling, clitic climbing, and even scope ambiguities. Since the CLSG approach leads directly to LTAG, the claim here is that it is the CLSG approach that leads to this

¹⁹ Strictly speaking this word order appears in subordinate clauses. See also Kroch & Joshi (1985); Kroch (1989); Kroch & Santorini (1991).

²⁰ suggested by a measure used in Bach, Brown, & Marslen-Wilson (1986) for evaluating the complexity of comprehension.

answer²¹

As a particular example, consider the phenomenon of *scrambling* (in German, for example) as illustrated below. We repeat (4) in Section 4.2 as (7) below.

(7) *Hans₁ Peter₂ Marie₃ schwimmen₃ lassen₂ sah₁*

The three verbs are in the standard order. However, it is possible, in principle, to have the three nouns in any order different from the order in (7), keeping the verbs in the same order as in (7). Thus (8) below is a scrambled version of (7).

(8) *Hans₁ Marie₃ Peter₂ schwimmen₃ lassen₂ sah₁*

Thus, in general, we can have

(9) $P(N_1, N_2, \dots, N_k) V_k V_{k-1} \dots V_1$

where $P(N_1, N_2, \dots, N_k)$ stands for some permutation of the k nouns. It is not surprising that sentences involving scrambling from more than two levels of embedding are indeed difficult to interpret and native speakers show reluctance to accepting these sentences. This situation is reminiscent of the difficulty of processing English sentences with more than two (or perhaps even more than one) center embedded relative clauses, as in (10) below.

(10) *The rat the cat the dog chased bit ate the cheese*

Such a difficulty is characterized as a performance property, say, P , where P is the property that beyond two levels²² of embedding the human processing device, whatever it is, simply fails. Thus, in analogy to the center embedding of relative clauses in English, we could say that in the case of scrambling, processing difficulty (or failure to process) for sentences with center embedding of complement clauses beyond two levels of embedding is also a performance property, say Q . From this perspective, both P and Q are seen as performance properties and not properties of the respective associated grammars. We will point out below that there is a very interesting difference between these two cases.

²¹ The possibility of some other formal approach leading to a similar answer cannot be ruled out. However, to my knowledge, no such formal approach is known so far.

²² We could set this level to one but this is not essential for the rest of the argument of this section. I will keep the level at two in order to make it parallel to the scrambling situation.

Let us first consider the case of center embedding of relative clauses in English. Is it possible to claim the property P is a property of a *class* of grammars, say Γ such that for all grammars in this class, P holds, i.e., for each grammar G in Γ , up to two levels of embedding, G will assign correct (i.e. semantically coherent) structural descriptions to all sentences, however, beyond two levels of embedding, there is no guarantee that for all such sentences the grammar will assign correct structural descriptions. If we can exhibit such a class Γ then we could claim that P is a competence property, i.e., it is a property of each grammar in this *class*. In other words, for each grammar G in Γ the property P holds up to two levels of embedding only and then fails to hold, and the failure is due to the inability of G to assign correct structural descriptions to all sentences beyond two levels of embedding. The idea here is that this lack of ability to assign correct structural descriptions (i.e., correct semantics) is the reason for the processing difficulty.

Now, it turns out that we do not know whether such a *class* Γ exists. The class of finite state grammars F will not do because although there is a grammar in this class that will work up to two levels of embedding and then fail, there is clearly another grammar that will work up to three levels of embedding, another up to four levels, and so on. Thus, P does not hold for the class Γ . We could choose the class of context-free grammars (CFG), the next one in the Chomsky hierarchy. In this case, an arbitrary number of center embeddings will be permitted. This is because once we allow center embedding (thus nested dependencies, which characterizes the class CFG) then we cannot put a bound on the number of embeddings. Hence, the class CFG will not work for us also. In fact, as far as we know, at present, there is no class of grammars which has the property P ²³.

Since we cannot find a class of grammars with the property P we have no opportunity to claim that P is competence property. So in a way, we have no choice and we must conclude that P is a performance property²⁴.

However, when we consider the case of scrambling from more than two levels of center embedded complement clauses (see (9) above) we can actually exhibit a class of grammars Δ which has the property that for each grammar in this class, say, G , the property Q (the property corresponding to the property P discussed above) holds, i.e., for each grammar G in this class, G will assign correct structural descriptions to all sentences with scrambling from up to two levels of embedding; however, beyond two levels of embedding, there is no

²³ Obviously, we cannot limit our search to the grammars in the Chomsky hierarchy. Any class of grammars defined in a formal framework is a possible candidate. However, to my knowledge, we do not have such a candidate at present.

²⁴ Miller & Chomsky (1963) were the first to discuss the relationship between formal grammars (and associated machines) and center embedding of relative clauses in English. Since then a large number of publications on this issue have appeared.

guarantee that G will assign correct structural description (correct semantics) to all such sentences. In other words, syntax allows for arbitrary number of levels of embedding but beyond two levels of embedding, correct semantics is not guaranteed for all sentences with more than two levels of embedding. It is shown in (Joshi, Becker, & Rambow, 2002) that the class of multicomponent LTAG (MC-LTAG) is such a class, i.e., it has the property Q .

So, now, we have an opportunity to claim that Q is a competence property, i.e., the difficulty of processing sentences with scrambling from more than two levels of embedding can be characterized as a property of the grammar. Note that the claim here is not that we *must* conclude that Q has to be a competence property. The claim is that we are presented with a *choice*. We can claim Q to be a competence property by adopting Δ (the class MC-LTAG) as the class of grammars for describing scrambling from center embedded complement clauses or we can continue to follow the traditional wisdom (as we did in the case of center embedding of relative clauses in English) and say that Q is a performance property. As far as we know, this is the first example of a situation where we have an opportunity to claim a particular processing difficulty as a competence property.

For the case of center embedding of relative clauses, we had no choice as we are unable to exhibit the relevant class of grammars, Γ . Structurally, center embedding of relative clauses in English is a different phenomenon as compared to the center embeddings of complement clauses in German. Whether this difference has something to do with the conclusion we have just reached is an open question. The class MC-LTAG does not work for the center embeddings of relative clauses in English, as can be seen in (7) above which, mathematically, looks the same as arbitrary number of center embeddings of relative clauses as in English.

To sum up, we have discussed the following issue: How do we decide whether a certain property of language is a competence property or a performance property? We showed that the answer to this question is not given *a priori*. It depends on the formal devices (formal grammars and machines) available to us for describing language. We showed that in the case of center embedding of relative clauses in English, we were forced to conclude that the property that sentences with more than two levels of embedding are difficult to interpret is a performance property and not a competence property. In contrast, for the case of scrambling (as in German), with respect to a corresponding property we can indeed exhibit a class of grammars such that, for all grammars in this class, the corresponding property holds. Thus, we are given the opportunity to claim this property to be a competence property, rather than a performance property. To the best of our knowledge, this is the first such case where a choice of this kind is presented. The lack of such a choice before this case was discovered has led to the traditional assumption that all properties similar to

the processing difficulty associated with center embedding of relative clauses in English should be treated as performance properties. Our main conclusion is that this assumption is not justified at all!

To avoid any misunderstanding, let us restate the main claim as follows. We are not claiming that by finding an appropriate class of grammars we can conclude that a property associated with processing difficulty automatically becomes a competence property. What we are claiming is that in this case we are presented with a *choice*. Without an associated class of grammars, we have no choice but to regard the property as a performance property. Until the appropriate class of grammars associated with the property discussed in this section (property Q) was discovered, we had no choice. Now we have a choice!

4.4 Related work on processing

Implications of the LTAG architecture have been explored in the areas of language acquisition and production (generation). Frank (1998) has studied structural complexity and the time course of grammatical development. He attempts to relate children's difficulties with certain constructions to the processing load and representational complexity in the context of a particular view of syntactic representation, which is derivable from the TAG architecture. Ferreira (2000) presents a perspective on production that relates the TAG architecture to a range of experiments about production.

Insights from the LTAG theory have been extended to the domain of discourse structure, primarily by treating discourse connectives as lexical anchors of LTAG like trees at the discourse level, thereby, blurring the line between sentence level and discourse level descriptions. This work is described in a series of recent papers (Webber, Joshi, Knott & Stone, 1999; Forbes, Miltsakaki, Prasad, Sarkar, Joshi, & Webber, 2001; Webber, Stone, Joshi & Knott, 2003).

5 Summary

We have described an approach to formal systems where we start with complex (rather than simple) primitive structures, complex enough to localize all relevant properties of lexical items. Under this approach (characterized as the CLSG approach—complicate locally, simplify globally), the grammar is characterized by the (finite) set of primitive structures together with two combining operations (substitution and adjoining), which are universal. Specification of the finite set of primitives is the linguistic theory, in a sense. We have illus-

trated the formal, linguistic, and computational consequences of the CLSG approach in the context of the formal system known as the Lexicalized Tree-Adjoining Grammar (LTAG) and some of its variants. We have also discussed the implications of this approach for several psycholinguistic issues. In general, under this approach, several important properties of language arise as corollaries of the formal system and therefore, they are not stipulative. Thus, this research is an example of productive interplay between formal analysis on the one hand and linguistic and processing issues on the other hand.

6 Acknowledgements

This work was partially supported by the National Science Foundation Grant STC-SBR 8920230. My work on the formal analysis of language began a little over three decades ago. I want to take this opportunity to express my sincere thanks to all my past and present students, collaborators, colleagues, and mentors (too many to enumerate by name) for their wonderful insights, advice, and comments throughout this period. I also greatly appreciate all the constructive comments provided by the three reviewers and the action editor, which resulted in significant improvements in the final version of this paper.

References

- Abeillé, A. & Candito, M. 2002. FTAG: A lexicalized tree adjoining grammar for French. In A. Abeillé and O. Rambow (Eds), *Tree-Adjoining Grammar*, CSLI, Stanford.
- Abeillé, A. 2002. Une grammaire électronique du français. CNRS Éditions. Paris.
- Bach, E. 1988. Categorical Grammars as Theories of Language. In R. T. Oehrle, E. Bach, & D. Wheeler (Eds), *Categorical Grammars and Natural Language Structures*, Dordrecht:Reidel, 17-34.
- Bach, E., Brown, C., & Marslen-Wilson, W. 1986. Crossed and nested dependencies in German and Dutch: a psycholinguistic study. *Language and Cognitive Processes*, 1(4), 249-262.
- Chen, J.C. & Vijay-Shanker, K. 2000. Automated extraction of the TAGs from the Penn Treebank. In *Proceedings of the 6th International Workshop on Parsing Technology (IWPT)*, 65-76.
- Chiang, D. Statistical parsing with an automatically extracted tree adjoining grammar. In *Proceedings of the Association for Computational Linguistics (ACL) 2000 Meeting*.
- Copestake, A., Flickinger, D., Sag, I., & Pollard, C. 1999. Minimal recursion semantics: An introduction. *Manuscript*, Stanford University.

- Frank, R. 1998. Structural complexity and the time course of grammatical development. *Cognition*, 66, 249-301.
- Frank, R. 2002. *Phrase Structure Composition and Syntactic Dependencies*. MIT Press, Cambridge.
- Ferreira, F. 2000. Syntax in Language Production: An Approach Using Tree-Adjoining Grammars. In L. Wheeldon (Ed), *Aspects of Language Production*, to appear.
- Forbes, K., Miltsakaki, E., Prasad, R., Sarkar, A., Joshi, A. K., & Webber, B. 2001. D-LTAG System-discourse parsing with a lexicalized tree-adjoining grammar. In *ESSLI'2001. Workshop on Information Structure, Discourse Structure and Discourse Semantics*, Helsinki.
- Joshi, A. K., Levy, L. S. & Takahashi, M. 1975. Tree adjunct grammars. *Journal of Computer and System Sciences*, 10:1.
- Joshi, A. K. 1985. Tree-adjoining grammars: How much context sensitivity is required to provide reasonable structural descriptions? In D. Dowty, L. Karttunen, & A. Zwicky (Eds), *Natural Language Parsing*, Cambridge University Press, Cambridge, 206-250.
- Joshi, A. K. 1990. Processing crossed and nested dependencies: An automaton perspective on the psycholinguistic results. *Language and Cognitive Processes*, 5, 1-27.
- Joshi, A.K., Vijay-Shanker, K., & Weir, D. 1991. The convergence of mildly context sensitive grammatical formalisms. In P. Sells, S. Shieber, & T. Wasow (Eds), *Foundational Issues in Natural Language Processing*. MIT Press, Cambridge.
- Joshi, A.K., & Srinivas, B. 1994. Disambiguation of super parts of speech (Supertags): Almost parsing, In *Proceedings of the 1994 International Conference on Computational Linguistics (COLING)*. Kyoto, Japan.
- Joshi, A. K. & Schabes, Y. 1997. Tree-Adjoining Grammars. In G. Rosenberg & A. Salomaa (Eds), *Handbook of Formal Languages*, Springer, Berlin, 69-123.
- Joshi, A. K. & Kulick, S. 1997. Partial proof trees as building blocks for a categorial grammar. *Linguistics and Philosophy*, 20, 637-667.
- Joshi, A.K. & Vijay-Shanker, K. 1999. Compositional semantics with lexicalized tree-adjoining grammar (LTAG): How much underspecification is necessary? In H.C. Bunt & E.G.C. Thijsse (Eds), *Proceedings of the Third International Workshop on Computational Semantics (IWCS-3)*, Tilburg, 131-145.
- Joshi A. K., Becker T., & Rambow O. 2002. Complexity of Scrambling: A New Twist to the Competence-Performance Distinction. In A. Abeillé & O. Rambow (Eds), *Tree-Adjoining Grammars*, CSLI, Stanford.
- Joshi, A. K. 2003. A note on the strong and weak generative powers of formal systems. *Theoretical Computer Science*, 293, 243-259.
- Joshi, A. K., Kallmeyer, L., & Romero, M. 2003. Flexible composition in LTAG: Quantifier scope and inverse linking. In *Proceedings of the International Workshop on Computational Semantics (IWCS-5)*, Tilburg.

- Kallmeyer, L. & Joshi, A. K. 1999. Factoring predicate argument and scope semantics: Underspecified semantics with LTAG. In *Proceedings of the Twelfth Amsterdam Colloquium*, University of Amsterdam, Amsterdam, 169-174, a revised version appears in *Language and Computation*, 2003.
- Kaplan, R. & Bresnan, J. 1983. Lexical-Functional Grammar: A Formal System of Grammatical Representation. In J. Bresnan (Ed), *The Mental Representation of Grammatical Relations*, MIT Press, Cambridge, MA.
- Kasper, R. Kiefer, B., Netter, K. & Vijay-Shanker, K. 1995. Compilation of HPSG to TAG. In *Proceedings of the Association for Computational Linguistics (ACL)*, 92-99.
- Kim, A., Srinivas, B., & Trueswell, J.C. 2002. The convergence of lexicalist perspectives in psycholinguistic and computational linguistics. In P. Merlo & S. Stevenson (eds), *Sentence Processing and the Lexicon: Formal, Computational and Experimental Perspectives*, John Benjamin Publishing, Philadelphia.
- Kroch, A. & Joshi, A. K. 1985. Linguistic relevance of tree-adjoining grammars. Technical Report, Department of Computer and Information Science, University of Pennsylvania.
- Kroch, A. 1989. Asymmetries in long distance extraction in a tree-adjoining grammar. In M. Baltin & Kroch, A. (Eds), *Alternative conceptions of phrase structure*, University of Chicago Press, Chicago.
- Kroch, A. & Santorini, B. 1991. The derived constituent structure of the West Germanic verb raising constructions. In R. Freiden (Ed), *Principles and parameters in comparative grammar*, MIT Press, Cambridge, 269-338.
- Kulick, S. 2000. *Constrained non-locality: Long-distance dependencies in TAG*. Ph.D. Dissertation, University of Pennsylvania, Philadelphia, PA, USA.
- Linz, P. 2001. *An Introduction to Formal Languages and Automata*, Jones and Bartlett Publishers, Sudbury, MA.
- MacDonald, M.C., Pearlmutter, N.J., Seidenberg, M.S. 1994. Lexical nature of syntactic ambiguity resolution. *Psychological Review*. 101, 676-703.
- Miller, P. M. 1999 Strong Generative Capacity, CSLI Publications, Stanford University, Stanford CA.
- Miller, G. A. & Chomsky, N. 1963. Finitary models of language users. In R.D. Luce, R. Bush & E. Galanter (Eds), *Handbook of Mathematical Psychology*, Volume II, Wiley, New York.
- Pollard, C. & Sag, I. A. 1994. *Head-Driven Phrase Structure Grammar*. Chicago University Press, Chicago.
- Prolo, C. 2003. LR parsing for tree-adjoining grammars and its applications to corpus based natural language parsing. Ph.D. Dissertation, University of Pennsylvania, Philadelphia.
- Rambow, O. 1994. *Formal and computational aspects of natural language syntax*. Ph.D. Dissertation, University of Pennsylvania, Philadelphia.
- Resnik, P. 1992. Probabilistic tree-adjoining grammars as a framework for statistical natural language processing. In *Proceedings of COLING '92*, Nantes, 2, 418-424.
- Sarkar, A. 2002. Combining labeled and unlabeled data in statistical natural language processing. Ph.D. Dissertation. University of Pennsylvania, Philadelphia.
- Schabes, Y. Stochastic lexicalized grammars. In *Proceedings of COLING '92*, 2, Nantes, 426-432.
- Schabes, Y. & Waters, R. C. 1994. Tree insertion grammars. Technical Report TR-94-13. Mitsubishi Electric Research Laboratories, Cambridge.
- Shen, L. & Joshi, A. 2003. A SNoW based supertagger the applications to NP Chunking. In *Proceedings of the Association for Computational Linguistics Meeting (ACL)*. Sapporo, Japan, 89-96.
- Srinivas B. & Joshi A. K. 1998. Supertagging: An approach to almost parsing. *Computational Linguistics*, 22, 1-29.
- Stabler, E. 1997. Derivational minimalism. In C. Retore (Ed), *Logical Aspects of Computational Linguistics*, Lecture Notes in Artificial Intelligence 1328, Springer Verlag, Heidelberg, 68-95.
- Steedman, M. 1996. *Surface Structure and Interpretation*. MIT Press, Cambridge, MA.
- Stone, M. & Doran, C. 1999. Sentence planning as description using tree-adjoining grammar. In *Proceedings of the Association for Computational Linguistics (ACL) Meeting*, Madrid.
- Vijay-Shanker K. 1987. *A Study of Tree-Adjoining Grammars*, Ph.D. Dissertation, University of Pennsylvania, Philadelphia.
- Waltz, D. 1975. Understanding line drawings of scenes with shadows. In P.H. Winston (ed), *The Psychology of Computer Vision*, McGraw-Hill, New York.
- Webber B., Joshi A. K., Knott A., & Stone M. 1999. What are little texts made of? A structural and presuppositional account using lexicalized TAG. In *Proceedings of the International Workshop on Levels of Representation in Discourse (LORID '99)*, Edinburgh, 151-156.
- Webber, B., Stone, M., Joshi, A. K., & Knott, A. 2003. Anaphora and discourse structure, *Computational Linguistics*, 29, 545-588.
- Weir, D. 1988. *Characterizing mildly context-sensitive grammar formalisms*. Ph. D. Dissertation. University of Pennsylvania, Philadelphia.
- The XTAG Research Group 2002. A Lexicalized Tree-Adjoining Grammar for English. *Technical Report*, 02-18, Institute for Research in Cognitive Science, University of Pennsylvania, Philadelphia.