STUART M. SHIEBER

# DIRECT PARSING OF ID/LP GRAMMARS

## 1. INTRODUCTION

The Immediate Dominance/Linear Precedence (ID/LP) formalism is a recent extension of Generalized Phrase Structure Grammar (GPSG) (Gazdar and Pullum, 1982; Gazdar and Pullum, 1981) designed to perform some of the tasks previously assigned to metarules – for example, modeling the word-order characteristics of so-called free-word-order languages. (See, e.g., Stucky, 1981; Pullum, 1982; Uszkoreit, 1982.) It allows a simple specification of classes of rules that differ only in constituent order. ID/LP grammars (as well as metarule grammars) have been proposed for use in parsing by expanding them into equivalent context-free grammars (Gazdar and Pullum, 1982; Thompson, 1982). We develop a parsing algorithm, based on the algorithm of Earley, for parsing ID/LP grammars directly, circumventing the initial expansion phase. A proof of correctness of the algorithm is supplied. We also discuss some aspects of the time complexity of the algorithm and some formal properties associated with ID/LP grammars and their relationship to context-free grammars.

## 2. AN INFORMAL INTRODUCTION TO ID/LP GRAMMARS

A complete explication of the motivation for and use of the ID/LP grammar formalism can be found in Gazdar and Pullum (1982). We provide here only a brief and informal introduction to ID/LP so as to convey something of its flavor as a tool for use in formal linguistics.

Languages such as English that rely heavily on word order to convey syntactic information are amenable to the kind of analysis in which the immediate dominance of constituents is correlated on a one-to-one basis with a particular linear ordering on constituents. Such a correlation is provided by the context-free phrase structure formalism. However, for languages in which linear order plays a considerably more limited role, the insistence upon such a strict correlation can be quite inefficient and unintuitive, leading to grammars that overlook key generalizations. For example, in the Bantu language Makua, constituent order is much freer than in English. In a sentence with a verb, a noun phrase complement,

*Linguistics and Philosophy* **7** (1984) 135–154. 0165-0157/84/0072-0135$02.00

and a sentential complement, three orders of the constituents are allowed.[1]

(1)(a) Aho-cúwél-a Híń-Sepété wiírá ikitáábwílé y-orééra u-sómá
SA/T-know-T Sepete that book-DEM SA-good INF-read
*Sepete knows that the book is good to read.*

(b) Híń-Sepété aho-cúwél-a wiírá ikitáábwílé y-orééra u-sómá
(c) Aho-cúwél-a wiírá ikitáábwílé y-orééra u-sómá Híń-Sepété

Rather than write three phrase structure rules for the possible orderings, we would like to express directly the following two orthogonal propositions:

- A sentence can be composed of a verb with *NP* and $\bar{S}$ complements.
- The verb must come before the $\bar{S}$.

That is, we would like to separate the specification of *immediate dominance* from that of *linear precedence*. In the ID/LP formalism presented in this paper, these facts would be encoded, respectively, as an ID rule

(2) $S \rightarrow_{ID} \{V, NP, \bar{S}\}$

and an LP rule

(3) $V < \bar{S}$.

The LP rule has force throughout the entire grammar. It therefore applies to any rule that introduces siblings of categories $V$ and $\bar{S}$. The LP relation is thus the repository of grammar-wide generalizations about linear order. By way of comparison, note that context-free phrase structure grammars offer no way of stating such generalizations.

ID rules are just one of several devices within the GPSG framework that provide a format for the inductive definition of a context-free grammar. The other devices include, most notably, a set of metarules (for deriving ID rules from other ID rules), and feature instantiation rules (for deriving instantiated ID rules from the previously derived ID rules). These devices have been traditionally described in the literature as engendering a process of grammar derivation by expansion from an initial set of basic rules to the end product of a context-free grammar, proceeding through the stages of metarule closure, feature instantiation, and finally, linearization according to the LP relation. An ID rule can thus be thought of (and has indeed been traditionally so regarded) as a schema for a set of

context-free phrase structure rules, namely, the set of 'linearizations' of the rule consistent with the LP relation.

## 3. Advantages of a direct interpretation of ID/LP

However, ID rules need not be so viewed. In fact, they can just as straightforwardly be viewed as node-admissibility constraints themselves – as distinguished from a metagrammatical device for specifying such constraints. This would lead to the possibility of deriving algorithms for parsing and generation that use ID rules directly, without first expanding them into an equivalent context-free grammar. We will discuss several advantages of such a direct interpretation of ID rules, some of them theoretical, some more practical in nature.

First, proponents of GPSG make the claim that a grammar with only context-free power can capture the generalizations that natural languages seem to embody. If GPSG is to be taken seriously as a model for the encoding of information used in human sentence processing, the information actually used by the sentence-processing mechanisms should encode such generalizations as well. That is, the processor should be using the grammar in a form in which generalizations are stated. But the demonstrations of some of the desirable properties of GPSGs, e.g., their generative and computational restrictiveness, are based on their expansion into forms that do not express these generalizations, namely, huge 'object grammars' of literally trillions of rules. Thus, an existence proof is needed that GPSGs can be used directly without the loss of these properties. We will shortly present such an existence proof for at least one segment of the problem, the direct parsing of ID/LP grammars.[2]

Second, if GPSGs are to be used as the basis of a practical computer implementation, such direct-interpretation algorithms will be necessary. The expansion of a GPSG into its equivalent context-free grammar has never been, and undoubtedly could not be, the basis of an implementation of the formalism. In this vein, it is notable that the algorithm to be presented here has already been the basis of two GPSG implementations.[3] In fact, many implementations already do the work of feature instantiation directly without expansion of the grammar (e.g., Bear and Karttunen, 1982; Gawron *et al.*, 1982; Rosenschein and Shieber, 1982) and some work has been done on direct parsing with metarules (see Stucky [1983] and references cited therein). But as linearization is the final stage in the expansion cascade, removal of linearization (by direct parsing of ID/LP) is a prerequisite for removal of the other stages of expansion. Implementations that perform feature instantiation 'on the fly' therefore

cannot include the ID/LP device (unless they parse ID/LP directly). Thus direct parsing of ID/LP is critical to allowing direct parsing of earlier stages in the GPSG cascade.

Third, as formalisms get more complex in their design, it may be impossible to directly expand grammars written in the formalism into a strongly equivalent context-free grammar, even though the formalism is only of context-free power. This is the case with the Kleene star notation in ID rules (Klein, 1983), and the extended version of GPSG due to Culy (1983). In these cases, the need for direct interpretation is obvious. In fact, in Section 6 we discuss application of the ID/LP parsing algorithm to the Kleene star notation.

Finally, although we will see that the ID/LP formalism and context-free grammars have virtually the same generative capacity (in a sense to be made precise later) and the respective algorithms for parsing have the same time complexity, it seems that under certain conditions, the ID/LP representation of grammars would allow more efficient parsing. This factor will be discussed in Section 7.

## 4. A FOMAL DEFINITION OF ID/LP GRAMMARS

We now provide a formalization of ID/LP grammars that is independent of their linearization into context-free grammars. We will use this formalization to define and prove the correctness of a parsing algorithm for ID/LP grammars, as well as to discuss the mathematical properties of such grammars. The formalism defined here differs slightly from the one implicit in Gazdar and Pullum (1982) in being somewhat more restrictive. We will mention these differences where they arise.

An ID/LP grammar $G$ is a quintuple $\langle N, \Sigma, ID, LP, S\rangle$ where

- $N$ is a finite nonempty set, the set of nonterminals.
- $\Sigma$ is a finite nonempty set disjoint from $N$, the set of terminals.
- $ID$ is a finite set of ordered pairs $\langle \alpha, \beta\rangle$, where $\alpha \in N$ and $\beta \in \mathcal{P}(N \cup \Sigma)$.

$\langle \alpha, \beta\rangle \in ID$ is notated $\alpha \rightarrow_{ID} \beta$. Note that the right-hand sides of rules are sets of terminals and nonterminals, so that repeated items are disallowed – unlike the ID/LP characterization of Gazdar and Pullum (1982) and Klein (1983). The extension to multisets, however, is straightforward. We merely let $\beta$ range over multisets of elements of $N \cup \Sigma$ rather than sets. The changes required in the algorithm will be discussed later.

- $LP$ is a strict partial ordering of $N \cup \Sigma$, that is, an irreflexive, asymmetric, transitive relation in $(N \cup \Sigma) \times (N \cup \Sigma)$.

$\langle \alpha, \beta \rangle \in LP$ is conventionally notated $\alpha < \beta$. This notation is extended to apply to sets or strings of symbols so that, if $\beta$ is a set or string, then $\alpha < \beta$ iff $\forall \beta' \in \beta$: $\alpha < \beta'$ and, similarly, for $\alpha$ a set or string. We will also use the symbol $\not>$ both in the standard way and in its extended form covering sets and strings.

– $S$ is an element of $N$, the start symbol.

We now define ID/LP derivation and some concomitant notions. Note that these definitions hold equally well for the multiset characterization of ID/LP under the appropriate interpretations of set operations as their multiset counterparts. (We notate the empty string as $\Lambda$, the empty set as $\emptyset$.)

DEFINITION 1. **LP-acceptability:** A string $\alpha = \alpha_1 \ldots \alpha_n$, where $\alpha_i \in N \cup \Sigma$ *is LP-acceptable* iff $\forall i, j$: $1 \leq i \leq j \leq n \rightarrow (\alpha_i \not> \alpha_j)$.

Intuitively understood, a string is LP-acceptable if the order of its elements does not violate the LP relation.

DEFINITION 2. **The permute function:** A function *permute* from $\mathcal{P}(N \cup \Sigma)$ to $\mathcal{P}((N \cup \Sigma)^*)$ is defined as follows:

– $permute(\emptyset) = \emptyset$, and
– for all $\alpha \in \mathcal{P}(N \cup \Sigma)$, $permute(\alpha) = \{A\beta \mid A \in \alpha \wedge \beta \in permute(\alpha - A)\}$.

Intuitively understood, $permute(\alpha)$ is the set of all strings that are permutations of the elements of the set $\alpha$.

DEFINITION 3. **The expand function:** A function *expand* from $\mathcal{P}(N \cup \Sigma)$ to $\mathcal{P}((N \cup \Sigma)^*)$ is defined as follows:

– $expand(\alpha) = \{\beta \mid \beta \in permute(\alpha)$ and $\beta$ is LP-acceptable$\}$.

Intuitively understood, $expand(\alpha)$ is the set of permutations of $\alpha$ whose order does not violate the LP relation.

We now define the *yields* and *derives* relations and the notion of an ID/LP grammar's *language* in the standard manner.

DEFINITION 4. **The yields relation:** $A$ *yields* $\alpha$ (notated $A \rightarrow \alpha$) iff $A \rightarrow_{ID} \alpha'$ and $\alpha \in expand(\alpha')$.

DEFINITION 5. **The derives relation:** $\alpha A \beta$ *derives* $\alpha \gamma \beta$ (notated $\alpha A \beta \Rightarrow \alpha \gamma \beta$) iff $A \rightarrow \gamma$. The relation denoted by $\Rightarrow^*$ is the reflexive,

transitive closure of the derives relation.

DEFINITION 6. **The language of an ID/LP grammar:** The language $L$ of an ID/LP grammar is defined as follows: $L = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$.

## 5. An algorithm for parsing ID/LP grammars directly

Given an independent characterization of ID/LP, we can pursue the possibility of using ID/LP grammars directly for generation or parsing without first expanding them into an equivalent context-free grammar. We present one such algorithm for parsing strings directly with respect to ID/LP grammars. The algorithm is based on the context-free parsing algorithm of Earley (1970).

The Earley parsing algorithm for context-free grammars works on a string $a_1 \ldots a_n$ by building a set of *parse lists* $I_0, \ldots, I_n$, each list containing several *items* of the form $[A \rightarrow \alpha \,.\, \beta, i]$, where $A \in N$, $\alpha$, $\beta \in (N \cup \Sigma)^*$, $0 \leq i \leq n$, and $A \rightarrow \alpha\beta$ is a production in the context-free grammar. The existence of such an item on parse list $I_j$ means that the algorithm is 'looking for' a parse of the string starting at $a_{i+1}$ as an $A$, and has succeeded in locating the $\alpha$ part of the constituent in the string $a_{i+1} \ldots a_j$, with only the $\beta$ part of the constituent still to be found. We can express this more formally as follows: an item $[A \rightarrow \alpha \,.\, \beta, i]$ is on $I_j$ if and only if $\alpha \Rightarrow^* a_{i+1} \ldots a_j$ and there exist strings $\gamma$, $\delta \in (N \cup \Sigma)^*$, $S \Rightarrow^* \gamma A \delta$ and $\gamma \Rightarrow^* a_1 \ldots a_i$. Clearly, if an item of the form $[S \rightarrow \alpha \,.\,, 0]$ is found on parse list $I_n$, then the algorithm has found an $S$ covering the whole string and the string is accepted. The Earley algorithm is correct in the sense that, if there is a parse for the string, then such an item is certain to show up eventually, and, if no parse exists, no such item will show up. Finally, an algorithm exists for extracting a parse from the engendered parse lists.

The ID/LP version of the grammar works quite similarly. An item is of the form $[A, \alpha, \beta, i]$,[4] where $A$, $\alpha$ and $i$ are as before, $\beta \in \mathcal{P}(N \cup \Sigma)$ and, if $\alpha = \alpha_1 \ldots \alpha_n$, then $A \rightarrow_{ID} \{\alpha_1, \ldots, \alpha_n\} \cup \beta$ is an ID rule. Furthermore, $\alpha$ is LP-acceptable. The interpretation is almost as before: we have found the $\alpha$ part of an $A$ between the symbols $a_{i+1}$ and $a_j$, and we are looking for a parse of the symbols after $a_j$ as some *LP-acceptable permutation of* $\beta$.[5] As a consequence, the existence of an item of the form $[S, \alpha, \emptyset, 0]$ on parse list $I_n$ indicates recognition of the string.

We are now ready to present the details of the algorithm for constructing

the parse lists $I_j$.

Given a grammar $G = \langle N, \Sigma, ID, LP, S \rangle$ and an input string $w = a_1 a_2 \ldots a_n \in \Sigma^*$, we construct parse lists $I_0, \ldots, I_n$ as follows:

(1) For all $S \rightarrow_{ID} \beta$, add $[S, \Lambda, \beta, 0]$ to $I_0$.

Next we perform Steps 2 and 3 until no new items can be added to $I_0$.

(2) For all $[B, \gamma, \emptyset, 0] \in I_0$, and for all $[A, \alpha, \{B\} \cup \beta, 0] \in I_0$ such that $B \not> \beta$ and $B \notin \beta$, add $[A, \alpha B, \beta, 0]$ to $I_0$.

(3) For all $[A, \alpha, \{B\} \cup \beta, 0] \in I_0$, such that $B \not> \beta$ and $B \notin \beta$, for all $B \rightarrow_{ID} \gamma$, add the item $[B, \Lambda, \gamma, 0]$ to $I_0$.

We now construct $I_j$, having constructed $I_0, \ldots, I_{j-1}$.

(4) For each item $[A, \alpha, \{a\} \cup \beta, i] \in I_{j-1}$ such that $a = a_j$ and $a \not> \beta$ and $a \notin \beta$, add the item $[A, \alpha a, \beta, i]$ to $I_j$.

We perform Steps 5 and 6 until no new items can be added to $I_j$.

(5) For all $[B, \gamma, \emptyset, i] \in I_j$ and for all $[A, \alpha, \{B\} \cup \beta, k] \in I_i$ such that $B \not> \beta$ and $B \notin \beta$, add the item $[A, \alpha B, \beta, k]$ to $I_j$.

(6) For all $[A, \alpha, \{B\} \cup \beta, i] \in I_j$ such that $B \not> \beta$ and $B \notin \beta$, and for all $B \rightarrow_{ID} \gamma$, add the item $[B, \Lambda, \gamma, j]$ to $I_j$.

The string is accepted if and only if some item of the form $[S, \alpha, \emptyset, 0] \in I_n$.

We defer to Appendix I the proof (Theorem 7) that the parse lists constructed by the algorithm possess the properties outlined above. As an immediate corollary of Theorem 7, we have the following guarantee of the algorithm's correctness:

THEOREM 1. The algorithm accepts an input $w$ if and only if $S \Rightarrow^* w$.

In Appendix II, we present the algorithm for extracting a parse from the parse lists. The algorithm is completely analogous to the extraction algorithm for Earley parse lists (Aho and Ullman, 1972).

## 6. An example

The operation of the algorithm can be seen in a simple example. Consider the following grammar, an abstraction of the Makua grammar fragment presented previously.

$S \rightarrow_{ID} \{A, B, C\}$
$A \rightarrow_{ID} \{a\}$
$B \rightarrow_{ID} \{b\}$
$C \rightarrow_{ID} \{c\}$

$A < B$

Suppose we are given the string *acb* to parse. The algorithm would begin in Step 1 by adding the single item $[S, \Lambda, \{A, B, C\}, 0]$ to parse list $I_0$. Since $A \not> \{B, C\}$ and $C \not> \{A, C\}$, we add the items $[A, \Lambda, \{a\}, 0]$ and $[C, \Lambda, \{c\}, 0]$ to $I_0$ by Step 3. However, note that $B > \{A, C\}$ so that we do not add $[B, \Lambda, \{b\}, 0]$.

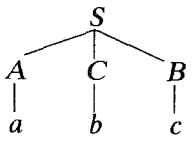
No more iterations of Steps 2 or 3 are possible, so we move on to step 4 to construct $I_1$. We add $[A, a, \emptyset, 0]$ to $I_1$ because the first element of the string is an $a$. By Step 5, we add $[S, A, \{B, C\}, 0]$ to $I_1$, thus recording the fact that we have found an $A$ and need only find a $BC$ or $CB$ to complete the $S$. This newly added item causes the items $[B, \Lambda, \{b\}, 1]$ and $[C, \Lambda\{c\}, 1]$ to be added to $I_1$ by Step 6.

Continuing in this manner, we construct parse lists $I_2$ and $I_3$. The complete set of parse lists is given below.

| List | Item | Step |
|---|---|---|
| $I_0$: | $[S, \Lambda, \{A, B, C\}, 0]$ | 1 |
| | $[A, \Lambda, \{a\}, 0]$ | 3 |
| | $[C, \Lambda, \{c\}, 0]$ | 3 |
| $I_1$: | $[A, a, \emptyset, 0]$ | 4 |
| | $[S, A, \{B, C\}, 0]$ | 5 |
| | $[B, \Lambda, \{b\}, 1]$ | 6 |
| | $[C, \Lambda, \{c\}, 1]$ | 6 |
| $I_2$: | $[C, c, \emptyset, 1]$ | 4 |
| | $[S, AC, \{B\}, 0]$ | 5 |
| | $[B, \Lambda, \{b\}, 2]$ | 6 |
| $I_3$: | $[B, b, \emptyset, 2]$ | 4 |
| | $[S, ACB, \emptyset, 0]$ | 5 |

accept.

Since the final parse list $I_3$ contains the item $[S, ACB, \emptyset, 0]$, we accept the input. Using the algorithm of Appendix II, we compute a rightmost derivation corresponding to the parse tree




If we look at the progression of $S$ items derived in the parse lists – $[S, \Lambda, \{A, B, C\}, 0]$, $[S, A, \{B, C\}, 0]$, $[S, AC, \{B\}, 0]$, and $[S, ACB, \emptyset, 0]$ – we can see clearly the character of the algorithm, extracting elements from the set as they are found, making sure at each stage that the resulting string is LP-acceptable. This basic character can be maintained with the multiset characterization of ID/LP (Klein, 1983). The algorithm remains unchanged, except that the symbol $\cup$ must be reinterpreted as multiset union (allowing repetitions) and the constraints that $B \notin \beta$ in Steps 2, 3, 5 and 6, and that $a \notin \beta$ in Step 4 must be removed.

Another topic discussed by Klein is the use of Kleene star on elements of the right hand sides in ID rules to mean that 0 or more of the elements may occur and be distributed freely with respect to one another. Clearly, such an extended ID/LP formalism could not be utilized as ID/LP has been in the past, i.e., by expanding it into a strongly equivalent context-free grammar. Fortunately, a very simple charge in the algorithm of Section 5 allows direct parsing of the formalism. This change is the same as the multiset change, except that the union operation does not necessarily add duplicates if the singleton set added contains a symbol that was starred in the original rule. In addition, the notion of $\emptyset$ must be extended to include sets with members that are so starred. A similar technique could be used, of course, to augment Earley's algorithm itself for parsing context-free grammars with Kleene star on the right-hand sides of rules.

## 7. The Time Complexity of the Algorithm

We will not present a rigorous demonstration of time complexity, but it should be clear from the close relation between the presented algorithm and Earley's that the complexity is that of Earley's algorithm. In the worst case, where the LP rules always specify a unique ordering for the right-hand side of every ID rule, the presented algorithm reduces to Earley's algorithm. Since, given a grammar, checking the LP rules takes constant time,[6] the time complexity of the present algorithm is identical to

Earley's (though the constant of proportionality might be quite different). That is, it is $O(|G|^2 n^3)$, where $|G|$ is the size of the grammar (number of ID rules) and $n$ is the length of the input.

It is important to realize that this time complexity measure cannot be used directly to compare the efficiency of Earley's algorithm with that of the ID/LP algorithm working on 'equivalent' grammars, simply because the $|G|$s in the two complexity measures are different. Making such formal comparisons would be analogous to preferring over a GPSG a grammar based on some radically different notation for context-free grammars merely because $|G|$ under this notation could be construed to be much smaller than $|G|$ for the typical GPSG grammar with its hundreds of rules. Whether or not such a *preference* would be valid in particular cases, such *reasoning* is clearly specious. Nonetheless, we will engage in just such considerations, rationalizing them on the basis of the quite close similarity of the ID/LP and Earley algorithms – a relationship that does not hold in general between algorithms for different formalisms. The reader is forewarned, however, that the conclusions here are informal.

With this caveat, we move on to discuss the critical question of the relative efficiency of direct parsing of an ID/LP grammar relative to the context-free parsing of the linearized form of the grammar, keeping in mind that we are comparing two algorithms that differ in the constant. Comparisons of grammar size clearly favor the direct-parsing algorithm. Since differences in grammar size can often swamp complexity due to input length (Graham *et al.*, 1980), this factor could be crucial in practice. Similarly, since the proof of Earley's algorithm's $O(n^3)$ behavior is based on the number of items added to the lists, the ID/LP algorithm is favored because several Earley items are encoded as a single ID/LP item. Whether in practice these differences compensate for any difference in constant depends critically on such ill-defined properties as the relative costs of primitive operations and, as discussed above, relative grammar size. The safe assumption is that there will be some grammars for which expansion is better, other for which direct parsing is better; however, as the grammars grow freer in their order and larger, so that the factors mentioned above increase in importance, the benefits of the direct-parsing algorithm in terms of parsing efficiency will become more pronounced. It nevertheless seems clear that, as far as practical computer implementation is concerned, the direct-parsing algorithm is advantageous.

## 8. The expressive power of ID/LP

Finally, a comment should be made regarding expressive power, and the

relationship of this characterization of ID/LP to that proposed by Gazdar and Pullum (1982). Gazdar and Pullum note that the ID/LP format is restrictive in the sense that not every context-free grammar can be expressed as an ID/LP grammar. Specifically, they claim that ID/LP grammars exist only for context-free grammars with the Exhaustive Constant Partial Ordering (ECPO) property, according to which "the set of expansions for any given category is closed under a partial ordering that is constant for the expansion of all categories" (Gazdar and Pullum, 1982, p. 21). This is, of course, true of the characterization of ID/LP presented in Section 4.

However, the ECPO property is a property of grammars, not languages. And though the ECPO property is a restriction on CF grammars, ID/LP encodability is not a restriction on CF languages, for it can be shown straightforwardly that any context-free language can be generated by an ID/LP grammar and that, conversely, every ID/LP grammar has a strongly equivalent context-free phrase structure grammar. We prove these two statements below.

THEOREM 2. Every context-free grammar $G$ is weakly equivalent to an ID/LP grammar.[7]

*Proof.*[8] We construct an ID/LP grammar from $G$ as follows. First, we convert $G$ to a Chomsky-normal-form grammar $G'$. This transformation preserves weak-generative capacity and guarantees that all nonunary rules are of the form $A \rightarrow BC$, where $B$ and $C$ are nonterminals and all unary rules are of the form $A \rightarrow a$, where $a$ is a terminal. Second, we construct an ID/LP grammar $G''$ from $G'$. For each unary rule $A \rightarrow a$ of $G'$ we add $A \rightarrow_{ID} \{a\}$ to $G''$. Let $<$ be a total ordering on the nonterminal vocabulary of $G'$. For each rule in $G'$ of the form $A \rightarrow BC$, if $B < C$, then add $A \rightarrow_{ID} \{B, C\}$ to $G''$. If $B \not< C$, then add the rules $A \rightarrow_{ID} \{B, C'\}$ and $C' \rightarrow_{ID} \{C\}$ to $G''$ and $B < C'$ to the $<$ ordering.[9] $G''$ (along with the $<$ ordering) is an ID/LP grammar with the same weak-generative capacity as $G$. □

Theorem 2 leads us to the following trivial corollary.

COROLLARY 3. Every context-free language can be encoded in an ID/LP grammar.

THEOREM 4. Every ID/LP grammar $G$ is strongly equivalent to a context-free grammar.

*Proof.* We construct a context-free grammar $G'$ from $G$ as follows. For

every rule of $G$, $A \rightarrow_{ID} \beta$, we add $A \rightarrow \beta'$ to $G'$ for each $\beta' \in expand(\beta)$. $G'$ is a context-free grammar with the same strong-generative capacity as $G$. This is, of course, the construction that previous work on ID/LP has presupposed as a definition of the formalism. □

Thus, we see that the ECPO property and ID/LP format impose no *formal* limitations upon the context-free languages that can be encoded; one can always find an ECPO grammar for any context-free language. Gazdar and Pullum note that the importance of the ECPO property is that it may be a universal property of *linguistically motivated* grammars. This is a strong linguistic claim – though it should be clear that it is not a formal claim but rather an aesthetic one, subject to the frailties of linguistic judgment.

In any case, it is interesting to note that the extended ID/LP formalism used by Gazdar and Pullum (1982) even allows certain non-ECPO context-free grammars to be encoded, contrary to the statement of Gazdar and Pullum that "a CF-PSG can be put into ID/LP format if and only if it has the ECPO property" (Gazdar and Pullum, 1982, p. 21). Their ID/LP formalism is a slight extension of the one presented in Section 4. Besides allowing multisets on the right-hand side of rules (which does not change the structural generative capacity of the formalism), they permit the use in ID and LP rules of the special symbol $H$ to denote the head of the rule in the sense of $\bar{X}$ syntax (Jackendoff, 1977). Since the syntactic category represented by $H$ changes from one ID rule to another, certain non-ECPO grammars can be encoded with this extended formalism. For instance, the following grammar is non-ECPO:

$$A \rightarrow B\,C$$
$$E \rightarrow B\,C\,D$$
$$E \rightarrow C\,B\,D$$
$$E \rightarrow B\,D\,C$$

but an isomorphic grammar can be expressed in the Gazdar and Pullum ID/LP formalism (through the renaming of $A$ as $\bar{C}$ and $E$ as $\bar{D}$) with the ID/LP grammar:[10]

$$\bar{C} \rightarrow_{ID} \{B, H\}$$
$$\bar{D} \rightarrow_{ID} \{B, C, H\}$$

$$B < H.$$

Here, then, is a counterexample to the above-cited claim of Gazdar and Pullum. An alternative interpretation of the $H$ symbol as an abbreviation for a feature, say, *ishead*,[11] would remove the anomaly of non-ECPO

ID/LP grammars, since the category $C[+ishead]$ is different from $C[-ishead]$. The concept 'head' would then be defined as follows: "In a rule of the form $\beta_0 \rightarrow \beta_1 \ldots \beta_n$, where $\beta_i$ has the feature $[+ishead]$, $\beta_i$ is the head of $\beta_0$." The feature *ishead* would, ironically, not be a head feature, since it must not be required of the parent node. This presents no problem for the current theory of GPSG because several features (e.g., *foot*) are not required to effect agreement between a parent and its head.

Although, as we have just seen, the expressive power (in terms of strong-generative capacity) of Gazdar and Pullum's ID/LP formalism is greater than that presented here, the algorithm described could be augmented to handle the extension merely by changing the definitions of *ID* and *LP* in the obvious way. The multiset characterization and the use of Kleene star were discussed previously in Section 6.

## 9. Conclusion

One might be led to extrapolate from the existence of the presented algorithm to the conjecture that direct parsing algorithms would exist for other extended grammar formalisms. Indeed, the preponderance of parsers for annotated phrase-structure grammars that parse directly, e.g. (Bear and Karttunen, 1979; Gawron *et al.*, 1982; Rosenschein and Shieber, 1982) (as distinguished from converting the grammars to an equivalent context-free form) could be seen, in retrospect, to be further evidence. Such an extrapolation is quite dangerous, however. On the one hand, ID/LP is a limited formalism that is actually *less* expressive (in strong-generative capacity) than context-free grammars, while certain metarule formalisms have been shown to be much more powerful than context-free grammars (Uszkoreit and Peters, 1983). On the other hand, certain classes of non-context-free grammars have very efficient parsing algorithms – better, in fact, than context-free grammars in general. Thus, any conjectures about direct-parsing algorithms will have to be examined on a case-by-case basis. It is in this spirit that we are now pursuing research on the direct parsing of metarule grammars and on other grammar formalisms. The main thrust of such research, however, must necessarily be directed toward perspicuous *limited* formalisms that can express the types of generalizations linguists want to make – and that do so without inordinately expanding expressive power.

## Appendix I. A correctness proof

We now prove that the algorithm presented in Section 5 is correct, that is,

the algorithm accepts a string just in case the grammar derives it.[12]

LEMMA 5. For all $a \in N \cup \Sigma$, $\beta \in \mathcal{P}(N \cup \Sigma)$, and $\beta' \in \mathcal{P}((N \cup \Sigma)^*)$ such that $a \notin \beta$ and $\beta' \in expand(\beta)$, the following biconditional holds:

$$a \not> \beta \leftrightarrow a\beta' \in expand(\{a\} \cup \beta)$$

*Proof.*

**If direction:** $a\beta' \in expand(\{a\} \cup \beta)$, so $a\beta'$ is LP-acceptable and $a \not> \beta'$. Since $\beta'$ and $\beta$ have the same elements, $a \not> \beta$.

**Only-if direction:** $a \not> \beta$ and $\beta'$ is LP-acceptable, so $a\beta'$ is also LP-acceptable. Also, since $a \notin \beta$, $a\beta' \in permute(\{a\} \cup \beta)$. Therefore, $a\beta' \in expand(\{a\} \cup \beta)$. □

COROLLARY 6. If $A \rightarrow \alpha\gamma$ for all $\gamma \in expand(\{a\} \cup \beta)$ such that $a \notin \beta$ and $a \not> \beta$, then $A \rightarrow \alpha a\beta'$ for all $\beta' \in expand(\beta)$.

THEOREM 7. If parse lists are constructed as specified, then the item $[A, \alpha, \beta, i]$ is on $I_j$ iff

**claim i:** $\alpha \Rightarrow^* a_{i+1} \ldots a_j$, and

**claim ii:** there exist $\gamma$ and $\delta$ such that $S \Rightarrow^* \gamma A \delta$ and $\gamma \Rightarrow^* a_1 \ldots a_i$, and

**claim iii:** $A \rightarrow \alpha\beta'$ for all $\beta' \in expand(\beta)$.

*Proof.*

**Only-if direction:** We must show that, if $[A, \alpha, \beta, i]$ is added to $I_j$, then the claims hold. We prove the three claims by induction on the number of items that have been added to $I_0, \ldots, I_i$ before $[A, \alpha, \beta, i]$ is added to $I_j$. In the following analysis, each claim is handled in a separate numbered paragraph and *i.h.* refers to the induction hypotheses.

**Case 1: The Base Case**

The item $[A, \alpha, \beta, i]$ is added to $I_j$ before any other items have been added. This can arise only from Rule 1. Thus, $i = j = 0$, $A = S$, and $\alpha = \Lambda$.

(i) $\alpha \Rightarrow^* \Lambda$

(ii) $S \Rightarrow^* \gamma A \delta$, where $\gamma = \delta = \Lambda$

(iii) $S \rightarrow \beta'$ for all $\beta' \in expand(\beta)$, since $S \rightarrow_{ID} \beta$.

The claims therefore hold for the base case.

**The inductive step** comprises cases 2 through 6, depending on the rule used to introduce the item to the list.

**Case 2: Introduction by Rule 2**

The item $[A, \alpha, \beta, i]$ was added to $I_j$ by Rule 2. Thus, $i = j = 0$ and

there exist $\alpha'$ and $B$ such that $\alpha = \alpha' B$, $[B, \gamma, \emptyset, 0] \in I_0$, $[A, \alpha', \{B\} \cup \beta, 0] \in I_0$, and $B \not> \beta$.

(i) By i.h., $a' \Rightarrow^* \Lambda$, $\gamma \Rightarrow^* \Lambda$, and $B \rightarrow \gamma$. Thus, $\alpha = \alpha' B \Rightarrow \alpha' \gamma \Rightarrow^* \Lambda$.

(ii) By i.h., $S \Rightarrow^* \gamma' A \delta'$, where $\gamma' \Rightarrow^* \Lambda$.

(iii) Again by i.h., $A \rightarrow \alpha' \beta'$, where $\beta' \in expand(\{B\} \cup \beta)$, so that, since $B \not> \beta$, $A \rightarrow \alpha' B \beta'' = \alpha \beta''$, where $\beta'' \in expand(\beta)$ by Corollary 6.

**Case 3: Induction by Rule 3**

The item $[A, \Lambda, \beta, 0]$ was added to $I_0$ by Rule 3, so there exist $B$, $\gamma$, and $\delta$ such that $[B, \gamma, \{A\} \cup \delta, 0]$ is on $I_0$ and $A \not> \delta$ and $A \rightarrow_{ID} \beta$.

(i) $\alpha \Rightarrow^* a_{i+1} \ldots a_j$ follows trivially, since $i = j = 0$ and $\alpha = \Lambda$.

(ii) By i.h., $S \Rightarrow^* \gamma' B \delta'$, $\gamma' \Rightarrow^* \Lambda$, $\gamma \Rightarrow^* \Lambda$, and $B \rightarrow \beta'$ for all $\beta' \in expand(\{A\} \cup \delta)$. So $B \rightarrow A\delta''$ for all $\delta'' \in expand(\delta)$, since $A \not> \delta$ and by Corollary 6. Then $S \Rightarrow^* B\delta' \Rightarrow A\delta''\delta'$.

(iii) follows directly from $A \rightarrow_{ID} \beta$.

**Case 4: Introduction by Rule 4**

The item $[A, \alpha, \beta, i]$ was added to item $I_j$ by Rule 4. Thus, there exist $\alpha'$ and $a$ such that $\alpha = \alpha' a$, $[A, \alpha', \{a\} \cup \beta, i] \in I_{j-1}$, $a = a_j$, and $a \not> \beta$.

(i) By i.h., $\alpha' \Rightarrow^* a_{i+1} \ldots a_{j-1}$ so $\alpha = \alpha' a \Rightarrow a_{i+1} \ldots a_j$.

(ii) By i.h., $S \Rightarrow^* \gamma A \delta$, where $\delta \Rightarrow^* a_1 \ldots a_i$.

(iii) $A \rightarrow \alpha' \beta'$, where $\beta' \in expand(\{a\} \cup \beta)$, by i.h. So, $A \Rightarrow \alpha' a \beta''$, where $\beta'' \in expand(\beta)$, since $\alpha \not> \beta$ and by Corollary 6. Thus, $A \rightarrow \alpha\beta''$.

**Case 5: Introduction by Rule 5**

The item $[A, \alpha, \beta, i]$ was added to $I_j$ by Rule 5. Thus, there exist $\alpha'$, $\delta$, $k$, and $B$ such that $\alpha = \alpha' B$, $[B, \delta, \emptyset, k]$ is on $I_j$, $[A, \alpha', \{B\} \cup \beta, i]$ is on $I_k$, and $B \not> \beta$.

(i) By i.h., $\alpha' \Rightarrow^* a_{i+1} \ldots a_k$, $B \rightarrow \delta$, and $\delta \Rightarrow^* a_{k+1} \ldots a_j$. Thus, $\alpha = \alpha' B \Rightarrow \alpha' \delta \Rightarrow^* a_{i+1} \ldots a_k a_{k+1} \ldots a_j = a_{i+1} \ldots a_j$.

(ii) By i.h., $S \Rightarrow^* \gamma A \delta$, where $\delta \Rightarrow^* a_1 \ldots a_i$.

(iii) By i.h., $A \rightarrow \alpha' \beta''$, where $\beta'' \in expand(\{B\} \cup \beta)$. Since $B \not> \beta$, $A \Rightarrow \alpha' B \beta'$, where $\beta' \in expand(\beta)$ by Corollary 6.

**Case 6: Introduction by Rule 6**

The item $[A, \alpha, \beta, i]$ was added to $I_j$ by Rule 6. Thus, $i = j$, $\alpha = \Lambda$, and there exist $B$, $\gamma$, $\delta$, and $k$ such that $[B, \gamma, \{A\} \cup \delta, k]$ is on $I_j$, $A \neq \delta$, and $A \rightarrow_{ID} \beta$.

(i) $\alpha \Rightarrow^* \Lambda$ follows trivially, since $\alpha = \Lambda$.

(ii) By i.h., $S \Rightarrow^* \gamma' B \delta'$, where $\gamma' \Rightarrow^* a_1 \ldots a_k$. Also by i.h., $B \rightarrow \gamma\zeta$, where $\zeta \in expand(\{A\} \cup \delta)$. So, since $A \neq \delta$, $B \rightarrow \gamma A \zeta'$, where $\zeta' \in expand(\delta)$. Thus, $S \Rightarrow^* \gamma' \gamma A \zeta' \delta'$. Now, $\gamma \Rightarrow^* a_{k+1} \ldots a_j$ by i.h. So, $S \Rightarrow^* a_1 \ldots a_j A \zeta' \delta'$.

(iii) $A \Rightarrow^* \beta'$ for all $\beta' \in expand(\beta)$ follows directly from $A \rightarrow_{ID} \beta$.

Thus, the only-if direction is proved for all six cases.

**If direction:** We must show that, if the three claims hold, $[A, \alpha, \beta, i]$ is added to list $I_j$ by the algorithm. The proof is by induction on the rank $\rho$ of an instance $\mathcal{I}$ of the claims, where an instance is the septuple $[\alpha, \beta, \gamma, \delta, A, i, j]$ of elements necessary and sufficient to form a particular instance of the claims, and $\rho(\mathcal{I}) \equiv \mathcal{T}_1(\mathcal{I}) + 2[j + \mathcal{T}_2(\mathcal{I}) + \mathcal{T}_3(\mathcal{I})]$. Here $\mathcal{T}_1(\mathcal{I})$ is the length of a shortest derivation $S \Rightarrow^* \gamma A \delta$, $\mathcal{T}_2(\mathcal{I})$ is the length of a shortest derivation $\gamma \Rightarrow^* a_1 \ldots a_i$, and $\mathcal{T}_3(\mathcal{I})$ is the length of a shortest derivation $\alpha \Rightarrow^* a_{i+1} \ldots a_j$.

**The base case** – If $\rho(\mathcal{I}) = 0$, then $\mathcal{T}_1(\mathcal{I}) = \mathcal{T}_2(\mathcal{I}) = \mathcal{T}_3(\mathcal{I}) = j = 0$. Thus, $\alpha = \gamma = \delta = \Lambda$ and $A = S$. We must therefore show that $[S, \Lambda, \beta, 0]$ is on list $I_0$. This follows immediately from Rule 1, since we are given $S \rightarrow \beta'$ for all $\beta' \in expand(\beta)$, so it must be that $S \rightarrow_{ID} \beta$. But Rule 1 would then add the appropriate item to $I_0$.

**The induction step** – Let $\mathcal{I}$ be an instance such that $\rho(\mathcal{I}) > 0$ and let us assume that the induction hypothesis holds for all instances of lesser rank. Three cases now arise, depending on whether $\alpha$ ends in a terminal or a nonterminal or is $\Lambda$.

**Case 1:** $\alpha = \alpha' a$ **for some** $a \in \Sigma$

Since $\alpha \Rightarrow^* a_{i+1} \ldots a_j$, we conclude that $a = a_j$. Consider the instance $\mathcal{I}' = [\alpha', \{a_j\} \cup \beta, \gamma, \delta, A, i, j-1]$. Now, $\alpha \neq \beta$ and $a_j \in \alpha$ so $a_j \neq \beta$. Since $A \rightarrow \alpha' a_j \beta$, $\mathcal{I}'$ is an instance of the claims and is at most of rank $\mathcal{T}_1(\mathcal{I}) + 2[(j-1) + \mathcal{T}_2(\mathcal{I}) + \mathcal{T}_3(\mathcal{I})] = \rho(\mathcal{I}) - 2$. Thus, by i.h., $[A, \alpha', \{a_j\} \cup \beta, i]$ is on $I_{j-1}$ and, by Rule 4, $[A, \alpha, \beta, i]$ is on $I_j$.

**Case 2:** $\alpha = \alpha' B$ **for some** $B \in N$

There exists a $k$, $i \leq k \leq j$, such that $\alpha' \Rightarrow^* a_{i+1} \ldots a_k$, and $B \Rightarrow^* a_{k+1} \ldots a_j$. Furthermore, we can conclude from the instance $\mathcal{I}$ that

$A \rightarrow \alpha' B\beta$. Consider the instance $\mathscr{I}' = [\alpha', B\beta, \gamma, \delta, A, i, k]$. Its rank is $\mathscr{T}_1(\mathscr{I}) + 2[k + \mathscr{T}_2(\mathscr{I}) + \mathscr{T}_3(\mathscr{I}')]$ and, since $k < j$ and $\mathscr{T}_3(\mathscr{I}') < \mathscr{T}_3(\mathscr{I})$, it is of lower rank. Therefore, we can conclude that $[A, \alpha', \{B\} \cup \beta, i]$ is on $I_k$. Now, let $B \Rightarrow \zeta$ be the first step in a minimum-length derivation $B \Rightarrow^* a_{k+1} \ldots a_j$ and consider the instance $\mathscr{I}' = [\zeta, \emptyset, \gamma\alpha', B\delta, B, k, j]$. Since $S \Rightarrow^* \gamma A\delta \Rightarrow \gamma\alpha' B\beta\delta$, we conclude that $\mathscr{T}_1(\mathscr{I}') \leq \mathscr{T}_1(\mathscr{I}) + 1$. Let $n_1$ be the minimum number of steps in a derivation $\alpha' \Rightarrow^* a_{i+1} \ldots a_k$ and $n_2$ be the minimum number in a derivation $B \Rightarrow^* a_{k+1} \ldots a_j$. Then $\mathscr{T}_3(\mathscr{I}) = n_1 + n_2$. Since $B \Rightarrow \zeta \Rightarrow^* a_{k+1} \ldots a_j$, we conclude that $\mathscr{T}_3(\mathscr{I}') = n_2 - 1$. Now, $\mathscr{T}_2(\mathscr{I}') = \mathscr{T}_2(\mathscr{I}) + n_1$. So, $\mathscr{T}_2(\mathscr{I}') + \mathscr{T}_3(\mathscr{I}') = \mathscr{T}_2(\mathscr{I}) + n_1 + n_2 - 1 = \mathscr{T}_2(\mathscr{I}) + \mathscr{T}_3(\mathscr{I}) - 1$ and $\rho(\mathscr{I}') < \rho(\mathscr{I})$. By i.h., we conclude that $[B, \zeta, \emptyset, k]$ is on $I_j$. Thus, by Rule 2 or 5, we see that $[A, \alpha, \beta, i]$ is on list $I_j$.

**Case 3:** $\alpha = \Lambda$

We conclude that $i = j$ and $\mathscr{T}_3(I) = 0$. Since $\rho(\mathscr{I}) > 0$, we conclude that $\mathscr{T}_1(\mathscr{I}) > 0$ because, if $\mathscr{T}_1(\mathscr{I}) = 0$, then $S = \gamma A\delta$ and $\gamma = \Lambda$; $\mathscr{T}_2(\mathscr{I}) = \mathscr{T}_3(\mathscr{I}) = 0$ and $i = j = 0$ and $\alpha = \Lambda$, but this is just the base case. Thus, there exist $B \in N$ and $\gamma'$, $\gamma''$, $\delta'$, $\delta''$ such that $S \Rightarrow^* \gamma' B\delta' \Rightarrow \gamma'\gamma'' A\delta''\delta'$, where $B \rightarrow \gamma'' A\delta''$, $\gamma = \gamma'\gamma''$, $\delta = \delta''\delta'$, and $\gamma' B\delta'$ is the penultimate step of a shortest derivation $S \Rightarrow^* \gamma A\delta$. Consider the instance $\mathscr{I}' = [\gamma'', A\delta'', \gamma', \delta', B, k, j]$, where $k$ is an integer such that $\gamma' \Rightarrow^* a_1 \ldots a_k$ and $\gamma'' \Rightarrow^* a_{k+1} \ldots a_j$. Let the lengths of minimum-length derivations of these be $n_1$ and $n_2$, respectively. Then $\mathscr{T}_2(\mathscr{I}') = n_1$, $\mathscr{T}_3(\mathscr{I}') = n_2$, and $\mathscr{T}_2(\mathscr{I}) = n_1 + n_2$. We also note that $\mathscr{T}_1(\mathscr{I}') = \mathscr{T}_1(\mathscr{I}) - 1$. Thus, $\rho(\mathscr{I}') = \rho(\mathscr{I}) - 1$, and, by i.h., $[B, \gamma'', \{A\} \cup \delta'', k]$ is on list $I_j$. Since $B \rightarrow \gamma'' A\delta''$, we conclude that $A \not> \delta''$ and Rule 6 would add $[A, \emptyset, \beta, j]$ to list $I_j$. □

The following trivial corollary of Theorem 7 guarantees the correctness of the algorithm.

THEOREM 1. The algorithm accepts an input $w$ iff $S \Rightarrow^* w$.

## APPENDIX II. A PARSE EXTRACTION ALGORITHM

The following algorithm extracts a rightmost derivation from the parse lists generated by the ID/LP algorithm. The algorithm is presented merely for completeness, as its operation is virtually identical to that of Aho and Ullman (1972). It assumes that the elements of ID are numbered, and builds a list of production numbers corresponding to the production used in a rightmost derivation of the input sentence. This list of productions

contains the same information as a parse tree. The list is built in the global variable $\pi$, which is initially set to $\Lambda$. It is called with $R([S, \alpha, \emptyset, 0], n)$, where $[S, \alpha, \emptyset, 0]$ is on $I_n$.

$R([A, \beta_1 \ldots \beta_m, \emptyset, i], j)$:
  $h :=$ number of production $A \rightarrow_{ID} \{\beta_1, \ldots, \beta_m\}$;
  $\pi := h\pi$;
  $k := m$; $l := j$;
  repeat
    if $\beta_k \in \Sigma$
      then $k := k - 1$; $l := l_1$
    else
      find an item $[\beta_k, \gamma, \emptyset, r]$ on $I_l$ for some $r$ such that
        $[A, \beta_1 \ldots \beta_k - 1, \{\beta_k, \ldots, \beta_m\}, i]$ is in $I_r$;
      execute $R([\beta_k, \gamma, \emptyset, r], l)$;
      $k := k - 1$; $l := r$
  until $k = 0$.

## NOTES

* This research was supported by National Science Foundation Grant No. IST-8103550. The views and conclusions contained in this document are those of the author and should not be interpreted as representative of the official policies, either expressed or implied, of the National Science Foundation or the United States government. I am indebted to Chris Culy for his meticulous checking of the proofs and for the characterization of the proof of Theorem 2. I am also grateful to Stanley Peters, Geoff Pullum, Susan Stucky, Hans Uszkoreit, and two anonymous referees for their comments on earlier drafts.

[1] The data are from Stucky (1981). The analysis presented here is for expository purposes and is, of course, a grossly simplified distillation of the extensive analysis presented by Stucky.

[2] Another undesirable corollary of the indirect-interpretation method of ID/LP grammar use is that, viewed as a mere abbreviation for a context-free grammar, an ID/LP grammar for a free-word-order language would induce a much larger grammar than one for a fixed-word-order language. The problem is compounded in that the process of linearizing ID rules is exponential in rule length, and free-word-order languages tend to have longer rules (flatter structure) than fixed-word-order languages. From this would follow the unintuitive prediction that a processor for a free-word-order language would be manipulating a grammar orders of magnitude larger than a processor for a fixed-word-order language. Although based on a naive view of psychological reality, these arguments do seem to cast doubt on the view that humans would use in any sense a grammar encoding indirectly by expansion.

[3] The implementations of the direct ID/LP parsing algorithm are due to Roger Evans at the University of Sussex and Mark David at the University of California at Los Angeles, in Prolog and Franz Lisp respectively.

[4] The notation used is modeled on that of Aho and Ullman (1972), except that we denote an item by $[A, \alpha, \beta, i]$ rather than $[A \rightarrow \alpha \,.\, \beta, i]$ to highlight the fact that in the ID/LP version of the algorithm, since $\alpha$ and $\beta$ are a string and a set, respectively, they could not both be on the right-hand side of either a *yields* or a *dominates* arrow.

[5] The formal characterization of this statement, which is analogous to that given above for Earley's algorithm is found in the statement of Theorem 7 in Appendix I.

[6] "Any process concerning [the symbols of $G$] can be considered elementary." (Aho and Ullman, 1972, p. 326) Thus, given the grammar, the LP check (deciding whether a given symbol precedes a given set) can be done in constant time. This can be seen quite clearly by noting that, at worst, it could be performed as a table lookup, since there are only finite numbers of symbols and sets of symbols. Note that this is so even for the multiset case, since the check need be done only on the multiset with duplicates removed, and this can be precomputed.

[7] In fact, a stronger theorem can be proved: every context-free grammar is *structurally equivalent* to an ID/LP grammar, that is, it is strongly equivalent up to a renaming of nonterminal labels – so that the skeletal tree structure and the frontier are identical though the labelings may be different. Stronger still is the fact that the relabeling is such that there exists a homomorphism from the ID/LP tree labeling to the CF tree labeling. The proof of this stronger theorem is not supplied.

[8] This theorem is also a direct consequence of the existence of two CFG normal forms that are weakly equivalent to context-free grammars. Chomsky (1963) discusses *modified normal grammars*, defined as Chomsky-normal-form grammars in which a nonterminal may not occur as both a left nonterminal on the right-hand side of a rule and a right nonterminal of a (different or same) rule. The ability to define a partial order on the nonterminals of the grammar is obvious. Stanley (1965), in a short article providing a correction to some formal claims of Chomsky (1963), discusses an even stronger normal form, *α-normalized grammars*. Both articles intimate the existence of, but do not furnish, proofs of the weak equivalence of the normal forms to context-free grammars. I am indebted to Stanley Peters for bringing these normal forms to my attention.

[9] This also covers the case in which $B = C$.

[10] Even the requirement that a lexical category $X$ be introduced only under the category $\bar{X}$ does not affect the existence of such non-ECPO grammars, as can be easily shown. Stricter variants of $\bar{X}$ theory may disallow these pathological cases, though the literature has not provided any such constraints.

[11] Note that the *head* feature discussed by Gazdar and Pullum (1982) does *not* do the work of *ishead*. *Ishead* is a binary feature that is true of only one filial constituent. All the children can possess the feature *head*.

[12] The proof is modeled directly on the proof of correctness of Earley's algorithm given by Aho and Ullman (1972).

## REFERENCES

Aho, A. V. and J. D. Ullman: 1972, *Theory of Parsing, Translation and Compiling*, Vol. I (Prentice-Hall, Englewood Cliffs, New Jersey).

Bear, J. S. and L. Karttunen: 1982, 'PSG: A Simple Phrase Structure Parser', *Texas Linguistic Forum* **14**.

Chomsky, N.: 1963, 'Formal Properties of Grammars', in R. D. Luce, R. R. Bush, and E. Galanter (eds.), *Handbook of Mathematical Psychology*, Vol. II (John Wiley and Sons, New York), pp. 323–418.

Culy, C.: 1983, 'An Extension of Phrase Structure Rules and Its Application to Natural Language', unpublished M.A. dissertation (Stanford University, Stanford, California (May)).

Earley, J.: 1970, 'An Efficient Context-Free Parsing Algorithm', *Communications of the ACM* **13**, 94–102.

Gawron, J. M. *et al.*: 1982, 'The GPSG Linguistics System', in *Proceedings of the 20th*

*Annual Meeting of the Association for Computational Linguistics* (University of Toronto, Toronto, Ontario, Canada (16–18 June)).

Gazdar, G. and G. K. Pullum: 1982, 'Generalized Phrase Structure Grammar: A Theoretical Synopsis' (Indiana University Linguistics Club, Bloomington, Indiana (August)).

Gazdar, G. and G. K. Pullum: 1981, 'Subcategorization, Constituent Order, and the Notion "head"', in M. Moortgat, H. v. d. Hulst, and T. Hoekstra (eds.), *The Scope of Lexical Rules* (Foris, Dordrecht, Holland).

Graham, S. L., M. A. Harrison, and W. L. Ruzzo: 1980, 'An Improved Context-Free Recognizer', *ACM Transactions on Programming Languages and Systems* **2**, 415–462.

Jackendoff, R.: 1977, *$\bar{X}$ Syntax: A Study of Phrase Structure*, Linguistic Inquiry Monograph Two (MIT Press, Cambridge, Mass.).

Klein, E.: 1983, 'A Multiset Analysis of Immediate Domination Rules', unpublished manuscript.

Pullum, G. K.: 1982, 'Free Word Order and Phrase Structure Rules', in J. Pustejovsky and P. Sells (eds.), *Proceedings of the Twelfth Annual Meeting of the North Eastern Linguistic Society*, Graduate Linguistics Student Association (University of Massachusetts, Amherst, Mass.).

Rosenschein, S. and S. M. Shieber: 1982, 'Translating English into Logical Form', in *Proceedings of the 20th Annual Meeting of the Association for Computational Linguistics* (University of Toronto, Toronto, Ontario, Canada (16–18 June)).

Stanley, R. J.: 1965, 'Finite State Representations of Context-Free Languages', in *Quarterly Progress Report No. 76* (Research Laboratory of Electronics, MIT, Cambridge, Mass. (15 January)), pp. 276–279.

Stucky, S.: 1981, 'Word Order Variation in Makua: A Phrase Structure Grammar Analysis', Ph.D. Dissertation (University of Illinois at Urbana-Champaign).

Stucky, S. U.: 1983, 'Metarules As Meta-Node-Admissibility Conditions', SRI Technical Note 304 (SRI International, Menlo Park, Calif. (September)).

Thompson, H.: 1982, 'Handling Metarules in a Parser for GPSG', Research Paper Number 175 (Department of Artificial Intelligence and Cognitive Sciences Program, School of Epistemics, University of Edinburgh, Edinburgh, Scotland).

Uszkoreit, H.: 1982, 'German Word Order in GPSG', in D. Flickinger, M. Macken, and N. Wiegand (eds.), *Proc. of the First West Coast Conference on Formal Linguistics* (Stanford University, Stanford, Calif.).

Uszkoreit, H. and S. Peters: 1983, 'On Some Formal Properties of Metarules', SRI Technical Note 305 (SRI International, Menlo Park, California (September)).

*Artificial Intelligence Center, SRI International,*
333 *Ravenswood Avenue, Menlo Park, CA* 94025, *U.S.A.*