REVISED GENERALIZED PHRASE STRUCTURE GRAMMAR

Eric Sven Ristad[1]

M.I.T. Artificial Intelligence Lab
545 Technology Square, 805
Cambridge, MA 02139

Thinking Machines Corporation
245 First Street
Cambridge, MA 02142

ABSTRACT

In this paper, I revise generalized phrase structure grammar (GPSG) linguistic theory so that it is more tractable and linguistically constrained. Revised GPSG is also easier to understand, use, and implement. I provide an account of topicalization, explicative pronouns, and parasitic gaps in the revised system and conclude with suggestions for efficient parser design.

1 Introduction and Motivation

A linguistic theory specifies a computational process that assigns structural descriptions to utterances. This process requires certain computational resources, such as time or space. In a descriptively adequate linguistic theory, the computational resources available to the theory match those used by the ideal speaker-hearer. The goal of this paper is to revise generalized phrase structure grammar (GPSG) so that its computational power corresponds to the ability of the speaker–hearer.

The bulk of this paper is devoted to identifying what computational resources are used by GPSG theory, and deciding whether they are linguistically necessary. GPSG contains five formal devices, each of which provides the theory with the resources to model some linguistic phenomenon or ability. I identify those aspects of each device that cause intractability and then restrict the computational power of each device to more closely match the (inherent) complexity of the phenomenon or ability it models. The remainder of the paper presents the new formal system and exercises it in the domain of topicalization, explicative pronouns, and parasitic gaps. I conclude with suggestions for efficient parser design and future research.

In my opinion, the primary value of this work lies in the result (*revised* GPSG, or RGPSG) as well as in the methodology of using complexity analysis to improve linguistic theories. The methodology explicates how a tool of modern computer science can help us understand and improve theories of linguistic competence. More than that, complexity analysis forms the foundation of informed parser design. I feel RGPSG is of value both to linguists and computational linguists because it is more tractable and easier to understand, use, and implement. It can be efficiently implemented and appears to have better empirical coverage than its GPSG ancestor.

[1]The author is supported by a graduate fellowship from the IBM Corporation. This research was supported in part by Thinking Machines Corporation and by NSF Grant DCR-85552543, under a Presidential Young Investigator Award to Professor Robert C. Berwick. I wish to thank Ed Barton for stylistic improvements and helpful discussion; Robert Berwick for support, criticism, and suggesting I pursue this research; and Geoff Pullum for his patient help with GPSG theory.

2 Eliminating Intractability in GPSG

Ristad (1986a) examines the computational complexity of two components of the GPSG formal system (metarules and the feature system) and shows how each of these systems can lead to computational intractability. Ristad also proves that the universal recognition problem for GPSGs is EXP-POLY hard, and intractable.[2] In another words, the fastest recognition algorithm for GPSGs can take more than exponential time.

These results may appear surprising, given GPSG's weak context-free generative power. They also raise some important computational and linguistic questions: why GPSG-Recognition is so difficult, what aspects of the GPSG formalisms cause intractability, and whether they are linguistically necessary. I begin with an outline of the GPSG formal system, as presented in Gazdar, Klein, Pullum, and Sag (1985), GKPS hereafter. Subsequently, I identify and remove the excess computational power provided by each formal device.

2.1 Overview of GPSG Formalisms

From the perspective of classic formal language theory, a GPSG may be thought of as a grammar for generating a context-free grammar. The generation process begins with immediate dominance (ID) rules, which are context-free productions with unordered right-hand sides. An important feature of ID rules is that nonterminals in the rules are not atomic symbols (for example, NP). Rather, GPSG nonterminals are sets of [*feature, feature-value*] pairs. For example, [N +] is a [*feature, feature-value*] pair, and the set {[N +], [V -], [BAR 2]} is the GPSG representation of a noun phrase. Next, metarules apply to the ID rules, resulting in an enlarged set of ID rules. Metarules have fixed input and output patterns containing a distinguished multiset variable W in addition to constants. If an ID rule matches the input pattern under some specialization of the variable W, then the metarule generates an ID rule corresponding to the metarule's output pattern under the same specialization of W. For example, the passive metarule

$$\begin{array}{c} VP \rightarrow W, NP \\ \Downarrow \\ VP[\texttt{PAS}] \rightarrow W, (PP[\texttt{by}]) \end{array} \qquad (1)$$

says that "for every ID rule in the grammar which permits a VP to dominate an NP and some other material, there is also a rule

[2]The universal recognition problem most accurately reflects the difficulty of processing a grammatical formalism because it incorporates the grammar in the problem statement, as explained in Barton, Berwick, and Ristad (1987).

in the grammar which permits the passive category *VP*[PAS] to dominate just the other material from the original rule, together (optionally) with a *PP*[by]" (GKPS:59). In Ristad (1986a), the *finite closure* problem is used to determine the cost of metarule application. Principles of universal feature instantiation (UFI) apply to the resulting enlarged set of ID rules, defining a set of phrase structure trees of depth one (local trees). One principle of UFI is the head feature convention, which ensures that phrases are projected from lexical heads. Informally, the head feature convention is GPSG's X-theory. Ristad (1986a) uses the *category membership* problem to determine, in part, the cost of mapping ID rules to local trees. Finally, linear precedence statements are applied to the instantiated local trees. LP statements order the unordered daughters in the instantiated local trees. The ultimate result, therefore, is a set of ordered local trees, and these are equivalent to the context-free productions in a context-free grammar. The resulting context-free grammar derives the language of the GPSG.

The process of assigning structural descriptions to utterances consists of two steps in GPSG: the *projection* of ID rules to local trees and the *derivation* of utterances from nonterminals, using the local trees. Accordingly, formal devices may supply resources to either process.

2.2 Theory of Syntactic Features

In current GPSG theory, syntactic categories (nonterminals) encode linguistic relations as feature–value pairs. If a relation is true of two categories in a phrase structure tree, then the relation will be encoded in every category on the unique path between the two categories. The primary computational resource provided by the theory of syntactic features is polynomial space, primarily due to the large number of possible syntactic categories arising from finite feature closure. Ristad (1986a) observes that finite feature closure admits a surprisingly large number of possible categories: $\theta(3^{a \cdot b!})$ where a is the number of atomic-valued features and b the number of category-valued features. In fact, there are more that 10^{775} categories in the GKPS system.

Fortunately, the full power of embedded categories does not appear to be linguistically necessary because no category-valued feature need ever contain another.[3] In GPSG, there are three category-valued features: SLASH, which marks the path between a gap and its filler with the category of the filler; AGR, which marks the path between an argument and the functor that syntactically agrees with it (between the subject and matrix verb, for example); and WH, which marks the path between a *wh*-word and the minimal clause that contains it with the morphological type of the *wh*-word. AGR will never contain SLASH because a functor (verb or predicate) will never select a gap or a constituent containing a gap as it's argument. Conversely, SLASH will never be required to contain AGR because such a category corresponds to "the following imaginary (and rather weird) case: Suppose we found a language in which finite verb phrases could be fronted over an unbounded domain provided that they were in the agreement form associated with third-person-singular NP controllers" (Pullum, personal communication). Similarly, because the value of WH is the category of a *wh*- noun phrase, and because *wh*- nominals never contain gaps, WH can never contain SLASH or AGR. In point of fact, no category embeddings appear in the GKPS grammar for English, and it is difficult to see how they would appear in a GPSG for any other natural language.

The obvious revision, then, is *unit feature closure*: to limit category-valued features to containing only 0-level categories. (0-level categories do not contain any category-valued features). I adopt this strongly falsifiable constraint in RGPSG. The depth of category-embedding is purely an empirical issue, and hence unit closure is not *ad hoc*. The other revision is primarily notational: any RGPSG feature f may assume the distinguished values noBind or unBound in addition to those values determined by $\rho(f)$. A noBind value indicates that the feature may not receive a value in an extension of the given category, while unBound indicates that the feature does not currently have a value, and may receive one in extension.

2.3 Immediate Dominance/Linear Precedence

GPSG's ID/LP format models certain word order phenomena, such as the head parameter and some free word order facts. An ID rule is a context-free production

$$C_0 \rightarrow C_1, C_2, \ldots, C_k$$

whose left-hand side (LHS) is the mother category and whose right-hand side (RHS) is an unordered multiset of daughter categories, some of which may be designated as *head daughters*. The LHS *immediately dominates* the unordered RHS in a tree of depth one (a *local tree*).

2.3.1 Complexity in ID/LP

ID rules significantly increase the time resources available to the GPSG derivation process in four related ways. First, a derivation step is nondeterministic because a category may immediately dominate more than one RHS. Second, the derivation process may *alternate* between a derivation step involving the ID rules $C \rightarrow C_1 \mid \ldots \mid C_k$ that corresponds to an OR-transition (only one of k possible successors must yield a terminal string) and a derivation step involving an ID rule $C \rightarrow C_1, C_2, \ldots, C_k$ that corresponds to an AND-transition (all k successors must yield terminal strings). These two devices introduce lexical and structural ambiguity. As is well-known, ambiguity is a central property of natural languages. Therefore, I consider this aspect of ID rules linguistically essential, and it will be retained in RGPSG.

Third, unrestricted null transitions in ID rules are a source of intractability because they allow GPSGs to generate enormous phrase structure trees whose yield is the empty string (see Ristad, 1986a). Thus, a parser that used such a grammar must nondeterministically postulate elaborate phrase structure in between its input tokens. The indisputable unnaturalness of this ability motivates me to greatly restrict null transitions in RGPSG.

Fourth, the multiset RHS of an ID rule contributes to a large space of local phrase structure trees: an ID rule with a a RHS of cardinality b can, if unconstrained by LP statements, correspond to $b!$ ordered productions. In parsing practice, this can cause a combinatorial explosion in a context-free parser's state space (see Barton, 1985). In addition to causing nondeterminism in

[3] Let f and g be any distinct category-valued features. I am arguing that although f may appear inside g in some language, f will never be *required* to appear inside g.

any GPSG-based parser, the multiset RHS confers on GPSG the ability to count nonterminals. The apparent artificiality of this device, as discussed in Barton, Berwick, and Ristad (1987:260-261), will motivate me to adopt a substantive constraint of *short ID rules* in RGPSG (binary branching, for example).[4]

2.3.2 Revised ID/LP

RGPSG ID rules have exactly one mother and at least one head daughter. The heads are separated notationally from the nonheads by a colon, and appear to the left of the colon. The mother and all head daughters are implicitly specified for [NULL -]. For example, the RGPSG headed ID rule 2 corresponds to the GPSG ID rule 3.

$$V2 \rightarrow \texttt{[SUBCAT 2]} : N2 \tag{2}$$

$$V2\,\texttt{[NULL -]} \rightarrow \texttt{H[SUBCAT 2,NULL -]}, N2 \tag{3}$$

There is only one lexical element for the null string, and it is universal across all grammars:

$$X2\,\texttt{[SLASH}\ X2_1\texttt{,NULL +]}_1 \rightarrow \epsilon$$

Co-subscripting indicates that the two *X2* categories must be identical in any legal projection of the rule, with the exception of the [NULL +] and SLASH specifications. This restricted ID rule format, when coupled with a restriction on metarules that prevents them from affecting head daughters, prevents head daughters from ever being erased in a RGPSG derivation. Thus, null transitions are effectively eliminated from RGPSG.

An *ordered production* is an ID rule whose daughters are completely linearly ordered, that is, a string of daughter categories rather than multisets of head and nonhead daughters. An ordered production is *LP-acceptable* if all LP statements in the RGPSG are true of it.

The RGPSG ID/LP formalism does not contain formal constraints sufficient to guarantee polynomial-time recognition, although the linguistically justified use of *short ID rules* can render ID rules tractable, because ID/LP grammars with bounded rules can be parsed in time polynomial in the grammar size.[5]

2.4 Metarules

Metarules are lexical redundancy rules. Formally, they are functions that take *lexical ID rules*—ID rules with a lexical head—to sets of lexical ID rules. See the GKPS passive metarule above. The GKPS grammar for English also includes metarules for subject-aux inversion, extraposition, and transitivity alternations. The complete set of ID rules in a GPSG is the maximal set that can be arrived at by taking each metarule and applying it to the set of rules that did not themselves arise from the application of that metarule. This maximal set is called the *finite closure* $FC(M, R)$ of a set R of lexical ID rules under a set M of metarules.

2.4.1 Complexity of Metarules

Metarules can increase the time and space resources available to the derivation process by introducing null transitions and ambiguity in ID rules and by increasing the space of ID rules more than exponentially. They can also increase the cost of the projection process itself: finite closure is nondeterministic (NP-hard, in fact) because metarules are applied to ID rules nondeterministically.

2.4.2 Revised Metarules

Unrestricted null transitions are both linguistically and computationally undesirable. Moreover, the ability of metarules to affect lexical head daughters is in direct conflict with their linguistic purpose: "to express generalizations about the subcategorization possibilities of lexical heads." (GKPS:59) Unrestricted metarules can destroy the relation between a phrase and its lexical head, and thereby violate X-theory. The first step in revising metarules is to restrict them to *only* affect nonhead daughters in lexical ID rules. Because of this change, metarules cannot alter the implicit [NULL -] specification on the head daughters. Therefore, once a category is expanded in a derivation, it *must* be lexically realized in the derived string. This formal constraint ensures that the empty string does not have elaborate phrase structure in RGPSG.

Metarule finite closure generates many linguistically incorrect ID rules that must be excluded by other GPSG devices (FCRs, for example). The GKPS grammar for English contains six metarules; out of approximately 1944 possible metarule interactions in principle, only two such interactions appear to be productive (passive followed by subject-aux inversion or slash termination metarule 1).[6] Therefore, the second metarule restriction adopted by RGPSG is *biclosure*, instead of finite closure.[7]

[4] The binary branching constraint is independently motivated by the linguistic arguments of Kayne (1981) and others. In that work, Kayne argues that the path from a governed category to its governor (for example, from an anaphor to its antecedent) must be unambiguous—informally put, "an unambiguous path is a path such that, in tracing it out, one is never forced to make a choice between two (or more) unused branches, both pointing in the same direction" (Kayne 1981:146). The unambiguous path requirement sharply constrains fan-out in phrase structure trees because n-ary branching, for $n > 2$, is only possible when none of the n sister nodes must govern any other nodes in the phrase structure tree.

[5] If the length bound for natural language grammars is the constant b, then any ID/LP grammar G can be converted into a strongly-equivalent CFG G', of size $\theta(|G| \cdot b!) = \theta(|G|)$ by simply expanding out the constant number of linear precedence possibilities. In the GKPS and RGPSG grammars for English, $b = 3$ because double object constructions ([*give NP NP*], for example) are assigned a flat, ternary branching structure. (I ignore the iterating coordination schema, which licenses rules with unbounded right-hand sides.) It is important, however, that the short rules reflect a genuine constraint and that the grammar does not use some other mechanism to get the effect of longer rules (feature instantiation, for example).

[6] Given a set of n metarules, the number of possible metarule interactions is the number of ways to pick n or less metarules from the set, where order matters and repetitions are not allowed. That number is given by the total number of possible k-selections from the n metarules, where k varies from 0 (no metarules apply) to n (any combination of all metarules apply). Thus, the number of possible interactions $f(n)$ is: $\sum_{k=0}^{n} \frac{b!}{(b-i)!} \approx b! \cdot e$). This is not the size of metarule finite closure, because it does not consider the possibility of a metarule matching an ID rule in more than one way.

[7] Metarule biclosure does not overgenerate as badly as finite closure, and thereby promotes descriptive adequacy at the expense of some explanatory power. Biclosure has an edge in descriptive economy (explanatory power) over unit closure because simpler (and less) metarules are needed with biclosure. Thus, the length of metarule derivations is not totally *ad hoc* because it is subject to scientific criterion.

2.5 Principles of Universal Feature Instantiation

The ID rules obtained by taking the finite closure of the metarules on the ID rules are *projected* to local phrase structure trees. Abstractly, this process establishes the connection between those relations encoded in ID rules (for example, domination, subcategorization, case, modification, and predication) and the nonlocal linguistic relations. Local trees are projected from ID rules by mapping the categories in a rule into legal extensions of those categories in the projected local tree.

Principles of *universal feature instantiation* (UFI) constrain this projection by requiring categories in a local tree to agree in certain feature specifications when it is possible for them to do so. For example, the head feature convention (HFC) requires the mother to agree with all head features that the head daughters agree on, if agreement is possible. The HFC expresses $\bar{X}$-theory in part, requiring a phrase to be the projection of its head. It also plays a central role in the GPSG account of coordination phenomena, requiring the conjuncts in a coordinate structure to all participate in the same linguistic relations with the rest of the sentence. The two other principles of UFI are the *control agreement principle* and the *foot feature principle*. The control agreement principle represents the GPSG theory of predicate–argument relations; informally, it requires predicates to agree with their arguments (for example, verb phrases must agree with their subject NPs in English). The foot feature principle provides a partial account of gap–filler relations in the GPSG system, including parasitic gaps and the binding facts of reflexive and reciprocal pronouns; it plays a role strikingly similar to that of Pesetsky's (1982) path theory and Chomsky's (1986) binding and chain theories.[8] Informally, the foot feature principle ensures that certain syntactic information is not lost. "Exceptional" feature specifications are those feature specifications in an ID rule that should agree by virtue of a principle of UFI, but are unable to without changing a feature specification inherited from the ID rule.

2.5.1 Complexity of UFI

The three principles of UFI all cause intractability because they provide the derivation process with reusable space resources.

First, each principle of UFI can enforce nonlocal feature agreement in phrase structure. Ristad (1986b) shows how this causes NP-hardness, when coupled with lexical ambiguity or null transitions. A related source of intractability is that the projection of ID rules to local trees can create an astronomical space of local trees, which in turn increases parser search space. These two sources of intractability cannot be eliminated because they are essential to GPSG's account of linguistic agreement among conjuncts and between predicates and their arguments, gaps and their fillers, and phrases and their lexical heads.

The use of exceptional feature specifications in these principles allows a derivation to reuse the space resources provided by the ID rules and theory of syntactic features. In the reduction of Ristad (1986a), head features encode an alternating Turing machine tape. The HFC is used to transfer the tape contents for an ATM configuration C_0 (represented by the mother) to its immediate successors $C_1, C_2, \ldots, C_k$ (the head daughters). The configurations $C_0, C_1, \ldots, C_k$ have identical tapes, with the critical exception of one tape square. If the HFC enforced absolute agreement between the head features of the mother and head daughters, the polynomial space ATM computation could not be simulated in this manner.

2.5.2 Universal Feature Instantiation in RGPSG

Principles of universal feature instantiation in RGPSG all preserve a simple invariant across all ID rules. They are monotonic; that is, they never delete or alter existing feature specifications. The head feature convention, for example, ensures that the mother agrees exactly with all head feature specifications that the head daughters agree on, regardless of where the specifications come from.

Principles of UFI are first applied to the ID rule output of metarule unit closure. After this initial application, each principle always applies, governing the well-formedness of the ID rule extension relation. The resulting ID rules derive utterances in the language generated by the RGPSG.

Head feature convention. The head feature convention enforces the invariant that the mother is in absolute agreement with all head features on which the head daughters agree. It also requires the BAR value on a head daughter to be less than or equal to the BAR value on the mother. *HEAD* contains exactly those features that must be equivalent on the mother and head daughters of every ID rule.[9]

$$HEAD = \{\text{AGR}, \text{ADV}, \text{AUX}, \text{INV}, \text{LOC}, \text{N}, \text{NFORM}, \text{PAS}, \text{PAST}, \text{PER}, \text{PFORM}, \text{PLU}, \text{PRD}, \text{V}, \text{VFORM}\}$$

Control agreement principle. The control agreement principle (CAP) differs from the HFC in that it establishes equivalences (*links*) between the categories in an ID rule: when two categories are *linked* in an ID rule, the two categories must be identical in any legal extension of that rule. Links are calculated immediately after the HFC has applied to the ID rules for the first time; once a link is established in an ID rule, it cannot be changed or undone.[10] The first part of the CAP calculates control relations between categories, while the second part of the CAP establishs

[8] The possibility of expressing the control agreement and foot feature principles as local constraints on nonlocal relations falls out from the central role of c-command, or equivalently unambiguous paths, in binding theory. C-command is a local relation, in fact the primary source of locality in phrase structure (see Berwick and Wexler 1982). Similarly, the possibility of encoding multiple gap–filler relations in one feature specification of one category corresponds to the "no crossing" constraint of path theory. Pesetsky (1982:556) compares the predictions of path theory and principles of UFI when the two diverge in cases of double extraction (for example, *a problem that$_i$ I know who$_j$ to [$_\alpha$ talk to e_j about e_i]*) from coordinate structures. He concludes that "the apparent simplicity of the slash category solution fades when more complex cases are considered."

[9] In order to properly account for feature instantiation in the binary and iterating coordination schemata, the binary head (*BHEAD*) features BAR, SUBJ, SUBCAT, and SLASH are considered to be head features for the purposes of the HFC in all nonlexical, multiply-headed ID rules.

[10] In GKPS, only head feature specifications and inherited foot feature specifications determine the semantic types relevant to the definition of control. RGPSG simplifies this by considering inherited feature specifications and only some head feature specifications. Alternatively, control relations could be calculated every time the HFC instantiates a feature specification.

links using the control relations. In all cases, linking is indicated by co-subscripting.

RGPSG control relations are calculated as follows. A *predicate* is a *VP* or an instantiation of *XP*[+PRD] such as a predicate nominal or adjective phrase. The *control feature* of a category C_i, where $C_i(\texttt{BAR}) \neq 0$, is SLASH if C_i is specified for SLASH; otherwise, it is AGR. Control is calculated once and for all immediately after the HFC has applied to the ID rules resulting from metarule unit closure.

Let f be the control feature of a category C_1. Then C_1 is controlled by C_2 in a rule if and only if $C_1(f) = C_2$, $C_2 \sqsupseteq X2$, and either the rule is $C_0 \rightarrow C_1 : C_2$ (recall that C_1 is the head daughter), or the rule is $C_0 \rightarrow C_3 : C_1, C_2$, and $C_0, C_1 \sqsupseteq VP$.

The RGPSG control agreement principle states: In an ID rule

$$r = C_0 \rightarrow C_1, \ldots, C_j : C_{j+1}, \ldots, C_n$$

- If C_i controls C_k and f_k is the control feature of C_k, then $C_k(f_k)$ and C_i are linked.
- If there is a nonhead predicate C_i with no controller, then link $C_i(f_i)$ and $C_0(f_0)$, where f_i and f_0 are the control features of C_i and C_0, respectively.

In the theory of GKPS, the control agreement principle performs subject-verb agreement by enforcing a control relation between the two daughters of the rule

$$S \rightarrow \texttt{H[-SUBJ]}, X2$$

In RGPSG, this rule must be stated as

$$S \rightarrow X2\,\texttt{[-SUBJ, AGR}\ X2\texttt{]} : X2$$

if we wish to enforce the control relation between the two daughters. Because control relations in RGPSG are static (never recalculated), this control relation exists even if $X2 \neq NP$. Fortunately, no verb will ever be specified for [AGR *AP*] in the lexicon, and therefore any "questionable" control relations involving an *X2* other than *NP* are ignored at the lexical insertion level.

Foot feature principle. The foot feature principle (FFP) requires any foot feature specification instantiated on a daughter category to also be instantiated on the mother. The specification is identical to any instantiation of the same feature on other daughter categories. The FFP ensures that (1) the existence of inherited foot features on any category of an ID rule blocks instantiation of those foot features on any other component category of the rule, and (2) inherited foot features are equivalent across all component categories of the rule. This second condition may be too strong.

Because the empty string can be dominated only by a category of the form α[NULL +, SLASH α] in RGPSG, the FFP tries to ensure that every gap will have a unique filler. Unfortunately, it is impossible to truly guarantee recoverability of deletions in RGPSG, because the FFP can only locally constrain the rule-to-tree projection, and not the ID rules themselves. This situation is unavoidable in the GPSG framework, simply because SLASH does not always mark the complete path between a gap and its filler in accepted GPSG analyses. The classic example is the GPSG analysis of subject dependencies, where an *S/NP* is reanalyzed as a *VP*, effectively deleting an *NP* gap in subject position. In GKPS, this operation is performed by slash termination metarule 2 (GKPS:160-2): [SLASH *NP*] only marks the path from the filler to the mother of the reanalyzed *VP*. Another example is the GKPS (pp. 150-152) analysis of missing-object constructions such as *John is easy to please*. In missing-object constructions, [SLASH *NP*] only marks the path from the *NP* gap to the *V2*[INF]/*NP* dominating *to please*, failing to continue through the *AP easy to please* to the filler *John*. Many sweeping changes would be necessary before the FFP would be able to strictly enforce recoverability of deletions in RGPSG.

2.6 Marking Conventions

Feature co-occurrence restrictions (FCRs) and feature specification defaults (FSDs) are explicit marking conventions used in the GPSG system both to express language-particular facts and to restrict the overgeneration of other formal devices (both metarule and feature closure). FCRs and FSDs are restrictive predicates on categories, constructed by Boolean combination of feature specifications. All legal categories must unconditionally satisfy all FCRs. All categories must also satisfy all FSDs, if it is possible to do so without violating an FCR or a principle of universal feature instantiation. For example,

$$\textit{FCR 1:}\ \texttt{[INV +]} \supset (\texttt{[AUX +]} \wedge \texttt{[VFORM FIN]})$$

requires any category that bears the [INV +] feature specification to also bear the specifications [AUX +] and [VFORM FIN].

2.6.1 Complexity of Marking Conventions

FCRs and FSDs both provide significant resources to the GPSG projection process. First, they allow the projection process to reuse the polynomial space provided by the theory of syntactic features, because they can establish equivalences between the features in a category C and the features in a category contained in C. This ability to apply across embedded categories vastly increases the complexity of the rule-to-tree projection. To see why it is linguistically unnecessary, consider the role of embedded categories. A category-valued feature f expresses a nonlocal linguistic relation between a category C and the one or more categories that bear the feature specification $[f\ C]$. Thus, in the linguistically relevant cases, every embedded category eventually "surfaces" in phrase structure, where the marking conventions are free to apply. The one exception to this argument is FCR 13 in the GKPS grammar for English, which applies 'across' an embedded category.

$$\textit{FCR 13:}\ \texttt{[FIN, AGR}\ NP\texttt{]} \supset \texttt{[AGR}\ NP\texttt{[NOM]]}$$

In RGPSG, marking conventions may not apply to or across embedded categories. The effect of FCR 13 is achieved in RGPSG by a combination of the simple default SD 2 in section 3.2.2 below and carefully written ID rules.

Second, FCRs and FSDs of the "disjunctive consequence" form $[f\ v] \supset [f_1\ v_1] \vee \cdots \vee [f_n\ v_n]$ compute the direct analog of the NP-complete satisfiability problem: when several such

FCRs are used together, the GPSG must nondeterministically try all n feature-value combinations.

Third, the process of applying FSDs to local trees is very complex, in part because it is not informationally encapsulated. Rather than simply considering the (existing) feature specifications in each target category separately, FSD application is affected by the other categories in the ID rule, all principles of universal feature instantiation, and even FCRs.

2.6.2 Simple Defaults in RGPSG

There is no reason to believe that marking conventions need be so powerful and unconstrained. The approach RGPSG takes is to virtually eliminate marking conventions. Rather than stating the internal constraints on categories explicitly (and redundantly), as FCRs do, RGPSG eliminates FCRs altogether. Instead, the constraints FCRs express are implicitly stated in the rest of the grammar — in the way ID rules and metarules are written, for example. The sole explicit marking convention in RGPSG is the *simple default* (SD). Unlike FCRs and FSDs, SDs are constructive, easy to understand and computationally tractable. Each SD is applied (and may be understood) to each category independent of all other categories and RGPSG formal devices, including other SDs. SDs are applied to ID rules immediately after the initial application of principles of UFI.

An SD contains a predicate and a consequent. The consequent is a list of feature specifications. The predicate is a Boolean combination of truth-values and feature specifications such that if a category C bears or extends a given feature specification, that feature specification is true of C, else false. If the predicate is true of a given category C in a rule and the consequent includes only unbound and unlinked features, then the feature specifications listed in the consequent are instantiated on C. Each SD is applied simultaneously to every top-level category in every rule exactly once, in the order specified by the grammar. Consider the following SD:

SD 1: if [SUBCAT] *then* [BAR 0]

If the target category C in a ID rule is specified for the SUBCAT feature, but unspecified for the BAR feature, then the SD will force the feature specification [BAR 0] on C.

3 The Revised Theory

In this section, I explain how the formal subsystems described above fit together. I begin by formally specifying the class of RGPSGs and the languages they generate. I conclude by translating the GKPS analysis of topicalization, explicative pronouns, and parastic gaps to the RGPSG formal system.

Figure 1 shows the internal organization of RGPSG. The set of ID rules R' defined by metarule unit closure, UFI, and SD application generates the language of the RGPSG as follows. If R' contains a rule $A \rightarrow \gamma$ with an extension $A' \rightarrow \gamma'$ that satisfies all principles of UFI and is an LP-acceptable ordered production, then for any string of terminals α and nonterminals β, we write $\alpha A' \beta \Rightarrow \alpha\gamma'\beta$. This is a derivation step. The language of an RGPSG contains all terminal strings that can be derived, using

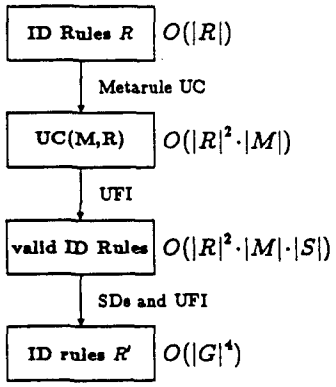

Figure 1: This diagram shows internal organization of an RGPSG G with ID rules R, metarules M, and simple defaults S. The O-bounds show the effect of various formal devices on derived grammar symbol size.

the ID rules, from any extension of the distinguished start category. Let $\stackrel{*}{\Rightarrow}$ be the reflexive transitive closure of $\Rightarrow$. Then the language $L(G)$ generated by G is

$$L(G) = \{\, x \mid x \in V_T^* \text{ and } \exists C \in K[(C \sqsupseteq \texttt{Start}) \wedge C \stackrel{*}{\Rightarrow} x]\}$$

Ristad (1986b) proves that universal recognition problem for RGPSG is NP-complete, a significant decrease in complexity from the EXP-POLY time hardness of GPSG-Recognition.[11] In fact, of the more than ten sources of intractability lurking in GPSG, only two remain in RGPSG — lexical ambiguity and nonlocal feature agreement. Critically, these two sources of intractability in RGPSG appear to be linguistically essential.

3.1 Efficient RGPSG Parsing

Intractability in RGPSG arises from a particularly deadly combination of feature agreement and lexical ambiguity. Underspecification of categories in ID rules and metarules can be costly. This suggests that limiting the number of head features or the scope of their agreement will mitigate the intractability. An efficient recognition algorithm might approximate grammaticality by failing to transfer all head features through coordinate structures (for example, letting them assume default values instead), or by aborting a parse in the face of excessive lexical or structural ambiguity. Efficient parsing techniques based on partial enforcement of UFI are also possible. One such implementation, which propagates feature specifications bottom up using Earley's algorithm, is in progress at Thinking Machines Corporation.

[11] This decrease in complexity is significant from both theoretical and practical perspectives. First, NP-complete problems typically have good average time algorithms, while EXP-POLY problems do not. Next, the fastest recognizer known for GPSGs can require double-exponential time in the worst case, while RGPSG has a simple exponential time recognizer. Finally, NP-complete problems have efficient witnesses, while EXP-POLY hard problems do not. This means that RGPSG parses can always be verified efficiently, while GPSG parses cannot, in general.

Barton (1986) proposes a constraint-based computational solution to intractability in the two-level Kimmo morphological analyzer. Intractability arises from unbounded agreement processes in that system, and similar techniques based on constraint propagation may be adapted to create an efficient *approximate* parsing algorithm for RGPSG. Tuples of features would correspond to constraint-propagation nodes, while tuples of sets of feature-values would correspond to node labels; features could receive multiple values in this implementation. Nodes would be connected by both RGPSG ID rules and principles of universal feature instantiation.

3.2 Linguistic Analysis of English

This section reproduces three of the more intricate linguistic analyses of GKPS in order to illustrate RGPSG's formalisms. To reproduce their comprehensive analysis of English in toto would be a disservice to that work and is beyond the scope of this paper. Instead, Ristad (1986b) provides an RGPSG roughly equivalent to their GPSG for English; the reader should consult GKPS for the accompanying linguistic exposition. In all cases, co-subscripting indicates linking.

3.2.1 Topicalization

The rule 4a expands clauses and rule 4b introduces unbounded dependency constructions (UDCs) in English.

$$\begin{array}{l} a.\ S \rightarrow X2\,[\texttt{SUBJ -, AGR } X2] : X2 \\ b.\ S \rightarrow X2\,[\texttt{SUBJ +, SLASH } X2] : X2 \end{array} \tag{4}$$

In both cases the *X2* nonhead daughter controls the head daughter, and the control agreement principle links the value of the head daughter's control feature with the *X2* daughter, creating the ID rules in 5.

$$\begin{array}{l} a.\ S \rightarrow VP\,[\texttt{AGR } X2_1] : X2_1 \\ b.\ S\,[\texttt{SLASH noBind}] \rightarrow S\,[\texttt{SLASH } X2_2] : X2\,[\texttt{SLASH noBind}]_2 \end{array} \tag{5}$$

In the following discussion, [3s] and [3p] abbreviate [PER 3,-PLU] and [PER 3,+PLU], respectively. Note that it is impossible to extract any constituent out of the *X2* daughter in 5b because the foot feature principle has forced [SLASH noBind] on the *X2* daughter and its mother. This explains the unacceptability of 6 in RGPSG, which is permissible in the theory of GKPS.

* New York [[the girl from ___] [we want ___ to succeed]] (6)

3.2.2 Explicative pronouns

Now I account for the distribution of the explicative pronouns *it* and *there* in infinitival constructions on the basis of postulated ID rules and principles of universal feature instantiation (see GKPS, pp.115-121). The feature specification [AGR *NP*[NFORM α]] is abbreviated as $+\alpha$ below, where α is it, there, or NORM.

The RGPSG for English includes the ID rules 7,

$$\begin{array}{l} a.\ S \rightarrow X2\,[\texttt{-SUBJ, AGR } X2] : X2 \\ b.\ VP \rightarrow [13] : VP[\texttt{INF}] \\ c.\ VP \rightarrow [16] : (PP[\texttt{to}]), VP[\texttt{INF}] \\ d.\ VP \rightarrow [17] : NP, VP[\texttt{INF}] \\ e.\ VP[\texttt{AGR } S] \rightarrow [20] : NP \end{array} \tag{7}$$

the simple defaults 8,

$$\begin{array}{l} a.\ SD\ 1\text{:}\ \textit{if}\ [\texttt{SUBCAT}]\ \textit{then}\ [\texttt{BAR 0}] \\ b.\ SD\ 2\text{:}\ \textit{if}\ [\texttt{+V,-N,-SUBJ}]\ \textit{then}\ [\texttt{+NORM}] \end{array} \tag{8}$$

the extraposition metarule 9,

$$\begin{array}{c} X2\,[\texttt{AGR } S] \rightarrow W \\ \Downarrow \\ X2\,[\texttt{+it}] \rightarrow W, S \end{array} \tag{9}$$

and the lexical entries 10. All other nouns are specified for [NFORM NORM] by their lexical entries.

$$\begin{array}{l} \langle it, NP[\texttt{PRO,-PLU,NFORM it}]\rangle \\ \langle there, NP[\texttt{PRO,NFORM there}]\rangle \end{array} \tag{10}$$

From the ID rules in 7, RGPSG generates the following ID rules.

$$\begin{array}{l} a.\ VP[\texttt{AGR}_1] \rightarrow V0[\texttt{13,AGR}_1] : VP[\texttt{INF,AGR}_1] \\ b.\ VP[\texttt{AGR}_1] \rightarrow V0[\texttt{16,AGR}_1] : (PP[\texttt{to}]), VP[\texttt{INF,AGR}_1] \end{array} \tag{11}$$

The absence of a controlling category allows the CAP to link the AGR values of the mother and *VP*[INF] predicate daughter. The HFC then links the AGR values of the mother and lexical head daughter. SD 1 specifies the head daughter for [BAR 0], while SD 2 cannot affect the linked AGR values.

$$VP[\texttt{AGR}_1\ NP[\texttt{NORM}]] \rightarrow V0[\texttt{14,AGR}_1\ NP[\texttt{NORM}]] : V2[\texttt{INF, AGR}_1\ NP[\texttt{NORM}]]$$

The CAP and HFC operate identically as in 11, except that the [+NORM] specification is inherited from the ID rule 7b and propagated through the rule by the CAP and HFC.

$$VP[\texttt{AGR}_2\ NP[\texttt{NORM}]] \rightarrow V0[\texttt{17,AGR}_2\ NP[\texttt{NORM}]] : NP_1, VP[\texttt{INF, AGR}_1\ NP] \tag{12}$$

The *NP* daughter controls its *VP*[INF] sister, and the CAP links the AGR value of the *VP* to its sister *NP*. SD 2 specifies the mother for [+NORM], and the HFC forces this specification on the head daughter.

The rules 13 introduce [+it] and [+there] specifications. Note that 13a is the result of the extraposition metarule on the ID rule 7e.

$$\begin{array}{l} a.\ VP[\texttt{+it}] \rightarrow [20] : NP, S \\ b.\ VP[\texttt{+it}] \rightarrow [21] : (PP[\texttt{to}]), S[\texttt{FIN}] \\ c.\ VP[\texttt{AGR } NP[\texttt{+there,PLU } \alpha]] \rightarrow [22] : NP[\texttt{PLU } \alpha] \end{array} \tag{13}$$

The rules in 13 may only expand the *VP* daughters of the ID rules 11 and 12 in a derivation (compare their AGR values). Thus, the grammar claims that explicative pronouns only occur in utterances generated using the rules in 13, in combination with the "extending" rules 11 and 12. This describes the following facts from GKPS, p. 120.[12]

$$\left\{ \begin{array}{c} \text{It} \\ \text{*There} \\ \text{*Kim} \end{array} \right\} [\ \text{continues}\ [\ \text{to bother}\ [\ \text{Lou}\ ]\ [\ \text{that Robin was chosen}\ ]]] \tag{14}$$

[12] In order to better understand these examples, associate each constituent with the ID rule that generated it. To help with this task, the main verbs and their SUBCAT values are: $\langle continue, 13\rangle$, $\langle appear, 16\rangle$, $\langle believe, 17\rangle$, $\langle bother, 20\rangle$, $\langle be, 22\rangle$.

$$\left\{ \begin{array}{c} *\text{It} \\ \text{There} \\ *\text{Kim} \end{array} \right\} \text{[appeared (to us) [to be [nothing in the park]]]} \quad (15)$$

$$\text{Leslie [believed } \left\{ \begin{array}{c} \text{it} \\ *\text{there} \\ *\text{Kim} \end{array} \right\} \text{[to bother [us] [that Lee lied]]]} \quad (16)$$

$$\text{We [believed } \left\{ \begin{array}{c} *\text{it} \\ \text{there} \\ *\text{Kim} \end{array} \right\} \text{[to be [no flaws in the argument]]]} \quad (17)$$

3.2.3 Parasitic gaps

Simple parasitic gaps, that is, those introduced in verb phrases by lexical rules, present no problem for RGPSG because the FFP demands all instantiations of SLASH on daughters to be equal to each other and equal to the SLASH instantiation on the mother.

$$\begin{array}{l} VP/NP \\ \quad V0[13] \\ \quad NP/NP \\ \quad PP[\text{to}]/NP \end{array} \quad (18)$$

Kim wondered which models

$$\text{Sandy } \left\{ \begin{array}{l} \text{[had sent [pictures of \_\_] [to \_\_]]} \\ \text{[had sent [pictures of \_\_] [to Bill]]} \\ \text{[had sent [pictures of Bill] [to \_\_]]} \end{array} \right\} \quad (19)$$

The FFP insists nonlexical heads be instantiated for SLASH if any nonhead daughter is, thereby explaining the unacceptability of 20 and the acceptability of 21.

a. $* S/NP$
$\quad NP/NP$
$\quad VP$ (20)
b. * Kim wondered which authors
[[reviewers of __] [always detested sushi]]

a. S/NP
$\quad NP/NP$
$\quad VP/NP$ (21)
b. Kim wondered which authors
[[reviewers of __] [always detested __]]

This analysis of parasitic gaps exactly follows the one presented in GKPS on matters of fact. These facts may be questionable, however. Some sentences considered acceptable in GKPS (for example, *Kim wondered which models Sandy had sent pictures of to Bill* and *Kim wondered which authors reviewers of always detested*) are marginal for some native English speakers. Note that both sentences are marked unacceptable in the GB framework because of subjacency violations.

4 Conclusion

This work is similar to that of Shieber (1986) in its attempt to reconstruct GPSG theory. Shieber, however, is concerned solely with creating a more easily implementable description of GPSG theory, rather than with changing the theory in a linguistically or computationally significant way.

It would be instructional to identify and restrict the computational resources provided by the formal devices in other linguistic theories (for example, lexical-functional grammar, government-binding theory, or morphological theory). Barton, Berwick, and Ristad (1987) explores the utility of complexity analysis in other linguistic domains, although the research strategy reported here is not the focus of that work.

5 References

Barton, E., 1985. On the complexity of ID/LP parsing. *Computational Linguistics* 11(4):205–218.

Barton, E., 1986. Constraint propagation in Kimmo systems. *Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics*. Columbia University, New York: Association for Computational Linguistics

Barton, E., R. Berwick, and E. Ristad, 1987. *Computational Complexity and Natural Language*. Cambridge, MA: MIT Press.

Berwick, R. and K. Wexler, 1982. Parsing efficiency and c-command. *Proceedings of the First West Coast Conference on Formal Linguistics*. Los Angeles, CA: University of California at Los Angeles, pp. 29–34.

Chomsky, N., 1986. *Knowledge of Language: Its Origins, Nature, and Use*. New York: Praeger Publishers.

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

Kayne, R., 1981. Unambiguous paths. In *Levels of Syntactic Representation*, R. May and J. Koster, eds. Dordrecht: Foris Publications, pp. 143–183.

Pesetsky, D., 1982. Paths and categories. Ph.D. dissertation, MIT Department of Linguistics and Philosophy, Cambridge, MA.

Ristad, E.S., 1986a. Computational complexity of current GPSG theory. *Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics*. Columbia University, New York: Association for Computational Linguistics, pp. 30–39.

Ristad, E.S., 1986b. Complexity of linguistic models: a computational analysis and reconstruction of generalized phrase structure grammar. S.M. Thesis, MIT Department of Electrical Engineering and Computer Science, Cambridge, MA.

Shieber, S., 1986. A simple reconstruction of GPSG. *Proceedings of the 11th International Conference on Computational Linguistics*. Bonn, West Germany, 20-22 August, 1986.