RELATIONSHIP BETWEEN STRONG AND WEAK GENERATIVE POWER OF FORMAL SYSTEMS*

Aravind K. Joshi

Department of Computer and Information Science and
Institute for Research in Cognitive Science
University of Pennsylvania
Philadelphia, PA 19104
USA
joshi@linc.cis.upenn.edu

Abstract

The relationship between strong and weak generative powers of formal systems is explored, in particular, from the point of view of squeezing more strong power out of a formal system without increasing its weak generative power. We examine a whole range of old and new results from this perspective. However, the main goal of this paper is to investigate the strong generative power of Lambek categorial grammars in the context of crossing dependencies, in view of the recent work of Tiede (1998).

Introduction

Strong generative power (SGP) relates to the set of structural descriptions (such as derivation trees, dags, proof trees, etc.) assigned by a formal system to the strings that it specifies. Weak generative power (WGP) refers to the set of strings characterized by the formal system. SGP is clearly the primary object of interest from the linguistic point of view. WGP is often used to locate a formal system within one or another hierarchy of formal grammars [1]. Clearly a study of the relationship between WGP and SGP is highly relevant, both formally and linguistically. Although there has been interest in the study of this relationship, almost from the beginning of the work in mathematical linguistics, the results are few, as this relationship is quite complex and not always easy to study mathematically (see Miller (1969) for a recent comprehensive discussion of SGP).

Our main goals in this paper are (1) to look at some old and recent results and try to put them in a general framework, a framework that can best be described by the slogan–How to squeeze more strong generative power out of a grammatical system?– and (2) to present a new result concerning Lambek categorial grammars. Our general discussion of the relationship of SGP and WGP will be in the context of context-free grammars, categorial grammars and lexicalized tree-adjoining grammars.

1. Context-Free Grammar (CFG)

McCawley(1967) was the first person to point out that the use of context-sensitive rules by linguists was really for checking structural descriptions (thus related to SGP) and not for generating strings (i.e., WGP), suggesting that this use of context-sensitive rules possibly does not

*This work was partially supported by NSF Grant SBR8920230

[1] SGP is also relevant in the context of annotated corpora. The annotations reflect aspects of SGP and not of the rules of a grammar and therefore WGP.

give more SGP than CFG's. Peters and Ritchie (1969) showed that this was indeed the case. These results are closely related to the notion of recognizable sets of trees (structural descriptions) as explained below.

In a CFG, G, the derivation trees of G correspond to the possible structural descriptions assignable by G. It is easily shown that there are tree sets whose yield language is context-free but the tree sets are not the tree sets of any CFG. That is, we are able to squeeze more strong power out of CFG's indirectly. Here is a simple example.

Let T be the set of trees defined by trees such as

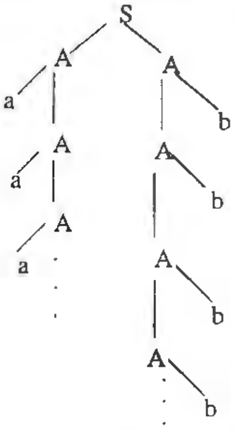


Figure 1: A Recognizable Set of Trees

T in Figure 1. is not a set of derivation trees for any CFG. Clearly in any CFG G, the rules for A will get mixed up and there will be no way we can make sure that all a's are on the left and all b's are on the right. The string language is, of course, $\{a^n b^m | m, n > 1\}$, which is a context-free language. What is the relationship between the trees of the CFG corresponding to this language and the set T? Thatcher (1967) showed that the relationship is very close. Sets such as T, called recognizable sets, are the same as the tree sets of CFG's (called local sets) except possibly for relabeling. It turns out that the tree sets 'analyzable' (i.e., checkable) by context-sensitive rules, as suggested by McCawley are indeed recognizable sets. All these systems have the property that they allow checking of 'local' constraints around a node in a tree. Thatcher's result shows that this notion of 'locality' can be captured by finite state tree automata. Later Joshi, Levy, and Yueh (1975) and Rogers (1997) showed that the notion of 'local context (local tree domains)' can be made substantially richer yet maintaining characterizability by finite state tree automata. All these results can be interpreted as attempts to squeeze more strong power out of a formal system, in this case, context-free grammars.

2. Lexicalized Tree-Adjoining Grammars (LTAG)

The earliest indication that more strong power can be obtained from LTAG by going to tree-local multicomponent TAG (still preserving the weak power of LTAG) is in Weir (1987) and Kroch and Joshi (1989). Shieber and Schabes (1992) introduced the notion of multiple adjoining at the same node in a derivation tree. This move can also be seen as an attempt to get more strong power out of LTAG without going beyond the weak power of LTAG. In fact, the whole range of recent works (Candito (1997), Joshi and Vijay-Shanker (1998), Kulick (2000), Kallmeyer and Joshi (1999) can be seen as attempts to get more SGP from LTAG without going beyond the WGP of LTAG. This is achieved by providing flexibility in interpreting the derivation trees in LTAG. In particular, in Joshi and Vijay-Shanker (1998), Kulick (2000)[2], and Kallmeyer and Joshi (1999), all of which use tree-local multi-component LTAG, flexibility is introduced in the derivation in LTAG resulting in increased strong power without exceeding the weak power of LTAG. This notion of flexibility (flexibility in composition, i.e., in the directionality of the composition) can be briefly defined as follows, at least for the approaches in Joshi and Vijay-Shanker (1998) and Kallmeyer and Joshi (1999). Given a pair of trees, say, $\gamma(1)$ and $\gamma(2)$ the composition (i.e., attachments by substitution and adjoining[3]) can proceed from $\gamma(1)$ to $\gamma(2)$, i.e., $\gamma(1)$ composes with $\gamma(2)$ if $\gamma(2)$ is an elementary tree, otherwise $\gamma(2)$ composes with $\gamma(1)$ if $\gamma(1)$ is an elementary tree, assuming, of course, that $\gamma(1)$ and $\gamma(2)$ are semantically related, i.e., composition of arbitrary unrelated trees is not allowed. Such a notion of flexibility can be introduced in CFG's as well as in Categorial Grammars. However, as far as I know, such a move does not open the door for squeezing more SGP out of the formal system. This is due to the fact that CFG's and Categorial Grammars are essentially string rewriting systems, while systems such as LTAG are tree rewriting systems and the complex topology of the initial trees, when combined with the flexibility discussed above, allows the possibility of augmenting the SGP of the system.

3. Categorial Grammars (CG)

It is well known that the Ajdukiewicz and Bar-Hillel categorial grammars (CG(AB)) are weakly equivalent to CFG's. The derivation trees of CG(AB) are essentially the same as the derivation trees of CFG's (i.e., local sets and therefore recognizable sets (see Tiede (1998)). The relationship of recognizable tree sets to the derivation trees of CG(AB) is not discussed by Tiede. The relationship is the same as between the derivation trees of CFG's and recognizable sets, i.e., they are the same except for relabeling[4].

However, for Lambek Grammars (LG) the situation can be different. In LG, the assignment of categories to lexical items is similar to the assignments in CG(AB) but we have the inference rules associated with the calculus. Although LG's (Lambek, 1958) were long conjectured to be weakly equivalent to CFG's, the conjecture was only recently proved to be true by Pentus (1993). So now the question arises: Do LG's provide more strong generative power than CFG's, in other words, is it possible to characterize the proof trees of LG in terms of something like the recognizable sets or even beyond recognizable sets. This question was raised by Buszkowski and van Benthem[5]. However, only recently a serious attempt has been made by Tiede (1998) to try to answer this question. Tiede (1998) covers a number of aspects and, in particular, suggests that the proof trees of LG may be beyond recognizable sets, i.e., there is a Lambek grammar whose proof tree language is not regular. In fact, he suggests that it will be possible to characterize crossing dependencies. Our main point in this paper is to show that this claim is very limited and that the crossing dependencies that can be described are very degenerate (i.e., the dependencies are between a lexical item and a lexically empty element).

First, note that if indeed true (i.e., nondegenerate) crossing dependencies can be characterized by the proof trees of LG then this would be very surprising indeed. From all that we know

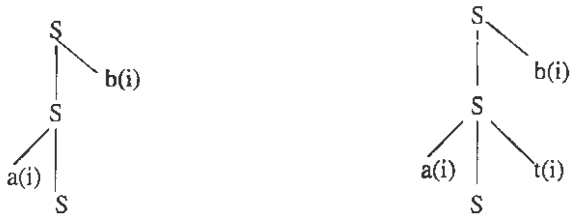
[2] The exact equivalence of the system in Kulick (2000) to tree-local TAG's has not been established yet.

[3] Adjoining at the root and substitution at the foot can be treated as attachments of the same kind.

[4] Another recent work concerning SGP and WGP of categorial grammars is by Jäger (1998) who has investigated the generative capacity of multimodal categorial grammars.

[5] Both these results are discussed in *Handbook of Logic and Language (eds. Johan van Benthem and Alice ter Meulen)*, MIT Press, Cambridge, 1998, pp. 683-736.

so far, any formal system that characterizes crossing dependencies (say, between the a's and b's in a^nb^n) is more powerful than CFG's, for example, TAG's, Combinatory Categorial grammars (CCG), Linear Indexed Grammars (LIG), etc. because they can all generate the language $\{a^nb^nc^n|n \geq 1\}$. Figure 2 shows the topologies needed to obtain the nondegenerate crossing dependencies[6]. Now once we have trees with this topology it is easy to see that the same topology can be used to generate the language $\{a^nb^nc^n|n \geq 1\}$. The relevant tree will be the same as the tree on the left in Figure 2 with c replacing t. Given that LG characterize only context-free languages, this would lead to a paradox.



Tree topologies needed for nondegenerate crossing dependencies

Figure 2: Nondegenerate Crossing Dependencies

We will show that the crossing dependencies claimed by Tiede are degenerate. In particular, they are dependencies between pairs, where the first element is a non empty lexical item and the other an empty element. We will illustrate this by the example in Figure 3.

L = { a, aa, aaa, . . . }   a: S, S/(A/A), S/(S/(A/A))

Proof tree for aa (natural deduction style)

$$\dfrac{\overset{a(1)}{S/(S/(A/A))} \qquad \dfrac{\dfrac{\overset{a(2)}{S/(A/A)} \qquad \dfrac{\dfrac{\overset{t(1)}{[A/A]} \quad \overset{t(2)}{[A]}}{A}\,[/E]}{A/A}\,[/I]}{S}\,[/E]}{S/(A/A)}\,[/I]}{S}\,[/E]$$

Proof trees with empty elements (t's).
Indices are shown for convenience
a(1) a(2) t(1) t(2)

Figure 3: Degenerate Crossing Dependencies in Lambek Grammar

By suitably arranging the introduction and discharge of assumptions in the hypothetical reasoning in the LG we have crossing dependency relations between the a's and the t's, where the t's are the empty elements. In Figure 3 the two assumptions [A/A] and [A] (assumptions are enclosed in []) are introduced at the top level of the deduction. These two assumptions correspond to the empty elements t1 and t2 respectively. Each one of these assumptions is then

[6] The auxiliary tree on the right in Figure 2 is adjoined at the interior nodes of the two trees. We have left out the details about the constraints on the nodes to get precisely the language mentioned above as this is not relevant to the present discussion. Without the constraints the language is more complex, however, this does not affect the argument presented here.

withdrawn using the [/I] rule, the introduction rule (the [/E] rule is used for elimination). Both the assumptions are withdrawn in the deduction as is required in the natural deduction proof. The assumptions that are introduced and then withdrawn have to appear always at the periphery of the proof tree. In our example in Figure 3 they appear at the right periphery. The dependencies between the a's and t's (corresponding to the assumptions) can be seen as follows. In the second [/E] step in the deduction (second from the top) the category A/A is eliminated in combination with S/(A/A) corresponding to a(2). The category A/A in this step resulted from the withdrawal of the assumption [A] (corresponding to t(2)) at the top level. Thus a(2) corresponds to t(2). Similarly a(1) corresponds to t(1). It is easy to see that a natural deduction proof can be constructed for each string in L. Thus we have crossing dependencies between the a's and t's.

For 'true' crossing dependencies both the elements have to be non-empty. The way this is accomplished is by creating two sets of nested dependencies, say between a's and t's and between t's and b's, where the t's are empty elements. Then the resulting dependencies between the a's and b's become crossed as shown in Figure 2 above. The dependencies between a's and t's are nested and those between t's and b's are also nested, resulting in crossing dependencies between a's and b's. Note that the empty elements, t's, are not at the periphery. It is not possible to achieve this in LG because the empty elements have to be at the periphery in the Lambek deduction. So in a real sense the crossing dependencies which Tiede talks about are degenerate and LG is incapable of capturing true crossing dependencies.

Since the Tree-Insertion Grammars (TIG) of Schabes and Waters (1993) are weakly equivalent to CFG's but not strongly, we will explore the implications of TIG for Tiede's work. In fact, we will show that the degenerate case studied by Tiede can be characterized in a TIG. In a TIG, both substitution and adjoining are used. However, adjoining is limited in the following way. First, in each auxiliary tree the footnode is the leftmost (or rightmost) daughter of the root. Further, adjoining is only allowed on the right (or left) frontier. Schabes and Waters (1993) have shown that TIG's are weakly equivalent to CFG's, They do not explore the issue of strong power. Their motivation was to show that TIG's lexicalize CFG's without going beyond the weak power of CFG. We show that strong power is increased, although only to the extent of covering the case of degenerate crossing dependencies considered by Tiede. This suggests the tantalizing conjecture that TIG's are adequate to characterize the proof trees of LG. We have no complete proof of this conjecture at this time.

The proof of the claim that TIG can characterize the degenerate case of crossing dependencies follows from the fact that these dependencies can be implemented by using the TIG in Figure 4. In the tree b1 the footnode is not the rightmost nonterminal on the frontier, however, the frontier to the right of this footnode is lexically empty[7]. Surprisingly, this possibility is allowed in the definition of TIG as it is crucially needed by Schabes and Waters to prove their main result–TIG's are equivalent to CFG's.

Now it is easily seen how the degenerate crossing dependencies of Tiede can be described in this TIG (see Figure 5; b1 is adjoined to a1 at the indicated node in a1).

4. Summary

We have explored a number of old and new results in the study of strong and weak generative powers of formal systems from the point of view of squeezing more strong generative power

[7] What is the equivalent of this result to the case of regular form TAG's defined by Rogers (1994)? TIG's are defined with respect to the topology of the elementary trees and a restriction on adjoining. However, regular form TAG's are defined with respect to derivations. Hence, it is not obvious how the construction will proceed. However, I would conjecture that it should be possible to get a similar result for the regular form TAG's.

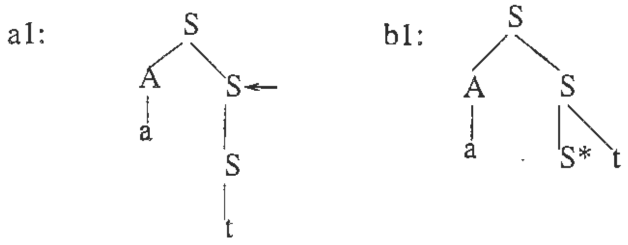
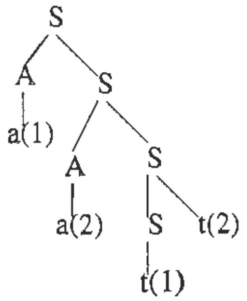


Figure 4: A TIG for Degenerate Crossing Dependencies



Indices are shown for convenience

a(1) a(2) t(1) t(2)

Figure 5: A Derivation in the TIG in Figure 4

out of a formal system without increasing its weak generative power. We have also presented some new results concerning the SGP of Lambek categorial grammars as they relate to crossing dependencies.

5. References

CANDITO M. & KAHANE S. (1998). Defining DTG derivations to get semantic graphs. In *Proc. of the TAG+4 Workshop*, p.25-28, University of Pennsylvania, Philadelphia, PA, USA.

JÄEGER G. (1998). On the generative capacity of multimodal categorial grammars. *Technical Report, 98-26*, Institute for Research in Cognitive Science, University of Pennsylvania, Philadelphia, PA, USA.

JOSHI A. K., LEVY L. & YUEH K. (1975). Local constraints on transformations. *SIAM Journal of Computing*, **8**(4).

KALLMEYER L. & JOSHI A. K. (1999). Factoring predicate argument and scope semantics: Underspecified semantics with LTAG. *Proc. of the Twelfth Amsterdam Colloquium*, p.169-174, University of Amsterdam, Amsterdam, The Netherlands.

KROCH A. & JOSHI A. K. (1985). Linguistic relevance of tree-adjoining grammars. *Technical Report*, University of Pennsylvania, Philadelphia, PA, USA.

KULICK S. (2000). Constrained non-locality: Long-distance dependencies in TAG. *Ph.D. Dissertation manuscript*, University of Pennsylvania, Philadelphia, PA, USA.

LAMBEK J. (1958). The mathematics of sentence structure. *American Mathematical Monthly*, **65**(5).

MCCAWLEY J. W. (1967). Concerning the base component of a transformational grammar. *Foundations of Language*, **4**(3).

MILLER P. H. (1999). *Strong Generative Capacity*, CSLI Publications, Stanford University, Stanford CA, USA.

PENTUS M. (1993). Lambek grammars are context-free. In *Proc. of the 8th Annual Symposium on Logic in Computer Science*, p.35-42, Utrecht, The Netherlands.

PETERS S. & RITCHIE R. (1969). Context sensitive immediate constituent analysis: Context-free languages revisited. In *Proc. ACM Symposium on Theory of Computing*, Marina del Rey, , USA.

ROGERS J. (1994) Capturing CFLs with tree adjoining grammars, In *Proc. of the 32nd Annual Meeting of the Association for Computational Linguistics*, Las Cruces, NM, USA.

ROGERS J. (1997). Grammarless phrase structure grammar. *Linguistics and Philosophy*, **20**(3).

SCHABES Y. & SHIEBER S. M. (1994). An alternative conception of tree-adjoining derivation. *Computational Linguistics* **20**(1).

SCHABES Y. & WATERS R. (1994). Tree insertion grammar. *Technical Report TR-94-13*, Mitsubishi Electric Research Laboratories, Cambridge, MA, USA.

THATCHER J. W. (1967). Characterizing derivation trees of context-free grammars through a generalization of finite automata theory. *Journal of Computer System Science*, **1**(4).

TIEDE H. (1998). Deductive Systems and Grammars. *Ph.D. Dissertation*, Indiana University, Bloomington, IA, USA.

WEIR D. (1988). Characterizing mildly context-sensitive grammar formalisms. *Ph. D. Dissertation*, University of Pennsylvania, Philadelphia. PA, USA.