The Generative Power of Probabilistic and Weighted Context-Free Grammars

Timothy A. D. Fowler

Department of Computer Science
University of Toronto
10 King's College Rd., Toronto, ON, M5S 3G4, Canada

Abstract. Over the last decade, probabilistic parsing has become the standard in the parsing literature where one of the purposes of those probabilities is to discard unlikely parses. We investigate the effect that discarding low probability parses has on both the weak and strong generative power of context-free grammars. We prove that probabilistic context-free grammars are more powerful than their non-probabilistic counterparts but in a way that is orthogonal to the Chomsky hierarchy. In particular, we show that the increase in power cannot be used to model any dependencies that discrete context-free grammars cannot.

1 Introduction

During the last twenty years, the field of computational linguistics has moved from a field which used primarily discrete grammars to a field which uses discrete grammars augmented with weights or probabilities. This revolution has ranged from speech to machine translation to parsing and has resulted in significant advances in terms of both accuracy and speed [4–6, 11].

A variety of work has addressed the power of these numerically augmented grammars. Some have directly analyzed the numerically augmented languages generated by these grammars [1–3, 19, 21]. Other work has analyzed the discrete languages obtained by filtering the weighted languages based on the weights [7, 10, 17, 20]. The results in the former case give us fundamental insight into the probabilistic systems involved but these are difficult to translate into practical consequences. The results in the latter case can be interpreted more easily but are entirely dependent on the filtering method.

This paper will fall into the latter type of work with a goal of providing insight into how context-free grammars (CFGs) are used in the fields of parsing and machine translation. We will be analyzing discrete systems augmented both with probabilities and with weights. Both [7] and [20] analyzed weighted grammars but their results are not relevant to the natural language processing community because they view weights as uninterpretable numerical values rather than something similar to probabilities. When we analyze weighted and probabilistic grammars, we will always abide by the principle that any filtering mechanism must prefer higher weights or probabilities to lower weights or probabilities. Abiding by this principle means defining a threshold which separates acceptable from unacceptable structure. This notion is exactly captured by Rabin's definition of cut-point [17] which we will adopt and then generalize.

Section 2 gives an overview of previous work in this area. In section 3, we investigate probabilistic and weighted CFGs and show that using the definition of cut-point from [17] restricts probabilistic CFGs to generating the finite languages. This leads us to generalize the definition of cut-point which we then use to establish a hierarchy of weighted and probabilistic grammars in section 4. Then, in section 5 we prove a number of results that situate this weighted hierarchy relative to the classic Chomsky hierarchy showing that weighted CFGs extend CFGs in a different way than tree-adjoining grammars (TAGs) [12]. Then, in section 6 we offer a characterization of the languages of weighted CFGs that shows that weighted CFGs cannot identify any dependencies that CFGs cannot.

2 Previous Work

The first work that addressed the mapping of weighted languages into discrete languages was [17]. That work introduced the notion of a *cut-point* which formalizes the notion of a threshold. [17] used cut-points to prove that certain probabilistic finite automata when restricted by cut-points can generate languages beyond the regular languages. In fact, his proof can also be used to show that those automata can generate string languages that even Turing machines cannot. This is done via an abuse of the uncountability of the space of real numbers in $[0, 1]$. This unintuitive result is remedied by requiring certain distances between probabilities in the automata resulting in probabilistic finite automata that are weakly equivalent to non-probabilistic finite automata. This paper extends [17] by examining CFGs rather than finite automata, but also, by altering the way that probabilistic grammars are defined. This is due to the fact that [17] defines probabilistic automata that do not result in probabilistic languages and therefore his results are not directly applicable to modern uses of probabilistic CFGs.

Probabilistic automata and probabilistic CFGs that do generate probabilistic languages have also been investigated [7, 20]. [7] introduced a more powerful method of filtering the weighted languages of a weighted finite automata. They allow any subset of the real numbers to act as the set that discriminates between allowed weights and discarded weights. In particular, restricting the output of a weighted finite automaton to a single value generates languages such as the palindromes that finite automata cannot. [20] extends these results by proving that weighted CFGs filtered by the same mechanism generate the language $\{a^n b^n c^n | n > 0\}$ and weighted finite automata filtered this way can generate the MIX language. These results are difficult to apply to any modern weighted or probabilistic system because they disobey the principle that higher weights are preferable.

Along similar lines, [10] analyzes the generative power of probabilistic CFGs when only the maximal derivation for each sentence is retained. This naturally does not increase the weak generative capacity of the system, but they show that these types of probabilistic CFGs do generate tree languages that CFGs cannot. This work and our work take a different approach in that we explore the effects of discarding derivations below a certain threshold whereas they discard any derivation with weight lower than the highest weighted derivation.

In addition to this work on tree languages there has also been work on the strong generative capacity of non-weighted grammars in terms of the dependencies that they produce especially those more powerful than CFGs [8, 14].

3 Weighted and Probabilistic Grammars

We take the definitions of weighted and probabilistic grammars from [2] and [21] with the exception that we define weights of strings to be the maximum of all derivations over that string rather than the sum. This change is a deviation from the standard definition in the theoretical literature but it more closely corresponds to the usage of such grammars in the natural language parsing literature. In particular, it is very common to try to determine the correct parse tree for a given sentence from the set of all possible parse trees to try to disambiguate the syntax of the sentence. Defining the weights of the strings to be the *maximum* over all parse trees identifies the weight of the string with the likeliest parse tree. Identifying the weight of the string with the *sum* over all parse trees would be more useful if we were interested in all possible syntactic structures for a sentence, which in practice we are usually not.

Definition 1. *A* grammar *G is a system specifying a set of* derivations *$\mathcal{D}(G)$. A* non-weighted grammar *is a grammar where the set of derivations is a set of trees. A* weighted grammar *is a grammar where the derivations are pairs $\langle t, w\rangle$ where t is a tree and w belongs to $\mathbb{R}$. A* probabilistic grammar *is a weighted grammar G where*

$$\sum_{\langle t,w\rangle \in \mathcal{D}(G)} w = 1$$

Definition 2. *A* context-free grammar (CFG) *is a quadruple $\langle N, T, S, R\rangle$ where N is a finite set of non-terminals, T is a finite set of terminals, $S \in N$ is a start symbol and R is a finite set of rules of the form $r \to r_1 \dots r_k$ where $r \in N$ and $r_i \in N \cup T$ for $1 \leq i \leq k$. The derivations of a CFG are defined in the usual way and consist of trees with S as their root and where each node in the tree appears as the left side of a rule in R where its children are the right side of that rule and the leaves are terminals. The rules of a derivation t, is the set of occurrences of rules $\mathcal{R}(t)$ in the derivation.*

The weighted CFG (WCFG) *$\langle G, W\rangle$ is a CFG $G = \langle N, T, S, R\rangle$ and a weight function W where $W : R \to \mathbb{R}^+$ is a map from rules to positive real numbers. The derivations of a WCFG are pairs $\langle t, w\rangle$ where $t \in \mathcal{D}(G)$ and*

$$w = \prod_{r \in \mathcal{R}(t)} W(r)$$

A probabilistic CFG (PCFG) *$\langle G, W\rangle$ is a WCFG $\langle G, W\rangle$ where for $r \in N$,*

$$\sum_{r \to r_1 \dots r_k \in R} W(r \to r_1 \dots r_k) = 1$$

By these definitions, we see that CFGs are a class of non-weighted grammars and WCFGs are a class of weighted grammars. However, PCFGs are not necessarily probabilistic grammars due to some probability being assigned to infinite derivations. [21] investigates this issue more closely and defines consistent PCFGs to be PCFGs which are probabilistic grammars and provides conditions that characterize that class. However, we will be ignoring inconsistency here.

Our primary purpose here will be to investigate the power of probabilistic and weighted CFGs while obeying the principle that higher weights or probabilities are preferable. [10] introduced one such method for filtering derivations while obeying that principle but their method of choosing the maximum derivation for a sentence can only be used to disambiguate multiple derivations for a given string and the weak generative capacity cannot increase. To abide by the principle that higher probabilities are preferable, we must return to the definition of *cut-point* provided by [17] as a formal way to encode the intuition behind discarding low weights or probabilities:

Definition 3. *A* cut-point *is a value $c \in \mathbb{R}$. The set of derivations of a probabilistic or weighted grammar $\langle G, W \rangle$ with cut-point $c \in \mathbb{R}$ is defined as:*

$$\mathcal{D}(\langle G, W \rangle, c) = \{t | t \in \mathcal{D}(G) \textit{ and } c < W(t)\}$$

The intuition is that the cut-point provides the threshold below which structures are unacceptable. We should also note that by our definitions, probabilistic or weighted grammars with cut-points are non-weighted grammars. The remainder of this paper will be spent investigating the weak and strong generative capacity of PCFGs and WCFGs with cut-points. We need the following definitions:

Definition 4. *A* grammar formalism *is a system for restricting the range of grammars from all possible grammars. For example, CFGs, PCFGs and WCFGs among many others are grammar formalisms.*

Definition 5. *A grammar formalism F_1 is* strongly included *in a grammar formalism F_2 if for every grammar $G_1 \in F_1$ there exists a grammar formalism $G_2 \in F_2$ such that $\mathcal{D}(G_1) = \mathcal{D}(G_1)$.*

Definition 6. *Let T be a set of trees. For a tree $t \in T$, the string of t, $\mathcal{S}(t)$, is the string of terminals found at the leaves of t. Then, the string language of the set of trees T, $\mathcal{S}(T)$, is*

$$\mathcal{S}(T) = \{\mathcal{S}(t) | t \in T\}$$

A non-weighted grammar formalism F_1 is weakly included *in a non-weighted grammar formalism F_2 if for every $G_1 \in F_1$ there exists a grammar $G_2 \in F_2$ such that $\mathcal{S}(\mathcal{D}(G_1)) = \mathcal{S}(\mathcal{D}(G_2))$.*

The intuition is that strong inclusion characterizes the languages of derivations of a grammar whereas weak inclusion characterizes the languages of strings. The notions of weak and strong inclusion induce partial orders on the space of grammar formalisms. We say that two grammar formalisms are weakly (strongly) equivalent if they are weakly (strongly) included in each other and weakly (strongly) incomparable if neither is weakly (strongly) included in the other. Our first theorem will lead us to a more sophisticated definition of cut-point in the next section.

Definition 7. *A finite grammar (FG) G is any system for describing a finite set of trees $\mathcal{D}(G)$.*

Theorem 1. *PCFGs with positive cut-points are strongly included in FGs.*

Proof. Let $\langle G, W \rangle$ be a PCFG and c be a cut-point. Let $D \subseteq \mathcal{D}(G)$ be such that for $t \in D$, $W(t) > c$. But, according to the definition of PCFGs, $|D| * c \leq 1$. Thus, $|D| \leq 1/c$ and since $c > 0$, D is a finite set of derivations.

This theorem gives us the somewhat surprising result that using a threshold on the probabilities of a PCFG generates a finite language. We will proceed with some more basic theorems before remedying this situation in the next section.

Theorem 2. *CFGs are strongly included in PCFGs with cut-point 0 and WCFGs with both positive cut-points and cut-point 0.*

Proof. Let G be a CFG. Then, any PCFG or WCFG produces only derivations with weights above 0 and a cut-point of 0 does nothing to the strong generative capacity. Similarly, by defining a WCFG $\mathcal{D}(\langle G, W \rangle)$ where W assigns weight 1 to every rule and using a cut-point of $\frac{1}{2}$, every derivation will be above the cut-point.

Theorem 3. *PCFGs with cut-point 0 and WCFGs with cut-point 0 are strongly included in CFGs.*

Proof. For a WCFG or PCFG $\langle G, W \rangle$ with cut-point 0, the CFG G generates exactly the derivations of the weighted or probabilistic version since the cut-point of 0 does nothing.

4 The Weighted Hierarchy

In the preceding section, we proved that PCFGs with cut-points generate only finite structure. This result is predicated on the fact that as a sentence grows, it necessarily contains more rules which ultimately decrease the probability given to the derivation. One way of remedying this situation is to allow for the cut-point to vary depending in some way on the length of the sentence or derivation. We define a generalization of cut-point that allows the cut-point to change with the length of the sentence.

Definition 8. *A* normalized cut-point *is a function $c_n : \mathbb{N} \rightarrow \mathbb{R}^+$ mapping from the natural numbers to the positive real numbers. The set of derivations of a probabilistic or weighted grammar $\langle G, W \rangle$ with* string normalized cut-point *c_n is defined as:*

$$\mathcal{D}(\langle G, W \rangle, c_n) = \{t | t \in \mathcal{D}(G) \wedge c_n(|\mathcal{S}(t)|) < W(t)\}$$

The set of derivations of a probabilistic or weighted grammar $\langle G, W \rangle$ with rule normalized cut-point *c_n is defined as*

$$\mathcal{D}(\langle G, W \rangle, c_n) = \{t | t \in \mathcal{D}(G) \wedge c_n(|\mathcal{R}(t)|) < W(t)\}$$

These normalized cut-points achieve the intended goal of allowing arbitrarily long sentences into the languages of PCFGs filtered by cut-points but their generality is too unwieldy for our purposes. Therefore, we define the following subclass requiring the cut-point to change by a constant factor as the sentence grows.

Definition 9. *A* geometric normalized cut-point (or geo-norm cut-point) *is a function* $c_g : \mathbb{N} \to \mathbb{R}^+$ *with common ratio* $0 < g < 1$ *defined as* $c_g(n) = g^n$.

Geo-norm cut-points are intended to simplify normalized cut-points and the precise differences in the power between the two is as yet unknown. Next, we introduce lexicalized CFGs [13] to give insight into the generative power of CFGs supplied with lexical information as in [6] and to allow us to prove results concerning string normalized cut-points.

Definition 10. *A* lexicalized CFG (LCFG) *is a CFG* $\langle N, T, S, R\rangle$ *where for each* $r \to r_1 \dots r_k \in R$ *there exists an* $1 \le i \le k$ *such that* $r_i \in T$ *and for* $j \ne i$, $r_j \in N$. *That is, each rule has exactly one terminal on the right side. Probabilistic and weighted LCFGs are defined in the obvious way.*

First, we will relate our definitions of string normalized cut-points and rule normalized cut-points via a relationship between probabilistic LCFGs and CFGs.

Theorem 4. *PLCFGs with string geo-norm cut-points are a special case of PCFGs with rule geo-norm cut-points.*

Proof. By the definition of LCFGs, in any derivation of an LCFG there is exactly one rule per non-terminal symbol. Thus, in this case, string normalized cut-points are identical to rule normalized cut-points. Furthermore, there certainly exist PCFGs that generate derivations that no PLCFG can generate simply by having a rule that is not lexicalized as part of a derivation with a high weight.

We now proceed with theorems proving inclusions of PCFGs in WCFGs.

Theorem 5. *PLCFGs with string geo-norm cut-points are strongly included in WLCFGs with cut-points.*

Proof. Let $\langle G, W\rangle$ be a PLCFG and let c_g be a geo-norm cut-point. We define a WLCFG $\langle G, W'\rangle$ where $W'(r) = W(r)/g$ for $r \in R$. Then, define a cut-point $c = 1$.

Let $t \in \mathcal{D}(G)$. By definition, t is generated by $\langle G, W\rangle$ with geo-norm cut-point c_g if and only if $W(t) > g^n$ where $n = |\mathcal{S}(t)|$. Furthermore, t has exactly n rules, since it is a derivation of an LCFG. Then,

$$W'(t) = \prod_{r \in \mathcal{R}(t)} W'(r) = \prod_{r \in \mathcal{R}(t)} W(r)/g$$

$$= \frac{1}{g^n} * \prod_{r \in \mathcal{R}(t)} W(r) = \frac{W(t)}{g^n}$$

Thus, $W'(t) = W(t)/g^n$. Then, $W(t) > g^n$ if and only if $W(t)/g^n > 1$ if and only if $W'(t) > 1$.

Theorem 6. *PCFGs with rule geo-norm cut-points are strongly included in WCFGs with cut-points.*

Proof. This proof is essentially identical to that of theorem 5 except that $|\mathcal{S}(t)|$ is replaced by $|\mathcal{R}(t)|$.

These results yield the probabilistic and weighted portion of the grammar hierarchy shown in figures 1 and 2.

5 The Weighted Hierarchy and the Chomsky Hierarchy

In the previous section, we proved a number of theorems showing the relationships between some probabilistic and weighted grammars in relation to CFGs. In this section, we will prove some theorems relating that hierarchy with the Chomsky hierarchy[1].

We will prove that the weighted and probabilistic grammars discussed in the previous section are more powerful than CFGs and contrast the weighted hierarchy with the Chomsky hierarchy. To do this we must define tree-adjoining grammar (TAG) [12], a grammar whose languages include the languages of CFGs.

Definition 11. *A* tree-adjoining grammar (TAG) *is a septuple $\langle N, T, S, I, A, SA, OA\rangle$ where N is a finite set of non-terminals, T is a finite set of terminals, $S \in N$ is a start symbol, I is a set of* initial trees*, A is a set of* auxiliary trees*, SA is a mapping from nodes in trees in $I \cup A$ to sets of trees in $I \cup A$ and OA is a mapping from nodes in trees in $I \cup A$ to Booleans.*

An initial tree in I is a tree whose non-leaf nodes are non-terminals from T and whose leaf nodes are from the set $N \cup T$. An auxiliary tree in A is a tree whose non-leaf nodes are non-terminals from T and whose leaf nodes are from the set $N \cup T$ and exactly one non-terminal leaf node is identified as the foot node *which is the same symbol as the root.*

The derivations of a TAG[2] consist of trees that are built through a series of adjunctions *and* substitutions*. The process begins with a tree in I with root S. Adjunction is an operation on a derivation tree T where a tree α from A is* adjoined *at a node $e \in T$ if the root of α shares the same symbol as e. If so, then the subtree rooted at e in T is deleted from T and inserted at the foot node of α and the root of the resulting tree is inserted at the old location of e in T. Substitution is an operation on a derivation tree T where a tree ι from I is* substituted *at a node $e \in T$ if the root of ι shares a symbol with e and e is a leaf in T. If so, then ι is inserted at e in T.*

SA assigns sets of trees in $I \cup A$ to each node e in each tree of $I \cup A$ indicating which trees are allowed to be adjoined at e. If SA maps a node to the empty set, then adjunction is disallowed at that node. OA assigns a Boolean to each node e in each tree of $I \cup A$ indicating whether adjunction is obligatory at that node.

[1] The original definition of the Chomsky hierarchy did not include the tree-adjoining languages, but it is natural to include them between the context-free languages and the context-sensitive languages.

[2] The usual distinction between derivation and derived trees in TAG is not necessary here.

TAGs are known to have generative power beyond the CFGs, most specifically in being able to generate the string languages $\{a^n b^n c^n | n > 0\}$, $\{a^n b^n c^n d^n | n > 0\}$ and $\{a^n b^m c^n d^m | n > 0, m > 0\}$. TAGs are important because they generate the necessary structures for representing the non-context-free cross-serial dependencies found in Swiss German [18] which require the generation of the string language $\{a^n b^m c^n d^m | n > 0, m > 0\}$. We will now proceed with some theorems outlining the relationship between CFGs and TAGs and weighted and probabilistic CFGs. The complete hierarchy based on these theorems is depicted in figures 1 and 2.

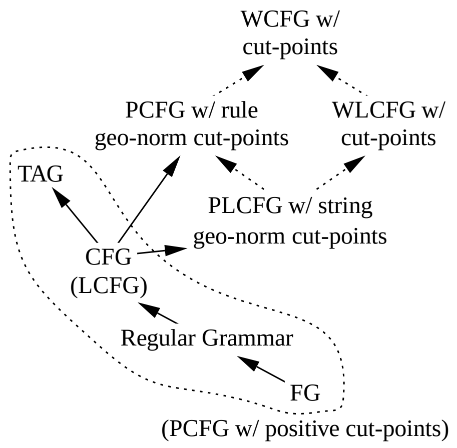
WCFG w/ cut-points
PCFG w/ rule geo-norm cut-points
WLCFG w/ cut-points
TAG
PLCFG w/ string geo-norm cut-points
CFG (LCFG)
Regular Grammar
FG
(PCFG w/ positive cut-points)

Fig. 1. The weak grammar hierarchy. Parentheses indicate equivalent grammar formalisms and dashed edges indicate possibly improper inclusions. The dashed bubble is the Chomsky hierarchy.

Theorem 7. *LCFGs are strongly included in PLCFGs with string geo-norm cut-points.*

Proof. Let $G = \langle N, T, S, R \rangle$ be an LCFG. For $l \in N$, let $R_n \subseteq R$ be the set of rules with l on the left side. Then, for $r \in R_n$, let $W'(r) = 1/|R|$. Then, let c_g be a string geo-norm cut-point where $g = 1/|R|$.

Then, for each rule in $r \in R$, $W'(r) > g$. Thus, for each derivation $t \in \mathcal{D}(G)$, $W'(t) > g^{|\mathcal{R}(t)|}$. But this implies that the derivation set of the WLCFG $\langle G, W \rangle$ with string geo-norm cut-point c_g is $\mathcal{D}(G)$. Furthermore, for each $l \in N$ the set of rules with l on their left side sums to 1 implying that $\langle G, W' \rangle$ is a PLCFG.

Corollary 1. *CFGs are strongly included in PCFGs with rule geo-norm cut-points.*

Proof. By a nearly identical proof to that of Theorem 7.

Corollary 2. *LCFGs are strongly included in WLCFGs with cut-points and CFGs are strongly included in WCFGs with cut-points.*

Proof. By the transitivity of inclusion and the inclusion results of the preceding section.

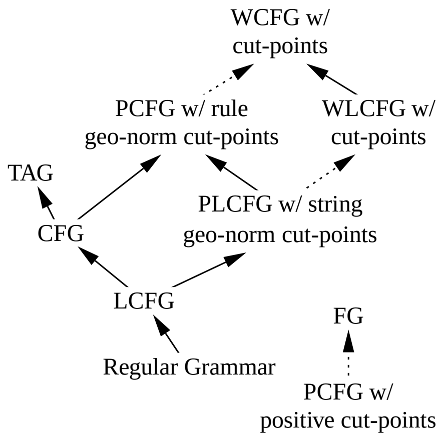
WCFG w/ cut-points
PCFG w/ rule geo-norm cut-points
WLCFG w/ cut-points
TAG
PLCFG w/ string geo-norm cut-points
CFG
LCFG
FG
Regular Grammar
PCFG w/ positive cut-points

Fig. 2. The strong grammar hierarchy. Dashed edges indicate inclusions that may be improper.

Proposition 1. *There is a PLCFG with geo-norm cut-points which generates the string language* $\{a^n b^n c^m | n \geq m-1 \textit{ and } m \geq 1\}$.

Proof. We define the following PLCFG $\langle G, W\rangle$ whose start symbol is S:

$$\begin{array}{lll} (1) & S \rightarrow aBC & 1 \\ (2) & A \rightarrow aB & 1 \\ (3) & B \rightarrow Ab & \frac{1}{2} \\ (4) & B \rightarrow b & \frac{1}{2} \\ (5) & C \rightarrow cC & \frac{1}{4} \\ (6) & C \rightarrow c & \frac{3}{4} \end{array}$$

and the geo-norm cut-point c_g where $g = \frac{1}{2}$.

First, note that any string of any derivation in $\mathcal{D}(G)$ is in $a^n b^n c^m$ for some $n, m \geq 1$. Let $t \in \mathcal{D}(G)$. Let k_a be the number of as in t and let k_c be the number of cs. Then, $|\mathcal{S}(t)| = 2 * k_a + k_c$ and

$$W(t) = \frac{1}{2^{k_a}} * \frac{1}{4^{k_c - 1}} * \frac{3}{4} = 3 * \frac{1}{2^{k_a + 2k_c}}$$

Then,

$$c_g(|\mathcal{S}(t)|) = \frac{1}{2^{2k_a + k_c}} = \frac{1}{2^{2k_a + k_c}}$$

Finally, since $2^2 > 3 > 2^1$, $W(t) > c_g(|\mathcal{S}(t)|)$ if and only if $2k_a + k_c \geq k_a + 2k_c - 1$ if and only if $k_a \geq k_c - 1$. Thus, $\mathcal{D}(\langle G, W\rangle, c_g)$ is exactly those derivations whose strings are $a^n b^n c^m$ where $n \geq m-1$ and $m \geq 1$.

To prove that this language is not generated by any CFG we need Ogden's lemma [9, 15].

Lemma 1 (Ogden's Lemma). *Let $L = \mathcal{S}(\mathcal{D}(G))$ for a CFG G. Then there is a constant n such that if $z \in L$ and we* mark *n or more positions of z then we can write $z = uvwxy$ such that:*

1. *v and x have at least one marked position between them*
2. *vwx has at most n marked positions*
3. *$uv^iwx^iy \in L$ for all $i \geq 0$*

Proposition 2. *There is no CFG that generates the string language $L = \{a^nb^nc^m | n \geq m - 1$ and $m \geq 1\}$.*

Proof. Assume on the contrary that L is generated by some context-free grammar. Let n be the constant given by Ogden's lemma. We consider the string $z = a^nb^nc^{n+1}$ in which all c positions are marked. Then there exist u, v, w, x and y such that $z = uvwxy$.

One of v and x must contain a c, by the first condition. v cannot contain an a and a b because then uv^2wx^2y would have a b preceding an a. Also, v cannot contain a b but no a, because then uv^2wx^2y would not have an equal number of as and bs. Similarly, v cannot contain only cs because then uv^2wx^2y would have at least $n + 2$ cs with only n as. Also, v cannot contain only as, since if it does then x must contain an equal number of bs and at least one c in which case uv^2wx^2y would have a c preceding a b. Finally, v cannot be empty because then x must either contain only cs, in which case uv^2wx^2y would contain too many cs or x must contain bs and cs in which case uv^2wx^2y would contain a c before a b.

Therefore, no such partitioning of z exists and, by Ogden's lemma, L is not a context-free language.

Theorem 8. *PLCFGs with string geo-norm cut-points are not weakly included in CFGs.*

Corollary 3. *PCFGs with rule geo-norm cut-points, WLCFGs with cut-points and WCFGs with cut-points are not weakly included in CFGs.*

These results establish that WCFGs with cut-points and PCFGs with normalized cut-points have generative power beyond CFGs. However, when compared with TAGs, the next step in the Chomsky hierarchy, we find that they are incomparable. Before we can proceed though, we need to provide a variant of Ogden's lemma that includes details about the derivations for the strings.

Lemma 2. *Let $L = \mathcal{S}(\mathcal{D}(G))$ for a CFG G. Then there is a constant n such that if $z \in L$ and we* mark *n or more positions of z then we can write $z = uvwxy$ such that:*

1. *v and x have at least one marked position between them*
2. *vwx has at most n marked positions*
3. *$uv^iwx^iy \in L$ for all $i \geq 0$*

Furthermore, for all derivations $t \in \mathcal{D}(G)$ such that $\mathcal{S}(t) = uv^nwx^ny$, t has a subtree t' whose root is a non-terminal A dominating only those terminals in vwx and t has another subtree t'' whose root is also A dominating the terminals in w.

Proof. This can be easily proven in an identical manner to Ogden's original proof except that care must be paid to the rules in the trees.

Theorem 9. *No WCFG with cut-points generates the string language* $L = \{a^n b^n c^n | n > 0\}$.

Proof. Assume on the contrary, that a WCFG $\langle G, W \rangle$ and a cut-point k exists such that $\mathcal{S}(\mathcal{D}(\langle G, W \rangle, k)) = \{a^n b^n c^n | n > 0\}$. Let n be the constant in lemma 2 , let $t \in \mathcal{D}(G)$ be the derivation with the highest weight such that $\mathcal{S}(t) = a^n b^n c^n$ for some n, let $z = a^n b^n c^n$, let all the positions be marked [3] and let $z = uvwxy$.

Then, by lemma 2, t has a subtree t' whose root is a non-terminal A dominating only those terminals in vwx and t has another subtree t' whose root is also A dominating only those terminals in w.

Then, since vwx has at most n marked positions, it must contain either all as, all bs, all cs, some as followed by bs or some bs followed by cs. Furthermore, one of v or x must contain a terminal. Let R be the set of rules in $t - t'$ and let $r_w = \prod_{r \in R} W(r)$ and let $z_w = W(t)$.

It must be the case that $z_w > k$, since otherwise $a^n b^n c^n \notin L$. If $r_w \geq 1$, then uv^2wx^2z does not contain equal numbers of as, bs and cs, but has weight $r_w * z_w > k$. If $r_w < 1$, then uwz does not contain equal numbers of as, bs and cs, but has weight $z_w / r_w > k$.

Corollary 4. *No WCFG with cut-points generates the string language* $\{a^n b^n c^n d^n | n > 0\}$.

Proof. The proof is essentially identical to the proof for the language $\{a^n b^n c^n | n > 0\}$.

Corollary 5. *No WCFG with cut-points generates the string language* $\{a^n b^m c^n d^m | n > 0, m > 0\}$.

Proof. The proof is only a slight variation on the proof for the language $\{a^n b^n c^n | n > 0\}$.

Corollary 6. *No PLCFG with string geo-norm cut-points, PCFG with rule geo-norm cut-points nor WLCFG with cut-points generates the string languages* $\{a^n b^n c^n | n > 0\}$, $\{a^n b^n c^n d^n | n > 0\}$ *or* $\{a^n b^m c^n d^m | n > 0, m > 0\}$.

Proposition 3. *There is a PLCFG with geo-norm cut-points that generates the string language* $\{a^n b^n c^m d^m e^o | n \geq m + o - 4$ *and* $m, o \geq 1\}$.

Proof. We define the following PLCFG $\langle G, W \rangle$ whose start symbol is S:

$$\begin{array}{llll}
(1)\ S \to aBCE & 1 & (6)\ D \to Cd & \frac{1}{8} \\
(2)\ A \to aB & 1 & (7)\ D \to d & \frac{7}{8} \\
(3)\ B \to Ab & \frac{1}{2} & (8)\ E \to Ee & \frac{1}{4} \\
(4)\ B \to b & \frac{1}{2} & (9)\ E \to e & \frac{3}{4} \\
(5)\ C \to cD & 1 & &
\end{array}$$

[3] Marking all positions is equivalent to the pumping lemma [9].

and the geo-norm cut-point c_g where $g = \frac{1}{2}$.

Then, the $\mathcal{D}(\langle G, W\rangle, c_g)$ consists of only those strings of the form $a^n b^n c^m d^m e^o$ but by the same reasoning as in Proposition 1, it must be the case that $n \geq m + o - 4$.

To proceed with our final theorem concerning the relationship between TAGs and CFGs, we need to state Ogden's lemma for the tree-adjoining languages given by [16]:

Lemma 3. *Let $L = \mathcal{S}(G)$ for a TAG G. Then there is a constant n such that if $z \in L$ and we mark n or more positions of z then we can write $z = qrstuvwxy$ such that:*

1. *One of the following holds:*
   - *q, r and s each have at least one marked position*
   - *s, t and u each have at least one marked position*
   - *u, v and w each have at least one marked position*
   - *w, x and y each have at least one marked position*
2. *tuv has at most n marked positions*
3. *$qr^i st^i uv^i wx^i y \in L$ for all $i \geq 0$*

Proposition 4. *There is no TAG that generates the string language $\{a^n b^n c^m d^m e^o | n \geq m + o - 4$ and $m, o \geq 1\}$.*

Proof. The proof is quite similar to the proof of proposition 2 except that we must use lemma 3 rather than Ogden's lemma.

Assume on the contrary that L is generated by some TAG. Let k be the constant given by lemma 3. We consider the string $z = a^n b^n c^k d^k e^k$ for $n = 2k - 4$ in which all ds are marked. Then there exist $q, r, s, t, u, v, w, x, y$ such that $z = qrstuvwxy$.

First, we consider the case where one of r, t, v or x contains two or more symbols from the set $\{a, b, c, d, e\}$. In this case, the string $qr^2 st^2 uv^2 wx^2 y$ must contain a pair of symbols that are out of order which results in a string that is not in L.

Therefore, each of r, t, v and x contain zero or one symbols from the set of terminals. By the pigeonhole principle, one of the symbols from $\{a, b, c, d, e\}$ must not appear in any of r, t, v or x.

That missing symbol cannot be d since only the ds are marked and by the first clause of lemma 3, one of r, t, v or x must contain a marked symbol. If that missing symbol was c, then the string $qr^2 st^2 uv^2 wx^2 y$ would not contain an equal number of cs and ds.

If the missing symbol is a, then b must also be missing since otherwise $qr^2 st^2 uv^2 wx^2 y$ would not contain an equal number of as and bs. However, if both are missing then $qr^2 st^2 uv^2 wx^2 y$ would contain at least $k + 1$ ds and at least k es but only $2k - 4$ as which would falsify the inequality of $n \geq m - o + 4$ required of all strings in L.

Therefore, the missing symbol must be e, all of which must occur in the substring y. By analysis of the proof of lemma 3, we know that must be a point in the derivation tree of z such that its descendants make up the substring y. Since y contains k es, we can mark them differently than our previous markings and apply lemma 3 again. By lemma 3, L must contain strings that have arbitrary numbers of es but only k as. This is a contradiction and proves the result.

Theorem 10. *PLCFGs with string geo-norm cut-points are weakly incomparable to TAGs.*

Corollary 7. *PCFGs with rule geo-norm cut-points, WLCFGs with cut-points and WCFGs with cut-points are weakly incomparable to TAGs.*

6 A Characterization of Weighted CFGs with Cut-Points

The preceding section introduced languages that the weighted and probabilistic CFGs generate but that neither CFGs nor TAGs can. In this section, we provide a characterization of WCFGs with cut-points. It is currently an open question as to how to characterize P(L)CFGs with geo-norm cut-points or if the strong inclusion in WCFGs is even proper.

This characterization is relative to the languages of CFGs via a linear inequality over rules. Because the languages of CFGs have not been precisely characterized, this characterization is itself not precise but it is illuminating.

Definition 12. *A* linear product inequality CFG (LPICFG) *is a pair $\langle G, L\rangle$ where L is an inequality over the rules of $G = \langle N, T, S, R\rangle$ and $\mathbb{R}$ of the form:*

$$c_1^{r_1} * c_2^{r_2} * \ldots * c_k^{r_k} > c$$

where $k = |R|$ and $c_i \in \mathbb{R}$ for $1 \leq i \leq k$. The derivations of $\langle G, L\rangle$ are exactly those derivations $t \in \mathcal{D}(G)$ where the variables r_i in the inequality are instantiated as the counts of the corresponding rules in t and the resulting inequality is true.

Theorem 11. *LPICFGs are strongly equivalent to WCFGs with cut-points.*

Proof. The correspondence between the two grammars is straightforward where the value c in the LPICFG corresponds to the cut-point in the WCFG and the weights c_i for $1 \leq i \leq k$ in the LPICFG correspond to the weights the weight function assigns to the rules. Then, the derivations that are eliminated by the cut-point are exactly those that are eliminated by having a false inequality.

The correspondence between LPICFGs and WCFGs with cut-points is straightforward but it helps to illuminate the restrictions that underlie the weighted and probabilistic CFGs. In particular, it means that cut-points can ensure a kind of dependency between parts of derivations that are arbitrarily distant but that dependency can only be a linear relationship and that dependency can only discriminate between parts of derivations by preferring one over another by some constant factor. Perhaps most importantly, WCFGs with cut-points cannot require that a certain structure in a derivation exist if and only if another structure exists except via the same mechanism used by a CFG. This prohibits weights from expressing the kind of links that dependencies express in discrete grammars.

7 Conclusion

We have investigated the effect that always preferring structures with higher weight to structures with lower weights has on the discrete languages of probabilistic and weighted CFGs. The result is that cut-points increase both the strong and weak generative power not only beyond CFGs but also beyond TAGs. These results show that the use of probabilities and weights, even in the form that they are typically used in modern parsing systems can add to the generative capacity of the practical systems in a way that moving up the Chomsky hierarchy to TAGs cannot. However, we also established that this increase in power is orthogonal to moving up the Chomsky hierarchy because the weights cannot establish the kinds of dependencies that TAGs can.

There remain a number of open problems in the domain of probabilistic and weighted CFGs with cut-points, which will require future research. In particular, whether the inclusion of PCFGs with normalized cut-points in WCFGs with cut-points is proper can help to determine whether it is important to continue to distinguish between weights and probabilities when abiding by the principle that higher weights are preferable.

The results that we have presented in this paper give us an understanding of the kinds of structures that probabilities can be used to generate in modern probabilistic parsers. In particular, we have shown that the advantage of using probabilities or weights to alter the generative capacity of a grammar formalism allows for arbitrarily distant grammatical structures to affect each other in the limited way described in the preceding sections. Whether this is linguistically useful requires more research.

In addition, the notions explored in this paper can be extended to other grammar formalisms such as regular grammars, finite automata, TAGs and combinatory categorial grammars. Such research can give us similar insights into other systems used in natural language processing that will help when choosing the grammar to use in a natural language system.

References

1. Abney, S., McAllester, D., Pereira, F.: Relating Probabilistic Grammars and Automata. In: Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics. p. 542–549. College Park, MD (1999)
2. Booth, T.L., Thompson, R.A.: Applying Probability Measures to Abstract Languages. IEEE Transactions on Computers 100(22), 442–450 (1973)
3. Chi, Z.: Statistical Properties of Probabilistic Context-free Grammars. Computational Linguistics 25(1), 131–160 (1999)
4. Chiang, D.: Hierarchical Phrase-based Translation. Computational Linguistics 33(2), 201–228 (2007)
5. Clark, S., Curran, J.R.: Wide-coverage Efficient Statistical Parsing with CCG and Log-linear Models. Computational Linguistics 33(4), 493–552 (2007)
6. Collins, M.: Head-driven Statistical Models for Natural Language Parsing. Computational Linguistics 29(4), 589–637 (2003)
7. Cortes, C., Mohri, M.: Context-free Recognition with Weighted Automata. Grammars 3(2), 133–150 (2000)
8. Hockenmaier, J., Young, P.: Non-local Scrambling: The Equivalence of TAG and CCG Revisited. In: Proceedings of the Ninth International Workshop on Tree Adjoining Grammars and Related Formalisms. p. 41–48. Tübingen, Germany (2008)
9. Hopcroft, J.E., Motwani, R., Ullman, J.D.: Introduction to Automata Theory, Languages, and Computation. Addison-wesley (2006)
10. Infante-Lopez, G., De Rijke, M.: A Note on the Wxpressive Power of Probabilistic Context free Grammars. Journal of Logic, Language and Information 15(3), 219–231 (2006)
11. Jelinek, F.: Statistical Methods for Speech Recognition. MIT press (1999)
12. Joshi, A.K., Schabes, Y.: Tree-adjoining grammars. Handbook of Formal Languages, Beyond Words 3, 69–123 (1997)

13. Kallmeyer, L.: Comparing Lexicalized Grammar Formalisms in an Empirically Adequate Way: The Notion of Generative Attachment Capacity. In: Proceedings of the International Conference on Linguistic Evidence. p. 154–156 (2006)
14. Kuhlmann, M.: Dependency structures and lexicalized grammars. Ph.D. thesis, Saarland University (2007)
15. Ogden, W.: A Helpful Result for Proving Inherent Ambiguity. Theory of Computing Systems 2(3), 191–194 (1968)
16. Palis, M.A., Shende, S.M.: Pumping Lemmas for the Control Language Hierarchy. Theory of Computing Systems 28(3), 199–213 (1995)
17. Rabin, M.O.: Probabilistic Automata. Information and Control 6, 230–245 (1963)
18. Shieber, S.M.: Evidence Against the Context-freeness of Natural Language. Linguistics and Philosophy 8(3), 333–343 (1985)
19. Smith, N.A., Johnson, M.: Weighted and Probabilistic Context-free Grammars are Equally Expressive. Computational Linguistics 33(4), 477–491 (2007)
20. Sogaard, A.: On the Weak Generative Capacity of Weighted Context-free Grammars. In: Proceedings of the 22nd International Conference on Computational Linguistics. p. 99–102 (2008)
21. Wetherell, C.S.: Probabilistic Languages: a Review and Some Open Questions. ACM Computing Surveys (CSUR) 12(4), 361–379 (1980)