Polynomial Learnability and Locality of Formal Grammars

Naoki Abe*
Department of Computer and Information Science,
University of Pennsylvania, Philadelphia, PA19104.

ABSTRACT

We apply a complexity theoretic notion of feasible learnability called "polynomial learnability" to the evaluation of grammatical formalisms for linguistic description. We show that a novel, nontrivial constraint on the degree of "locality" of grammars allows not only context free languages but also a rich class of mildy context sensitive languages to be polynomially learnable. We discuss possible implications of this result to the theory of natural language acquisition.

1 Introduction

Much of the formal modeling of natural language acquisition has been within the classic paradigm of "identification in the limit from positive examples" proposed by Gold [7]. A relatively restricted class of formal languages has been shown to be unlearnable in this sense, and the problem of learning formal grammars has long been considered intractable.[1] The following two controversial aspects of this paradigm, however, leave the implications of these negative results to the computational theory of language acquisition inconclusive. First, it places a very high demand on the accuracy of the learning that takes place - the hypothesized language must be exactly equal to the target language for it to be considered "correct". Second, it places a very permissive demand on the time and amount of data that may be required for the learning - all that is required of the learner is that it converge to the correct language in the limit.[2]

Of the many alternative paradigms of learning proposed, the notion of "polynomial learnability" recently formulated by Blumer et al. [6] is of particular interest because it addresses both of these problems in a unified way. This paradigm relaxes the criterion for learning by ruling a class of languages to be learnable, if each language in the class can be approximated, given only positive and negative examples,[3] with a desired degree of accuracy and with a desired degree of robustness (probability), but puts a higher demand on the complexity by requiring that the learner converge in time polynomial in these parameters (of accuracy and robustness) as well as the size (complexity) of the language being learned.

In this paper, we apply the criterion of polynomial learnability to subclasses of formal grammars that are of considerable linguistic interest. Specifically, we present a novel, nontrivial constraint on grammars called "k-locality", which enables context free grammars and indeed a rich class of mildly context sensitive grammars to be feasibly learnable. Importantly the constraint of k-locality is a nontrivial one because each k-local subclass is an exponential class [4] containing infinitely many infinite languages. To the best of the author's knowledge, "k-locality" is the first nontrivial constraint on grammars, which has been shown to allow a rich class of grammars of considerable linguistic interest to be polynomially learnable. We finally mention some recent negative result in this paradigm, and discuss possible implications of its contrast with the learnability of k-local classes.

2 Polynomial Learnability

"Polynomial learnability" is a complexity theoretic notion of feasible learnability recently formulated by Blumer et al. ([6]). This notion generalizes Valiant's theory of learnable boolean concepts [15], [14] to infinite objects such as formal languages. In this paradigm, the languages are presented via infinite sequences of pos-

*Supported by an IBM graduate fellowship. The author gratefully acknowledges his advisor, Scott Weinstein, for his guidance and encouragement throughout this research.

[1] Some interesting learnable subclasses of regular languages have been discovered and studied by Angluin [3].

[2] For a comprehensive survey of various paradigms related to "identification in the limit" that have been proposed to address the first issue, see Osherson, Stob and Weinstein [12]. As for the latter issue, Angluin ([5], [4]) investigates the feasible learnability of formal languages with the use of powerful oracles such as "MEMBERSHIP" and "EQUIVALENCE".

[3] We hold no particular stance on the the validity of the claim that children make no use of negative examples. We do, however, maintain that the investigation of learnability of grammars from both positive and negative examples is a worthwhile endeavour for at least two reasons: First, it has a potential application for the design of natural language systems that learn. Second, it is possible that children do make use of *indirect* negative information.

[4] A class of grammars $\mathcal{G}$ is an *exponential class* if each subclass of $\mathcal{G}$ with bounded size contains exponentially (in that size) many grammars.

itive and negative examples[5] drawn with an arbitrary but *time invariant* distribution over the entire space, that is in our case, $\Sigma_T{}^*$. Learners are to hypothesize a grammar at each finite initial segment of such a sequence, in other words, they are functions from finite sequences of members of $\Sigma_T{}^* \times \{0,1\}$ to grammars.[6] The criterion for learning is a complexity theoretic, approximate, and probabilistic one. A learner is said to learn if it can, with an arbitrarily high probability $(1-\delta)$, converge to an arbitrarily accurate (within ϵ) grammar in a feasible number of examples. "A feasible number of examples" means, more precisely, polynomial in the size of the grammar it is learning and the degrees of probability and accuracy that it achieves - δ^{-1} and ϵ^{-1}. "Accurate within ϵ" means, more precisely, that the output grammar can predict, with error probability ϵ, future events (examples) drawn from the *same distribution* on which it has been presented examples for learning. We now formally state this criterion.[7]

Definition 2.1 (Polynomial Learnability) *A collection of languages $\mathcal{L}$ with an associated 'size' function with respect to some fixed representation mechanism is polynomially learnable if and only if:*[8]

$$
\begin{array}{l}
\exists f \in \mathcal{F} \\
\quad \exists\, q\text{: } \textit{a polynomial function} \\
\quad\quad \forall L_1 \in \mathcal{L} \\
\quad\quad\quad \forall P\text{: } \textit{a probability measure on } \Sigma_T{}^* \\
\quad\quad\quad\quad \forall \epsilon, \delta > 0 \\
\quad\quad\quad\quad\quad \forall m \geq q(\epsilon^{-1}, \delta^{-1}, size(L_1)) \\
\quad\quad\quad\quad\quad [P^*(\{t \in \mathcal{EX}(L_1) \mid P(L(f(\bar{t}_m))\triangle L_1) \leq \epsilon\}) \\
\quad\quad\quad\quad\quad\quad \geq 1-\delta \\
\quad\quad\quad\quad\quad \textit{and } f \textit{ is computable in time polynomial} \\
\quad\quad\quad\quad\quad \textit{in the length of input}]
\end{array}
$$

If in addition all of f's output grammars on example sequences for languages in $\mathcal{L}$ belong to $\mathcal{G}$, then we say that $\mathcal{L}$ is polynomially learnable by $\mathcal{G}$.

Suppose we take the sequence of the hypotheses (grammars) made by a learner on successive initial finite sequences of examples, and plot the "errors" of those grammars with respect to the language being learned. The two learnability criteria, "identification in the limit" and "polynomial learnability", require different kinds of convergence behavior of such a sequence, as is illustrated in Figure 1.

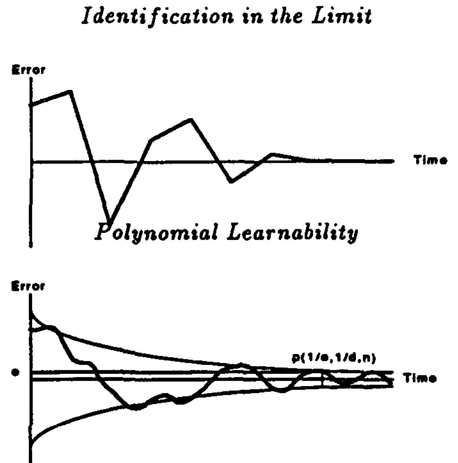

Figure 1: Convergence behaviour

Blumer et al. ([6]) shows an interesting connection between polynomial learnability and data compression. The connection is one way: If there exists a polynomial time algorithm which reliably "compresses" any sample of any language in a given collection to a provably small consistent grammar for it, then such an alogoritm polynomially learns that collection. We state this theorem in a slightly weaker form.

Definition 2.2 *Let $\mathcal{L}$ be a language collection with an associated size function "size", and for each n let $\mathcal{L}_n = \{L \in \mathcal{L} \mid size(L) \leq n\}$. Then $\mathcal{A}$ is an Occam algorithm for $\mathcal{L}$ with range size*[9] *$f(m,n)$ if and only if:*

$$
\begin{array}{l}
\forall n \in N \\
\quad \forall L \in \mathcal{L}_n \\
\quad\quad \forall t \in \mathcal{EX}(L) \\
\quad\quad\quad \forall m \in N \\
\quad\quad\quad\quad [\mathcal{A}(\bar{t}_m) \textit{ is consistent with}^{10} rng(\bar{t}_m) \\
\quad\quad\quad\quad \textit{and } \mathcal{A}(\bar{t}_m) \in \mathcal{L}_{f(n,m)} \\
\quad\quad\quad\quad \textit{and } \mathcal{A} \textit{ runs in time polynomial in } \mid \bar{t}_m \mid]
\end{array}
$$

Theorem 2.1 (Blumer et al.) *If $\mathcal{A}$ is an Occam algorithm for $\mathcal{L}$ with range size $f(n,m) = O(n^k m^\alpha)$ for some $k \geq 1$, $0 \leq \alpha < 1$ (i.e. less than linear in sample size and polynomial in complexity of language), then $\mathcal{A}$ polynomially learns $\mathcal{L}$.*

[5]We let $\mathcal{EX}(L)$ denote the set of infinite sequences which contain only positive and negative examples for L, so indicated.

[6]We let $\mathcal{F}$ denote the set of all such functions.

[7]The following presentation uses concepts and notation of formal learning theory, cf. [12]

[8]Note the following notation. The inital segment of a sequence t *up to* the n-th element is denoted by $\bar{t}_n$. L denotes some fixed mapping from grammars to languages: If G is a grammar, L(G) denotes the language generated by it. If L_1 is a language, $size(L_1)$ denotes the size of a minimal grammar for L_1. $A\triangle B$ denotes the symmetric difference, i.e. $(A-B)\cup(B-A)$. Finally, if P is a probability measure on $\Sigma_T{}^*$, then P* is the cannonical product extension of P.

[9]In [6], the notion of "range dimension" is used in place of "range size", which is the Vapnik-Chervonenkis dimension of the hypothesis class. Here, we use the fact that the dimension of a hypothesis class with a size bound is at most equal to that size bound.

[10]Grammar G is consistent with a sample S if $\{x \mid \langle x, 0\rangle \in S\} \subseteq L(G)$ & $L(G) \cap \{x \mid \langle x, 1\rangle \in S\} = \phi$.

3 K-Local Context Free Grammars

The notion of "k-locality" of a context free grammar is defined with respect to a formulation of derivations defined originally for TAG's by Vijay-Shanker, Weir, and Joshi [16] [17], which is a generalization of the notion of a parse tree. In their formulation, a derivation is a tree recording the history of rewritings. Each node of a derivation tree is labeled by a rewriting rule, and in particular, the root must be labeled with a rule with the starting symbol as its left hand side. Each edge corresponds to the application of a rewriting; the edge from a rule (host rule) to another rule (applied rule) is labeled with the "position" of the nonterminal in the right hand side of the host rule at which the rewriting takes place.

The degree of locality of a derivation is the number of distinct kinds of rewritings in it – including the immediate context in which rewritings take place. In terms of a derivation tree, the degree of locality is the number of different kinds of edges in it, where two edges are equivalent just in case the two end nodes are labeled by the same rules, and the edges themselves are labeled by the same node address.

Definition 3.1 *Let $\mathcal{D}(G)$ denote the set of all derivation trees of G, and let $\tau \in \mathcal{D}(G)$. Then, the <u>degree of locality</u> of τ, written $locality(\tau)$, is defined as follows. $locality(\tau) = card\{\langle p, q, \eta\rangle \mid$ there is an edge in τ from a node labeled with p to another labeled with q, and is itself labeled with $\eta\}$*

The degree of locality of a grammar is the maximum of those of all its derivations.

Definition 3.2 *A CFG G is called <u>k-local</u> if $max\{locality(\tau) \mid \tau \in \mathcal{D}(G)\} \leq k$.*
We write k-Local-CFG $= \{G \mid G \in$ CFG and G is k-Local$\}$ and k-Local-CFL $= \{L(G) \mid G \in$ k-Local-CFG $\}$.

Example 3.1 *$L_1 = \{a^n b^n a^m b^m \mid n, m \in N\} \in$ 4-Local-CFL since all the derivations of $G_1 = \langle\{S, S_1\}, \{a, b\}, S, \{S \to S_1 S_1, S_1 \to aS_1 b, S_1 \to \lambda\}\rangle$ generating L_1 have degree of locality at most 4. For example, the derivation for the string a^3b^3ab has degree of locality 4 as shown in Figure 2.*

A crucical property of k-local grammars, which we will utilize in proving the learnability result, is that for each k-local grammar, there exists another k-local grammar in a specific normal form, whose size is only

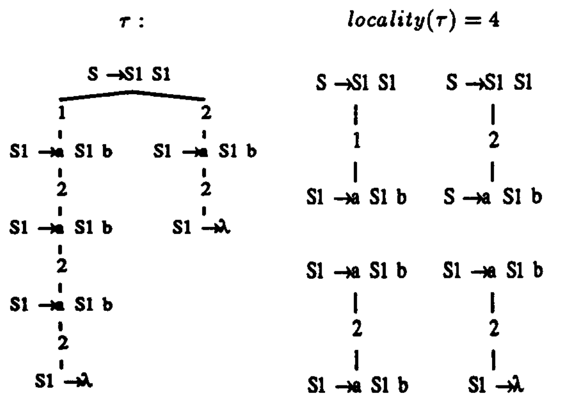

Figure 2: Degree of locality of a derivation of a^3b^3ab by G_1

polynomially larger than the original grammar. The normal form in effect puts the grammar into a disjoint union of small grammars each with at most k rules and k nonterminal occurences. By "the disjoint union" of an arbitrary set of n grammars, $g_1, ..., g_n$, we mean the grammar obtained by first reanaming nonterminals in each g_i so that the nonterminal set of each one is disjoint from that of any other, and then taking the union of the rules in all those grammars, and finally adding the rule $S \to S_i$ for each staring symbol S_i of g_i, and making a brand new symbol S the starting symbol of the grammar so obtained.

Lemma 3.1 (K-Local Normal Form) *For every k-local-CFG H, if $n = size(H)$, then there is a k-local-CFG G such that*

1. *$L(G) = L(H)$.*

2. *G is in k-local normal form, i.e. there is an index set I such that $G = \langle\Sigma_T, \cup_{i\in I}\Sigma_i, S, \{S \to S_i \mid i \in I\} \cup (\cup_{i\in I} R_i)\rangle$, and if we let $G_i = \langle\Sigma_T, \Sigma_i, S_i, R_i\rangle$ for each $i \in I$, then*

   (a) Each G_i is "k-simple"; $\forall i \in I \mid R_i \mid \leq k$ & $NTO(R_i) \leq k$.[11]

   (b) Each G_i has size bounded by size(G); $\forall i \in I$ $size(G_i) = O(n)$

   (c) All G_i's have disjoint nonterminal sets; $\forall i, j \in I (i \neq j) \to \Sigma_i \cap \Sigma_j = \phi$.

3. *$size(G) = O(n^{k+1})$.*

Definition 3.3 *We let ϕ and ψ to be any maps that satisfy: If G is any k-local-CFG in k-local normal form,*

[11] If R is a set of production rules, then $NTO(R_i)$ denotes the number of nonterminal occurrences in those rules.

then $\phi(G)$ is the set of all of its k-local components ($\hat{G}$ above.) If $\hat{G} = \{G_i \mid i \in I\}$ is a set of k-simple grammars, then $\psi(\hat{G})$ is a single grammar that is a "disjoint union" of all of the k-simple grammars in $\hat{G}$.

4 K-Local Context Free Languages Are Polynomially Learnable

In this section, we present a sketch of the proof of our main learnability result.

Theorem 4.1 *For each $k \in N$; k-local-CFL is polynomially learnable.*[12]

Proof:
We prove this by exhibiting an Occam algorithm $\mathcal{A}$ for k-local-CFL with some fixed k, with range size polynomial in the size of a minimal grammar and less than linear in the sample size.

We assume that $\mathcal{A}$ is given a labeled m-sample[13] S_L for some $L \in$ k-local-CFL with $size(H) = n$ where H is its minimal k-local-CFG. We let $length(S_L) = \Sigma_{s \in S}\ length(s) = l$.[14] We let S_L^+ and S_L^- denote the positive and negative portions of S_L respectively, i.e., $S_L^+ = \{x \mid \exists s \in S_L \text{ such that } s = \langle x, 0\rangle\}$ and $S_L^- = \{x \mid \exists s \in S_L \text{ such that } s = \langle x, 1\rangle\}$. We fix a minimal grammar in k-local normal form G that is consistent with S_L with $size(G) \leq p(n)$ for some fixed polynomial p by Lemma 3.1. and the fact that a minimal consistent k-local-CFG is not larger than H. Further, we let $\hat{G}$ be the set of all of "k-simple components" of G and define $L(\hat{G}) = \cup_{G_i \in \hat{G}} L(G_i)$. Then note $L(\hat{G}) = L(G)$. Since each k-simple component has at most k nonterminals, we assume without loss of generality that each G_i in $\hat{G}$ has the same nonterminal set of size k, say $\Sigma_k = \{A_1, ..., A_k\}$.

The idea for constructing $\mathcal{A}$ is straightforward. **Step 1.** We generate all possible rules that may be in the portion of $\hat{G}$ that is *relevant* to S_L^+. That is, if we fix a set of derivations $\mathcal{D}$, one for each string in S_L^+ from $\hat{G}$, then the set of rules that we generate will contain all the rules that participate in any derivation in $\mathcal{D}$. (We let $Rel(\hat{G}, S_L^+)$ denote the *restriction* of $\hat{G}$ to S_L^+ with respect to some $\mathcal{D}$ in this fashion.) We use k-locality of G to show that such a set will be polynomially bounded in the length of S_L^+. **Step 2.** We then generate the set of all possible grammars having at most k of these rules. Since each k-simple component of $\hat{G}$ has at most k rules, the generated set of grammars will include all of the k-simple components of $\hat{G}$. **Step 3.** We then use the negative portion of the sample, S_L^- to filter out the "inconsistent" ones. What we have at this stage is a polynomially bounded set of k-simple grammars with varying sizes, which do not generate any of S_L^-, and contain all the k-simple grammars of $\hat{G}$. Associated with each k-simple grammar is the portion of S_L^+ that it "covers" and its size. **Step 4.** What an Occam algorithm needs to do, then, is to find some subset of these k-simple grammars that "covers" S_L^+, and has a total size that is provably only polynomially larger than a minimal total size of a subset that covers S_L^+, and is less than linear is the sample size, m. We formalize this as a variant of "Set Cover" problem which we call "Weighted Set Cover" (WSC), and prove the existence of an approximation algorithm with a performance guarantee which suffices to ensure that the output of $\mathcal{A}$ will be a grammar that is provably only polynomially larger than the minimal one, and is less than linear in the sample size. The algorithm runs in time polynomial in the size of the grammar being learned and the sample length.

Step 1.
A crucial consequence of the way k-locality is defined is that the "terminal yield" of any rule body that is used to derive any string in the language could be split into at most $k+1$ intervals. (We define the "terminal yield" of a rule body R to be $h(R)$, where h is a homomorphism that preserves terminal symbols and deletes nonterminal symbols.)

Definition 4.1 (Subyields) *For an arbitrary $i \in N$, an i-tuple of members of Σ_T^* $w = \langle v_1, v_2, ..., v_i\rangle$ is said to be a* subyield *of s, if there are some $u_1, ..., u_i, u_{i+1} \in \Sigma_T^*$ such that $s = u_1 v_1 u_2 v_2 ... u_i v_i u_{i+1}$. We let $SubYields(i, s) = \{w \in (\Sigma_T^*)^x \mid x \leq i \text{ \& } w \text{ is a subyield of } s\}$.*

We then let $SubYields_k(S_L^+)$ denote the set of all subyields of strings in S_L^+ that may have come from a rule body in a k-local-CFG, i.e. subyields that are tuples of at most $k+1$ strings.

Definition 4.2
$SubYields_k(S_L^+) = \cup_{s \in S_L^+} Subyields(k+1, s)$.

Claim 4.1 $card(SubYields_k(S_L^+)) = O(l^{2k+3})$.

Proof:
This is obvious, since given a string s of length a, there

[12] We use the size of a minimal k-local CFG as the size of a k-local-CFL, i.e., $\forall L \in$ k-local-CFL $size(L) = min\{size(G) \mid G \in$ k-local-CFG $\&$ $L(G) = L\}$.

[13] S_L is a labeled m-sample for L if $S \subseteq graph(char(L))$ and $card(S) = m$. $graph(char(L))$ is the graph of the characteristic function of L, i.e. is the set $\{\langle x, 0\rangle \mid x \in L\} \cup \{\langle x, 1\rangle \mid x \notin L\}$.

[14] In the sequel, we refer to the number of strings in a sample as the sample size, and the total length of the strings in a sample as the sample length.

are only $O(a^{2(k+1)})$ ways of choosing $2(k+1)$ different positions in the string. This completely specifies all the elements of $SubYields_{k+1}(s)$. Since the number of strings (m) in S_L^+ and the length of each string in S_L^+ are each bounded by the sample length (l), we have at most $O(l) \times O(l^{2(k+1)})$ strings in $SubYields_k(S_L^+)$. □

Thus we now have a polynomially generable set of possible yields of rule bodies in $\hat{G}$. The next step is to generate the set of all possible rules having these yields. Now, by k-locality, in any derivation of $\hat{G}$ we have at most k distinct "kinds" of rewritings present. So, each rule has at most k useful nonterminal occurrences and since $\hat{G}$ is minimal, it is free of useless nonterminals. We generate all possible rules with at most k nonterminal occurrences from some fixed set of k nonterminals (Σ_k), having as terminal subyields, one of $SubYields_k(S_L^+)$. We will then have generated all possible rules of $Rel(\hat{G}, S_L^+)$. In other words, such a set will provably contain all the rules of $Rel(\hat{G}, S_L^+)$. We let TFRules(Σ_k) denote the set of "terminal free rules" $\{A_{i_0} \rightarrow x_1 A_{i_1} x_2 .... x_n A_{i_n} x_{n+1} \mid n \leq k \ \& \ \forall j \leq n \ A_{i_j} \in \Sigma_k\}$ We note that the cardinality of such a set is a function only of k. We then "assign" members of $SubYields_k(S_L^+)$ to TFRules(Σ_k), wherever it is possible (or the arities agree). We let $CRules(k, S_l^+)$ denote the set of "candidate rules" so obtained.

Definition 4.3 $CRules(k, S_L^+) = \{R(w_1/x_1, ..., w_n/x_n) \mid R \in TFRules(\Sigma_k) \ \& \ w \in SubYields_k(S_L^+) \ \& \ arity(w) = arity(R) = n\}$

It is easy to see that the number of rules in such a set is also polynomially bounded.

Claim 4.2 $card(CRules(k, S_L^+)) = O(l^{2k+3})$

Step 2.
Recall that we have assumed that they each have a nonterminal set contained in some fixed set of k nonterminals, Σ_k. So if we generate all subsets of $CRules(k, S_L^+)$ with at most k rules, then these will include all the k-simple grammars in $\hat{G}$.

Definition 4.4
$CGrams(k, S_L^+) = \mathcal{P}_k(CRules(k, S_L^+))$.[15]

Step 3.
Now we finally make use of the negative portion of the sample, S_L^-, to ensure that we do not include any inconsistent grammars in our candidates.

[15] $\mathcal{P}_k(X)$ in general denotes the set of all subsets of X with cardinality at most k.

Definition 4.5 $FGrams(k, S_L) = \{H \mid H \in CGrams(k, S_L^+) \ \& \ L(H) \cap S_L^- = \phi\}$

This filtering can be computed in time polynomial in the length of S_L, because for testing consistency of each grammar in $CGrams(k, S_L^+)$, all that is involved is the membership question for strings in S_L^- with that grammar.

Step 4.
What we have at this stage is a set of 'subcovers' of S_L^+, each with a size (or 'weight') associated with it, and we wish to find a subset of these 'subcovers' that cover the entire S_L^+, but has a provably small 'total weight'. We abstract this as the following problem.

WEIGHTED-SET-COVER(WSC)
INSTANCE: $\langle X, Y, w\rangle$ where X is a finite set and Y is a subset of $\mathcal{P}(X)$ and w is a function from Y to N^+. Intuitively, Y is a set of subcovers of the set X, each associated with its 'weight'.

NOTATION: For every subset Z of Y, we let $cover(Z) = \cup\{z \mid z \in Z\}$, and $totalweight(Z) = \Sigma_{z \in Z}\, w(z)$.

QUESTION: What subset of Y is a set-cover of X with a minimal total weight, i.e. find $Z \subseteq Y$ with the following properties:

(i) $cover(Z) = X$.
(ii) $\forall Z' \subseteq Y$ if $cover(Z') = X$ then $totalweight(Z') \geq totalweight(Z)$.

We now prove the existence of an approximation algorithm for this problem with the desired performance guarantee.

Lemma 4.1 *There is an algorithm $\mathcal{B}$ and a polynomial p such that given an arbitrary instance $\langle X, Y, w\rangle$ of WEIGHTED-SET-COVER with $\mid X \mid = n$, always outputs Z such that;*

1. *$Z \subseteq Y$*
2. *Z is a cover for X, i.e. $\cup Z = X$*
3. *If Z' is a minimal weight set cover for $\langle X, Y, w\rangle$, then $\Sigma_{y \in Z}\, w(y) \leq p(\Sigma_{y \in Z'}\, w(y)) \times \log n$.*
4. *$\mathcal{B}$ runs in time polynomial in the size of the instance.*

Proof: To exhibit an algorithm with this property, we make use of the greedy algorithm $\mathcal{C}$ for the standard set-cover problem due to Johnson ([8]), with a performance guarantee. SET-COVER can be thought of as a special case of WEIGHTED-SET-COVER with weight function being the constant funtion 1.

Theorem 4.2 (David S. Johnson)
There is a greedy algorithm $\mathcal{C}$ for SET-COVER such that given an arbitrary instance $\langle X, Y \rangle$ with an optimal solution Z', outputs a solution Z, such that $card(Z) = O(\log \mid X \mid \times card(Z'))$ and runs in time polynomial in the instance size.

Now we present the algorithm for WSC. The idea of the algorithm is simple. It applies $\mathcal{C}$ on X and successive subclasses of Y with bounded weights, upto the maximum weight there is, but using only powers of 2 as the bounds. It then outputs one with a minimal total weight among those.

Algorithm $\mathcal{B}$: $(\langle X, Y, w \rangle)$

```
$maxweight := max\{w(y) \mid y \in Y\}$
$m := \lceil \log maxweight \rceil$
/* this loop gets an approximate solution using $\mathcal{C}$
for subsets of Y each defined by putting an upperbound
on the weights */
For i = 1 to m do:
    $Y[i] := \{y \mid y \in Y \ \& \ w(y) \leq 2^i\}$
    $s[i] := \mathcal{C}(\langle X, Y[i] \rangle)$
End /* For */
/* this loop replaces all 'bad' (i.e. does not cover X)
solutions with Y - the solution with the maximum
total weight */
For i = 1 to m do:
$s[i] := s[i]$ if $cover(s[i]) = X$
    $:= Y$ otherwise
End /* For */
$mintotalweight := min\{totalweight(s[j]) \mid j \in [m]\}$
Return $s[min\{i \mid totalweight(s[i]) = mintotalweight\}]$
End /* Algorithm $\mathcal{B}$ */
```

Time Analysis

Clearly, Algorithm $\mathcal{B}$ runs in time polynomial in the instance size, since Algorithm $\mathcal{C}$ runs in time polynomial in the instance size and there are only $m = \lceil \log maxweight \rceil$ calls to it, which certainly does not exceed the instance size.

Performance Guarantee

Let $\langle X, Y, w \rangle$ be a given instance with $card(X) = n$. Then let Z^* be an optimal solution of that instance, i.e., it is a minimal total weight set cover. Let $totalweight(Z^*) = w^*$. Now let $m^* = \lceil \log max\{w(z) \mid z \in Z^*\} \rceil$. Then $m^* \leq min(n, \lceil \log maxweight \rceil)$. So when $\mathcal{C}$ is called with an instance $\langle X, Y[m^*] \rangle$ in the m^*-th iteration of the first 'For'-loop in the algorithm, every member of Z^* is in $Y[m^*]$. Hence, the optimal solution of this instance equals Z^*. Thus, by the performance guarantee of $\mathcal{C}$, $s[m^*]$ will be a cover of X with cardinality at most $card(Z^\circ) \times \log n$. Thus, we have $card(s[m^*]) \leq card(Z^*) \times \log n$. Now, for every member t of $s[m^*]$, $w(t) \leq 2^{m^*} \leq 2^{\lceil \log w^* \rceil} \leq 2w^*$. Therefore, $totalweight(s[m^*]) = card(Z^*) \times \log n \times O(2w^*) = O(w^*) \times \log n \times O(2w^*)$, since w^* certainly is at least as large as $card(Z^*)$. Hence, we have $totalweight(s[m^*]) = O(w^{*2} \times \log n)$. Now it is clear that the output of $\mathcal{B}$ will be a cover, and its total weight will not exceed the total weight of $s[m^*]$. We conclude therefore that $\mathcal{B}(\langle X, Y, w \rangle)$ will be a set-cover for X, with total weight bounded above by $O(w^{*2} \times \log n)$, where w^* is the total weight of a minimal weight cover and $n = \mid X \mid$.
□

Now, to apply algorithm $\mathcal{B}$ to our learning problem, we let $Y = \{S_L^+ \cap L(H) \mid H \in FGrams(k, S_L)\}$ and define the weight function $w : Y \rightarrow N^+$ by $\forall y \in Y \ w(y) = min\{size(H) \mid H \in FGrams(k, S_L) \ \& \ y = L(H) \cap S_L^+\}$ and call $\mathcal{B}$ on $\langle S_L^+, Y, w \rangle$. We then output the grammar 'corresponding' to $\mathcal{B}(\langle S_L^+, Y, w \rangle)$. In other words, we let $\hat{H} = \{mingrammar(y) \mid y \in \mathcal{B}(\langle S_L^+, Y, w \rangle)\}$ where $mingrammar(y)$ is a minimal-size grammar H in $FGrams(k, S_L)$ such that $L(H) \cap S_L^+ = y$. The final output grammar H will be the "disjoint union" of all the grammars in $\hat{H}$, i.e. $H = \psi(\hat{H})$. H is clearly consistent with S_L, and since the minimal total weight solution of this instance of WSC is no larger than $Rel(\hat{G}, S_L^+)$, by the performance guarantee on the algorithm $\mathcal{B}$, $size(H) \leq p(size(Rel(\hat{G}, S_L^+))) \times O(\log m)$ for some polynomial p, where m is the sample size. $size(G) \geq size(Rel(\hat{G}, S_L^+))$ is also bounded by a polynomial in the size of a minimal grammar consistent with S_L. We therefore have shown the existence of an Occam algorithm with range size polymomial in the size of a minimal consistent grammar and less than linear in the sample size. Hence, Theorem 4.1 has been proved.

Q.E.D.

5 Extension to Mildly Context Sensitive Languages

The learnability of k-local subclasses of CFG may appear to be quite restricted. It turns out, however, that the learnability of k-local subclasses extends to a rich class of mildly context sensitive grammars which we call "Ranked Node Rewriting Grammars" (RNRG's). RNRG's are based on the underlying ideas of Tree Adjoining Grammars (TAG's) [16], and are also a specical case of context free tree grammars [13] in which unrestricted use of variables for moving, copying and deleting, is not permitted. In other words each rewriting in this system replaces a "ranked" nonterminal node of say rank j with an "incomplete" tree containing exactly j edges that have no descendants. If we define a hierarchy of languages generated by subclasses of RNRG's having nodes and rules with bounded rank j (RNRL$_j$), then RNRL$_0$ = CFL, and RNRL$_1$ = TAL.[17] It turns out that *each* k-local subclass of *each* RNRL$_j$ is polynomially learnable. Further, the constraint of k-locality on RNRG's is an interesting one because not only each k-local subclass is an exponential class containing infinitely many infinite languages, but also k-local subclasses of the RNRG hierarchy become progressively more complex as we go higher in the hierarchy. In particular, for each j, RNRG$_j$ can "count up to" $2(j+1)$ and for each $k \geq 2$, k-local-RNRG$_j$ can also count up to $2(j+1)$.[18]

We will omit a detailed definition of RNRG's (see [2]), and informally illustrate them by some examples.[19]

Example 5.1 *$L_1 = \{a^n b^n \mid n \in N\} \in CFL$ is generated by the following $RNRG_0$ grammar, where α is shown in Figure 3. $G_1 = \langle\{S\}, \{s,a,b\}, \sharp, \{S\}, \{S \rightarrow \alpha, S \rightarrow s(\lambda)\}\rangle$*

Example 5.2 *$L_2 = \{a^n b^n c^n d^n \mid n \in N\} \in TAL$ is generated by the following $RNRG_1$ grammar, where β is shown in Figure 3. $G_2 = \langle\{S\}, \{s,a,b,c,d\}, \sharp, \{(S(\lambda))\}, \{S \rightarrow \beta, S \rightarrow s(\sharp)\}\rangle$*

Example 5.3 *$L_3 = \{a^n b^n c^n d^n e^n f^n \mid n \in N\} \notin TAL$ is generated by the following $RNRG_2$ grammar, where γ is shown in Figure 3. $G_3 = \langle\{S\}, \{s,a,b,c,d,e,f\}, \sharp, \{(S(\lambda,\lambda))\}, \{S \rightarrow \gamma, S \rightarrow s(\sharp,\sharp)\}\rangle$. An example of a tree in the tree language of G_3 having as its yield 'aabbccddeeff' is also shown in Figure 3.*

[16] Tree adjoining grammars were introduced as a formalism for linguistic description by Joshi et al. [10], [9]. Various formal and computational properties of TAG's were studied in [16]. Its linguistic relevance was demonstrated in [11].

[17] This hierarchy is different from the hierarchy of "meta-TAL's" invented and studied extensively by Weir in [18].

[18] A class of grammars $\mathcal{G}$ is said to be able to "count up to" j, just in case $\{a_1^n a_2^n \ldots a_j^n \mid n \in N\} \in \{L(G) \mid G \in \mathcal{G}\}$ but $\{a_1^n a_2^n \ldots a_{j+1}^n \mid n \in N\} \notin \{L(G) \mid G \in \mathcal{G}\}$.

[19] Simpler trees are represented as term structures, whereas more involved trees are shown in the figure. Also note that we use uppercase letters for nonterminals and lowercase for terminals. Note the use of the special symbol $\sharp$ to indicate an edge with no descendent.

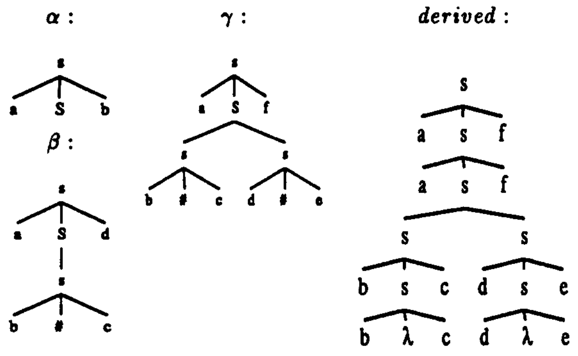

Figure 3: α, β, γ and deriving '$aabbccddeeff$' by G_3

We state the learnability result of $RNRL_j$'s below as a theorem, and again refer the reader to [2] for details. Note that this theorem sumsumes Theorem 4.1 as the case $j = 0$.

Theorem 5.1 *$\forall j, k \in N$ k-local-$RNRL_j$ is polynomially learnable.*[20]

6 Some Negative Results

The reader's reaction to the result described above may be an illusion that the learnability of k-local grammars follows from "bounding by k". On the contrary, we present a case where "bounding by k" not only does not help feasible learning, but in some sense makes it harder to learn. Let us consider Tree Adjoining Grammars *without local constraints*, TAG(wolc) for the sake of comparison.[21] Then an anlogous argument to the one for the learnability of k-local-CFL shows that k-local-TAL(wolc) is polynomially learnable for any k.

Theorem 6.1 *$\forall k \in N^+$ k-local-TAL(wolc) is polynomially learnable.*

Now let us define subclasses of TAG(wolc) with a bounded number of initial trees; k-initial-tree-TAG(wolc) is the class of TAG(wolc) with at most k initial trees. Then surprisingly, for the case of single letter alphabet, we already have the following striking result. (For full detail, see [1].)

Theorem 6.2 *(i) TAL(wolc) on 1-letter alphabet is polynomially learnable.*

[20] We use the size of a minimal k-local RNRG$_j$ as the size of a k-local RNRL$_j$, i.e., $\forall j \in N\ \forall L \in$ k-local-RNRL$_j$ $size(L) = min\{size(G) \mid G \in$ k-local-RNRG$_j$ & $L(G) = L\}$.

[21] Tree Adjoining Grammar formalism was never defined *without* local constrains.

(ii) $\forall k \geq 3$ k-initial-tree-TAL(wolc) on 1-letter alphabet is not polynomially learnable by k-initial-tree-TAG(wolc).

As a corollary to the second part of the above theorem, we have that k-initial-tree-TAL(wolc) on an arbitrary alphabet is not polynomially learnable (by k-initial-tree-TAG(wolc)). This is because we would be able to use a learning algorithm for an arbitrary alphabet to construct one for the single letter alphabet case.

Corollary 6.1 *k-initial-tree-TAL(wolc) is not polynomially learnable by k-initial-tree-TAG(wolc).*

The learnability of k-local-TAL(wolc) and the non-learnability of k-initial-tree-TAL(wolc) is an interesting contrast. Intuitively, in the former case, the "k-bound" is placed so that the grammar is forced to be an arbitrarily "wide" union of boundedly small grammars, whereas, in the latter, the grammar is forced to be a boundedly "narrow" union of arbitrarily large grammars. It is suggestive of the possibility that in fact human infants when acquiring her native tongue may start developing small special purpose grammars for different uses and contexts and slowly start to generalize and compress the large set of similar grammars into a smaller set.

7 Conclusions

We have investigated the use of complexity theory to the evaluation of grammatical systems as linguistic formalisms from the point of view of feasible learnability. In particular, we have demonstrated that a single, natural and non-trivial constraint of "locality" on the grammars allows a rich class of mildly context sensitive languages to be feasibly learnable, in a well-defined complexity theoretic sense. Our work differs from recent works on efficient learning of formal languages, for example by Angluin ([4]), in that it uses only examples and no other powerful oracles. We hope to have demonstrated that learning formal grammars need not be doomed to be necessarily computationally intractable, and the investigation of alternative formulations of this problem is a worthwhile endeavour.

References

[1] Naoki Abe. Polynomial learnability of semilinear sets. 1988. Unpublished manuscript.

[2] Naoki Abe. Polynomially learnable subclasses of mildy context sensitive languages. In *Proceedings of COLING*, August 1988.

[3] Dana Angluin. Inference of reversible languages. *Journal of A.C.M.*, 29:741–765, 1982.

[4] Dana Angluin. *Learing k-bounded context-free grammars.* Technical Report YALEU/DCS/TR-557, Yale University, August 1987.

[5] Dana Angluin. *Learning Regular Sets from Queries and Counter-examples.* Technical Report YALEU/DCS/TR-464, Yale University, March 1986.

[6] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. *Classifying Learnable Geometric Concepts with the Vapnik-Chervonenkis Dimension.* Technical Report UCSC CRL-86-5, University of California at Santa Cruz, March 1986.

[7] E. Mark Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.

[8] David S. Johnson. Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences*, 9:256–278, 1974.

[9] A. K. Joshi. How much context-sensitivity is necessary for characterizing structural description – tree adjoining grammars. In D. Dowty, L. Karttunen, and A. Zwicky, editors, *Natural Language Processing – Theoretical, Computational, and Psychological Perspectives*, Cambridege University Press, 1983.

[10] Aravind K. Joshi, Leon Levy, and Masako Takahashi. Tree adjunct grammars. *Journal of Computer and System Sciences*, 10:136–163, 1975.

[11] A. Kroch and A. K. Joshi. Linguistic relevance of tree adjoining grammars. 1989. To appear in Linguistics and Philosophy.

[12] Daniel N. Osherson, Michael Stob, and Scott Weinstein. *Systems That Learn.* The MIT Press, 1986.

[13] William C. Rounds. Context-free grammars on trees. In *ACM Symposium on Theory of Computing*, pages 143–148, 1969.

[14] Leslie G. Valiant. Learning disjunctions of conjunctions. In *The 9th IJCAI*, 1985.

[15] Leslie G. Valiant. A theory of the learnable. *Communications of A.C.M.*, 27:1134–1142, 1984.

[16] K. Vijay-Shanker and A. K. Joshi. Some computational properties of tree adjoining grammars. In *23rd Meeting of A.C.L.*, 1985.

[17] K. Vijay-Shanker, D. J. Weir, and A. K. Joshi. Characterizing structural descriptions produced by various grammatical formalisms. In *25th Meeting of A.C.L.*, 1987.

[18] David J. Weir. *From Context-Free Grammars to Tree Adjoining Grammars and Beyond – A dissertation proposal.* Technical Report MS-CIS-87-42, University of Pennsylvania, 1987.