Relating Probabilistic Grammars and Automata

Steven Abney **David McAllester** **Fernando Pereira**

AT&T Labs-Research
180 Park Ave
Florham Park NJ 07932
{abney, dmac, pereira}@research.att.com

Abstract

Both probabilistic context-free grammars (PCFGs) and shift-reduce probabilistic pushdown automata (PPDAs) have been used for language modeling and maximum likelihood parsing. We investigate the precise relationship between these two formalisms, showing that, while they define the same classes of probabilistic languages, they appear to impose different inductive biases.

1 Introduction

Current work in stochastic language models and maximum likelihood parsers falls into two main approaches. The first approach (Collins, 1998; Charniak, 1997) uses directly the definition of stochastic grammar, defining the probability of a parse tree as the probability that a certain top-down stochastic generative process produces that tree. The second approach (Briscoe and Carroll, 1993; Black et al., 1992; Magerman, 1994; Ratnaparkhi, 1997; Chelba and Jelinek, 1998) defines the probability of a parse tree as the probability that a certain shift-reduce stochastic parsing automaton outputs that tree. These two approaches correspond to the classical notions of context-free grammars and nondeterministic pushdown automata respectively. It is well known that these two classical formalisms define the same language class. In this paper, we show that *probabilistic context-free grammars* (PCFGs) and *probabilistic pushdown automata* (PPDAs) define the same class of distributions on strings, thus extending the classical result to the stochastic case. We also touch on the perhaps more interesting question of whether PCFGs and shift-reduce parsing models have the same *inductive bias* with respect to the automatic learning of model parameters from data. Though we cannot provide a definitive answer, the constructions we use to answer the equivalence question involve blow-ups in the number of parameters in both directions, suggesting that the two models impose different inductive biases.

We are concerned here with probabilistic shift-reduce parsing models that define probability distributions over word sequences, and in particular the model of Chelba and Jelinek (1998). Most other probabilistic shift-reduce parsing models (Briscoe and Carroll, 1993; Black et al., 1992; Magerman, 1994; Ratnaparkhi, 1997) give only the conditional probability of a parse tree given a word sequence. Collins (1998) has argued that those models fail to capture the appropriate dependency relations of natural language. Furthermore, they are not directly comparable to PCFGs, which define probability distributions over word sequences.

To make the discussion somewhat more concrete, we now present a simplified version of the Chelba-Jelinek model. Consider the following sentence:

> The small woman gave the fat man her sandwich.

The model under discussion is based on *shift-reduce* PPDAs. In such a model, *shift* transitions generate the next word w and its associated syntactic category X and push the pair $\langle X, w \rangle$ on the stack. Each shift transition is followed by zero or more *reduce* transitions that combine topmost stack entries. For example the stack elements $\langle$`Det`, the$\rangle$, $\langle$`Adj`, small$\rangle$, $\langle$`N`, woman$\rangle$ can be combined to form the single entry $\langle$`NP`, woman$\rangle$ representing the phrase "the small woman". In general each stack entry consists of a syntactic category and a head word. After generating the prefix "The small woman gave the fat man" the stack might contain the sequence $\langle$`NP`, woman$\rangle\langle$`V`, gave$\rangle\langle$`NP`, man$\rangle$. The Chelba-Jelinek model then executes a shift tran-

$$
\begin{array}{rcl}
S & \rightarrow & \langle \texttt{S}, \text{ admired} \rangle \\
\langle \texttt{S}, \text{ admired} \rangle & \rightarrow & \langle \texttt{NP}, \text{ Mary} \rangle \, \langle \texttt{VP}, \text{ admired} \rangle \\
\langle \texttt{VP}, \text{ admired} \rangle & \rightarrow & \langle \texttt{V}, \text{ admired} \rangle \langle \texttt{NP}, \text{ oak} \rangle \\
\langle \texttt{NP}, \text{ oak} \rangle & \rightarrow & \langle \texttt{Det}, \text{ the} \rangle \langle \texttt{N}, \text{ oak} \rangle \\
\langle \texttt{N}, \text{ oak} \rangle & \rightarrow & \langle \texttt{Adj}, \text{ towering} \rangle \langle \texttt{N}, \text{ oak} \rangle \\
\langle \texttt{N}, \text{ oak} \rangle & \rightarrow & \langle \texttt{Adj}, \text{ strong} \rangle \langle \texttt{N}, \text{ oak} \rangle \\
\langle \texttt{N}, \text{ oak} \rangle & \rightarrow & \langle \texttt{Adj}, \text{ old} \rangle \langle \texttt{N}, \text{ oak} \rangle \\
\langle \texttt{NP}, \text{ Mary} \rangle & \rightarrow & \text{Mary} \\
 & \vdots & \\
\langle \texttt{N}, \text{ oak} \rangle & \rightarrow & \text{oak}
\end{array}
$$

Figure 1: Lexicalized context-free grammar

sition by generating the next word. This is done in a manner similar to that of a trigram model except that, rather than generate the next word based on the two preceding words, it generates the next word based on the two topmost stack entries. In this example the Chelba-Jelinek model generates the word "her" from $\langle \texttt{V}, \text{ gave} \rangle \langle \texttt{NP}, \text{ man} \rangle$ while a classical trigram model would generate "her" from "fat man".

We now contrast Chelba-Jelinek style models with lexicalized PCFG models. A PCFG is a context-free grammar in which each production is associated with a weight in the interval [0, 1] and such that the weights of the productions from any given nonterminal sum to 1. For instance, the sentence

Mary admired the towering strong old oak

can be derived using a lexicalized PCFG based on the productions in Figure 1. Production probabilities in the PCFG would reflect the likelihood that a phrase headed by a certain word can be expanded in a certain way. Since it can be difficult to estimate fully these likelihoods, we might restrict ourselves to models based on *bilexical* relationships (Eisner, 1997), those between pairs of words. The simplest bilexical relationship is a bigram statistic, the fraction of times that "oak" follows "old". Bilexical relationships for a PCFG include that between the head-word of a phrase and the head-word of a non-head immediate constituent, for instance. In particular, the generation of the above sentence using a PCFG based on Figure 1 would exploit a bilexical statistic between "towering" and "oak" contained in the weight of the fifth production. This bilexical relationship between "towering" and "oak" would not be exploited in either a trigram model or in a Chelba-Jelinek style model. In a Chelba-Jelinek style model one must generate "towering" before generating "oak" and then "oak" must be generated from $\langle \texttt{Adj}, \text{ strong} \rangle$, $\langle \texttt{Adj}, \text{ old} \rangle$. In this example the Chelba-Jelinek model behaves more like a classical trigram model than like a PCFG model.

This contrast between PPDAs and PCFGs is formalized in theorem 1, which exhibits a PCFG for which no stochastic parameterization of the corresponding shift-reduce parser yields the same probability distribution over strings. That is, the standard shift-reduce translation from CFGs to PDAs cannot be generalized to the stochastic case.

We give two ways of getting around the above difficulty. The first is to construct a *top-down* PPDA that mimics directly the process of generating a PCFG derivation from the start symbol by repeatedly replacing the leftmost nonterminal in a sentential form by the right-hand side of one of its rules. Theorem 2 states that any PCFG can be translated into a top-down PPDA. Conversely, theorem 3 states that any PPDA can be translated to a PCFG, not just those that are top-down PPDAs for some PCFG. Hence PCFGs and general PPDAs define the same class of stochastic languages.

Unfortunately, top-down PPDAs do not allow the simple left-to-right processing that motivates shift-reduce PPDAs. A second way around the difficulty formalized in theorem 1 is to encode additional information about the derivation context with richer stack and state alphabets. Theorem 7 shows that it is thus possible to translate an arbitrary PCFG to a shift-reduce PPDA . The construction requires a fair amount of machinery including proofs that any PCFG can be put in Chomsky normal form, that weights can be renormalized to ensure that the result of grammar transformations can be made into PCFGs, that any PCFG can be put in Greibach normal form, and, finally, that a Greibach normal form PCFG can be converted to a shift-reduce PPDA.

The construction also involves a blow-up in the size of the shift-reduce parsing automaton. This suggests that some languages that are concisely describable by a PCFG are not concisely describable by a shift-reduce PPDA, hence that the class of PCFGs and the class of shift-reduce PPDAs impose different inductive biases on the

CF languages. In the conversion from shift-reduce PPDAs to PCFGs, there is also a blow-up, if a less dramatic one, leaving open the possibility that the biases are incomparable, and that neither formalism is inherently more concise.

Our main conclusion is then that, while the generative and shift-reduce parsing approaches are weakly equivalent, they impose different inductive biases.

2 Probabilistic and Weighted Grammars

For the remainder of the paper, we fix a terminal alphabet Σ and a nonterminal alphabet N, to which we may add auxiliary symbols as needed.

A weighted context-free grammar (WCFG) consists of a distinguished start symbol $S \in N$ plus a finite set of weighted productions of the form $X \stackrel{u}{\to} \alpha$, (alternately, $u : X \to \alpha$), where $X \in N$, $\alpha \in (N \cup \Sigma)^*$ and the weight u is a nonnegative real number. A probabilistic context-free grammar (PCFG) is a WCFG such that for all X, $\sum_{u:X\to\alpha} u = 1$. Since weights are nonnegative, this also implies that $u \leq 1$ for any individual production.

A PCFG defines a stochastic process with sentential forms as states, and leftmost rewriting steps as transitions. In the more general case of WCFGs, we can no longer speak of stochastic processes; but weighted parse trees and sets of weighted parse trees are still well-defined notions.

We define a parse tree to be a tree whose nodes are labeled with productions. Suppose node ξ is labeled $X \stackrel{u}{\to} \alpha[Y_1, \ldots, Y_n]$, where we write $\alpha[Y_1, \ldots, Y_n]$ for a string whose nonterminal symbols are $Y_1, \ldots, Y_n$. We say that ξ's nonterminal label is X and its weight is u. The subtree rooted at ξ is said to be rooted in X. ξ is well-labeled just in case it has n children, whose nonterminal labels are $Y_1, \ldots, Y_n$, respectively. Note that a terminal node is well-labeled only if α is empty or consists exclusively of terminal symbols. We say a WCFG G *admits* a tree d just in case all nodes of d are well-labeled, and all labels are productions of G. Note that no requirement is placed on the nonterminal of the root node of d; in particular, it need not be S.

We define the weight of a tree d, denoted $W_G(d)$, or $W(d)$ if G is clear from context, to be the product of weights of its nodes. The *depth* $\tau(d)$ of d is the length of the longest path from root to leaf in d. The *root production* $\pi(d)$ is the label of the root node. The *root symbol* $\rho(d)$ is the left-hand side of $\pi(d)$. The *yield* $\sigma(d)$ of the tree d is defined in the standard way as the string of terminal symbols "parsed" by the tree.

It is convenient to treat the functions π, ρ, σ, and τ as random variables over trees. We write, for example, $\{\rho = X\}$ as an abbreviation for $\{d|\rho(d) = X\}$; and $W_G(\rho = X)$ represents the sum of weights of such trees. If the sum diverges, we set $W_G(\rho = X) = \infty$. We call $\|X\|_G = W_G(\rho = X)$ the *norm* of X, and $\|G\| = \|S\|_G$ the norm of the grammar.

A WCFG G is called *convergent* if $\|G\| < \infty$. If G is a PCFG then $\|G\| = W_G(\rho = S) \leq 1$, that is, all PCFGs are convergent. A PCFG G is called *consistent* if $\|G\| = 1$. A sufficient condition for the consistency of a PCFG is given in (Booth and Thompson, 1973). If Φ and Ψ are two sets of parse trees such that $0 < W_G(\Psi) < \infty$ we define $P_G(\Phi|\Psi)$ to be $W_G(\Phi \cap \Psi)/W_G(\Psi)$. For any terminal string y and grammar G such that $0 < W_G(\rho = S) < \infty$ we define $P_G(y)$ to be $P_G(\sigma = y|\rho = S)$.

3 Stochastic Push-Down Automata

We use a somewhat nonstandard definition of pushdown automaton for convenience, but all our results hold for a variety of essentially equivalent definitions. In addition to the terminal alphabet Σ, we will use sets of stack symbols and states as needed. A weighted push-down automaton (WPDA) consists of a distinguished start state q_0, a distinguished start stack symbol X_0 and a finite set of transitions of the following form where p and q are states, $a \in \Sigma \cup \{\epsilon\}$, X and Z_1, ..., Z_n are stack symbols, and w is a nonnegative real weight:

$$X, p \stackrel{a,w}{\to} Z_1 \ \ldots \ Z_n, q$$

A WPDA is a probabilistic push-down automaton (PPDA) if all weights are in the interval [0, 1] and for each pair of a stack symbol X and a state q the sum of the weights of all transitions of the form $X, p \stackrel{a,w}{\to} Z_1 \ldots Z_n, q$ equals 1. A machine configuration is a pair $\langle\beta,\ q\rangle$ of a finite sequence β of stack symbols (a stack) and a machine state q. A machine configuration is called *halting* if the stack is empty. If M is a PPDA containing the transition $X, p \stackrel{a,w}{\to} Z_1 \ldots Z_n, q$ then any configuration of the form $\langle\beta X,\ p\rangle$ has probability w of being transformed into the configuration $\langle \beta Z_1 \ldots Z_n,\ q \rangle$ where this transformation has the effect of "outputting" a if $a \neq \epsilon$. A complete execution of M is a sequence of transitions between configurations starting in the initial configuration $\langle X_0,\ q_0 \rangle$ and ending in a configuration with an empty stack. The probability of a complete execution is the product of the probabilities of the individual transitions between configurations in that execution. For any PPDA M and $y \in \Sigma^*$ we define $P_M(y)$ to be the sum of the probabilities of all complete executions outputting y. A PPDA M is called consistent if $\sum_{y \in \Sigma^*} P_M(y) = 1$.

We first show that the well known shift-reduce conversion of CFGs into PDAs can not be made to handle the stochastic case. Given a (non-probabilistic) CFG G in Chomsky normal form we define a (non-probabilistic) shift-reduce PDA $SR(G)$ as follows. The stack symbols of $SR(G)$ are taken to be nonterminals of G plus the special symbols $\top$ and $\bot$. The states of $SR(G)$ are in one-to-one correspondence with the stack symbols and we will abuse notation by using the same symbols for both states and stack symbols. The initial stack symbol is $\bot$ and the initial state is (the state corresponding to) $\bot$. For each production of the form $X \rightarrow a$ in G the PDA $SR(G)$ contains all shift transitions of the following form

$$Y, Z \xrightarrow{a} YZ, X$$

The PDA $SR(G)$ also contains the following termination transitions where S is the start symbol of G.

$$\bot, S \xrightarrow{\epsilon}, \top$$
$$\bot, \top \xrightarrow{\epsilon}, \top$$

Note that if G consists entirely of productions of the form $S \rightarrow a$ these transitions suffice. More generally, for each production of the form $X \rightarrow YZ$ in G the PDA $SR(G)$ contains the following reduce transitions.

$$Y, Z \xrightarrow{\epsilon}, X$$

All reachable configurations are in one of the following four forms where the first is the initial configuration, the second is a template for all intermediate configurations with $\alpha \in N^*$, and the last two are terminal configurations.

$$\langle \bot,\ \bot \rangle,\ \ \langle \bot\bot\alpha,\ X \rangle,\ \ \langle \bot, \top \rangle,\ \ \langle \epsilon,\ \top \rangle$$

Furthermore, a configuration of the form $\langle \bot\bot\alpha,\ X \rangle$ can be reached after outputting y if and only if $\alpha X \stackrel{*}{\Rightarrow} y$. In particular, the machine can reach configuration $\langle \bot\bot,\ S \rangle$ outputting y if and only if $S \stackrel{*}{\Rightarrow} y$. So the machine $SR(G)$ generates the same language as G.

We now show that the shift-reduce translation of CFGs into PDAs does not generalize to the stochastic case. For any PCFG G we define the underlying CFG to be the result of erasing all weights from the productions of G.

Theorem 1 *There exists a consistent PCFG G in Chomsky normal form with underlying CFG G' such that no consistent weighting M of the PDA $SR(G')$ has the property that $P_M(y) = P_G(y)$ for all $y \in \Sigma^*$.*

To prove the theorem take G to be the following grammar.

$$S \xrightarrow{\frac{1}{2}} AX_1,\ S \xrightarrow{\frac{1}{2}} BY_1$$
$$X_1 \xrightarrow{1} CX_2,\ X_2 \xrightarrow{1} CA$$
$$Y_1 \xrightarrow{1} CY_2,\ Y_2 \xrightarrow{1} CB$$
$$A \xrightarrow{1} a,\ B \xrightarrow{1} b,\ C \xrightarrow{1} c$$

Note that G generates $acca$ and $bccb$ each with probability $\frac{1}{2}$. Let M be a consistent PPDA whose transitions consist of some weighting of the transitions of $SR(G')$. We will assume that $P_M(y) = P_G(y)$ for all $y \in \Sigma^*$ and derive a contradiction. Call the nonterminals A, B, and C preterminals. Note that the only reduce transitions in $SR(G')$ combining two preterminals are $C, A \xrightarrow{\epsilon}, X_2$ and $C, B \xrightarrow{\epsilon}, Y_2$. Hence the only machine configuration reachable after outputting the sequence acc is $\langle \bot\bot AC, C \rangle$. If $P_M(acca) = \frac{1}{2}$ and $P_M(accb) = 0$ then the machine in configuration $\langle \bot\bot AC, C \rangle$ must deterministically move to configuration $\langle \bot\bot ACC, A \rangle$. But this implies that configuration $\langle \bot\bot BC, C \rangle$ also deterministically moves to configuration $\langle \bot\bot BCC, A \rangle$ so we have $P_M(bccb) = 0$ which violates the assumptions about M. ∎

Although the standard shift-reduce translation of CFGs into PDAs fails to generalize to the stochastic case, the standard top-down conversion easily generalizes. A top-down PPDA is one in which only ϵ transitions can cause the stack to grow and transitions which output a word must pop the stack.

Theorem 2 *Any string distribution definable by a consistent PCFG is also definable by a top-down PPDA.*

Here we consider only PCFGs in Chomsky normal form—the generalization to arbitrary PCFGs is straightforward. Any PCFG in Chomsky normal form can be translated to a top-down PPDA by translating each weighted production of the form $X \stackrel{w}{\rightarrow} YZ$ to the set of *expansion* moves of the form $W, X \stackrel{\epsilon,w}{\rightarrow} WZ, Y$ and each production of the form $X \stackrel{w}{\rightarrow} a$ to the set of pop moves of the form $Z, X \stackrel{a,w}{\rightarrow}, Z$. ∎

We also have the following converse of the above theorem.

Theorem 3 *Any string distribution definable by a consistent PPDA is definable by a PCFG.*

The proof, omitted here, uses a weighted version of the standard translation of a PDA into a CFG followed by a renormalization step using lemma 5. We note that it does in general involve an increase in the number of parameters in the derived PCFG.

In this paper we are primarily interested in shift-reduce PPDAs which we now define formally. In a shift-reduce PPDA there is a one-to-one correspondence between states and stack symbols and every transition has one of the following two forms.

$$Y, Z \stackrel{a,w}{\rightarrow} YZ, X \quad a \neq \epsilon$$
$$Y, Z \stackrel{\epsilon,w}{\rightarrow}, X$$

Transitions of the first type are called *shift* transitions and transitions of the second type are called *reduce* transitions. Shift transitions output a terminal symbol and push a single symbol on the stack. Reduce transitions are ϵ-transitions that combine two stack symbols. The above theorems leave open the question of whether shift-reduce PPDAs can express arbitrary context-free distributions. Our main theorem is that they can. To prove this some additional machinery is needed.

4 Chomsky Normal Form

A PCFG is in Chomsky normal form (CNF) if all productions are either of the form $X \stackrel{u}{\rightarrow} a$, $a \in \Sigma$ or $X \stackrel{u}{\rightarrow} Y_1Y_2$, $Y_1, Y_2 \in N$. Our next theorem states, in essence, that any PCFG can be converted to Chomsky normal form.

Theorem 4 *For any consistent PCFG G with $P_G(\epsilon) < 1$ there exists a consistent PCFG $C(G)$ in Chomsky normal form such that, for all $y \in \Sigma^+$:*

$$P_{C(G)}(y) = \frac{P_G(y)}{1 - P_G(\epsilon)} = P_G(y | y \neq \epsilon)$$

To prove the theorem, note first that, without loss of generality, we can assume that all productions in G are of one of the forms $X \stackrel{u}{\rightarrow} YZ$, $X \stackrel{u}{\rightarrow} Y$, $X \stackrel{u}{\rightarrow} a$, or $X \stackrel{u}{\rightarrow} \epsilon$. More specifically, any production not in one of these forms must have the form $X \stackrel{u}{\rightarrow} \alpha\beta$ where α and β are nonempty strings. Such a production can be replaced by $X \stackrel{u}{\rightarrow} AB$, $A \stackrel{1}{\rightarrow} \alpha$, and $B \stackrel{1}{\rightarrow} \beta$ where A and B are fresh nonterminal symbols. By repeatedly applying this *binarization* transformation we get a grammar in the desired form defining the same distribution on strings.

We now assume that all productions of G are in one of the above four forms. This implies that a node in a G-derivation has at most two children. A node with two children will be called a *branching node*. Branching nodes must be labeled with a production of the form $X \stackrel{u}{\rightarrow} YZ$. Because G can contain productions of the form $X \stackrel{u}{\rightarrow} \epsilon$ there may be arbitrarily large G-derivations with empty yield. Even G-derivations with nonempty yield may contain arbitrarily large subtrees with empty yield. A branching node in the G-derivation will be called *ephemeral* if either of its children has empty yield. Any G-derivation d with $|\sigma(d)| \geq 2$ must contain a unique shallowest non-ephemeral branching node, labeled by some production $X \stackrel{u}{\rightarrow} YZ$. In this case, define $\beta(d) = YZ$. Otherwise ($|\sigma(d)| < 2$), let $\beta(d) = \sigma(d)$. We say that a nonterminal X is *nontrivial* in the grammar G if $P_G(\sigma \neq \epsilon \mid \rho = X) > 0$. We now define the grammar G' to consist of all productions of the following form where X, Y, and Z are nontrivial nonterminals of G and a is a terminal symbol appearing in G.

$$X \stackrel{P_G(\beta = YZ \mid \rho = X, \sigma \neq \epsilon)}{\rightarrow} YZ$$
$$X \stackrel{P_G(\beta = a \mid \rho = X, \sigma \neq \epsilon)}{\rightarrow} a$$

We leave it to the reader to verify that G' has the property stated in theorem 4. ∎

The above proof of theorem 4 is nonconstructive in that it does not provide any way of computing the conditional probabilities $P_G(\beta = YZ \mid \rho = X,\ \sigma \neq \epsilon)$ and $P_G(\beta = a \mid \rho = X,\ \sigma \neq \epsilon)$. However, it is not difficult to compute probabilities of the form $P_G(\Phi \mid \rho = X,\ \tau \leq t+1)$ from probabilities of the form $P_G(\Phi \mid \rho = X,\ \tau \leq t)$, and $P_G(\Phi \mid \rho = X)$ is the limit as t goes to infinity of $P_G(\Phi \mid \rho = X,\ \tau \leq t)$. We omit the details here.

5 Renormalization

A nonterminal X is called *reachable* in a grammar G if either X is S or there is some (recursively) reachable nonterminal Y such that G contains a production of the form $Y \xrightarrow{u} \alpha$ where α contains X. A nonterminal X is nonempty in G if G contains $X \xrightarrow{u} \alpha$ where $u > 0$ and α contains only terminal symbols, or G contains $X \xrightarrow{u} \alpha[Y_1,\ \ldots,\ Y_k]$ where $u > 0$ and each Y_i is (recursively) nonempty. A WCFG G is *proper* if every nonterminal is both reachable and nonempty. It is possible to efficiently compute the set of reachable and nonempty nonterminals in any grammar. Furthermore, the subset of productions involving only nonterminals that are both reachable and nonempty defines the same weight distribution on strings. So without loss of generality we need only consider proper WCFGs. A *reweighting* of G is any WCFG derived from G by changing the weights of the productions of G.

Lemma 5 *For any convergent proper WCFG G, there exists a reweighting G' of G such that G' is a consistent PCFG such that for all terminal strings y we have $P_{G'}(y) = P_G(y)$.*

Proof: Since G is convergent, and every nonterminal X is reachable, we must have $\|X\|_G < \infty$. We now renormalize all the productions from X as follows. For each production $X \xrightarrow{u} \alpha[Y_1, \ldots, Y_n]$ we replace u by

$$u' = u \frac{\prod_i \|Y_i\|_G}{\|X\|_G} \quad .$$

To show that G' is a PCFG we must show that the sum of the weights of all productions from X equals 1:

$$\begin{aligned}
&\textstyle\sum_{u':X\to\alpha[Y_1,\ldots,Y_n]} u' \\
&= \textstyle\sum_{u:X\to\alpha[Y_1,\ldots,Y_n]} u \frac{\prod_i \|Y_i\|_G}{\|X\|_G} \\
&= \textstyle\frac{1}{\|X\|_G} \sum_{u:X\to\alpha[Y_1,\ldots,Y_n]} u \prod_i \|Y_i\|_G \\
&= \textstyle\frac{1}{\|X\|_G} \sum_{u:X\to\alpha[Y_1,\ldots,Y_n]} u \prod_i W_G(\rho = Y_i) \\
&= \textstyle\frac{1}{W_G(\rho=X)} W_G(\rho = X) \\
&= 1
\end{aligned}$$

For any parse tree d admitted by G let d' be the corresponding tree admitted by G', that is, the result of reweighting the productions in d. One can show by induction on the depth of parse trees that if $\rho(d) = X$ then $W_{G'}(d') = \frac{1}{\|X\|_G} W_G(d)$. Therefore $\|X\|_{G'} = \sum_{\{d|\rho(d)=X\}} W_{G'}(d') = \frac{1}{\|X\|_G} \sum_{\{d|\rho(d)=X\}} W_G(d) = \frac{W_G(X)}{\|X\|_G} = 1$. In particular, $\|G'\| = \|S\|_{G'} = 1$, that is, G' is consistent. This implies that for any terminal string y we have $P_{G'}(y) = \frac{1}{\|G'\|} W_{G'}(\sigma = y,\ \rho = S) = W_{G'}(\sigma = y,\ \rho = S)$. Furthermore, for any tree d with $\rho(d) = S$ we have $W_{G'}(d') = \frac{1}{\|S\|_G} W_G(d)$ and so $W_{G'}(\sigma = y,\ \rho = S) = \frac{1}{\|S\|_G} W_G(\sigma = y,\ \rho = S) = P_G(y)$. ∎

6 Greibach Normal Form

A PCFG is in Greibach normal form (GNF) if every production $X \xrightarrow{u} \alpha$ satisfies $\alpha \in \Sigma N^*$. The following holds:

Theorem 6 *For any consistent PCFG G in CNF there exists a consistent PCFG G' in GNF such that $P_{G'}(y) = P_G(y)$ for $y \in \Sigma^*$.*

Proof: A *left corner G-derivation from X to Y* is a G-derivation from X where the leftmost leaf, rather than being labeled with a production, is simply labeled with the nonterminal Y. For example, if G contains the productions $X \xrightarrow{w_1} YZ$ and $Z \xrightarrow{w_2} a$ then we can construct a left corner G-derivation from X to Y by building a tree with a root labeled by $X \xrightarrow{w_1} YZ$, a left child labeled with Y and a right child labeled with $Z \xrightarrow{w_2} a$. The weight of a left corner G-derivation is the product of the productions on the nodes. A tree consisting of a single node labeled with X is a left corner G-derivation from X to X.

For each pair of nonterminals X, Y in G we introduce a new nonterminal symbol X/Y.

The H-derivations from X/Y will be in one to one correspondence with the left-corner G-derivations from X to Y. For each production in G of the form $X \stackrel{u}{\rightarrow} a$ we include the following in H where S is the start symbol of G:

$$S \stackrel{u}{\rightarrow} a\ S/X \qquad .$$

We also include in H all productions of the following form where X is any nonterminal in G:

$$X/X \stackrel{1}{\rightarrow} \epsilon \qquad .$$

If G consists only of productions of the form $S \stackrel{u}{\rightarrow} a$ these productions suffice. More generally, for each nonterminal X/Y of H and each pair of productions $U \stackrel{w_1}{\rightarrow} YZ$, $W \stackrel{w_2}{\rightarrow} a$ we include in H the following:

$$X/Y \stackrel{w_1 w_2}{\rightarrow} a\ Z/W\ X/U \qquad .$$

Because of the productions $X/X \stackrel{1}{\rightarrow} \epsilon$, $W_H(\rho = X/X) \geq 1$, and H is not quite in GNF. These two issues will be addressed momentarily.

Standard arguments can be used to show that the H-derivations from X/Y are in one-to-one correspondence with the left corner G-derivations from X to Y. Furthermore, this one-to-one correspondence preserves weight—if d is the H-derivation rooted at X/Y corresponding to the left corner G-derivation from X to Y then $W_H(d)$ is the product of the weights of the productions in the G-derivation.

The weight-preserving one-to-one correspondence between left-corner G-derivations from X to Y and H-derivations from X/Y yields the following.

$$\begin{aligned} & W_H(a\alpha) \\ & = \textstyle\sum_{(S \stackrel{u}{\rightarrow} aS/X) \in H} uW_H(\sigma = \alpha \mid \rho = S/X) \\ & = \textstyle\sum_{(X \stackrel{u}{\rightarrow} a) \in G} uW_H(\sigma = \alpha \mid \rho = S/X) \\ & = P_G(a\alpha) \end{aligned}$$

Theorem 5 implies that we can reweight the *proper subset* of H (the reachable and nonempty productions of H) so as to construct a consistent PCFG J with $P_J(\alpha) = P_G(\alpha)$. To prove theorem 6 it now suffices to show that the productions of the form $X/X \stackrel{u}{\rightarrow} \epsilon$ can be eliminated from the PCFG J. Indeed, we can eliminate the ϵ productions from J in a manner similar to that used in the proof of theorem 4. A node in an J-derivation is *ephemeral* if it is labeled $X \stackrel{u}{\rightarrow} \epsilon$ for some X. We now define a function γ on J-derivations d as follows. If the root of d is labeled with $X \stackrel{u}{\rightarrow} aYZ$ then we have four subcases. If neither child of the root is ephemeral then $\gamma(d)$ is the string aYZ. If only the left child is ephemeral then $\gamma(d)$ is aZ. If only the right child is ephemeral then $\gamma(d)$ is aY and if both children are ephemeral then $\gamma(d)$ is a. Analogously, if the root is labeled with $X \stackrel{u}{\rightarrow} aY$, then $\gamma(d)$ is aY if the child is not ephemeral and a otherwise. If the root is labeled with $X \stackrel{u}{\rightarrow} \epsilon$ then $\gamma(d)$ is ϵ.

A nonterminal X in K will be called trivial if $P_J(\gamma = \epsilon \mid \rho = X) = 1$. We now define the final grammar G' to consist of all productions of the following form where X, Y, and Z are nontrivial nonterminals appearing in J and a is a terminal symbol appearing in J.

$$\begin{aligned} X &\stackrel{P_J(\sigma = a \mid \rho = X,\ \gamma \neq \epsilon)}{\longrightarrow} a \\ X &\stackrel{P_J(\sigma = aY \mid \rho = X,\ \gamma \neq \epsilon)}{\longrightarrow} aY \\ X &\stackrel{P_J(\sigma = aYZ \mid \rho = X,\ \gamma \neq \epsilon)}{\longrightarrow} aYZ \end{aligned}$$

As in section 4, for every nontrivial nonterminal X in K and terminal string α we have $P_K(\sigma = \alpha \mid \rho = X) = P_J(\sigma = \alpha \mid \rho = X,\ \sigma \neq \epsilon)$. In particular, since $P_J(\epsilon) = P_G(\epsilon) = 0$, we have the following:

$$\begin{aligned} P_K(\alpha) &= P_J(\sigma = \alpha \mid \rho = S,\ \sigma \neq \epsilon) \\ &= P_J(\sigma = \alpha \mid \rho = S) \\ &= P_J(\sigma) \\ &= P_G(\sigma) \end{aligned}$$

The PCFG K is the desired PCFG in Greibach normal form. ■

The construction in this proof is essentially the standard left-corner transformation (Rosenkrantz and II, 1970), as extended by Salomaa and Soittola (1978, theorem 2.3) to algebraic formal power series.

7 The Main Theorem

We can now prove our main theorem.

Theorem 7 *For any consistent PCFG G there exists a shift-reduce PPDA M such that $P_M(y) = P_G(y)$ for all $y \in \Sigma^*$.*

Let G be an arbitrary consistent PCFG. By theorems 4 and 6, we can assume that G consists of productions of the form $S \stackrel{w}{\rightarrow} \epsilon$ and

$S \stackrel{1-w}{\rightarrow} S'$ plus productions in Greibach normal form not mentioning S. We can then replace the rule $S \stackrel{1-w}{\rightarrow} S'$ with all rules of the form $S \stackrel{(1-w)w'}{\rightarrow} \alpha$ where G contains $S' \stackrel{w'}{\rightarrow} \alpha$. We now assume without loss of generality that G consists of a single production of the form $S \stackrel{u}{\rightarrow} \epsilon$ plus productions in Greibach normal form not mentioning S on the right hand side.

The stack symbols of M are of the form W_α where $\alpha \in N^*$ is a proper suffix of the right hand side of some production in G. For example, if G contains the production $X \stackrel{u}{\rightarrow} aYZ$ then the symbols of M include W_{YZ}, W_Y, and W_ϵ. The initial state is W_S and the initial stack symbol is $\perp$. We have assumed that G contains a unique production of the form $S \stackrel{u}{\rightarrow} \epsilon$. We include the following transition in M corresponding to this production.

$$\perp, W_S \stackrel{\epsilon,w}{\rightarrow}, \top$$

Then, for each rule of the form $X \stackrel{u}{\rightarrow} a\beta$ in G and each symbol of the form $W_{X\alpha}$ we include the following in M:

$$Z, W_{X\alpha} \stackrel{a,w}{\rightarrow} ZW_{X\alpha}, W_\beta$$

We also include all "post-processing" rules of the following form:

$$\begin{gathered} W_{X\alpha}W_\epsilon \stackrel{\epsilon,1}{\rightarrow} W_\alpha \\ \perp, W_\epsilon \stackrel{\epsilon,1}{\rightarrow}, \top \\ \perp, \top \stackrel{\epsilon,1}{\rightarrow}, \top \end{gathered}$$

Note that all reduction transitions are deterministic with the single exception of the first rule listed above. The nondeterministic shift transitions of M are in one-to-one correspondence with the productions of G. This yields the property that $P_M(y) = P_G(y)$. ∎

8 Conclusions

The relationship between PCFGs and PPDAs is subtler than a direct application of the classical constructions relating general CFGs and PDAs. Although PCFGs can be concisely translated into top-down PPDAs, we conjecture that there is no *concise* translation of PCFGs into shift-reduce PPDAs. Conversely, there appears to be no concise translation of shift-reduce PPDAs to PCFGs. Our main result is that PCFGs and shift-reduce PPDAs are intertranslatable, hence weakly equivalent. However, the nonconciseness of our translations is consistent with the view that stochastic top-down generation models are significantly different from shift-reduce stochastic parsing models, affecting the ability to learn a model from examples.

References

Alfred V. Aho and Jeffrey D. Ullman. 1972. *The Theory of Parsing, Translation and Compiling*, volume I. Prentice-Hall, Englewood Cliffs, New Jersey.

Ezra Black, Fred Jelinek, John Lafferty, David Magerman, Robert Mercer, and Salim Roukos. 1992. Towards history-based grammars: Using richer models for probabilistic parsing. In *Proceedings of the 5th DARPA Speech and Natural Language Workshop.*

Taylor Booth and Richard Thompson. 1973. Applying probability measures to abstract languages. *IEEE Transactions on Computers*, C-22(5):442–450.

Ted Briscoe and John Carroll. 1993. Generalized probabilistic LR parsing of natural language (corpora) with unification-based grammars. *Computational Linguistics*, 19(1):25–59.

Eugene Charniak. 1997. Statistical parsing with a context-free grammar and word statistics. In *Fourteenth National Conference on Artificial Intelligence*, pages 598–603. AAAI Press/MIT Press.

Ciprian Chelba and Fred Jelinek. 1998. Exploiting syntactic structure for language modeling. In *COLING-ACL '98*, pages 225–231.

Michael Collins. 1998. *Head-Driven Statistical Models for Natural Language Parsing.* Ph.D. thesis, University of Pennsylvania.

Jason Eisner. 1997. Bilexical grammars and a cubic-time probabilistic parser. In *Proceedings of the International Workshop on Parsing Technologies.*

David M. Magerman. 1994. *Natural Language Parsing as Statistical Pattern Recognition.* Ph.D. thesis, Department of Computer Science, Stanford University.

Adwait Ratnaparkhi. 1997. A linear oberved time statistical parser based on maximum entropy models. In Claire Cardie and Ralph Weischedel, editors, *Second Conference on Empirical Methods in Natural Language Processing (EMNLP-2)*, Somerset, New Jersey. Association For Computational Linguistics.

Daniel J. Rosenkrantz and Philip M. Lewis II. 1970. Deterministic left corner parser. In *IEEE Conference Record of the 11th Annual Symposium on Switching and Automata Theory*, pages 139–152.

Arto Salomaa and Matti Soittola. 1978. *Automata-Theoretic Aspects of Formal Power Series.* Springer-Verlag, New York.