PROBABILISTIC PARSING AS INTERSECTION

Mark-Jan Nederhof [*]
Faculty of Arts
University of Groningen
P.O. Box 716
NL-9700 AS Groningen
The Netherlands
`markjan@let.rug.nl`

Giorgio Satta
Department of Information Engineering
University of Padua
via Gradenigo, 6/A
I-35131 Padova
Italy
`satta@dei.unipd.it`

Abstract

We show that a well-known algorithm to compute the intersection of a context-free language and a regular language can be extended to apply to a probabilistic context-free grammar and a probabilistic finite automaton, provided the two probabilistic models are combined through multiplication. The result is a probabilistic context-free grammar that contains joint information about the original grammar and automaton.

1 Introduction

One of the foundations of the theory of tabular context-free parsing was established by [4]. They showed that the intersection of a context-free language $L_{\mathcal{G}}$ and a regular language $L_{\mathcal{M}}$ was again a context-free language. Their proof consisted in an effective construction of a context-free grammar $\mathcal{G}_\cap$ generating $L_{\mathcal{G}} \cap L_{\mathcal{M}}$ from a context-free grammar $\mathcal{G}$ generating $L_{\mathcal{G}}$ and a finite automaton $\mathcal{M}$ accepting $L_{\mathcal{M}}$.

By taking the finite automaton $\mathcal{M}$ to be of a special kind, accepting only a single string, this result anticipated many of the early tabular parsing algorithms, such as CKY parsing [2], Earley's algorithm [16] and the algorithm for tabular simulation of pushdown automata from [19, 6]. Techniques for parsing of word graphs used in speech recognition systems [3] are based on the same theoretical framework.

A very useful property of the construction by [4] is that the resulting grammar $\mathcal{G}_\cap$ is structured similarly to the original grammar $\mathcal{G}$. More precisely, $\mathcal{G}_\cap$ encodes the set of all parse trees, formed according to $\mathcal{G}$, of strings in $L_{\mathcal{G}} \cap L_{\mathcal{M}}$, and individual parse trees can be extracted in an effective manner.

In this article we investigate the extension of the construction to probabilities. We show that for two probability distributions $p_{\mathcal{G}}$ and $p_{\mathcal{M}}$ on strings, one described by a probabilistic context-free grammar $\mathcal{G}$ and one described by a probabilistic finite automaton $\mathcal{M}$, there is a probability distribution $p_\cap$ defined by $p_\cap(w) = \frac{1}{C} \cdot p_{\mathcal{G}}(w) \cdot p_{\mathcal{M}}(w)$, for an appropriate constant C, such that $p_\cap$ is described by a probabilistic context-free grammar $\mathcal{G}_\cap$. Moreover, each derivation $d_\cap$ in $\mathcal{G}_\cap$ can be related to a derivation d in $\mathcal{G}$ and a computation c in $\mathcal{M}$, such that $p_\cap(d_\cap) = \frac{1}{C} \cdot p_{\mathcal{G}}(d) \cdot p_{\mathcal{M}}(c)$, where the probability functions $p_\cap, p_{\mathcal{G}}, p_{\mathcal{M}}$ on derivations or

[*]Supported by the Royal Netherlands Academy of Arts and Sciences. Secondary affiliation is the German Research Center for Artificial Intelligence (DFKI).

computations are related to probability functions $p_\cap, p_\mathcal{G}, p_\mathcal{M}$ on strings in a natural way, as will become clear in the sequel.

Our work is motivated by the increasing interest in models that combine submodels of context-free power dealing with global structure and submodels of finite-state power dealing with local constraints within short sequences of words or parts of speech, as exemplified by [21, 23, 5, 25]. A different but related way to combine constraints on structure and constraints on words is the lexicalization of context-free models [13, 11, 10].

There are two aspects of our approach that together distinguish it from others. First, we assume that probabilistic grammar $\mathcal{G}$ and probabilistic finite automaton $\mathcal{M}$ are developed independently; the probabilities can be determined e.g. by supervised or unsupervised learning [22]. When we combine $\mathcal{G}$ and $\mathcal{M}$ into $\mathcal{G}_\cap$, we do not use any additional training material. Training $\mathcal{G}$ and $\mathcal{M}$ separately may avoid the scarce data problem that could result from training a combined model $\mathcal{G}_\cap$ directly.

Secondly, the resulting grammar $\mathcal{G}_\cap$ has elegant properties such as properness and consistency. These can be useful for identifying 'optimal' derivations, i.e., those with the highest probability. Typical algorithms for this that rely on techniques such as best-first parsing [9] and beam search [8] build derivations incrementally, connecting parts of derivations into larger parts if they seem promising to eventually grow into the optimal derivation, or one that approximates it. In order to estimate how promising partial derivations are, it may be advantageous to have the set of all derivations encoded in the form of a single, normalized model such as $\mathcal{G}_\cap$.

Our construction of $\mathcal{G}_\cap$ out of $\mathcal{G}$ and $\mathcal{M}$ heavily relies on renormalization of probabilistic context-free grammars, by means of an algorithm that has been described before but, as far as we know, never in a very explicit form. The existence of renormalization was stated on p. 149 of [12], but the construction itself and the proof of correctness were embedded into the more general, but less transparent treatment of Gibbs distributions. Also [1] treated renormalization, but only inside the proof of their Lemma 5, and crucial aspects were not made explicit. Furthermore their proof was incomplete.

The paper is organized as follows. After giving standard definitions in Section 2, we present an algorithm to renormalize probabilistic context-free grammars in Section 3, with proofs of correctness. Section 4 discusses the combination of a probabilistic (or more generally, weighted) context-free grammar and a probabilistic (or weighted) finite automaton. An application to computation of infix probabilities is discussed in Section 5. A different way of combining probabilistic grammatical models is investigated in Section 6.

2 Preliminaries

Many of the definitions on weighted and probabilistic context-free grammars are based on [26, 7] and the definitions on weighted and probabilistic finite automata are based on [24, 29].

A *weighted context-free grammar* (WCFG) $\mathcal{G}$ is a 5-tuple (Σ, N, S, R, μ), where Σ and N are two finite disjoint sets of *terminals* and *nonterminals*, respectively, $S \in N$ is the *start symbol*, R is a finite set of *rules*, each of the form $A \rightarrow \alpha$, where $A \in N$ and $\alpha \in (\Sigma \cup N)^*$, and μ is a function from rules in R to non-negative real numbers.

In what follows, symbol a ranges over the set Σ, symbols w, v range over the set Σ^*, symbols A, B range over the set N, symbol X ranges over the set $\Sigma \cup N$, symbols α, β, γ range over the set $(\Sigma \cup N)^*$, symbol π ranges over the set R, and symbols d, e range over the set R^*. With slight abuse of notation, we treat a rule $\pi = (A \to \alpha) \in R$ as an *atomic* symbol when it occurs within a string $d\pi e \in R^*$. The symbol ϵ denotes the empty string. String concatenation is represented by operator $\cdot$ or by empty space.

For a fixed WCFG $\mathcal{G}$, we define the relation $\Rightarrow$ on triples consisting of two strings $\alpha, \beta \in (\Sigma \cup N)^*$ and a rule $\pi \in R$ by: $\alpha \overset{\pi}{\Rightarrow} \beta$ if and only if α is of the form $wA\delta$ and β is of the form $w\gamma\delta$, for some $w \in \Sigma^*$ and $\delta \in (\Sigma \cup N)^*$, and $\pi = (A \to \gamma)$. A *left-most derivation* (in $\mathcal{G}$) is a string $d = \pi_1 \cdots \pi_m$, $m \geq 0$, such that $\alpha_0 \overset{\pi_1}{\Rightarrow} \alpha_1 \overset{\pi_2}{\Rightarrow} \cdots \overset{\pi_m}{\Rightarrow} \alpha_m$, for some $\alpha_0, \ldots, \alpha_m \in (\Sigma \cup N)^*$; $d = \epsilon$ is always a left-most derivation. In the remainder of this paper, we will let the term 'derivation' refer to 'left-most derivation', unless specified otherwise. If $\alpha_0 \overset{\pi_1}{\Rightarrow} \cdots \overset{\pi_m}{\Rightarrow} \alpha_m$ for some $\alpha_0, \ldots, \alpha_m \in (\Sigma \cup N)^*$, then we say that $d = \pi_1 \cdots \pi_m$ *derives* α_m from α_0 and we write $\alpha_0 \overset{d}{\Rightarrow} \alpha_m$; $d = \epsilon$ derives any $\alpha_0 \in (\Sigma \cup N)^*$ from itself.

For $\alpha, \beta \in (\Sigma \cup N)^*$ and $d = \pi_1 \cdots \pi_m \in R^*$, $m \geq 0$, we define $\mu(\alpha \overset{d}{\Rightarrow} \beta) = \prod_{i=1}^{m} \mu(\pi_i)$ if $\alpha \overset{d}{\Rightarrow} \beta$, and $\mu(\alpha \overset{d}{\Rightarrow} \beta) = 0$ otherwise. The weight $\mu(w)$ of a string $w \in \Sigma^*$ is defined to be $\sum_d \mu(S \overset{d}{\Rightarrow} w)$.

A WCFG $\mathcal{G} = (\Sigma, N, S, R, \mu)$ that is such that μ is a function from R to real numbers in the range $[0, 1]$ is said to be a *probabilistic context-free grammar* (PCFG). A PCFG is said to be *proper* if $\sum_{\pi,\alpha} \mu(A \overset{\pi}{\Rightarrow} \alpha) = 1$ for all $A \in N$, i.e., if the weights of all rules $\pi = (A \to \alpha)$ with left-hand side A sum to 1. A WCFG is said to be *convergent* if $\sum_{d,w} \mu(S \overset{d}{\Rightarrow} w) < \infty$.

A WCFG is said to be *consistent* if $\sum_{d,w} \mu(S \overset{d}{\Rightarrow} w) = 1$. Consistency implies that the WCFG defines a probability distribution on the set of terminal strings. Although consistency is often considered in the context of PCFGs, we may also consider consistent WCFGs that are not PCFGs. There is a practical sufficient condition for consistency that is decidable [7].

A WCFG is said to be *reduced* if for each nonterminal A there are $d_1, d_2 \in R^*$, $w_1, w_2 \in \Sigma^*$ and $\beta \in (\Sigma \cup N)^*$ such that $\mu(S \overset{d_1}{\Rightarrow} w_1 A \beta) \cdot \mu(w_1 A \beta \overset{d_2}{\Rightarrow} w_1 w_2) > 0$. In words, if a WCFG is reduced, then for each nonterminal A, there is at least one derivation $d_1 d_2$ with weight strictly larger than 0 that derives a string $w_1 w_2$ from S and that includes some rule with left-hand side A. A WCFG that is not reduced can be turned into one that is reduced and that assigns the same weights to strings, provided in the original WCFG we have $\sum_w \mu(w) > 0$. This *reduction* consists in removing from the grammar any nonterminal A for which the above conditions do not hold, together with any rule that contains such a nonterminal; see [2] for reduction of context-free grammars, which is very similar.

A *weighted finite automaton* (WFA) $\mathcal{M}$ is a 6-tuple $(\Sigma, Q, q_0, q_f, T, \nu)$, where Σ and Q are two finite disjoint sets of *terminals* and *states*, respectively, $q_0, q_f \in Q$ are the *initial* and *final* states, respectively, T is a finite set of *transitions*, each of the form $r \overset{a}{\mapsto} s$, where $r, s \in Q$ and $a \in \Sigma$, and ν is a function from transitions in T to non-negative real numbers.[1]

In what follows, symbols q, r, s range over the set Q, symbol τ ranges over the set T, and symbol c ranges over the set T^*.

For a fixed WFA, we define a *configuration* to be an element of $Q \times \Sigma^*$, and we define the relation $\vdash$ on triples consisting of two configurations and a transition $\tau \in T$ by: $(r, w) \overset{\tau}{\vdash} (s, w')$

[1]That we only allow one final state is not a serious restriction with regard to the set of strings we can process; only when the empty string is to be recognized could this lead to difficulties. Lifting the restriction would encumber the presentation with treatment of additional cases, without affecting however the validity of the main results.

if and only if w is of the form aw', for some $a \in \Sigma$, and $\tau = (r \overset{a}{\mapsto} s)$. A *computation* (in $\mathcal{M}$) is a string $c = \tau_1 \cdots \tau_m$, $m \geq 0$, such that $(r_0, w_0) \overset{\tau_1}{\vdash} (r_1, w_1) \overset{\tau_2}{\vdash} \cdots \overset{\tau_m}{\vdash} (r_m, w_m)$, for some (r_0, w_0), $\ldots$, $(r_m, w_m) \in Q \times \Sigma^*$; $c = \epsilon$ is always a computation. If $(r_0, w_0) \overset{\tau_1}{\vdash} \cdots \overset{\tau_m}{\vdash} (r_m, w_m)$ for some (r_0, w_0), $\ldots$, $(r_m, w_m) \in Q \times \Sigma^*$ and $c = \tau_1 \cdots \tau_m \in T^*$, then we write $(r_0, w_0) \overset{c}{\vdash} (r_m, w_m)$. We say that c *recognizes* w if $(q_0, w) \overset{c}{\vdash} (q_f, \epsilon)$.

For $(q, w), (s, v) \in Q \times \Sigma^*$ and $c = \tau_1 \cdots \tau_m \in T^*$ we define $\nu((q, w) \overset{c}{\vdash} (s, v)) = \prod_{i=1}^{m} \nu(\tau_i)$ if $(q, w) \overset{c}{\vdash} (s, v)$, and $\nu((q, w) \overset{c}{\vdash} (s, v)) = 0$ otherwise. The weight $\nu(w)$ of a string $w \in \Sigma^*$ is defined to be $\sum_c \nu((q_0, w) \overset{c}{\vdash} (q_f, \epsilon))$.

A WFA $\mathcal{M} = (\Sigma, Q, q_0, q_f, T, \nu)$ that is such that ν is a function from T to real numbers in the range $[0, 1]$ is said to be a *probabilistic finite automaton* (PFA). A WFA is said to be *convergent* if $\sum_{c,w} \nu((q_0, w) \overset{c}{\vdash} (q_f, \epsilon)) < \infty$.

There are no essential differences between PFAs and Hidden Markov Models (HMMs) [22]. Although the term 'Hidden Markov Model' and its common definition are used more frequently in areas such as language modelling, we have chosen to base our study on PFAs, since they have slight notational advantages over HMMs for describing the algorithms to be developed in the following sections.

3 Renormalization

For much of the results elaborated here, a sketch was presented inside the proof of Lemma 5 from [1]. Similar results, embedded in a discussion of Gibbs distributions, are due to [12].

Lemma 1 *For each WCFG $\mathcal{G} = (\Sigma, N, S, R, \mu)$ that is reduced and convergent and for each $\alpha \in (\Sigma \cup N)^*$,*

$$0 \; < \; \sum_{d,w} \mu(\alpha \overset{d}{\Rightarrow} w) \; < \; \infty.$$

Proof. Let $\alpha = X_1 \cdots X_m$, where $X_1, \ldots, X_m \in (\Sigma \cup N)$ and $0 \leq m$. Then

$$\begin{aligned} \sum_{d,w} \mu(\alpha \overset{d}{\Rightarrow} w) &= \sum_{d_1,\ldots,d_m,w_1,\ldots,w_m} \prod_{1 \leq i \leq m} \mu(X_i \overset{d_i}{\Rightarrow} w_i) \\ &= \prod_{1 \leq i \leq m} \sum_{d_i,w_i} \mu(X_i \overset{d_i}{\Rightarrow} w_i). \end{aligned}$$

Assume that $\sum_{d,w} \mu(\alpha \overset{d}{\Rightarrow} w) = \infty$. Then there is at least one i, $1 \leq i \leq m$, such that $\sum_{d_i,w_i} \mu(X_i \overset{d_i}{\Rightarrow} w_i) = \infty$. Since $\mathcal{G}$ is reduced, there are $e_1, e \in R^*$, $v_1, v \in \Sigma^*$ and $\beta \in (\Sigma \cup N)^*$ such that $\mu(S \overset{e_1}{\Rightarrow} v_1 X_i \beta) \cdot \mu(v_1 X_i \beta \overset{e}{\Rightarrow} v_1 v) > 0$.

Let e_2, e_3 and v_2, v_3 be such that $e_2 e_3 = e$, $v_2 v_3 = v$, $X_i \overset{e_2}{\Rightarrow} v_2$, and $\beta \overset{e_3}{\Rightarrow} v_3$. Since $\mu(v_1 X_i \beta \overset{e}{\Rightarrow} v_1 v) > 0$, also $\mu(\beta \overset{e_3}{\Rightarrow} v_3) > 0$. Note that if $X_i \overset{d_i}{\Rightarrow} w_i$, for some d_i and w_i, then $S \overset{e_1 d_i e_3}{\Rightarrow} v_1 w_i v_3$. Hence,

$$\begin{aligned} \sum_{d,w} \mu(S \overset{d}{\Rightarrow} w) &\geq \sum_{d_i,w_i} \mu(S \overset{e_1 d_i e_3}{\Rightarrow} v_1 w_i v_3) \\ &= \mu(S \overset{e_1}{\Rightarrow} v_1 X_i \beta) \cdot \sum_{d_i,w_i} \mu(X_i \overset{d_i}{\Rightarrow} w_i) \cdot \mu(\beta \overset{e_3}{\Rightarrow} v_3) \\ &= \infty. \end{aligned}$$

This is in contradiction with convergence of $\mathcal{G}$. Therefore, $\sum_{d,w} \mu(\alpha \overset{d}{\Rightarrow} w) < \infty$.

Now assume that $\sum_{d,w} \mu(\alpha \overset{d}{\Rightarrow} w) = 0$. Then there is at least one i such that $\sum_{d_i,w_i} \mu(X_i \overset{d_i}{\Rightarrow} w_i) = 0$ and $\mu(X_i \overset{d_i}{\Rightarrow} w_i) = 0$, for all d_i and w_i. This means that there are no e_1, e, v_1, v and β such that $\mu(S \overset{e_1}{\Rightarrow} v_1 X_i \beta) \cdot \mu(v_1 X_i \beta \overset{e}{\Rightarrow} v_1 v) > 0$. This is in contradiction with the assumption that $\mathcal{G}$ is reduced. Therefore, $\sum_{d,w} \mu(\alpha \overset{d}{\Rightarrow} w) > 0$. ■

In the remainder of this section, let $\mathcal{G} = (\Sigma, N, S, R, \mu)$ be a fixed reduced and convergent WCFG.

Definition 2 *We define the WCFG $\mathcal{R}(\mathcal{G})$ to be (Σ, N, S, R, p), where p is defined by*

$$p(\pi) \quad = \quad \frac{\mu(\pi) \cdot \sum_{d,w} \mu(\alpha \overset{d}{\Rightarrow} w)}{\sum_{d',w'} \mu(A \overset{d'}{\Rightarrow} w')}$$

for each rule $\pi = (A \to \alpha) \in R$.

We refer to the application of $\mathcal{R}$ to a WCFG as *renormalization*.

Lemma 3 *$\mathcal{R}(\mathcal{G})$ is reduced.*

Proof. This follows directly from the definition of $\mathcal{R}$ and Lemma 1. ■

Lemma 4 *$\mathcal{R}(\mathcal{G})$ is a proper PCFG.*

Proof. Let A be a nonterminal in N. Then,

$$\begin{aligned}
\sum_{\pi,\alpha} p(A \overset{\pi}{\Rightarrow} \alpha) \quad &= \quad \sum_{\pi=(A\to\alpha)} \frac{\mu(\pi) \cdot \sum_{d,w} \mu(\alpha \overset{d}{\Rightarrow} w)}{\sum_{d',w'} \mu(A \overset{d'}{\Rightarrow} w')} \\
&= \quad \frac{\sum_{\pi,\alpha} \left(\mu(A \overset{\pi}{\Rightarrow} \alpha) \cdot \sum_{d,w} \mu(\alpha \overset{d}{\Rightarrow} w)\right)}{\sum_{d',w'} \mu(A \overset{d'}{\Rightarrow} w')} \\
&= \quad \frac{\sum_{\pi,d,w} \mu(A \overset{\pi d}{\Rightarrow} w)}{\sum_{d',w'} \mu(A \overset{d'}{\Rightarrow} w')} \quad = \quad 1.
\end{aligned}$$

Since the definition of p implies that $p(\pi)$ is non-negative for each rule π, it follows that $0 \leq p(\pi) = p(A \overset{\pi}{\Rightarrow} \alpha) \leq 1$ for each rule $\pi = (A \to \alpha)$. ■

Lemma 5 *For each $\beta \in (\Sigma \cup N)^*$, each $d \in R^*$ and each $w \in \Sigma^*$,*

$$p(\beta \overset{d}{\Rightarrow} w) \quad = \quad \frac{\mu(\beta \overset{d}{\Rightarrow} w)}{\sum_{d',w'} \mu(\beta \overset{d'}{\Rightarrow} w')}. \tag{1}$$

Proof. The proof is by induction on pairs $(|d|, |\beta|)$, containing the length $|d|$ of d and the length $|\beta|$ of β, where we define $(|d_1|, |\beta_1|) < (|d_2|, |\beta_2|)$ iff $|d_1| < |d_2|$, or $|d_1| = |d_2|$ and $|\beta_1| < |\beta_2|$.

First note that by Lemma 1, the denominator in (1) is a finite positive number. Secondly, note that if d does not derive w from β, then both $p(\beta \overset{d}{\Rightarrow} w)$ and $\mu(\beta \overset{d}{\Rightarrow} w)$ are 0, and the result follows. Otherwise, we distinguish the following three cases.

Case 1. Assume $\beta = \epsilon$ or $\beta = a \in \Sigma$. Then we must have $d = \epsilon$ and $w = \beta$, and $p(\beta \overset{d}{\Rightarrow} w)$ and $\frac{\mu(\beta \overset{d}{\Rightarrow} w)}{\sum_{d',w'} \mu(\beta \overset{d'}{\Rightarrow} w')}$ are both 1.

Case 2. Assume $\beta = \beta_1\beta_2$, for some $\beta_1 \neq \epsilon$ and $\beta_2 \neq \epsilon$. Then there must be $d_1, d_2 \in R^*$ and $w_1, w_2 \in \Sigma^*$ such that $d = d_1 d_2$, $w = w_1 w_2$, and $\beta_1 \overset{d_1}{\Rightarrow} w_1$ and $\beta_2 \overset{d_2}{\Rightarrow} w_2$. Using the induction hypothesis twice, we obtain:

$$\begin{aligned}
p(\beta \overset{d}{\Rightarrow} w) &= p(\beta_1 \overset{d_1}{\Rightarrow} w_1) \cdot p(\beta_2 \overset{d_2}{\Rightarrow} w_2) \\
&= \frac{\mu(\beta_1 \overset{d_1}{\Rightarrow} w_1)}{\sum_{d'_1, w'_1} \mu(\beta_1 \overset{d'_1}{\Rightarrow} w'_1)} \cdot \frac{\mu(\beta_2 \overset{d_2}{\Rightarrow} w_2)}{\sum_{d'_2, w'_2} \mu(\beta_2 \overset{d'_2}{\Rightarrow} w'_2)} \\
&= \frac{\mu(\beta \overset{d}{\Rightarrow} w)}{\sum_{d', w'} \mu(\beta \overset{d'}{\Rightarrow} w')}.
\end{aligned}$$

Case 3. Assume $\beta = A \in N$. This means that $d = \pi d_0$ for some rule $\pi = (A \to \alpha)$ and derivation d_0. By using the definition of p and the induction hypothesis, we obtain:

$$\begin{aligned}
p(A \overset{\pi d_0}{\Rightarrow} w) &= p(A \overset{\pi}{\Rightarrow} \alpha) \cdot p(\alpha \overset{d_0}{\Rightarrow} w) \\
&= \frac{\mu(A \overset{\pi}{\Rightarrow} \alpha) \cdot \sum_{d'', w''} \mu(\alpha \overset{d''}{\Rightarrow} w'')}{\sum_{d', w'} \mu(A \overset{d'}{\Rightarrow} w')} \cdot \frac{\mu(\alpha \overset{d_0}{\Rightarrow} w)}{\sum_{d'', w''} \mu(\alpha \overset{d''}{\Rightarrow} w'')} \\
&= \frac{\mu(A \overset{\pi}{\Rightarrow} \alpha) \cdot \mu(\alpha \overset{d_0}{\Rightarrow} w)}{\sum_{d', w'} \mu(A \overset{d'}{\Rightarrow} w')} \\
&= \frac{\mu(A \overset{\pi d_0}{\Rightarrow} w)}{\sum_{d', w'} \mu(A \overset{d'}{\Rightarrow} w')}. \quad \blacksquare
\end{aligned}$$

Corollary 6 *$\mathcal{R}(\mathcal{G})$ is consistent.*

Proof. Due to Lemma 5,

$$\begin{aligned}
\sum_{d,w} p(S \overset{d}{\Rightarrow} w) &= \sum_{d,w} \frac{\mu(S \overset{d}{\Rightarrow} w)}{\sum_{d', w'} \mu(S \overset{d'}{\Rightarrow} w')} \\
&= \frac{\sum_{d,w} \mu(S \overset{d}{\Rightarrow} w)}{\sum_{d', w'} \mu(S \overset{d'}{\Rightarrow} w')} = 1. \quad \blacksquare
\end{aligned}$$

Recapitulating the preceding results, applying Lemma 5 for $\beta = S$, we obtain the following.

Theorem 7 *For each WCFG $\mathcal{G} = (\Sigma, N, S, R, \mu)$ that is reduced and convergent, $\mathcal{R}(\mathcal{G}) = (\Sigma, N, S, R, p)$ is a reduced, proper and consistent PCFG, and $p(S \overset{d}{\Rightarrow} w) = \frac{1}{C} \cdot \mu(S \overset{d}{\Rightarrow} w)$, for each d and w, where $C = \sum_{d', w'} \mu(S \overset{d'}{\Rightarrow} w')$.*

It follows that renormalization preserves the ratios between weights of derivations. Amongst other things, this implies that for each string, the derivation with the highest weight according to $\mathcal{G}$ equals the most probable derivation according to $\mathcal{R}(\mathcal{G})$. For parsing algorithms that rely on properness and consistency, the renormalized grammar may be more suitable however than the original one.

Properness and consistency may play a role in parsers that use a priority queue to guide the parsing process. The ranking that parsing items obtain in this queue may be based on the weights computed from the applied rules. When such weights are renormalized, they may reflect the probability of parsing items more accurately, and thereby guide the parsing process more effectively.

It is interesting to consider renormalization for the special cases of WCFGs $\mathcal{G}$ that are proper but not consistent, or consistent but not proper. In the latter case, $C = 1$ in Theorem 7, and therefore we obtain:

Corollary 8 *For each WCFG $\mathcal{G} = (\Sigma, N, S, R, \mu)$ that is reduced and consistent, $\mathcal{R}(\mathcal{G}) = (\Sigma, N, S, R, p)$ is a reduced, proper and consistent PCFG, and $p(S \overset{d}{\Rightarrow} w) = \mu(S \overset{d}{\Rightarrow} w)$, for each d and w.*

This means that the expressive power of proper and consistent PCFGs is no more restricted than that of WCFGs that are consistent but not necessarily proper. Both subformalisms of the WCFGs can describe the same set of probability distributions on derivations and on strings.[2]

We still need to explain how to compute the values of expressions of the form $\sum_{d,w} \mu(A \overset{d}{\Rightarrow} w)$, which we need to compute the probabilities of rules in Definition 2. Although we cannot hope to obtain closed-form expressions in general, we can approximate these values.[3] The required definitions, presented below, have been adopted from [7].

First, for each $A \in N$ let m_A be the number of rules in R with left-hand side A, and let the i-th rule $\pi_{A,i}$, $1 \leq i \leq m_A$, be of the form $A \to X_{i,1} \cdots X_{i,m_{A,i}}$, where $m_{A,i}$ is the length of the right-hand side of $\pi_{A,i}$.

Now, define $\mu_k(a) = 1$ for each $a \in \Sigma$ and integer $k \geq 0$, and let $\mu_0(A) = 0$ for each $A \in N$. Now define for $k = 0, 1, 2, \ldots$ and each $A \in N$:

$$\mu_{k+1}(A) = \sum_{1 \leq i \leq m_A} \Big(\mu(\pi_{A,i}) \cdot \prod_{1 \leq j \leq m_{A,i}} \mu_k(X_{i,j})\Big).$$

If the grammar is convergent, then the value of $\mu_k(A)$ for $k = 0, 1, 2, \ldots$ converges to $\sum_{d,w} \mu(A \overset{d}{\Rightarrow} w)$, and for sufficiently large k, this can be taken as a suitable approximation of $\sum_{d,w} \mu(A \overset{d}{\Rightarrow} w)$. Further, note that $\sum_{d,w} \mu(a \overset{d}{\Rightarrow} w) = 1$ for each $a \in \Sigma$ and $\sum_{d,w} \mu(\alpha_1\alpha_2 \overset{d}{\Rightarrow} w) = \sum_{d,w} \mu(\alpha_1 \overset{d}{\Rightarrow} w) \cdot \sum_{d,w} \mu(\alpha_2 \overset{d}{\Rightarrow} w)$ for each $\alpha_1, \alpha_2 \in (\Sigma \cup N)^*$.

4 Weighted intersection

In this section we investigate an extension of the construction from [4] that computes the intersection of a context-free language and a regular language. We will refer to this extended construction as *weighted intersection* (of a weighted context-free language and a weighted regular language). The input consists of a WCFG $\mathcal{G} = (\Sigma, N, S, R, \mu)$ and a WFA $\mathcal{M} = (\Sigma, Q, q_0, q_f, T, \nu)$; note that we assume, without loss of generality, that $\mathcal{G}$ and $\mathcal{M}$ share the same set of terminals Σ.

The output of the construction is WCFG $\mathcal{G}_\cap = (\Sigma, N_\cap, S_\cap, R_\cap, \mu_\cap)$, where $N_\cap = Q \times (\Sigma \cup N) \times Q$, $S_\cap = (q_0, S, q_f)$, and $R_\cap$ consists of the set of rules that is obtained as follows.

- For each rule $\pi = (A \to X_1 \cdots X_m) \in R$, $m \geq 0$, and each sequence of states $r_0, \ldots, r_m \in Q$, let the rule $\pi_\cap = ((r_0, A, r_m) \to (r_0, X_1, r_1) \cdots (r_{m-1}, X_m, r_m))$ be in $R_\cap$, and let $\mu_\cap(\pi_\cap) = \mu(\pi)$; for $m = 0$, $R_\cap$ contains a rule $\pi_\cap = ((r_0, A, r_0) \to \epsilon)$ for each state r_0.

[2]We owe this observation to fruitful discussions with Detlef Prescher.

[3]We can compute the exact values if the grammar is non-recursive however, and in fact, for the use of renormalization to be discussed in the following section, the grammar may well be non-recursive for many practical applications.

- For each transition $\tau = (r \overset{a}{\mapsto} s) \in T$, let the rule $\pi_\cap = ((r, a, s) \to a)$ be in $R_\cap$, and let $\mu_\cap(\pi_\cap) = \nu(\tau)$.

Note that for each rule $(r_0, A, r_m) \to (r_0, X_1, r_1) \cdots (r_{m-1}, X_m, r_m)$ from $R_\cap$ there is a unique rule $A \to X_1 \cdots X_m$ from R from which it has been constructed by the above. Similarly, each rule $(r, a, s) \to a$ uniquely identifies a transition $r \overset{a}{\mapsto} s$. This means that if we take a derivation $d_\cap$ in $\mathcal{G}_\cap$, we can divide it into a sequence $h_1(d_\cap)$ of rules from $\mathcal{G}$ and a sequence $h_2(d_\cap)$ of transitions from $\mathcal{M}$, where h_1 and h_2 are string homomorphisms that we define point-wise as:

$$\begin{aligned}
h_1(\pi_\cap) &= \pi \quad \text{if } \pi_\cap = ((r_0, A, r_m) \to (r_0, X_1, r_1) \cdots (r_{m-1}, X_m, r_m)) \text{ and } \pi = (A \to X_1 \cdots X_m) \\
&\ \epsilon \quad \text{if } \pi_\cap = ((r, a, s) \to a) \\
h_2(\pi_\cap) &= \tau \quad \text{if } \pi_\cap = ((r, a, s) \to a) \text{ and } \tau = (r \overset{a}{\mapsto} s) \\
&\ \epsilon \quad \text{if } \pi_\cap = ((r_0, A, r_m) \to (r_0, X_1, r_1) \cdots (r_{m-1}, X_m, r_m))
\end{aligned}$$

We define $h(d_\cap) = (h_1(d_\cap), h_2(d_\cap))$. It can be easily seen that if $h(d_\cap) = (d, c)$ and $S_\cap \overset{d_\cap}{\Rightarrow} w$, then for the same w we have $S \overset{d}{\Rightarrow} w$ and $(q_0, w) \overset{c}{\vdash} (q_f, \epsilon)$. Conversely, if for some w, d and c we have $S \overset{d}{\Rightarrow} w$ and $(q_0, w) \overset{c}{\vdash} (q_f, \epsilon)$, then there is a derivation $d_\cap$ such that $h(d_\cap) = (d, c)$ and $S_\cap \overset{d_\cap}{\Rightarrow} w$.

We also define $\mathcal{D}_\mathcal{G}(w) = \{d \mid S \overset{d}{\Rightarrow} w\}$, $\mathcal{D}_\mathcal{M}(w) = \{c \mid (q_0, w) \overset{c}{\vdash} (q_f, \epsilon)\}$, $\mathcal{D}_\cap(w) = \{d_\cap \mid S_\cap \overset{d_\cap}{\Rightarrow} w\}$. The following can now be stated without further proof:

Lemma 9 *By restricting its domain, h becomes a bijection from $\mathcal{D}_\cap(w)$ to $\mathcal{D}_\mathcal{G}(w) \times \mathcal{D}_\mathcal{M}(w)$, for each $w \in \Sigma^*$. Furthermore, $\mu_\cap(S_\cap \overset{d_\cap}{\Rightarrow} w) = \mu(S \overset{h_1(d_\cap)}{\Rightarrow} w) \cdot \nu((q_0, w) \overset{h_2(d_\cap)}{\vdash} (q_f, \epsilon))$, for each $w \in \Sigma^*$ and $d_\cap \in \mathcal{D}_\cap(w)$.*

If we are only interested in the weights assigned to strings, then we can use the following:

Lemma 10 *For each $w \in \Sigma^*$, we have $\mu_\cap(w) = \mu(w) \cdot \nu(w)$.*

Proof. Due to the existence of h, with the properties stated in Lemma 9:

$$\mu_\cap(w) \;=\; \sum_{d_\cap} \mu_\cap(S_\cap \overset{d_\cap}{\Rightarrow} w) \;=\; \sum_{d} \mu(S \overset{d}{\Rightarrow} w) \cdot \sum_{c} \nu((q_0, w) \overset{c}{\vdash} (q_f, \epsilon)) \;=\; \mu(w) \cdot \nu(w). \quad \blacksquare$$

Before we can apply renormalization to $\mathcal{G}_\cap$, we need to ensure convergence:

Lemma 11 *If $\mathcal{G}$ and $\mathcal{M}$ are convergent, then so is $\mathcal{G}_\cap$.*

Proof. If $\mathcal{G}$ and $\mathcal{M}$ are convergent, then:

$$\begin{aligned}
\sum_{d_\cap, w} \mu_\cap(S_\cap \overset{d_\cap}{\Rightarrow} w) \;&=\; \sum_{w} \Big(\sum_{d} \mu(S \overset{d}{\Rightarrow} w) \cdot \sum_{c} \nu((q_0, w) \overset{c}{\vdash} (q_f, \epsilon)) \Big) \\
&\leq\; \sum_{d, w} \mu(S \overset{d}{\Rightarrow} w) \cdot \sum_{c, w} \nu((q_0, w) \overset{c}{\vdash} (q_f, \epsilon)) \;<\; \infty. \quad \blacksquare
\end{aligned}$$

We now come to the main result of this paper, which can be seen as a probabilistic extension of the aforementioned theorem by [4]:

Theorem 12 *For any convergent WCFG $\mathcal{G} = (\Sigma, N, S, R, \mu)$ and convergent WFA $\mathcal{M} = (\Sigma, Q, q_0, q_f, T, \nu)$ such that $\sum_w (\mu(w) \cdot \nu(w)) > 0$, there are:*
(i) *a reduced, proper and consistent PCFG $\mathcal{G}''_\cap = (\Sigma, N'_\cap, S_\cap, R'_\cap, \mu''_\cap)$,*
(ii) *a function h defined by $h(d_\cap) = (h_1(d_\cap), h_2(d_\cap))$, where h_1 and h_2 are two string homomorphisms, such that by restricting the domain of h, it becomes a bijection from $\mathcal{D}_\cap(w)$ to $\mathcal{D}_\mathcal{G}(w) \times \mathcal{D}_\mathcal{M}(w)$, for each $w \in \Sigma^*$,*

(iii) *a constant C,*

such that $\mu''_\cap(S_\cap \overset{d_\cap}{\Rightarrow} w) = \frac{1}{C} \cdot \mu(S \overset{h_1(d_\cap)}{\Rightarrow} w) \cdot \nu((q_0, w) \overset{h_2(d_\cap)}{\vdash} (q_f, \epsilon))$, for each $d_\cap \in (R'_\cap)^*$ and $w \in \Sigma^*$.

Proof. By the construction at the beginning of this section, we can obtain a WCFG $\mathcal{G}_\cap = (\Sigma, N_\cap, S_\cap, R_\cap, \mu_\cap)$ and a function h with the properties stated in Lemmas 9 and 11. Since $\sum_w(\mu(w) \cdot \nu(w)) > 0$, also $\sum_w \mu_\cap(w) > 0$ by Lemma 10. This means $\mathcal{G}_\cap$ can be reduced; let the reduced grammar be $\mathcal{G}'_\cap = (\Sigma, N'_\cap, S_\cap, R'_\cap, \mu'_\cap)$. We can now apply Theorem 7 to obtain a renormalized PCFG $\mathcal{G}''_\cap = (\Sigma, N'_\cap, S_\cap, R'_\cap, \mu''_\cap)$, with the required properties. ■

Our results carry over to weighted intersection of probabilistic tree-adjoining grammars [27] and WFAs. We can also extend our results in a trivial way to weighted intersection of a pair of WCFGs, one of which is non-recursive; apart from renormalization, this is shown by [14].

5 Infix probabilities

One may apply the construction from Section 4 also when its input grammar $\mathcal{G}$ is a WCFG but its input automaton $\mathcal{M}$ is a (non-weighted) finite automaton, or when $\mathcal{G}$ is a (non-weighted) context-free grammar and $\mathcal{M}$ is a WFA. This is possible by treating a finite automaton as a WFA of which all transitions have weight 1, and a context-free grammar as a WCFG of which all rules have weight 1. For an application of weighted intersection of a WFA and a (non-recursive) context-free grammar, see [20].

Let us now investigate the case more closely that we have an input WCFG $\mathcal{G} = (\Sigma, N, S, R, \mu)$ and an input automaton $\mathcal{M} = (\Sigma, Q, q_0, q_f, T, \mathbf{1})$, where $\mathbf{1}$ is defined by $\mathbf{1}(\tau) = 1$ for all $\tau \in T$. Now, $\mathbf{1}(w)$ equals the number of computations c in $\mathcal{M}$ such that $(q_0, w) \overset{c}{\vdash} (q_f, \epsilon)$. We can define the language accepted by $\mathcal{M}$ as $L(\mathcal{M}) = \{w \mid \mathbf{1}(w) \geq 1\}$.

If $\mathcal{M}$ is unambiguous, i.e., if there is at most one computation that recognizes a given string w, then $\mathbf{1}(w)$ can only be 0 or 1, and thereby the equation $\mu_\cap(w) = \mu(w) \cdot \mathbf{1}(w)$, from Lemma 10, implies that $\mu_\cap(w) = \mu(w)$ for $w \in L(\mathcal{M})$ and $\mu_\cap(w) = 0$ for $w \notin L(\mathcal{M})$. Let us remark that a sufficient condition for $\mathcal{M}$ to be unambiguous is that $\mathcal{M}$ is deterministic.

Now consider the class of problems where we need to compute $\sum_{w \in L} \mu(w)$, for a certain WCFG $\mathcal{G}$ with weight function μ and a certain regular language L. From the above, it is clear that this problem can be solved if we can do the following:
(i) construct an unambiguous $\mathcal{M}$ with weight function $\mathbf{1}$ such that $L = L(\mathcal{M})$; *and*
(ii) compute $\sum_w \mu_\cap(w)$, which by (i) equals $\sum_{w \in L} \mu(w)$.
One instance of the above problem is the computation of prefix probabilities [17]. The *prefix probability* in a PCFG $\mathcal{G}$ of a string $v \in \Sigma^*$ is defined as $\sum_{w \in L_v} \mu(w)$, where $L_v = \{vw \mid w \in \Sigma^*\}$. One can compute prefix probabilities as indicated above, by constructing an unambiguous $\mathcal{M}$ accepting L_v, which is trivial.

Similarly, the *infix probability* in $\mathcal{G}$ of a string $v \in \Sigma^*$ is defined by [15] as $\sum_{w \in L_v} \mu(w)$, where $L_v = \{w_1 v w_2 \mid w_1, w_2 \in \Sigma^*\}$. To obtain an unambiguous $\mathcal{M}$ accepting L_v, we can straightforwardly apply the technique from [18]. This can be generalized to the *island probability* in $\mathcal{G}$ of a list of strings $v_1, v_2, \ldots, v_m \in \Sigma^*$, defined by [15] as $\sum_{w \in L_{v_1, v_2, \ldots, v_m}} \mu(w)$, where $L_{v_1, v_2, \ldots, v_m} = \{w_0 v_1 w_1 \cdots v_m w_m \mid w_0, w_1, \ldots, w_m \in \Sigma^*\}$.

The computation of quantity $\sum_w \mu_\cap(w)$ in (ii) requires some discussion. Polynomial time algorithms have been presented by [17, 30] for its computation in the case of prefix probabilities. However, it has been conjectured by [15] that no similarly efficient solution exists for the problem of infix probabilities, even assuming that the WCFG $\mathcal{G}$ is fixed, as is the case for many practical applications.

6 Related work

The results from Section 4 imply that for a PCFG $\mathcal{G}$ with probability function $p_{\mathcal{G}}$ and a PFA $\mathcal{M}$ with probability function $p_{\mathcal{M}}$ we can find a PCFG $\mathcal{G}_\cap$ with probability function $p_\cap$ such that:

$$p_\cap(w) = \frac{1}{C} \cdot p_{\mathcal{G}}(w) \cdot p_{\mathcal{M}}(w),$$

for each string w, where C is a constant determined by $\mathcal{G}$ and $\mathcal{M}$.

A different combination of context-free and finite-state models (to be exact, N-gram models) is proposed by [5], using linear interpolation. This results in a probability distribution p_+ on strings given by:

$$p_+(w) = \lambda \cdot p_{\mathcal{G}}(w) + (1-\lambda) \cdot p_{\mathcal{M}}(w),$$

for a certain constant λ between 0 and 1 that can be freely chosen.

This probability function p_+ on strings can be described as a PCFG $\mathcal{G}_+$ with probability function p_+ on rules, which is constructed as follows. First, rewrite PFA $\mathcal{M}$ to a PCFG $\mathcal{G}_{\mathcal{M}}$ that describes the same probability distribution on strings as $\mathcal{M}$. (This is possible since the Chomsky hierarchy carries over to probabilistic models [26].) Now we define the set of nonterminals of $\mathcal{G}_+$ as the disjoint union of the sets of nonterminals from $\mathcal{G}$ and $\mathcal{G}_{\mathcal{M}}$ plus the new nonterminal S_+, which becomes the start symbol of $\mathcal{G}_+$. The set of rules of $\mathcal{G}_+$ is defined as the union of the sets of rules from $\mathcal{G}$ and $\mathcal{G}_{\mathcal{M}}$, which keep the probabilities as determined by $p_{\mathcal{G}}$ and $p_{\mathcal{M}}$, plus the following two rules, with their probabilities between brackets:

$$\begin{array}{ll} S_+ \rightarrow S_{\mathcal{G}} & (\lambda) \\ S_+ \rightarrow S_{\mathcal{M}} & (1-\lambda), \end{array}$$

where $S_{\mathcal{G}}$ is the start symbol of $\mathcal{G}$ and $S_{\mathcal{M}}$ is the start symbol of $\mathcal{G}_{\mathcal{M}}$.

However, a grammar $\mathcal{G}_+$ constructed in this way does not, in general, have the kind of favourable properties we saw in Theorem 12. In particular, a derivation in $\mathcal{G}_+$ encodes either a derivation in $\mathcal{G}$ or a computation in $\mathcal{M}$, but not both.

This has implications for the problem of finding an 'optimal' string $w \in \Sigma^*$. If we may assume that $\mathcal{G}$ and $\mathcal{M}$ are unambiguous, then we may find a string w that maximizes $p_\cap(w)$ by finding the derivation $d_\cap$ in $\mathcal{G}_\cap$ that maximizes $p_\cap(d)$. However, there is no obvious way to find a string w that maximizes $p_+(w)$ without doing exhaustive parsing, computing all strings generated by $\mathcal{G}$ and all strings generated by $\mathcal{G}_{\mathcal{M}}$ separately, and summing $\lambda \cdot p_{\mathcal{G}}(w)$ and $(1-\lambda) \cdot p_{\mathcal{M}}(w)$ for each string w that is generated by either $\mathcal{G}$ or $\mathcal{G}_{\mathcal{M}}$. This seems difficult even when these two grammars are unambiguous. It is not clear however whether this problem is as difficult as some of the problems discussed by [28].

7 Conclusions

We have shown how a PCFG and a PFA can be combined into a new PCFG in a constructive way, and we have stated and proven correct a number of useful properties of this construction.

Although it seems to have been ignored in the existing literature, this construction forms a theoretical basis for much of the ongoing work on probabilistic parsing.

Acknowledgements

The first author had many fruitful discussions with Detlef Prescher about renormalization of PCFGs.

References

[1] S. Abney, D. McAllester, and F. Pereira. Relating probabilistic grammars and automata. In *37th Annual Meeting of the ACL*, pages 542–549, 1999.

[2] A.V. Aho and J.D. Ullman. *Parsing*, The Theory of Parsing, Translation and Compiling, volume 1. Prentice-Hall, 1972.

[3] H. Aust, M. Oerder, F. Seide, and V. Steinbiss. The Philips automatic train timetable information system. *Speech Communication*, 17:249–262, 1995.

[4] Y. Bar-Hillel, M. Perles, and E. Shamir. On formal properties of simple phrase structure grammars. In Y. Bar-Hillel, editor, *Language and Information: Selected Essays on their Theory and Application*, chapter 9, pages 116–150. Addison-Wesley, 1964.

[5] J.-M. Benedí and J.A. Sánchez. Combination of n-grams and stochastic context-free grammars for language modeling. In *The 18th International Conference on Computational Linguistics*, volume 1, pages 55–61, 2000.

[6] S. Billot and B. Lang. The structure of shared forests in ambiguous parsing. In *27th Annual Meeting of the ACL*, pages 143–151, 1989.

[7] T.L. Booth and R.A. Thompson. Applying probabilistic measures to abstract languages. *IEEE Transactions on Computers*, C-22(5):442–450, May 1973.

[8] T. Brants and M. Crocker. Probabilistic parsing and psychological plausibility. In *The 18th International Conference on Computational Linguistics*, volume 1, pages 111–117, 2000.

[9] S.A. Caraballo and E. Charniak. New figures of merit for best-first probabilistic chart parsing. *Computational Linguistics*, 24(2):275–298, 1998.

[10] E. Charniak. Immediate-head parsing for language models. In *39th Annual Meeting of the ACL*, pages 116–123, 2001.

[11] C. Chelba and F. Jelinek. Exploiting syntactic structure for language modeling. In *36th Annual Meeting of the ACL*, volume 1, pages 225–231, 1998.

[12] Z. Chi. Statistical properties of probabilistic context-free grammars. *Computational Linguistics*, 25(1):131–160, 1999.

[13] M. Collins. Three generative, lexicalised models for statistical parsing. In *35th Annual Meeting of the ACL*, pages 16–23, 1997.

[14] A. Corazza. Integration of two stochastic context-free grammars. In *Proceedings of the Seventh International Conference on Spoken Language Processing (ICSLP 2002)*, pages 909–912, 2002.

[15] A. Corazza, R. De Mori, R. Gretter, and G. Satta. Computation of probabilities for an island-driven parser. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(9):936–950, 1991.

[16] J. Earley. An efficient context-free parsing algorithm. *Communications of the ACM*, 13(2):94–102, February 1970.

[17] F. Jelinek and J.D. Lafferty. Computation of the probability of initial substring generation by stochastic context-free grammars. *Computational Linguistics*, 17(3):315–323, 1991.

[18] D.E. Knuth, J.H. Morris, Jr., and V.R. Pratt. Fast pattern matching in strings. *SIAM Journal on Computing*, 6(2):323–350, 1977.

[19] B. Lang. Deterministic techniques for efficient non-deterministic parsers. In *Automata, Languages and Programming, 2nd Colloquium*, LNCS 14, pages 255–269, 1974. Springer-Verlag.

[20] I. Langkilde. Forest-based statistical sentence generation. In *1st Meeting of the North American Chapter of the ACL*, pages 170–177, 2000.

[21] D.M. Magerman and M.P. Marcus. $\mathcal{P}$earl: A probabilistic chart parser. In *Proc. of the Second International Workshop on Parsing Technologies*, pages 193–199, 1991.

[22] C.D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing.* Massachusetts Institute of Technology, 1999.

[23] K.E. Mark, M.I. Miller, and U. Grenander. Constrained stochastic language models. In S.E. Levinson and L. Shepp, editors, *Image Models (and their Speech Model Cousins)*, pages 131–140. Springer Verlag, 1996.

[24] A. Paz. *Introduction to Probabilistic Automata.* Academic Press, 1971.

[25] B. Roark. Probabilistic top-down parsing and language modeling. *Computational Linguistics*, 27(2):249–276, 2001.

[26] E.S. Santos. Probabilistic grammars and automata. *Information and Control*, 21:27–47, 1972.

[27] Y. Schabes. Stochastic lexicalized tree-adjoining grammars. In *Proc. of the fifteenth International Conference on Computational Linguistics*, volume 2, pages 426–432, 1992.

[28] K. Sima'an. Computational complexity of probabilistic disambiguation. *Grammars*, 5:125–151, 2002.

[29] P.H. Starke. *Abstract Automata.* North-Holland Publishing Company, 1972.

[30] A. Stolcke. An efficient probabilistic context-free parsing algorithm that computes prefix probabilities. *Computational Linguistics*, 21(2):167–201, 1995.