Probabilistic Parsing Strategies

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

Abstract

We present new results on the relation between context-free parsing strategies and their probabilistic counter-parts. We provide a necessary condition and a sufficient condition for the probabilistic extension of parsing strategies. These results generalize existing results in the literature that were obtained by considering parsing strategies in isolation.

1 Introduction

Context-free grammars (CFGs) are standardly used in computational linguistics as formal models of the syntax of natural language, associating sentences with all their possible derivations. Other computational models with the same generative capacity as CFGs are also adopted, as for instance push-down automata (PDAs). One of the advantages of the use of PDAs is that these devices provide an operational specification that determines which steps must be performed when parsing an input string, something that is not offered by CFGs. In other words, PDAs can be associated to **parsing strategies** for context-free languages. More precisely, parsing strategies are traditionally specified as constructions that map CFGs to language-equivalent PDAs. Popular examples of parsing strategies are the standard constructions of top-down PDAs (Harrison, 1978), left-corner PDAs (Rosenkrantz and Lewis II, 1970), shift-reduce PDAs (Aho and Ullman, 1972) and LR PDAs (Sippu and Soisalon-Soininen, 1990).

CFGs and PDAs have probabilistic counterparts, called probabilistic CFGs (PCFGs) and probabilistic PDAs (PPDAs). These models are very popular in natural language processing applications, where they are used to define a probability distribution function on the domain of all derivations for sentences in the language of interest. In PCFGs and PPDAs, probabilities are assigned to rules or transitions, respectively. However, these probabilities cannot be chosen entirely arbitrarily. For example, for a given nonterminal A in a PCFG, the sum of the probabilities of all rules rewriting A must be 1. This means that, out of a total of say m rules rewriting A, only $m - 1$ rules represent "free" parameters.

Depending on the choice of the parsing strategy, the constructed PDA may allow different probability distributions than the underlying CFG, since the set of free parameters may differ between the CFG and the PDA, both quantitatively and qualitatively. For example, (Sornlertlamvanich et al., 1999) and (Roark and Johnson, 1999) have shown that a probability distribution that can be obtained by training the probabilities of a CFG on the basis of a corpus can be less accurate than the probability distribution obtained by training the probabilities of a PDA constructed by a particular parsing strategy, on the basis of the same corpus. Also the results from (Chitrao and Grishman, 1990), (Charniak and Carroll, 1994) and (Manning and Carpenter, 2000) could be seen in this light.

The question arises of whether parsing strategies can be extended probabilistically, i.e., whether a given construction of PDAs from CFGs can be "augmented" with a function defining the probabilities for the target PDA, given the probabilities associated with the input CFG, in such a way that the obtained probabilistic distributions on the CFG derivations and the corresponding PDA computations are equivalent. Some first results on this issue have been presented by (Tendeau, 1995), who shows that the already mentioned left-corner parsing strategy can be extended probabilistically, and later by (Abney et al., 1999) who show that the pure top-down parsing strategy and a specific type of shift-reduce parsing strategy can be probabilistically extended.

One might think that any "practical" parsing strategy can be probabilistically extended, but this turns out not to be the case. We briefly discuss here a counter-example, in order to motivate the approach we have taken in this paper. Probabilistic LR parsing has been investigated in the literature (Wright and Wrigley, 1991; Briscoe and Carroll, 1993; Inui et al., 2000) under the assumption that it would allow more fine-grained probability distributions than the underlying PCFGs. However, this

is not the case in general. Consider a PCFG with rule/probability pairs:

$$\begin{array}{llll} S \to AB, & 1 & B \to bC, & \frac{2}{3} \\ A \to aC, & \frac{1}{3} & B \to bD, & \frac{1}{3} \\ A \to aD, & \frac{2}{3} & C \to xc, & 1 \\ & & D \to xd, & 1 \end{array}$$

There are two key transitions in the associated LR automaton, which represent shift actions over c and d (we denote LR states by their sets of kernel items and encode these states into stack symbols):

$$\begin{array}{ll} \tau_c: & \{C \to x \bullet c, D \to x \bullet d\} \overset{c}{\mapsto} \\ & \{C \to x \bullet c, D \to x \bullet d\}\ \{C \to xc \bullet\} \\ \tau_d: & \{C \to x \bullet c, D \to x \bullet d\} \overset{d}{\mapsto} \\ & \{C \to x \bullet c, D \to x \bullet d\}\ \{D \to xd \bullet\} \end{array}$$

Assume a proper assignment of probabilities to the transitions of the LR automaton, i.e., the sum of transition probabilities for a given LR state is 1. It can be easily seen that we must assign probability 1 to all transitions except τ_c and τ_d, since this is the only pair of distinct transitions that can be applied for one and the same top-of-stack symbol, viz. $\{C \to x \bullet c, D \to x \bullet d\}$. However, in the PCFG model we have

$$\frac{\Pr(axcbxd)}{\Pr(axdbxc)} = \frac{\Pr(A \to aC) \cdot \Pr(B \to bD)}{\Pr(A \to aD) \cdot \Pr(B \to bC)} = \frac{\frac{1}{3} \cdot \frac{1}{3}}{\frac{2}{3} \cdot \frac{2}{3}} = \frac{1}{4}$$

whereas in the LR PPDA model we have

$$\frac{\Pr(axcbxd)}{\Pr(axdbxc)} = \frac{\Pr(\tau_c) \cdot \Pr(\tau_d)}{\Pr(\tau_d) \cdot \Pr(\tau_c)} = 1 \neq \frac{1}{4}.$$

Thus we conclude that there is no proper assignment of probabilities to the transitions of the LR automaton that would result in a distribution on the generated language that is equivalent to the one induced by the source PCFG. Therefore the LR strategy does not allow probabilistic extension.

One may seemingly solve this problem by dropping the constraint of properness, letting each transition that outputs a rule have the same probability as that rule in the PCFG, and letting other transitions have probability 1. However, the properness condition for PDAs has been heavily exploited in parsing applications, in doing incremental left-to-right probability computation for beam search (Roark and Johnson, 1999; Manning and Carpenter, 2000), and more generally in integration with other linear probabilistic models. Furthermore, commonly used training algorithms for PCFGS/PPDAs always produce proper probability assignments, and many desired mathematical properties of these methods are based on such an assumption (Chi and Geman, 1998; Sánchez and Benedí, 1997). We may therefore discard non-proper probability assignments in the current study.

However, such probability assignments are outside the reach of the usual training algorithms for PDAs, which always produce proper PDAs. Therefore, we may discard such assignments in the current study, which investigates aspects of the potential of training algorithms for CFGs and PDAs.

What has been lacking in the literature is a theoretical framework to relate the parameter space of a CFG to that of a PDA constructed from the CFG by a particular parsing strategy, in terms of the set of allowable probability distributions over derivations. Note that the number of free parameters alone is not a satisfactory characterization of the parameter space. In fact, if the "nature" of the parameters is ill-chosen, then an increase in the number of parameters may lead to a deterioration of the accuracy of the model, due to sparseness of data.

In this paper we extend previous results, where only a few specific parsing strategies were considered in isolation, and provide some general characterization of parsing strategies that can be probabilistically extended. Our main contribution can be stated as follows.

- We define a theoretical framework to relate the parameter space defined by a CFG and that defined by a PDA constructed from the CFG by a particular parsing strategy.
- We provide a necessary condition and a sufficient condition for the probabilistic extension of parsing strategies.

We use the above findings to establish new results about probabilistic extensions of parsing strategies that are used in standard practice in computational linguistics, as well as to provide simpler proofs of already known results.

We introduce our framework in Section 3 and report our main results in Sections 4 and 5. We discuss applications of our results in Section 6.

2 Preliminaries

In this paper we assume some familiarity with definitions of (P)CFGs and (P)PDAs. We refer the reader to standard textbooks and publications as for instance (Harrison, 1978; Booth and Thompson, 1973; Santos, 1972).

A CFG $\mathcal{G}$ is a tuple (Σ, N, S, R), with Σ and N the sets of terminals and nonterminals, respectively, S the start symbol and R the set of rules. In this paper we only consider left-most derivations, represented as strings $d \in R^*$ and simply called deriva-

tions. For $\alpha, \beta \in (\Sigma \cup N)^*$, we write $\alpha \Rightarrow^d \beta$ with the usual meaning. If $\alpha = S$ and $\beta = w \in \Sigma^*$, we call d a **complete** derivation of w. We say a CFG is **reduced** if each rule in R occurs in some complete derivation.

A PCFG is a pair $(\mathcal{G}, p)$ consisting of a CFG $\mathcal{G}$ and a probability function p from R to real numbers in the interval $[0, 1]$. A PCFG is **proper** if $\sum_{\pi=(A\to\alpha)\in R}\ p(\pi) = 1$ for each $A \in N$. The probability of a (left-most) derivation $d = \pi_1 \cdots \pi_m$, $\pi_i \in R$ for $1 \leq i \leq m$, is $p(d) = \prod_{i=1}^{m}\ p(\pi_i)$. The probability of a string $w \in \Sigma^*$ is $p(w) = \sum_{S\Rightarrow^d w}\ p(d)$. A PCFG is **consistent** if $\Sigma_{w\in\Sigma^*}\ p(w) = 1$. A PCFG $(\mathcal{G}, p)$ is reduced if $\mathcal{G}$ is reduced.

In this paper we will mainly consider push-down transducers rather than push-down automata. Push-down transducers not only compute derivations of the grammar while processing an input string, but they also explicitly produce output strings from which these derivations can be obtained. We use transducers for two reasons. First, constraints on the output strings allow us to restrict our attention to "reasonable" parsing strategies. Those strategies that cannot be formalized within these constraints are unlikely to be of practical interest. Secondly, mappings from input strings to derivations, such as those realized by push-down transducers, turn out to be a very powerful abstraction and allow direct proofs of several general results.

Contrary to many textbooks, our push-down devices do not possess states next to stack symbols. This is without loss of generality, since states can be encoded into the stack symbols, given the types of transitions that we allow. Thus, a PDT $\mathcal{A}$ is a 6-tuple $(\Sigma_1, \Sigma_2, Q, X_{in}, X_{fin}, \Delta)$, with Σ_1 and Σ_2 the input and output alphabets, respectively, Q the set of stack symbols, including the initial and final stack symbols X_{in} and X_{fin}, respectively, and Δ the set of transitions. Each transition has one of the following three forms: $X \mapsto XY$, called a push transition, $YX \mapsto Z$, called a pop transition, or $X \overset{x,y}{\mapsto} Y$, called a swap transition; here X, Y, $Z \in Q$, $x \in \Sigma_1 \cup \{\varepsilon\}$ is the input read by the transition and $y \in \Sigma_2^*$ is the written output. Note that in our notation, stacks grow from left to right, i.e., the top-most stack symbol will be found at the right end. A **configuration** of a PDT is a triple (α, w, v), where $\alpha \in Q^*$ is a stack, $w \in \Sigma_1^*$ is the remaining input, and $v \in \Sigma_2^*$ is the output generated so far. Computations are represented as strings $c \in \Delta^*$. For configurations (α, w, v) and (β, w', v'), we write $(\alpha, w, v) \vdash^c (\beta, w', v')$ with the usual meaning, and write $(\alpha, w, v) \vdash^* (\beta, w', v')$ when c is of no importance. If $(X_{in}, w, \varepsilon) \vdash^c (X_{fin}, \varepsilon, v)$, then c is a **complete** computation of w, and the output string v is denoted $out(c)$. A PDT is **reduced** if each transition in Δ occurs in some complete computation.

Without loss of generality, we assume that combinations of different types of transitions are not allowed for a given stack symbol. More precisely, for each stack symbol $X \neq X_{fin}$, the PDA can only take transitions of a single type (push, pop or swap). A PDT can easily be brought in this form by introducing for each X three new stack symbols X_{push}, X_{pop} and X_{swap} and new swap transitions $X \overset{\varepsilon,\varepsilon}{\mapsto} X_{push}$, $X \overset{\varepsilon,\varepsilon}{\mapsto} X_{pop}$ and $X \overset{\varepsilon,\varepsilon}{\mapsto} X_{swap}$. In each existing transition that operates on top-of-stack X, we then replace X by one from X_{push}, X_{pop} or X_{swap}, depending on the type of that transition. We also assume that X_{fin} does not occur in the left-hand side of a transition, again without loss of generality.

A PPDT is a pair $(\mathcal{A}, p)$ consisting of a PDT $\mathcal{A}$ and a probability function p from Δ to real numbers in the interval $[0, 1]$. A PPDT is **proper** if

- $\Sigma_{\tau=(X\mapsto XY)\in\Delta}\ p(\tau) = 1$ for each $X \in Q$ such that there is at least one transition $X \mapsto XY$, $Y \in Q$;
- $\Sigma_{\tau=(X\overset{x,y}{\mapsto}Y)\in\Delta}\ p(\tau) = 1$ for each $X \in Q$ such that there is at least one transition $X \overset{x,y}{\mapsto} Y$, $x \in \Sigma_1 \cup \{\varepsilon\}, y \in \Sigma_2^*, Y \in Q$; and
- $\Sigma_{\tau=(YX\mapsto Z)\in\Delta}\ p(\tau) = 1$, for each $X, Y \in Q$ such that there is at least one transition $YX \mapsto Z$, $Z \in Q$.

The probability of a computation $c = \tau_1 \cdots \tau_m$, $\tau_i \in \Delta$ for $1 \leq i \leq m$, is $p(c) = \prod_{i=1}^{m}\ p(\tau_i)$. The probability of a string w is $p(w) = \sum_{(X_{in},w,\varepsilon)\vdash^c(X_{fin},\varepsilon,v)}\ p(c)$. A PPDT is **consistent** if $\Sigma_{w\in\Sigma^*}\ p(w) = 1$. A PPDT $(\mathcal{A}, p)$ is reduced if $\mathcal{A}$ is reduced.

3 Parsing Strategies

The term "parsing strategy" is often used informally to refer to a class of parsing algorithms that behave similarly in some way. In this paper, we assign a formal meaning to this term, relying on the observation by (Lang, 1974) and (Billot and Lang, 1989) that many parsing algorithms for CFGs can be described in two steps. The first is a construction of push-down devices from CFGs, and the second is a method for handling nondeterminism (e.g. backtracking or dynamic programming). Parsing algorithms that handle nondeterminism in different ways

but apply the same construction of push-down devices from CFGs are seen as realizations of the same parsing strategy.

Thus, we define a **parsing strategy** to be a function $\mathcal{S}$ that maps a reduced CFG $\mathcal{G} = (\Sigma_1, N, S, R)$ to a pair $\mathcal{S}(\mathcal{G}) = (\mathcal{A}, f)$ consisting of a reduced PDT $\mathcal{A} = (\Sigma_1, \Sigma_2, Q, X_{in}, X_{fin}, \Delta)$, and a function f that maps a subset of Σ_2^* to a subset of R^*, with the following properties:

- $R \subseteq \Sigma_2$.
- For each string $w \in \Sigma_1^*$ and each complete computation c on w, $f(out(c)) = d$ is a (leftmost) derivation of w. Furthermore, each symbol from R occurs as often in $out(c)$ as it occurs in d.
- Conversely, for each string $w \in \Sigma_1^*$ and each derivation d of w, there is precisely one complete computation c on w such that $f(out(c)) = d$.

If c is a complete computation, we will write $f(c)$ to denote $f(out(c))$. The conditions above then imply that f is a bijection from complete computations to complete derivations. Note that output strings of (complete) computations may contain symbols that are not in R, and the symbols that are in R may occur in a different order in v than in $f(v) = d$. The purpose of the symbols in $\Sigma_2 - R$ is to help this process of reordering of symbols from R in v, as needed for instance in the case of the left-corner parsing strategy (see (Nijholt, 1980, pp. 22–23) for discussion).

A **probabilistic parsing strategy** is defined to be a function $\mathcal{S}$ that maps a reduced, proper and consistent PCFG $(\mathcal{G}, p_\mathcal{G})$ to a triple $\mathcal{S}(\mathcal{G}, p_\mathcal{G}) = (\mathcal{A}, p_\mathcal{A}, f)$, where $(\mathcal{A}, p_\mathcal{A})$ is a reduced, proper and consistent PPDT, with the same properties as a (non-probabilistic) parsing strategy, and in addition:

- For each complete derivation d and each complete computation c such that $f(c) = d$, $p_\mathcal{G}(d)$ equals $p_\mathcal{A}(c)$.

In other words, a complete computation has the same probability as the complete derivation that it is mapped to by function f. An implication of this property is that for each string $w \in \Sigma_1^*$, the probabilities assigned to that string by $(\mathcal{G}, p_\mathcal{G})$ and $(\mathcal{A}, p_\mathcal{A})$ are equal.

We say that probabilistic parsing strategy $\mathcal{S}'$ is an **extension** of parsing strategy $\mathcal{S}$ if for each reduced CFG $\mathcal{G}$ and probability function $p_\mathcal{G}$ we have $\mathcal{S}(\mathcal{G}) = (\mathcal{A}, f)$ if and only if $\mathcal{S}'(\mathcal{G}, p_\mathcal{G}) = (\mathcal{A}, p_\mathcal{A}, f)$ for some $p_\mathcal{A}$.

4 Correct-Prefix Property

In this section we present a necessary condition for the probabilistic extension of a parsing strategy. For a given PDT, we say a computation c is **dead** if $(X_{in}, w_1, \varepsilon) \vdash^c (\alpha, \varepsilon, v_1)$, for some $\alpha \in Q^*$, $w_1 \in \Sigma_1^*$ and $v_1 \in \Sigma_2^*$, and there are no $w_2 \in \Sigma_1^*$ and $v_2 \in \Sigma_2^*$ such that $(\alpha, w_2, \varepsilon) \vdash^* (X_{fin}, \varepsilon, v_2)$. Informally, a dead computation is a computation that cannot be continued to become a complete computation. We say that a PDT has the **correct-prefix property** (CPP) if it does not allow any dead computations. We also say that a parsing strategy has the CPP if it maps each reduced CFG to a PDT that has the CPP.

Lemma 1 *For each reduced CFG $\mathcal{G}$, there is a probability function $p_\mathcal{G}$ such that PCFG $(\mathcal{G}, p_\mathcal{G})$ is proper and consistent, and $p_\mathcal{G}(d) > 0$ for all complete derivations d.*

Proof. Since $\mathcal{G}$ is reduced, there is a finite set D consisting of complete derivations d, such that for each rule π in $\mathcal{G}$ there is at least one $d \in D$ in which π occurs. Let $n_{\pi,d}$ be the number of occurrences of rule π in derivation $d \in D$, and let n_π be $\Sigma_{d \in D}\, n_{\pi,d}$, the total number of occurrences of π in D. Let n_A be the sum of n_π for all rules π with A in the left-hand side. A probability function $p_\mathcal{G}$ can be defined through "maximum-likelihood estimation" such that $p_\mathcal{G}(\pi) = \frac{n_\pi}{n_A}$ for each rule $\pi = A \to \alpha$.

For all nonterminals A, $\Sigma_{\pi=A\to\alpha}\; p_\mathcal{G}(\pi) = \Sigma_{\pi=A\to\alpha}\; \frac{n_\pi}{n_A} = \frac{n_A}{n_A} = 1$, which means that the PCFG $(\mathcal{G}, p_\mathcal{G})$ is proper. Furthermore, it has been shown in (Chi and Geman, 1998; Sánchez and Benedí, 1997) that a PCFG $(\mathcal{G}, p_\mathcal{G})$ is consistent if $p_\mathcal{G}$ was obtained by maximum-likelihood estimation using a set of derivations. Finally, since $n_\pi > 0$ for each π, also $p_\mathcal{G}(\pi) > 0$ for each π, and $p_\mathcal{G}(d) > 0$ for all complete derivations d. ∎

We say a computation is a shortest dead computation if it is dead and none of its proper prefixes is dead. Note that each dead computation has a unique prefix that is a shortest dead computation. For a PDT $\mathcal{A}$, let $\mathcal{T}_\mathcal{A}$ be the union of the set of all complete computations and the set of all shortest dead computations.

Lemma 2 *For each proper PPDT $(\mathcal{A}, p_\mathcal{A})$, $\Sigma_{c \in \mathcal{T}_\mathcal{A}}\; p_\mathcal{A}(c) \leq 1$.*

Proof. The proof is a trivial variant of the proof that for a proper PCFG $(\mathcal{G}, p_\mathcal{G})$, the sum of $p_\mathcal{G}(d)$ for all derivations d cannot exceed 1, which is shown by (Booth and Thompson, 1973). ∎

From this, the main result of this section follows.

Theorem 3 *A parsing strategy that lacks the CPP cannot be extended to become a probabilistic parsing strategy.*

Proof. Take a parsing strategy $\mathcal{S}$ that does not have the CPP. Then there is a reduced CFG $\mathcal{G} = (\Sigma_1, N, S, R)$, with $\mathcal{S}(\mathcal{G}) = (\mathcal{A}, f)$ for some $\mathcal{A}$ and f, and a shortest dead computation c allowed by $\mathcal{A}$.

It follows from Lemma 1 that there is a probability function $p_\mathcal{G}$ such that $(\mathcal{G}, p_\mathcal{G})$ is a proper and consistent PCFG and $p_\mathcal{G}(d) > 0$ for all complete derivations d. Assume we also have a probability function $p_\mathcal{A}$ such that $(\mathcal{A}, p_\mathcal{A})$ is a proper and consistent PPDT and $p_\mathcal{A}(c') = p_\mathcal{G}(f(c'))$ for each complete computation c'. Since $\mathcal{A}$ is reduced, each transition τ must occur in some complete computation c'. Furthermore, for each complete computation c' there is a complete derivation d such that $f(c') = d$, and $p_\mathcal{A}(c') = p_\mathcal{G}(d) > 0$. Therefore, $p_\mathcal{A}(\tau) > 0$ for each transition τ, and $p_\mathcal{A}(c) > 0$, where c is the above-mentioned dead computation.

Due to Lemma 2, $1 \geq \Sigma_{c' \in \mathcal{T}_\mathcal{A}}\ p_\mathcal{A}(c') \geq \Sigma_{w \in \Sigma_1^*}\ p_\mathcal{A}(w) + p_\mathcal{A}(c) > \Sigma_{w \in \Sigma_1^*}\ p_\mathcal{A}(w) = \Sigma_{w \in \Sigma_1^*}\ p_\mathcal{G}(w)$. This is in contradiction with the consistency of $(\mathcal{G}, p_\mathcal{G})$. Hence, a probability function $p_\mathcal{A}$ with the properties we required above cannot exist, and therefore $\mathcal{S}$ cannot be extended to become a probabilistic parsing strategy. ■

5 Strong Predictiveness

In this section we present our main result, which is a sufficient condition allowing the probabilistic extension of a parsing strategy. We start with a technical result that was proven in (Abney et al., 1999; Chi, 1999; Nederhof and Satta, 2003).

Lemma 4 *Given a non-proper PCFG $(\mathcal{G}, p_\mathcal{G})$, $\mathcal{G} = (\Sigma, N, S, R)$, there is a probability function $p'_\mathcal{G}$ such that PCFG $(\mathcal{G}, p'_\mathcal{G})$ is proper and, for every complete derivation d, $p'_\mathcal{G}(d) = \frac{1}{C} \cdot p_\mathcal{G}(d)$, where $C = \sum_{S \Rightarrow^{d'} w, w \in \Sigma^*} p_\mathcal{G}(d')$.*

Note that if PCFG $(\mathcal{G}, p_\mathcal{G})$ in the above lemma is consistent, then $C = 1$ and $(\mathcal{G}, p'_\mathcal{G})$ and $(\mathcal{G}, p_\mathcal{G})$ define the same distribution on derivations. The normalization procedure underlying Lemma 4 makes use of quantities $\sum_{A \Rightarrow^d w, w \in \Sigma^*}\ p_\mathcal{G}(d)$ for each $A \in N$. These quantities can be computed to any degree of precision, as discussed for instance in (Booth and Thompson, 1973) and (Stolcke, 1995). Thus normalization of a PCFG can be effectively computed.

For a fixed PDT, we define the binary relation $\rightsquigarrow$ on stack symbols by: $Y \rightsquigarrow Y'$ if and only if $(Y, w, \varepsilon) \vdash^* (Y', \varepsilon, v)$ for some $w \in \Sigma_1^*$ and $v \in \Sigma_2^*$. In words, some subcomputation of the PDT may start with stack Y and end with stack Y'. Note that all stacks that occur in such a subcomputation must have height of 1 or more. We say that a (P)PDA or a (P)PDT has the **strong predictiveness property** (SPP) if the existence of three transitions $X \mapsto XY$, $XY_1 \mapsto Z_1$ and $XY_2 \mapsto Z_2$ such that $Y \rightsquigarrow Y_1$ and $Y \rightsquigarrow Y_2$ implies $Z_1 = Z_2$. Informally, this means that when a subcomputation starts with some stack α and some push transition τ, then solely on the basis of τ we can uniquely determine what stack symbol $Z_1 = Z_2$ will be on top of the stack in the firstly reached configuration with stack height equal to $|\alpha|$. Another way of looking at it is that no information may flow from higher stack elements to lower stack elements that was not already predicted before these higher stack elements came into being, hence the term "strong predictiveness". We say that a parsing strategy has the SPP if it maps each reduced CFG to a PDT with the SPP.

Theorem 5 *Any parsing strategy that has the CPP and the SPP can be extended to become a probabilistic parsing strategy.*

Proof. Consider a parsing strategy $\mathcal{S}$ that has the CPP and the SPP, and a proper, consistent and reduced PCFG $(\mathcal{G}, p_\mathcal{G})$, $\mathcal{G} = (\Sigma_1, N, S, R)$. Let $\mathcal{S}(\mathcal{G}) = (\mathcal{A}, f)$, $\mathcal{A} = (\Sigma_1, \Sigma_2, Q, X_{in}, X_{fin}, \Delta)$. We will show that there is a probability function $p_\mathcal{A}$ such that $(\mathcal{A}, p_\mathcal{A})$ is a proper and consistent PPDT, and $p_\mathcal{A}(c) = p_\mathcal{G}(f(c))$ for all complete computations c.

We first construct a PPDT $(\mathcal{A}, p'_\mathcal{A})$ as follows. For each scan transition $\tau = X \overset{x,y}{\mapsto} Y$ in Δ, let $p'_\mathcal{A}(\tau) = p_\mathcal{G}(y)$ in case $y \in R$, and $p'_\mathcal{A}(\tau) = 1$ otherwise. For all remaining transitions $\tau \in \Delta$, let $p'_\mathcal{A}(\tau) = 1$. Note that $(\mathcal{A}, p'_\mathcal{A})$ may be non-proper. Still, from the definition of f it follows that, for each complete computation c, we have

$$p'_\mathcal{A}(c) \quad = \quad p_\mathcal{G}(f(c)), \tag{1}$$

and so our PPDT is consistent.

We now map $(\mathcal{A}, p'_\mathcal{A})$ to a language-equivalent PCFG $(\mathcal{G}', p_{\mathcal{G}'})$, $\mathcal{G}' = (\Sigma_1, Q, X_{in}, R')$, where R' contains the following rules with the specified associated probabilities:

- $X \to YZ$ with $p_{\mathcal{G}'}(X \to YZ) = p'_\mathcal{A}(X \mapsto XY)$, for each $X \mapsto XY \in \Delta$ with Z the unique stack symbol such that there is at least one transition $XY' \mapsto Z$ with $Y \rightsquigarrow Y'$;
- $X \to xY$ with $p_{\mathcal{G}'}(X \to xY) = p'_\mathcal{A}(X \overset{x}{\mapsto} Y)$, for each transition $X \overset{x}{\mapsto} Y \in \Delta$;

- $Y \to \varepsilon$ with $p_{\mathcal{G}'}(X \to \varepsilon) = 1$, for each stack symbol Y such that there is at least one transition $XY \mapsto Z \in \Delta$ or such that $Y = X_{fin}$.

It is not difficult to see that there exists a bijection f' from complete computations of $\mathcal{A}$ to complete derivations of $\mathcal{G}'$, and that we have

$$p_{\mathcal{G}'}(f'(c)) \quad = \quad p'_{\mathcal{A}}(c), \tag{2}$$

for each complete computation c. Thus $(\mathcal{G}', p_{\mathcal{G}'})$ is consistent. However, note that $(\mathcal{G}', p_{\mathcal{G}'})$ is not proper.

By Lemma 4, we can construct a new PCFG $(\mathcal{G}', p'_{\mathcal{G}'})$ that is proper and consistent, and such that $p_{\mathcal{G}'}(d) = p'_{\mathcal{G}'}(d)$, for each complete derivation d of $\mathcal{G}'$. Thus, for each complete computation c of $\mathcal{A}$, we have

$$p'_{\mathcal{G}'}(f'(c)) \quad = \quad p_{\mathcal{G}'}(f'(c)). \tag{3}$$

We now transfer back the probabilities of rules of $(\mathcal{G}', p'_{\mathcal{G}'})$ to the transitions of $\mathcal{A}$. Formally, we define a new probability function $p_{\mathcal{A}}$ such that, for each $\tau \in \Delta$, $p_{\mathcal{A}}(\tau) = p'_{\mathcal{G}'}(\pi)$, where π is the rule in R' that has been constructed from τ as specified above. It is easy to see that PPDT $(\mathcal{A}, p_{\mathcal{A}})$ is now proper. Furthermore, for each complete computation c of $\mathcal{A}$ we have

$$p_{\mathcal{A}}(c) \quad = \quad p'_{\mathcal{G}'}(f'(c)), \tag{4}$$

and so $(\mathcal{A}, p_{\mathcal{A}})$ is also consistent. By combining equations (1) to (4) we conclude that, for each complete computation c of $\mathcal{A}$, $p_{\mathcal{A}}(c) = p'_{\mathcal{G}'}(f'(c)) = p_{\mathcal{G}'}(f'(c)) = p'_{\mathcal{A}}(c) = p_{\mathcal{G}}(f(c))$. Thus our parsing strategy $\mathcal{S}$ can be probabilistically extended. ■

Note that the construction in the proof above can be effectively computed (see discussion in Section 4 for effective computation of normalized PCFGs).

The definition of $p'_{\mathcal{A}}$ in the proof of Theorem 5 relies on the strings output by $\mathcal{A}$. This is the main reason why we needed to consider PDTs rather than PDAs. Now assume an appropriate probability function $p_{\mathcal{A}}$ has been computed, such that the source PCFG and $(\mathcal{A}, p_{\mathcal{A}})$ define equivalent distributions on derivations/computations. Then the probabilities assigned to strings over the input alphabet are also equal. We may subsequently ignore the output strings if the application at hand merely requires probabilistic recognition rather than probabilistic transduction, or in other words, we may simplify PDTs to PDAs.

The proof of Theorem 5 also leads to the observation that parsing strategies with the CPP and the SPP as well as their probabilistic extensions can be described as grammar transformations, as follows. A given (P)CFG is mapped to an equivalent (P)PDT by a (probabilistic) parsing strategy. By ignoring the output components of swap transitions we obtain a (P)PDA, which can be mapped to an equivalent (P)CFG as shown above. This observation gives rise to an extension with probabilities of the work on covers by (Nijholt, 1980; Leermakers, 1989).

6 Applications

Many well-known parsing strategies with the CPP also have the SPP. This is for instance the case for top-down parsing and left-corner parsing. As discussed in the introduction, it has already been shown that for any PCFG $\mathcal{G}$, there are equivalent PPDTs implementing these strategies, as reported in (Abney et al., 1999) and (Tendeau, 1995), respectively. Those results more simply follow now from our general characterization. Furthermore, PLR parsing (Soisalon-Soininen and Ukkonen, 1979; Nederhof, 1994) can be expressed in our framework as a parsing strategy with the CPP and the SPP, and thus we obtain as a new result that this strategy allows probabilistic extension.

The above strategies are in contrast to the LR parsing strategy, which has the CPP but lacks the SPP, and therefore falls outside our sufficient condition. As we have already seen in the introduction, it turns out that LR parsing cannot be extended to become a probabilistic parsing strategy. Related to LR parsing is ELR parsing (Purdom and Brown, 1981; Nederhof, 1994), which also lacks the SPP. By an argument similar to the one provided for LR, we can show that also ELR parsing cannot be extended to become a probabilistic parsing strategy. (See (Tendeau, 1997) for earlier observations related to this.) These two cases might suggest that the sufficient condition in Theorem 5 is tight in practice.

Decidability of the CPP and the SPP obviously depends on how a parsing strategy is specified. As far as we know, in all practical cases of parsing strategies these properties can be easily decided. Also, observe that our results do not depend on the general behaviour of a parsing strategy $\mathcal{S}$, but just on its "point-wise" behaviour on each input CFG. Specifically, if $\mathcal{S}$ does not have the CPP and the SPP, but for some fixed CFG $\mathcal{G}$ of interest we obtain a PDT $\mathcal{A}$ that has the CPP and the SPP, then we can still apply the construction in Theorem 5. In this way, any probability function $p_{\mathcal{G}}$ associated with $\mathcal{G}$ can be converted into a probability function $p_{\mathcal{A}}$, such that the resulting PCFG and PPDT induce equivalent distributions. We point out that decidability of the CPP and the SPP for a fixed PDT can

be efficiently decided using dynamic programming.

One more consequence of our results is this. As discussed in the introduction, the properness condition reduces the number of parameters of a PPDT. However, our results show that if the PPDT has the CPP and the SPP then the properness assumption is not restrictive, i.e., by lifting properness we do not gain new distributions with respect to those induced by the underlying PCFG.

7 Conclusions

We have formalized the notion of CFG parsing strategy as a mapping from CFGs to PDTs, and have investigated the extension to probabilities. We have shown that the question of which parsing strategies can be extended to become probabilistic heavily relies on two properties, the correct-prefix property and the strong predictiveness property. As far as we know, this is the first general characterization that has been provided in the literature for probabilistic extension of CFG parsing strategies. We have also shown that there is at least one strategy of practical interest with the CPP but without the SPP, namely LR parsing, that cannot be extended to become a probabilistic parsing strategy.

Acknowledgements

The first author is supported by the PIONIER Project *Algorithms for Linguistic Processing*, funded by NWO (Dutch Organization for Scientific Research). The second author is partially supported by MIUR under project PRIN No. 2003091149_005.

References

S. Abney, D. McAllester, and F. Pereira. 1999. Relating probabilistic grammars and automata. In *37th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 542–549, Maryland, USA, June.

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

S. Billot and B. Lang. 1989. The structure of shared forests in ambiguous parsing. In *27th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 143–151, Vancouver, British Columbia, Canada, June.

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

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

E. Charniak and G. Carroll. 1994. Context-sensitive statistics for improved grammatical language models. In *Proceedings Twelfth National Conference on Artificial Intelligence*, volume 1, pages 728–733, Seattle, Washington.

Z. Chi and S. Geman. 1998. Estimation of probabilistic context-free grammars. *Computational Linguistics*, 24(2):299–305.

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

M.V. Chitrao and R. Grishman. 1990. Statistical parsing of messages. In *Speech and Natural Language, Proceedings*, pages 263–266, Hidden Valley, Pennsylvania, June.

M.A. Harrison. 1978. *Introduction to Formal Language Theory*. Addison-Wesley.

K. Inui, V. Sornlertlamvanich, H. Tanaka, and T. Tokunaga. 2000. Probabilistic GLR parsing. In H. Bunt and A. Nijholt, editors, *Advances in Probabilistic and other Parsing Technologies*, chapter 5, pages 85–104. Kluwer Academic Publishers.

B. Lang. 1974. Deterministic techniques for efficient non-deterministic parsers. In *Automata, Languages and Programming, 2nd Colloquium*, volume 14 of *Lecture Notes in Computer Science*, pages 255–269, Saarbrücken. Springer-Verlag.

R. Leermakers. 1989. How to cover a grammar. In *27th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 135–142, Vancouver, British Columbia, Canada, June.

C.D. Manning and B. Carpenter. 2000. Probabilistic parsing using left corner language models. In H. Bunt and A. Nijholt, editors, *Advances in Probabilistic and other Parsing Technologies*, chapter 6, pages 105–124. Kluwer Academic Publishers.

M.-J. Nederhof and G. Satta. 2003. Probabilistic parsing as intersection. In *8th International Workshop on Parsing Technologies*, pages 137–148, LORIA, Nancy, France, April.

M.-J. Nederhof. 1994. An optimal tabular parsing algorithm. In *32nd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 117–124, Las Cruces, New Mexico, USA, June.

A. Nijholt. 1980. *Context-Free Grammars: Covers, Normal Forms, and Parsing*, volume 93 of

Lecture Notes in Computer Science. Springer-Verlag.

P.W. Purdom, Jr. and C.A. Brown. 1981. Parsing extended LR(k) grammars. *Acta Informatica*, 15:115–127.

B. Roark and M. Johnson. 1999. Efficient probabilistic top-down and left-corner parsing. In *37th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 421–428, Maryland, USA, June.

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

J.-A. Sánchez and J.-M. Benedí. 1997. Consistency of stochastic context-free grammars from probabilistic estimation based on growth transformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(9):1052–1055, September.

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

S. Sippu and E. Soisalon-Soininen. 1990. *Parsing Theory, Vol. II: LR(k) and LL(k) Parsing*, volume 20 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag.

E. Soisalon-Soininen and E. Ukkonen. 1979. A method for transforming grammars into LL(k) form. *Acta Informatica*, 12:339–369.

V. Sornlertlamvanich, K. Inui, H. Tanaka, T. Tokunaga, and T. Takezawa. 1999. Empirical support for new probabilistic generalized LR parsing. *Journal of Natural Language Processing*, 6(3):3–22.

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

F. Tendeau. 1995. Stochastic parse-tree recognition by a pushdown automaton. In *Fourth International Workshop on Parsing Technologies*, pages 234–249, Prague and Karlovy Vary, Czech Republic, September.

F. Tendeau. 1997. *Analyse syntaxique et sémantique avec évaluation d'attributs dans un demi-anneau*. Ph.D. thesis, University of Orléans.

J.H. Wright and E.N. Wrigley. 1991. GLR parsing with probability. In M. Tomita, editor, *Generalized LR Parsing*, chapter 8, pages 113–128. Kluwer Academic Publishers.