Cross-moments computation for stochastic context-free grammars

Velimir M. Ilić\textsuperscript{a,b,*}, Miroslav D. Ćirić\textsuperscript{b}, Miomir S. Stanković\textsuperscript{c}

\textsuperscript{a}Mathematical Institute of the Serbian Academy of Sciences and Arts, Kneza Mihaila 36, 11000 Beograd, Serbia
\textsuperscript{b}University of Niš, Faculty of Sciences and Mathematics, Višegradska 33, 18000 Niš, Serbia
\textsuperscript{c}University of Niš, Faculty of Occupational Safety, Čarnojevića 10a, 18000 Niš, Serbia

Abstract

In this paper we consider the problem of efficient computation of cross-moments of a vector random variable represented by a stochastic context-free grammar. Two types of cross-moments are discussed. The sample space of the first one is the set of all derivations of the context-free grammar, and the sample space of the second one is the set of all derivations which generate a string belonging to the language of the grammar. In the past, this problem has been widely studied, but mainly for cross-moments of scalar variables and up to the second order. This paper presents new algorithms for computing the cross-moments of an arbitrary order, and the previously developed ones are derived as special cases.

Keywords: stochastic context-free grammar, cross-moments, semiring, moment generating function, partition function, inside-outside algorithm

1. Introduction

The cross-moments of random variables modeled with stochastic context-free grammars (SCFG) are important quantities for the SCFG parameter estimation \cite{8}. They are defined as expected value of the product of integer powers of the entries of random vector variable, which can represent string or derivation length, the number of rule occurrences in derivation or uncertainty associated with the occurring rule. The expectation can be taken either with respect to the sample space of all SCFG derivations, or with respect to the sample space of all derivations which generate a string belonging to the language of the grammar. Throughout this paper, the name cross-moments is usually used in the former case, while in the latter case we talk about conditional cross-moments.

The computation of cross-moments may become demanding if the sample space is large. In the past, this problem has been widely studied, but mainly for the cross-moments of scalar variables (called simply moments) and up to the second order. The first order moments computation, such as expected length of derivations and expected string length, are given in \cite{19}. The computation of SCFG entropy is considered in \cite{12}. The procedure for computing the moments of string and derivation length is given in \cite{8}, where the explicit formulas for the moments up to the second order are derived. First order conditional cross-moments are considered in \cite{9}, where the algorithm for conditional SCFG entropy is derived. A more general algorithm for computing the conditional cross-moments of a vector variable of the second order is derived in \cite{11}.

In this paper we give the recursive formulas for computing the cross-moments and the conditional cross-moments of a vector variable of an arbitrary order. The formulas are derived by differentiating the recursive equations for the moment generating function \cite{16}, which are obtained from the algorithms for computing
the partition function of a SCFG \cite{15} for the cross-moments and with the inside algorithm \cite{10, 5} for the conditional cross-moments.

The paper is organized as follows. Section 2 introduces multi-index notation, which is used throughout the paper, and reviews some preliminary notions about generalized Leibniz's formula, basic algebraic structures, and context free grammars. In Section 3 we give the formal definition of SCFG cross-moments and moment generating function. The recursive equations for cross-moments are given in Section 4 for the case when the sample space is the set of all derivations, while in Section 5 we consider the set of all derivations which generate a string belonging to the language of the grammar as the sample space.

2. Preliminaries

This section gives some basic definitions and theorems which are used in the paper. We review multi-index notation, which is used throughout the paper, and review some preliminary notions about generalized Leibniz’s formula, basic algebraic structures, and context free grammars, according to \cite{13} and \cite{14}.

2.1. Generalized Leibniz’s formula

For any multi-index $\alpha = (\alpha_1, \ldots, \alpha_d)$ $\in \mathbb{N}_0^d$, we define its length as the sum $|\alpha| = \alpha_1 + \alpha_2 + \cdots + \alpha_d$. The multi-index factorial is $\alpha! = \alpha_1! \cdot \alpha_2! \cdots \alpha_d!$. The zero multi-index is $0 = (0, \ldots, 0)$.

If $\beta = (\beta_1, \ldots, \beta_d) \in \mathbb{N}_0^d$, we write $\beta < \alpha$ if $\beta_i < \alpha_i$ for all $i = 1, \ldots, d$. We write $\beta \leq \alpha$ provided $\beta_i \leq \alpha_i$ for $i = 1, \ldots, d$. In such a case, we set $\alpha \pm \beta = (\alpha_1 \pm \beta_1, \ldots, \alpha_d \pm \beta_d)$, and

$$
\binom{\alpha}{\beta} = \binom{\alpha_1}{\beta_1} \cdots \binom{\alpha_d}{\beta_d}.
$$

(1)

If $\beta_1 + \cdots + \beta_N = \alpha$, we define the multinomial coefficients by

$$
\binom{\alpha}{\beta_1, \ldots, \beta_N} = \frac{\alpha!}{\prod \beta_i}.
$$

(2)

For $z = (z_1, \ldots, z_d) \in \mathbb{R}^d$ and $y = (y_1, \ldots, y_d) \in \mathbb{N}_0^d$ the multi-index power is defined by:

$$
z^y = z_1^{y_1} \cdots z_d^{y_d}.
$$

(3)

With these settings, the multinomial theorem \cite{4} can be expressed as

$$
\left( \sum_{i=1}^{N} z_i \right)^{\alpha} = \sum_{\beta_i + \cdots + \beta_N = \alpha} \binom{\alpha}{\beta_1, \ldots, \beta_N} z^\beta,
$$

(4)

for $z = (z_1, \ldots, z_d) \in \mathbb{R}^d$ and $y = (y_1, \ldots, y_d) \in \mathbb{N}_0^d$.

Let $\nu = (\nu_1, \ldots, \nu_d)$ and let $C_\nu$ denotes the set of functions $u : \mathbb{R}^d \to \mathbb{R}$ which have $\nu$-th partial derivative at zero. For every $u \in C_\alpha$ we define the mapping $D_{\nu} : C_\alpha \to \mathbb{R}$ with

$$
D_{\nu}[u] = \left. \frac{\partial^{\nu} u(t)}{\partial t_1^{\nu_1} \cdots \partial t_d^{\nu_d}} \right|_{t=0}.
$$

(5)

Note that $D_0[u(t)] = u(0)$. According to the generalized Leibniz’s formula \cite{15}, the following equality holds

$$
D_\alpha[F \cdot G] = \sum_{0 \leq \beta \leq \alpha} \binom{\alpha}{\beta} D_\beta[F] \cdot D_{\alpha-\beta}[G],
$$

(6)

for all $F, G \in C_\alpha$. The derivative of the product of more than two functions can be found according to \cite{18}

$$
D_\alpha \left( \prod_{i=1}^{m} F_i \right) = \sum_{\beta_1 + \cdots + \beta_m = \nu} \binom{\alpha}{\beta_1, \ldots, \beta_m} \prod_{i=1}^{m} D_{\beta_i}[F_i(x)],
$$

(7)

for all $F_i \in C_{\nu_i}$, $i = 1, \ldots, m$. 

2
2.2. Semirings

A **monoid** is a triple \((K, \oplus, 0)\) where \(\oplus\) is an associative binary operation on the set \(K\), and 0 is the identity element for \(\oplus\), i.e., \(a \oplus 0 = 0 \oplus a = a\), for all \(a \in K\). A monoid is commutative if the operation \(\oplus\) is commutative.

**Example 2.1.** Let \(G\) be a weighted context-free grammar whose rules are the set \(R\). A nonterminal \(A\) accessible from \(B\) is accessible from \(A\), otherwise, it is useless.

A **semiring** is a tuple \((K, \oplus, \otimes, 0, 1)\) such that

1. \((K, \oplus, 0)\) is a commutative monoid with 0 as the identity element for \(\oplus\),
2. \((K, \otimes, 1)\) is a monoid with 1 as the identity element for \(\otimes\),
3. \(\otimes\) distributes over \(\oplus\), i.e., \((a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)\) and \(c \otimes (a \oplus b) = (c \otimes a) \oplus (c \otimes b)\), for all \(a, b, c \in K\),
4. 0 is an annihilator for \(\oplus\), i.e., \(a \otimes 0 = 0 \otimes a = 0\), for every \(a \in K\).

A semiring is commutative if the operation \(\oplus\) is commutative. The operations \(\oplus\) and \(\otimes\) are called the addition and the multiplication in \(K\). For a topology \(\tau\) we define the topological semiring as a pair \((K, \tau)\).

**Example 2.2.** The semiring of \(\nu\)-continuous functions at zero is the semiring \((C_\nu, +, \cdot, 0, 1)\) with the standard uniform norm topology, where the sum and the product of functions \(F, G \in C_\nu\), are defined through the sum and the product of real numbers, \((F + G)(t) = F(t) + G(t), (F \cdot G)(t) = F(t) \cdot G(t)\), and the identities \(0, 1 \in C_\nu\) are the zero and identity functions defined by \(0(t) = 0\) and \(1(t) = 1\), for all \(t \in \mathbb{R}\).

2.3. Weighted and stochastic context-free grammars

By a weighted context-free grammar (WCFG) over a commutative semiring \((K, +, \cdot, 0, 1)\) we mean a tuple \(G = (\Sigma, N, S, R, \mu)\), where

- \(\Sigma = \{w_1, \ldots, w_\Sigma\}\) is a finite set of **terminals**, 
- \(N = \{A_1, \ldots, A_N\}\) is a finite set of **nonterminals** disjoint with \(\Sigma\),
- \(S \in N\) is called the **start symbol** (throughout the paper it is usually assumed that \(S = A_1\)),
- \(R \subseteq N \times (\Sigma \cup N)^*\) is a finite set of rules. A rule \((A, \alpha) \in R\) is commonly written as \(A \rightarrow \alpha\), where the nonterminal \(A\) is called the **premise**. The set of all rules \(A_i \rightarrow B_{i,j}, B_{i,j} \in (N \cup \Sigma)^*\) will be denoted by \(R_i\).
- \(\mu : R \rightarrow K\) is the function called **weight**.

The **left-most rewriting relation** \(\Rightarrow\) associated with \(G\) is defined as the set of triples \((\alpha, \pi, \beta) \in (\Sigma \cup N)^* \times R \times (\Sigma \cup N)^*\) for which there are a terminal string \(u \in \Sigma^*\), a nonterminal string \(\delta \in (\Sigma \cup N)^*\), along with a nonterminal \(A \in N\) and a string \(\gamma \in (\Sigma \cup N)^*\) such that \(\alpha = uA\delta, \beta = u\gamma\delta\) and \(\pi = A \rightarrow \gamma\) is a rule from \(R\). The left-most relation triple \((\alpha, \pi, \beta)\) will be denoted by \(\alpha \overset{\pi}{\Rightarrow} \beta\). The left-most derivation (in the further text the **derivation**) in this grammar is a string \(\pi_1, \ldots, \pi_n \in R^*\) for which there are grammar symbols \(\alpha, \beta \in \Sigma \cup N\) such that we can derive \(\beta\) from \(\alpha\) by applying the rewriting rules \(\pi_1, \ldots, \pi_n\) \(\overset{\pi_1, \ldots, \pi_n}{\Rightarrow}\) \(\beta\). The weight function is extended to derivations such that \(\mu(\pi_1 \cdots \pi_n) = \mu(\pi_1) \cdots \mu(\pi_n)\), for all \(\pi_1, \ldots, \pi_n \in R^*\). A nonterminal \(A\) is **productive** if there exists a derivation \(\pi_1, \ldots, \pi_k\) such that \(A \overset{\pi_1, \ldots, \pi_k}{\Rightarrow} u, u \in \Sigma^*\). A nonterminal \(A\) is accessible from a nonterminal \(B\) if there exist derivations \(\pi_1, \ldots, \pi_k\) such that \(B \overset{\pi_1, \ldots, \pi_k}{\Rightarrow} \eta A\xi\) where \(\eta, \xi \in (\Sigma \cup N)^*\) (if \(A\) is accessible from \(B\), then it is simply accessible). A nonterminal \(A\) is **useful** if it is accessible and productive (otherwise, it is **useless**).

A weighted context-free grammar \(G = (\Sigma, N, A_1, R, \mu)\) over the probability semiring \((\mathbb{R}_+, +, \cdot, 0, 1)\) is called a **stochastic context-free grammar** (SCFG) if the weight \(p\) maps all rules to the real unit interval.
A SCFG is reduced if \( p(A \rightarrow \gamma) > 0 \) for all \( A \rightarrow \gamma \in R \) and each nonterminal \( A \), and all nonterminals are useful. In this paper we consider only reduced SCFGs. In addition, we assume that the SCFG is proper, which means that the weight function \( p \) gives us a probability distribution over the rules that we can apply, i.e., \( \sum_{j=1}^{|R_i|} p(A_i \rightarrow B_{ij}) = 1 \) for all \( 1 \leq i \leq |N| \).

For a stochastic context-free grammar \( G = (\Sigma, N, A_1, R, p) \) we define the subgrammar \( G'_i = (\Sigma, N'_i, A_i, R'_i, p') \) with the start symbol \( A_i \), where \( N'_i \) is the set which consists of \( A_i \) and nonterminals accessible from \( A_i \) and \( R'_i \subseteq R \) is the set of rules in which only nonterminals from \( N'_i \) appear as premises and \( p'(\pi) = p(\pi) \) for each \( \pi \in R'_i \). Note that if \( G \) is reduced, then \( G'_i \) also has this property.

3. Moment generating function of SCFG

Let \( G = (\Sigma, N, A_1, R, p) \) be a stochastic context-free grammar, \( \Omega \) the set of all derivations in \( G \) and \( \Omega_i \) the set of all derivations starting at \( A_i \in N \). The grammar \( G \) is consistent if

\[
\sum_{\pi \in \Omega_i} p(\pi) = 1,
\]

for \( 1 \leq i \leq |N| \). Booth and Thompson [1] gave the consistency condition for the start symbol \( S = A_1 \) by the following theorem.

**Theorem 3.1.** A reduced stochastic context-free grammar \( G \) is consistent if \( \rho(M) < 1 \), where \( \rho(M) \) is the absolute value of the largest eigenvalue of the expectation matrix \( M = [M_{i,n}] \), \( 1 \leq i, n \leq |N| \) defined by

\[
M_{i,n} = \sum_{j=1}^{|R_i|} p(A_i \rightarrow B_{ij}) r_n(i, j),
\]

where \( r_n(i, j) \) denotes the number of times the nonterminal \( A_n \) appears on the right hand side of the rule \( \pi = A_i \rightarrow B_{ij} \).

Note that the expectation matrices \( M^{(\ell)} \) of all subgrammars \( G'_i \) are the principal submatrices of \( M \), and according to [2] (Corollary 8.1.20) \( \rho(M^{(\ell)}) \leq \rho(M) \) and \( G'_i \) are also consistent, i.e.,

\[
\sum_{\pi \in \Omega_i} p(\pi) = 1.
\]

For the vector function \( X : \Omega \rightarrow \mathbb{R}^D \), we define the \( i \)-th moment generating function (MGF) of \( X \), as the function \( M_X : \mathbb{R}^D \rightarrow \mathbb{R} \) where

\[
M^{(\ell)}_X(t) = \sum_{\pi \in \Omega_i} p(\pi) e^{tX(\pi)},
\]

for all \( t \in \mathbb{R}^D \). The \( i \)-th cross-moment of an order \( \nu = (\nu_1, \ldots, \nu_D) \) is defined with

\[
m^{(\nu)}_i[X] = \sum_{\pi \in \Omega_i} p(\pi) \cdot X_{1}(\pi)^{\nu_1} \cdots X_{D}(\pi)^{\nu_D} = \sum_{\pi \in \Omega_i} p(\pi) X(\pi)^\nu,
\]

where \( X(\pi) = [X_{1}(\pi), \ldots, X_{D}(\pi)]^T \). The cross-moment can be retrieved from the MGF by differentiating [3]:

\[
m^{(\nu)}_i[X] = \frac{\partial^{\nu_1}M^{(0)}_X(t)}{\partial^{\nu_1}t_1 \cdots \partial^{\nu_D}t_D} \big|_{t=0} = D_{\nu}[M^{(0)}_X],
\]
The $i$-th conditional moment generating function, $M_{X|u}^{(i)}$, and the $i$-th conditional cross-moment of an order $\nu$, $\nu_{ij}^{(i)}[X \mid u]$, are defined in the similar manner if the summing is performed over the set of all derivations starting at $A_i$ and ending with a string $u \in \Sigma^*$.

The direct computation of (11) by enumerating all derivations is inefficient, since it requires the $O(|\Omega|)$ operations, and it even becomes infeasible when $\Omega$ is an infinite set. On the other hand, if we can derive the expressions for efficient computation of the moments generating function (10), the moment can be retrieved by differentiation. The expression can be obtained if the random vector $X$ can be represented as the sum of random vectors $Y : R \to \mathbb{R}$:

$$X(\pi_1 \cdots \pi_N) = Y(\pi_1) + \cdots + Y(\pi_N),$$

for all $\pi_1 \cdots \pi_N \in \Omega$. Then, for $G = (\Sigma, N, A_1, R, p)$, we can construct $\tilde{G} = (\Sigma, N, A_1, R, \mu)$, the moment generating grammar, with the weight function $\mu : R \to \mathbb{C}$, defined with

$$\mu(\pi) = p(\pi)e^{\tau Y(\pi)}$$

for all $\pi \in R$. A derivation $\pi = \pi_1 \cdots \pi_N$ in $G$ with the weight $p(\pi) = p(\pi_1) \cdots p(\pi_N)$ is also a derivation in $\tilde{G}$, for which the weight is given with

$$\mu(\pi) = \mu(\pi_1) \cdots \mu(\pi_N) = p(\pi_1)e^{\tau Y(\pi_1)} \cdots p(\pi_N)e^{\tau Y(\pi_N)} = p(\pi)e^{\tau X(\pi)}.$$ 

The MGF of $X$ can now be can be expressed as the sum of derivation weights in $\tilde{G}$ as

$$M_X(t) = \sum_{\pi \in \Omega} p(\pi)e^{\tau X(\pi)} = \sum_{\pi \in \Omega} \mu(\pi).$$

Thus, the problem of MGF computation is reduced to the problem of the partition function computation [5], and the conditional MGF can be computed using the inside algorithm [5] over the semiring of $\nu$-continuous functions at zero. In the following sections we show how the expressions for the cross-moments and conditional cross-moments can be derived from (16).

4. Cross-moments computation of SCFG

Let $G = (\Sigma, N, A_1, R, \mu)$ be a weighted context-free grammar over a commutative semiring $(\mathbb{K}, +, \cdot, 1, 0)$ endowed with a topology $\tau$. Assuming that for $1 \leq i \leq |N|$ the infinite collections $\{\mu(\pi)\}_{\pi \in \Omega}$ are summable in $\tau$, and that the distributive law for infinite sums holds, we define the partition function $Z : N \to \mathbb{K}$ which to every nonterminal $A_i \in N$ associates the sum

$$Z_i = \sum_{\pi \in \Omega} \mu(\pi).$$

By factoring out the first rewriting of each derivation in the sum, using the distributive law, the partition function can be expressed with the system [13]:

$$Z_i = \sum_{j=1}^{|R_i|} \mu(A_i \rightarrow B_{i,j}) \cdot \prod_{L=1}^{|N|} Z_k(L_{i,j}),$$

where $1 \leq i \leq |N|$.

Now, let $G = (\Sigma, N, A_1, R, \mu)$ be the moment generating grammar for $G = (\Sigma, N, A_1, R, p)$ with

$$\mu(\pi) = p(\pi)e^{\tau Y(\pi)}$$

(19)
\[ X(\pi_1 \cdots \pi_N) = Y(\pi_1) + \cdots + Y(\pi_N). \quad (20) \]

According to the discussion made in the section 3, the value of the partition function at the nonterminal \( A_i \) corresponds to the \( i \)-th moment generating function, \( Z_i = M^{(i)}_X \), and the \( i \)-th cross-moment
\[ m_i^{(a)}(X) = D_a(M^{(i)}_X) = D_a[Z_i] = \sum_{\pi \in \Omega_i} p(\pi)X(\pi)^a, \quad (21) \]
and it can be computed by differentiating (18) and solving the resulting equation. Note that
\[ m_i^{(0)}(X) = D_0[Z_i] = \left( \sum_{\pi \in \Omega_i} p(\pi)e^{ix}(\pi) \right)_{x=0} = \sum_{\pi} p(\pi) = 1, \quad (22) \]
for all \( 1 \leq i \leq |\mathcal{N}| \). The cross-moments of higher order can be obtained by applying the generalized Leibniz’s formula (6) to (18), which leads us to the following system
\[ m_i^{(a)}(X) = \sum_{j=1}^{[\mathcal{R}]} \sum_{\beta \leq \alpha} \binom{\alpha}{\beta} D_{a-\beta}[\mu(A_i \rightarrow B_{ij})] \cdot D_{\beta}[\prod_{k=1}^{[\mathcal{N}]} Z_k^{(n_j)}], \quad (23) \]
where,
\[ D_{a-\beta}[\mu(A_i \rightarrow B_{ij})] = p(A_i \rightarrow B_{ij}) \cdot Y(A_i \rightarrow B_{ij})^{a-\beta}, \quad (24) \]
since \( p(\pi) = p(\pi)e^{ix}(\pi) \), for \( \pi \in \mathcal{R} \). According to the generalized Leibniz’s rule (7), we have
\[ D_{\beta}[\prod_{k=1}^{[\mathcal{N}]} Z_k^{(n_j)}] = \sum_{\gamma_1, \ldots, \gamma_{[\mathcal{R}]} \neq \emptyset} \binom{\beta}{\gamma_1, \ldots, \gamma_{[\mathcal{R}]}} \prod_{k=1}^{[\mathcal{N}]} D_{\gamma_k}[z_k^{(n_j)}] \quad (25) \]
and
\[ D_{\gamma_k}[z_k^{(n_j)}] = D_{\gamma_k} \left( \prod_{l=1}^{[\mathcal{R}]} z_k \right) = \sum_{b_1, \ldots, b_{[\mathcal{R}]}(n_j) = y_k} \binom{\gamma_k}{b_1, \ldots, b_{[\mathcal{R}]}(n_j)} \prod_{l=1}^{[\mathcal{R}]} m_{k}^{(n_j)}(X). \quad (26) \]

By substituting (26) and (25) in (23), we obtain:
\[ m_i^{(a)}(X) = \sum_{j=1}^{[\mathcal{R}]} \sum_{\beta \leq \alpha} Q_{i,j}(\alpha, \beta), \quad (27) \]
where
\[ Q_{i,j}(\alpha, \beta) = \binom{\alpha}{\beta} p(A_i \rightarrow B_{ij}) \cdot Y(A_i \rightarrow B_{ij})^{a-\beta}: \]
\[ \sum_{\gamma_1, \ldots, \gamma_{[\mathcal{R}]} \neq \emptyset} \binom{\beta}{\gamma_1, \ldots, \gamma_{[\mathcal{R}]}} \prod_{k=1}^{[\mathcal{N}]} b_{1, \ldots, b_{[\mathcal{R}]}(n_j) = y_k} \binom{\gamma_k}{b_1, \ldots, b_{[\mathcal{R}]}(n_j)} \prod_{l=1}^{[\mathcal{R}]} m_{k}^{(n_j)}(X). \quad (28) \]

To solve the system (27), we split it into two parts: one depending and the other not depending on \( m_i^{(a)}(X) \):
\[ m_i^{(a)}(X) = \sum_{j=1}^{[\mathcal{R}]} Q_{i,j}(\alpha, \alpha) + \sum_{\beta \leq \alpha} Q_{i,j}(\alpha, \beta), \quad (29) \]
Further, if we set
\[ Q_{ij}(\alpha, \alpha) = p(A_i \rightarrow B_{ij}) \cdot W_{ij}(\alpha) \] (30)
and
\[ W_{ij}(\alpha) = \sum_{\gamma_1, \ldots, \gamma_n=\alpha} \left( \gamma_1, \ldots, \gamma_n \right) \prod_{k=1}^{[N]} \sum_{\delta_1, \ldots, \delta_n(\alpha, j)=\gamma_k} \left( \delta_1, \ldots, \delta_n(\alpha, j) \right) \prod_{i=1}^{\gamma_k} m_k^{\delta_i}[X]. \] (31)

Finally, after use of \( m_k^{\delta_i}[X] = 1 \) we obtain
\[ H_{ij}(y_1, \ldots, y_{[N]}) = \sum_{\delta_1, \ldots, \delta_n(\alpha, j)=\alpha} \left( \delta_1, \ldots, \delta_n(\alpha, j) \right) \prod_{i=1}^{\gamma_k} m_k^{\delta_i}[X]. \] (34)
which can be rewritten using the same procedure as
\[ H_{ij}(y_1, \ldots, y_{[N]}) = \sum_{s=1}^{r_{n}(B_{ij})} m_n^{\alpha}[X] + \sum_{\delta_1, \ldots, \delta_n(\alpha, j)<\alpha} \left( \delta_1, \ldots, \delta_n(\alpha, j) \right) \prod_{i=1}^{\gamma_k} m_k^{\delta_i}[X] = \]
\[ r_n(B_{ij}) \cdot m_n^{\alpha}[X] + \sum_{\delta_1, \ldots, \delta_n(\alpha, j)<\alpha} \left( \delta_1, \ldots, \delta_n(\alpha, j) \right) \prod_{i=1}^{\gamma_k} m_k^{\delta_i}[X]. \] (36)

By substituting (35) in (33) it follows that:
\[ W_{ij}(\alpha) = \sum_{n=1}^{[N]} r_n(B_{ij}) \cdot m_n^{\alpha}[X] + \]
\[ \sum_{n=1}^{[N]} \sum_{\delta_1, \ldots, \delta_n(\alpha, j)=\alpha} \left( \delta_1, \ldots, \delta_n(\alpha, j) \right) \prod_{i=1}^{\gamma_k} m_k^{\delta_i}[X] + \sum_{\gamma_1, \ldots, \gamma_n=\alpha} H_{ij}(y_1, \ldots, y_{[N]}). \] (37)
Further, by substituting (37) and (30) in (29), the moment can be expressed with:

\[ m_i^{(a)} \{X\} = \sum_{j=1}^{\lvert R \rvert} p(A_i \rightarrow B_{i,j}) \sum_{n=1}^{\lvert N \rvert} m_n \{X\} + \sum_{j=1}^{\lvert R \rvert} p(A_i \rightarrow B_{i,j}) \sum_{n=1}^{\lvert N \rvert} \left( \delta_{1}, \ldots, \delta_{n(B_{i,j})} \right) m_n \{X\} + \sum_{j=1}^{\lvert R \rvert} p(A_i \rightarrow B_{i,j}) \sum_{n=1}^{\lvert N \rvert} Q_{i,j} \{\alpha, \beta\}, \quad (38) \]

where \( H_{i,j}(Y_1, \ldots, Y_N) \) and \( Q_{i,j}(\alpha, \beta) \) are given with (32) and (28). Finally, if we introduce

\[ c_i^{(a)} = \sum_{j=1}^{\lvert R \rvert} p(A_i \rightarrow B_{i,j}) \sum_{n=1}^{\lvert N \rvert} \left( \delta_{1}, \ldots, \delta_{n(B_{i,j})} \right) m_n \{X\} + \sum_{j=1}^{\lvert R \rvert} p(A_i \rightarrow B_{i,j}) \sum_{n=1}^{\lvert N \rvert} Q_{i,j} \{\alpha, \beta\}, \quad (39) \]

the equation (38) can be more compactly written as:

\[ m_i^{(a)} \{X\} = \sum_{n=1}^{\lvert N \rvert} M_{i,n} \cdot m_n^{(a)} \{X\} + c_i^{(a)}, \quad (40) \]

or in matrix form:

\[ m^{(a)} = M \cdot m^{(a)} + c^{(a)}, \quad (41) \]

where \( m^{(a)} = [m_1^{(a)} \{X\}, \ldots, m_N^{(a)} \{X\}]^T \) is the cross-moment vector, \( c^{(a)} = [c_1^{(a)}, \ldots, c_N^{(a)}]^T \) and \( M \) is the momentum matrix defined in Theorem 3.1. Since we assume that the condition \( \rho(M) < 1 \) given in Theorem 3.1 is satisfied, \( I - M \) is invertible, and the matrix equation has a unique solution given with

\[ m^{(a)} = (I - M)^{-1} c^{(a)}. \quad (42) \]

Provided that we have computed the inverse \((I - M)^{-1}\), which does not depend on \( \alpha \), the cross-moment is completely determined by the term \( c^{(a)} \), which depends on all cross-moments of order lower than \( \alpha \) and can be computed using (39). In the following section, we derive \( c^{(a)} \) for scalar random variables up to the second order, and retrieve the previous results for the first and second order moments [1], [8] as a special case of the equation (42).

4.1. First order moments

In the case of the first order moments \( \alpha = 1 \) and the expression (21) reduces to the expectation of \( X \) with respect to the sample space \( \Omega_i \),

\[ n_i^{(1)} \{X\} = \sum_{\pi \in \Omega_i} p(\pi)X(\pi). \quad (43) \]

\[ ^1 \text{One dimensional multi-indexes are written without parentheses.} \]
The moment vector, \( m^{(\alpha)} = [m^{(1)}_1(X), \ldots, m^{(1)}_{|X|}(X)] \), is computed as in the equation (42),

\[
m^{(1)} = (1 - M)^{-1} c^{(1)},
\]

where \( c^{(1)} = [c^{(1)}_1, \ldots, c^{(1)}_{|X|}]^T \). The first and second sum in the expression (39) for \( c^{(\alpha)}_i \) reduce to zero and \( c^{(1)}_i = 4 \sum_{j=1}^{|X|} Q_i (1, 0) \), or, after use of the expression (28) for \( Q_i (\alpha, \beta) \),

\[
c^{(1)}_i = \sum_{j=1}^{|X|} p(A_i \rightarrow B_{i,j}) \cdot Y(A_i \rightarrow B_{i,j}).
\]

Let, \( \pi_1 \cdots \pi_N \) be a derivation starting at the start symbol \( A_1 \) and ending with a string \( u \in \Sigma^* \). If we set \( Y(A_1 \rightarrow B_{i,j}) = 1 \), according to (20), we have \( X(\pi_1 \cdots \pi_N) = \sum_{n=1}^N Y(\pi_n) = N \), i.e., \( X \) is the length of the derivation. According to the expression (43), the moment \( m^{(1)}_1(X) \) is expected derivation length which agrees with (11) and (8).

Similarly, if we set \( Y(A_1 \rightarrow B_{i,j}) = \sum_{j=1}^{|X|} l_e(i, j) \), where \( l_e(i, j) \) denotes the number of terminals in the string \( B_{i,j} \), the variable \( X(\pi_1 \cdots \pi_N) \) reduces to the length of word derived from \( \pi_1 \cdots \pi_N \). In this case, the moment \( m^{(1)}_1(X) \) reduces to the expected string length and the formula (45) reduces to the result from (11).

4.2. Second order moments

The formula for the second order moments is a somewhat more complicated. In the case when \( \alpha = 2 \), we have that \( c^{(\alpha)}_i \) is reduced to:

\[
c^{(2)}_i = \sum_{j=1}^{|X|} p(A_i \rightarrow B_{i,j}) \sum_{n=1}^{|X|} \sum_{\delta_1, \ldots, \delta_{n}(B_{i,j}) = \alpha} \left( \begin{array}{c} 2 \\ \delta_1, \ldots, \delta_{n}(B_{i,j}) = \alpha \end{array} \right) \prod_{l=1}^{n(B_{i,j})} m^{(b)}_n \cdot X
\]

The first sum in the previous expression can be transformed to:

\[
\sum_{j=1}^{|X|} p(A_i \rightarrow B_{i,j}) \sum_{n=1}^{|X|} \sum_{\delta_1, \ldots, \delta_{n}(B_{i,j}) = \alpha} \left( \begin{array}{c} 2 \\ \delta_1, \ldots, \delta_{n}(B_{i,j}) = \alpha \end{array} \right) \prod_{l=1}^{n(B_{i,j})} m^{(b)}_n \cdot X
\]

To compute the second sum we introduce \( H^{(a,b)}_{i,j} (\gamma_1, \ldots, \gamma_{|X|}) \), which is \( H_{i,j} (\gamma_1, \ldots, \gamma_{|X|}) \) with \( \gamma_1 = \gamma_{b-1} = 1 \) and with all other \( \gamma \)-s equals to zero. We have:

\[
H^{(a,b)}_{i,j} (\gamma_1, \ldots, \gamma_{|X|}) = 2 \cdot \sum_{\delta_1, \ldots, \delta_{n(B_{i,j})} = \gamma_1} \left( \begin{array}{c} \gamma_1 \\ \delta_1, \ldots, \delta_{n(B_{i,j})} = \gamma_1 \end{array} \right) \prod_{l=1}^{n(B_{i,j})} m^{(b)}_n \cdot X
\]

\[
\sum_{\delta_1, \ldots, \delta_{n(B_{i,j})} = \gamma_1} \left( \begin{array}{c} \gamma_1 \\ \delta_1, \ldots, \delta_{n(B_{i,j})} = \gamma_1 \end{array} \right) \prod_{l=1}^{n(B_{i,j})} m^{(b)}_n \cdot X \sum_{k=1}^{|X|} \sum_{\delta_1, \ldots, \delta_{n(B_{i,j})} = \gamma_1} \left( \begin{array}{c} \gamma_1 \\ \delta_1, \ldots, \delta_{n(B_{i,j})} = \gamma_1 \end{array} \right) \prod_{l=1}^{n(B_{i,j})} m^{(b)}_n \cdot X.
\]
and

\[
F_{i,j}^{(a,b)}(y_1, \ldots, y_{|N|}) = 2 \cdot \sum_{k=1}^{r_a(B_{i,j})} m_k^{(1)}(x) \cdot \sum_{b=1}^{r_b(B_{i,j})} m_b^{(1)}(x) = 2 \cdot r_a(B_{i,j}) \cdot r_b(B_{i,j}) \cdot m_a^{(1)}(x)m_b^{(1)}(x). \tag{49}
\]

By substituting the second sum in (46)

\[
\sum_{j=1}^{[R]} p(A_i \rightarrow B_{i,j}) \sum_{y_1, \ldots, y_{|N|} \in \mathcal{R}} H_{i,j}(y_1, \ldots, y_{|N|}) = \sum_{j=1}^{[R]} p(A_i \rightarrow B_{i,j}) \sum_{y_1, \ldots, y_{|N|} \in \mathcal{R}} H_{i,j}^{(a,b)}(y_1, \ldots, y_{|N|}) =
\]

\[
= 2 \cdot \sum_{j=1}^{[R]} p(A_i \rightarrow B_{i,j}) \sum_{a=1}^{[N]} \sum_{b=1}^{[N]} r_a(B_{i,j}) \cdot r_b(B_{i,j}) \cdot m_a^{(1)}(x) m_b^{(1)}(x) =
\]

\[
= \sum_{j=1}^{[R]} p(A_i \rightarrow B_{i,j}) \sum_{a=1}^{[N]} r_a(B_{i,j}) \cdot r_a(B_{i,j}) \cdot m_a^{(1)}(x) m_a^{(1)}(x) - \sum_{j=1}^{[R]} p(A_i \rightarrow B_{i,j}) \sum_{n=1}^{[N]} r_n(B_{i,j})^2 m_n^{(1)}(x)^2. \tag{50}
\]

Now, (46) reduces to

\[
c_i^{(2)} = CR_i + \sum_{j=1}^{[R]} Q_{i,j}(2,0) + \sum_{j=1}^{[R]} Q_{i,j}(2,1) \tag{51}
\]

where

\[
CR_i = \sum_{j=1}^{[R]} p(A_i \rightarrow B_{i,j}) \sum_{a=1}^{[N]} \sum_{b=1}^{[N]} r_a(B_{i,j}) \cdot r_b(B_{i,j}) \cdot m_a^{(1)}(x) m_b^{(1)}(x) - \sum_{j=1}^{[R]} p(A_i \rightarrow B_{i,j}) \sum_{n=1}^{[N]} r_n(B_{i,j})^2 m_n^{(1)}(x)^2. \tag{52}
\]

and

\[
Q_{i,j}(2,0) = p(A_i \rightarrow B_{i,j}) \cdot \gamma(A_i \rightarrow B_{i,j})^2, \tag{53}
\]

\[
Q_{i,j}(2,1) = 2 \cdot p(A_i \rightarrow B_{i,j}) \cdot \gamma(A_i \rightarrow B_{i,j}) \sum_{n=1}^{[N]} m_n^{(1)}(x) = 2 \cdot p(A_i \rightarrow B_{i,j}) \cdot \gamma(A_i \rightarrow B_{i,j}) \sum_{n=1}^{[N]} m_n^{(1)}(x). \tag{54}
\]

By setting $\gamma(A_i \rightarrow B_{i,j}) = 1$ for all $A_i \rightarrow B_{i,j} \in \mathcal{R}$, $X$ becomes derivation length. The formula for computing the second order moments of derivation length is given in [8] and it can be derived from the equation (51) since

\[
\sum_{j=1}^{[R]} Q_{i,j}(2,0) = \sum_{j=1}^{[R]} p(A_i \rightarrow B_{i,j}) \cdot \gamma(A_i \rightarrow B_{i,j})^2 = \sum_{j=1}^{[R]} p(A_i \rightarrow B_{i,j}) = 1, \tag{55}
\]

\[
\sum_{j=1}^{[R]} Q_{i,j}(2,1) = 2 \cdot \sum_{n=1}^{[N]} \left( \sum_{j=1}^{[R]} p(A_i \rightarrow B_{i,j}) \cdot r_n(B_{i,j}) m_n^{(1)}(x) \right) = 2 \cdot \sum_{n=1}^{[N]} c_{i,n} m_n^{(1)}(x) = 2 \cdot m_1^{(1)}(x) - 2, \tag{56}
\]

where the last equation follows from (42), and

\[
c_i^{(2)} = CR_i + 2 \cdot m_1^{(1)}(x) - 1. \tag{57}
\]

Finally, by substituting (57) in (42) we obtain

\[
m^{(a)}(x) = (1 - M)^{-1} \cdot (CR_i + 2 \cdot m_1 - 1), \tag{58}
\]

where $CR_i = [CR_1, \ldots, CR_{|N|}]$ and $1 = [1, \ldots, 1]$, in agreement with [8].
5. Conditional cross-moments computation for SCFGs

Let \( \tilde{G} = (\Sigma, N, A_1, R, \mu) \) be a weighted context-free grammar over a commutative semiring \((K, +, \cdot, 1, 0)\), and \( \Omega_i(u) \) be a set of all derivations which derive \( u \in \Sigma^* \) starting from \( A_i \). The inside weight of the weighted grammar \( \tilde{G} \) is the function \( \sigma_i : N \times \Sigma^* \to C_\nu \), defined as the sum of weights of all derivations starting with \( A_i \) and ending with \( u \),

\[
\sigma_i(u) = \sum_{\pi \in \Omega_i(u)} \mu(\pi),
\]

for \( 1 \leq i \leq |R| \) and \( u \in \Sigma^* \). Let \( A_i \rightarrow B_{i,j} \in R \) and

\[
B_{i,j} = v_1A_i v_2 A_i \cdots v_k A_i v_{k+1} u,
\]

where \( v_i \in \Sigma^* \) and \( A_i \in N \). For the cycle-free reduced grammars the inside weight can be computed using the inside algorithm \([5]\) and \([17]\) which after recursive application of

\[
\sigma_i(u) = \sum_{j=1}^{|R|} \sum_{u_1, u_2, \ldots, u_\beta \in \Sigma^*} \mu(A_i \rightarrow B_{i,j}) \cdot \prod_{j=1}^k \sigma_j(u_j)
\]

ends with the equation in which only rules \( A_i \rightarrow u, u \in \Sigma^* \) appears on the right hand side:

\[
\sigma_i(u) = \mu(A_i \rightarrow u).
\]

Now, let \( G = (\Sigma, N, A_1, R, \mu) \) be the moment generating grammar for \( \tilde{G} = (\Sigma, N, A_1, R, \mu) \) with

\[
\mu(\pi) = p(\pi)e^{\pi Y(\pi)}
\]

and

\[
X(\pi_1, \ldots, \pi_N) = Y(\pi_1) + \cdots + Y(\pi_N).
\]

According to the discussion in Section 3 the value of the partition function at the nonterminal \( A_i \) corresponds to the \( i \)-th moment generating function, \( \sigma_i = M^{(0)}_{X|u} \), and the \( i \)-th cross-moment

\[
m_i^{(0)} \{ X \mid u \} = D_\alpha \{ M^{(0)}_{X|u} \} = D_\alpha \{ \sigma_i(u) \} = \sum_{\pi \in \Omega_i(u)} p(\pi)X(\pi)^i,
\]

and it can be computed by differentiating \( 61 \) and solving the resulting equation:

\[
D_\alpha \{ \sigma_i(u) \} = \sum_{j=1}^{\mid R \mid} \sum_{u_1, u_2, \ldots, u_\beta \in \Sigma^*} \sum_{\beta \leq \alpha} \beta \cdot D_\alpha - \beta \{ \mu(A_i \rightarrow B_{i,j}) \} \cdot D_\beta \{ \prod_{j=1}^k \sigma_j(u_j) \}.
\]

By use of the generalized Leibniz’s rule \([6]\) we have

\[
D_\alpha \{ \sigma_i(u) \} = \sum_{j=1}^{\mid R \mid} \sum_{u_1, u_2, \ldots, u_\beta \in \Sigma^*} \sum_{\beta \leq \alpha} \beta \cdot D_\alpha - \beta \{ \mu(A_i \rightarrow B_{i,j}) \} \cdot \sum_{y_1, y_2, \ldots, y_k} \left( y_1, \ldots, y_k \right) \prod_{j=1}^k D_{y_j} \{ \sigma_j(u_j) \},
\]

and after use of the equality

\[
D_\alpha - \beta \{ \mu(A_i \rightarrow B_{i,j}) \} = p(A_i \rightarrow B_{i,j}) \cdot Y(A_i \rightarrow B_{i,j})^{\alpha - \beta},
\]

11
the recursive equation becomes

\[ m^{(\alpha)}_i(X | u) = \sum_{j=1}^{[N]} \sum_{u_1, u_2, \ldots, u_j} \sum_{\beta \leq \alpha} \left( \frac{\alpha}{\beta} \right) p(A_i \rightarrow B_{ij}) \cdot Y(A_i \rightarrow B_{ij}) \alpha - \beta. \]

\[ \sum_{Y_1, \ldots, Y_k} \prod_{j=1}^{k} \left( \frac{\beta}{Y_j} \right) m^{(\gamma)}_{ij}(X | u). \] (69)

The base case is derived by differentiating (62):

\[ m^{(\gamma)}_i(X | u) = D_{\gamma} \phi(A_i | u) = D_{\gamma} \mu(A_i | u) = p(A_i | u) \cdot Y(A_i | u). \] (70)

The previously developed algorithms for the cross-moments computation generalize the algorithms by Li and Eisner \[ [11] \], for the second moment of order \( \alpha \) having the property \( D_{\alpha} \leq 0 \), which is the generalization of the first and second order entropy semirings. After that we run the inside algorithm over the binomial semiring and show that this approach yields the equations \[ (69)-(70) \].

5.1. Conditional cross-moments computation using the inside algorithm over the binomial semiring

Let \( \nu = (\nu_1, \ldots, \nu_d) \in \mathbb{N}^d \) be a multi-index and \( |\nu| = \nu_1 + \cdots + \nu_d \) and \( N_\nu \) be the cardinality of the set \( \{ \alpha \leq \nu \} \). We define the map \( \phi : C_\nu \rightarrow \mathbb{R}^{N_\nu} \), which to every \( f \in C_\nu \) associates the tuple \( f^\prime \), indexed by a multi-index \( \alpha \leq \nu \), such that

\[ f^\prime = \phi(f) = (D_{\alpha} f) \] (71)

Conversely, we define the map \( \tilde{\phi} : \mathbb{R}^{N_\nu} \rightarrow C_\nu \), which to for every \( f = (f_\alpha)_{\alpha \leq \nu} \in \mathbb{R}^{N_\nu} \) associates the polynomial function

\[ f = \tilde{\phi}(f); \quad f(t) = \sum_{\alpha \leq \nu} \frac{f_\alpha}{\alpha!} \cdot t^\alpha; \quad \text{for all } t \in \mathbb{R}, \] (72)

having the property \( D_{\alpha} f = f_\alpha \) and \( \phi(f) = (f_\alpha)_{\alpha \leq \nu} = f^\prime \). Hereinafter, for the pair \( (f, f) \) we always assume \( \tilde{\tilde{\phi}} = \phi(f) \) and \( \tilde{\phi}(f) = (f_\alpha)_{\alpha \leq \nu} = f^\prime \). After use of the generalized Leibniz's product rule \[ \text{(II)}, \] we have

\[ f \oplus \tilde{\phi} = \phi(f + g) = (D_{\alpha} f + D_{\alpha} g) \] (73)

\[ \tilde{f} \odot \tilde{\phi} = \phi(f \cdot g) = \sum_{\beta \leq \nu} \frac{\alpha}{\beta} D_{\beta} \tilde{f} \cdot D_{\alpha - \beta} \tilde{g} \] (74)

By map \( \phi \), the zero function is mapped to the tuple \( \tilde{\phi}(0) = (D_{\alpha} 0)_{\alpha \leq \nu}, \) which is 0 at all \( \alpha \)-positions, while the identity function is mapped to \( \tilde{1} = (D_{\alpha} 1)_{\alpha \leq \nu}, \) which is 1 at the position \( \nu = 0, \) since \( D_{\alpha} 1 \) is identity map and 0 at all other \( \alpha \)-positions:

\[ \tilde{\phi}(0) = (0, 0, \ldots, 0), \quad \tilde{1} = (1, 0, \ldots, 0). \] (75)
With these settings, the tuple \((\mathbb{R}^{N_1}, \oplus, \ominus, 0, 1)\) is a commutative semiring, which we call the binomial semiring of an order \(\nu\).

**Example 5.1.** The first order entropy semiring \([2]\) is a special case of the binomial semiring for \(\alpha = 1\). In this case the binomial semiring reduces to \((\mathbb{R}^{2}, \oplus, \ominus, 0, 1)\), with \(\tilde{\nu} = (0, 0), \tilde{\nu} = (1, 0)\) and

\[
\begin{align*}
\tilde{f} \oplus \tilde{g} &= (f_0 + g_0, f_1 + g_1), \\
\tilde{f} \ominus \tilde{g} &= (f_0 \cdot g_0, f_0 \cdot g_1 + f_1 \cdot g_0),
\end{align*}
\]

where the first component corresponds to \(\alpha = 0\) and the second one to \(\alpha = 1\), in agreement with \([2]\).

**Example 5.2.** The second order entropy semiring \([11]\) is a special case of the binomial semiring for \(\alpha = 1\). In this case the binomial semiring reduces to \((\mathbb{R}^{4}, \oplus, \ominus, 0, 1)\), with \(\tilde{\nu} = (0, 0, 0, 0), \tilde{\nu} = (1, 0, 0, 0)\) and

\[
\begin{align*}
\tilde{f} \oplus \tilde{g} &= (f_{00} + g_{00}, f_{01} + g_{01}, f_{10} + g_{10}, f_{11} + g_{11}), \\
\tilde{f} \ominus \tilde{g} &= (f_{00} \cdot g_{00}, f_{00} \cdot g_{01} + f_{01} \cdot g_{00}, f_{01} \cdot g_{10} + f_{10} \cdot g_{00}, f_{00} \cdot g_{11} + f_{01} \cdot g_{10} + f_{10} \cdot g_{01} + f_{11} \cdot g_{00}),
\end{align*}
\]

where the first component corresponds to \(\alpha = (0, 0)\), the second to \(\alpha = (0, 0)\), the third to \(\alpha = (0, 0)\), and the fourth to \(\alpha = (0, 0)\), which agrees with \([11]\).

By induction and by use of \([7]\), we can obtain the following equations which hold for arbitrary tuples \(u^{(1)}, \ldots, u^{(N)}\) for \(\mathbb{R}^{|\nu|}\):

\[
(\bigotimes_{n=1}^{N} \tilde{u}^{(n)})_\alpha = \phi_{\alpha} \left( \prod_{n=1}^{N} \tilde{u}^{(n)} \right) = D_{\alpha} \left( \prod_{n=1}^{N} \tilde{u}^{(n)} \right) = \sum_{\beta_1 + \ldots + \beta_N = \alpha} \left( \beta_1, \ldots, \beta_N \right) \prod_{n=1}^{N} u^{(n)}_{\beta_n}. \tag{80}
\]

Accordingly, by setting

\[
\mu(\pi_n) = \left( \mu\alpha(\pi_n) \right)_{\alpha \leq \nu} = \left( p(\pi_n) Y(\pi_n)^\alpha \right)_{\alpha \leq \nu} \tag{81}
\]

for each derivation \(\pi_1 \ldots \pi_N \in \Omega_i\) we have

\[
(\bigotimes_{n=1}^{N} \mu(\pi_n))_\alpha = \sum_{\beta_1 + \ldots + \beta_N = \alpha} \left( \beta_1, \ldots, \beta_N \right) \prod_{n=1}^{N} \beta(\pi_n) = \sum_{\beta_1 + \ldots + \beta_N = \alpha} \left( \beta_1, \ldots, \beta_N \right) \prod_{n=1}^{N} p(\pi_n) Y(\pi_n)^{\beta_n} \tag{82}
\]

In addition,

\[
(\bigoplus_{n=1}^{N} \tilde{u}^{(n)})_\alpha = \phi_{\alpha} \left( \sum_{n=1}^{N} \tilde{u}^{(n)} \right) = D_{\alpha} \left[ \sum_{n=1}^{N} \tilde{u}^{(n)} \right] = \sum_{n=1}^{N} D_{\alpha} \left( \tilde{u}^{(n)} \right) = \sum_{n=1}^{N} u^{(n)}_{\alpha}. \tag{83}
\]

and the inside weight is

\[
\sigma_i(u) = \bigoplus_{\pi \in \mathcal{I}_i(u)} \mu(\pi) = \bigoplus_{\pi \in \mathcal{I}_i(u)} \bigotimes_{n=1}^{N} \mu(\pi_n) = \left( \sum_{\pi \in \mathcal{I}_i(u)} p(\pi) \cdot X(\pi)^\alpha \right)_{\alpha \leq \nu}. \tag{84}
\]
for $1 \leq i \leq |N|$ and $u \in \Sigma^*$. Let $A_i \rightarrow B_{ij} \in R$ and $B_{ij} = v_1A_i v_2A_i \cdots v_kv_{k+1}$, where $v_i \in \Sigma^*$ and $A_i \in N$. The inside recursive equations in the entropy semiring have the form:

$$
\left( \sigma_i(u) \right)^{(a)} = \left( \bigoplus_{j=1}^{|R|} \bigoplus_{m_1, m_2, \ldots, m_k \in \Sigma^*} \mu(A_i \rightarrow B_{ij}) \otimes \bigotimes_{j=1}^k \sigma_i(u_j) \right)^{a} = \\
\sum_{j=1}^{|R|} \sum_{m_1, m_2, \ldots, m_k \in \Sigma^*} \mu(A_i \rightarrow B_{ij}) \otimes \bigotimes_{j=1}^k \sigma_i(u_j)^{a} =
$$

(85)

and after substituting $\mu(A_i \rightarrow B_{ij})^{a-\beta} = p(A_i \rightarrow B_{ij}) \cdot Y(A_i \rightarrow B_{ij})^{a-\beta}$, and the equation (80) for the product of $k$ terms in the binomial semiring, we get the recursive equation

$$
\left( \sigma_i(u) \right)^{(a-\beta)} = \sum_{j=1}^{|R|} \sum_{m_1, m_2, \ldots, m_k \in \Sigma^*} \sum_{\beta \leq a} \left( \begin{array}{c} \alpha \\ \beta \end{array} \right) p(A_i \rightarrow B_{ij}) \cdot Y(A_i \rightarrow B_{ij})^{a-\beta} = \\
\sum_{y_1, \ldots, y_k = \frac{a}{\beta}}^{k} \left( \begin{array}{c} \beta \\ y_1, \ldots, y_k \end{array} \right) \sigma_i(u_j)^{(\beta,y)}
$$

(87)

with the base case:

$$
\left( \sigma_i(u) \right)^{(\gamma)} = \mu(A_i \rightarrow u) = p(A_i \rightarrow u) \cdot Y(A_i \rightarrow u)^{\gamma}
$$

(88)

which is the same recursive procedure as (69)-(70).

5.2. First order conditional moments

In the case of first order conditional moments $a = 1$, the conditional cross-moment (65) is the expectation of $X$:

$$
m^{(1)}_i(X \mid u) = \sum_{\pi \in \Omega_\Sigma(u)} p(\pi)X(\pi).
$$

(89)

In this case, the recursive equations (69)-(70) reduce to

$$
m^{(0)}_i(X \mid u) = \sum_{j=1}^{|N|} \sum_{m_1, m_2, \ldots, m_k \in \Sigma^*} p(A_i \rightarrow B_{ij}) \cdot \prod_{j=1}^k m^{(0)}_{i_j}(X \mid u_j),
$$

(90)

$$
m^{(1)}_i(X \mid u) = \sum_{j=1}^{|N|} \sum_{m_1, m_2, \ldots, m_k \in \Sigma^*} p(A_i \rightarrow B_{ij}) \cdot Y(A_i \rightarrow B_{ij}) \cdot \prod_{j=1}^k m^{(0)}_{i_j}(X \mid u_j) + \\
p(A_i \rightarrow B_{ij}) \sum_{n=1}^{k} m^{(1)}_{i_n}(X \mid u_n) \prod_{j=1, j \neq n}^k m^{(0)}_{i_j}(X \mid u_j),
$$

(91)
with the base case:

\[ m_i^{(0)}(X | u) = p(A_i \rightarrow u), \quad m_i^{(1)}(X | u) = p(A_i \rightarrow u) \cdot Y(A_i \rightarrow u). \quad (92) \]

In [9], Hwa considered the conditional entropy of the grammar given in Chomsky form for which \( B_{ij} = v_1 A_i v_2 A_i v_3 \) and \( v_1, v_2, v_3 \) are equal to the empty string. The conditional entropy is obtained as the moment \( m_i^{(1)}(X | u) \), where \( X(\pi) = - \log p(\pi) \), for all \( \pi \in \Omega \) and Hwa’s algorithm can be retrieved by imposing Chomsky form condition in (69)-(70), with \( Y(\pi_i) = - \log p(\pi_i) \).

6. Conclusion

In this paper we considered the problem of computing the cross-moments and the conditional cross-moments of a vector variable represented by a stochastic context-free grammar. The recursive formulas for cross-moments of arbitrary order are obtained, and the previously developed formulas for moments [8], [19] and conditional cross-moments [11] are derived as special cases.

References

[1] T. L. Booth, R. A. Thompson, Applying probability measures to abstract languages, IEEE Transactions on Computers 22 (1973) 442–450.
[2] S. B. Cohen, R. J. Simmons, N. A. Smith, Products of weighted logic programs, Theory and Practice of Logic Programming 11 (2011) 263–296.
[3] C. Cortes, M. Mohri, A. Rastogi, M. Riley, On the computation of the relative entropy of probabilistic automata, International Journal of Foundations of Computer Science 19 (2008) 219–242.
[4] L. E. Fraenkel, Formulae for high derivatives of composite functions, Mathematical Proceedings of the Cambridge Philosophical Society 83 (1978) 159–165.
[5] J. Goodman, Semiring parsing, Computational Linguistics 25 (1999) 573–605.
[6] T. E. Harris, The Theory of Branching Processes, Dover Books on Advanced Mathematics, Dover Pubns, 1963.
[7] R. A. Horn, C. R. Johnson, Matrix analysis, Cambridge University Press, New York, NY, USA, 1985.
[8] S. E. Hutchins, Moments of string and derivation lengths of stochastic context-free grammars, Information Sciences 4 (1972) 179–191.
[9] R. Hwa, Sample selection for statistical grammar induction, Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics - Volume 13, pp. 45–52, Morristown, NJ, USA, 2000. Association for Computational Linguistics.
[10] K. Lari, S. J. Young, The estimation of stochastic context-free grammars using the inside-outside algorithm, Computer Speech & Language 4 (1990) 35–56.
[11] Z. Li, J. Eisner, First- and second-order expectation semirings with applications to minimum-risk training on translation forests, In: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1, EMNLP ’09, pp. 40–51, Morristown, NJ, USA, 2009. Association for Computational Linguistics.
[12] M.-J. Nederhof, G. Satta, Kullback-leibler distance between probabilistic context-free grammars and probabilistic finite automata, In: Proceedings of the 20th international conference on Computational Linguistics, COLING ’04, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics.
[13] M.-J. Nederhof, G. Satta, Computing partition functions of pcfgs, Research on Language and Computation, 6 (2008) 139–162.
[14] M.-J. Nederhof, G. Satta, Probabilistic parsing, In: New Developments in Formal Languages and Applications, volume 113 of Studies in Computational Intelligence, pp. 229–258, Springer Berlin / Heidelberg, 2008.
[15] X. S. Raymond, Elementary Introduction to the Theory of Pseudodifferential Operators, Studies in Advanced Mathematics, CRC Press, Boca Raton, 1991.
[16] R. B. Lund, Elementary probability theory with stochastic processes and an introduction to mathematical finance, The American Statistician 58 (2004) 173–174.
[17] F. Tendeu, Computing abstract decorations of parse forests using dynamic programming and algebraic power series, Theoretical Computer Science 199 (1998) 145–166.
[18] A. B. Thaheema, A. Laradji, Classroom note: A generalization of Leibniz rule for higher derivatives, International Journal of Mathematical Education in Science and Technology 34 (2001) 905–907.
[19] C. S. Wetherell, Probabilistic languages: A review and some open questions, ACM Computing Surveys 12 (1980) 361–379.