TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Informatik

Technische Berichte

Technical Reports

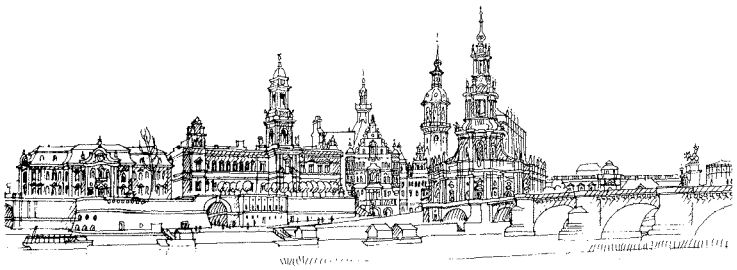
ISSN 1430-211X

TUD-FI12-04-März 2012

Torsten Stüber

Faculty of Computer Science, Technische Universität Dresden

Consistency of Probabilistic Context-Free Grammars



Technische Universität Dresden
Fakultät Informatik
D-01062 Dresden
Germany

URL: http://www.inf.tu-dresden.de/

Consistency of Probabilistic Context-Free Grammars

Torsten Stüber

Faculty of Computer Science, Technische Universität Dresden

D-01062 Dresden, Germany

torsten.stueber@tu-dresden.de

March 13, 2012

We present an algorithm for deciding whether an arbitrary proper probabilistic context-free grammar is consistent, i.e., whether the probability that a derivation terminates is one. Our procedure has time complexity $\mathcal{O}(n^3)$ in the unit-cost model of computation. Moreover, we develop a novel characterization of consistent probabilistic context-free grammars. A simple corollary of our result is that training methods for probabilistic context-free grammars that are based on maximum-likelihood estimation always yield consistent grammars.

1 Introduction

Probabilistic grammars – also called stochastic grammars – are a class of statistical models widely used in, e.g., natural-language processing (Manning and Schütze, 1999) and for the description of the secondary structure of RNA (Sakakibara et al., 1994; Durbin et al., 1998). Important representatives in this class are, e.g., probabilistic context-free grammars (for short: pcfg) (Booth, 1969; Booth and Thompson, 1973), weighted regular tree grammars (Alexandrakis and Bozapalidis, 1987; Gécseg and Steinby, 1984), synchronous context-free grammars (Lewis and Stearns, 1968; Chiang, 2005), and (synchronous) tree-adjoining grammars (Joshi et al., 1975; Shieber and Schabes, 1990).

Probabilistic grammars are obtained from their non-probabilistic counterparts by associating every grammar rule with a probability. Usually one requires that the probabilities of all rules having the same left-hand side sum up to one – in this case we say that the grammar is *proper*. However, this condition does not guarantee that the probabilistic grammar generates a probabilistic language; more precisely, the sum of the probabilities of all leftmost derivations of the grammar can be less than one. If this sum happens to be one, then we say that the grammar is *consistent*. For example, the context-free grammar having the two rules

$$S \to SS\,, \qquad\qquad S \to a\,,$$

with probabilities p and $1 - p$, respectively, is not consistent if $p > 0.5$; cf. Nederhof and Satta (2008). The reason for a grammar to be inconsistent is that the derivation trees can

have a nonzero probability of never terminating, i.e., some probability mass is lost to infinite derivations. A basic computational problem of probabilistic grammars is to decide whether a given proper grammar is also consistent.

This problem has first been investigated in the seminal paper written by Booth and Thompson (1973). Theorem 2 of their paper states that a proper pcfg is consistent if $\rho(M_{\text{fm}}) < 1$ and it is not consistent if $\rho(M_{\text{fm}}) > 1$, where M_{fm} is the first-moment matrix of the grammar and $\rho(M_{\text{fm}})$ is the spectral radius of M_{fm}, i.e., the magnitude of its largest eigenvalue. The proof of Theorem 2 is based on classical results on branching processes; cf. Harris (1963). However, the classical result by Booth and Thompson, which is widely cited in the literature, is unsatisfactory for three reasons: (i) in general the result is wrong if the grammar is not reduced, i.e., if it contains nonterminals that are unreachable from the initial nonterminal. This suggests that the authors make the assumption that the grammar is reduced but do not mention this. (ii) The result is inconclusive if $\rho(M_{\text{fm}}) = 1$. In fact, Booth and Thompson omit this situation because they argue that many special cases have to be considered and instead refer the reader to the discussion in Harris (1963), albeit not even Harris (1963) gives a complete characterization and proof for this special situation. For a more thorough discussion on this issue see Etessami and Yannakakis (2009, Section 8). (iii) Booth and Thompson provide no algorithm for deciding whether $\rho(M_{\text{fm}}) < 1$ or $\rho(M_{\text{fm}}) > 1$ and, thus, no algorithm for deciding whether a given proper pcfg is consistent. Note that, however, there are simple methods for *approximating* the spectral radius of M_{fm}.

Wetherell (1980) repeats the result given in Booth and Thompson (1973); moreover, Wetherell gives an algorithm for semideciding whether $\rho(M_{\text{fm}}) < 1$. According to Etessami and Yannakakis (2009), more recent papers that deal with the related concept of consistency of multitype branching processes typically merely restate the results in Harris (1963).

The three issues of the result in Booth and Thompson (1973) have been solved recently by Etessami and Yannakakis (2009). In particular, they address the special case $\rho(M_{\text{fm}}) = 1$ and show that such a grammar is always consistent whenever it is reduced; moreover, they demonstrate that the problem to decide whether a proper pcfg is consistent can be solved in polynomial time by reducing this problem to linear programming (Matoušek and Gärtner, 2007). The proof of this result is structurally similar to the proof in Harris (1963) and makes use of Perron-Frobenius theory of matrices (Lancaster and Tismenetsky, 1985; Horn and Johnson, 1990).

In this paper we provide an alternative complete procedure for deciding whether a proper pcfg is consistent. Our proof is considerably different from the proofs in the literature (Booth and Thompson, 1973; Etessami and Yannakakis, 2009); it is not based on Perron-Frobenius theory and is simpler than the proof in Etessami and Yannakakis (2009). The algorithm that we provide does not make use of linear programming. It has time complexity $\mathcal{O}(n^3)$ in the unit-cost model of computation[1] and, therefore, is faster than the linear programming approach by Etessami and Yannakakis in terms of this cost model; for a discussion on the complexity of linear programming algorithms we refer the reader to, e.g., Matoušek and Gärtner (2007); Gass (2010). Our main result is stated in Theorem 2; it is based on Theorem 5, which gives a characterization of proper, restricted, and consistent pcfg. The main part of our decision procedure is shown in Figure 4.

In many applications the probabilities of a probabilistic grammar are not generated manually but by an automatic training procedure. In the field of natural-language processing one

[1] i.e., every arithmetic operation takes constant time independent of the size of the operands.

usually employs training procedures that are based on maximum-likelihood estimation (Dempster et al., 1977; Prescher, 2001; Graehl et al., 2008). It is well-known that for pcfg such a training procedure always yields a consistent grammar (Chaudhuri et al., 1983; Sánchez and Benedí, 1997; Chi and Geman, 1998). In Theorem 26 we repeat this result and show that it follows as a simple corollary from Theorem 5.

This paper is organized as follows. In Section 2 we recall the concept of probabilistic context-free grammars and state the main result of this paper (Theorems 2 and 5, and the algorithm shown in Figure 4). We will prove these results in the subsequent three sections. In Section 3 we discuss the notion of reduced grammars and show that we can restrict ourselves to such grammars. Section 4 is concerned with the main part of the proof of Theorem 5. In Section 5 we will demonstrate the correctness of the algorithm in Figure 4. In Section 6 we will deal with training of grammars using the maximum-likelihood method and we will prove Theorem 26.

2 Probabilistic Context-Free Grammars

In this section we recall the definition of probabilistic context-free grammars (Booth, 1969; Booth and Thompson, 1973) and state our main theorems (see Theorems 2 and 5).

Let Σ be an alphabet. A *probabilistic context-free grammar* (for short: *pcfg*) over Σ is a tuple $G = (N, S, R, \kappa)$ where

- N is a finite set (of *nonterminals*) disjoint from Σ,
- $S \in N$,
- R is a finite set of *rules* of the form $A \to w$ with $A \in N$ and $w \in (\Sigma \cup N)^*$, and
- $\kappa : R \to [0, 1]$.

For every $A \in N$ we denote by R_A the set of rules whose left-hand side is A. We say that G is *proper* if $\sum_{r \in R_A} \kappa(r) = 1$ for every $A \in N$. For the remainder of this paper we fix an alphabet Σ and a pcfg (N, S, R, κ) over Σ.

Example 1. Let $\Sigma = \{a, b\}$ and consider the pcfg $G = (N, A, R, \kappa)$ where $N = \{A, B, C, D, E\}$ and R contains the rules:

$$
\begin{array}{lll}
A \xrightarrow{1/2} A^2C^4aDE^4\,, & A \xrightarrow{1/2} a\,, & \\
B \xrightarrow{1/4} bB\,, & B \xrightarrow{1/2} C^2E^2\,, & B \xrightarrow{1/4} ab\,, \\
C \xrightarrow{1/2} E^4\,, & C \xrightarrow{1/2} bb\,, & \\
D \xrightarrow{1/4} bA\,, & D \xrightarrow{1/2} C^4bD^2E^2\,, & D \xrightarrow{1/4} aa\,, \\
E \xrightarrow{1/4} Bab\,, & E \xrightarrow{3/4} ba\,. &
\end{array}
$$

A^2 is the abbreviation for AA. The value $\kappa(r)$ is written on top of the arrow of the rule r. Observe that G is proper. □

In order to simplify notation, we adopt the following convention. Given $r \in R$, whenever we write $r = A \to w_0A_1w_1 \cdots w_{n-1}A_nw_n$ we indicate that the left-hand side of r is A and that the

right-hand side contains precisely the nonterminals $A_1, \ldots, A_n$ in this order; i.e., $w_0, \ldots, w_n$ are terminal words. However, whenever we write $r = A \to wBw'$, we merely state that the left-hand side is A and that the nonterminal B *occurs* on the right-hand side; i.e., w and w' may contain terminals as well as nonterminals.

Now we recall the concept of derivations of context-free grammars. Since we consider probabilistic grammars, we have to ensure that we do not count derivations multiply that merely result from expanding nonterminals in different orders. One way to achieve this is to restrict ourselves to leftmost derivations. Here we adopt the more elegant approach to represent derivations as syntax trees, thereby we avoid the need to choose a representative in a class of corresponding derivations. For every $A \in N$ the set T_A of *syntax trees* for A is the set

$$T_A = \big\{ r(t_1, \ldots, t_n) \mid r = A \to w_0 A_1 w_1 \cdots w_{n-1} A_n w_n \in R, t_1 \in T_{A_1}, \ldots, t_n \in T_{A_n} \big\} \,. \tag{1}$$

For every $t \in T_A$ we define the *probability* $P_\kappa(t) \in [0,1]$ of t and the *yield* $\mathrm{yield}(t) \in \Sigma^*$ of t by recursion:

$$\begin{aligned} P_\kappa(t) &= \kappa(r) \cdot P_\kappa(t_1) \cdots P_\kappa(t_n) \,, \\ \mathrm{yield}(t) &= w_0 \mathrm{yield}(t_1) w_1 \cdots w_{n-1} \mathrm{yield}(t_n) w_n \,. \end{aligned} \tag{2}$$

where $t = r(t_1, \ldots, t_n)$ and $r = A \to w_0 A_1 w_1 \cdots w_{n-1} A_n w_n$. We drop κ from P_κ if it is clear from the context. Given $w \in \Sigma^*$, the *probability* $P(w)$ of w is

$$P(w) = \sum \big(P(t) \mid t \in T_S, \mathrm{yield}(t) = w \big) \,.$$

We say that the grammar G is *consistent* if $\sum_{w \in \Sigma^*} P(w) = 1$; otherwise we say that G is *inconsistent*. Note that $\sum_{w \in \Sigma^*} P(w) = \sum \big(P(t) \mid t \in T_S \big)$. Now we present our main theorem.

Theorem 2. *There is an $\mathcal{O}(n^3)$ unit-cost algorithm that decides whether a given proper pcfg is consistent.*

The proof of this theorem is given at the end of Section 5.

We conclude this section by presenting Theorem 5, the second main result of this paper. To this end we first introduce some auxiliary definitions. Let $A \in N$. We call A *productive* if there is a $t \in T_A$ with $P(t) > 0$. Moreover, A is *reachable* if there is a sequence $A_1, \ldots, A_n \in N$ with $A_1 = S$, $A_n = A$, and for every $1 \leq i < n$ there is a rule $r = A_i \to w A_{i+1} w'$ with $\kappa(r) > 0$. We say that G is *reduced* if every $A \in N$ is productive and reachable. Observe that the grammar in Example 1 is reduced: every nonterminal can be reached from the start symbol A and every nonterminal is productive; in fact, every nonterminal can be transformed to a terminal string in one derivation step.

By $\mathbf{0}$ and $\mathbf{1}$ we denote the vectors with $\mathbf{0}(A) = 0$ and $\mathbf{1}(A) = 1$ for every $A \in N$. The definition of the following mapping is inspired by Booth and Thompson (1973, page 445). Define $f : \mathbb{R}^N \to \mathbb{R}^N$ as follows for every vector $v \in \mathbb{R}^N$ and $A \in N$:

$$f(v)(A) = \sum \Big(\kappa(r) \cdot \textstyle\prod_{i=1}^{n} v(A_i) \mid r = A \to w_0 A_1 w_1 \cdots w_{n-1} A_n w_n \Big) \,.$$

The significance of f is that its least nonnegative fixed point are the inside probabilities of the grammar (see Lemma 7). Moreover, let

$$\mathrm{Pre} = \big\{ v \in [0,1]^N \mid \forall\, A \in N : f(v)(A) \leq v(A) \big\} \,,$$

$$\text{StrictPre} = \left\{ v \in \text{Pre} \setminus \{\mathbf{1}\} \mid \forall\, A \in N : v(A) = 1 \text{ or } f(v)(A) < v(A) \right\} .$$

Roughly speaking, the set Pre is the set of prefixed points of f and the set StrictPre contains the strict prefixed points of f.

Example 3 (Cont. of Example 1). Observe that

$$\begin{aligned}
f(v)(A) &= \frac{v(A)^2 \cdot v(C)^4 \cdot v(D) \cdot v(E)^4}{2} + \frac{1}{2} , \\
f(v)(B) &= \frac{v(B)}{4} + \frac{v(C)^2 \cdot v(E)^2}{2} + \frac{1}{4} , \\
f(v)(C) &= \frac{v(E)^4}{2} + \frac{1}{2} , \\
f(v)(D) &= \frac{v(A)}{4} + \frac{v(C)^4 \cdot v(D)^2 \cdot v(E)^2}{2} + \frac{1}{4} , \\
f(v)(E) &= \frac{v(B)}{4} + \frac{3}{4} .
\end{aligned}$$

Note that $\mathbf{1}$ is a fixed point of f and, thus, $\mathbf{1} \in \text{Pre}$. However, Pre contains other points; e.g., for $v \in [0,1]^N$ with

$$v(A) = 0.8 , \qquad v(B) = 1 , \qquad v(C) = 1 , \qquad v(D) = 0.8 , \qquad v(E) = 1 ,$$

we obtain that

$$f(v)(A) = 0.756 , \quad f(v)(B) = 1 , \quad f(v)(C) = 1 , \quad f(v)(D) = 0.77 , \quad f(v)(E) = 1 .$$

Hence, $v \in \text{Pre}$ and even $v \in \text{StrictPre}$. □

We define two matrices $M_{\text{fm}}, M \in \mathbb{R}^{N \times N}$ by letting for every $A, B \in N$:

$$\begin{aligned}
M_{\text{fm}}(A,B) &= \sum\nolimits_{r\,=\,A \to w} \kappa(r) \cdot |w|_B , \\
M &= M_{\text{fm}} - E ,
\end{aligned}$$

where $|w|_B$ is the number of occurrences of B in w and E is the unit matrix. The matrix M_{fm} is called the *first-moment matrix associated with* G in Booth and Thompson (1973). By $\delta(M_{\text{fm}})$ we denote the spectral radius of M_{fm}, i.e., the maximal absolute value of the eigenvalues of M_{fm}.

Example 4 (Cont. of Example 3). For our example grammar we have

$$M_{\text{fm}} = \begin{pmatrix} 1 & 0 & 2 & 1/2 & 2 \\ 0 & 1/4 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 2 \\ 1/4 & 0 & 2 & 1 & 1 \\ 0 & 1/4 & 0 & 0 & 0 \end{pmatrix} , \qquad M = \begin{pmatrix} 0 & 0 & 2 & 1/2 & 2 \\ 0 & -3/4 & 1 & 0 & 1 \\ 0 & 0 & -1 & 0 & 2 \\ 1/4 & 0 & 2 & 0 & 1 \\ 0 & 1/4 & 0 & 0 & -1 \end{pmatrix} ,$$

where the first row/column stands for nonterminal A and so on. □

Now we state the main characterization result for proper, reduced, and consistent pcfgs; it serves as the central tool for proving Theorem 2. Let $\mathbb{R}_{\geq 0}$ denote the set of nonnegative reals.

Theorem 5. *Let G be proper and reduced. Then the following statements are equivalent:*

1. *G is consistent.*
2. $\mathrm{Pre} = \{\mathbf{1}\}$.
3. $\mathrm{StrictPre} = \emptyset$.
4. *For every $v \in \mathbb{R}^N_{\geq 0} \setminus \{\mathbf{0}\}$ there is an $A \in N$ such that $v(A) > 0$ and $(Mv)(A) \leq 0$.*
5. $\rho(M_{\mathrm{fm}}) \leq 1$.
6. *The algorithm in Figure 4 returns "does not have Property (B)" upon input M.*

Note that the equivalence of the first and the fifth statement of Theorem 5 is essentially Lemma 8.2 and Lemma 8.4 in Etessami and Yannakakis (2009) and, therefore, the main statement underlying Theorem 8.1 of that paper. We included this equivalence only for completeness, it is not required for proving Theorem 2. The algorithm that is referenced in the sixth statement of Theorem 5 is shown on page 15.

We remark that the matrix M is of the type $\mathbb{R}^{N\times N}$; however, the algorithm in Figure 4 requires a matrix of the form $\mathbb{R}^{n\times n}$. We do not distinguish between these types because we can consider the set N to be of the form $\{A_1, \ldots, A_n\}$ for some n; then there is an obvious one-to-one correspondence between the sets $\mathbb{R}^{N\times N}$ and $\mathbb{R}^{n\times n}$.

As a first application of Theorem 5 observe that the example grammar given in Example 1 is inconsistent because $\mathrm{StrictPre} \neq \emptyset$ as demonstrated in Example 3.

We will break up the proof of Theorem 5 into small steps. It follows from Lemmas 12, 14, 15, 17, and 24.

3 Inside Probabilities and Reduced Grammars

As a first step toward a proof of our main result, we recall the useful concept of inside probabilities. In the literature the collection of inside probabilities is also called the partition function of the grammar, e.g., in Nederhof and Satta (2008). For $A \in N$, we define the *inside probability* $\beta(A)$ of A as follows:

$$\beta(A) = \sum \big(P(t) \mid t \in T_A\big) \; . \tag{3}$$

By definition, G is consistent iff $\beta(S) = 1$. We derive

$$\begin{aligned}
\beta(A) &= \sum \big(P(r(t_1, \ldots, t_n)) \mid r \;=\; A \to w_0 A_1 w_1 \cdots w_{n-1} A_n w_n \in R, t_i \in T_{A_i}\big) && \text{(by (1))} \\
&= \sum \big(\kappa(r) \cdot P(t_1) \cdots P(t_n) \mid r \;=\; A \to w_0 A_1 w_1 \cdots w_{n-1} A_n w_n \in R, t_i \in T_{A_i}\big) && \text{(by (2))} \\
&= \sum \Big(\kappa(r) \cdot \textstyle\prod_{i=1}^{n} \sum \big(P(t_i) \mid t_i \in T_{A_i}\big) \;\Big|\; r \;=\; A \to w_0 A_1 w_1 \cdots w_{n-1} A_n w_n\Big) \\
&= \sum \Big(\kappa(r) \cdot \textstyle\prod_{i=1}^{n} \beta(A_i) \;\Big|\; r \;=\; A \to w_0 A_1 w_1 \cdots w_{n-1} A_n w_n\Big) && (4) \\
&= f(\beta)(A) \; .
\end{aligned}$$

Hence, the vector $\beta \in \mathbb{R}^N$ is a fixed point of the mapping f, i.e., $f(\beta) = \beta$. Now we face the question whether f has other fixed points and whether β is outstanding amongst all fixed points of f. The following observation follows immediately from the definition of properness and the definition of f.

Observation 6. *If G is proper, then the vector $\mathbf{1}$ is a fixed point of f.*

It is easy to construct a counterexample that demonstrates that $\mathbf{1}$ is not necessarily a fixed point of f if G is not proper.

We use $\leq$ to denote the usual component-wise ordering of vectors in $\mathbb{R}^N$, i.e., $v_1 \leq v_2$ iff $v_1(A) \leq v_2(A)$ for every $A \in N$. Note that due to the fact that f is a polynomial with nonnegative coefficients, f is monotone for nonnegative vectors; more formally, for $v, v' \in \mathbb{R}^N_{\geq 0}$ with $v \leq v'$ we obtain $f(v) \leq f(v')$. The following lemma is well known – see, e.g., Nederhof and Satta (2008).

Lemma 7. *The vector β is the least fixed point of f in $\mathbb{R}^N_{\geq 0}$ with respect to $\leq$. More formally, β is the least element in $\{v \in \mathbb{R}^N_{\geq 0} \mid f(v) = v\}$ with respect to $\leq$.*

The following corollary is a consequence of Observation 6 and Lemma 7.

Corollary 8. *If G is proper, then $\mathbf{0} \leq \beta \leq \mathbf{1}$, i.e., $\beta \in [0,1]^N$.*

Theorem 2 states that there is an algorithm that decides consistency for arbitrary proper pcfgs. However, Theorem 5 is only applicable for reduced proper pcfgs. It turns out that the characterization result in Theorem 5 is sufficient for proving Theorem 2. More precisely, we show that every proper pcfg is either inconsistent for obvious reasons or there is an equivalent reduced pcfg that is consistent iff the original pcfg is consistent (Lemma 11).

Lemma 9. *Suppose that G is proper, A is reachable, and $\beta(A) < 1$. Then $\beta(S) < 1$.*

Proof. From Equation (4) and Corollary 8 we conclude that, for every $B, C \in N$, if there is a rule $r = B \to wCw'$ with $\kappa(r) > 0$, then $\beta(C) < 1$ implies $\beta(B) < 1$. Hence, it is easy to show by induction that the facts that A is reachable and $\beta(A) < 1$ imply $\beta(S) < 1$. ∎

For more background on the following observation we refer the reader to Sipser (1996).

Observation 10. *The set of reachable nonterminals and the set of productive nonterminals of G can be constructed in time $\mathcal{O}(\text{size}(R))$, where $\text{size}(R) = \sum_{A \to w \in R} |w|$.*

Lemma 11. *Suppose that G is proper. If there is a reachable non-productive nonterminal, then G is inconsistent.*

Otherwise, let $G' = (N', S, R', \kappa|_{R'})$ where N' and R' are obtained from N and R, respectively, by removing all non-reachable nonterminals and all rules containing non-reachable nonterminals, respectively. Then G' is reduced and G' is inconsistent iff G is inconsistent.

Proof. Assume that there is a reachable non-productive nonterminal A. The definition of β yields $\beta(A) = 0$; thus, G is inconsistent due to Lemma 9.

If every non-productive nonterminal is not reachable, then G' is obviously reduced and equivalent to G, i.e., $P_G(w) = P_{G'}(w)$ for every $w \in \Sigma^*$, where $P_G(w)$ and $P_{G'}(w)$ are the probabilities of w in G and G', respectively. Thus, G' is consistent iff G is consistent. ∎

4 Characterization of Consistent PCFG

In this section we prove the equivalence of the first five statements of Theorem 5. Let us begin with exposing the equivalence of the first two statements. Note that Pre is the set of vectors v in $[0,1]^N$ such that $f(v) \leq v$.

Lemma 12. *Let G be proper and reduced. Then G is consistent iff* $\mathrm{Pre} = \{\mathbf{1}\}$.

Proof. Let $\perp \in [0,1]^N$ be defined by $\perp(A) = \inf\{v(A) \mid v \in \mathrm{Pre}\}$ for every $A \in N$; this is defined because Pre only contains nonnegative vectors and it is nonempty: in fact, it contains $\mathbf{1}$ because $\mathbf{1}$ is a fixed point of f due to Observation 6. Clearly, $\perp$ is the infimum of Pre with respect to $\leq$.

By an argument similar to the fixed point theorem of Tarski (1955, Theorem 1), we show that $\perp = \beta$. For every $v \in \mathrm{Pre}$, $\perp \leq v$ and, hence, $f(\perp) \leq f(v) \leq v$ by the monotonicity of f. Thus, $f(\perp)$ is a lower bound of Pre and therefore $f(\perp) \leq \perp$, which implies $\perp \in \mathrm{Pre}$. Since, $f(\perp) \leq \perp$, also $f(f(\perp)) \leq f(\perp)$, i.e., $f(\perp) \in \mathrm{Pre}$ and therefore $\perp \leq f(\perp)$. We conclude that $f(\perp) = \perp$, i.e., that $\perp$ is a fixed point of f. Hence, $\beta \leq \perp$ by Lemma 7. Moreover, $\perp \leq \beta$ because $\beta \in [0,1]^N$ by Corollary 8 and, thus, $\beta \in \mathrm{Pre}$.

Now Lemma 9 and the fact that G is reduced imply: G is consistent iff $\beta(A) = 1$ for every $A \in N$ iff $\beta = \mathbf{1}$ iff $\perp = \mathbf{1}$ iff $\mathrm{Pre} = \{\mathbf{1}\}$. ∎

Next let us deal with the equivalence of the second and third statement of Theorem 5. Although both statements look similar, their equivalence is one of the core statements of Theorem 5 and its proof is fairly involved. Speaking in terms of topology, Pre is a closed set and StrictPre the open set of the interior points of Pre – see Figure 1. Roughly speaking, the equivalence of the second and third statement of Theorem 5 means that Pre has an interior if $\mathrm{Pre} \neq \{\mathbf{1}\}$. However, there may be degenerate cases, where Pre has only boundary points in spite of $\mathrm{Pre} \neq \{\mathbf{1}\}$ (e.g., Pre is a line segment). Therefore, we included these special boundary points in the definition of StrictPre by requiring "$v(A) = 1$ or $f(v)(A) < v(A)$" instead of solely "$f(v)(A) < v(A)$".

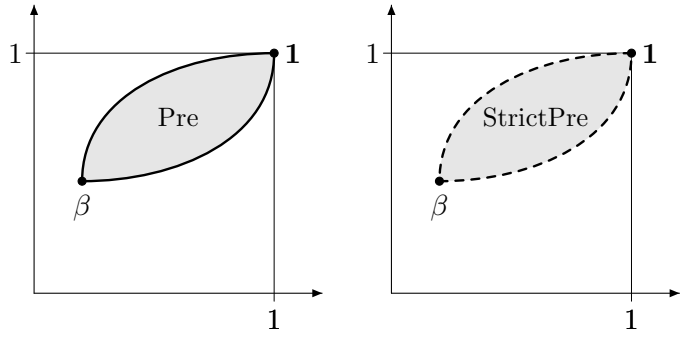


Figure 1: Comparison of the sets Pre and StrictPre.

We first prove an auxiliary statement. For every $A \in N$ we define $f_A : \mathbb{R}^N \to \mathbb{R}$ and $g_A : \mathbb{R}^N \to \mathbb{R}$ by $f_A(v) = f(v)(A)$ and $g_A(v) = f_A(v) - v(A)$. The following statement concerns the values of g_A for the points on the line segment between two vectors v_1 and v_2 in $[0,1]^N$ with $v_1 \leq v_2$ (see Figure 2 for the case $|N| = 2$). It turns out that all these values are nonpositive if g_A is nonpositive at both v_1 and v_2; moreover, they are negative if g_A is negative at v_1 or v_2.

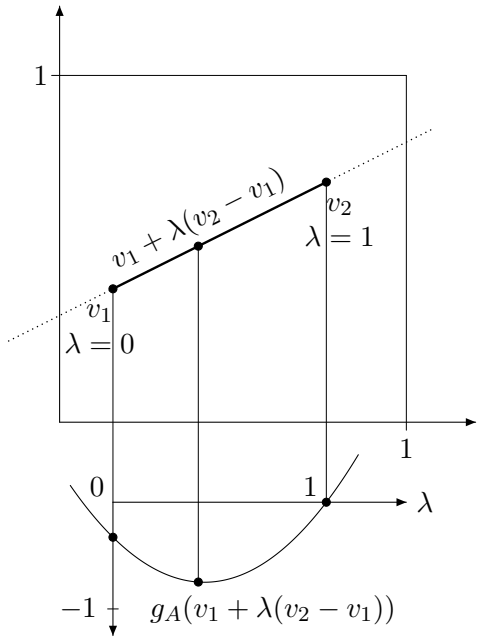


Figure 2: Illustration of Lemma 13.

Lemma 13. *Let $A \in N$ and $v_1, v_2 \in [0,1]^N$ such that $v_1 \leq v_2$ and $g_A(v_1), g_A(v_2) \leq 0$. Moreover, let $h : \mathbb{R} \to \mathbb{R}^N : \lambda \mapsto v_1 + (v_2 - v_1)\lambda$.*

1. *$g_A(h(\lambda)) \leq 0$ for every scalar $\lambda \in [0,1]$.*
2. *If $g_A(v_1) < 0$ or $g_A(v_2) < 0$, then $g_A(h(\lambda)) < 0$ for every scalar $\lambda \in]0,1[$.*
3. *Suppose $g_A(v_2) = 0$. Then $(g_A \circ h)'(1) \geq 0$ and if $g_A(v_1) < 0$, then $(g_A \circ h)'(1) > 0$.*[2]
4. *Let $N' = \{B \in N \mid v_1(B) \neq v_2(B)\}$. If $g_A(h(\lambda)) = 0$ for some $\lambda \in]0,1[$, then*
   - *for every $r = A \to w_0 A_1 w_1 \cdots w_{n-1} A_n w_n$ there is at most one $i \in \{1, \ldots, n\}$ with $A_i \in N'$, and*
   - *$\sum_{B \in N'} (v_2(B) - v_1(B)) d_B \geq v_2(A) - v_1(A)$, where $d_B = \sum_{r = A \to wBw'} \kappa(r)$.*

Proof. First we derive two auxiliary statements that will be useful throughout this proof. Observe that for every $\lambda \in \mathbb{R}$:

$$\begin{aligned} (g_A \circ h)(\lambda) &= (f_A \circ h)(\lambda) - v_1(A) - \big(v_2(A) - v_1(A)\big)\lambda \,, \\ (g_A \circ h)'(\lambda) &= (f_A \circ h)'(\lambda) - \big(v_2(A) - v_1(A)\big) \,, \\ (g_A \circ h)''(\lambda) &= (f_A \circ h)''(\lambda) \,. \end{aligned} \tag{5}$$

Note that $f_A \circ h$ is a polynomial with nonnegative coefficients because f_A is a polynomial in N with nonnegative coefficients and $v_2(B) - v_1(B) \geq 0$ for every $B \in N$ due to the condition that $v_1 \leq v_2$. Thus, $(g_A \circ h)''(\lambda) = (f_A \circ h)''(\lambda) \geq 0$ for every $\lambda \geq 0$. By applying the mean-value

[2] $(g_a \circ h)'$ denotes the first derivative of $g_a \circ h$.

theorem we obtain that for every $\lambda_1, \lambda_2, \lambda_3$ with $0 \leq \lambda_1 < \lambda_2 < \lambda_3$:

$$\begin{aligned}&(g_A \circ h)(\lambda_1) < (g_A \circ h)(\lambda_2) \text{ implies } (g_A \circ h)(\lambda_2) < (g_A \circ h)(\lambda_3) \text{ and}\\ &(g_A \circ h)(\lambda_1) = (g_A \circ h)(\lambda_2) \text{ implies } (g_A \circ h)(\lambda_2) \leq (g_A \circ h)(\lambda_3)\ .\end{aligned} \tag{6}$$

The first two statements of this lemma immediately follow from (6) and the facts that $g_A(v_1) = (g_A \circ h)(0)$ and $g_A(v_2) = (g_A \circ h)(1)$. Likewise, the mean-value theorem implies the third statement.

We are left with the task to prove the fourth statement of this lemma. Let us suppose that $(g_A \circ h)(\lambda) = 0$ for some $\lambda \in]0,1[$. Therefore the second statement implies $(g_A \circ h)(0) = (g_A \circ h)(1) = 0$. Observe that then (6) yields $(g_A \circ h)(\lambda) = 0$ for every $\lambda \in [0,1]$. By Equation (5), every $\lambda \in [0,1]$ satisfies $v_1(A) + \big(v_2(A) - v_1(A)\big)\lambda = (f_A \circ h)(\lambda)$. Thus, $f_A \circ h$ is a linear polynomial in λ. Since

$$(f_A \circ h)(\lambda) = \sum\nolimits_{r\,=\,A \to w_0A_1w_1\cdots w_{n-1}A_nw_n} \Big(\kappa(r) \cdot \prod\nolimits_{i=1}^{n} v_1(A_i) + \big(v_2(A_i) - v_1(A_i)\big)\lambda\Big), \tag{7}$$

we conclude that, for every $r = A \to w_0A_1w_1 \cdots w_{n-1}A_nw_n$, there is at most one $i \in \{1, \ldots, n\}$ with $v_1(A_i) \neq v_2(A_i)$; let us write $N(r) = A_i$ if there is such a nonterminal A_i and let us write $N(r) = \emptyset$ if there is no such A_i (i.e., $v_1(A_i) = v_2(A_i)$ for every $i \in \{1, \ldots, n\}$). This yields the first assertion of the fourth statement of this lemma.

Now define the constant $c_r = \prod\big(v_1(A_j) \mid 1 \leq j \leq n, v_1(A_j) = v_2(A_j)\big)$ for every rule $r = A \to w_0A_1w_1 \cdots w_{n-1}A_nw_n$. Clearly, $c_r \leq 1$ because $v_1 \in [0,1]^N$. Using this definition we obtain that if $N(r) = \emptyset$, then

$$\prod\nolimits_{j=1}^{n} v_1(A_j) + \big(v_2(A_j) - v_1(A_j)\big)\lambda = c_r\ ,$$

and if $N(r) = A_i$, then

$$\begin{aligned}\prod\nolimits_{j=1}^{n} v_1(A_j) + \big(v_2(A_j) - v_1(A_j)\big)\lambda &= \big(v_1(A_i) + (v_2(A_i) - v_1(A_i))\lambda\big)c_r\\ &= v_1(A_i)c_r + \big(v_2(A_i) - v_1(A_i)\big)c_r\lambda\ .\end{aligned}$$

Using the fact $v_1(A) + \big(v_2(A) - v_1(A)\big)\lambda = (f_A \circ h)(\lambda)$, we can reformulate (7) as

$$\begin{aligned}&v_1(A) + \big(v_2(A) - v_1(A)\big)\lambda\\ &= \sum\nolimits_{\substack{r \in R_A\\ N(r)=\emptyset}} \kappa(r)c_r + \sum\nolimits_{B \in N'} \sum\nolimits_{\substack{r \in R_A\\ N(r)=B}} \kappa(r)v_1(B)c_r + \kappa(r)\big(v_2(B) - v_1(B)\big)c_r\lambda\ .\end{aligned}$$

Since both sides of the equation are the same linear function, we conclude that the first-order coefficients are equal, i.e.,

$$\begin{aligned}v_2(A) - v_1(A) &= \sum\nolimits_{B \in N'} \sum\nolimits_{\substack{r \in R_A\\ N(r)=B}} \kappa(r)\big(v_2(B) - v_1(B)\big)c_r\\ &\leq \sum\nolimits_{B \in N'} \big(v_2(B) - v_1(B)\big) \sum\nolimits_{\substack{r \in R_A\\ N(r)=B}} \kappa(r)\ ,\end{aligned}$$

because $c_r \leq 1$. The fact that $\sum_{\substack{r \in R_A\\ N(r)=B}} \kappa(r) = d_B$ yields the second assertion of the fourth statement of this lemma. ∎

Now we can prove the equivalence of the second and third statement of Theorem 5.

Lemma 14. *Let G be proper and reduced. Then* $\mathrm{Pre} = \{\mathbf{1}\}$ *iff* $\mathrm{StrictPre} = \emptyset$.

Proof. The only-if-part is trivial. It remains to prove the if-part. To this end let $S = \mathrm{Pre}\setminus\{\mathbf{1}\}$ and put $N(v) = \{A \in N \mid v(A) \neq 1 \text{ and } f_A(v) \not\leq v(A)\}$ for every $v \in S$. Clearly, $\mathrm{StrictPre} = \{v \in S \mid N(v) = \emptyset\}$. This proof is based on the following claim:

$$\text{For every } v \in S \text{ with } N(v) \neq \emptyset \text{ there is some } v' \in S \text{ with } N(v') \subsetneq N(v). \tag{Claim}$$

This claim implies our assertion as the following argument demonstrates. Assume that $\mathrm{Pre} \neq \{\mathbf{1}\}$. Then $S \neq \emptyset$ because $\mathbf{1} \in \mathrm{Pre}$ by Observation 6. By iterating the claim a finite number of times we find a $v \in S$ with $N(v) = \emptyset$. Hence, $\mathrm{StrictPre} \neq \emptyset$.

It remains to prove the claim. Let $v \in S$ with $N(v) \neq \emptyset$. Define $v_0 \in [0,1]^N$ by letting for every $A \in N$:

$$v_0(A) = \begin{cases} 1\,, & \text{if } A \in N(v), \\ v(A)\,, & \text{otherwise.} \end{cases}$$

Observe that, for every $A \in N$, the fact that $v_0 \leq \mathbf{1}$ yields $f_A(v_0) \leq f_A(\mathbf{1}) = 1$; hence,

$$\text{if } A \in N(v) \text{ or } v(A) = 1, \text{ then } g_A(v_0) = f_A(v_0) - v_0(A) = f_A(v_0) - 1 \leq 0\,. \tag{8}$$

For every $A \in N$ with $f_A(v) < v(A)$ we obtain $g_A(v) < 0$; since g_A is continuous, there is an $\varepsilon > 0$ such that $g_A(v') < 0$ for every v' with $\|v' - v\| < \varepsilon$. Thus, there is a $\lambda \in]0,1[$ such that $g_A\big(v + (v_0 - v)\lambda\big) < 0$ for every $A \in N$ with $f_A(v) < v(A)$. We put $v' = v + (v_0 - v)\lambda$.

It remains to show that $v' \in S$ and $N(v') \subsetneq N(v)$. Let $A \in N$. We make four observations.

(a) If $f_A(v) < v(A)$, then the definition of v' yields $g_A(v') < 0$ and, hence, $f_A(v') < v'(A)$.

(b) If $A \in N(v)$ or $v(A) = 1$, then $g_A(v_0) \leq 0$ by (8); since $v \in S \subseteq \mathrm{Pre}$, also $f_A(v) \leq v(A)$ or, equivalently, $g_A(v) \leq 0$; clearly, $v \leq v_0$ and therefore Lemma 13(1) implies $g_A(v') \leq 0$, i.e., $f_A(v') \leq v'(A)$.

(c) If $v(A) = 1$, then also $v_0(A) = 1$ and, hence, $v'(A) = 1$.

(d) $f_A(v') \leq v'(A)$. This holds due to Observations (a) and (b) and due to the fact that at least one of the three statements $f_A(v) < v(A)$, $A \in N(v)$, or $v(A) = 1$ is true.

Observation (d) implies that $f(v') \leq v'$. Clearly, $v' \in [0,1]^N \setminus \{\mathbf{1}\}$ because $v \in [0,1]^N \setminus \{\mathbf{1}\}$, $v_0 \in [0,1]^N$, and $\lambda \in]0,1[$. Hence, $v' \in S$. Moreover, by Observations (a) and (c) we obtain $N(v') \subseteq N(v)$.

What is left is to show that $N(v') \neq N(v)$. On the contrary, suppose that $N(v') = N(v)$. We will derive a contradiction.

Let $A \in N(v) = N(v')$. Then $f_A(v) \not\leq v(A)$ and, since $v \in \mathrm{Pre}$, we conclude that $f_A(v) = v(A)$; hence $g_A(v) = 0$. Likewise, we obtain $g_A(v') = 0$. Furthermore, $g_A(v_0) \leq 0$ by (8). By Lemma 13(2) we can assert that $g_A(v_0) = 0$. Observe that $\{B \in N \mid v(B) \neq v_0(B)\} = N(v)$. Then the second statement of Lemma 13(4) yields

$$\sum\nolimits_{B \in N(v)} (1 - v(B)) d_B^A \geq 1 - v(A)\,, \tag{9}$$

where $d_B^A = \sum_{r = A \to wBw'} \kappa(r)$. Moreover, by the first statement of Lemma 13(4),

$$\begin{aligned}\sum\nolimits_{B \in N(v)} d_B^A &= \sum\nolimits_{B \in N(v)} \sum\nolimits_{r = A \to wBw'} \kappa(r) \\ &= \sum \big(\kappa(r) \mid r \in R_A, \exists\, B \in N(v) : r \,=\, A \to wBw'\big) \\ &\leq \sum\nolimits_{r \in R_A} \kappa(r) = 1 \,. \end{aligned} \tag{10}$$

For the remainder of this proof choose and fix an $A \in N(v)$ such that $v(A)$ is minimal, i.e., $v(A) \leq v(B)$ for every $B \in N(v)$. Such an A exists because $N(v) \neq \emptyset$ by assumption. Let N_0 be the set of all $B \in N(v)$ with $d_B^A > 0$. If $N_0 \neq \emptyset$, then (9) and (10) imply:

$$1 - v(A) \leq \sum\nolimits_{B \in N_0} (1 - v(B)) d_B^A \leq \sum\nolimits_{B \in N_0} (1 - v(B)) d_B^A \big(\sum\nolimits_{B \in N_0} d_B^A\big)^{-1} \,.$$

The latter term is a weighted average of the terms $(1 - v(B))$ for $B \in N_0$. Then it is easy to see that the minimality of $v(A)$ implies $v(A) = v(B)$ for every $B \in N_0$. Let us reformulate this property. Define the relation $\prec$ on $N(v)$ for every $B, C \in N(v)$ by: $B \prec C$ iff $d_C^B > 0$. We have just shown that if $v(A)$ is minimal, then $v(A) = v(B)$ for every $B \in N(v)$ with $A \prec B$; hence, for every such B, $v(B)$ is also minimal. By induction, we obtain $v(A) = v(B)$ for every $B \in N(v)$ with $A \prec^* B$, where $\prec^*$ is the transitive-reflexive closure of $\prec$.

Let $B \in N(v)$ with $A \prec^* B$. By (9):

$$1 - v(A) = 1 - v(B) \leq \sum\nolimits_{\substack{C \in N(v) \\ B \prec C}} (1 - v(C)) d_C^B = (1 - v(A)) \sum\nolimits_{\substack{C \in N(v) \\ B \prec C}} d_C^B \,.$$

Since $A \in N(v)$, $v(A) \neq 1$, and hence, $1 \leq \sum_{C \in N(v)} d_C^B$. Together with (10) we obtain $\sum_{C \in N(v)} d_C^B = 1$. Then the first statement of Lemma 13(4) yields

$$\begin{aligned} 1 &= \sum\nolimits_{C \in N(v)} d_C^B = \sum\nolimits_{\substack{C \in N(v) \\ B \prec C}} d_C^B = \sum\nolimits_{\substack{C \in N(v) \\ B \prec C}} \sum\nolimits_{r = B \to wCw'} \kappa(r) \\ &= \sum \big(\kappa(r) \mid r \in R_B, \exists\, C \in N(v) : B \prec C \text{ and } r \,=\, B \to wCw'\big) \,. \end{aligned}$$

Since G is proper, this implies that for every $r \in R_B$ with $\kappa(r) > 0$, there is a $C \in N(v)$ with $A \prec^* C$ such that C occurs on the right-hand side of r. Hence, none of the nonterminals $B \in N(v)$ with $A \prec^* B$ are productive, a contradiction to the condition that G is reduced. ∎

Next we prove the equivalence of the third and fourth statement of Theorem 5. Observe that for every $A, B \in N$:

$$M_{\mathrm{fm}}(A, B) = \frac{\partial f_A}{\partial B}(\mathbf{1}) \,, \qquad\qquad M(A, B) = \frac{\partial g_A}{\partial B}(\mathbf{1}) \,, \tag{11}$$

i.e., $M_{\mathrm{fm}}(A, B)$ is the partial derivative of the polynomial f_A with respect to the B-component of the argument vector. This follows immediately from the definition of M_{fm}, M, f_A, and g_A.

The following lemma states the equivalence of the third and fourth statement of Theorem 5. Its proof idea is as follows. Suppose that StrictPre contains a vector v. Using the definition of StrictPre and Lemma 13 we obtain that for $A \in N$, the directional derivative of g_A along the line connecting v and $\mathbf{1}$ is positive at $\mathbf{1}$ (see Figure 3 for an illustration of this situation) – note that this explanation is simplified as there might be degenerate cases. Using Equation (11), we obtain that this directional derivative can be expressed as $\big(M(\mathbf{1} - v)\big)(A)$. Then the vector $\mathbf{1} - v$ satisfies the conditions of the fourth statement of Theorem 5.

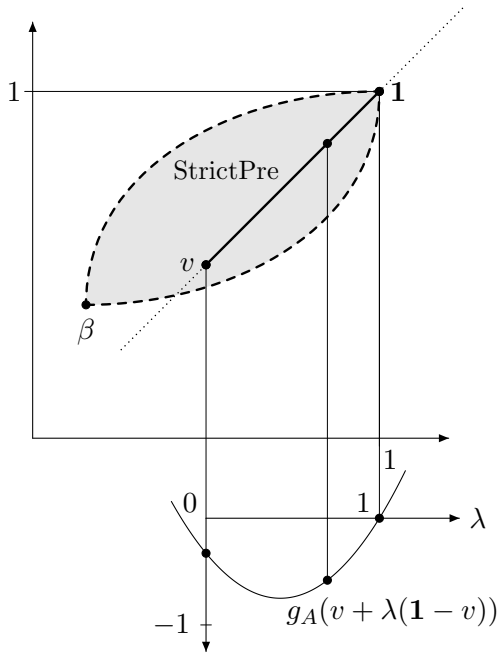


Figure 3: Illustration of Lemma 15.

Lemma 15. StrictPre $= \emptyset$ *iff for every* $v \in \mathbb{R}^N_{\geq 0} \setminus \{\mathbf{0}\}$ *there is an* $A \in N$ *such that* $v(A) > 0$ *and* $(Mv)(A) \leq 0$.

Proof. Before we begin with the main part of this proof, we derive some auxiliary statements. Let $v \in [0,1]^N \setminus \{\mathbf{1}\}$ and $h_v : \mathbb{R} \to \mathbb{R}^N : \lambda \mapsto v + (\mathbf{1} - v)\lambda$. Then for every $A \in N$ the chain rule of multivariable functions yields

$$\begin{aligned}(g_A \circ h_v)'(1) &= \sum\nolimits_{B \in N} \frac{\partial g_A}{\partial B}(h_v(1)) \cdot (1 - v(B)) = \sum\nolimits_{B \in N} \frac{\partial g_A}{\partial B}(\mathbf{1}) \cdot (1 - v(B)) \\ &= \sum\nolimits_{B \in N} M(A,B) \cdot (\mathbf{1} - v)(B) = \big(M(\mathbf{1} - v)\big)(A)\ . \end{aligned} \tag{12}$$

First we prove the if-part of the lemma. To this end we assume that StrictPre $\neq \emptyset$, say $v \in$ StrictPre. Then $v' = \mathbf{1} - v$ is in the set $\mathbb{R}^N_{\geq 0} \setminus \{\mathbf{0}\}$ by definition of StrictPre. We show that v' has the desired properties, i.e., for every $A \in N$ with $v'(A) > 0$ we have $(Mv')(A) > 0$.

Let $A \in N$ with $v'(A) > 0$; then $v(A) \neq 1$. Since $v \in$ StrictPre, this implies $f_A(v) < v(A)$ or, equivalently, $g_A(v) < 0$. Clearly, $g_A(\mathbf{1}) = 0$ due to Observation 6. Then Lemma 13(3) and the fact that $g_A(v) < 0$ imply $(g_A \circ h_v)'(1) > 0$. By (12), $(g_A \circ h_v)'(1) = \big(M(\mathbf{1} - v)\big)(A) = (Mv')(A) > 0$.

It remains to prove the only-if part of our lemma. Assume that there is a $v \in \mathbb{R}^N_{\geq 0} \setminus \{\mathbf{0}\}$ such that $(Mv)(A) > 0$ for every $A \in N$ with $v(A) > 0$. Without loss of generality we can assume that $v(A) \leq 1$ for every $A \in N$, otherwise we can scale v by an appropriate positive scalar. Let $v' = \mathbf{1} - v$; then $v' \in [0,1]^N \setminus \{\mathbf{1}\}$. By (12), $(Mv)(A) = (g_A \circ h_{v'})'(1)$ for every $A \in N$. Clearly, $(g_A \circ h_{v'})(1) = g_A(\mathbf{1}) = 0$ for every $A \in N$. Since $(g_A \circ h_{v'})$ is continuous for every A, there is a $\lambda \in]0,1[$ such that $(g_A \circ h_{v'})(\lambda) < 0$ for every $A \in N$ with $(g_A \circ h_{v'})'(1) > 0$; put $v_0 = h_{v'}(\lambda)$. We show that $v_0 \in$ StrictPre.

First observe that $v_0 \in [0,1]^N \setminus \{\mathbf{1}\}$ because $\lambda < 1$ and $v' \in [0,1]^N \setminus \{\mathbf{1}\}$. Let $A \in N$. It remains to show that $f_A(v_0) \leq v_0(A)$ and that $v_0(A) = 1$ or $f_A(v_0) < v_0(A)$. We distinguish two cases.

Case 1: $(Mv)(A) > 0$. Then $(g_A \circ h_{v'})'(1) > 0$ and, thus, $g_A(v_0) = (g_A \circ h_{v'})(\lambda) < 0$ by the definition of λ. Hence, $f_A(v_0) < v_0(A)$.

Case 2: $(Mv)(A) \leq 0$. Then $v(A) = 0$ by the definition of v. Thus, $v'(A) = 1$ and, therefore, $v_0(A) = 1$. It remains to show that $f_A(v_0) \leq v_0(A)$. This follows from $v_0(A) = 1$, $v_0 \leq \mathbf{1}$, the monotonicity of f_A and Observation 6. ∎

We conclude this section by proving the equivalence of the fourth and fifth statement of Theorem 5. We included this equivalence only for completeness, it is not required for proving Theorem 2. In order to demonstrate the correctness of this statement, we need to dip into Perron-Frobenius theory of matrices. The following lemma is taken from Horn and Johnson (1990, Corollary 8.3.3).

Lemma 16. *Let $M' \in \mathbb{R}_{\geq 0}^{N \times N}$. Then*

$$\rho(M') = \max_{v \in \mathbb{R}_{\geq 0}^N \setminus \{\mathbf{0}\}} \min_{\substack{A \in N \\ v(A) \neq 0}} \frac{(M'v)(A)}{v(A)} \,.$$

Lemma 17. *$\rho(M_{\mathrm{fm}}) \leq 1$ iff for every $v \in \mathbb{R}_{\geq 0}^N \setminus \{\mathbf{0}\}$ there is an $A \in N$ such that $v(A) > 0$ and $(Mv)(A) \leq 0$.*

Proof. By Lemma 16, $\rho(M_{\mathrm{fm}}) \leq 1$ iff for every $v \in \mathbb{R}_{\geq 0}^N \setminus \{\mathbf{0}\}$ there is an $A \in N$ with $v(A) \neq 0$ and $(M_{\mathrm{fm}}v)(A) \leq v(A)$. Then the lemma follows from the fact that $(Mv)(A) = \sum_{B \in N} M(A,B) \cdot v(B) = \big(\sum_{B \in N} M_{\mathrm{fm}}(A,B) \cdot v(B)\big) - v(A) = (M_{\mathrm{fm}}v)(A) - v(A)$. ∎

The following corollary concludes this section; it is based on those parts of Theorem 5 that we have proved in this section. Let $\mathbb{R}_{>0}$ denote the set of positive reals.

Corollary 18. *Let G be reduced and $w \in \mathbb{R}_{>0}^N$ such that $w^T M_{\mathrm{fm}} v \leq w^T v$ for every vector $v \in \mathbb{R}_{\geq 0}^N \setminus \{\mathbf{0}\}$. Then G is consistent.*

Proof. Assume that G is inconsistent. We will derive a contradiction. According to Theorem 5 there is a $v \in \mathbb{R}_{\geq 0}^N \setminus \{\mathbf{0}\}$ such that $(Mv)(A) > 0$ for every $A \in N$ with $v(A) > 0$. Then

$$\begin{aligned} w^T M_{\mathrm{fm}} v &= \sum\nolimits_{A \in N} w(A) \cdot (M_{\mathrm{fm}}v)(A) = \sum\nolimits_{A \in N} w(A) \cdot \big((Mv)(A) + v(A)\big) \\ &= \Big(\sum\nolimits_{A \in N} w(A) \cdot (Mv)(A)\Big) + w^T v \,. \end{aligned}$$

Since $v \neq \mathbf{0}$, there is at least one $A \in N$ such that $v(A) > 0$; hence, $(Mv)(A) > 0$. Since $w(A) > 0$ for every $A \in N$, we conclude that $\sum_{A \in N} w(A) \cdot (Mv)(A) > 0$. Thus, $w^T M_{\mathrm{fm}} v > w^T v$, a contradiction. ∎

5 Decision Procedure

In this section we prove the remaining part of Theorem 5, viz., the equivalence of its fourth and sixth statement; moreover, we present a procedure for deciding whether an arbitrary pcfg is consistent. The algorithm in Figure 4 on page 15 forms the main part of this decision procedure.

Before diving into an explanation of this algorithm, we first need to define two properties of $n \times n$-matrices, where n is some positive integer. Let $M' \in \mathbb{R}^{n \times n}$. We say that M' has *Property (A)* if

$$M'(i,j) \geq 0 \text{ for every } i \neq j \; . \tag{A}$$

Moreover, we say that M' has *Property (B)* if

$$\exists\, v \in \mathbb{R}^n_{\geq 0} \setminus \{\mathbf{0}\} : (M'v)(i) > 0 \text{ for every } 1 \leq i \leq n \text{ with } v(i) > 0 \; . \tag{B}$$

Recall that we do not distinguish between matrices of the form $\mathbb{R}^{n \times n}$ and $\mathbb{R}^{N \times N}$; cf. the remark after Theorem 5. In this section we merely introduce matrices of the form $\mathbb{R}^{n \times n}$ in order to employ a canonical order of the rows and columns of the matrix.

The algorithm in Figure 4 takes a matrix having Property (A) as input and decides whether it has Property (B). Note that the definition of the matrix M implies that M has Property (A). Furthermore, observe that Property (B) is precisely the complement of the fourth statement of Theorem 5. Hence, if the algorithm in Figure 4 is correct, then the fourth and sixth statement of Theorem 5 are equivalent.

Input: matrix $M \in \mathbb{R}^{n \times n}$ having Property (A)
Output: whether M has Property (B)

```
 1  for k ← n down to 1
 2      i_null ← 0, i_neg ← 0
 3      for i ← 1 to k
 4          if M(i,i) > 0, then return "M has Property (B)"
 5          else if M(i,i) < 0, then i_neg ← i
 6          else if M(i,1) = M(i,2) = ... = M(i,k) = 0, then i_null ← i
 7      if i_null > 0, then i_neg ← i_null
 8      if i_neg = 0, then return "M has Property (B)"
 9      for j ← 1 to k
10          swap entry (i_neg, j) with (k,j) in M
11      for j ← 1 to k
12          swap entry (j, i_neg) with (j,k) in M
13      if i_null = 0, then
14          for i ← 1 to k − 1
15              for j ← 1 to k − 1
16                  M(i,j) ← M(k,j) · M(i,k) − M(i,j) · M(k,k)
17  return "M does not have Property (B)"
```

Figure 4: Algorithm for deciding whether a matrix with Property (A) has Property (B).

Now we present an example execution of the algorithm in Figure 4.

Example 19 (Cont. of Example 4). First observe that M has Property (A). Consider the first execution of the outer loop, i.e., for $k = 5$. The loop in lines 3–6 will quit prematurely if there is a positive diagonal element in M. This is not the case. Otherwise, after the execution of this inner loop, i_{null} is the index of the last row containing only 0; if there is no such row – as in our case – then i_{null} is 0. The variable i_{neg} is the index of the last negative diagonal element; thus, $i_{\text{neg}} = 5$ in our situation.

The statements in lines 7 and 8 do not have any effect in this first execution of the outer loop. The two loops in lines 9–12 swap line i_{neg} with line k and row i_{neg} with row k (see Figure 5). Since $i_{\text{neg}} = 5 = k$, these loops will have no effect right now.

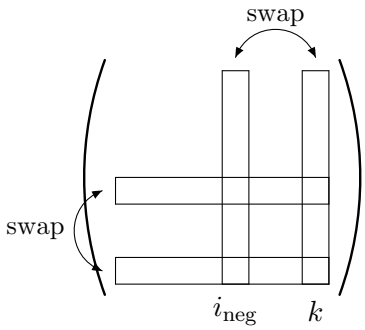


Figure 5: Swapping of row/column i_{neg} with row/column k in lines 9–11.

Finally the algorithm executes the loop in lines 13–16, which is essentially a single step of Gaussian elimination (eliminating row k); however, for reasons of efficiency the k-th entry of each row is not eliminated as this entry is not needed throughout the later stages of the algorithm. We obtain the following matrix after the first iteration of the outer loop

$$\begin{pmatrix} 0 & 1/2 & 2 & 1/2 \\ 0 & -1/2 & 1 & 0 \\ 0 & 1/2 & -1 & 0 \\ 1/4 & 1/4 & 2 & 0 \end{pmatrix};$$

note that we omitted the last row and last column because they are not needed for the remainder of the execution.

In the next iteration of the outer loop (i.e., $k = 4$) the algorithm computes $i_{\text{null}} = 0$ and $i_{\text{neg}} = 3$; hence, in lines 9–12, the third and fourth row and the third and fourth column are swapped, resulting in the matrix

$$\begin{pmatrix} 0 & 1/2 & 1/2 & 2 \\ 0 & -1/2 & 0 & 1 \\ 1/4 & 1/4 & 0 & 2 \\ 0 & 1/2 & 0 & -1 \end{pmatrix}.$$

Another step of Gaussian elimination in lines 13–16 yields the matrix (this time omitting the last two rows and columns):

$$\begin{pmatrix} 0 & 3/2 & 1/2 \\ 0 & 0 & 0 \\ 1/4 & 5/4 & 0 \end{pmatrix}.$$

In the next iteration of the outer loop we obtain $i_{\text{null}} = 2$ and $i_{\text{neg}} = 0$ after the loop in lines 3–6. Then in line 7, i_{neg} will be assigned the value 2. Hence, in lines 9–12, the second and third row/column are swapped. Since $i_{\text{null}} \neq 0$, there will be no Gaussian elimination in this execution of the outer loop. Hence, we obtain the matrix

$$\begin{pmatrix} 0 & 1/2 \\ 1/4 & 0 \end{pmatrix} .$$

In the fourth iteration of the outer loop we compute $i_{\text{null}} = i_{\text{neg}} = 0$. Hence, the algorithm will terminate and answer "M has Property (B)" in line 8.

It is easy to see that our algorithm is correct for this example because M does actually have Property (B). In fact, $M \cdot (4, 0, 0, 2, 0)^T = (1, 0, 0, 1, 0)^T$. □

Intuitively, the algorithm performs the following steps throughout every iteration of the outer loop:

- It looks for a row i in the current matrix having a negative diagonal element or having all entries equal to zero. If there is no such row or if the matrix contains a positive diagonal element, then the algorithm halts and outputs "M has Property (B)".
- It interchanges row i and column i with the last row and column.
- If necessary, it eliminates the last row by means of Gaussian elimination.
- It removes the last row and column from the current matrix – albeit this is just a logical removal, it is not actually performed in memory.

All these operations are guaranteed to conserve Properties (A) and (B). This is expressed formally by the following lemmas. Putting these lemmas together yields the correctness of the algorithm in Figure 4; see Lemma 24.

Observation 20. *Let $1 \leq k \leq n$ and let $M' \in \mathbb{R}^{n \times n}$ originate from M by interchanging row k with row n and by interchanging column k with column n, i.e., $M'(i, j) = M(\pi(i), \pi(j))$, where $\pi(k) = n$, $\pi(n) = k$, and $\pi(i) = i$ for every other i.*

If M has Property (A), then M' has Property (A). Furthermore, M has Property (B) iff M' has Property (B).

Lemma 21. *Suppose that M has Property (A) and $M(n, i) = 0$ for every $1 \leq i \leq n$. Define $M' \in \mathbb{R}^{n-1 \times n-1}$ by letting $M'(i, j) = M(i, j)$. Then M' has Property (A). Furthermore, M' has Property (B) iff M has Property (B).*

Proof. It is easy to see that M' has Property (A). Suppose that M' has Property (B). Hence, there is a $v' \in \mathbb{R}^{n-1}_{\geq 0} \setminus \{\mathbf{0}\}$ such that $(M'v')(i) > 0$ for every $1 \leq i \leq n-1$ with $v'(i) > 0$. Define $v \in \mathbb{R}^n_{\geq 0} \setminus \{\mathbf{0}\}$ by $v(n) = 0$ and $v(i) = v'(i)$ for every $1 \leq i \leq n-1$. The fact that $M(i, n) \geq 0$ for every $1 \leq i \leq n-1$ implies that $(Mv)(i) > 0$ for every $1 \leq i \leq n$ with $v(i) > 0$. Thus, M has Property (B).

Now suppose that M has Property (B). Let $v \in \mathbb{R}^n_{\geq 0} \setminus \{\mathbf{0}\}$ such that $(Mv)(i) > 0$ for every $1 \leq i \leq n$ with $v(i) > 0$. Define $v' \in \mathbb{R}^{n-1}_{\geq 0}$ by $v'(i) = v(i)$ for every $1 \leq i \leq n-1$. Clearly, $(Mv)(n) = 0$ and we conclude that $v(n) = 0$; hence, $v' \neq \mathbf{0}$ and $(M'v')(i) = (Mv)(i)$ for every $1 \leq i \leq n-1$. It is easy to see that this implies that M' has Property (B). ■

Lemma 22. *Suppose that $M(i,j) \geq 0$ for every $1 \leq i,j \leq n$. Then M has Property (B) iff there is a nonempty set $I \subseteq \{1,\ldots,n\}$ such that for every $i \in I$ there is a $j \in I$ with $M(i,j) > 0$.*

PROOF. For every $v \in \mathbb{R}^n_{\geq 0}$ let $I_v = \{i \mid 1 \leq i \leq n, v(i) > 0\}$. If M has Property (B), then there is a $v \in \mathbb{R}^n_{\geq 0}$ such that I_v is nonempty and for every $i \in I_v$, $(Mv)(i) > 0$. Clearly, $(Mv)(i) > 0$ iff there is a $j \in I_v$ with $M(i,j) > 0$. This implies the only-if-part.

The if-part follows from a similar argument: for a given nonempty set $I \subseteq \{1,\ldots,n\}$ let $v(i) = 1$ if $i \in I$ and $v(i) = 0$ otherwise. Then it is easy to see that v witnesses that M has Property (B). ∎

The following lemma states that the Gaussian elimination performed in lines 13–16 of Figure 4 preserves Properties (A) and (B). It is the heart of the correctness proof of the algorithm.

Lemma 23. *Suppose that M has Property (A) and $M(n,n) < 0$. Let $M' \in \mathbb{R}^{n-1\times n-1}$ with*

$$M'(i,j) = M(n,j) \cdot M(i,n) - M(i,j) \cdot M(n,n) \; .$$

Then M' has Property (A). Furthermore, M' has Property (B) iff M has Property (B).

PROOF. First we show that M' has Property (A). Let $i \neq j$. Since M has Property (A), we conclude that $M(n,j) \geq 0$, $M(i,n) \geq 0$ and $M(i,j) \geq 0$. Then the assumption $M(n,n) < 0$ implies the assertion.

Next we show that M' has Property (B) iff M has Property (B). For the if-part suppose that M has Property (B). Then there is a $v \in \mathbb{R}^n_{\geq 0} \setminus \{\mathbf{0}\}$ such that $(Mv)(i) > 0$ for every $1 \leq i \leq n$ with $v(i) > 0$. Let $v' \in \mathbb{R}^{n-1}_{\geq 0}$ with $v'(j) = v(j)$ for every $1 \leq j \leq n-1$.

Assume that $v' = \mathbf{0}$. Then $(Mv)(n) = M(n,n) \cdot v(n)$. Since $v \neq \mathbf{0}$, we conclude that $v(n) > 0$. Together with $M(n,n) < 0$ this yields $(Mv)(n) < 0$, a contradiction. Thus, $v' \in \mathbb{R}^{n-1}_{\geq 0} \setminus \{\mathbf{0}\}$. Let $1 \leq i \leq n-1$ with $v'(i) > 0$. We show that $(M'v')(i) > 0$. First observe that for every $1 \leq i \leq n-1$:

$$\begin{aligned}
(M'v')(i) &= \sum\nolimits_{j=1}^{n-1} M'(i,j) \cdot v'(j) \\
&= \sum\nolimits_{j=1}^{n-1} \big(M(n,j) \cdot M(i,n) \cdot v(j) - M(i,j) \cdot M(n,n) \cdot v(j)\big) \\
&= M(i,n) \cdot \Big(\sum\nolimits_{j=1}^{n-1} M(n,j) \cdot v(j)\Big) - M(n,n) \cdot \Big(\sum\nolimits_{j=1}^{n-1} M(i,j) \cdot v(j)\Big) \\
&= M(i,n) \cdot \big((Mv)(n) - M(n,n) \cdot v(n)\big) - M(n,n) \cdot \big((Mv)(i) - M(i,n) \cdot v(n)\big) \\
&= M(i,n) \cdot (Mv)(n) - M(n,n) \cdot (Mv)(i) \; .
\end{aligned}$$

Since $v'(i) > 0$, also $v(i) > 0$ and, hence, $(Mv)(i) > 0$. Moreover, $(Mv)(n) \geq 0$: if $v(n) = 0$, then $(Mv)(n) \geq 0$ follows from the fact that M has Property (A); if $v(n) > 0$, then $(Mv)(n) \geq 0$ follows from the definition of v. Then the facts that $M(i,n) \geq 0$ (by Property (A)) and $M(n,n) < 0$ imply that $(M'v')(i) > 0$. We conclude that M' has Property (B).

It remains to prove the only-if-part. Assume that M' has Property (B). Then there is a vector $v' \in \mathbb{R}^{n-1}_{\geq 0} \setminus \{\mathbf{0}\}$ with $(M'v')(i) > 0$ for every $1 \leq i \leq n-1$ with $v'(i) > 0$. Let $1 \leq i \leq n-1$. Then

$$(M'v')(i) = \sum\nolimits_{j=1}^{n-1} \big(M(n,j) \cdot M(i,n) - M(i,j) \cdot M(n,n)\big) \cdot v'(j)$$

$$= M(i,n) \cdot \Big(\sum_{j=1}^{n-1} M(n,j) \cdot v'(j)\Big) - M(n,n) \cdot \Big(\sum_{j=1}^{n-1} M(i,j) \cdot v'(j)\Big) \,. \tag{13}$$

We distinguish two cases.

Case 1: $\sum_{j=1}^{n-1} M(n,j) \cdot v'(j) = 0$. We define $v \in \mathbb{R}^n_{\geq 0} \setminus \{\mathbf{0}\}$ by letting $v(n) = 0$ and $v(j) = v'(j)$ for every $1 \leq j \leq n-1$. Then $(Mv)(n) = \sum_{j=1}^{n-1} M(n,j) \cdot v'(j) = 0$. Together with (13) we conclude

$$(M'v')(i) = -M(n,n) \cdot \Big(\sum_{j=1}^{n-1} M(i,j) \cdot v'(j)\Big) = -M(n,n) \cdot (Mv)(i)$$

for every $1 \leq i \leq n-1$. Then it is easy to see that $(Mv)(i) > 0$ for every $1 \leq i \leq n$ with $v(i) > 0$. Hence, M has Property (B).

Case 2: $\sum_{j=1}^{n-1} M(n,j) \cdot v'(j) \neq 0$. Then $\sum_{j=1}^{n-1} M(n,j) \cdot v'(j) > 0$ because M has Property (A).

Let $1 \leq i \leq n-1$ such that $M(i,n) > 0$. Assume that $(M'v')(i) = 0$. Then $v'(i) = 0$ and we conclude that $\sum_{j=1}^{n-1} M(i,j) \cdot v'(j) \geq 0$ because M has Property (A). Together with $M(n,n) < 0$ and $M(i,n) > 0$, Equation (13) yields $(M'v')(i) > 0$, a contradiction. Hence, $(M'v')(i) > 0$. Now (13) implies

$$\frac{\sum_{j=1}^{n-1} M(n,j) \cdot v'(j)}{-M(n,n)} > \frac{-\sum_{j=1}^{n-1} M(i,j) \cdot v'(j)}{M(i,n)} \,. \qquad \text{(because } M(n,n) < 0\text{)}$$

Note that the left-hand side of this inequality is positive by assumption.

Thus, there is an $r > 0$ such that for every $1 \leq i \leq n-1$ with $M(i,n) > 0$:

$$r < \frac{\sum_{j=1}^{n-1} M(n,j) \cdot v'(j)}{-M(n,n)} \,, \text{ and} \tag{14}$$

$$r > \frac{-\sum_{j=1}^{n-1} M(i,j) \cdot v'(j)}{M(i,n)} \,. \tag{15}$$

Define $v \in \mathbb{R}^n_{\geq 0} \setminus \{\mathbf{0}\}$ by $v(n) = r$ and $v(j) = v'(j)$ for every $1 \leq j \leq n-1$. Our proof is finished if we show that $(Mv)(i) > 0$ for every $1 \leq i \leq n$ with $v(i) > 0$. Let $1 \leq i \leq n$ with $v(i) > 0$. We distinguish three cases.

If $i = n$, then $(Mv)(i) = \sum_{j=1}^{n} M(n,j) \cdot v(j) = M(n,n) \cdot r + \sum_{j=1}^{n-1} M(n,j) \cdot v'(j) > 0$ due to (14).

If $1 \leq i \leq n-1$ and $M(i,n) > 0$, then $(Mv)(i) = M(i,n) \cdot r + \sum_{j=1}^{n-1} M(i,j) \cdot v'(j) > 0$ due to (15).

If $1 \leq i \leq n-1$ and $M(i,n) = 0$, then $(M'v')(i) = -M(n,n) \cdot \Big(\sum_{j=1}^{n-1} M(i,j) \cdot v'(j)\Big) = -M(n,n) \cdot (Mv)(i)$ due to (13). Then the fact that $v(i) > 0$ implies $v'(i) > 0$, which yields $(M'v')(i) > 0$; hence $(Mv)(i) > 0$. ∎

Lemma 24. *On input M, the algorithm in Figure 4 returns "M has Property (B)" iff M has Property (B).*

Proof. In order to distinguish the variable M, whose value changes during the execution of the algorithm, from the input matrix M, we denote the input matrix by M_{input} throughout this algorithm.

For every $0 \leq k \leq n$ let $M_k \in \mathbb{R}_{\geq 0}^{k \times k}$ such that $M_k(i,j) = M(i,j)$ for every $1 \leq i,j \leq k$. First we prove that the following statement is an invariant of the outer loop (from line 1 to line 16):

$$M_k \text{ has Property (A), and } M_k \text{ has Property (B) iff } M_{\text{input}} \text{ has Property (B).} \qquad (\mathbf{I})$$

The invariant trivially holds on entry into the loop. We show that the invariant remains true on every iteration of the loop. Let $0 < k \leq n$ and consider the according iteration of the outer loop, i.e., if $k = n$, then consider the first iteration, etc. Suppose that this iteration does not quit prematurely by executing the return statements in lines 4 or 8. Consider the current value of M at the beginning of this iteration and denote by M' the value of the matrix at the end of this iteration. Accordingly, we denote by M_k the upper left $k \times k$-submatrix of M and by M'_{k-1} the upper left $(k-1) \times (k-1)$-submatrix of M'.

Suppose that the invariant (**I**) holds at the beginning of this iteration, i.e., that M_k has Property (A) and, furthermore, that M_k has Property (B) iff M_{input} has Property (B). We need to establish that

- M'_{k-1} has Property (A), and
- M'_{k-1} has Property (B) iff M_{input} has Property (B).

Consider the value of the variables i_{null} and i_{neg} after the execution of the loop from lines 3 to 6. At least one of these values is different from zero, otherwise the algorithm would quit in line 8; this contradicts the assumption that the algorithm does not quit in this iteration of the outer loop. We distinguish two cases.

Case 1: $i_{\text{null}} > 0$. Then $M(i_{\text{null}}, 1) = M(i_{\text{null}}, 2) = \ldots = M(i_{\text{null}}, k) = 0$ due to line 6, i.e., the row i_{null} in the matrix M_k is zero. In lines 9 to 12 the rows i_{null} and k as well as the columns i_{null} and k of the matrix M_k are interchanged. Let us denote the resulting matrix by M_k^{swap}. Due to Observation 20, M_k^{swap} has Property (A); moreover, M_k^{swap} has Property (B) iff M_k has Property (B). Since $i_{\text{null}} > 0$, lines 14 to 16 are not executed and M'_{k-1} results from M_k^{swap} by removing the k-th row and k-th column. Since every element of the k-th row of M_k^{swap} is zero, Lemma 21 yields that M'_{k-1} has Property (A); moreover, M'_{k-1} has Property (B) iff M_k^{swap} has Property (B). Hence, the invariant holds after this execution of the loop.

Case 2: $i_{\text{null}} = 0$ and $i_{\text{neg}} > 0$. Then $M(i_{\text{neg}}, i_{\text{neg}}) < 0$. Similarly to Case 1, let M_k^{swap} be the value of M_k after executing the loops from line 9 to line 12. Again, M_k^{swap} has Property (A) and, furthermore, M_k^{swap} has Property (B) iff M_k has Property (B). Observe that $M_k^{\text{swap}}(k,k) < 0$. Since $i_{\text{null}} = 0$, the loop from line 14 to line 16 is executed. Lemma 23 yields that M'_{k-1} has Property (A); moreover, M'_{k-1} has Property (B) iff M_k^{swap} has Property (B).

This finishes the proof that (**I**) is an invariant of the outer loop. Now we continue the correctness proof of the algorithm. We need to distinguish three cases.

Case 1: the algorithm terminates in line 4. Then there is a $1 \leq i \leq k$ such that $M_k(i,i) > 0$. Thus, M_k has Property (B): let $v \in \mathbb{R}_{\geq 0}^k \setminus \{\mathbf{0}\}$ with $v(i) = 1$ and $v(l) = 0$ for every other l; then $(M_k v)(i) = M_k(i,i) > 0$ and $(M_k v)(l) = M_k(l,i) \geq 0$ for every other l because M_k has Property (A) due to invariant (**I**). Furthermore, (**I**) implies that also M_{input} has Property (B).

Case 2: the algorithm terminates in line 8. Thus, both i_{null} and i_{neg} are zero after the execution of the loop from line 3 to line 6. We conclude that, for every $1 \leq i \leq k$, $M_k(i,i) = 0$ and that there is a $1 \leq j \leq k$ such that $M_k(i,j) \neq 0$. This yields that $M_k \in \mathbb{R}_{\geq 0}^{k \times k}$ because M_k has Property (A) due to (**I**). Lemma 22 implies that M_k has Property (B); this is witnessed by letting $I = \{1, \ldots, k\}$. Invariant (**I**) implies that also M_{input} has Property (B).

Case 3: the algorithm terminates in line 17. Clearly, k is zero after the execution of the outer loop. The loop invariant yields that M_0 has Property (B) iff M_{input} has Property (B). However, M_0 does not have Property (B): every vector v in $\mathbb{R}_{\geq 0}^{0\times 0}$ is equal to the zero-dimensional zero-vector. Thus, M_{input} does not have Property (B) either. ∎

Proof of Theorem 2

We conclude this section by giving a proof of Theorem 2. Given a proper pcfg $G = (N, S, R, \kappa)$, denote the set of non-reachable (resp. non-productive) nonterminals of G by N^{nr} (resp. N^{np}). One can decide whether G is consistent by employing the following steps:

1. Construct the sets N^{nr} and N^{np}.
2. If $N^{\text{np}} \not\subseteq N^{\text{nr}}$, then G is inconsistent.
3. Remove all nonterminals in N^{nr} and all rules containing nonterminals in N^{nr} from G.
4. Construct M.
5. Execute the algorithm in Figure 4 on input M. If it returns "does have Property (B)", then G is inconsistent; otherwise G is consistent.

The correctness of these steps follows from Lemma 11 and Theorem 5. The first four steps have time complexity $\mathcal{O}(\text{size}(R))$, see Observation 10. The last step has time complexity $\mathcal{O}(|N|^3)$ in the unit-cost model of computation. Hence, the complete method has time complexity $\mathcal{O}(n^3)$, where $n = \max\{\text{size}(R), |N|\}$.

6 Consistency of Grammars Resulting from Training

In the field of natural-language processing the rule probabilities of a pcfg are usually estimated from data by means of maximum-likelihood estimation (Wetherell, 1980; Prescher, 2005). Then the following question arises: is a grammar that results from such a training process guaranteed to be consistent? This question has first been investigated in Wetherell (1980) and has been answered affirmatively by Chaudhuri et al. (1983); Chi and Geman (1998). In this section we give a simpler alternative proof of the result by Chi and Geman (1998); in fact, we show that it is a simple corollary of Theorem 5. Moreover, our result (see Theorem 26) extends the result by Chi and Geman: it is applicable to a broader variety of data and we will show that every maximum-likelihood estimate yields a consistent grammar.

First let us recall some concepts. The data are represented as a corpus, which is either (i) a sequence $t_1, \ldots, t_n$ of parse trees – or, equivalently, syntax trees in T_S – (called *complete-data corpus*) or (ii) a sequence $w_1, \ldots, w_n$ of strings in Σ^* (called *incomplete-data corpus*). The latter type of corpora is called incomplete because for every string w there are multiple possible syntax trees that represent w, i.e., whose front is w. More precisely, there is a mapping $\mathcal{A} : \Sigma^* \to \mathcal{P}(T_S)$ such that $\mathcal{A}(w) \cap \mathcal{A}(w') = \emptyset$ for every $w \neq w'$; $\mathcal{A}$ assigns to every string the set of syntax trees of that string. This mapping is called an *analyzer*.

In general, incomplete data can emerge in more varied forms, e.g., fragments of parse trees or fragmentary strings. Hence, in this general form incomplete-data corpora are sequences $y_1, \ldots, y_n$ of elements taken from a set Y of *observations*. Moreover, there is an analyzer $\mathcal{A} : Y \to \mathcal{P}(T_S)$ such that $\mathcal{A}(y) \cap \mathcal{A}(y') = \emptyset$ for every $y \neq y'$; $\mathcal{A}$ associates every observation

with possible syntax trees that are consistent with the observation. Note that, given this definition every complete-data corpus is also an incomplete-data corpus by letting $Y = T_S$ and by letting $\mathcal{A}$ be the identity; therefore, we restrict ourselves to incomplete-data corpora in the following discussion.

In the sequel we fix a pcfg $G = (N, S, R, \cdot)$; the last component of G is irrelevant. We say that a probability assignment $\kappa : R \to [0, 1]$ is *proper* (*consistent*, respectively) if (N, S, R, κ) is proper (consistent, respectively). For every syntax tree $t \in T_S$, and $y \in Y$ we define

$$P_\kappa(y) = \sum\nolimits_{t' \in \mathcal{A}(y)} P_\kappa(t') \;, \qquad P_\kappa(t \mid y) = \begin{cases} P_\kappa(t)/P_\kappa(y) \;, & \text{if } t \in A(y) \\ 0 \;, & \text{otherwise} \end{cases}$$

Observe that this definition implies

$$\sum\nolimits_{t \in T_S} P_\kappa(t \mid y) = 1 \;. \tag{16}$$

Given a corpus $y_1, \ldots, y_n$ a *maximum-likelihood estimate* (for short: *mle*) of $y_1, \ldots, y_n$ is a proper probability assignment κ such that

$$\prod\nolimits_{i=1}^{n} P_\kappa(y_i) \geq \prod\nolimits_{i=1}^{n} P_{\kappa'}(y_i) \;.$$

for every proper probability assignment κ'; see Chi and Geman (1998); Prescher (2005).

The process of training a grammar consists of (i) determining an mle of the given corpus and (ii) using the result as the probability assignment of the grammar. In general there is no closed form solution of the first step. However, an mle can be approximated by means of the EM algorithm (Baker, 1979; Dempster et al., 1977). A representation of this algorithm that is based on Prescher (2005) is shown in Figure 6. The argmax that is executed in the M step is defined over all proper probability assignments $\kappa : R \to [0, 1]$. Note that Prescher defines corpora as multisets $f : Y \to \mathbb{R}_{\geq 0}$ (with real-valued frequencies) instead of sequences $y_1, \ldots, y_n$.

Input: probability assignment $\kappa_0 : R \to [0, 1]$, corpus $y_1, \ldots, y_n \in Y^n$
Output: a sequence $\kappa_1, \kappa_2, \ldots$ of probability assignments
Variables: a mapping $h : T_S \to \mathbb{R}_{\geq 0}$
1 **for each** $i \leftarrow 0, 1, 2, \ldots$
2 *E step:* $h(t) \leftarrow \sum_{j=1}^{n} P_{\kappa_i}(t \mid y_j)$
3 *M step:* $\kappa_{i+1} \leftarrow \operatorname{argmax}_\kappa \prod_{t \in T_S} P_\kappa(t)^{h(t)}$

Figure 6: EM algorithm for training pcfgs based on Prescher (2005).

The outer loop of the EM algorithm halts after a predetermined number of executions or when the κ_i stabilize. The EM algorithm does not necessarily approach an mle, but it consecutively increases the likelihood of the corpus; more precisely (Prescher, 2005, Theorem 5),

$$\prod\nolimits_{i=1}^{n} P_{\kappa_0}(y_i) \leq \prod\nolimits_{i=1}^{n} P_{\kappa_1}(y_i) \leq \prod\nolimits_{i=1}^{n} P_{\kappa_2}(y_i) \leq \cdots \tag{17}$$

Given a proper probability assignment κ we say that κ *results from training* if there is an incomplete-data corpus $y_1, \ldots, y_n$ such that

- κ is an mle of $y_1, \ldots, y_n$ or
- there is a probability assignment κ_0 such that κ is generated in some iteration of the EM algorithm on input κ_0 and $y_1, \ldots, y_n$; i.e., κ is of the form κ_i.

Roughly speaking, κ results from training if it is the actual or an approximative mle. In the remainder of this section we will show that every such κ is consistent. The following lemma is essential for the subsequent derivation.

Lemma 25. *Suppose that κ results from training. Then there is an $h : T_S \to \mathbb{R}_{\geq 0}$ such that, for every $r = A \to w \in R$,*

$$\kappa(r) = \frac{\sum_{t \in T_S} h(t) \cdot \#(r,t)}{\sum_{r' \in R_A} \sum_{t \in T_S} h(t) \cdot \#(r',t)} ,$$

where $\#(r,t)$ is the number of occurrences of r in t.

Proof. First we show that there is a $h : T_S \to \mathbb{R}_{\geq 0}$ such that

$$\prod_{t \in T_S} P_\kappa(t)^{h(t)} \geq \prod_{t \in T_S} P_{\kappa'}(t)^{h(t)} \tag{18}$$

for every proper probability assignment κ'. Since κ results from training there is a corpus $y_1, \ldots, y_n$ such that κ is an mle of $y_1, \ldots, y_n$ or it is generated by the EM algorithm. The latter case trivially implies (18) due to the definition of the M step of the EM algorithm. Now let us consider the former case, i.e., κ is an mle. For every t let $h(t) = \sum_{i=1}^n P_\kappa(t \mid y_i)$. Using Equation (16) we obtain that for every proper probability assignment κ',

$$\frac{\prod_{t \in T_S} P_{\kappa'}(t)^{h(t)}}{\prod_{i=1}^n P_{\kappa'}(y_i)} = \frac{\prod_{t \in T_S} \prod_{i=1}^n P_{\kappa'}(t)^{P_\kappa(t|y_i)}}{\prod_{i=1}^n \prod_{t \in T_S} P_{\kappa'}(y_i)^{P_\kappa(t|y_i)}} = \prod_{i=1}^n \prod_{t \in T_S} P_{\kappa'}(t \mid y_i)^{P_\kappa(t|y_i)} . \tag{19}$$

By Equation (16) and Theorem 1(i) of Prescher (2005) we find that, for every i:

$$\prod_{t \in T_S} P_\kappa(t \mid y_i)^{P_\kappa(t|y_i)} \geq \prod_{t \in T_S} P_{\kappa'}(t \mid y_i)^{P_\kappa(t|y_i)} .$$

Then Equation (19) yields that for every κ':

$$\frac{\prod_{t \in T_S} P_\kappa(t)^{h(t)}}{\prod_{i=1}^n P_\kappa(y_i)} \geq \frac{\prod_{t \in T_S} P_{\kappa'}(t)^{h(t)}}{\prod_{i=1}^n P_{\kappa'}(y_i)} .$$

Now (18) follows from the fact that κ is an mle of $y_1, \ldots, y_n$ and, hence, $\prod_{i=1}^n P_\kappa(y_i) \geq \prod_{i=1}^n P_{\kappa'}(y_i)$. This finishes the proof of (18).

The assertion of the lemma follows from (18) and Theorem 10 of Prescher (2005). ∎

We will now employ Corollary 18 and Lemma 25 in order to show that every κ that results from training is consistent.

Theorem 26. *Suppose that κ results from training. Then κ is consistent.*

PROOF. Let $G = (N, S, R, \kappa)$. By Lemma 25 there is a $h : T_S \to \mathbb{R}_{\geq 0}$ such that, for every $r = A \to w \in R$,

$$\kappa(r) = \frac{\sum_{t \in T_S} h(t) \cdot \#(r,t)}{\sum_{r' \in R_A} \sum_{t \in T_S} h(t) \cdot \#(r',t)} \ .$$

We conclude that $\kappa(r) > 0$ iff r occurs in some $t \in T_S$ with $h(t) > 0$. Then it is easy to see that every reachable nonterminal of G is productive. We can assume that G is reduced by Lemma 11.

Define $w \in \mathbb{R}^N_{\geq 0}$ by letting $w(A) = \sum_{r \in R_A} \sum_{t \in T_S} h(t) \cdot \#(r,t)$ for every $A \in N$. Since G is reduced, every nonterminal is productive. In particular, for every $A \in N$ there is an $r \in R_A$ with $\kappa(r) > 0$; it is easy to see that this implies $w(A) > 0$. Hence, $w \in \mathbb{R}^N_{>0}$. In view of Corollary 18 it suffices to show that $w^T M_{\mathrm{fm}} v \leq w^T v$ for every $v \in \mathbb{R}^N_{\geq 0} \setminus \{\mathbf{0}\}$:

$$\begin{aligned}
w^T M_{\mathrm{fm}} v &= \sum\nolimits_{A,B \in N} w(A) \cdot M_{\mathrm{fm}}(A,B) \cdot v(B) \\
&= \sum\nolimits_{A,B \in N} w(A) \cdot \sum\nolimits_{r = A \to w} \kappa(r) \cdot |w|_B \cdot v(B) \\
&= \sum\nolimits_{A,B \in N} w(A) \cdot \sum\nolimits_{r = A \to w} \frac{\sum_{t \in T_S} h(t) \cdot \#(r,t)}{\sum_{r' \in R_A} \sum_{t \in T_S} h(t) \cdot \#(r',t)} \cdot |w|_B \cdot v(B) \\
&= \sum\nolimits_{A,B \in N} \sum\nolimits_{r = A \to w} \sum\nolimits_{t \in T_S} h(t) \cdot \#(r,t) \cdot |w|_B \cdot v(B) \\
&= \sum\nolimits_{B \in N} v(B) \cdot \Big(\sum\nolimits_{t \in T_S} h(t) \cdot \Big(\sum\nolimits_{A \in N} \sum\nolimits_{r = A \to w} \#(r,t) \cdot |w|_B \Big) \Big) \\
&\leq \sum\nolimits_{B \in N} v(B) \cdot \Big(\sum\nolimits_{t \in T_S} h(t) \cdot \Big(\sum\nolimits_{r \in R_B} \#(r,t) \Big) \Big) = w^T v \ .
\end{aligned}$$

In the last line we used that fact $\sum_{A \in N} \sum_{r = A \to w} \#(r,t) \cdot |w|_B \leq \sum_{r \in R_B} \#(r,t)$ which is a simple property of every syntax tree t. ∎

7 Conclusion

We have presented an algorithm for deciding whether an arbitrary proper probabilistic context-free grammar is consistent. This algorithm has time complexity $\mathcal{O}(n^3)$. Furthermore, we have shown that maximum-likelihood training of pcfg always yields a consistent grammar, even if the maximum-likelihood estimate is approximated by means of the EM algorithm.

Our method carries over to other formalisms. The structure of the set of derivations of more sophisticated grammar models – like weighted regular tree grammars, synchronous context-free grammars and tree-adjoining grammars (see Section 1) – can be described by pcfgs; hence, the algorithm presented in this paper can be employed for deciding consistency of those models.

The following questions and problems have not been covered in this paper and suggest further research.

- Can the result from Section 6 be extended to more advanced grammar models?
- Given a partition function β' (see Section 3 and Nederhof and Satta (2008)), can one employ a modification of our algorithm to decide the correctness of β', i.e., whether $\beta = \beta'$? Is this even possible when we drop the requirement that the input grammar is proper?

- Nederhof and Satta (2008) investigated methods for approximating the inside probabilities of a pcfg (i.e., its partition function). Is it possible to develop alternative approaches for approximating the partition function by employing the techniques developed in this paper?

- The equivalence of the first and fifth statement of Theorem 5 suggests that there are grammars that are *borderline consistent*, i.e., they are still consistent but almost inconsistent. Such grammars have the property that $\rho(M_{\mathrm{fm}}) = 1$. This implies that there is a vector $v \in \mathbb{R}^N_{\geq 0} \setminus \{\mathbf{0}\}$ and a nonterminal A such that $v(A) > 0$ and $(Mv)(A) = 0$. We conjecture that such grammars have some curious properties: e.g., that there is a polynomial p such that $P(|w| = n) \approx p(n)^{-1}$, where $P(|w| = n)$ denotes the probability to derive a word of length n. We conjecture that for every other grammar (i.e., inconsistent and non-borderline consistent grammars) $P(|w| = n)$ does not fall polynomially but exponentially. It is worthwhile to characterize and study such grammars more thoroughly, e.g., to develop automatic training methods that always yield grammars that are borderline consistent.

References

Alexandrakis, A. and Bozapalidis, S. (1987). Weighted grammars and Kleene's theorem. *Inform. Process. Lett.*, 24(1):1–4.

Baker, J. (1979). Trainable grammars for speech recognition. In Klatt, D. and Wolf, J., editors, *Speech Communication Papers for the 97th Meeting of the Acoustical Society of America. Boston, MA*, pages 547–550.

Booth, T. (1969). Probabilistic representation of formal languages. In *IEEE Conference Record of 1969 Tenth Annual Symposium on Switching and Automata Theory*, pages 74–81. IEEE.

Booth, T. and Thompson, R. (1973). Applying probability measures to abstract languages. *Computers, IEEE Transactions on*, 100(5):442–450.

Chaudhuri, R., Pham, S., and Garcia, O. (1983). Solution of an open problem on probabilistic grammars. *Computers, IEEE Transactions on*, 100(8):748–750.

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

Chiang, D. (2005). A hierarchical phrase-based model for statistical machine translation. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 263–270. Association for Computational Linguistics.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 39:1–38.

Durbin, R., Eddy, S., Krogh, A., and Mitchison, G. (1998). *Biological Sequence Analysis*. Cambridge Universiy Press.

Etessami, K. and Yannakakis, M. (2009). Recursive markov chains, stochastic grammars, and monotone systems of nonlinear equations. *Journal of the ACM (JACM)*, 56(1):1–66.

Gass, S. (2010). *Linear Programming: Methods and Applications: Fifth Edition*. Dover Pubns.

Gécseg, F. and Steinby, M. (1984). *Tree Automata*. Akadémiai Kiadó, Budapest.

Graehl, J., Knight, K., and May, J. (2008). Training tree transducers. *Computational Linguistics*, 34(3):391–427.

Harris, T. (1963). *The Theory of Branching Processes*. Springer-Verlag Berlin.

Horn, R. and Johnson, C. (1990). *Matrix analysis*. Cambridge University Press.

Joshi, A. K., Levy, L. S., and Takahashi, M. (1975). Tree Adjunct Grammars. *J. Comput. System Sci.*, 10:136–163.

Lancaster, P. and Tismenetsky, M. (1985). *The Theory of Matrices: with Applications*. Academic Press.

Lewis, P. and Stearns, R. (1968). Syntay-directed transduction. *J. ACM*, 15(3):465–488.

Manning, C. and Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*, volume 59. MIT Press.

Matoušek, J. and Gärtner, B. (2007). *Understanding and Using Linear Programming*. Springer-Verlag Berlin.

Nederhof, M. and Satta, G. (2008). Computing partition functions of pcfgs. *Research on Language & Computation*, 6(2):139–162.

Prescher, D. (2001). *EM-basierte maschinelle Lernverfahren für natürliche Sprachen*. PhD thesis, Universität Stuttgart.

Prescher, D. (2005). A Tutorial on the Expectation-Maximization Algorithm Including Maximum-Likelihood Estimation and EM Training of Probabilisitic Context-Free Grammars. Technical report, University of Amsterdam.

Sakakibara, Y., Brown, M., Hughey, R., Mian, I., Sjölander, K., Underwood, R., and Haussler, D. (1994). Stochastic context-free grammers for trna modeling. *Nucleic acids research*, 22(23):5112.

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

Shieber, S. M. and Schabes, Y. (1990). Synchronous tree-adjoining grammars. In *Proceedings of the 13th International Conference on Computational Linguistics*, volume 3, pages 253–258, Helsinki, Finland.

Sipser, M. (1996). *Introduction to the Theory of Computation*. International Thomson Publishing.

Tarski, A. (1955). A lattice-theoretical fixpoint theorem and its applications. *Pacific journal of Mathematics*, 5(2):285–309.

Wetherell, C. (1980). Probabilistic languages: A review and some open questions. *ACM Computing Surveys (CSUR)*, 12(4):361–379.