Conjunctive languages are closed under inverse homomorphism

Alexander Okhotin
okhotin@cs.queensu.ca

Technical report 2003–468
School of Computing, Queen's University,
Kingston, Ontario, Canada K7L 3N6

August 2003

Abstract

It is proved that the inverse homomorphic image of every language generated by a conjunctive grammar can be generated by a conjunctive grammar as well, and, given an arbitrary grammar G over an alphabet Γ and a homomorphism $h : \Sigma^* \to \Gamma^*$, a grammar for the language $h^{-1}(L(G)) \subseteq \Sigma^*$ can be effectively constructed. Together with the known results on conjunctive grammars, this implies that the language family they generate is a pre-AFL.

Résumé

On montre que l'image homomorphic inverse de chaque langage produite par une grammaire conjonctive peut être aussi bien produite par une grammaire conjonctive, et, donné une grammaire arbitraire G sur d'un alphabet Γ et un homomorphisme $h : \Sigma^* \to \Gamma^*$, une grammaire pour le langage $h^{-1}(L(G)) \subseteq \Sigma^*$ peut être efficacement construit. Avec les résultats connus sur les grammaires conjonctives, ceci implique que la famille de langages qu'elles produisent est une pre-AFL.

1 Introduction

Conjunctive grammars [2] generalize context-free grammars by allowing the use of an explicit intersection operation in its rules. Besides being a convenient descriptive device, this additional operation gives a certain increase in the generative power of the model, so that every finite intersection of context-free languages and even certain languages that are known to be not representable as such intersection can be described with conjunctive grammars.

On the other hand, the complexity of string recognition [2], as well as of the general membership problem [4], is the same as in the context-free case ($O(n^3)$ and **P**-complete respectively), the membership of a string in a language can also be represented in the form of a tree [2] (in this case, with shared leaves), and most practically used context-free parsing algorithms have more general analogs applicable to conjunctive grammars [3, 4, 5]. This confirms the value of the model and gives motivation for researching its properties.

However, the theoretical properties of this family of languages are yet to be studied: even some of the basic closure properties of the family are still not known. It is obviously closed under union, intersection, concatenation and star, and is easy to prove not to be closed under homomorphism and quotient. The closure of the family under complement, ϵ-free homomorphism and inverse homomorphism so far remains an open problem.

Concerning the closure under inverse homomorphism, this property is possessed by all the classic language families in the Chomsky hierarchy [6] , and is required by all types of abstract families of languages, from *trio* and *pre-AFL* to *full AFL* [1]. In this paper we positively solve the problem of the closure of conjunctive languages under inverse homomorphism, describing an algorithmic method to construct a grammar for the inverse homomorphic image of a given conjunctive language.

Section 2 gives the definition of conjunctive grammars and describes some of their basic properties. One possible method of constructing a conjunctive grammar for inverse homomorphic image of a given language is explained in Section 3; this method is easy in terms of the computational complexity of construction and easy to explain as well. In Section 4 the construction described in Section 3 is illustrated on an example. Section 5 contains a formal proof of this construction.

An alternative construction of a grammar G', such that $L(G') = h^{-1}(G)$, for any given G and h is given in Section 6. This construction is computationally harder (it involves solving **P**-complete problems), but it yields grammars of smaller size. The correctness of this construction is obtained as a corollary of the correctness of the first method.

2 Overview of conjunctive grammars

Definition 1. *A conjunctive grammar is a quadruple $G = (\Sigma, N, P, S)$, where Σ and N are disjoint finite nonempty sets of terminal and nonterminal symbols; P is a finite set of grammar rules of the form*

$$A \to \alpha_1 \& \ldots \& \alpha_n \quad (A \in N,\ n \geqslant 1, \text{ for all } i\ \alpha_i \in (\Sigma \cup N)^*), \tag{1}$$

where the strings α_i are distinct, and their order is considered insignificant; $S \in N$ is a nonterminal designated as the start symbol.

Three additional special symbols will be used: '(', '&' and ')'; it is assumed that none of them is in $\Sigma \cup V$.

For each rule of the form (1) and for each i ($1 \leqslant i \leqslant n$), $A \to \alpha_i$ is called a conjunct. Let $conjuncts(P)$ denote the set of all conjuncts.

A conjunctive grammar generates strings by deriving them from the start symbol, generally in the same way as the context-free grammars do. Intermediate strings used in course of a derivation are actually formulae under concatenation and conjunction:

Definition 2. *Let $G = (\Sigma, N, P, S)$ be a conjunctive grammar. The set of conjunctive formulae $\mathcal{F}$ is defined inductively:*

- *The empty string ϵ is a conjunctive formula.*
- *Any symbol from $\Sigma \cup N$ is a formula.*
- *If $\mathcal{A}$ and $\mathcal{B}$ are nonempty formulae, then $\mathcal{AB}$ is a formula.*
- *If $\mathcal{A}_1, \ldots, \mathcal{A}_n$ ($n \geqslant 1$) are formulae, then $(\mathcal{A}_1 \& \ldots \& \mathcal{A}_n)$ is a formula.*

Definition 3. *Let $G = (\Sigma, N, P, S)$ be a conjunctive grammar. Define $\overset{G}{\Longrightarrow}$, the relation of one-step derivability on the set of conjunctive formulae.*

1. *Any nonterminal in any formula may be rewritten with the body of any rule for that nonterminal enclosed in parentheses. Formally, for any* $s', s'' \in (\Sigma \cup N \cup \{\text{'('}, \text{'\&'}, \text{')'}\})^*$ *and* $A \in N$*, such that* $s'As''$ *is a formula, and for all* $A \to \alpha_1 \& \ldots \& \alpha_n \in P$*,*

$$s'As'' \overset{G}{\Longrightarrow} s'(\alpha_1 \& \ldots \& \alpha_n)s'' \tag{2}$$

2. *If a formula contains a subformula of the form of a conjunction of one or more identical terminal strings enclosed in parentheses, then these multiple copies of the string can be glued into one by rewriting the subformula with the same terminal string without parentheses. Formally, for any* $s', s'' \in (\Sigma \cup N \cup \{\text{'('}, \text{'\&'}, \text{')'}\})^*$*,* $n \geqslant 1$ *and* $w \in \Sigma^*$*, such that* $s'(\underbrace{w \& \ldots \& w}_{n})s''$ *is a formula,*

$$s'(\underbrace{w \& \ldots \& w}_{n})s'' \overset{G}{\Longrightarrow} s'ws'' \tag{3}$$

Let $\overset{G}{\Longrightarrow}^*$ *denote the reflexive and transitive closure of* $\overset{G}{\Longrightarrow}$ *.*

Definition 4. *Let* $G = (\Sigma, N, P, S)$ *be a conjunctive grammar. The language of a formula is the set of all terminal strings derivable from the formula:* $L_G(\mathcal{A}) = \{w \in \Sigma^* \mid \mathcal{A} \overset{G}{\Longrightarrow}^* w\}$*. The language generated by the grammar is the language generated by its start symbol:* $L(G) = L_G(S)$*.*

The semantics of conjunctive grammars is well characterized by the following equalities [2]:

Theorem 1. *Let* $G = (\Sigma, N, P, S)$ *be a conjunctive grammar. Let* $\mathcal{A}_1, \ldots, \mathcal{A}_n, \mathcal{B}$ *be formulae, let* $A \in N$*, let* $a \in \Sigma$*. Then,*

$$L_G(\epsilon) = \{\epsilon\} \tag{4a}$$

$$L_G(a) = \{a\} \tag{4b}$$

$$L_G(A) = \bigcup_{A \to \alpha_1 \& \ldots \& \alpha_m \in P} L_G((\alpha_1 \& \ldots \& \alpha_m)) \tag{4c}$$

$$L_G(\mathcal{A}\mathcal{B}) = L_G(\mathcal{A}) \cdot L_G(\mathcal{B}) \tag{4d}$$

$$L_G((\mathcal{A}_1 \& \ldots \& \mathcal{A}_n)) = \bigcap_{i=1}^{n} L_G(\mathcal{A}_i) \tag{4e}$$

The representation of a context-free derivation in a form of a tree is inherited by conjunctive grammars. Every conjunctive derivation

$$A \Longrightarrow \ldots \Longrightarrow \mathcal{B}, \tag{5}$$

where $A \in N$ and $\mathcal{B}$ is an arbitrary formula, can be represented in the form of a tree with shared leaves. The leaves of the tree are labeled with symbols from $\Sigma \cup N \cup \{\epsilon\}$. Nonepsilon leaves correspond to terminal and nonterminal symbols from $\mathcal{A}$, and a leaf can have in-degree of more than one only if it is labeled with a terminal symbol. Internal vertices of the tree are labeled with the rules used in the derivation. For each vertex, all outgoing arcs are considered ordered.

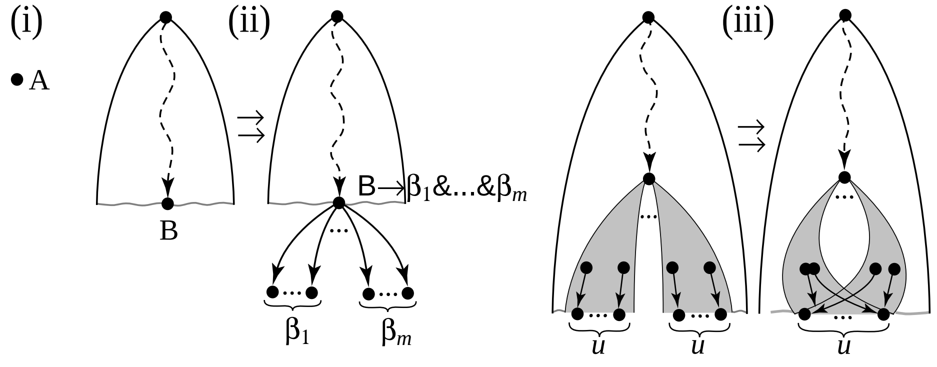

Figure 1: Construction of a derivation tree.

The derivation tree corresponding to a derivation is defined inductively on the length of derivation:

- The tree corresponding to a 0-step derivation $A \Longrightarrow^* A$ is a single node labeled with A (see Figure 1(i)).
- The tree corresponding to an $(n+1)$-step derivation of the form

  $$A \Longrightarrow \ldots \Longrightarrow s_1 B s_2 \Longrightarrow s_1(\beta_1 \& \ldots \& \beta_m) s_2 \tag{6}$$

  is made from the tree corresponding to the first n steps of the derivation (6) by relabelling the leaf corresponding to B with a rule $B \to \beta_1 \& \ldots \& \beta_m$ and linking it to the new leaves corresponding to the symbols in the rule, as in Figure 1(ii).

  If the rule contains an empty conjunct, then a single ϵ leaf is created.

- The tree corresponding to an $(n+1)$-step derivation

$$A \Longrightarrow \ldots \Longrightarrow s_1(u\& \ldots \&u)s_2 \Longrightarrow s_1 u s_2 \tag{7}$$

  is made from the tree corresponding to the first n steps of (7) by identifying for every i $(1 \leqslant i \leqslant |u|)$ the leaves corresponding to i-th characters of all words u being glued.

  This case is illustrated in Figure 1(iii).

For every language $L \subseteq \Sigma^*$, let substrings(L) denote the set of *proper nonempty substrings* of the strings from L, let prefixes(L) denote the set of *proper prefixes* of the strings from L, and let suffixes(L) denote the set of *proper suffixes* of the strings from L:

$$\text{substrings}(L) = \{w \mid w \in \Sigma^+,\ uwv \in L \text{ for some } u, v \in \Sigma^*, |uv| \geqslant 1\}, \tag{8a}$$
$$\text{prefixes}(L) = \{u \mid u \in \Sigma^*,\ uv \in L \text{ for some } v \in \Sigma^+\}, \tag{8b}$$
$$\text{suffixes}(L) = \{v \mid v \in \Sigma^*,\ uv \in L \text{ for some } u \in \Sigma^+\} \tag{8c}$$

In the following we shall use this notation only for finite languages. For instance, if $L = \{ab, abb, abbb\}$, then substrings$(L) = \{a, b, ab, bb, abb, bbb\}$, prefixes$(L) = \{\epsilon, a, ab, abb\}$ and suffixes$(L) = \{\epsilon, b, bb, bbb\}$.

3 The construction

In this section we describe and explain one possible way to construct a conjunctive grammar for the inverse homomorphic image of a language denoted by a conjunctive grammar, given the latter conjunctive grammar and the homomorphism. This construction results in a constant times blowup of the number of nonterminals (where the constant depends on the homomorphism alone) in the grammar, and can be performed in deterministic logarithmic space, which allows to use it in complexity-theoretic proofs. Later, in Section 6, this construction will be modified to create less nonterminals, but that other construction will be computationally harder and not easily explainable without referring to the more verbose construction that will now be presented.

Let $h : \Sigma^* \to \Gamma^*$ be an arbitrary homomorphism and let $G = (\Gamma, N, P, S)$ be a conjunctive grammar in the binary normal form. Our goal is to construct a conjunctive grammar $G' = (\Sigma, N', P', S')$, such that $L(G') = h^{-1}(L(G))$.

Let us partition the alphabet Σ into two disjoint classes Σ_0 and Σ_1, such that

$$\Sigma_0 = \{a \mid a \in \Sigma,\ h(a) = \epsilon\}, \tag{9a}$$
$$\Sigma_1 = \{a \mid a \in \Sigma,\ h(a) \neq \epsilon\}, \tag{9b}$$

and $\Sigma = \Sigma_0 \cup \Sigma_1$.

Define

$$images(h) = \{h(a) \mid a \in \Sigma\} \subseteq \Gamma^* \tag{10}$$

Construct the grammar $G' = (\Sigma, N', P', S')$, in which the set of nonterminals contains so-called "short" and "long" nonterminals.

$$N' = N'_s \cup N'_l \tag{11a}$$
$$N'_s = N \times \text{substrings}(\text{images}(h)) \tag{11b}$$
$$N'_l = \text{suffixes}(\text{images}(h)) \times N \times \text{prefixes}(\text{images}(h)) \tag{11c}$$

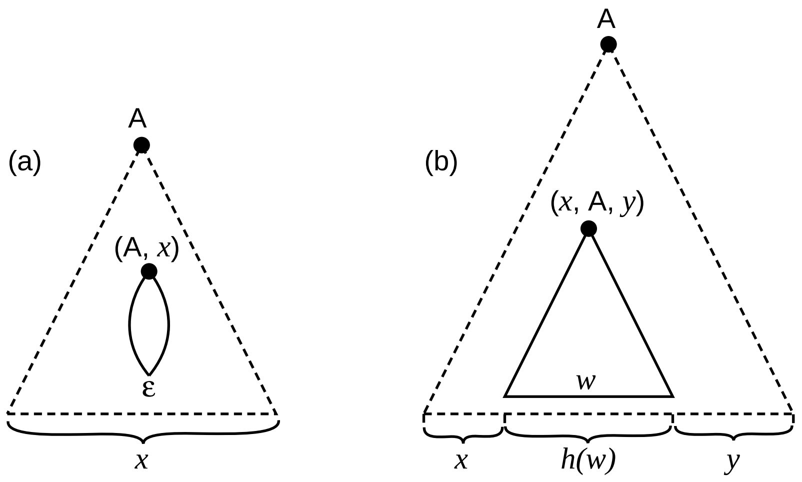

Figure 2: Two types of nonterminals: (a) "short" and (b) "long".

The goal of the construction is as follows:

- Any "short" nonterminal $(A, x) \in N'_s$ should generate the string ϵ according to the grammar G' if and only if the nonterminal $A \in N$ generates the string x according to the grammar G. Short nonterminals should not derive any other strings.

- Any "long" nonterminal $(x, A, y) \in N'_l$ should generate some string $w \in \Sigma^*$ according to the grammar G' if and only if the nonterminal $A \in N$ generates the string $x \cdot h(w) \cdot y$ according to the grammar G.

  The first and the third component of a "long" nonterminal will be called its *left margin* and *right margin* respecively.

These two types of nonterminals are illustrated in Figure 2, in which dotted lines denote derivation trees according to G (these derivations are *simulated* by G') and thick lines denote derivation trees according to G' (these derivations are actually carried out by G' in course of the simulation).

The set P' contains the following rules:

1. Creation of "short" nonterminals. For all $a \in \Sigma_1$, $t \in \Gamma$ and $A \in N$, such that t is a proper substring of $h(a)$ and $A \to t \in P$, the new grammar contains the rule

$$(A, t) \to \epsilon \tag{12}$$

This type of rule is illustrated in Figure 3(a), where a rule 12, shown

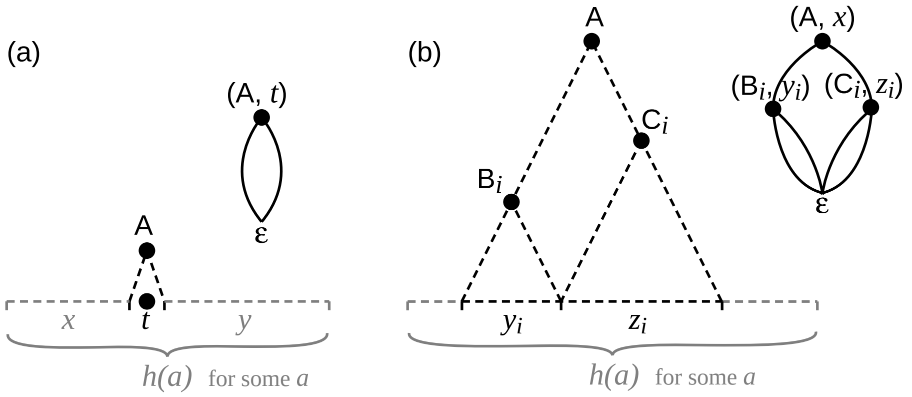

Figure 3: Operations with "short" nonterminals: (a) creation; (b) extension.

in thick lines, simulates a rule $A \to t$, shown in black dotted lines, in the intended context shown by gray dotted lines.

2. Extension of "short" nonterminals. For every rule

$$A \to B_1C_1 \& \ldots \& B_nC_n \in P \tag{13}$$

of the original grammar, for every $x \in \text{substrings}(\text{images}(h))$, such that $|x| \geqslant 2$, and for every n factorizations $x = y_1 z_1 = \ldots = y_n z_n$, where $y_i, z_i \in \Gamma^+$, the set P' contains a rule

$$(A, x) \to (B_1, y_1)(C_1, z_1)\& \ldots \&(B_n, y_n)(C_n, z_n) \tag{14}$$

Figure 3 depicts the operation of one single conjunct $A \to B_i C_i$ of the rule (13) from the grammar G, which is being simulated by the conjunct $(A, x_i) \to (B_i, y_i)(C_i, z_i)$ of the rule (14) from the grammar G'.

An application of the rule (13) in a derivation according to G includes all its n conjuncts, and is consequently simulated by an application of the rule (14) with all n of its conjuncts.

3. **Conversion of "short" nonterminals to "long".** For every $x \in \text{suffixes}(\text{images}(h))$ and $A \in N$ there is a rule

$$(x, A, \epsilon) \to (A, x) \tag{15a}$$

For every $y \in \text{prefixes}(\text{images}(h))$ and $A \in N$ there is a rule

$$(\epsilon, A, y) \to (A, y) \tag{15b}$$

The rules (15) of the grammar G' do not correspond to any derivations according to the grammar G; their function is restricted to internal data management within G' (the transition from "short" to "long" nonterminals), specific to the present construction.

4. **Processing of "long" nonterminals.** For every rule

$$A \to B_1 C_1 \& \ldots \& B_n C_n \in P \tag{16}$$

of the original grammar and for every $x \in \text{suffixes}(\text{images}(h))$ and $y \in \text{prefixes}(\text{images}(h))$, the set P' contains a rule of the form

$$(x, A, y) \to \alpha_1 \& \ldots \& \alpha_n, \tag{17}$$

where, for every i $(1 \leqslant i \leqslant n)$, $\alpha_i \in (\Sigma \cup N')^+$ if and only if one of the following is true:

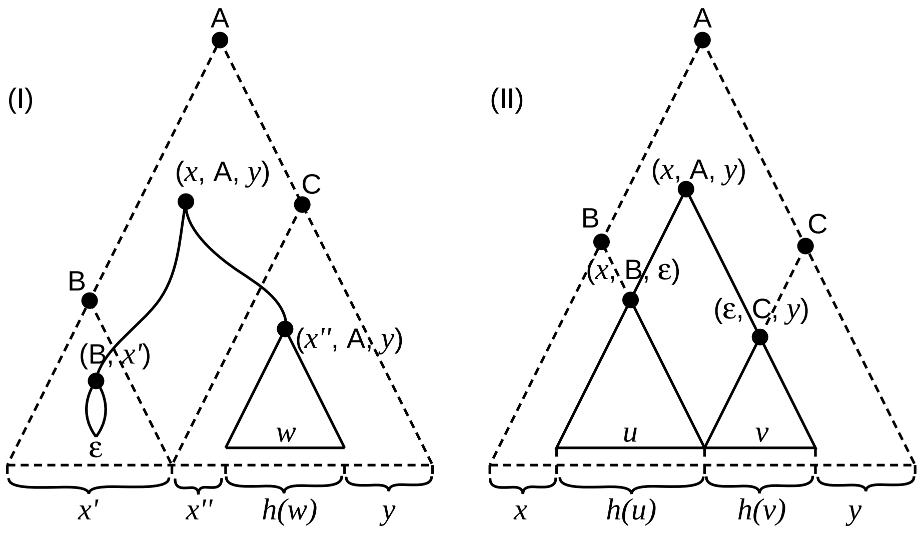

Figure 4: "Long" nonterminals: conjuncts of types I and II.

I. $\alpha_i = (B_i, x')(x'', C, y)$, where $x'x'' = x$ and $x', x'' \neq \epsilon$. Here a "short" nonterminal is concatenated to a "long" nonterminal from the left, thus increasing its left margin from x'' to $x = x'x''$, but not deriving any additional terminal symbols. This concatenation is shown in Figure 4(I).

II. $\alpha_i = (x, B_i, \epsilon)(\epsilon, C_i, y)$ (see Figure 4(II)). Here a "long" nonterminal with an empty right margin is concatenated to a "long" nonterminal with an empty left margin.

If (x, B_i, ϵ) derives a string $u \in \Sigma^*$ according to the grammar G' and thus simulates a derivation of $x \cdot h(u) \in \Gamma^*$ from B_i according to the grammar G, and if (ϵ, C_i, y) derives $v \in \Sigma^*$ and thus simulates a derivation of $h(v) \cdot y$ from C_i, then the concatenation α_i derives uv and has margins x and y, therefore simulating a derivation of $x \cdot h(uv) \cdot y$ from B_iC_i according to the grammar G.

III. $\alpha_i = (x, B_i, z_i')a(z_i'', C_i, y)$, where $a \in \Sigma$, $z_i, z_i'' \neq \epsilon$ and $h(a) = z_i'z_i''$. Here two "long" nonterminals with nonempty margins are concatenated to each other in such a way that concatenation of the right margin or the first nonterminal and the left margin of the second nonterminal forms a homomorphic image of some symbol from Σ. This symbol is then is placed between the nonterminals being concatenated, as the materialization of the joint of margins. This type of concatenation is shown in Figure 5(III).

IV. $\alpha_i = (x, B_i, z_i')(C_i, z_i'')$, where $z_i'z_i'' = y$.
Concatenation of a "short" nonterminal to a "long" nonterminal from the right. This case is a mirror image of case I and is illustrated in Figure 5(IV).

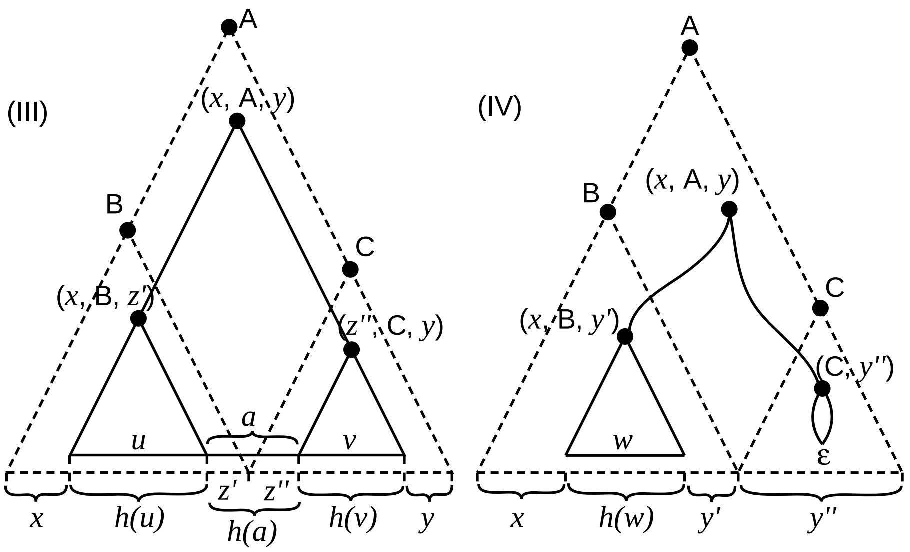

Figure 5: "Long" nonterminals: conjuncts of types (III) $A \to (x, B_i, z_i')a(z_i'', C_i, y)$ and (IV) $A \to (x, B_i, z_i')(C_i, z_i'')$.

These four possible ways of simulating a derivation from BC in the grammar G correspond to four types of factorization of a string $x \cdot h(w) \cdot y$ ($w \in \Sigma^*$) into two substrings. For every such factorization there exists a corresponding conjunct of one of these types (I, II, III or IV), which derives $w \in \Sigma^*$ according to G', at the same time simulating a derivation of $x \cdot h(w) \cdot y \in \Gamma^+$ from BC according to G.

There are many rules of the form (17) in P' corresponding to a single rule (16) from P, and each of the rules (17) corresponds to a certain set of use cases of (16). A real derivation from (x, A, y) using one of these rules would be illustrated by a juxtaposition of n instances of Figures 4 and 5, one for every conjunct in the rule (17).

5. Handling of symbols with homomorphic images of length one. For every $a \in \Sigma$, such that $h(a) = s \in \Gamma$, and for every $A \in N$, such that $A \to s \in P$, the new grammar contains a rule

$$(\epsilon, A, \epsilon) \to a \tag{18}$$

6. Handling of symbols with null homomorphic images. For every nonterminal $(x, A, y) \in N'$ and for every $b \in \Sigma_0$, there are two rules

$$(x, A, y) \to b(x, A, y) \tag{19a}$$
$$(x, A, y) \to (x, A, y)b \tag{19b}$$

These rules allow to derive terminals from Σ_0 in arbitrary quantity from any "long" nonterminal, effectively closing the language generated by (x, A, y) under left- and right-concatenation with Σ_0.

7. Handling of the empty string. If $S \to \epsilon \in P$, then P' contains the rule

$$(\epsilon, S, \epsilon) \to \epsilon \tag{20}$$

Finally, the start symbol of the new grammar is defined as $S' = (\epsilon, S, \epsilon)$.

4 Example

Before starting to prove the correctness of our construction, let us illustrate it on an example.

Let $\Gamma = \{s, t\}$ and consider the language $L = \{ts^2ts^4t \ldots ts^{2n-2}ts^{2n}t \,|\, n \geqslant 0\} = \{t, tsst, tsstsssst, tsstsssstsssssst, \ldots\}$. Following is one of the possible ways to construct a conjunctive grammar for this language:

$$\begin{aligned} S &\to Mt\&SC \mid t \\ M &\to tMss \mid sM \mid tss \\ C &\to sC \mid t \end{aligned} \tag{21}$$

After conversion to the binary normal form, the grammar (21) becomes

$$\begin{aligned}
&S \to MB \& SC \mid t \\
&M \to ND \mid AM \mid BD \\
&N \to BM \\
&A \to s \\
&B \to t \\
&C \to AC \mid t \\
&D \to AA
\end{aligned} \tag{22}$$

Now let $\Sigma = \{a, b, c, d\}$ and define a homomorphism $h : \Sigma^* \to \Gamma^*$ as $h(a) = tss$, $h(b) = t$, $h(c) = ss$ and $h(d) = \epsilon$. Then

$$\text{images}(h) = \{tss, t, ss, \epsilon\}, \tag{23a}$$
$$\text{prefixes}(\text{images}(h)) = \{\epsilon, s, t, ts\}, \tag{23b}$$
$$\text{suffixes}(\text{images}(h)) = \{\epsilon, s, t, ss\}, \tag{23c}$$
$$\text{substrings}(\text{images}(h)) = \{t, s, ts, ss\} \tag{23d}$$

Let $G' = (\Sigma, N', P', S')$ be the grammar for $h^{-1}(L)$ constructed by the method of Section 3.

The set of nonterminals contains $|N| \cdot |\text{substrings}(\text{images}(h))| = 7 \cdot 4 = 28$ "short" nonterminals of the form (A, x) and $|\text{suffixes}(\text{images}(h))| \cdot |N| \cdot |\text{prefixes}(\text{images}(h))| = 4 \cdot 7 \cdot 4 = 112$ "long" nonterminals of the form (x, A, y). The full list of rules in P' is omitted due to space considerations; below we list a subset of these rules, which are used in a given sample derivation.

Consider the string $adb \in \Sigma^*$. $h(adb) = tss \cdot \epsilon \cdot t = tsst \in L$, and therefore $adb \in h^{-1}(L)$. The derivation tree of the string $tsst$ according to the grammar G is given in Figure 6(a), and one of the derivation trees of adb according to G' is given in Figure 6(b). Let us list the rule used in this derivation of adb:

- $(A, s) \to \epsilon$, $(B, t) \to \epsilon$ and $(S, t) \to \epsilon$ are rules of type 1 (creation of "short" nonterminals).
- $(D, ss) \to (A, s)(A, s)$ is a rule of type 2 (extension of "short" nonterminals).
- The rules $(\epsilon, B, t) \to (B, t)$, $(ss, D, \epsilon) \to (D, ss)$ and $(\epsilon, S, t) \to (S, t)$ are rules of type 3 (conversion of "short" nonterminals to "long").

- The rule $(\epsilon, S, \epsilon) \to (\epsilon, M, \epsilon)(\epsilon, B, \epsilon)\&(\epsilon, S, t)a(ss, C, \epsilon)$ is a rule of type 4 (processing of "long" nonterminals) made out of the rule $S \to MB\&SC \in P$: the first conjunct is of type II according, while the second is of type III.

  There are also three one-conjunct rules of type 4: the rule $(\epsilon, M, \epsilon) \to (\epsilon, B, t)a(ss, D, \epsilon)$ is made out of $M \to BD \in P$ and consists of a single conjunct of type III; the rules $(ss, C, \epsilon) \to (A, s)(s, C, \epsilon)$ and $(s, C, \epsilon) \to (A, s)(\epsilon, C, \epsilon)$ are both made out of $C \to AC \in P$ and each of them consists of a single type I conjunct.

- The rules $(\epsilon, B, \epsilon) \to b$ and $(\epsilon, C, \epsilon) \to b$ are of type 5 (handling of symbols with homomorphic images of length one), and are made out of the rules $B \to t \in P$ and $C \to t \in P$, because $h(b) = t$.

- The rules $(\epsilon, M, \epsilon) \to (\epsilon, M, \epsilon)d$ and $(ss, C, \epsilon) \to d(ss, C, \epsilon)$ are rules of type 6 (handling of symbols with null homomorphic images) and are in P' because $h(d) = \epsilon$.

Let us see that the derivation of adb illustrated in Figure 6(b) actually *simulates* the derivation of $h(adb) = tsst$ illustrated in Figure 6(a). In order to emphasize this simulation, the components of the derivation trees corresponding to each other are printed in black, while the elements specific to each tree are shown in gray. There are 13 black vertices in each of the two trees in Figure 6, and their relation to each other is basically the same in both trees.

For instance, the root vertex $S \to MB\&SC$ of the first tree is simulated by the root vertex

$$(\epsilon, S, \epsilon) \to (\epsilon, M, \epsilon)(\epsilon, B, \epsilon)\&(\epsilon, S, t)a(ss, C, \epsilon) \tag{24}$$

of the second tree; the vertex

$$(\epsilon, M, \epsilon) \to (\epsilon, B, t)a(ss, D, \epsilon) \tag{25}$$

of the second tree corresponds to the vertex $M \to BD$ of the first tree, while the immediate descendant $(\epsilon, M, \epsilon) \to (\epsilon, M, \epsilon)d$ of the root of the second tree (shown in gray) does the work specific to the grammar G' (generating the symbol with an empty homomorphic image), which does not have an analog in the grammar G. Similarly, the immediate descendant of $M \to BD$ labeled

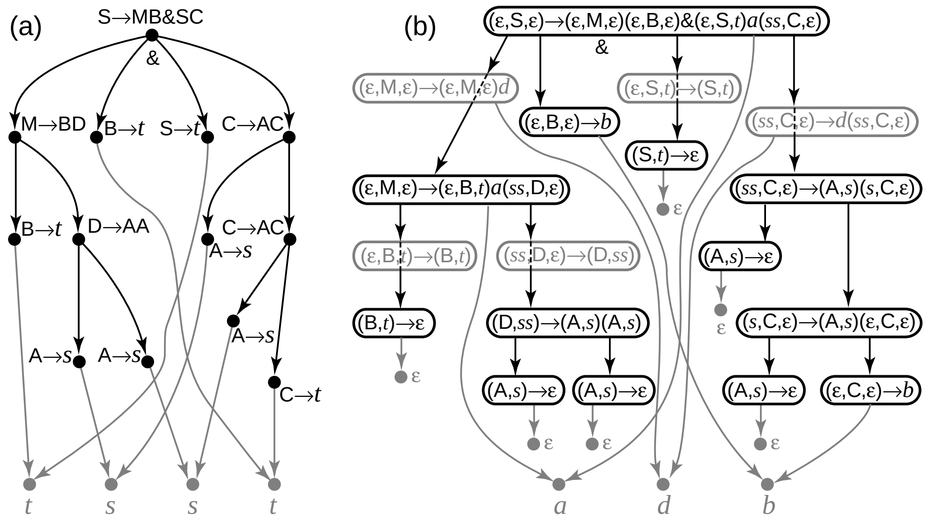

Figure 6: (a) A derivation tree of $tsst \in L$; (b) a derivation tree of $adb \in h^{-1}(L)$, which simulates the derivation of $h(adb) = tsst \in L$.

with $B \to t$ (on the left) is simulated by the vertex $(B, t) \to \epsilon$ (also on the left), which is not an immediate descendant of the vertex (25), while the immediate descendant of (25), $(\epsilon, B, t) \to (B, t)$ (on the left; shown in gray), represents internal data management within G' and does not correspond to any actions in the derivation of $tsst \in L(G)$.

5 The proof of the construction

Observation 1 (Languages generated by "short" nonterminals). *For any* $x \in \text{substrings}(\text{images}(h)) \subseteq \Gamma^+$, $L_{G'}(A, x) \subseteq \{\epsilon\}$.

Lemma 1 (Correctness of "short" nonterminals). *For any string* $x \in \text{substrings}(\text{images}(h))$, $(A, x) \overset{G'}{\Longrightarrow}{}^* \epsilon$ *if and only if* $A \overset{G}{\Longrightarrow}{}^* x$.

Proof. Let us first note that every rule for (A, x) is of the form (12) if $|x| = 1$, and of the form (14) if $|x| \geqslant 2$.

The proof is an induction on the length of x.

Basis $|x| = 1$. Let $x = b \in \Gamma$. Then every rule for (A, x) is of the form (12).

$(A, b) \overset{G'}{\Longrightarrow}^* \epsilon$ if and only if $(A, b) \to \epsilon \in P'$, which, by the construction of G', holds if and only if $A \to b \in P$, which is in turn equivalent to $A \overset{G}{\Longrightarrow}^* b$, since G is in the binary normal form.

Induction step. $(A, x) \overset{G'}{\Longrightarrow} \epsilon$ if and only if there exists a rule of the form

$$(A, x) \to (B_1, y_1)(C_1, z_1)\& \ldots \&(B_n, y_n)(C_n, z_n) \in P', \quad \text{s.t.} \tag{26a}$$

$$(B_i, y_i) \overset{G'}{\Longrightarrow}^* \epsilon \quad (\text{for all } i\text{: } 1 \leqslant i \leqslant n), \tag{26b}$$

$$(C_i, z_i) \overset{G'}{\Longrightarrow}^* \epsilon \quad (\text{for all } i\text{: } 1 \leqslant i \leqslant n) \tag{26c}$$

By the construction of G', a rule of the form (26a) is in P' if and only if $A \to B_1C_1\& \ldots \&B_nC_n \in P$, $y_i, z_i \in \Gamma^+$ and $y_iz_i = x$ for all i. By the induction hypothesis, $(B_i, y_i) \overset{G'}{\Longrightarrow}^* \epsilon$ if and only if $B_i \overset{G}{\Longrightarrow}^* y_i$, and $(C_i, z_i) \overset{G'}{\Longrightarrow}^* \epsilon$ if and only if $C_i \overset{G}{\Longrightarrow}^* z_i$.

Therefore, (26) is equivalent to the existence of a rule

$$A \to B_1C_1\& \ldots \&B_nC_n \in P \tag{27a}$$

and of a factorization

$$x = y_1z_1 = \ldots = y_nz_n, \quad \text{such that} \tag{27b}$$

$$B_i \overset{G}{\Longrightarrow}^* y_i \quad (\text{for all } i), \tag{27c}$$

$$C_i \overset{G}{\Longrightarrow}^* z_i \quad (\text{for all } i) \tag{27d}$$

Now (26) holds if and only if $A \overset{G}{\Longrightarrow}^* x$, which completes the proof. $\square$

Lemma 2 (Correctness of "long" nonterminals). *For any string $w \in \Sigma^*$, $(x, A, y) \overset{G'}{\Longrightarrow}^* w$ if and only if $A \overset{G}{\Longrightarrow}^* x \cdot h(w) \cdot y$.*

Proof. $\textcircled{\Rightarrow}$ The proof is an induction on the length l of the derivation

$$(x, A, y) \overset{G'}{\Longrightarrow} \ldots \overset{G'}{\Longrightarrow} w \tag{28}$$

Depending on the type of rule applied to (x, A, y) at the first step of the derivation (28), there are five cases to consider; let us enumerate these cases from 3 to 7, in correspondence to the types of rules defined in Section 3.

3. Conversion of "short" nonterminals to "long". Consider a rule $(x, A, \epsilon) \to (A, x)$ of type 3, and let the derivation (28) be of the form

$$(x, A, \epsilon) \overset{G'}{\Longrightarrow} \big((A, x)\big) \overset{G'}{\Longrightarrow} \ldots \overset{G'}{\Longrightarrow} w \tag{29}$$

Then $w = \epsilon$ by Observation 1, $y = \epsilon$, and thus $x \cdot h(w) \cdot y = x$. By Lemma 1, $(A, x) \overset{G'}{\Longrightarrow}{}^* \epsilon$ implies $A \overset{G}{\Longrightarrow}{}^* x$, which proves this case.

The case of rules of the form $(\epsilon, A, y) \to (A, y)$ is handled similarly.

4. Processing of "long" nonterminals. Let the first rule in (28) be of type 4 – i.e., of the form $(x, A, y) \to \alpha_1 \& \ldots \& \alpha_n$, where there is a rule $A \to B_1C_1 \& \ldots \& B_nC_n \in P$, such that for every i ($1 \leqslant i \leqslant n$) the string α_i is of the form specified in the construction in Section 3. Then the derivation is of the form

$$(x, A, y) \overset{G'}{\Longrightarrow} \big(\alpha_1 \& \ldots \& \alpha_n\big) \overset{G'}{\Longrightarrow} \ldots \overset{G'}{\Longrightarrow} w, \tag{30}$$

and consequently each α_i derives w in less than l steps.

Now it suffices to prove that $\alpha_i \overset{G'}{\Longrightarrow}{}^* w$ implies $B_iC_i \overset{G}{\Longrightarrow}{}^* x \cdot h(w) \cdot y$ for all i. By the construction of the grammar G', for each i there are four cases to consider.

I. $\alpha_i = (B_i, x')(x'', C, y)$, where $x'x'' = x$ and $x', x'' \neq \epsilon$.

By Lemma 1,

$$B_i \overset{G}{\Longrightarrow}{}^* x' \tag{31}$$

The nonterminal (x'', C, y) derives w in less than l steps and thus

$$C_i \overset{G'}{\Longrightarrow}{}^* x'' \cdot h(w) \cdot y \tag{32}$$

by the induction hypothesis. Now (31) and (32) are easily combined to obtain a derivation of $x'x'' \cdot h(w) \cdot y$ out of B_iC_i.

II. $\alpha_i = (x, B_i, \epsilon)(\epsilon, C_i, y)$. If α_i derives w, then there exists a factorization $w = uv$, such that

$$(x, B_i, \epsilon) \overset{G'}{\Longrightarrow}{}^* u \tag{33a}$$

$$(\epsilon, C_i, y) \overset{G'}{\Longrightarrow}{}^* v \tag{33b}$$

By the induction hypothesis, this implies that $B_i \overset{G}{\Longrightarrow}{}^* x \cdot h(u)$ and $C_i \overset{G}{\Longrightarrow}{}^* h(v) \cdot y$, and thus $B_iC_i \overset{x}{\Longrightarrow}{}^* \cdot h(u) \cdot h(v) \cdot y = x \cdot h(w) \cdot y$.

III. $\alpha_i = (x, B_i, z'_i)a(z''_i, C_i, y)$, where $a \in \Sigma$, $z_i, z''_i \neq \epsilon$ and $h(a) = z'_i z''_i$. There exists a factorization $w = uav$, such that

$$(x, B_i, z') \overset{G'}{\Longrightarrow}^* u \tag{34a}$$

$$(z'', C_i, y) \overset{G'}{\Longrightarrow}^* v \tag{34b}$$

By the induction hypothesis, $B_i \overset{G}{\Longrightarrow}^* x \cdot h(u) \cdot z'$ and $C_i \overset{G}{\Longrightarrow}^* z'' \cdot h(v) \cdot y$, and we can derive $B_i C_i \overset{G}{\Longrightarrow} x \cdot h(u) \cdot z' \cdot z'' \cdot h(v) \cdot y$, where

$$x \cdot h(u) \cdot z' \cdot z'' \cdot h(v) \cdot y = x \cdot h(u) \cdot h(a) \cdot h(v) \cdot y = x \cdot h(w) \cdot y \tag{35}$$

IV. $\alpha_i = (x, B_i, z'_i)(C_i, z''_i)$, where $z'_i z''_i = y$. This case is treated similarly to case I.

5. **Handling of symbols with homomorphic images of length one.** If the first rule of the derivation is of the form $(\epsilon, A, \epsilon) \to a$, where $a \in \Sigma$, $h(a) = s \in \Gamma$ and $A \to s \in P$, then $A \overset{G}{\Longrightarrow}^* \epsilon \cdot h(a) \cdot \epsilon = s$ using the rule $A \to s$.

6. **Handling of symbols with null homomorphic images.** Let the derivation start from the application of the rule $(x, A, y) \to b(x, A, y)$. Then $w = bu$ and $(x, A, y) \overset{G'}{\Longrightarrow}^* u$ in less than l steps. By the induction hypothesis,

$$A \overset{G}{\Longrightarrow}^* x \cdot h(u) \cdot y, \tag{36}$$

and thus, since $h(w) = h(bu) = h(b) \cdot h(u) = h(u)$, $A \overset{G}{\Longrightarrow}^* x \cdot h(w) \cdot y$.

The case of a rule of the form $(x, A, y) \to (x, A, y)b$ is proved similarly.

7. **Handling of the empty string.** If the first rule in (28) is $(\epsilon, S, \epsilon) \to \epsilon$, then, by the construction of G', $S \to \epsilon \in P$, and therefore

$$S \overset{G'}{\Longrightarrow}^* \epsilon = \epsilon \cdot h(\epsilon) \cdot \epsilon \tag{37}$$

$\Leftarrow$ Let $A \overset{G}{\Longrightarrow}^* x \cdot h(w) \cdot y$ for some $A \in N$, $w \in \Sigma^*$, $x \in$ suffixes(images(h)) and $y \in$ prefixes(images(h)). By induction on $|x \cdot h(w) \cdot y|$, let us prove that $(x, A, y) \overset{G'}{\Longrightarrow}^* w$.

Consider some factorization $w = w_0' w_1 w_0''$, such that $w_0', w_0'' \in \Sigma_0^*$ and w_1 is neither in $\Sigma_0 \Sigma^*$ nor in $\Sigma^* \Sigma_0$ (note that this factorization is unique unless $h(w) = \epsilon$). The proof is an induction on the pair

$$(|x \cdot h(w) \cdot y|, |w_0' w_0''|) \tag{38}$$

using the lexicographical order (i.e., $(i_1, j_1) < (i_2, j_2)$ iff $i_1 < i_2$ or $i_1 = i_2$ and $j_1 < j_2$).

Basis $(0, 0)$**.** Since $|x \cdot h(w) \cdot y| = 0$, it follows that $x = \epsilon$, $h(w) = \epsilon$, $y = \epsilon$ and

$$A \overset{G}{\Longrightarrow}^* \epsilon \tag{39}$$

On the other hand, the emptiness of $h(w)$ means that $w \in \Sigma_0^*$. Since the second component of the pair (38) is zero, $w = \epsilon$.

Since G is in the binary normal form, (39) implies that $A = S$ and there is a rule $S \to \epsilon$ in P. Thus, by the construction of the grammar G', the rule $(\epsilon, S, \epsilon) \to \epsilon$ is in P'.

Basis $(1, 0)$**.** If $|x \cdot h(w) \cdot y| = s \in \Gamma$ and $A \overset{G}{\Longrightarrow}^* s$, then there is a rule $A \to s \in P$. There are three cases to consider:

- **Case I.** $x = s$, $h(w) = \epsilon$, $y = \epsilon$. Since the second component of (38) is 0, the emptiness of $h(w)$ implies $w = \epsilon$.

  Since $s \in$ prefixes(images(h)) and $s \notin \epsilon$, it is clear that $s \in$ substrings(images(h)). Therefore, there exists a nonterminal $(A, s) \in N'$, and, by Lemma 1,

  $$(A, s) \overset{G'}{\Longrightarrow}^* \epsilon \tag{40}$$

  By the construction of the grammar G', there is a rule $(s, A, \epsilon) \to (A, s)$ in P', and thus

  $$(s, A, \epsilon) \overset{G'}{\Longrightarrow} \big((A, s)\big) \overset{G'}{\Longrightarrow} \ldots \overset{G'}{\Longrightarrow} (\epsilon) \overset{G'}{\Longrightarrow} \epsilon, \tag{41}$$

  which proves Case I.

- **Case II.** $x = \epsilon$, $h(w) = s$, $y = \epsilon$. Then $w = a \in \Sigma_1$, such that $h(a) = s$. Then, by the construction of G', there exists a rule $(\epsilon, A, \epsilon) \to a$, which can be used to derive a from (ϵ, A, ϵ) in one step, thus proving Case II.

- **Case III.** $x = \epsilon$, $h(w) = \epsilon$, $y = s$. This case is handled similarly to Case I.

Induction step $\{(l, k) \mid l < n,\ k \geqslant 0\} \to (n, 0)$ **(**$n \geqslant 2$**).** Let $|x \cdot h(w) \cdot y| = n \geqslant 2$. Then there exists a rule of the original grammar $A \to B_1C_1\& \dots \&B_nC_n \in P$, such that

$$B_iC_i \overset{G}{\Longrightarrow}{}^* x \cdot h(w) \cdot y \quad \text{(for every } i\text{, such that } 1 \leqslant i \leqslant n) \tag{42}$$

Fix an arbitrary i and consider the i-th conjunct $A \to B_iC_i$ of the rule. By (42) and Theorem 1, there exists a factorization

$$x \cdot h(w) \cdot y = z_i' \cdot z_i'' \quad (z_i', z_i'' \in \Gamma^+), \quad \text{such that} \tag{43a}$$

$$B_i \overset{G}{\Longrightarrow}{}^* z_i' \quad \text{and} \tag{43b}$$

$$C_i \overset{G}{\Longrightarrow}{}^* z_i'' \tag{43c}$$

The boundary between z_i' and z_i'' can fall anywhere within the string

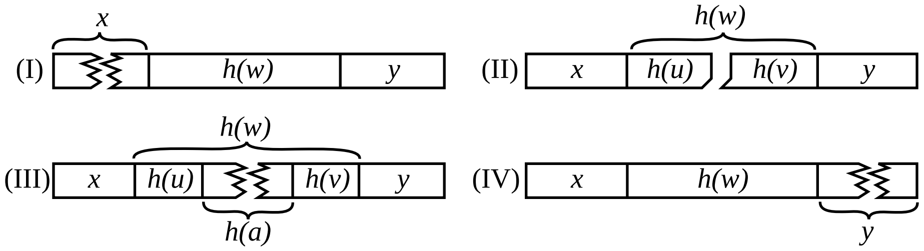

Figure 7: Possible factorizations of the string $x \cdot h(w) \cdot y$.

$x \cdot h(w) \cdot y$; the four possibilities are shown in Figure 7(I, II, III, IV). Let us consider each of these cases:

I. $z_i' = x'$ and $z_i'' = x'' \cdot h(w) \cdot y$, where $x'x'' = x$, $x' \in \Gamma^+$ and $x'' \in \Gamma^*$. By Lemma 1, (43b) is equivalent to

$$(B_i, x') \overset{G'}{\Longrightarrow} \dots \overset{G'}{\Longrightarrow} \epsilon \tag{44}$$

The string $x'' \cdot h(w) \cdot y$ is shorter than $x \cdot h(w) \cdot y$, and hence we can invoke the induction hypothesis to obtain that (43c) implies

$$(x'', C_i, y) \overset{G'}{\Longrightarrow} \dots \overset{G'}{\Longrightarrow} w \tag{45}$$

Now (44) can be combined with (45) to obtain a derivation of w from $(B_i, x') \cdot (x'', C_i, u)$, similar to what was illustrated in the earlier Figure 4(I).

Define $\alpha_i = (B_i, x')(x'', C_i, y)$.

II. $z'_i = x \cdot h(u)$ and $z''_i = h(v) \cdot y$, where $w = uv$.

We apply the induction hypothesis twice – for (43b) and for (43c) – to get

$$(x, B_i, \epsilon) \overset{G'}{\Longrightarrow} \dots \overset{G'}{\Longrightarrow} u \tag{46a}$$

$$(\epsilon, C_i, y) \overset{G'}{\Longrightarrow} \dots \overset{G'}{\Longrightarrow} v \tag{46b}$$

and consequently derive $w = uv$ from the concatenation $(x, B_i, \epsilon) \cdot (\epsilon, C_i, y)$ (see the earlier Figure 4(II)).

Define $\alpha_i = (x, B_i, \epsilon)(\epsilon, C_i, y)$.

III. $z'_i = x \cdot h(u) \cdot x'$ and $z''_i = y' \cdot h(v) \cdot y$, where $w = uav$ ($a \in \Sigma$) and $h(x'y') = a$.

As in the previous case, the induction hypothesis is used for (43b) and (43c), resulting in

$$(x, B_i, x') \overset{G'}{\Longrightarrow} \dots \overset{G'}{\Longrightarrow} u \tag{47a}$$

$$(y', C_i, y) \overset{G'}{\Longrightarrow} \dots \overset{G'}{\Longrightarrow} v \tag{47b}$$

Now the concatenation $(x, B_i, x') \cdot a \cdot (y', C_i, y)$ is easily seen to derive the string $uav = w$.

Define $\alpha_i = (x, B_i, x')a(y', C_i, y)$.

IV. $z'_i = x \cdot h(w) \cdot y'$ and $z''_i = y''$, where $y'y'' = y$, $y' \in \Gamma^*$ and $y'' \in \Gamma^+$. This case is handled similarly to Case I and is illustrated in the earlier Figure 5(IV). Define $\alpha_i = (x, B_i, y')(C_i, y'')$.

Now, by the construction of the grammar G', there exists a rule

$$(x, A, y) \to \alpha_1 \& \dots \& \alpha_n \in P', \tag{48}$$

which can be used to derive w from (x, A, y).

Induction step $(n, k-1) \to (n, k)$ $(n \geqslant 0, k > 0)$. If $|w_0' w_0''| > 0$, then either $w = bw'$ $(b \in \Sigma_0, w' \in \Sigma^*)$, or $w = w'b$ $(b \in \Sigma_0, w' \in \Sigma^*)$, or both.

Assume, without loss of generality, that $w = bw'$ (the case $w = w'b$ is handled in exactly the same way). The string w' has $k-1$ lateral symbols.

We know that $A \overset{G}{\Longrightarrow}^* x \cdot h(w) \cdot y$, and it is clear that $h(w) = h(bw') = h(w')$, and therefore

$$A \overset{G}{\Longrightarrow}^* x \cdot h(w') \cdot y \tag{49}$$

We can now apply the induction hypothesis to x, y and w' to obtain that (49) implies

$$(x, A, y) \overset{G'}{\Longrightarrow}^* w' \tag{50}$$

By the construction of G', there is a rule $(x, A, y) \to b(x, A, y)$ in P', and thus one can construct a derivation

$$(x, A, y) \overset{G'}{\Longrightarrow} (b(x, A, y)) \overset{G'}{\Longrightarrow} \ldots \overset{G'}{\Longrightarrow} (bw') \overset{G'}{\Longrightarrow} bw' = w, \tag{51}$$

which proves this last case. □

6 A more succinct construction

The construction given in Section 5 produces grammars of size bounded by constant times the size of the original grammar; this constant, however, is excessively large, because many superfluous nonterminals are being created. For the purpose of practical construction it is important to reduce this constant.

Since every "short" nonterminal generates either $\emptyset$ of $\{\epsilon\}$, it is possible to remove the rules that refer to the "short" nonterminals that generate $\emptyset$, omit all the instances of the "short" nonterminals generating $\{\epsilon\}$ in the rules, and then remove all the short nonterminals from the grammar.

In this section we present an alternative construction, which does not utilize "short" nonterminals at all, and instead several times solves the membership problem for the original grammar. This simpler construction is, however, computationally harder than the construction given in Section 3, because the membership problem for conjunctive grammars is known to be

P-complete, and is easily seen to remain **P**-complete even when the string being recognized is fixed.

Construct the grammar $G' = (\Sigma, N', P', S')$, in which the set of nonterminals is defined as

$$N' = \text{suffixes}(\text{images}(h)) \times N \times \text{prefixes}(\text{images}(h)), \tag{52}$$

and the set P' contains the following rules:

3. Creation of nonterminals. For every $A \in N$ and $x \in \text{suffixes}(\text{images}(h))$, such that $A \overset{G}{\Longrightarrow}{}^* x$, there is a rule

$$(x, A, \epsilon) \to \epsilon \tag{53a}$$

For every $A \in N$ and $y \in \text{prefixes}(\text{images}(h))$, such that $A \overset{G}{\Longrightarrow}{}^* y$, there is a rule

$$(\epsilon, A, y) \to \epsilon \tag{53b}$$

4. Processing of nonterminals. For every rule

$$A \to B_1C_1\& \dots \&B_nC_n \in P \tag{54}$$

of the original grammar and for every $x \in \text{suffixes}(\text{images}(h))$ and $y \in \text{prefixes}(\text{images}(h))$, the set P' contains a rule of the form

$$(x, A, y) \to \alpha_1\& \dots \&\alpha_n, \tag{55}$$

where, for every i ($1 \leqslant i \leqslant n$), $\alpha_i \in (\Sigma \cup N')^+$ if and only if one of the following is true:

I. $\alpha_i = (x'', C, y)$, where $x'x'' = x$, $x', x'' \neq \epsilon$ and $B_i \overset{G}{\Longrightarrow}{}^* x'$.

II. $\alpha_i = (x, B_i, \epsilon)(\epsilon, C_i, y)$.

III. $\alpha_i = (x, B_i, z_i')a(z_i'', C_i, y)$, where $a \in \Sigma$, $z_i, z_i'' \neq \epsilon$ and $h(a) = z_i'z_i''$.

IV. $\alpha_i = (x, B_i, y')$, where $y'y'' = y$, $y', y'' \neq \epsilon$ and $C_i \overset{G}{\Longrightarrow}{}^* y''$.

5. Handling of symbols with homomorphic images of length one. For every $a \in \Sigma$, such that $h(a) = s \in \Gamma$, and for every $A \in N$, such that $A \to s \in P$, the new grammar contains a rule

$$(\epsilon, A, \epsilon) \to a \tag{56}$$

6. Handling of symbols with null homomorphic images. For every nonterminal $(x, A, y) \in N'$ and for every $b \in \Sigma_0$, there are two rules

$$(x, A, y) \to b(x, A, y) \tag{57a}$$

$$(x, A, y) \to (x, A, y)b \tag{57b}$$

7. Handling of the empty string. If $S \to \epsilon \in P$, then P' contains the rule

$$(\epsilon, S, \epsilon) \to \epsilon \tag{58}$$

Finally, the start symbol of the new grammar is defined as $S' = (\epsilon, S, \epsilon)$.

By Lemma 1, this grammar is easily seen to be equivalent to the grammar constructed in Section 3.

7 Conclusion

It was proved that the family of languages generated by conjunctive grammars is closed under inverse homomorphism.

One obvious theoretical implication of this result is that it allows to characterize conjunctive languages in terms of abstract families of languages. A language family is called a *pre-AFL* [6] if it is closed under intersection with regular languages, union with $\{\epsilon\}$, concatenation through a center marker, transitive closure of concatenation through a center marker and inverse homomorphism. Since the family generated by conjunctive grammars is easily seen to be closed under general intersection, union, concatenation and transitive closure of concatenation [2], the closure of conjunctive languages under inverse homomorphism proved in this paper allows to conclude that this family is a pre-AFL.

From a more practical point of view, this effective closure means that a certain class of languages can be generated by conjunctive grammars, and a grammar for each of these languages can be relatively easily constructed.

References

[1] A. Mateescu and A. Salomaa, "Aspects of Classical Language Theory", in *Handbook of Formal Languages*, Vol. 1, 175–251, Springer-Verlag, Berlin, 1997.

[2] A. Okhotin, “Conjunctive grammars”, *Journal of Automata, Languages and Combinatorics*, 6:4 (2001), 519–535.

[3] A. Okhotin, “Top-down parsing of conjunctive languages”, *Grammars*, 5:1 (2002), 21–40.

[4] A. Okhotin, “A recognition and parsing algorithm for arbitrary conjunctive grammars”, *Theoretical Computer Science*, 302 (2003), 365–399.

[5] A. Okhotin, “LR parsing for conjunctive grammars”, *Grammars* 5:2 (2002), 81–124.

[6] A. Salomaa, *Formal Languages*, Academic Press, 1973.