Mikhail Barash | Alexander Okhotin

Defining contexts in context-free grammars

Turku Centre *for* Computer Science

TUCS Technical Report
No 1025, May 2012

Defining contexts in context-free grammars

Mikhail Barash
mikbar@utu.fi
Department of Mathematics, University of Turku, *and*
Turku Centre for Computer Science
Turku FI–20014, Finland

Alexander Okhotin
alexander.okhotin@utu.fi
Department of Mathematics, University of Turku, *and*
Turku Centre for Computer Science
Turku FI–20014, Finland

TUCS Technical Report
No 1025, May 2012

Abstract

Conjunctive grammars (Okhotin, 2001) are an extension of the standard context-free grammars with a conjunction operation, which maintains most of their practical properties, including many parsing algorithms. This paper introduces a further extension to the model, which is equipped with quantifiers for referring to the left context, in which the substring being defined does occur. For example, a rule $A \to a \,\&\, \lhd B$ defines a string a, as long as it is preceded by any string defined by B. The paper gives two equivalent definitions of the model—by logical deduction and by language equations—and establishes its basic properties, including a transformation to a normal form, a cubic-time parsing algorithm, and another recognition algorithm working in linear space.

Keywords: context-free grammars, conjunctive grammars, contexts, context-sensitive grammars, parsing.

TUCS Laboratory
Discrete Mathematics for Information Technology

1 Introduction

Context-free grammars are a logic for defining the syntax of languages. In this logic, the definitions are inductive, so that the properties of a string are determined by the properties of its substrings. This is how a rule $S \to aSb$ asserts, that if a string $a^{n-1}b^{n-1}$ has the property S, then the string a^nb^n has the property S as well. Besides the concatenation, the formalism of this logic has an implicit disjunction operation, represented by having multiple rules for a single symbol. This logic can be further augmented with conjunction and negation operations, which was done by the second author [11, 13] in *conjunctive grammars* and *Boolean grammars*, respectively. These grammars preserve the main idea of the context-free grammars—that of defining syntax inductively, as described above—maintain most of their practically important features, such as efficient parsing algorithms [13, 15, 16, 18], and have been a subject of diverse research [1, 4, 8, 9, 10, 19, 20]. As the applicability of a rule of a Boolean grammar to a substring is independent of the context, in which the substring occurs, Boolean grammars constitute a natural general case of context-free grammars. Standard context-free grammars can be viewed as their disjunctive fragment.

When Chomsky [2] introduced the term "context-free grammar" for an intuitively obvious model of syntax, he had a further idea of a more powerful model, in which one could define rules applicable only in some particular contexts. However, Chomsky's attempt to formalize his idea using the tools available at the time (namely, string-rewriting systems) led to nothing but space-bounded nondeterministic Turing machines. Even though the resulting devices are still known under the name of "context-sensitive grammars", they have nothing to do with the syntax of languages, and, in particular, fail to implement Chomsky's original idea of a phrase-structure rule applicable in a context.

This paper undertakes to reconsider Chomsky's [2] idea of contexts in grammars, this time using the appropriate tools of deduction systems and language equations, and drawing from the experience of developing the conjunctive grammars. The model proposed in this paper are *grammars with one-sided contexts*, which introduce two special quantifiers for representing left contexts of a string. The first quantifier refers to the "past" of the current substring: an expression $\lhd\alpha$ defines any substring that is directly preceded by a prefix of the form α. This quantifier is meant to be used along with usual, unquantified specifications of the structure of the current substring, using conjunction to combine several specifications. For example, consider the rule $A \to BC \,\&\, \lhd D$, which represents any substring of the form BC preceded by a substring of the form D. If the grammar contains additional rules $B \to b$, $C \to c$ and $D \to d$, then the above rule for A specifies that a substring bc of a string $w = dbc\ldots$ has the property A; however, this rule will not produce the same substring occurring in the strings $w' = abc$ or

$w'' = adbc$. The other quantifier, $\leqslant\alpha$, represents the form of the current substring together with its left context, so that the rules $A \to B \,\&\, \leqslant E$, $B \to b$, $E \to ab$ define that the substring b occurring in the string $w = ab$ has the property A. One can symmetrically define right contexts, denoted by the quantifiers $\rhd\alpha$ and $\geqslant\alpha$.

In the literature, related ideas have occasionally arisen in connection with parsing, where right contexts—$\geqslant\alpha\Sigma^*$, in the terminology of this paper—are considered as "lookahead strings" and are used to guide a deterministic parser. If α represents a regular language, these simple forms of contexts occur in LR-regular [3], LL-regular [7] and LL(*) [21] parsers. Some software tools for engineering parsers, such as those developed by Parr and Fischer [21] and by Ford [5], allow specifying contexts $\geqslant\alpha\Sigma^*$, with α defined within the grammar, and such specifications can be used by a programmer for *ad hoc* adjustment of the behaviour of a deterministic recursive descent parser.

In this paper, the above intuitive definition of grammars with one-sided contexts is formalized in two equivalent ways. The first possibility, pursued in Section 2, is to consider deduction of elementary propositions of the form $[A, u\langle v\rangle]$, where $u\langle v\rangle$ denotes a substring v in left context u (that is, occurring in a string uvw) and A is a syntactic property defined by the grammar ("nonterminal symbol" in Chomsky's terminology); this proposition asserts that v has the property A in the context u. Then, each rule of the grammar, which is of the general form $A \to \alpha_1 \,\&\, \ldots \,\&\, \alpha_k \,\&\, \lhd\beta_1 \,\&\, \ldots \,\&\, \lhd\beta_m \,\&\, \leqslant\gamma_1 \,\&\, \ldots \,\&\, \leqslant\gamma_n$, becomes a deduction scheme for inferring elementary propositions of this form from each other, and the language generated by the grammar is ultimately defined as the set of all such strings w, that $[S, \varepsilon\langle w\rangle]$ can be deduced. A standard proof tree of such a deduction constitutes a parse tree of the string w. This definition generalizes the representation of standard context-free grammars by deduction—assumed, for instance, in a monograph by Sikkel [22]—as well as the extension of this representation to conjunctive grammars [14].

An alternative, equivalent definition given in Section 3 uses a generalization of language equations, in which the unknowns are *sets of pairs* of a string and its left contexts. All connectives and quantifiers in the rules of a grammar—that is, concatenation, disjunction, conjunction and both context quantifiers—are then interpreted as operations on such sets, and the resulting system of equations is proved to have a least fixpoint, as in the known cases of standard context-free grammars [6] and conjunctive grammars [12]. This least solution defines the language generated by the grammar.

These definitions ensure that the proposed grammars with one-sided contexts define the properties of strings inductively from the properties of their substrings and the contexts, in which these substrings occur. There is no uncontrollable rewriting of "sentential forms" involved, and hence the proposed model avoids falling into the same pit as Chomsky's "context-sensitive grammars", that of being able to simulate computations of space-bounded

Turing machines.

This paper settles the basic properties of grammars with one-sided contexts. First, a transformation to a normal form generalizing the Chomsky normal form is devised in Section 4; the construction proceeds in the usual way, first by eliminating empty strings, and then by removing cyclic dependencies. This normal form is then used to extend the basic Cocke–Kasami–Younger parsing algorithm to grammars with one-sided contexts; the algorithm, described in Section 5, works in time $O(n^3)$, where n is the length of the input string. Finally, in Section 6, it is demonstrated that every language defined by a grammar with one-sided contexts can be recognized in deterministic linear space.

2 Definition by deduction

A grammar with one-sided contexts uses concatenation, conjunction and disjunction, as well as quantifiers, either only for left contexts ($\lhd$, $\trianglelefteq$), or only for right contexts ($\rhd$, $\trianglerighteq$). Though left contexts are assumed throughout this paper, all results symmetrically hold for grammars with right contexts.

Definition 1. *A grammar with left contexts is a quadruple $G = (\Sigma, N, P, S)$, where*

- *Σ is the alphabet of the language being defined;*
- *N is a finite set of auxiliary symbols ("nonterminal symbols" in Chomsky's terminology), disjoint with Σ, which denote the properties of strings defined in the grammar;*
- *P is a finite set of grammar rules, each of the form*

$$A \to \alpha_1 \,\&\, \ldots \,\&\, \alpha_k \,\&\, \lhd\beta_1 \,\&\, \ldots \,\&\, \lhd\beta_m \,\&\, \trianglelefteq\gamma_1 \,\&\, \ldots \,\&\, \trianglelefteq\gamma_n, \quad (1)$$

  with $A \in N$, $k, m, n \geqslant 0$, $\alpha_i, \beta_i, \gamma_i \in (\Sigma \cup N)^*$;
- *$S \in N$ is a symbol representing correct sentences (in the common jargon, "the start symbol").*

For each grammar rule (1), each term α_i, $\lhd\beta_i$ and $\trianglelefteq\gamma_i$ is called a *conjunct*. Each unquantified conjunct α_i gives a representation of the string being defined. A conjunct $\lhd\beta_i$ similarly describes the form of the *left context* or the *past*, relative to the string being defined. Conjuncts of the form $\trianglelefteq\gamma_i$ refer to the form of the left context and the current string, concatenated into a single string.

A grammar with left contexts degenerates to a conjunctive grammar, if the context quantifiers are never used, that is, if $m = n = 0$ for every rule (1);

and further to a standard context-free grammar, if conjunction is never used, that is, if $k = 1$ in every rule.

Intuitively, a rule (1) can be read as follows: a substring v occurring in a left context u has the property A, if

- for each $i \in \{1, \ldots, k\}$, the string v is representable as a concatenation $\alpha_i = X_1 \ldots X_\ell$, with $X_1, \ldots, X_\ell \in \Sigma \cup N$, where each symbol $X_i \in N$ represents any substring with the property s_i, *and*
- for each $i \in \{1, \ldots, m\}$, the left context u is representable as a concatenation β_i, *and*
- for each $i \in \{1, \ldots, n\}$, the string uv is representable as a concatenation γ_i.

Thus, the conjunction sign in (1) indeed represents logical conjunction. As in a standard context-free grammar, multiple rules $A \to \mathcal{A}_1, \ldots, A \to \mathcal{A}_\ell$ for a single nonterminal A represent logical disjunction of conditions, and are denoted by

$$A \to \mathcal{A}_1 \mid \ldots \mid \mathcal{A}_\ell.$$

Formally, the semantics of grammars with contexts are defined by a deduction system of elementary propositions (items) of the form "a string $v \in \Sigma^*$ written in left context $u \in \Sigma^*$ has the property $X \in \Sigma \cup N$", denoted by $[X, u\langle v\rangle]$.

Definition 2. *Let $G = (\Sigma, N, P, S)$ be a grammar with contexts, and define the following deduction system of items of the form $[X, u\langle v\rangle]$, with $X \in \Sigma \cup N$ and $u, v \in \Sigma^*$. There is a single axiom scheme:*

$$\vdash_G [a, x\langle a\rangle] \quad (\textit{for all } a \in \Sigma \textit{ and } x \in \Sigma^*).$$

Each rule $A \to \alpha_1 \,\&\, \ldots \,\&\, \alpha_k \,\&\, \lhd\beta_1 \,\&\, \ldots \,\&\, \lhd\beta_m \,\&\, \trianglelefteq\gamma_1 \,\&\, \ldots \,\&\, \trianglelefteq\gamma_n$ in the grammar defines the following scheme for deduction rules:

$$I \vdash_G [A, u\langle v\rangle],$$

for all $u, v \in \Sigma^*$ and for every set of items I satisfying the below properties:

- *For every unquantified conjunct $\alpha_i = X_1 \ldots X_\ell$ with $\ell \geqslant 0$ and $X_j \in \Sigma \cup N$, there should exist a partition $v = v_1 \ldots v_\ell$ with $[X_j, uv_1 \ldots v_{j-1}\langle v_j\rangle] \in I$ for all $j \in \{1, \ldots, \ell\}$.*
- *For every conjunct $\lhd\beta_i = \lhd X_1 \ldots X_\ell$ with $\ell \geqslant 0$ and $X_j \in \Sigma \cup N$, there should be such a partition $u = u_1 \ldots u_\ell$, that $[X_j, u_1 \ldots u_{j-1}\langle u_j\rangle] \in I$ for all $j \in \{1, \ldots, \ell\}$.*
- *Every conjunct $\trianglelefteq\gamma_i = \trianglelefteq X_1 \ldots X_\ell$ with $\ell \geqslant 0$ and $X_j \in \Sigma \cup N$ should have a corresponding partition $uv = w_1 \ldots w_\ell$ with $[X_j, w_1 \ldots w_{j-1}\langle w_j\rangle] \in I$ for all j.*

Then the language generated by a nonterminal symbol A is defined as

$$L_G(A) = \{\, u\langle v\rangle \mid u, v \in \Sigma^*,\ \vdash_G [A, u\langle v\rangle] \,\}.$$

The language generated by the grammar G is the set of all strings with left context ε generated by S:

$$L(G) = \{\, w \mid w \in \Sigma^*,\ \vdash_G [S, \varepsilon\langle w\rangle] \,\}.$$

Note that if $\alpha_i = \varepsilon$ in the first case, then the given condition implies $v = \varepsilon$, and similarly, $\beta_i = \varepsilon$ implies $u = \varepsilon$, and $\gamma_i = \varepsilon$ implies $u = v = \varepsilon$.

Using both kinds of past quantifiers ($\lhd$ and $\trianglelefteq$) is actually redundant, since each of them can be expressed through the other as follows:

- $\lhd D$ is equivalent to $D'\Sigma^*$, for a new nonterminal symbol D' with the sole rule $D' \to \varepsilon \,\&\, \trianglelefteq D$;
- $\trianglelefteq E$ can be replaced by $\Sigma^* E''$, where the new nonterminal E'' has the unique rule $E'' \to \varepsilon \,\&\, \lhd E$.

Using both types of contexts shall be essential later in Section 4, when transforming the grammar to a normal form, in which the empty string is prohibited.

Ambiguity in grammars with contexts can be defined in exactly the same way as in the case of conjunctive grammars [17].

Definition 3. *A grammar with contexts $G = (\Sigma, N, P, S)$ is said to be unambiguous, if*

I. for every item $\vdash_G [A, u\langle v\rangle]$ with $A \in N$ and $u, v \in \Sigma^*$, there exists a unique rule, by which this item is deduced (in other words, different rules generate disjoint languages), and

II. for every conjunct $X_1 \dots X_\ell$, $\lhd X_1 \dots X_\ell$ or $\trianglelefteq X_1 \dots X_\ell$ that occurs in any rule, and for all $u, v \in \Sigma^*$, there exists at most one partition $v = v_1 \dots v_\ell$ with $uv_1 \dots v_{i-1}\langle v_i\rangle \in L_G(X_i)$ for all i.

The following sample grammar with contexts defines a rather simple language, which is an intersection of two standard context-free languages, and hence could be defined by a conjunctive grammar without contexts. The value of this example is in demonstrating the machinery of contexts in action.

Example 1. The following grammar generates the language $\{\, a^n b^n c^n d^n \mid n \geqslant 0 \,\}$:

$$\begin{aligned} S &\to aSd \mid bSc \mid \varepsilon \,\&\, \lhd A \\ A &\to aAb \mid \varepsilon \end{aligned}$$

The grammar is unambiguous.

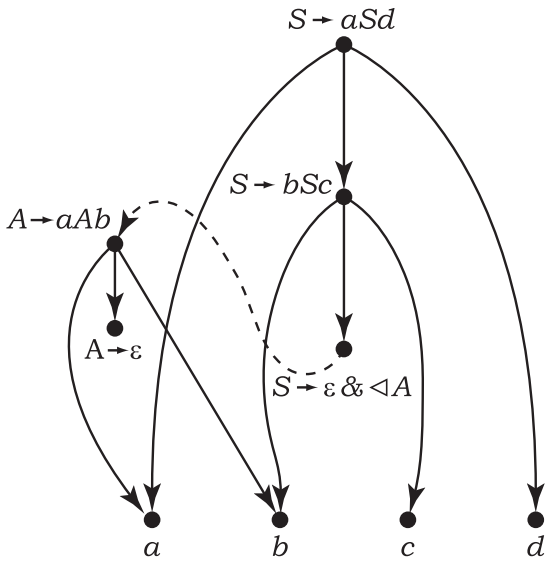

Figure 1: A parse tree of the string *abcd* according to the grammar in Example 1.

The symbol A generates all strings $a^n b^n$ with $n \geqslant 0$ in any context. Without the context specification $\lhd A$, the symbol S would define all strings of the form $wh(w^R)$, where $w \in \{a, b\}^*$ and the homomorphism h maps a to d and b to c. However, the rule $S \to \varepsilon \,\&\, \lhd A$ ensures that the first half of the string is of the form $a^n b^n$ for some $n \geqslant 0$, and therefore S generates only strings of the form $a^n b^n c^n d^n$ with $n \geqslant 0$. Consider the following logical derivation of the fact that the string *abcd* with left context ε is defined by S.

$$\begin{aligned}
\vdash [a, \varepsilon\langle a\rangle] \qquad & (axiom)\\
\vdash [b, a\langle b\rangle] \qquad & (axiom)\\
\vdash [c, ab\langle c\rangle] \qquad & (axiom)\\
\vdash [d, abc\langle d\rangle] \qquad & (axiom)\\
\vdash [A, a\langle\varepsilon\rangle] \qquad & (A \to \varepsilon)\\
[a, \varepsilon\langle a\rangle], [A, a\langle\varepsilon\rangle], [b, a\langle b\rangle] \vdash [A, \varepsilon\langle ab\rangle] \qquad & (A \to aAb)\\
[A, \varepsilon\langle ab\rangle] \vdash [S, ab\langle\varepsilon\rangle] \qquad & (S \to \varepsilon \,\&\, \lhd A)\\
[b, a\langle b\rangle], [S, ab\langle\varepsilon\rangle], [c, ab\langle c\rangle] \vdash [S, a\langle bc\rangle] \qquad & (S \to bSc)\\
[a, \varepsilon\langle a\rangle], [S, a\langle bc\rangle], [d, abc\langle d\rangle] \vdash [S, \varepsilon\langle abcd\rangle] \qquad & (S \to aSd)
\end{aligned}$$

The tree corresponding to this deduction is given in Figure 1, where the dependence upon a context is marked by a dotted arrow.

To see that the grammar is unambiguous, consider that each string in $L_G(S)$ either begins with a and is generated by the rule $S \to aSd$, or begins with b and is generated by $S \to bSc$, or is empty and is generated by the last rule $S \to \varepsilon \,\&\, \lhd A$; hence, the rules for S generate disjoint languages.

Similarly, one of the two rules for A generates non-empty strings that begin with a, and the other generates the empty string.

The next grammar defines a language, for which no Boolean grammar is known.

Example 2. The following grammar with contexts generates the language $\{u_1 \dots u_n \mid \text{for every } i,\ u_i \in a^*c,\ \textbf{or}\ \text{there exist } j,k \text{ with } u_i = b^kc \text{ and } u_j = a^kc\}$:

$$\begin{aligned}
S &\to AcS \mid CcS \mid BcS \,\&\, DcE \mid \varepsilon \\
A &\to aA \mid \varepsilon \\
B &\to bB \mid \varepsilon \\
C &\to B \,\&\, \preccurlyeq EF \\
D &\to bDa \mid cE \\
E &\to AcE \mid BcE \mid \varepsilon \\
F &\to aFb \mid cE
\end{aligned}$$

This is an abstract language representing declaration of identifiers *before or after* their use. Substrings of the form a^kc represent declarations, while every substring of the form b^kc is a reference to a declaration of the form a^kc.

The idea of the grammar is that S should generate a string $u_1 \dots u_\ell \langle u_{\ell+1} \dots u_n \rangle$ with $u_i \in a^*c \cup b^*c$ if every reference in the suffix $u_{\ell+1} \dots u_n$ has a corresponding declaration in the whole string $u_1 \dots u_n$. This condition is defined inductively on ℓ. The rule $S \to \varepsilon$ is the basis of induction: the string $u_1 \dots u_n \langle \varepsilon \rangle$ has the desired property. The rule $S \to CcS$ appends a reference of the form $(b^* \,\&\, \preccurlyeq EF)c$, where the context specification ensures that this reference has a matching *earlier* declaration. The possibility of a *later* declaration is checked by another rule $S \to BcS \,\&\, DcE$.

3 Definition by language equations

The representation of standard context-free grammars by language equations, introduced by Ginsburg and Rice [6], is one of the several ways of defining their semantics. For example, a grammar $S \to aSb \mid \varepsilon$ is regarded as an equation $S = \big(\{a\} \cdot S \cdot \{b\}\big) \cup \{\varepsilon\}$, which has the unique solution $S = \{\, a^n b^n \mid n \geqslant 0 \,\}$. Conjunctive grammars inherit the same definition by equations [12], with the conjunction represented by the intersection operation.

This important representation can be extended to grammars with contexts. However, in order to include the contexts in the equations, the whole model has to be extended from ordinary formal languages to sets of pairs of the form $u\langle v \rangle$, that is, to languages of pairs $L \subseteq \Sigma^* \times \Sigma^*$.

All usual operations on languages used in equations will have to be extended to languages of pairs. For all $K, L \subseteq \Sigma^* \times \Sigma^*$, consider their

- *union* $K \cup L = \{\, u\langle v\rangle \mid u\langle v\rangle \in K \text{ or } u\langle v\rangle \in L \,\}$;
- *intersection* $K \cap L = \{\, u\langle v\rangle \mid u\langle v\rangle \in K,\ u\langle v\rangle \in L \,\}$;
- *concatenation* $K \cdot L = \{\, u\langle vw\rangle \mid u\langle v\rangle \in K,\ uv\langle w\rangle \in L \,\}$;
- $\lhd$*-context* $\lhd L = \{\, u\langle v\rangle \mid \varepsilon\langle u\rangle \in L,\ v \in \Sigma^* \,\}$;
- $\trianglelefteqslant$*-context* $\trianglelefteqslant L = \{\, u\langle v\rangle \mid \varepsilon\langle uv\rangle \in L \,\}$.

Definition 4. *For every grammar with contexts $G = (\Sigma, N, P, S)$, the associated system of language equations is a system of equations in variables N, in which each variable assumes a value of a language of pairs $L \subseteq \Sigma^* \times \Sigma^*$, and which contains the following equations for every variable A:*

$$A = \bigcup_{\substack{A \to \alpha_1 \& \ldots \& \alpha_k \& \\ \& \lhd \beta_1 \& \ldots \& \lhd \beta_m \& \\ \& \trianglelefteqslant \gamma_1 \& \ldots \& \trianglelefteqslant \gamma_n \in P}} \Big[\bigcap_{i=1}^{k} \alpha_i \ \cap \bigcap_{i=1}^{m} \lhd \beta_i \ \cap \bigcap_{i=1}^{n} \trianglelefteqslant \gamma_i \Big]. \tag{2}$$

Each instance of a symbol $a \in \Sigma$ in such a system defines the language $\{\, x\langle a\rangle \mid x \in \Sigma^* \,\}$, and each empty string denotes the language $\{\, x\langle\varepsilon\rangle \mid x \in \Sigma^* \,\}$. A solution of such a system is a vector of languages $(\ldots, L_A, \ldots)_{A \in N}$, such that the substitution of L_A for A, for all $A \in N$, turns each equation (2) into an equality.

This system always has solutions, and among them the *least solution* with respect to the partial order $\sqsubseteq$ of componentwise inclusion on the set $(2^{\Sigma^* \times \Sigma^*})^n$. For any two n-tuples of languages of pairs, let $(K_1, \ldots, K_n) \sqsubseteq (L_1, \ldots, L_n)$ if and only if $K_i \subseteq L_i$. Its least element is $\bot = (\varnothing, \ldots, \varnothing)$.

Consider a system of language equations of the form

$$X_i = \varphi_i(X_1, \ldots, X_n) \qquad (1 \leqslant i \leqslant n),$$

where $\varphi_i : (2^{\Sigma^* \times \Sigma^*})^n \to 2^{\Sigma^* \times \Sigma^*}$ are functions of $X_1, \ldots, X_n$, defined using the operations of concatenation, union, intersection, and the $\lhd$- and $\trianglelefteqslant$-contexts.

The right-hand sides of such a system can be represented as a vector function $\varphi = (\varphi_1, \ldots, \varphi_n)$, which has the following properties:

Lemma 1. *Vector function $\varphi = (\varphi_1, \ldots, \varphi_n)$ is monotone, in the sense that for any two vectors K and L, the inequality $K \sqsubseteq L$ implies $\varphi(K) \sqsubseteq \varphi(L)$.*

Lemma 2. *Vector function $\varphi = (\varphi_1, \ldots, \varphi_n)$ is continuous, in the sense that for every sequence of vectors of languages of pairs $\{L^{(i)}\}_{i=1}^{\infty}$, it holds that*

$$\bigsqcup_{i=1}^{\infty} \varphi(L^{(i)}) = \varphi\Big(\bigsqcup_{i=1}^{\infty} L^{(i)} \Big)$$

The next result follows by the standard method of least fixed points.

Lemma 3. *If φ is monotone and continuous, then the least solution of a system $X = \varphi(X)$ is the vector*

$$\bigsqcup_{k=0}^{\infty} \varphi^k(\bot).$$

Therefore, every system of equations corresponding to a grammar with contexts has a least solution, which shall be used to give an equivalent definition of the language generated by a grammar.

Definition 5. *Let $G = (\Sigma, N, P, S)$ be a grammar with contexts, let $X = \varphi(X)$ be the associated system of language equations, and let $(L_{A_1}, \ldots, L_{A_n})$ with $L_{A_i} \subseteq \Sigma^* \times \Sigma^*$, $A_i \in N$ be its least solution. Define the language generated by each nonterminal symbol $A \in N$ as the corresponding component of this solution: $L_G(A) = L_A$. Let $L(G) = \{\, w \mid \varepsilon\langle w\rangle \in L_S \,\}$.*

Example 3. The following system of equations represents the grammar in Example 1:

$$\begin{aligned} S &= (\Sigma^*\langle a\rangle \cdot S \cdot \Sigma^*\langle d\rangle) \cup (\Sigma^*\langle b\rangle \cdot S \cdot \Sigma^*\langle c\rangle) \cup \big(\Sigma^*\langle\varepsilon\rangle \cap \lhd A\big) \\ A &= (\Sigma^*\langle a\rangle \cdot A \cdot \Sigma^*\langle b\rangle) \cup \Sigma^*\langle\varepsilon\rangle \end{aligned}$$

Consider the sequence of iterations leading to the least solution of the system.

$$\varphi^0(\bot) = \begin{pmatrix} \varnothing \\ \varnothing \end{pmatrix};$$

$$\varphi^1(\bot) = \begin{pmatrix} \varnothing \\ \Sigma^*\langle\varepsilon\rangle \end{pmatrix};$$

$$\varphi^2(\bot) = \begin{pmatrix} \varepsilon\langle\varepsilon\rangle \\ \Sigma^*\langle\varepsilon\rangle \cup \Sigma^*\langle ab\rangle \end{pmatrix};$$

$$\varphi^3(\bot) = \begin{pmatrix} \varepsilon\langle\varepsilon\rangle \cup ab\langle\varepsilon\rangle \\ \Sigma^*\langle\varepsilon\rangle \cup \Sigma^*\langle ab\rangle \cup \Sigma^*\langle aabb\rangle \end{pmatrix};$$

$$\varphi^4(\bot) = \begin{pmatrix} \varepsilon\langle\varepsilon\rangle \cup ab\langle\varepsilon\rangle \cup a\langle bc\rangle \cup aabb\langle\varepsilon\rangle \\ \Sigma^*\langle\varepsilon\rangle \cup \Sigma^*\langle ab\rangle \cup \Sigma^*\langle a^2b^2\rangle \cup \Sigma^*\langle a^3b^3\rangle \end{pmatrix};$$

$$\varphi^5(\bot) = \begin{pmatrix} \varepsilon\langle\varepsilon\rangle \cup ab\langle\varepsilon\rangle \cup a^2b^2\langle\varepsilon\rangle \cup a^3b^3\langle\varepsilon\rangle \cup a\langle bc\rangle \cup aab\langle bc\rangle \cup \varepsilon\langle abcd\rangle \\ \Sigma^*\langle\varepsilon\rangle \cup \Sigma^*\langle ab\rangle \cup \Sigma^*\langle a^2b^2\rangle \cup \Sigma^*\langle a^3b^3\rangle \cup \Sigma^*\langle a^4b^4\rangle \end{pmatrix}.$$

Thus, the unique solution of the system is $S = \{\, a^i\langle a^{n-i}b^nc^nd^n\rangle \mid n \geqslant i \geqslant 0 \,\} \cup \{\, a^nb^i\langle b^{n-i}c^nd^n\rangle \mid n \geqslant i \geqslant 0 \,\}$, and $A = \{\, x\langle a^nb^n\rangle \mid x \in \Sigma^* \,\}$.

Definitions 2 and 5 are proved equivalent as follows. Let $\varphi = (\varphi_1, \ldots, \varphi_n)$ be a vector function. Denote by $[\varphi]_i$ the i-th component of the vector φ.

Theorem 1. *Let $G = (\Sigma, N, P, S)$ be a grammar with contexts and let $X = \varphi(X)$ be the associated system of language equations. Let $u\langle v\rangle \in \Sigma^* \times \Sigma^*$ be a string with a context. Then, for every $A \in N$,*

$$u\langle v\rangle \in \Big[\bigsqcup_{t\geqslant 0} \varphi^t(\varnothing, \ldots, \varnothing)\Big]_A \quad \textit{if and only if} \quad \vdash_G [A, u\langle v\rangle].$$

The following obvious property of concatenation of languages with contexts shall be referenced in the proof of the theorem.

Lemma 4. *Let $L_1, \ldots, L_k \subseteq \Sigma^* \times \Sigma^*$ and let $u\langle v\rangle$ be a string with a context. Then $u\langle v\rangle \in L_1 \cdot \ldots \cdot L_k$ if and only if there exists a factorization $v = v_1 \ldots v_k$ such that $uv_1 \ldots v_{i-1}\langle v_i\rangle \in L_i$ for all $i \in \{1, \ldots, k\}$.*

Proof. $\Longrightarrow$ Let $u\langle v\rangle \in L_1 \cdot \ldots \cdot L_k$ be a string with a context. It can be factorized as $u\langle v\rangle = u_1\langle v_1\rangle \cdot \ldots \cdot u_k\langle v_k\rangle$, with $u_i\langle v_i\rangle \in L_i$ for all $i \in \{1, \ldots, \ell\}$.

For a pair of strings $u_i\langle v_i\rangle$ and $u_j\langle v_j\rangle$, according to the definition of concatenation of strings with a context, $u_i\langle v_i\rangle \cdot u_j\langle v_j\rangle = u_i\langle v_i v_j\rangle$ with $u_j = u_i v_i$. Concatenation of the form $u_1\langle v_1\rangle \cdot \ldots \cdot u_k\langle v_k\rangle = u\langle v\rangle$ can only be obtained if $u_1 = u$ and $v = v_1 \ldots v_k$.
$\Longleftarrow$ Let $u \in \Sigma^*$ and $v = v_1 \ldots v_k$ (with $v_i \in \Sigma$) be strings with a context, such that $uv_1 \ldots v_{i-1}\langle v_i\rangle \in L_i$ for a given set of languages $L_1, \ldots, L_k$.

According to the definition of concatenation of two strings with contexts,

$$\begin{aligned} u\langle v_1\rangle \cdot uv_1\langle v_2\rangle \cdot uv_1v_2\langle v_3\rangle \cdot \ldots \cdot uv_1 \ldots v_{k-1}\langle v_k\rangle &= \\ = u\langle v_1v_2\rangle \cdot uv_1v_2\langle v_3\rangle \cdot \ldots \cdot uv_1 \ldots v_{k-1}\langle v_k\rangle &= \\ = u\langle v_1v_2v_3\rangle \cdot \ldots \cdot uv_1 \ldots v_{k-1}\langle v_k\rangle &= \\ = u\langle v_1 \ldots v_k\rangle = u\langle v\rangle \in L_1 \cdot \ldots \cdot L_k. & \end{aligned}$$

□

Proof of Theorem 1. $\Longrightarrow$ The proof is by induction on t, the number of iterations made until $u\langle v\rangle \in \Big[\varphi^t(\varnothing, \ldots, \varnothing)\Big]_A$.

Basis. Let $t = 0$, then there are no $A \in N$ satisfying the assumptions.

Induction step. Let the string $u\langle v\rangle$ be obtained in t iterations, that is, $u\langle v\rangle \in \varphi_A(\varphi^{t-1}(\bot))$, where $\bot = (\varnothing, \ldots, \varnothing)$ is the least element. Then, substituting $\varphi^{t-1}(\bot)$ into the equation for A yields

$$u\langle v\rangle \in \bigcup_{\substack{A\to\alpha_1\&\ldots\&\alpha_k\& \\ \&\lhd\beta_1\&\ldots\&\lhd\beta_m\& \\ \&\trianglelefteq\gamma_1\&\ldots\&\trianglelefteq\gamma_n\in P}} \Big[\bigcap_{i=1}^{k} \alpha_i(\varphi^{t-1}(\bot)) \cap \bigcap_{i=1}^{m} \lhd\beta_i(\varphi^{t-1}(\bot)) \cap \bigcap_{i=1}^{n} \trianglelefteq\gamma_i(\varphi^{t-1}(\bot))\Big].$$

Hence, there should exist a rule

$$A \to \alpha_1 \,\&\, \ldots \,\&\, \alpha_k \,\&\, \lhd\beta_1 \,\&\, \ldots \lhd\beta_m \,\&\, \trianglelefteq\gamma_1 \,\&\, \ldots \,\&\, \trianglelefteq\gamma_n, \tag{3}$$

such that

- for each $\alpha_i = X_1 \ldots X_\ell$ with $i \in \{1, \ldots, k\}$, $uv_1 \ldots v_{j-1}\langle v_j\rangle \in X_j(\varphi^{t-1}(\bot))$, for all $j \in \{1, \ldots, \ell\}$;
- for each $\lhd\beta_i = \lhd Y_1 \ldots Y_\ell$ with $i \in \{1, \ldots, m\}$, $u_1 \ldots u_{j-1}\langle u_j\rangle \in X_j(\varphi^{t-1}(\bot))$, for all $j \in \{1, \ldots, \ell\}$;
- for each $\trianglelefteq\gamma_i = \trianglelefteq Z_1 \ldots Z_\ell$ with $i \in \{1, \ldots, n\}$, $w_1 \ldots w_{j-1}\langle w_j\rangle \in X_j(\varphi^{t-1}(\bot))$, with $uv = w_1 \ldots w_\ell$, for all $j \in \{1, \ldots, \ell\}$.

Consider first the conjuncts of the form α_i, for $i \in \{1, \ldots, k\}$. Let $\alpha_i = X_1 \ldots X_\ell$ with $X_j \in \Sigma \cup N$ and $j \in \{1, \ldots, \ell\}$. Substituting $\varphi^{t-1}(\bot)$ into each X_j gives $\alpha_i(\varphi^{t-1}(\bot)) = X_1(\varphi^{t-1}(\bot)) \cdot \ldots \cdot X_\ell(\varphi^{t-1}(\bot))$. If $X_j = a \in \Sigma$, then the substitution forms a constant language: $X_j(\varphi^{t-1}(\bot)) = \{\, x\langle a\rangle \mid x \in \Sigma^* \,\}$. If X_j is a nonterminal $B \in N$, then $X_j(\varphi^{t-1}(\bot)) = \left[\varphi^{t-1}(\bot)\right]_B$.

By Lemma 4, if a string $u\langle v\rangle$ belongs to the concatenation $X_1 \cdot \ldots \cdot X_\ell$, then it is of the form $u\langle v\rangle = u\langle v_1\rangle \cdot uv_1\langle v_2\rangle \cdot \ldots \cdot uv_1 \ldots v_{\ell-1}\langle v_\ell\rangle$, with

$$uv_1 \ldots v_{j-1}\langle v_j\rangle \in X_j(\varphi^{t-1}(\bot)). \tag{4}$$

Then, for each nonterminal symbol $X_j = B \in N$, the string with a context $uv_1 \ldots v_{j-1}\langle v_j\rangle$ should be in $\left[\varphi^{t-1}(\bot)\right]_B$, and hence, by the induction hypothesis, $\vdash_G [X_j, uv_1 \ldots v_{j-1}\langle v_j\rangle]$. For each $X_j = a \in \Sigma$, the above condition (4) implies that $v_j = a$, and hence $\vdash_G [X_j, uv_1 \ldots v_{j-1}\langle v_j\rangle]$ as an axiom.

The cases of conjuncts $\lhd\beta_i$ and $\trianglelefteq\gamma_i$ are analogous: one can show that $\vdash_G [Y_j, u_1 \ldots u_{j-1}\langle u_j\rangle]$ and $\vdash_G [Z_j, w_1 \ldots w_{j-1}\langle w_j\rangle]$, using the induction hypothesis for nonterminal symbols and the axiom for terminal symbols.

Thus all the premises for deducing the item $[A, u\langle v\rangle]$ according to the rule (3) have been obtained, and one can make the desired step of deduction: $\vdash_G [A, u\langle v\rangle]$.

$\Leftarrow$ Assume that $\vdash_G [A, u\langle v\rangle]$, and that it is deduced in p steps. It is claimed that $u\langle v\rangle \in \left[\varphi^t(\bot)\right]_A$ for some $t = t(p)$. The proof is by induction on p.

Basis. Let $p = 1$, that is, the item $[A, u\langle v\rangle]$ is deduced in a single step, without any premises. Then it is obtained by a rule comprised of one or more conjuncts of the form v, $\lhd u$ and $\trianglelefteq uv$ (such as $A \to v$ or $A \to v\&\lhd u\&\trianglelefteq uv$). Then $u\langle v\rangle \in \varphi^1(\varnothing, \ldots, \varnothing)$.

Induction step. Assume that an item $[A, u\langle v\rangle]$ is deduced in p steps.

The pth step of deduction is applying some rule

$$A \to \alpha_1 \,\&\, \ldots \,\&\, \alpha_k \,\&\, \lhd\beta_1 \,\&\, \ldots \,\&\, \lhd\beta_m \,\&\, \trianglelefteq\gamma_1 \,\&\, \ldots \,\&\, \trianglelefteq\gamma_n,$$

such that

- $\vdash_G [X_j, uv_1 \dots v_{j-1}\langle v_j \rangle]$, with $j \in \{1, \dots, \ell\}$, for each conjunct $\alpha_i = X_1 \dots X_\ell$ in the rule;
- $\vdash_G [Y_j, u_1 \dots u_{j-1}\langle u_j \rangle]$, $j \in \{1, \dots, \ell\}$, for each conjunct $\lhd\beta_i = \lhd Y_1 \dots Y_\ell$ in the rule;
- $\vdash_G [Z_j, w_1 \dots w_{j-1}\langle w_j \rangle]$, $j \in \{1, \dots, \ell\}$, $uv = w_1 \dots w_\ell$, for each conjunct $\leqslant\gamma_i = \leqslant Z_1 \dots Z_\ell$ in the rule.

Consider any unquantified conjunct $\alpha_i = X_1 \dots X_\ell$ with $X_j \in \Sigma \cup N$, and fix any j-th symbol X_j. It is known that $\vdash_G [X_j, uv_1 \dots v_{j-1}\langle v_j \rangle]$. If $X_j = B \in N$, then the deduction of the item $[X_j, uv_1 \dots v_{j-1}\langle v_j \rangle]$ takes less than p steps, and therefore, by the induction hypothesis, there exist a number $t_{i,j}$, such that $uv_1 \dots v_{j-1}\langle v_j \rangle \in B(\varphi^{t_{i,j}}(\bot))$. If $X_j = a \in \Sigma$, then $uv_1 \dots v_{j-1}\langle v_j \rangle = uv_1 \dots v_{j-1}\langle a \rangle \in \Sigma^*\langle a \rangle$. As the concatenation of these strings with contexts is $u\langle v_1 \rangle \cdot uv_1\langle v_2 \rangle \cdot \ldots \cdot uv_1 \dots v_{j-1}\langle v_j \rangle = u\langle v \rangle$ by Lemma 4, altogether, $u\langle v \rangle \in \alpha_i(\varphi^{t_i}(\bot))$, where $t_i = \max_j t_{i,j}$.

Applying the same procedure to each conjunct α_i, $\lhd\beta_i$ or $\leqslant\gamma_i$ similarly yields $u\langle v \rangle \in \alpha_i(\varphi^{t_i}(\bot))$, $u\langle v \rangle \in \lhd\beta_i(\varphi^{t_i}(\bot))$ or $u\langle v \rangle \in \leqslant\gamma_i(\varphi^{t_i}(\bot))$, for some appropriate number t_i. Let t be the maximum of all these numbers. Then,

$$u\langle v \rangle \in \bigcap_{i=1}^{k} \alpha_i(\varphi^t(\bot)) \cap \bigcap_{i=1}^{m} \lhd\beta_i(\varphi^t(\bot)) \cap \bigcap_{i=1}^{n} \leqslant\gamma_i(\varphi^t(\bot)) \subseteq \left[\varphi^{t+1}(\bot)\right]_A,$$

as desired. □

4 Normal form

The origin of the normal form for grammars with one-sided contexts developed in this section is the Chomsky normal form for standard context-free grammars, in which all rules are of the form $A \to BC$ and $A \to a$. Its extension to conjunctive grammars allows rules of the form $A \to B_1C_1\& \dots \&B_kC_k$. The transformation to this normal form is done by first eliminating *ε-conjuncts*, that is, rules of the form $A \to \varepsilon\& \dots$, and then removing *unit conjuncts*, or rules of the form $A \to B\& \dots$. This transformation shall now be further extended to grammars with contexts.

The task of eliminating ε-conjuncts is formulated in the same way: for any given grammar with contexts, the goal is to construct an equivalent (with exception of the membership of ε) grammar without epsilon conjuncts. A similar construction for context-free grammars (as well as for conjunctive grammars) begins with determining the set of nonterminals that generate the empty string, which is obtained as a least upper bound of an ascending sequence of sets of nonterminals. For grammars with contexts, it is necessary to consider pairs of the form (A, R), with $A \in N$ and $R \subseteq N$, representing the intuitive idea that A generates ε in the context of the form described by

all nonterminals in R. The set of all such pairs is obtained as a limit of a sequence of sets as follows.

Definition 6. *Let $G = (\Sigma, N, P, S)$ be a grammar with contexts. Assume, without loss of generality, that in each rule of the grammar, the context quantifiers are applied only to single nonterminal symbols rather than concatenations. Construct the sequence of sets $\textsc{Nullable}_i(G) \subseteq N \times 2^N$, with $i \geqslant 0$, by setting*

$$\textsc{Nullable}_0(G) = \varnothing,$$

$$\begin{aligned}\textsc{Nullable}_{i+1}(G) = \big\{\, (A, \{D_1, \ldots, D_m\} \cup \{E_1, \ldots, E_n\} \cup R_1 \cup \ldots \cup R_k) \;\big|\; \\ A \to \alpha_1 \,\&\, \ldots \,\&\, \alpha_k \,\&\, \lhd D_1 \,\&\, \ldots \,\&\, \lhd D_m \,\&\, \trianglelefteqslant E_1 \,\&\, \ldots \,\&\, \trianglelefteqslant E_n \in P, \\ \exists\, R_1, \ldots, R_k \subseteq N :\ (\alpha_1, R_1), \ldots, (\alpha_k, R_k) \in \textsc{Nullable}^*_i(G) \,\big\},\end{aligned}$$

where $\mathcal{S}^*$, for any set $\mathcal{S} \subseteq N \times 2^N$, denotes the set of all pairs $(A_1 \ldots A_\ell, R_1 \cup \ldots \cup R_\ell)$ with $\ell \geqslant 0$ and $(A_i, R_i) \in \mathcal{S}$.

Finally, let

$$\textsc{Nullable}(G) = \bigcup_{i \geqslant 0} \textsc{Nullable}_i(G).$$

In the definition of $\mathcal{S}^*$, note that $\varnothing^* = \{(\varepsilon, \varnothing)\}$. This value of $\textsc{Nullable}^*_i(G)$ is used in the construction of $\textsc{Nullable}_1(G)$.

The next lemma explains how the set $\textsc{Nullable}(G)$ represents the generation of ε by different nonterminals in different contexts.

Lemma 5. *Let $G = (\Sigma, N, P, S)$ be a grammar with contexts, let $A \in N$ and $u \in \Sigma^*$. Then, $u\langle\varepsilon\rangle \in L_G(A)$ if and only if there exist $K_1, \ldots, K_t \in N$, such that $(A, \{K_1, \ldots, K_t\}) \in \textsc{Nullable}(G)$ and $\varepsilon\langle u\rangle \in L_G(K_1)$, $\ldots$, $\varepsilon\langle u\rangle \in L_G(K_t)$.*

Let us now state an extended lemma, which shall clearly imply Lemma 5.

Lemma 5′. Let $G = (\Sigma, N, P, S)$ be a grammar with contexts, let $A \in N$ and $u \in \Sigma^*$. Then the following statements are equivalent:

1. there exist $K_1, \ldots, K_t \in N$, such that $(A, \{K_1, \ldots, K_t\}) \in \textsc{Nullable}(G)$ and $\varepsilon\langle u\rangle \in L_G(K_1)$, $\ldots$, $\varepsilon\langle u\rangle \in L_G(K_t)$;

2. $u\langle\varepsilon\rangle \in L_G(A)$;

3. there exists a rule

$$A \to \alpha_1 \,\&\, \ldots \,\&\, \alpha_k \,\&\, \lhd D_1 \,\&\, \ldots \,\&\, \lhd D_m \,\&\, \trianglelefteqslant E_1 \,\&\, \ldots \,\&\, \trianglelefteqslant E_n, \qquad (5)$$

   such that there exist $R_1, \ldots, R_k \subseteq N$ with $(\alpha_1, R_1), \ldots, (\alpha_k, R_k) \in \textsc{Nullable}^*(G)$ and $\varepsilon\langle u\rangle \in L_G(K)$ for all $K \in \{D_1, \ldots, D_m, E_1, \ldots, E_n\} \cup R_1 \cup \ldots \cup R_k$.

Proof. $\boxed{1 \Longrightarrow 2}$ Let $R = \{K_1, \ldots, K_t\}$ and $(A, R) \in \textsc{Nullable}(G)$. According to Definition 6, $(A, R) \in \textsc{Nullable}_n(G)$ for some $n \geqslant 0$. The proof is by induction on n.

Basis. Let $n = 0$, then $\textsc{Nullable}_0(G) = \varnothing$ and there is no $A \in N$ satisfying the assumptions.

Induction step. Let $(A, R) \in \textsc{Nullable}_n(G)$ and $\varepsilon\langle u\rangle \in L_G(K)$ for all $K \in R$.

According to Definition 6, the set P contains a rule of the form

$$A \to \alpha_1 \,\&\, \ldots \,\&\, \alpha_k \,\&\, \lhd K'_1 \,\&\, \ldots \,\&\, \lhd K'_q \,\&\, \trianglelefteqslant K'_{q+1} \,\&\, \ldots \,\&\, \trianglelefteqslant K'_{t'},$$

where $R' = \{K'_1, \ldots, K'_{t'}\} \subseteq R$.

For each unquantified conjunct α_i in this rule, let $R_i \subseteq R$ be any set with $(\alpha_i, R_i) \in \textsc{Nullable}^*_{n-1}(G)$; such a set exists according to Definition 6. Let $\alpha_i = X_{i,1} \ldots X_{i,\ell}$ with $\ell \geqslant 0$ and $X_{i,1}, \ldots, X_{i,\ell} \in \Sigma \cup N$. Then, by the definition of a "star" of $\textsc{Nullable}^*_{n-1}(G)$, there exist sets $R_{i,1}, \ldots, R_{i,\ell} \subseteq N$ with $R_{i,1} \cup \ldots \cup R_{i,\ell} = R_i$ and $(X_{i,j}, R_{i,j}) \in \textsc{Nullable}_{n-1}(G)$ for each j. Therefore, by the induction hypothesis ℓ times, $u\langle\varepsilon\rangle \in L_G(X_{i,j})$ for each j, that is, $\vdash_G [X_{i,1}, u\langle\varepsilon\rangle]$, $\ldots$, $[X_{i,\ell}, u\langle\varepsilon\rangle]$.

The same procedure is repeated for every pair $(\alpha_i, R_i) \in \textsc{Nullable}^*_{n-1}(G)$, where $\alpha_i = X_{i,1} \ldots X_{i,\ell_i}$, which gives all the premises for deducing the membership of $u\langle\varepsilon\rangle$ in $L_G(A)$:

$$[X_{1,1}, u\langle\varepsilon\rangle], \ldots, [X_{k,\ell}, u\langle\varepsilon\rangle], [K'_1, \varepsilon\langle u\rangle], \ldots, [K'_{t'}, \varepsilon\langle u\rangle] \vdash_G [A, u\langle\varepsilon\rangle].$$

$\boxed{2 \Longrightarrow 3}$ The proof is by induction on p, the number of steps used in deduction of an item $[A, u\langle\varepsilon\rangle]$.

Basis. Let $p = 1$. Consider an item $[A, u\langle v\rangle]$ which is obtained by some rule $A \to \varepsilon \in P$. Then $(A, \varnothing) \in \textsc{Nullable}(G)$.

Induction step. Consider an item $[A, u\langle\varepsilon\rangle]$ which is obtained by some rule

$$A \to \alpha_1 \,\&\, \ldots \,\&\, \alpha_k \,\&\, \lhd K'_1 \,\&\, \ldots \,\&\, \lhd K'_q \,\&\, \trianglelefteqslant K'_{q+1} \,\&\, \ldots \,\&\, \trianglelefteqslant K'_{t'} \in P, \quad (6)$$

such that $u\langle\varepsilon\rangle \in L_G(X_{i,j})$ (for each unquantified conjunct $\alpha_i = X_{i,1} \ldots X_{i,\ell_i}$) and $\varepsilon\langle u\rangle \in L_G(K'_i)$ (for each quantified conjunct K'_i of the rule, with $i \in \{1, \ldots, t'\}$).

Consider some conjunct $\alpha_i = X_{i,1} \ldots X_{i,\ell_i}$ of the rule (6). By definition of a "star" of $\textsc{Nullable}(G)$, there exist sets $R_{i,1}$, $\ldots$, R_{i,ℓ_i} with $R_{i,1} \cup \ldots \cup R_{i,\ell_i} = R_i$ such that $(X_{i,j}, R_{i,j}) \in \textsc{Nullable}^*(G)$, for $j \in \{1, \ldots, \ell_i\}$. According to Definition 6, $(\alpha_i, R_i) \in \textsc{Nullable}^*(G)$.

The same procedure is repeated for each unquantified conjunct of the rule (6).

Thus, $(\alpha_i, R_i) \in \textsc{Nullable}^*(G)$, for all $i \in \{1, \ldots, k\}$.

$\boxed{3 \Longrightarrow 1}$. Let the rule of the form (5) be in P. By Definition 6, $(A, \{D_1, \ldots, D_m, E_1, \ldots, E_n\} \cup R_1 \cup \ldots \cup R_k) \in \textsc{Nullable}_n(G)$ for some $n \geqslant 0$, where $R_1, \ldots, R_k \subseteq N$ and $(\alpha_i, R_i) \in \textsc{Nullable}^*_{n-1}(G)$ for each

$i \in \{1, \ldots, k\}$. Then $\{K_1, \ldots, K_t\} = \{D_1, \ldots, D_m, E_1, \ldots, E_n\} \cup R_1 \cup \ldots \cup R_k$, and the string $\varepsilon\langle u \rangle$ is in $L_G(K)$ for all $K \in \{K_1, \ldots, K_t\}$. □

It is convenient to begin the elimination of ε-conjuncts with the following pre-processing stage.

Lemma 6. *For every grammar with contexts, there exists and can be effectively constructed another grammar with contexts generating the same language, in which all rules are of the form*

$$A \to BC \tag{7a}$$
$$A \to a \tag{7b}$$
$$A \to B_1 \,\&\, \ldots \,\&\, B_k \,\&\, \lhd D_1 \,\&\, \ldots \,\&\, \lhd D_m \,\&\, \trianglelefteq E_1 \,\&\, \ldots \,\&\, \trianglelefteq E_n \tag{7c}$$
$$A \to \varepsilon, \tag{7d}$$

where $a \in \Sigma$ and $A, B, C, D_i, E_i \in N$.

Proof. Each conjunct in the given grammar $G = (\Sigma, N, P, S)$ is of one of the following forms:

- α, with $A \in N$, $\alpha \in (\Sigma \cup N)^*$, $|\alpha| \geqslant 1$;
- a, with $A \in N$, $a \in \Sigma$;
- ε, with $A \in N$.

All unquantified conjuncts α of A ($\alpha \in (\Sigma \cup N)^*$) which have more than two symbols, should be split into two nonterminals by introducing new ones. The resulting conjunct should be of the form BC with $B, C \in N$. Every appearance of a terminal $a \in \Sigma$ in a conjunct α_i is replaced with a new nonterminal X_a which has the unique rule $X_a \to a$.

All quantified conjuncts $\lhd\beta$ (with $|\beta| > 1$, $\beta \in (\Sigma \cup N)^*$) and $\trianglelefteq\gamma$ (with $|\gamma| > 1$, $\gamma \in (\Sigma \cup N)^*$) should be moved into a separate rule with a new nonterminal in its left-hand side. Each such new nonterminal should have one rule only.

All quantified conjuncts $\lhd a$ or $\trianglelefteq a$ (with $a \in \Sigma$) should be replaced with a conjunct of the form $\lhd X_a$ or $\trianglelefteq X_a$, respectively, and a new rule $X_a \to a$ should be added to the grammar.

All conjuncts of the form ε in a rule for nonterminal A are included in P' without any alteration. □

Lemma 7. *Let $G = (\Sigma, N, P, S)$ be a grammar with contexts, let $A \in N$ and $u \in \Sigma^*$, and assume that there exist two distinct sets $R, R' \subseteq N$ with $(A, R), (A, R') \in \textsc{Nullable}(G)$ and $\varepsilon\langle u \rangle \in L_G(K)$ for each $K \in R \cup R'$. Then the grammar is ambiguous.*

Proof. According to Definition 6, $(A, R), (A, R') \in \textsc{Nullable}_n(G)$ for some $n \geqslant 0$. Let n be the least such number.

Prove by induction on n, that the grammar G has an ambiguity of choice of rule for some nonterminal and for the string $u\langle\varepsilon\rangle$.

Basis. Let $n = 0$, then $\textsc{Nullable}_n(G) = \varnothing$ and there is no $A \in N$ satisfying the assumptions.

Induction step. Consider a pair $(A, R) \in \textsc{Nullable}_n(G)$. Applying Lemma 5′ gives that $u\langle\varepsilon\rangle \in L_G(A)$ and the last step of its deduction uses a rule

$$A \to \alpha_1 \,\&\, \ldots \,\&\, \alpha_k \,\&\, \lhd D_1 \,\&\, \ldots \,\&\, \lhd D_m \,\&\, \lhdeq E_1 \,\&\, \ldots \,\&\, \lhdeq E_n, \qquad (8)$$

where $R = \{D_1, \ldots, D_m, E_1, \ldots, E_n\} \cup R_1 \cup \ldots \cup R_k$ for some $R_1, \ldots, R_k \subseteq N$ with $(\alpha_i, R_i) \in \textsc{Nullable}^*(G)$ for each $i \in \{1, \ldots, k\}$.

On the other hand, consider a pair $(A, R') \in \textsc{Nullable}_n(G)$. Applying Lemma 5′ gives that $u\langle\varepsilon\rangle \in L_G(A)$, where the last step of its deduction uses a rule

$$A \to \alpha'_1 \,\&\, \ldots \,\&\, \alpha'_{k'} \,\&\, \lhd D'_1 \,\&\, \ldots \,\&\, \lhd D'_{m'} \,\&\, \lhdeq E'_1 \,\&\, \ldots \,\&\, \lhdeq E'_{n'}, \qquad (9)$$

with $R' = \{D'_1, \ldots, D'_{m'}, E'_1, \ldots, E'_{n'}\} \cup R'_1 \cup \ldots \cup R'_{k'}$ for some $R'_1, \ldots, R'_{k'} \subseteq N$ with $(\alpha'_i, R'_i) \in \textsc{Nullable}^*(G)$ for each $i \in \{1, \ldots, k'\}$.

If the rules (8) and (9) are distinct, then the grammar G has ambiguity of choice of rule, which proves the lemma.

Assume that (8) and (9) are the same rule. Then the processing of rule (8) in the construction of the set $\textsc{Nullable}(G)$ has yielded two distinct pairs $(A, R), (A, R') \in \textsc{Nullable}_n(G)$ with different sets of contexts: $R \neq R'$.

Thus, there exist (possibly empty) sets $R_1, \ldots, R_k, R'_1, \ldots, R'_k$ such that

$$(A, \{D_1, \ldots, D_m, E_1, \ldots, E_n\} \cup R_1 \cup \ldots \cup R_k) \in \textsc{Nullable}_n(G)$$

and

$$(A, \{D_1, \ldots, D_m, E_1, \ldots, E_n\} \cup R'_1 \cup \ldots \cup R'_k) \in \textsc{Nullable}_n(G),$$

such that $R_1 \cup \ldots \cup R_k \neq R'_1 \cup \ldots \cup R'_k$, and at least for one $j \in \{1, \ldots, k\}$ it holds that $(\alpha_j, R_j) \neq (\alpha_j, R'_j)$.

Let $\alpha_j = X_1 \ldots X_\ell$ with $X_1, \ldots, X_\ell \in N$. Then for some nonterminal $B = X_i$ it holds that $(B, \widetilde{R}), (B, \widetilde{R}') \in \textsc{Nullable}^*_{n-1}(G)$ with $\widetilde{R}, \widetilde{R}' \subseteq N$ and $\widetilde{R} \neq \widetilde{R}'$. By induction hypothesis, there exist two rules for the nonterminal B such that $x\langle\varepsilon\rangle \in L_G(B)$ (for some $x \in \Sigma^*$). □

The following transformation eliminates all epsilon conjuncts from a given grammar with contexts.

Construction 1. Let $G = (\Sigma, N, P, S)$ be a grammar with contexts, and assume, without loss of generality, that G is of the form obtained in Lemma 6. Consider the set $\textsc{Nullable}(G)$ and construct the following grammar with contexts $G' = (\Sigma, N, P', S)$.

1. Add to P' all rules of the form $A \to a \in P$.

2. Add to P' all rules of the form

$$A \to B_1 \,\&\, \ldots \,\&\, B_k \,\&\, \lhd D_1 \,\&\, \ldots \,\&\, \lhd D_m \,\&\, \lhdeq E_1 \,\&\, \ldots \,\&\, \lhdeq E_n \in P.$$

   Additionally, if $\varepsilon\langle\varepsilon\rangle \in L_G(D_1) \cap \ldots \cap L_G(D_m)$, then add a rule $A \to B_1 \,\&\, \ldots \,\&\, B_k \,\&\, E_1 \,\&\, \ldots \,\&\, E_n \,\&\, \lhd\varepsilon$ to P'.

3. For every rule of the form $A \to BC$, add it to P' together with the following ones:

   (a) $A \to B \,\&\, \lhdeq K_1 \,\&\, \ldots \,\&\, \lhdeq K_t$, for all $(C, \{K_1, \ldots, K_t\}) \in \textsc{Nullable}(G)$;

   (b) $A \to C \,\&\, \lhd K_1 \,\&\, \ldots \,\&\, \lhd K_t$, for all $(B, \{K_1, \ldots, K_t\}) \in \textsc{Nullable}(G)$;

   (c) $A \to C \,\&\, \lhd\varepsilon$, if there exists a nonempty set $\{K_1, \ldots, K_t\} \subseteq N$, such that $(B, \{K_1, \ldots, K_t\}) \in \textsc{Nullable}(G)$ and $\varepsilon\langle\varepsilon\rangle \in L_G(K_1) \cap \ldots \cap L_G(K_t)$.

Theorem 2. *Let $G = (\Sigma, N, P, S)$ be a grammar with contexts. Then the grammar $G' = (\Sigma, N', P', S)$ obtained by Construction 1, generates the language $L(G') = L(G) \setminus \{\varepsilon\}$.*

Proof. $\subseteq$ Show by induction on p, the number of steps used in deduction of an item $[A, u\langle v\rangle]$, that $\vdash_G [A, u\langle v\rangle]$, $A \in N$, $u \in \Sigma^*$, $v \in \Sigma^+$ implies $\vdash_{G'} [A, u\langle v\rangle]$.

Basis. Let $p = 1$. Consider an item $[A, u\langle v\rangle]$ with $v = a \in \Sigma$, which is obtained by some rule $A \to a \in P$. According to Construction 1, the rule $A \to a$ is also in P', and hence $\vdash_{G'} [A, u\langle v\rangle]$.

Induction step. Consider an item $[A, u\langle v\rangle]$ which is deduced in p steos either by some rule $A \to BC \in P$ or by some rule $A \to B_1 \,\&\, \ldots \,\&\, B_k \,\&\, \lhd D_1 \,\&\, \ldots \,\&\, \lhd D_m \,\&\, \lhdeq E_1 \,\&\, \ldots \,\&\, \lhdeq D_n \in P$.

Consider the both possible cases.

Case 1. The item $[A, u\langle v\rangle]$ is deduced by applying the inference rule to the items $[B, u\langle v_1\rangle]$, $[C, uv_1\langle v_2\rangle]$ (with $u, v_1, v_2 \in \Sigma^*$, $v_1 v_2 = v$), each of which is deduced in less than p steps. The following three possibilities have to be considered in this case:

1. Let $v_1 \neq \varepsilon$ and $v_2 \neq \varepsilon$. By induction hypothesis, $\vdash_{G'} [B, u\langle v_1\rangle]$ and $\vdash_{G'} [C, uv_1\langle v_2\rangle]$. By the rule $A \to BC \in P$ (which is added to P' by Construction 1), $[B, u\langle v_1\rangle]$, $[C, uv_1\langle v_2\rangle] \vdash_{G'} [A, u\langle v\rangle]$.

2. Let $v_1 \neq \varepsilon$ and $v_2 = \varepsilon$. That is, $\vdash_G [B, u\langle v_1\rangle]$ and $\vdash_G [C, uv_1\langle\varepsilon\rangle]$. By induction hypothesis, $\vdash_{G'} [B, u\langle v_1\rangle]$. According to Lemma 5′, there exist $K_1, \ldots, K_t \in N$, such that $(C, \{K_1, \ldots, K_t\}) \in \textsc{Nullable}(G)$

and $\varepsilon\langle uv_1\rangle \in L_G(K_i)$ for all $i \in \{1, \ldots, t\}$. Obviously, $\vdash_G [K_1, \varepsilon\langle uv_1\rangle]$, $\ldots$, $[K_t, \varepsilon\langle uv_1\rangle]$. By induction hypothesis, each of these items can be deduced in the grammar G'.

According to Construction 1, the processung of some rule $A \to BC \in P$ with $(C, \{K_1, \ldots, K_t\}) \in \textsc{Nullable}(G)$ adds to P' a rule of the form $A \to B \,\&\, \trianglelefteqslant K_1 \,\&\, \ldots \,\&\, \trianglelefteqslant K_t$, such that $[B, u\langle v_1\rangle]$, $[K_1, \varepsilon\langle uv_1\rangle]$, $\ldots$, $[K_t, \varepsilon\langle uv_1\rangle] \vdash_{G'} [A, u\langle v_1\rangle]$. That is, $\vdash_{G'} [A, u\langle v\rangle]$, since $v_1 = v$.

3. Let $v_1 = \varepsilon$ and $v_2 \neq \varepsilon$. Similarly to the previous case, $\vdash_{G'} [K_1, \varepsilon\langle u\rangle]$, $\ldots$, $[K_t, \varepsilon\langle u\rangle]$, (with $K_1, \ldots, K_t \in N$ such that $(B, \{K_1, \ldots, K_t\}) \in \textsc{Nullable}(G)$), $\vdash_{G'} [C, u\langle v_2\rangle]$, and a rule $A \to C \,\&\, \triangleleft K_1 \,\&\, \ldots \,\&\, \triangleleft K_t$ is added to P', by which $[C, u\langle v_2\rangle]$, $[K_1, \varepsilon\langle u\rangle]$, $\ldots$, $[K_t, \varepsilon\langle u\rangle] \vdash_G [A, u\langle v\rangle]$ (since $v_2 = v$).

Case 2. The item $[A, u\langle v\rangle]$ is deduced by appyling the inference rule to the items $[B_1, u\langle v\rangle]$, $\ldots$, $[B_k, u\langle v\rangle]$, $[D_1, \varepsilon\langle u\rangle]$, $\ldots$, $[D_m, \varepsilon\langle u\rangle]$, $[E_1, \varepsilon\langle uv\rangle]$, $\ldots$, $[E_n, \varepsilon\langle uv\rangle]$ (with $u, v \in \Sigma^*$), each of which is deduced in less than p steps.

By induction hypothesis, each of these items can be deduced in the grammar G'.

By the rule $A \to B_1 \,\&\, \ldots \,\&\, B_k \,\&\, \triangleleft D_1 \,\&\, \ldots \,\&\, \triangleleft D_m \,\&\, \trianglelefteqslant E_1 \,\&\, \ldots \,\&\, \trianglelefteqslant E_n \in P'$ (which is added to P' by Construction 1), $[B_1, u\langle v\rangle]$, $\ldots$, $[B_k, u\langle v\rangle]$, $[D_1, \varepsilon\langle u\rangle]$, $\ldots$, $[D_m, \varepsilon\langle u\rangle]$, $[E_1, \varepsilon\langle uv\rangle]$, $\ldots$, $[E_n, \varepsilon\langle uv\rangle] \vdash_{G'} [A, u\langle v\rangle]$.

$\textcircled{\scriptsize ⩾}$ Show by induction on p, the number of steps used in deduction of an item $[A, u\langle v\rangle]$, that $\vdash_{G'} [A, u\langle v\rangle]$ implies $\vdash_G [A, u\langle v\rangle]$ and $v \neq \varepsilon$.

Basis. Let $p = 1$. Consider an item $[A, u\langle v\rangle]$ with $v = a \in \Sigma$, which is deduced in the grammar G' by some rule $A \to a \in P'$. According to Construction 1, $A \to a \in P$ and $\vdash_G [A, u\langle v\rangle]$.

Induction step. Consider an item $[A, u\langle v\rangle]$ which is deduced in the grammar G' by some rule $p \in P'$.

1. Let $p = B_1 \,\&\, \ldots \,\&\, B_k \,\&\, \triangleleft D_1 \,\&\, \ldots \,\&\, \triangleleft D_m \,\&\, \trianglelefteqslant E_1 \,\&\, \ldots \,\&\, \trianglelefteqslant E_n \in P'$. Then $\vdash_{G'} [B_1, u\langle v\rangle]$, $\ldots$, $[B_k, u\langle v\rangle]$, $[D_1, \varepsilon\langle u\rangle]$, $\ldots$, $[D_m, \varepsilon\langle u\rangle]$, $[E_1, \varepsilon\langle uv\rangle]$, $\ldots$, $[E_n, \varepsilon\langle uv\rangle]$. By induction hypothesis, each of these items is deducible in the grammar G.

   According to Construction 1, the rule p is in P.

   Thus, $[B_1, u\langle v\rangle]$, $\ldots$, $[B_k, u\langle v\rangle]$, $[D_1, \varepsilon\langle u\rangle]$, $\ldots$, $[D_m, \varepsilon\langle u\rangle]$, $[E_1, \varepsilon\langle uv\rangle]$, $\ldots$, $[E_n, \varepsilon\langle uv\rangle] \vdash_G [A, u\langle v\rangle]$.

2. Let $p = B_1 \,\&\, \ldots \,\&\, B_k \,\&\, E_1 \,\&\, \ldots \,\&\, E_n \,\&\, \triangleleft\varepsilon \in P'$. Similarly to the previous case, $\vdash_G [B_1, u\langle v\rangle]$, $\ldots$, $[B_k, u\langle v\rangle]$, $[E_1, u\langle v\rangle]$, $\ldots$, $[E_k, u\langle v\rangle]$.

   Hence, $\vdash_G [A, u\langle v\rangle]$ by the rule p (which is P by Construction 1).

3. Let $p = A \to BC \in P$, implying that $\vdash_G [B, u\langle v_1\rangle]$, $[C, uv_1\langle v_2\rangle]$, with $v_1 v_2 = v$. By induction hypothesis, both of these items can be deduced in G.

   According to Construction 1, p is also contained in the grammar G, and the item $[A, u\langle v\rangle]$ can be inferred: $[B, u\langle v_1\rangle]$, $[C, uv_1\langle v_2\rangle] \vdash_G [A, u\langle v\rangle]$.

4. Let $p = B \,\&\, \trianglelefteq K_1 \,\&\, \ldots \,\&\, \trianglelefteq K_t \in P'$. That is, the items $[B, u\langle v\rangle]$, $[K_1, \varepsilon\langle uv\rangle]$, $\ldots$, $[K_t, \varepsilon\langle uv\rangle]$ can be deduced in both the grammar G and G' (the latter holds by induction hypothesis).

   According to Construction 1, the grammar G should have a rule of the form $A \to BC$, such that $(C, \{K_1, \ldots, K_t\}) \in \textsc{Nullable}(G)$. Then, according to Lemma 5′, $\vdash_G [C, uv\langle\varepsilon\rangle]$.

   By the rule $A \to BC \in P$, $[B, u\langle v\rangle]$, $[C, uv\langle\varepsilon\rangle] \vdash_G [A, u\langle v\rangle]$.

5. Let $p = B \,\&\, \triangleleft K_1 \,\&\, \ldots \,\&\, \triangleleft K_t \in P'$. Similarly to the previous case, one can conclude that $\vdash_{G'} [C, u\langle v\rangle]$, $[K_1, \varepsilon\langle u\rangle]$, $\ldots$, $[K_t, \varepsilon\langle u\rangle]$ (with $K_1, \ldots, K_t \in N$ such that $(B, \{K_1, \ldots, K_t\}) \in \textsc{Nullable}(G)$) and $\vdash_G [B, u\langle\varepsilon\rangle]$.

   By the rule $A \to BC \in P$, $[B, u\langle\varepsilon\rangle]$, $[C, u\langle v\rangle] \vdash_G [A, u\langle v\rangle]$.

6. Let $p = C \,\&\, \triangleleft\varepsilon \in P'$. That is, the item $[C, \varepsilon\langle v\rangle]$ is deducible in both grammars G' and G (by induction hypothesis).

   Similarly to the previous cases, $[B, u\langle\varepsilon\rangle]$, $[C, u\langle v\rangle] \vdash_G [A, u\langle v\rangle]$, by the rule $A \to BC$ (which is in P by Construction 1).

□

Lemma 8. *Construction 1 preserves unambiguity of the grammar.*

Proof. Let $G = (\Sigma, N, P, S)$ be a grammar with contexts to be transformed by Construction 1 to an equivalent epsilon-free grammar $G' = (\Sigma, N, P', S)$. Assume, without loss of generality, that G is of the form obtained in Lemma 6.

Since the steps 1 and 2 of Construction 1 do not add to the set P' any rules which are not already in P, the ambiguity of the grammar is preserved.

Consider a rule of the form $A \to BC \in P$. Suppose that some string $u\langle v\rangle$ is generated by more than one rule of the forms (3a)–(3c), which can be added to P' according to step 3 of Construction 1.

Then the following cases are possible.

1. *The string $u\langle v\rangle$ is generated by two rules of the form (3a).* That is, $(C, R), (C, R') \in \textsc{Nullable}(G)$ with $R, R' \subseteq N$ and $R \neq R'$. By Lemma 5, $\varepsilon\langle u\rangle \in L_G(K)$ for all $K \in R \cup R'$. Applying Lemma 7 to $(C, R), (C, R') \in \textsc{Nullable}(G)$ gives that the grammar G is ambiguous, which is a contradiction to the assumptions.

2. *The string $u\langle v\rangle$ is generated by two rules of the form (3b).* That is, $(B, R), (B, R') \in \textsc{Nullable}(G)$ with $R, R' \subseteq N$ and $R \neq R'$. By Lemma 5, $\varepsilon\langle u\rangle \in L_G(K)$ for all $K \in R \cup R'$. Applying Lemma 7 to $(B, R), (B, R') \in \textsc{Nullable}(G)$ gives that the grammar G is ambiguous, which is a contradiction to the assumptions.

3. *The string $u\langle v\rangle$ is generated by two rules, one of the form (3a) and the other of the form (3b).* This directly implies $u\langle v\rangle \in L_G(B)$ and $u\langle v\rangle \in L_G(C)$. By the construction of these two rules, $(C, R), (B, R') \in \textsc{Nullable}(G)$ with $R, R' \subseteq N$. The two following cases are possible.

   - Let $C = B$ and $R \neq R'$. Then applying Lemma 7 to $(C, R), (C, R') \in \textsc{Nullable}(G)$ gives that the grammar G is ambiguous, which is a contradiction to the assumptions.
   - Let $C \neq B$. Since $(C, R) \in \textsc{Nullable}(G)$ and $\varepsilon\langle uv\rangle \in L_G(K)$ for all $K \in R$, then $uv\langle\varepsilon\rangle \in L_G(C)$ by Lemma 5. Similarly, by Lemma 5, $(B, R') \in \textsc{Nullable}(G)$, and $\varepsilon\langle u\rangle \in L_G(K)$ for all $K \in R'$ imply that $u\langle\varepsilon\rangle \in L_G(B)$.

     Therefore, the grammar G has an ambiguity of concatenation for the string $u\langle v\rangle = u\langle\varepsilon\rangle{\cdot}u\langle v\rangle = u\langle v\rangle{\cdot}uv\langle\varepsilon\rangle$, with $u\langle\varepsilon\rangle, u\langle v\rangle \in L_G(B)$ and $u\langle v\rangle, uv\langle\varepsilon\rangle \in L_G(C)$.

4. *The string $u\langle v\rangle$ is generated by two rules, one of the form (3b) and the other of the form (3c).* The latter rule requires $u = \varepsilon$. Furthermore, the rule of the form (3b) may not have any contexts, since $\varepsilon\langle\varepsilon\rangle \notin L_{G'}(K)$ for all $K \in R$. That is, $(B, \varnothing), (B, R) \in \textsc{Nullable}(G)$ with $R \subseteq N$ and $\varepsilon\langle\varepsilon\rangle \in L_G(K)$ for all $K \in R$. Applying Lemma 7 to $(B, \varnothing), (B, R) \in \textsc{Nullable}(G)$ and the string $\varepsilon\langle\varepsilon\rangle$ gives that the grammar G is ambiguous, which is a contradiction to the assumptions.

5. *The string $u\langle v\rangle$ is generated by two rules, one of the form (3a) and the other of the form (3c).* This implies that $u\langle v\rangle \in L_G(B)$ and $u\langle v\rangle \in L_G(C)$, since the latter rule requires $u = \varepsilon$. By the construction of the two rules, $(C, R), (B, R') \in \textsc{Nullable}(G)$ and $\varepsilon\langle\varepsilon\rangle \in L_G(K)$ for all $K \in R'$.

   Since $(C, R) \in \textsc{Nullable}(G)$ and $\varepsilon\langle v\rangle \in L_G(K)$ for all $K \in R$, by Lemma 5, $v\langle\varepsilon\rangle \in L_G(C)$. Similarly, by Lemma 5, $(B, R') \in \textsc{Nullable}(G)$ and $\varepsilon\langle\varepsilon\rangle \in L_G(K)$ for all $K \in R'$. This implies that the string $\varepsilon\langle\varepsilon\rangle$ is in $L_G(B)$.

   Thus, there is an ambiguity of concatenation in the grammar G: $\varepsilon\langle v\rangle = \varepsilon\langle v\rangle{\cdot}v\langle\varepsilon\rangle = \varepsilon\langle\varepsilon\rangle{\cdot}\varepsilon\langle v\rangle$, with $\varepsilon\langle v\rangle, \varepsilon\langle\varepsilon\rangle \in L_G(B)$ and $v\langle\varepsilon\rangle, \varepsilon\langle v\rangle \in L_G(C)$.

□

The second stage of the transformation to the normal form is removing the *unit conjuncts* in rules of the form $A \to B\& \ldots$. Already for conjunctive grammars, the only known transformation involves substituting all rules for B into all rules for A, and results in a worst-case exponential blowup. The same construction applies verbatim for grammars with contexts.

Lemma 9. *Let $G = (\Sigma, N, P, S)$ be a grammar with contexts without epsilon conjuncts. Let*

$$A \to B \,\&\, \beta_2 \,\&\, \ldots \,\&\, \beta_k \,\&\, \lhd D_1 \,\&\, \ldots \,\&\, \lhd D_m \,\&\, \trianglelefteqslant E_1 \,\&\, \ldots \,\&\, \trianglelefteqslant E_n \tag{10}$$

be any rule with a unit conjunct B.

If $A \neq B$, let $B \to \alpha_{i1} \,\&\, \ldots \,\&\, \alpha_{ik_i} \,\&\, \lhd D_{i1} \,\&\, \ldots \,\&\, \lhd D_{im_i} \,\&\, \&\, \trianglelefteqslant E_{i1} \,\&\, \ldots \,\&\, \trianglelefteqslant E_{in_i}$, with $1 \leqslant i \leqslant r$, be all rules for nonterminal B. Then the rule (10) can be replaced with a collection of rules of the form

$$\begin{aligned} &A \to \alpha_{i1} \,\&\, \ldots \,\&\, \alpha_{ik_i} \,\&\, \lhd D_{i1} \,\&\, \ldots \,\&\, \lhd D_{im_i} \,\&\, \trianglelefteqslant E_{i1} \,\&\, \ldots \,\&\, \trianglelefteqslant E_{in_i} \,\&\\ &\&\, \beta_2 \,\&\, \ldots \,\&\, \beta_k \,\&\, \lhd D_1 \,\&\, \ldots \,\&\, \lhd D_m \,\&\, \trianglelefteqslant E_1 \,\&\, \ldots \,\&\, \trianglelefteqslant E_n \quad (\textit{for } 1 \leqslant i \leqslant r) \end{aligned}$$

without altering the language generated by the grammar.

If $A = B$, then the rule (10) can be removed.

In both cases, if G is unambiguous, then so is the resulting grammar.

Altogether, the transformations defined in this section lead to the following normal form, which extends the binary normal form of conjunctive grammars [11].

Definition 7. *A grammar with contexts $G = (\Sigma, N, P, S)$ is said to be in the binary normal form if each rule in P is in one of the forms*

$$A \to B_1C_1 \,\&\, \ldots \,\&\, B_kC_k \,\&\, \lhd D_1 \,\&\, \ldots \,\&\, \lhd D_m \,\&\, \trianglelefteqslant E_1 \,\&\, \ldots \,\&\, \trianglelefteqslant E_n \tag{11a}$$

$$A \to a \,\&\, \lhd D_1 \,\&\, \ldots \,\&\, \lhd D_m \,\&\, \trianglelefteqslant E_1 \,\&\, \ldots \,\&\, \trianglelefteqslant E_n \tag{11b}$$

$$A \to B_1C_1 \,\&\, \ldots \,\&\, B_kC_k \,\&\, \lhd\varepsilon \tag{11c}$$

$$A \to a \,\&\, \lhd\varepsilon \tag{11d}$$

Theorem 3. *For each grammar with contexts $G = (\Sigma, N, P, S)$ there exists and can be effectively constructed a grammar with contexts $G' = (\Sigma, N', P', S)$ in the binary normal form, such that $L(G) = L(G')$. The construction preserves unambiguity of the grammar.*

Proof. First, the grammar is transformed using Lemma 6. Then epsilon conjuncts are eliminated according to Theorem 2, and unit conjuncts are removed by applying the transformation in Lemma 9 multiple times, as in the case of conjunctive grammars [11, Lem. 2]. Finally, if $\varepsilon \in L(G)$, then a new start symbol S' is introduced, and a rule $S' \to \varepsilon$ is added to P'. For each rule $S \to \alpha_1 \,\&\, \ldots \,\&\, \alpha_k \,\&\, \lhd D_1 \,\&\, \ldots \,\&\, \lhd D_m \,\&\, \trianglelefteqslant E_1 \,\&\, \ldots \,\&\, \trianglelefteqslant E_n \in$

P', a rule $S' \to \alpha_1 \,\&\, \ldots \,\&\, \alpha_k \,\&\, \lhd D_1 \,\&\, \ldots \,\&\, \lhd D_m \,\&\, \trianglelefteq E_1 \,\&\, \ldots \,\&\, \trianglelefteq E_n$ is be added to P', as well.

If the grammar G is unambiguous, then so is the grammar G', since each step of the transformation preserves unambiguity (by virtue of Lemmata 8 and 9). □

5 Parsing algorithm

5.1 General parsing algorithm

For each grammar with one-sided contexts in the binary normal form, there exists a parsing algorithm that determines the membership of all substrings of the input string in the languages generated by all nonterminals of the grammar, storing the results in a table. This algorithm elaborates a similar procedure for conjunctive grammars [11], which in turn generalizes the Cocke–Kasami–Younger algorithm for standard context-free grammars.

Let $G = (\Sigma, N, P, S)$ be a grammar with contexts in the binary normal form, and let $w = a_1 \ldots a_n \in \Sigma^+$ with $n \geqslant 1$ and $a_k \in \Sigma$ be some string. For every two positions i, j, with $0 \leqslant i < j \leqslant n$, define

$$T_{i,j} = \{\, A \mid A \in N,\, \vdash_G [A, a_1 \ldots a_i \langle a_{i+1} \ldots a_j \rangle] \,\}$$

The string $w \in L(G)$ if and only if $S \in T_{0,n}$.

Theorem 4. *For every grammar with one-sided contexts G in the binary normal form, Algorithm 1 correctly determines the membership of a given string $w = a_1 \ldots a_n$ in $L(G)$, and does so in time $\mathcal{O}(|G|^2 \cdot n^3)$, using space $\mathcal{O}(|G| \cdot n^2)$.*

From the loops in lines 3, 4, 8 and 10, it is evident that the innermost statement in line 11 is executed $\mathcal{O}(|G| \cdot n^3)$ times. In order to do the calculations in line 11 in time proportional to $|G|$, one can replace the Cartesian product by considering only the conjuncts occurring somewhere in the grammar. The memory requirements of the algorithm are given by the size of the sets $T_{i,j}$.

Example 4. Consider the grammar from Example 1 generating the language $\{\, a^n b^n c^n d^n \mid n \geqslant 0 \,\}$. It can be transformed to the following grammar in the binary normal form:

Algorithm 1. Let $G = (\Sigma, N, P, S)$ be a grammar with contexts in the binary normal form. Let $w = a_1 \dots a_n \in \Sigma^+$ (with $n \geqslant 1$ and $a_i \in \Sigma$) be the input string. For all i, j with $0 \leqslant i < j \leqslant n$, compute $T_{i,j}$ as follows.

1: **if** $A \to a_1 \,\&\, \lhd\varepsilon \in P$ or $A \to a_1 \in P$ **then**
2: $\quad T_{0,1} = T_{0,1} \cup \{A\}$
3: **for** $j = 1, \dots, n$ **do**
4: $\quad$ **while** $T_{0,j}$ changes **do**
5: $\quad\quad$ **for all** $A \to a \,\&\, \lhd D_1 \,\&\, \dots \,\&\, \lhd D_{m'} \,\&\, \lhdeq E_1 \,\&\, \dots \,\&\, \lhdeq E_{m''} \in P$ **do**
6: $\quad\quad\quad$ **if** $a_j = a$ and $D_1, \dots, D_{m'} \in T_{0,j-1}$ and $E_1, \dots, E_{m''} \in T_{0,j}$ **then**
7: $\quad\quad\quad\quad T_{j-1,j} = T_{j-1,j} \cup \{A\}$
8: $\quad\quad$ **for** $i = j - 2$ to 0 **do**
9: $\quad\quad\quad$ let $R = \varnothing$ $(R \subseteq N \times N)$
10: $\quad\quad\quad$ **for** $k = i + 1$ to $j - 1$ **do**
11: $\quad\quad\quad\quad R = R \cup (T_{i,k} \times T_{k,j})$
12: $\quad\quad\quad$ **for all** $A \to B_1 C_1 \,\&\, \dots \,\&\, B_m C_m \,\&\, \lhd D_1 \,\&\, \dots \,\&\, \lhd D_{m'} \,\&\, \lhdeq E_1 \,\&\, \dots \,\&\, \lhdeq E_{m''}$ **do**
13: $\quad\quad\quad\quad$ **if** $(B_1, C_1), \dots, (B_m, C_m) \in R$, $D_1, \dots, D_{m'} \in T_{0,i}$, and $E_1, \dots, E_{m''} \in T_{0,j}$ **then**
14: $\quad\quad\quad\quad\quad T_{i,j} = T_{i,j} \cup \{A\}$
15: $\quad\quad\quad$ **for all** $A \to B_1 C_1 \,\&\, \dots \,\&\, B_m C_m \,\&\, \lhd\varepsilon$ **do**
16: $\quad\quad\quad\quad$ **if** $(B_1, C_1), \dots, (B_m, C_m) \in R$ and $i = 0$ **then**
17: $\quad\quad\quad\quad\quad T_{i,j} = T_{i,j} \cup \{A\}$
18: accept if and only if $S \in T_{0,n}$

$$
\begin{aligned}
S &\rightarrow S_a D_0 \mid S_b C_0 \\
A &\rightarrow A_a B_0 \\
S_a &\rightarrow A_0 S \mid a \,\&\, \trianglelefteqslant A \\
S_b &\rightarrow B_0 S \mid b \,\&\, \trianglelefteqslant A \\
A_a &\rightarrow A_0 A \mid a \\
A_0 &\rightarrow a \\
B_0 &\rightarrow b \\
C_0 &\rightarrow c \\
D_0 &\rightarrow d
\end{aligned}
$$

The parsing table constructed by Algorithm 1 on the input $w = aabbccdd$ is given in Figure 2. The arrows in the figure indicate the logical dependencies, and the dotted arrow refers to the dependence defined by the conjunct $\trianglelefteqslant A$.

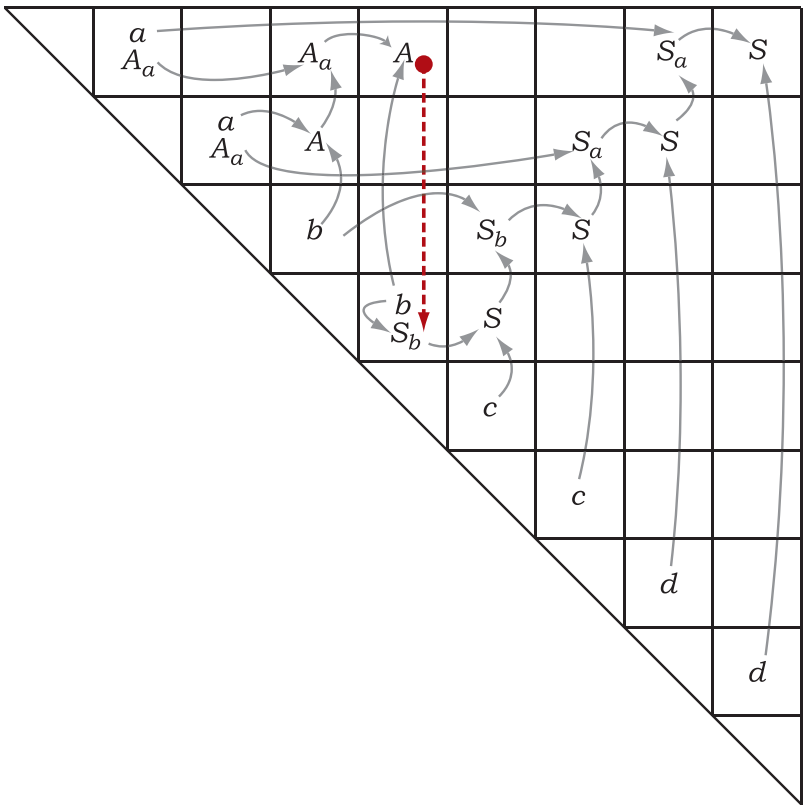

Figure 2: Parsing table of string $w = aabbccdd$.

5.2 Parsing unambiguous grammars

The parsing algorithm for unambiguous grammars with contexts uses dynamic programming technique to construct a two-dimensional table T indexed by positions in the input and nonterminal symbols. Each entry of the table assumes the value of a set of positions in the input string, which are stored as a list in an ascending order. The element corresponding to a position k (with $1 \leqslant k \leqslant n$) and a nonterminal $A \in N$ is denoted by $T_k[A]$.

By definition, a position i should be in $T_k[A]$ if and only if $0 \leqslant i < k$ and $a_1 \dots a_i \langle a_{i+1} \dots a_k \rangle \in L_G(A)$, where $a = a_1 \dots a_n$ is the input string. In the end of the computation, each list $T_k[A]$ will contain exactly these numbers. The input string $a_1 \dots a_n \in L(G)$ if and only if the position 0 is in $T_n[S]$.

The first loop (line 2) handles the substring a_1 of the input string, since that is the only possible substring whose context is empty.

Each jth iteration of the outer loop (line 5) determines the membership of substrings ending at jth position in $L_G(A)$, for all $A \in N$.

The first nested loop (line 7) handles substrings of length 1 (that is, $a_1, \dots, a_n$) and stores in $T_j[A]$ the value 0 if $a_j \in L_G(A)$.

Substrings of greater lengths ending at jth position are processed in the second nested loop by k (line 14). This loop constructs an auxiliary data structure R: for each $i \in \{0, \dots, j-2\}$, the element R_i contains all pairs $(B, C) \in \mathcal{U}$, for which the substring $a_1 \dots a_i \langle a_{i+1} \dots a_j \rangle$ is in $L_G(BC)$.

The kth iteration of this loop is denoted by (j, k); it handles the substrings $a_1 \dots a_i \langle a_{i+1} \dots a_j \rangle$ of various lengths, with $i + 1 \in \{1, \dots, k\}$. The goal is to determine all such substrings $u\langle vw \rangle$, which belong to $L_G(BC)$ for some $(B, C) \in \mathcal{U}$, and in which the middle point in their factorization into $u\langle v \rangle \in L_G(B)$ and $uv\langle w \rangle \in L_G(C)$ is exactly $k + 1$, that is, the first part—$u\langle v \rangle$—ends at the position k and the second part—$uv\langle w \rangle$—starts at the position $k+1$. The substrings $u\langle vw \rangle$ are identified by first considering the appropriate conjunct, then checking the membership of the second substring in $L_G(C)$, and finally by enumerating all appropriate first parts using the data stored in $T_k[B]$. This is used to fill the elements of R, namely $R_{k-1}, R_{k-2}, \dots, R_0$, with appropriate conjuncts. An element R_{k-1} gets completely filled after of iteration (j, k).

The proof of correctness of the algorithm has to establish the following three claims:

1. The implementation of $T_j[A]$ by lists faithfully represents the high-level set operations.

2. The algorithm is a correct recognizer, that is, it accepts a string w if and only if $w \in L(G)$.

3. The algorithm runs in time $\mathcal{O}(n^2)$ for unambiguous grammars with contexts, and in time $\mathcal{O}(n^3)$ for arbitrary ones.

Algorithm 2. Let $G = (\Sigma, N, P, S)$ be a grammar with one-sided contexts in the binary normal form. Let $w = a_1 \dots a_n \in \Sigma^+$ $(n \geqslant 1)$ be an input string. Let $\mathcal{U} = \{\, (B, C) \mid A \to BC \,\&\, \dots \in P \,\}$. For each $j \in \{1, \dots, n\}$, let $T_j[A]$ be a variable ranging over the subsets of $\{0, \dots, j-1\}$. Let R_k range over the subsets of $\mathcal{U}$, for each $k \in \{0, \dots, n-1\}$.

1: let $T_j[A] = \varnothing$ for all $j = 1, \dots, n$, $A \in N$
2: **for all** $A \to a \,\&\, \lhd\varepsilon$ **do**
3: **if** $a_1 = a$ **then**
4: $T_1[A] = \{0\}$
5: **for** $j = 1$ to n **do**
6: **while** T changes **do**
7: **for all** $A \in N$ **do**
8: **if** $A \to a \,\&\, \lhd D_1 \,\&\, \dots \,\&\, \lhd D_{m'} \,\&\, \lhdeq E_1 \,\&\, \dots \,\&\, \lhdeq E_{m''} \in P$ **then**
9: **if** $0 \in T_{j-1}[D_1] \cap \dots \cap T_{j-1}[D_{m'}]$ and $0 \in T_j[E_1] \cap \dots \cap T_j[E_{m''}]$ **then**
10: $T_j[A] = j - 1$
11: **else**
12: $T_j[A] = \varnothing$
13: let $R_k = \varnothing$ for all $k = 0, \dots, j-1$
14: **for** $k = j - 1$ to 1 **do**
15: **for all** $(B, C) \in \mathcal{U}$ **do**
16: **if** $k \in T_j[C]$ **then**
17: **for all** $i \in T_k[B]$ **do**
18: $R_i = R_i \cup \{(B, C)\}$
19: **for all** $A \in N$ **do**
20: **for all** $A \to B_1C_1 \,\&\, \dots B_mC_m \,\&\, \lhd D_1 \,\&\, \dots \,\&\, \lhd D_{m'} \,\&\, \,\&\, \lhdeq E_1 \,\&\, \dots \,\&\, \lhdeq E_{m''} \in P$ **do**
21: **if** $(B_1, C_1), \dots, (B_m, C_m) \in R_{k-1}$ and $0 \in T_{k-2}[D]$ (for all $D \in \{D_1, \dots, D_{m'}\}$) and $0 \in T_j[E]$ (for all $E \in \{E_1, \dots, E_{m''}\}$) **then**
22: $T_j[A] = T_j[A] \cup \{k - 1\}$
23: **for all** $A \to B_1C_1 \,\&\, \dots B_mC_m \,\&\, \lhd\varepsilon$ **do**
24: **if** $(B_1, C_1), \dots, (B_m, C_m) \in R_0$ **then**
25: $T_j[A] = T_j[A] \cup \{0\}$
26: accept if and only if $0 \in T_n[S]$

This can be done similarly to the paper of Okhotin [17].

The following lemma [17] claims the truth of the first condition.

Lemma 10 ([17]). *Each list $T_j[A]$ always remains sorted. Each time the algorithm checks the condition in line 16, every set $T_j[A]$ does not contain elements less than k. Each time the algorithm is about to execute the line 22 or 25, the set $T_j[A]$ does not contain elements less than k.*

The correctness of the algorithm is verbatim to the case of conjunctive grammars [17].

Lemma 11 ([17]). *For every grammar with contexts in the binary normal form, in the computation of the algorithm 2 on an input string $w \in \Sigma^+$,*

1. *after iteration j, for each $A \in N$ and for each $\ell \in \{1, \ldots, j\}$,*

$$T_\ell[A] = \{\, i \mid 0 \leqslant i < \ell,\ a_1 \ldots a_i \langle a_{i+1} \ldots a_\ell \rangle \in L_G(A) \,\};$$

2. *after iteration (j, k), for each $A \in N$,*

$$T_j[A] = \{\, i \mid k - 1 \leqslant i < j,\ a_1 \ldots a_i \langle a_{i+1} \ldots a_\ell \rangle \in L_G(A) \,\};$$

3. *after iteration (j, k), for each i with $0 \leqslant i < j$,*

$$R_i = \big\{ (B, C) \in \mathcal{U} \;\big|\; \exists \ell : k \leqslant \ell < j, a_1 \ldots a_i \langle a_{i+1} \ldots a_\ell \rangle \in L_G(B), \\ a_1 \ldots a_\ell \langle a_{\ell+1} \ldots a_j \rangle \in L_G(C) \big\}.$$

The following lemma [17] states the time complexity of the algorithm 2 on unambiguous grammars with contexts.

Lemma 12 ([17]). *Let $G = (\Sigma, N, P, S)$ be grammar with contexts in the binary normal form, and let $w = a_1 \ldots a_n$ be the input string. Assume that G satisfies Condition II in the definition 3 of an unambiguous grammar. Then Algorithm 2, implemented on a random access machine, runs in time $\mathcal{O}(n^2)$. If the grammar G is ambiguous, the algorithm runs in time $\mathcal{O}(n^3)$.*

Since for every grammar with contexts one can construct an equivalent grammar in the binary normal form, the following result can be stated.

Theorem 5. *For every unambiguous grammar with contexts $G = (\Sigma, N, P, S)$ there exists an algorithm to test the membership of given strings in $L(G)$ in time $\mathcal{O}(n^2)$.*

Example 5. Consider the grammar with contexts from Example 1 and the equivalent grammar in the binary normal form presented in Example 4. The grammar is clearly unambiguous.

The parsing table constructed by Algorithm 2 is given in Figure 3.

	S	A	S_a	S_b	A_a	A_0	B_0	C_0	D_0
$a_1 = a$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\{0\}$	$\{0\}$	$\varnothing$	$\varnothing$	$\varnothing$
$a_2 = a$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\{1\}$	$\{1\}$	$\varnothing$	$\varnothing$	$\varnothing$
$a_3 = b$	$\varnothing$	$\{1\}$	$\varnothing$	$\varnothing$	$\{0\}$	$\varnothing$	$\{2\}$	$\varnothing$	$\varnothing$
$a_4 = b$	$\varnothing$	$\{0\}$	$\varnothing$	$\{3\}$	$\varnothing$	$\varnothing$	$\{3\}$	$\varnothing$	$\varnothing$
$a_5 = c$	$\{3\}$	$\varnothing$	$\varnothing$	$\{2\}$	$\varnothing$	$\varnothing$	$\varnothing$	$\{4\}$	$\varnothing$
$a_6 = c$	$\{2\}$	$\varnothing$	$\{1\}$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\{5\}$	$\varnothing$
$a_7 = d$	$\{1\}$	$\varnothing$	$\{0\}$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\{6\}$
$a_8 = d$	$\{0\}$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\{7\}$

Figure 3: Parsing table of string $w = aabbccdd$ constructed by Algorithm 2.

Elements located at the intersection of the columns A_0, B_0, C_0, and D_0 and rows $a_1, \ldots, a_8$ of the parsing table represent the substrings a, b, c, and d of the input string, respectively. Thus, for example, the set $\{2\}$ located in the position (B_0, a_3) of the table corresponds to the fact that the substring starting from position $i = 2+1$ and ending at position $j = 3$ has the property B_0.

Similarly, a set $\{2\}$ located in the position (S_b, a_4) states that the string $a_1 \ldots a_3\langle a_{3+1} \ldots a_4\rangle = a_1 \ldots a_3\langle a_4\rangle = aab\langle b\rangle$ has the property S_b (by the rule $S_b \to b \,\&\, \preccurlyeq A$). The context $\preccurlyeq A$ in that rule is satisfied, since $\{0\}$ is in the position (A, a_4) of the parsing table, meaning that $\varepsilon\langle aabb\rangle \in L_G(A)$.

The set $\{1\}$ located in the position (S_a, a_6) means that the string $a_1\langle a_{1+1} \ldots a_6\rangle = a_1\langle a_2 \ldots a_6\rangle = a\langle abbcc\rangle$ has the property S_a (by the rule $S_a \to A_0 S$).

Finally, $\{0\}$, located in the position (S, a_8) of the table, means that the string $a_0\langle a_{0+1} \ldots a_8\rangle = \varepsilon\langle a_1 \ldots a_8\rangle = \varepsilon\langle w\rangle$ has the desired property S. Therefore, the string $w = aabbccdd$ is in the language $L(G)$.

6 Recognition in linear space

The cubic-time algorithm in Section 5 uses quadratic space, as do its context-free and conjunctive prototypes. For conjunctive grammars, the membership of a string can be recognized in linear space and exponential time [13] by using deterministic rewriting of terms of a linearly bounded size. In this section, this method is extended to handle the case of grammars with one-sided contexts.

Let $G = (\Sigma, N, P, S)$ be a grammar with one-sided contexts and let $w = a_1 \ldots a_n$ be an input string. The linear-space parsing algorithm presented below constructs the sets $T_{0,1}$, $T_{0,2}$, $\ldots$, $T_{0,n}$, as in the top row of the table in Algorithm 1, with

$$T_{0,i} = \{\, A \in N \mid \varepsilon\langle a_1 \ldots a_i\rangle \in L_G(A) \,\}.$$

The membership of symbols in each set $T_{0,\ell}$ is computed using term rewriting similar to the one used for conjunctive grammars [13], which refers to the previously computed sets $T_{0,1}, \ldots, T_{0,\ell-1}$, as well as to the partially computed set $T_{0,\ell}$.

Consider every prefix $a_1 \ldots a_\ell$ of the input string. For every symbol $A \in N$, define its *height* $h_\ell(A)$ as follows. Consider the shortest deduction of $[A, \varepsilon\langle a_1 \ldots a_\ell \rangle]$, and let $A \to B_1 C_1 \,\&\, \ldots \,\&\, B_m C_m \,\&\, \lhd D_1 \,\&\, \ldots \,\&\, \lhd D_{m'} \,\&\, \trianglelefteq E_1 \,\&\, \ldots \,\&\, \trianglelefteq E_{m''}$ be the rule used at the last step of this deduction. If $m'' = 0$, then $h_\ell(A)$ is defined as 0; otherwise, $h_\ell(A) = \max\big(h_\ell(E_1), \ldots, h_\ell(E_{m''})\big) + 1$.

Lemma 13. *There exists a deterministic term-rewriting procedure, which, given a string $a_1 \ldots a_\ell$ with $a_i \in \Sigma$, a nonterminal $A \in N$, the sets $T_{0,i} = \{\, D \in N \mid \varepsilon\langle a_1 \ldots a_i \rangle \in L_G(D) \,\}$ with $i \in \{1, \ldots, \ell - 1\}$ and the partially constructed set $T_{0,\ell}$, given as $X = \{\, E \in N \mid \varepsilon\langle a_1 \ldots a_\ell \rangle \in L_G(E),\ h_\ell(E) < h_0 \,\}$ for some $h_0 \geqslant 0$, determines whether $\varepsilon\langle a_1 \ldots a_\ell \rangle \in L_G(A)$ and $h_\ell(A) \leqslant h_0$, and does so using linearly bounded space.*

The procedure tests the conjunction of these two conditions. If $\varepsilon\langle a_1 \ldots a_\ell \rangle \notin L_G(A)$ or $h_\ell(A) > h_0$, in both cases it will give a negative answer.

The rewriting system claimed by Lemma 13 is constructed as follows. Let $G = (\Sigma, N, P, S)$ be a grammar with contexts. Assume, without loss of generality, that it is in the binary normal form and that the set of rules P is linearly ordered.

Let P' be the set of rules of G with marked conjuncts, that is $P' = \{\, p_i \mid p \text{ is of the form (11a) or (11c)}, 1 \leqslant i \leqslant k \,\}$. Let $N^+ = \{\, A^+ \mid A \in N \,\}$ and $N^- = \{\, A^- \mid A \in N \,\}$.

For every $p = A \to \varphi \in P$, denote $L(p) = L(\varphi)$; if the k-th conjunct of p is an unquantified conjunct BC, denote $L(p_k) = L_G(BC)$.

Terms are represented as strings over the alphabet $\Sigma \cup N \cup N^+ \cup N^- \cup P' \cup \{\, \text{“(”}, \text{“)”} \,\}$ as follows:

- for every $u \in \Sigma^+$ and $A \in N$, the following are terms for A: $A(u)$, $A^+(u)$, $A^-(u)$.
- if p_i is a conjunct of the form BC and t_1, t_2 are terms for B and C respectively, then $p_i(t_1 t_2)$ is a term for A.

The *string value* of a term t is defined as follows:

- $\sigma(A(u)) = \sigma(A^+(u)) = \sigma(A^-(u)) = u$ for all $u \in \Sigma^+$;
- $\sigma(p_i(t_1 t_2)) = \sigma(t_1) \cdot \sigma(t_2)$.

Consider any term $\hat{t}(t)$, where t is a distinguished subterm. The *left context* of this subterm within its parent term is defined by replacing t with

a special marker #, and then obtaining the string value of all symbols to the left of this marker. This value is given by $\lambda(\hat{t}(\#))$, which is inductively defined as follows:

$$\begin{aligned} \lambda(p_i(t_1 t_2(\#))) &= \sigma(t_1)\lambda(t_2(\#)) \\ \lambda(p_i(t_1(\#)t_2)) &= \lambda(t_1(\#)) \\ \lambda(\#) &= \varepsilon \end{aligned}$$

The notion of a *true term with a distinguished subterm* (understood as a pair) is defined inductively on the size of the subterm:

- a term $\hat{t}(t)$ with a subterm $t = A(u)$ is always true;
- a term $\hat{t}(t)$ with a subterm $t = A^+(u)$ is true if and only if $\lambda(\hat{t}(\#)\langle u\rangle \in L(A)$;
- a term $\hat{t}(t)$ with a subterm $A^-(u)$ is true if and only if $\lambda(\hat{t}(\#)\langle u\rangle \notin L(A)$;
- a term $\hat{t}(t)$ with a subterm $t = p_i(t_1 t_2)$ is true if and only if all of the following conditions hold:

  I. both subterms t_1 and t_2 of the term $\hat{t}(p_i(t_1 t_2))$ are true;

  II. for every rule r for A that precedes p it holds that $\lambda(\hat{t}(\#))\langle\sigma(t)\rangle \notin L(r)$;

  III. for every conjunct p_j of the rule p that precedes p_i, it holds that $\lambda(\hat{t}(\#))\langle\sigma(t)\rangle \in L(p_j)$;

  IV. for every factorization $\sigma(t_1)\sigma(t_2) = uv$, with $0 < |u| < |\sigma(t_1)|$ it holds that $\lambda(\hat{t}(\#))\langle u\rangle \notin L(B)$ or $\lambda(\hat{t}(\#))u\langle v\rangle \notin L(C)$, where the conjunct p_i is of the form BC.

Now the set of *rewriting rules* can be defined.

1. In a term $\hat{t}(A(a))$, its subterm $A(a)$ with $a \in \Sigma$ is rewritten as follows.

   If there exists a rule of the form $A \to a \,\&\, \lhd D_1 \,\&\, \ldots \,\&\, \lhd D_m \,\&\, \trianglelefteq E_1 \,\&\, \ldots \,\&\, \trianglelefteq E_n$ in P, for which $D_1, \ldots, D_m \in T_{0,d}$ (with $d = |\lambda(\hat{t}(\#))|$) and $E_1, \ldots, E_n \in T_{0,e}$ (with $e = |\lambda(\hat{t}(\#))\sigma(A(a))|$), then the subterm $A(a)$ is rewritten with $A^+(a)$.

   If the grammar contains the rule $A \to a \,\&\, \lhd\varepsilon$ and $\lambda(\hat{t}(\#)) = \varepsilon$, then $A(a)$ is also rewritten with $A^+(a)$.

   If there are no rules satisfying the above conditions, then the subterm $A(a)$ is rewritten with $A^-(a)$.

2. Consider a term $\hat{t}(A(u))$ with a subterm $A(u)$ with $u = a_j a_{j+1} \dots a_\ell$. If there are no rules of the form (11a) or (11c) for A, this subterm is rewritten with $A^-(u)$. Otherwise, let p be the first such rule and let $p_1 = BC$ be its first conjunct. Then the subterm $A(u)$ is rewritten with $p_1(B(a_j)C(a_{j+1} \dots a_\ell))$.

3. In a term $\hat{t}(p_i(B^+(u)C^+(v)))$, its subterm $p_i(B^+(u)C^+(v))$ is rewritten as follows.

   - If the ith conjunct of p_i is the last unquantified conjunct in this rule, then do as follows. If there exists a rule of the form $A \to B_1C_1 \,\&\, \dots \,\&\, B_kC_k \,\&\, \lhd D_1 \,\&\, \dots \,\&\, \lhd D_m \,\&\, \trianglelefteq E_1 \,\&\, \dots \,\&\, \trianglelefteq E_n \in P$, for which $D_1, \dots, D_m \in T_{0,d}$ (where $d = |\lambda(\hat{t}(\#))|$) and $E_1, \dots, E_n \in T_{0,e}$, (where $e = |\lambda(\hat{t}(\#))\sigma(p_i(B^+(u)C^+(v)))|$), and $i = k$, then the subterm $p_i(B^+(u)C^+(v))$ is rewritten with $A^+(uv)$.

     In case the grammar contains a rule of the form $A \to B_1C_1 \,\&\, \dots \,\&\, B_kC_k \,\&\, \lhd\varepsilon \in P$, such that $\lambda(\hat{t}(\#)) = \varepsilon$, then the subterm $p_i(B^+(u)C^+(v))$ is rewritten with $A^+(uv)$.

     If no suitable rules are contained in the grammar, the subterm $p_i(B^+(u)C^+(v))$ is rewritten with $A^-(uv)$.
   - If there are more conjuncts in the rule, the subterm $p_i(B^+(u)C^+(v))$ is rewritten with $p_{i+1}(F(a_j)G(a_{j+1} \dots a_\ell))$, where $uv = a_j a_{j+1} \dots a_\ell$ and the $(i+1)$th conjunct in rule p is FG.

4. Any of the subterms t of the form $p_i(B^+(u)C^-(v))$, $p_i(B^-(u)C^+(v))$ or $p_i(B^-(u)C^-(v))$ of the term $\hat{t}(t)$ is rewritten as follows:

   - if $|v| > 1$, let $v = ax$ (with $a \in \Sigma$, $x \in \Sigma^+$) then t is rewritten with $p_i(B(ua)C(x))$;
   - if $|v| = 1$ and p is the last rule for A, t is rewritten with $A^-(uv)$;
   - if $|v| = 1$ and r is the next rule for A, t is rewritten with $r_1(F(a_j)G(a_{j+1} \dots a_\ell))$, where $uv = a_j a_{j+1} \dots a_\ell$ and the first conjunct of r is FG.

Claim 1. *The rewriting preserves truth, that is, a true term is rewritten with a true term.*

Proof. There are numerous cases of rewriting to consider. This proof handles two representative cases, and the rest are omitted for brevity.

- Consider the rewriting of a term with a distinguished subterm

$$\hat{t}(p_i(B^-(a_k \dots a_{\ell-1})C^+(a_\ell))) \tag{12a}$$

  with

$$\hat{t}(r_1(F(a_k)G(a_{k+1} \dots a_\ell))), \tag{12b}$$

where BC is a conjunct of a rule for nonterminal A and r is the next rule for this nonterminal.

To show that the term (12b) is true, the conditions (I)–(IV) need to be proven held:

- *Condition I.* By definition of a true term, the subterms $F(a_j)$ and $G(a_{j+1} \dots a_\ell)$ are always true.
- *Condition III* for the term (12b) is trivially satisfied, since the conjunct r_1 of the rule r is the first one.
- *Condition IV* for the term (12b) is trivially satisfied, since the factorization $a_j \cdot a_{j+1} \dots a_\ell$ is the first factorization of the string $a_j a_{j+1} \dots a_\ell$.
- *Condition II.* Since the term (12a) is true by assumption, the condition (IV) implies that $u_1\langle v_1\rangle \notin L(B)$ or $u_2\langle v_2\rangle \notin L(C)$ for all factorizations $u_1\langle v_1\rangle \cdot u_2\langle v_2\rangle = a_1 \dots a_{j-1}\langle a_j \dots a_\ell\rangle$, with $u_1, v_1, v_2 \in \Sigma^+$, $u_2 = u_1 v_1$ and $|v_2| < \ell - 1$. The condition (I) for (12a) gives that $a_1 \dots a_{j-1}\langle a_j \dots a_{\ell-1}\rangle \notin L(B)$. The combination of these two statements gives that $u_1\langle v_1\rangle \notin L(B)$ or $u_2\langle v_2\rangle \notin L(C)$ for every factorization $u_1\langle v_1\rangle \cdot u_2\langle v_2\rangle = a_1 \dots a_{j-1}\langle a_j \dots a_\ell\rangle$. Thus, $a_1 \dots a_{j-1}\langle a_j \dots a_\ell\rangle \notin L(BC)$, and $a_1 \dots a_{j-1}\langle a_j \dots a_\ell\rangle \notin L(p_i)$. That is, $a_1 \dots a_{j-1}\langle a_j \dots a_\ell\rangle \notin L(p)$. The condition (II) for the true term (12a) implies that $a_1 \dots a_{j-1}\langle a_j \dots a_\ell\rangle \notin L(q)$ for all rules q that precede p. Then, $a_1 \dots a_{j-1}\langle a_j \dots a_\ell\rangle \notin L(r)$, since $a_1 \dots a_{j-1}\langle a_j \dots a_\ell\rangle \notin L(p)$ and p precedes r.

- Consider the rewriting of

$$\hat{t}(p_k(B^+(a_j \dots a_{\ell-1})C^+(a_\ell))) \tag{13a}$$

with

$$\hat{t}(A^+(a_j \dots a_\ell)), \tag{13b}$$

where $p_k = BC$ is the last conjunct of a rule p for nonterminal A. According to the definition of a true subterm, the subterm (13b) is true if $a_1 \dots a_{j-1}\langle a_j \dots a_\ell\rangle \in L(A)$.

Let the rule p be $A \to B_1C_1 \,\&\, \dots \,\&\, B_kC_k \,\&\, \lhd D_1 \,\&\, \dots \,\&\, \lhd D_m \,\&\, \&\, \leqslant E_1 \,\&\, \dots \,\&\, \leqslant E_n \in P$, where $B_kC_k = BC$ (the case of p of the form $A \to B_1C_1 \,\&\, \dots \,\&\, B_kC_k \,\&\, \lhd\varepsilon$ is handled similarly and is omitted).

Since the subterm (13a) is true by assumption, its subterms $B^+(a_j \dots a_{\ell-1})$ and $C^+(a_\ell)$ are true as well. Hence, $a_1 \dots a_{j-1}\langle a_j \dots a_{\ell-1}\rangle \in L(B)$ and $a_1 \dots a_{\ell-1}\langle a_\ell\rangle \in L(C)$.

Let d be the length of the left context of the true term (13a): $d = |\lambda(\hat{t}(\#))| = |a_1 \dots a_{j-1}|$, and let e be the length of concatenation of

(13a) and its left context: $e = |\lambda(\hat{t}(\#))\sigma(p_k(B^+(a_j \dots a_{\ell-1})C^+(a_\ell)))|$. For the rewriting (13) to proceed, $D_1, \dots, D_m$ must be in $T_{0,d}$ and $E_1, \dots, E_n$ must be in $T_{0,e}$.

Then, by the assumption of Lemma 13, $\varepsilon\langle a_1 \dots a_{j-1}\rangle \in \lhd L(D_1) \cap \dots \cap \lhd L(D_m)$, and $\varepsilon\langle a_1 \dots a_\ell\rangle \in \lhd\!\!\!= L(E_1) \cap \dots \cap \lhd\!\!\!= L(E_n)$, with $h_\ell(E_i) < h_0$ $(i \in \{1, \dots, n\})$ for some $h_0 \geqslant 0$.

Therefore, the string $a_1 \dots a_{j-1}\langle a_j \dots a_\ell\rangle$ is in $L(A)$, and the term $\hat{t}(A^+(a_j \dots a_\ell))$ is thus true by definition.

□

Furthermore, it can be proved [13, Claims 1–6] that

- all terms t obtained during the rewriting maintain the string value $\sigma(t) = w$;
- the size of a term with a string value w is bounded by $O(|w|)$;
- there exists a linear ordering on the set of terms, so that each rewriting step increases the term.

Thus, any rewriting that begins with a true term, proceeds over true terms of linearly bounded size, and each term $A(w)$ is eventually converted either to $A^+(w)$ or to $A^-(w)$.

The space complexity of grammars with contexts is now stated in the following theorem.

Theorem 6. *Every language generated by a grammar with one-sided contexts is in* DSPACE(n).

Proof. Let $G = (\Sigma, N, P, S)$ be an arbitrary grammar with one-sided contexts and assume without loss of generality that it is in the binary normal form.

The following algorithm constructs the sets $T_{0,\ell}$:

Algorithm 3.

```
1: for ℓ = 1, ..., n do
2:     let T_{0,ℓ} = ∅
3:     while T_{0,ℓ} changes do
4:         for all A ∉ T_{0,ℓ} do
5:             if rewrite(A, a_1 ... a_ℓ, T_{0,1}, ..., T_{0,ℓ-1}, T_{0,ℓ}) then
6:                 T_{0,ℓ} = T_{0,ℓ} ∪ {A}
```

Each ℓth iteration of the outer loop in line 1 constructs the set $T_{0,\ell}$ of all nonterminals that generate a prefix of length ℓ of the input string.

Each kth iteration of the loop in line 3 begins with $T_{0,\ell}$ containing all elements $A \in N$ with $\varepsilon\langle a_1 \dots a_\ell\rangle \in L_G(A)$ and $h_\ell(A) < k - 1$. Then, by

Lemma 13, lines 4–6 add all elements with $h_\ell(A) = k - 1$ to $T_{0,\ell}$. The set $T_{0,\ell}$ is completely constructed if no more iterations of the while-loop in line 3 can be done.

Finally, as in the cubic-time parsing algorithm in Section 5, the input string is accepted if the start symbol S of the grammar is contained in the set $T_{0,n}$, which represents the whole input string. □

7 Future work

The new model leaves many theoretical questions to ponder. For instance, is there a parsing algorithm for grammars with one-sided contexts working in less than cubic time? For standard context-free grammars, Valiant [23] discovered an algorithm that offloads the most intensive computations into calls to a Boolean matrix multiplication procedure, and thus can work in time $O(n^\omega)$, with $\omega < 3$; according to the current knowledge on matrix multiplication, ω can be reduced to 2.373. The main idea of Valiant's algorithm equally applies to Boolean grammars, which can be parsed in time $O(n^\omega)$ as well [18]. However, extending it to grammars with contexts, as defined in this paper, seems to be inherently impossible, because the logical dependencies between the properties of substrings (that is, between the entries of the table $T_{i,j}$) now have a more complicated structure, and the order of calculating these entries apparently rules out grouping multiple operations into Boolean matrix multiplication. However, there might exist a different $o(n^3)$-time parsing strategy for these grammars, which would be interesting to discover.

Another direction is to develop practical parsing algorithms for grammars with one-sided contexts. An obvious technique to try is the recursive descent parsing, where *ad hoc* restrictions resembling contexts of the form $\trianglerighteq D\Sigma^*$ have long been used to guide deterministic computation. The Lang–Tomita Generalized LR parsing is worth being investigated as well.

A more general direction for further research is to consider grammars with two-sided contexts, which would allow rules of the form $A \to BC\&\triangleleft D\&\trianglelefteq E\&\trianglerighteq F\&\triangleright G$. Such grammars would implement Chomsky's [2] idea of defining phrase-structure rules applicable in a context in full—which is something that was for the first time properly approached in this paper.

Acknowledgements

Research supported by the Academy of Finland under grant 134860.

References

[1] T. Aizikowitz, M. Kaminski, "LR(0) conjunctive grammars and deterministic synchronized alternating pushdown automata", *Computer Science in Russia* (CSR 2011, St. Petersburg, Russia, 14–18 June 2011), LNCS 6651, 345–358.

[2] N. Chomsky, "On certain formal properties of grammars", *Information and Control*, 2:2 (1959), 137–167.

[3] K. Čulík II, R. Cohen, "LR-regular grammars—an extension of LR(k) grammars", *Journal of Computer and System Sciences*, 7:1 (1973), 66–96.

[4] Z. Ésik, W. Kuich, "Boolean fuzzy sets", *International Journal of Foundations of Computer Science*, 18:6 (2007), 1197–1207.

[5] B. Ford, "Parsing expression grammars: a recognition-based syntactic foundation", *Proceedings of POPL 2004* (Venice, Italy, January 14–16, 2004), 111–122.

[6] S. Ginsburg, H. G. Rice, "Two families of languages related to ALGOL", *Journal of the ACM*, 9 (1962), 350–371.

[7] S. Jarzabek, T. Krawczyk, "LL-regular grammars", *Information Processing Letters*, 4 (1975), 31–37.

[8] A. Jeż, "Conjunctive grammars can generate non-regular unary languages", *International Journal of Foundations of Computer Science*, 19:3 (2008), 597–615.

[9] A. Jeż, A. Okhotin, "Conjunctive grammars over a unary alphabet: undecidability and unbounded growth", *Theory of Computing Systems*, 46:1 (2010), 27–58.

[10] V. Kountouriotis, Ch. Nomikos, P. Rondogiannis, "Well-founded semantics for Boolean grammars", *Information and Computation*, 207:9 (2009), 945–967.

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

[12] A. Okhotin, "Conjunctive grammars and systems of language equations", *Programming and Computer Science*, 28:5 (2002), 243–249.

[13] A. Okhotin, "Boolean grammars", *Information and Computation*, 194:1 (2004), 19–48.

[14] A. Okhotin, "The dual of concatenation", *Theoretical Computer Science*, 345:2–3 (2005), 425–447.

[15] A. Okhotin, "Generalized LR parsing algorithm for Boolean grammars", *International Journal of Foundations of Computer Science*, 17:3 (2006), 629–664.

[16] A. Okhotin, "Recursive descent parsing for Boolean grammars", *Acta Informatica*, 44:3–4 (2007), 167–189.

[17] A. Okhotin, "Unambiguous Boolean grammars", *Information and Computation*, 206 (2008), 1234–1247.

[18] A. Okhotin, "Fast parsing for Boolean grammars: a generalization of Valiant's algorithm", *Developments in Language Theory* (DLT 2010, London, Ontario, Canada, August 17–20, 2010), LNCS 6224, 340–351.

[19] A. Okhotin, C. Reitwießner, "Conjunctive grammars with restricted disjunction", *Theoretical Computer Science*, 411:26–28 (2010), 2559–2571.

[20] A. Okhotin, P. Rondogiannis, "On the expressive power of univariate equations over sets of natural numbers", *Information and Computation*, 212 (2012), 1–14.

[21] T. Parr, K. Fisher, "LL(*): the foundation of the ANTLR parser generator", Programming Language Design and Implementation (PLDI 2011, San Jose, USA, 4–8 June 2011), 425–436.

[22] K. Sikkel, *Parsing Schemata*, Springer-Verlag, 1997.

[23] L. G. Valiant, "General context-free recognition in less than cubic time", *Journal of Computer and System Sciences*, 10:2 (1975), 308–314.

Turku
Centre *for*
Computer
Science

Lemminkäisenkatu 14 A, 20520 Turku, Finland | www.tucs.fi

University of Turku
- Department of Information Technology
- Department of Mathematical Sciences

Åbo Akademi University
- Department of Computer Science
- Institute for Advanced Management Systems Research

Turku School of Economics and Business Administration
- Institute of Information Systems Sciences

ISBN 978-952-12-2662-5
ISSN 1239-1891