Fundamenta Informaticae 179 (2021) 361–384
DOI 10.3233/FI-2021-2028
IOS Press

Scattered Context Grammars with One Non-Context-Free Production are Computationally Complete

Zbyněk Křivka, Alexander Meduna[*]
Brno University of Technology, Faculty of Information Technology
IT4Innovations Centre of Excellence
Božetěchova 2, 612 66 Brno, Czech Republic
krivka@fit.vutbr.cz, meduna@fit.vutbr.cz

Abstract. This paper investigates the reduction of scattered context grammars with respect to the number of non-context-free productions. It proves that every recursively enumerable language is generated by a scattered context grammar that has no more than one non-context-free production. An open problem is formulated.

Keywords: Scattered context grammars, size reduction, the number of non-context-free productions, parallel productions, computational completeness, descriptional complexity

1. Introduction

Formal language theory has always aimed at reducing their grammars as much as possible (for an overview of results concerning this reduction in terms of classical grammars, consult Sections 1.2 and 1.3 in Chapter 4 in [1]). As a central topic, this line of research has studied how to reduce the number of grammatical components, such as nonterminals or productions, without disturbing the generative power. The present paper contributes to this line of research in terms of scattered context grammars introduced in [2] (for an overview of important results concerning the reduction in terms of scattered context grammars, consult Chapter 6 in [3] and journal papers [4, 5, 6]).

[*]Address for correspondence: Brno University of Technology, Faculty of Information Technology, IT4Innovations Centre of Excellence, Božetěchova 2, 612 66 Brno, Czech Republic.

Received October 2019; revised February 2021.

Concerning the number of nonterminals in scattered context grammars, two-nonterminal scattered context grammars are computationally complete—that is, they characterize the family of recursively enumerable languages (see [7]). On the other hand, one-nonterminal scattered context grammars are less powerful (see [8]). In terms of the number of non-context-free productions, a reduction like this is studied in [4] and [5]; most importantly, the question of whether scattered context grammars with a single non-context-free production are computationally complete is formulated in the former.

The present paper reduces the number of non-context-free productions in scattered context grammars. In fact, it proves that scattered context grammars with a single non-context-free production are computationally complete. Of course, this statement represents the best possible result regarding these reductions because scattered context grammars without any non-context-free production only characterize the family of context-free languages which is properly included in the family of scattered context languages.

2. Definitions

This paper assumes that the reader is familiar with formal language theory (see [9]), including scattered context grammars (see [2], [3], [10]).

For a set, Q, $\mathrm{card}(Q)$ denotes the cardinality of Q. For an alphabet, V, V^* represents the free monoid generated by V under the operation of concatenation. The unit of V^* is denoted by ε. Set $V^+ = V^* - \{\varepsilon\}$; algebraically, V^+ is thus the free semigroup generated by V under the operation of concatenation. For $w \in V^*$, $|w|$ and $\mathrm{Reverse}(w)$ denote the length of w and the reversal of w, respectively. Furthermore, $\mathrm{suffix}(w)$ denotes the set of all suffixes of w, and $\mathrm{prefix}(w)$ denotes the set of all prefixes of w. For $w \in V^*$ and $T \subseteq V$, $\mathrm{occur}(w, T)$ denotes the number of occurrences of symbols from T in w, and $\mathrm{Erase}(w, T)$ denotes the string obtained by removing all occurrences of symbols from T in w. For instance, $\mathrm{occur}(abdabc, \{a, d\}) = 3$ and $\mathrm{Erase}(abdabc, \{a, d\}) = bbc$. If $T = \{a\}$, where $a \in V$, we simplify $\mathrm{occur}(w, \{a\})$ and $\mathrm{Erase}(w, \{a\})$ to $\mathrm{occur}(w, a)$ and $\mathrm{Erase}(w, a)$, respectively.

A *scattered context grammar* is a quadruple, $G = (N, T, P, S)$, where N and T are alphabets such that $N \cap T = \emptyset$. Symbols in N are referred to as nonterminals while symbols in T are terminals. N contains S—the start symbol of G. P is a finite non-empty set of productions such that every $p \in P$ has the form

$$(A_1, A_2, \dots, A_n) \to (x_1, x_2, \dots, x_n),$$

where $n \geq 1$, and for all $i = 1, 2, \dots, n$, $A_i \in N$ and $x_i \in (N \cup T)^*$. If $n = 1$, then $(A_1) \to (x_1)$ is referred to as a context-free production; for brevity, we hereafter write $A_1 \to x_1$ instead of $(A_1) \to (x_1)$. If for some $n \geq 1$, $(A_1, A_2, \dots, A_n) \to (x_1, x_2, \dots, x_n) \in P, v = u_1A_1u_2A_2 \dots u_nA_nu_{n+1}$, and $w = u_1x_1u_2x_2 \cdots u_nx_nu_{n+1}$ with $u_i \in (N \cup T)^*$ for all $i = 1, 2, \dots, n$, then v directly derives w in G, written as $v \Rightarrow w$ $[(A_1, A_2, \dots, A_n) \to (x_1, x_2, \dots, x_n)]$ or, simply, $v \Rightarrow w$ in G. In the standard manner, extend $\Rightarrow$ to $\Rightarrow^n$, where $n \geq 0$; then, based on $\Rightarrow^n$, define $\Rightarrow^+$ and $\Rightarrow^*$. The language of G, $L(G)$, is defined as $L(G) = \{w \in T^* : S \Rightarrow^* w\}$. A derivation of the form $S \Rightarrow^* w$ with $w \in T^*$ is called a successful derivation.

Example 2.1. Let $G = (\{S, A, B, C, D\}, \{a, b, c, d\}, P, S)$ be a scattered context grammar, where P contains

1. $S \rightarrow ABCD$
2. $(A, C) \rightarrow (aA, Cc)$
3. $(B, D) \rightarrow (Bb, dD)$
4. $(A, C) \rightarrow (a, c)$
5. $(B, D) \rightarrow (b, d)$

For instance, we can generate $aaabbcccdd$ in the following successful derivation:

$$S \Rightarrow ABCD \Rightarrow ABbCdD \Rightarrow aABbCcdD \Rightarrow$$

$$aaABbCccdD \Rightarrow aaAbbCccdd \Rightarrow aaabbcccdd$$

Observe that $L(G) = \{a^m b^n c^m d^n \colon m, n \geq 1\}$ which is a non-context-free language.

A *queue grammar* (see [11]) is a sextuple, $Q = (V, T, W, F, s, R)$, where V and W are alphabets of symbols and states, respectively, satisfying $V \cap W = \emptyset$, $T \subseteq V$, $F \subseteq W$, $s \in (V - T)(W - F)$, and $R \subseteq (V \times (W - F)) \times (V^* \times W)$ is a finite relation such that for every $a \in V$, there exists an element $(a, p, z, q) \in R$. If $u \in V^+W$ and $v \in V^*W$ such that $u = arp$; $v = rzq$; $a \in V$; $r, z \in V^*$; $p, q \in W$; and $(a, p, z, q) \in R$, then $u \Rightarrow v$ $[(a, p, z, q)]$ in Q or, simply, $u \Rightarrow v$. In the standard manner, extend $\Rightarrow$ to $\Rightarrow^n$, where $n \geq 0$; then, based on $\Rightarrow^n$, define $\Rightarrow^+$ and $\Rightarrow^*$. The language of Q, $L(Q)$, is defined as $L(Q) = \{w \in T^* : s \Rightarrow^* wf, f \in F\}$.

As a slight modification of the notion of a queue grammar, we introduce the notion of a left-extended queue grammar (see [12]) such that it is a queue grammar that during every derivation step shifts the rewritten symbol in front of the beginning of its sentential form; in this way, it records the derivation history in front of the special symbol $\#$. The derivation history plays a crucial role in the proof of Lemma 3.7 in the next section. Formally, *left-extended queue grammar* is a sextuple, $Q = (V, T, W, F, s, R)$, where V, T, W, F, and s are defined as in a queue grammar. $R \subseteq (V \times (W - F)) \times (V^* \times W)$ is a finite relation (as opposed to an ordinary queue grammar, this definition does not require that for every $a \in V$, there exists an element $(a, p, z, q) \in R$). Furthermore, assume that $\# \notin V \cup W$. If $u, v \in V^*\{\#\}V^*W$ so that $u = w\#arp; v = wa\#rzq; a \in V; r, z, w \in V^*; p, q \in W$; and $(a, p, z, q) \in R$, then $u \Rightarrow v$ $[(a, p, z, q)]$ in Q or, simply, $u \Rightarrow v$. In the standard manner, we extend $\Rightarrow$ to $\Rightarrow^n$, where $n \geq 0$; then, based on $\Rightarrow^n$, define $\Rightarrow^+$ and $\Rightarrow^*$. The language of $Q, L(Q)$, is defined as $L(Q) = \{v \in T^* : \#s \Rightarrow^* w\#vf$ for some $w \in V^*$ and $f \in F\}$.

In general, two grammars are *equivalent* if both generate the same language.

Example 2.2. Let $Q = (V, T, W, F, s, R)$ be a left-extended queue grammar, where $V = \{S, \$, a\}$, $T = \{a\}$, $W = \{p, f\}$, $F = \{f\}$, $s = Sp$, and R contains

1. $(S, p, \$a, p)$
2. $(\$, p, \$, p)$

3. (a, p, aa, p)
4. $(\$, p, \varepsilon, f)$

For instance, we can generate $aaaa$ in the following successful derivation (for brevity, in the brackets, we only use the labels denoting the productions above):

$$\# Sp \Rightarrow S \# \$ap\ [1] \Rightarrow S\$ \# a\$p\ [2] \Rightarrow S\$a \# \$aap\ [3] \Rightarrow S\$a\$ \# aa\$p\ [2] \Rightarrow$$

$$S\$a\$a \# a\$aap\ [3] \Rightarrow S\$a\$aa \# \$aaaap\ [3] \Rightarrow S\$a\$aa\$ \# aaaaf\ [4]$$

To give a more general insight into derivations in Q, consider any sentential form w. Observe that w has this form

$$S\$a\$aa\$ \cdots \#aa \cdots aaq$$

That is, w consists of three parts—(1) the part preceding $\#$ represents, intuitively speaking, the rewriting history, (2) the rightmost symbol q is the current state, and (3) in between $\#$ and q, the queue of as, processed in the left-to-right way, occurs. On sentential forms of this form, Q works so it places aa at the queue end for each a read at the queue head. Whenever \$ is read, Q can either repeat this doubling mechanism or enter f, thus successfully completing the derivation. As a result, the length of any sentence in $L(Q)$ is a power of two.

Observe that $L(Q) = \{a^{2^n} : n \geq 0\}$ which is a non-context-free language.

3. Results

This section demonstrates that for every recursively enumerable language, L, there exists a scattered context grammar, $G = (N, T, P, S)$, such that $L = L(G)$ and P contains a single non-context-free production of the form $(1, 2, 0, 3, 0, 2, 1) \to (2, \varepsilon, \varepsilon, \varepsilon, \varepsilon, \varepsilon, 2)$. This demonstration is based on left-extended queue grammars, which are computationally complete (see Theorem 4.3 in [13]).

Next, we establish the first normal form of left-extended queue grammars, subsequently used in the proofs of Lemmas 3.4 and 3.7. In essence, in this form, these grammars never rewrite terminals; in addition, at the queue end represented by the rightmost symbol of the sentential form occurring in front of the current state, they produce strings consisting purely of either terminals or nonterminals.

Definition 3.1. Let $Q = (V, T, W, F, s, R)$ be a left-extended queue grammar. Q is said to be in *normal form 1* if every $(a, p, x, q) \in R$ satisfies $a \in V - T$, $p \in W - F$, $q \in W$, and $x \in ((V - T)^* \cup T^*)$.

Next, we show how to turn any left-extended queue grammar H to an equivalent left-extended queue grammar Q in normal form 1. To give an insight into this transformation, consider any successful generation of x from $L(H)$ in H. By using two-component states and extra symbol 1, Q simulates this generation in the following three phases.

(i) Before applying productions that generate only terminals, Q performs the generation exactly like H does except that Q only uses nonterminals, each of which encodes a symbol from H. Noteworthy, terminals of H are encoded in this way, too.

(ii) Q generates an extra symbol 1 that marks the position behind which only terminals can be produced.

(iii Q enters a special two-component state, rewrites 1 and goes on generating only terminals to obtain x as the result.

Lemma 3.2. For every left-extended queue grammar H, there exists an equivalent left-extended queue grammar Q in normal form 1.

Proof:
Let $H = (\bar{V}, T, \bar{W}, \bar{F}, \bar{s}, \bar{R})$ be any left-extended queue grammar. Set $\bar{W}' = \{q' : q \in \bar{W}\}$, $\bar{W}'' = \{q'' : q \in \bar{W}\}$, and $\bar{V}' = \{a' : a \in \bar{V}\}$. Define the bijection α from $\bar{W}$ to $\bar{W}'$ as $\alpha(q) = q'$ for every $q \in \bar{W}$. Analogously, define the bijection β from $\bar{W}$ to $\bar{W}''$ as $\beta(q) = q''$ for every $q \in \bar{W}$. Finally, define the bijection δ from $\bar{V}$ to $\bar{V}'$ as $\delta(a) = a'$ for every $a \in \bar{V}$. In the standard manner, extend δ so it is defined from $\bar{V}^*$ to $(\bar{V}')^*$. Set

$$U = \{\langle y, q\rangle : y \in T^*, q \in \bar{W}, \text{and } (a, p, xy, q) \in \bar{R} \text{ for some } a \in \bar{V}, p \in \bar{W} - \bar{F}, x \in \bar{V}^*\}$$

Consider new objects 1, f outside $\delta(\bar{V}) \cup T \cup \alpha(\bar{W}) \cup \beta(\bar{W}) \cup U$. Set $V = \delta(\bar{V}) \cup \{1\} \cup T$, $W = \alpha(\bar{W}) \cup \beta(\bar{W}) \cup \{f\} \cup U$, $F = \{f\}$, and $s = \delta(a)\alpha(q)$ where $\bar{s} = aq$. Define the left-extended queue grammar

$$Q = (V, T, W, F, s, R)$$

with R constructed in the following way:

I. if $(a, p, xy, q) \in \bar{R}$, where $a \in \bar{V}; p \in \bar{W} - \bar{F}; x, y \in \bar{V}^*$; and $q \in \bar{W}$, then add $(\delta(a), \alpha(p), \delta(x)\delta(y), \alpha(q))$ and $(\delta(a), \alpha(p), \delta(x)1\delta(y), \alpha(q))$ to R;

II. if $(a, p, xy, q) \in \bar{R}$, where $a \in \bar{V}; p \in \bar{W} - \bar{F}; x \in \bar{V}^*; y \in T^*$; $q \in \bar{W}$; and $\langle y, q\rangle \in U$, then add $(\delta(a), \alpha(p), \delta(x), \langle y, q\rangle)$ and $(1, \langle y, q\rangle, y, \beta(q))$ to R;

III. if $(a, p, x, q) \in \bar{R}$, where $a \in \bar{V}; p \in \bar{W} - \bar{F}; x \in T^*$; and $q \in \bar{W}$, then add $(\delta(a), \beta(p), x, \beta(q))$ to R;

IV. if $(a, p, x, q) \in \bar{R}$, where $a \in \bar{V}; p \in \bar{W} - \bar{F}; x \in T^*$; and $q \in \bar{F}$, then add $(\delta(a), \beta(p), x, f)$ to R (recall that $F = \{f\}$).

Clearly, for every $(a, p, x, q) \in R$, we have $a \in V - T$, $p \in W - F$, $q \in W$, and $x \in ((V - T)^* \cup T^*)$. Next, we prove that $L(H) = L(Q)$.

To see that $L(H) \subseteq L(Q)$, consider any $v \in L(H)$. As $v \in L(H)$,

$$\#\bar{s} \Rightarrow^* w\#vt$$

in $H, w \in \bar{V}^*, v \in T^*$, and $t \in \bar{F}$. Express $\#\bar{s} \Rightarrow^* w\#vt$ in H as

$$\#\bar{s} \Rightarrow^* u\#zp \Rightarrow ua\#xyq \Rightarrow^* w\#vt$$

where $a \in \bar{V}, u, x \in \bar{V}^*, y \in \text{prefix}(v), z = ax, w = uax$, and during $ua\#xyq \Rightarrow^* w\#vt$, only terminals are generated so that the resulting terminal string equals v. Q simulates $\#\bar{s} \Rightarrow^* u\#zp \Rightarrow ua\#xyq \Rightarrow^* w\#vt$ as follows. First, Q uses productions introduced in I to simulate $\#\bar{s} \Rightarrow^* u\#zp$. During this initial simulation, it once uses a production that generates 1 so that it can then simulate $u\#zp \Rightarrow ua\#xyq$ by making two derivation steps according to productions $(\delta(a), \alpha(p), \delta(x), \langle y, q\rangle)$ and $(1, \langle y, q\rangle, y, \beta(q))$ (see II). Notice that by using $(1, \langle y, q\rangle, y, \beta(q))$, Q produces y, which is a prefix of v. After the application of $(1, \langle y, q\rangle, y, \beta(q))$, Q simulates $ua\#xyq \Rightarrow^* w\#vt$ by using productions introduced in III followed by one application of a production constructed in IV, during which Q enters f and, thereby, completes the generation of v. Thus, $L(H) \subseteq L(Q)$.

To establish that $L(Q) \subseteq L(H)$, consider any $v \in L(Q)$. Since $v \in L(Q)$,

$$\#s \Rightarrow^* w\#vf$$

in Q, where $w \in V^*$ and $v \in T^*$. Examine I through IV. Observe that Q passes through states of $\alpha(\bar{W}), U, \beta(\bar{W})$, and $\{f\}$ in this order so that it occurs several times in states of $\alpha(\bar{W})$, once in a state of U, several times in states of $\beta(\bar{W})$, and once in f. As a result, Q uses productions introduced in I, and during this initial part of derivation it precisely once uses a production that generates 1 (observe that any subsequent generation of 1 would rule out the generation of a terminal string). After this, it can make two consecutive derivation steps according to $(\delta(a), \alpha(p), \delta(x), \langle y, q\rangle)$ and $(1, \langle y, q\rangle, y, \beta(q))$ (see II). By using the latter, Q produces y, which is a prefix of v. After the application of $(1, \langle y, q\rangle, y, \beta(q))$, Q applies productions introduced in III, which always use states of $\beta(\bar{W})$. Finally, it once applies a production constructed in IV to enter f and, thereby, complete the generation of v. To summarize these observations, we can express $\#s \Rightarrow^* w\#vf$ in Q as

$$\#s \Rightarrow^* u\#zp \Rightarrow ua\#xyq \Rightarrow^* w\#vf$$

where $a \in V, x \in V^*, y \in T^*, w = uax$ so that during $\#s \Rightarrow^* u\#zp$, Q uses productions introduced in I, then it applies $(1, \langle y, q\rangle, y, \beta(q))$ from II to make $u\#zp \Rightarrow ua\#xyq$, and finally it performs $ua\#xyq \Rightarrow^* w\#vf$ by several applications of productions introduced in III and one application of a production constructed in IV. At this point, by an examination of I through IV, we see that H makes

$$\#\bar{s} \Rightarrow^* u\#zp \Rightarrow ua\#xyq \Rightarrow^* w\#vt$$

with $t \in \bar{F}$, so $v \in L(H)$. Therefore, $L(H) \subseteq L(Q)$.

As $L(H) \subseteq L(Q)$ and $L(Q) \subseteq L(H), L(H) = L(Q)$. □

Next, we establish the second normal form of left-extended queue grammars, subsequently used in the proof of Lemma 3.7. This normal form requests any left-extended queue grammar in normal form 1 to satisfy the following property:

$$(a, p, x, q) \in R \text{ and } (a', p, x', q') \in R \text{ imply that } a = a'$$

where $a, a' \in V - T, p \in W - F, q, q' \in W$ and $x, x' \in (V - T)^* \cup T^*$.

Definition 3.3. Let $Q = (V, T, W, F, s, R)$ be a left-extended queue grammar. Q is said to be in *normal form 2* if it is in normal form 1 and in addition, for every $p \in W - F$, there is no more than one $a \in V - T$ such that $(a, p, x, q) \in R$, where $x \in (V - T)^* \cup T^*$ and $q \in W$.

Next, we explain how to convert any left-extended queue grammar H in normal form 1 to an equivalent left-extended queue grammar Q in normal form 2. To give a brief insight into this conversion, every state in Q is of the form $\langle q, a\rangle$, where q is a state from H and a is a nonterminal. In fact, a prescribes the nonterminal that has to be rewritten at the beginning of the queue when simulating the rewriting in the original state q. To complete this insight, let us point out that for any $a, a' \in V - T$, Q has to change the current state to $\langle q, a\rangle$ or $\langle q, a'\rangle$ in order to rewrite a or a', respectively.

Lemma 3.4. For every left-extended queue grammar H, there exists an equivalent left-extended queue grammar Q in normal form 2.

Proof:
Let $H = (V, T, \bar{W}, F, \bar{s}, \bar{R})$ be any left-extended queue grammar in normal form 1, where $\bar{s} = a_0 q_0$, $a_0 \in V - T$, and $q_0 \in \bar{W} - F$. Set $W = \{\langle q, a\rangle : q \in \bar{W}, a \in V - T\} \cup F$, and let $s = a_0\langle q_0, a_0\rangle$. Define the left-extended queue grammar $Q = (V, T, W, F, s, R)$ with R constructed in the following way:

(1) if $(a, p, y, q) \in \bar{R}$, where $a \in V-T, p, q \in \bar{W}-F, y \in (V-T)^* \cup T^*$, then add $(a, \langle p, a\rangle, y, \langle q, b\rangle)$ to R, for all $b \in V - T$;

(2) if $(a, p, y, q) \in \bar{R}$, where $a \in V - T$, $p \in \bar{W} - F$, $y \in T^*$, and $q \in F$, then add $(a, \langle p, a\rangle, y, q)$ to R.

Clearly, in Q, we can rewrite symbol a that follows # if and only if we are in state $\langle p, a\rangle$ from $W - F$ for some $p \in \bar{W}$. Next, we demonstrate that $L(H) = L(Q)$.

To see that $L(H) \subseteq L(Q)$, consider any $w \in L(H)$. As $w \in L(H)$, there is a derivation

$$\#\bar{s} \Rightarrow^* \gamma\#cvq \Rightarrow \gamma c\#vyf$$

in H where $c \in V-T$, $q \in \bar{W}-F$, $w = vy$, and $f \in F$. To every derivation step $z\#axp \Rightarrow za\#xx'p'$ from $\#\bar{s} \Rightarrow^* \gamma\#cvq$ in H where $a \in V - T$, $p, p' \in \bar{W} - F$, and $x' \in V^*$, there is a corresponding derivation step in Q

$$z\#ax\langle p, a\rangle \Rightarrow za\#xx'\langle p', b\rangle$$

where $a \in V - T$ and $\langle p, a\rangle, \langle p', b\rangle \in W - F$.

The inclusion $L(Q) \subseteq L(H)$ is obvious as well. Q starts from $\#a_0\langle q_0, a_0\rangle$ and continues the derivation by performing steps of this form

$$z\#ax\langle p, a\rangle \Rightarrow z'\#x\langle q, b\rangle$$

where $a, b \in V - T$, $x, z, z' \in V^*$, and $\langle p, a\rangle, \langle q, b\rangle \in W$. Observe that every derivation step performed in Q corresponds to exactly one step in H. In addition, Q non-deterministically chooses a symbol b by entering state $\langle q, b\rangle$ one step earlier than H does; however, this entrance is actually be performed only if Q has properly chosen b, which can be rewritten in state q from $\bar{W}$. In the last step, Q enters the final state from F. □

The upcoming Lemmas 3.5 and 3.7 are crucially important because their construction parts turn any left-extended queue grammar Q to an equivalent scattered context grammar G with a single non-context-free production—that is, the objective of this paper as a whole. In essence, the proof of Lemma 3.5 turns any left-extended queue grammar to an equivalent scattered context grammar H satisfying a prescribed property concerning the way it works, after which the proof of Lemma 3.7 transforms H, resulting from Lemma 3.5, to an equivalent scattered context grammar with one non-context-free production. As obvious, both lemmas might be combined and proved as a single statement. However, a proof like this would be so unbearably complicated that we undertake the two-lemma approach in what follows. Before that, however, we briefly sketch both Lemma 3.5 and Lemma 3.7 slightly more precisely.

In the construction part of the proof of Lemma 3.5, we turn any left-extended queue grammar Q satisfying normal form 2 to an equivalent scattered context grammar H in the following way. Take any $x \in L(Q)$. Every successful derivation of x in Q is simulated in H by a derivation consisting of two phases. During the first phase, in an utterly non-deterministic way, by using only context-free productions, it derives uxv from its start symbol. During the second part, by using only non-context-free productions, in an inside-out way, it gradually eliminates u and v and, thereby, verifies that the first non-deterministic phase of simulation has been actually performed properly. In this way, it generates x. Thus, $L(H) = L(Q)$.

Then, Lemma 3.7 demonstrates that for any left-extended queue grammar Q, there is an equivalent scattered context grammar with a single non-context-free production—the goal of the present study. Without any loss of generality, the construction part of the proof of Lemma 3.7 assumes that Q is transformed to a scattered context grammar H that works in the way described in Lemma 3.5 and turns it to an equivalent scattered context grammar G with a single non-context-free production in the following way. First, it encodes the nonterminals in H by codes over $\{0, 1, 2, 3\}$. Then, it introduces $(1, 2, 0, 3, 0, 2, 1) \rightarrow (2, \varepsilon, \varepsilon, \varepsilon, \varepsilon, \varepsilon, 2)$ as its only non-context-free production. By using this production, it removes u and v by analogy with the way performed in the proof of Lemma 3.5.

To give an intuitive and more detailed insight into Lemma 3.5, take any left-extended queue grammar Q. Lemma 3.5 says that there exists a scattered context grammar H such that $L(Q) = L(H)$ and, in addition, H simulates every successful generation $\#s \Rightarrow^* w\#yf$ in Q, $y \in L(Q)$, in the following two-phase way.

(i) In a non-deterministic and context-free way, H generates a sentential form corresponding to a sentential form in Q. H performs this generation so that it includes all the generation steps of the corresponding generation in Q preceeding the step when # is hit (this performance resembles the simulation given in [10] on page 129).

(ii) H checks that all previous non-deterministic choices in (i) were made properly. This check is made by eliminating special nonterminals A, $\bar{A}$, $\bar{L}$, and $\bar{R}$ in an inside-out way. Uniform productions of the form $(A, \bar{L}, \bar{R}, \bar{A}) \rightarrow (\bar{L}, \varepsilon, \varepsilon, \bar{R})$ are used during this elimination.

Lemma 3.5. For any left-extended queue grammar $Q = (V, T, W, F, s, R)$, there exists an equivalent scattered context grammar $H = (K, T, P, S)$ so the next equivalence holds true for any $y \in T^*$,

$$S \Rightarrow^* \nu_1 \bar{L} y \bar{R} \nu_2 \Rightarrow^* \bar{L} y \bar{R} \Rightarrow^2 y \text{ in } H \text{ if and only if } \#s \Rightarrow^* w\#yf \text{ in } Q$$

where $s = a_0q_0$, $w \in (V - T)^*$, $f \in F$, and $\nu_1, \nu_2 \in (K - \{\bar{L}, \bar{R}\})^m$, $m \geq 0$ such that $S \Rightarrow^* \nu_1 \bar{L} y \bar{R} \nu_2$ uses only context-free productions from P and $\nu_1 \bar{L} y \bar{R} \nu_2 \Rightarrow^* \bar{L} y \bar{R}$ uses only non-context-free productions from P.

Proof:
Let $Q = (V, T, W, F, s, R)$ be a left-extended queue grammar. Without any loss of generality (by Lemma 3.4), assume that Q is in normal form 2.

Construction. Consider new objects S, $\bar{L}$, $\bar{R}$ outside $V \cup W \cup N \cup M$. Set $U = \{\langle p, i\rangle : p \in W - F \text{ and } i \in \{1, 2\}\} \cup \{S, \bar{L}, \bar{R}\}$. Set $N = \{\lceil \underline{a}, q\rceil : a \in V - T, q \in W - F\}$ and $M = \{\lfloor a, \underline{q}\rfloor : a \in V - T, q \in W - F\}$. Introduce the scattered context grammar $H = (U \cup N \cup M, T, P, S)$ with $P = P_1 \cup P_2 \cup P_3$ constructed in the following way.

To construct P_1, perform (i) through (v), given next.

(i) if $a_0q_0 = s$, where $a_0 \in V - T$ and $q_0 \in W - F$, then add $S \to \lceil \underline{a_0}, q\rceil \langle q_0, 1\rangle \lfloor a, \underline{q_0}\rfloor$ to P_1, for all $q \in W - F$ and all $a \in V - T$;

(ii) if $(a, q, y, p) \in R$, where $a \in V - T$, $p, q \in W - F$, $y \in (V - T)^*$, and $y = a_1a_2 \cdots a_{|y|}$, then add $\langle q, 1\rangle \to \lceil \underline{a_1}, q_1\rceil \lceil \underline{a_2}, q_2\rceil \cdots \lceil \underline{a_{|y|}}, q_{|y|}\rceil \langle p, 1\rangle \lfloor b, \underline{p}\rfloor$ with $q_1 = p$ to P_1, for all $q_2, q_3, \ldots, q_{|y|} \in W - F$ and for all $b \in V - T$;

(iii) for every $q \in W - F$, add $\langle q, 1\rangle \to \bar{L}\langle q, 2\rangle$ to P_1;

(iv) if $(a, q, y, p) \in R$, where $a \in V - T$, $p, q \in W - F$, $y \in T^*$, then add $\langle q, 2\rangle \to y\langle p, 2\rangle \lfloor b, \underline{p}\rfloor$ to P_1, for all $b \in V - T$;

(v) if $(a, q, y, p) \in R$, where $a \in V - T$, $q \in W - F$, $y \in T^*$, and $p \in F$, then add $\langle q, 2\rangle \to y\bar{R}$ to P_1.

Set

$$P_2 = \{(\lceil \underline{a}, q\rceil, \bar{L}, \bar{R}, \lfloor a, \underline{q}\rfloor) \to (\bar{L}, \varepsilon, \varepsilon, \bar{R}) : a \in V - T, q \in W - F\}$$

and

$$P_3 = \{\bar{L} \to \varepsilon, \bar{R} \to \varepsilon\}$$

Proof (Gist). Next, we sketch the reason why $L(H) = L(Q)$. What we need to demonstrate is that for any $y \in T^*$,

$$S \Rightarrow^* y \text{ in } H \text{ if and only if } \#s \Rightarrow^* w\#yf \text{ in } Q$$

with $s = a_0q_0$, $w \in (V - T)^*$, and $f \in F$.

To rephrase this equivalence more precisely, we need to show that $S \Rightarrow^* y$ in H if and only if for some $m \geq 1$, Q makes $\#a_0q_0 \Rightarrow^* a_0 \cdots a_m\#yf$ according to (a_0, q_0, z_0, q_1) through (a_m, q_m, z_m, q_{m+1}), where $q_{m+1} = f$. To see why this equivalence holds true, take any $S \Rightarrow^* y$ with $y \in L(H)$. Examine the construction of P to see that $S \Rightarrow^* y$ in H has, in a greater detail, the form

$$S \Rightarrow^* \nu_1 \bar{L} y \bar{R} \nu_2 \Rightarrow^* y$$

with $\nu_1 \in N^*$, $\nu_2 \in M^*$, $|\nu_1| = \ell$, $|\nu_2| = k$, where $\ell, k \geq 1$. Consequently, we see that proving the equivalence requires a demonstration that in the derivation of the above form in H,

(I) $m = \ell = k$;

(II) $\nu_1 = \lceil \underline{a_0}, q_0 \rceil \cdots \lceil \underline{a_\ell}, q_\ell \rceil$ for some $q_0, \ldots, q_\ell \in W - F$ with $a_0, \ldots, a_\ell \in V - T$;

(III) $\nu_2 = \lfloor a_k, \underline{q_k} \rfloor \cdots \lfloor a_0, \underline{q_0} \rfloor$ for some $a_0, \ldots, a_k \in V - T$ with $q_0, \ldots, q_k \in W - F$.

Consider (II) above. Observe that ν_1 encodes the prefix of all the front queue symbols (including the erased symbols) rewritten during the generation of y. This is the reason why we assume that Q is a left-extended queue grammar, which records this prefix as opposed to any ordinary queue grammar, which throws it away.

During the sketch of this basic idea, we refer to all symbols that occur somewhere to the left of y as *left nonterminals*, and we refer to all symbols that occur somewhere to the right of y as *right nonterminals*.

Let us examine $\nu_1 \bar{L} y \bar{R} \nu_2 \Rightarrow^* y$ in a greater detail. In front of y, $\bar{L}$ is made by a production from (iii), and behind y, $\bar{R}$ is produced by a production from (v). Observe that all the left and right nonterminals can be removed only by productions from $P_2 \cup P_3$. In $\nu_1 \bar{L} y \bar{R} \nu_2$, there exist the only occurrence of each special symbol, $\bar{L}$ and $\bar{R}$. Productions from P_2 are applicable as soon as these nonterminals appear in the rewritten string, and the application of a production from P_2 does not change the number of these occurrences. Consequently, during $\nu_1 \bar{L} y \bar{R} \nu_2 \Rightarrow^* y$, productions from P_3 are applied only during the last two steps while all the preceding steps are made by using productions from P_2. $\bar{L}$ is always a left nonterminal and $\bar{R}$ occurs always as a right nonterminal. Considering these observations and P_2, we see that if a string contains a nonterminal from $N \cup M$ somewhere in between $\bar{L}$ and $\bar{R}$, then H cannot derive a terminal string from it. Consequently, during $\nu_1 \bar{L} y \bar{R} \nu_2 \Rightarrow^* y$, H eliminates symbols from $N \cup M$ in an inside-out way so that it always rewrites the rightmost occurrence within the left nonterminals from N and, simultaneously, the leftmost occurrence among the right nonterminals from M.

In this way, H simulates the successful derivation of y performed by Q so

$$S \Rightarrow^* \nu_1 \bar{L} y \bar{R} \nu_2 \Rightarrow^* \bar{L} y \bar{R} \Rightarrow^2 y$$

with $\nu_1 \in N^*, \nu_2 \in M^*, \nu_1 = \lceil \underline{a_0}, q_0 \rceil \cdots \lceil \underline{a_m}, q_m \rceil$ with $a_0, \ldots, a_m \in V - T$, and $\nu_2 = \lfloor a_m, \underline{q_m} \rfloor \cdots \lfloor a_0, \underline{q_0} \rfloor$ with $q_0, \ldots, q_m \in W - F$. Thus, $L(Q) = L(H)$.

Next, we illustrate the construction of a simulating scattered context grammar for a very simple left-extended queue grammar.

Example 3.6. Consider a left-extended queue grammar $Q = (V, T, W, \{f\}, Ss, R)$ in normal form 2 with $V = \{S, X, a, b\}, T = \{a, b\}, W = \{s, p, q, f\}$. Let $R = \{(S, s, XS, p), (X, p, aa, q), (S, q, bb, f)\}$

Considering the construction from Lemma 3.5, we present all productions in the resulting scattered context grammar, $H = (K, T, P, S')$, in the corresponding steps (productions used in the example of a derivation are in bold):

(i): Add $\boldsymbol{S' \rightarrow \lceil \underline{S}, s\rceil \langle s, 1\rangle \lfloor S, \underline{s}\rfloor}$,
$S' \rightarrow \lceil \underline{S}, s\rceil \langle s, 1\rangle \lfloor X, \underline{s}\rfloor$,
$S' \rightarrow \lceil \underline{S}, p\rceil \langle s, 1\rangle \lfloor S, \underline{s}\rfloor$,
$S' \rightarrow \lceil \underline{S}, p\rceil \langle s, 1\rangle \lfloor X, \underline{s}\rfloor$,
$S' \rightarrow \lceil \underline{S}, q\rceil \langle s, 1\rangle \lfloor S, \underline{s}\rfloor$,
$S' \rightarrow \lceil \underline{S}, q\rceil \langle s, 1\rangle \lfloor X, \underline{s}\rfloor$ into P;

(ii): For $(S, s, XS, p) \in R$, add
$\langle s, 1\rangle \rightarrow \lceil \underline{X}, p\rceil \lceil \underline{S}, s\rceil \langle p, 1\rangle \lfloor S, \underline{p}\rfloor$,
$\langle s, 1\rangle \rightarrow \lceil \underline{X}, p\rceil \lceil \underline{S}, s\rceil \langle p, 1\rangle \lfloor X, \underline{p}\rfloor$,
$\langle s, 1\rangle \rightarrow \lceil \underline{X}, p\rceil \lceil \underline{S}, p\rceil \langle p, 1\rangle \lfloor S, \underline{p}\rfloor$,
$\langle s, 1\rangle \rightarrow \lceil \underline{X}, p\rceil \lceil \underline{S}, p\rceil \langle p, 1\rangle \lfloor X, \underline{p}\rfloor$,
$\langle s, 1\rangle \rightarrow \lceil \underline{X}, p\rceil \lceil \underline{S}, q\rceil \langle p, 1\rangle \lfloor S, \underline{p}\rfloor$,
$\boldsymbol{\langle s, 1\rangle \rightarrow \lceil \underline{X}, p\rceil \lceil \underline{S}, q\rceil \langle p, 1\rangle \lfloor X, \underline{p}\rfloor}$ into P;

(iii): Add $\langle s, 1\rangle \rightarrow \bar{L}\langle s, 2\rangle$, $\boldsymbol{\langle p, 1\rangle \rightarrow \bar{L}\langle p, 2\rangle}$, $\langle q, 1\rangle \rightarrow \bar{L}\langle q, 2\rangle$ into P;

(iv): For $(X, p, aa, q) \in R$, add
$\boldsymbol{\langle p, 2\rangle \rightarrow aa\langle q, 2\rangle \lfloor S, \underline{q}\rfloor}$,
$\langle p, 2\rangle \rightarrow aa\langle q, 2\rangle \lfloor X, \underline{q}\rfloor$ into P;

(v): For $(S, q, bb, f) \in R$, add $\boldsymbol{\langle q, 2\rangle \rightarrow bb\bar{R}}$ into P.

Observe that in the production $\langle s, 1\rangle \rightarrow \lceil \underline{X}, p\rceil \lceil \underline{S}, q\rceil \langle p, 1\rangle \lfloor X, \underline{p}\rfloor$ from (ii), we non-deterministically choose the nonterminal $\lceil \underline{S}, q\rceil$ for S in XS so it will match some $\lfloor S, \underline{q}\rfloor$ later to proceed the derivation. Similarly, in the production $\langle p, 2\rangle \rightarrow aa\langle q, 2\rangle \lfloor S, \underline{q}\rfloor$ from (iv), we non-deterministically choose S as the symbol in $\lfloor S, \underline{q}\rfloor$ to match $\lceil \underline{S}, q\rceil$ generated earlier.

Now, we explore how a derivation

$$\#Ss \Rightarrow S\#XSp \Rightarrow SX\#Saaq \Rightarrow SXS\#aabbf$$

in Q is simulated in H.

We explore the first phase of a derivation of $aabb$ in H:

$$\begin{array}{lrcl}
 & & S' & \\
\Rightarrow & \lceil \underline{S}, s\rceil & \langle s, 1\rangle & \lfloor S, \underline{s}\rfloor \\
\Rightarrow & \lceil \underline{S}, s\rceil \lceil \underline{X}, p\rceil \lceil \underline{S}, q\rceil & \langle p, 1\rangle & \lfloor X, \underline{p}\rfloor \lfloor S, \underline{s}\rfloor \\
\Rightarrow & \lceil \underline{S}, s\rceil \lceil \underline{X}, p\rceil \lceil \underline{S}, q\rceil \bar{L} & \langle p, 2\rangle & \lfloor X, \underline{p}\rfloor \lfloor S, \underline{s}\rfloor \\
\Rightarrow & \lceil \underline{S}, s\rceil \lceil \underline{X}, p\rceil \lceil \underline{S}, q\rceil \bar{L} & aa\,\langle q, 2\rangle & \lfloor S, \underline{q}\rfloor \lfloor X, \underline{p}\rfloor \lfloor S, \underline{s}\rfloor \\
\Rightarrow & \lceil \underline{S}, s\rceil \lceil \underline{X}, p\rceil \lceil \underline{S}, q\rceil \bar{L} & aabb & \bar{R}\,\lfloor S, \underline{q}\rfloor \lfloor X, \underline{p}\rfloor \lfloor S, \underline{s}\rfloor
\end{array}$$

Now we illustrate an inside-out erasure in the second phase of the derivation in H:

$$\begin{array}{lrcl}
 & \lceil \underline{S}, s \rceil \lceil \underline{X}, p \rceil \lceil \underline{S}, q \rceil \bar{L} & aabb & \bar{R} \lfloor S, \underline{q} \rfloor \lfloor X, \underline{p} \rfloor \lfloor S, \underline{s} \rfloor \\
\Rightarrow & \lceil \underline{S}, s \rceil \lceil \underline{X}, p \rceil \bar{L} & aabb & \bar{R} \lfloor X, \underline{p} \rfloor \lfloor S, \underline{s} \rfloor \\
\Rightarrow & \lceil \underline{S}, s \rceil \bar{L} & aabb & \bar{R} \lfloor S, \underline{s} \rfloor \\
\Rightarrow & \bar{L} & aabb & \bar{R} \\
\Rightarrow^2 & & aabb &
\end{array}$$

So H eliminates all nonterminals in the second phase to match that the non-deterministic context-free generation was proper and $aabb$ was generated. □

In the proof of Lemma 3.7, given next, we turn the grammar H, constructed in the proof of Lemma 3.5, to an equivalent scatterred context grammar G with a single non-context-free production.

In essence, the transformation of H to G is performed as follows:

1. Consider the set of productions P_1 and the set of nonterminals in H. Transform the productions of P_1 to the productions of P' in G by using two encoding functions—denoted by ι and κ in the upcoming proof—in a way that nonterminals from U that simulate the current state of Q remain intact.

2. In P', consider productions originated in step (i) in the proof of Lemma 3.5. Modify the right-hand side of each production constructed therein. Place symbol 1 in front of this side, and place 1 behind it, too.

3. On the right-hand sides of each production in P', replace $\bar{L}$ and $\bar{R}$ with 20 and 302, respectively.

To give a more detailed insight into the removal made by the only non-context-free production in the proof of Lemma 3.7, suppose that G makes

$$S \Rightarrow^* u_1 u_2 \cdots u_{m-1} u_m x v_m v_{m-1} \cdots v_2 v_1$$

where each u_j and v_j encodes a production applied during the jth derivation step in Q, and all of them have the same number of 1s. More specifically, each $u_i = {}_p u_i \, {}_s u_i$ where ${}_p u_i$ is a prefix of u_i over $\{0, 1, 3\}$ and ${}_s u_i$ is a suffix of u_i over $\{0, 1\}$. Let us call the beginning of ${}_s u_i$ as *u_i-break*. Similarly, each $v_i = {}_p v_i \, {}_s v_i$ where ${}_p v_i$ is a prefix of v_i over $\{0, 1\}$ and ${}_s v_i$ is a suffix of v_i over $\{0, 1, 3\}$. Let us call the end of ${}_p v_i$ as *v_i-break*. The position of u_i-break and v_i-break encodes a production in Q so this encoding satisfies the following property:

if (a, q, x, p) and (b, o, y, r) are two productions
encoded by the same break position, then $p = r$ and $x = y$.

During the second phase, G has to make sure that the simulation of the generation of x in Q is performed correctly. To do so, G has to verify that for $j = m, \ldots, 2, 1$, both u_j and v_j encode the same production r_j in Q. G makes this verification solely by using $(1, 2, 0, 3, 0, 2, 1) \rightarrow (2, \varepsilon, \varepsilon, \varepsilon, \varepsilon, \varepsilon, 2)$

so it eliminates all u_m through u_1 and, simultaneously, v_m through v_1 in the inside-out way. To explain the elimination process more precisely, consider this portion

$$u_1 u_2 \cdots {}_p u_i \bullet {}_s u_i \; 20 \; x \; 302 \; {}_p v_i \bullet {}_s v_i \cdots v_2 v_1$$

where $\bullet$ points out the position of u_i-break and v_i-break, respectively. G can eliminate the codes u_i and v_i if and only if u_i-break and v_i-break are simultaneously rewritten by $(1, 2, 0, 3, 0, 2, 1) \rightarrow (2, \varepsilon, \varepsilon, \varepsilon, \varepsilon, \varepsilon, 2)$; thereby, it guaranties that ${}_s u_i$ and ${}_p v_i$ have the same number of 1s and 0s, therefore, u_i and v_i encode the same production r_i. Finally, concerning the existence of substrings 20 and 302 surrounding x from left and right, respectively, a note is in order: both have originated from the previous elimination of codes u_{i+1} and v_{i+1}.

For instance,

$$\cdots 103 \cdots 1031 \bullet 10101010 \cdots 101020x302301301 \cdots 301 \bullet 03010101 \cdots 01 \cdots$$

where

$$u_i = 103 \cdots 1031 \bullet 10101010 \cdots 1010$$

and

$$v_i = 301301 \cdots 301 \bullet 03010101 \cdots 01$$

During the second phase, G simultaneously eliminates u_i and v_i by using $(1, 2, 0, 3, 0, 2, 1) \rightarrow (2, \varepsilon, \varepsilon, \varepsilon, \varepsilon, \varepsilon, 2)$ in the way sketched next

$$\begin{array}{lrcl}
 & 1103 \cdots 103110101010 \cdots 10\underline{1}0\underline{20} & x & \underline{302}30\underline{1}301 \cdots 30103010101 \cdots 011 \\
\Rightarrow & 1103 \cdots 103110101010 \cdots \underline{1}0\underline{20} & x & \underline{302}30\underline{1} \cdots 30103010101 \cdots 011 \\
\Rightarrow & 1103 \cdots 103110101010 \cdots 20 & x & 302 \cdots 30103010101 \cdots 011 \\
\Rightarrow & & & \\
 & & \vdots & \\
\Rightarrow & 1103 \cdots 1031031\underline{1}0\underline{20} & x & \underline{302}30\underline{1}0301010101 \cdots 011 \\
\Rightarrow & 1103 \cdots 103103\underline{120} & x & \underline{302}030\underline{1}010101 \cdots 011 \\
\Rightarrow & 1103 \cdots 103\underline{1}03\underline{2} & x & \underline{0302}0\underline{1}0101 \cdots 011 \\
\Rightarrow & 1103 \cdots 103203 & x & 020101 \cdots 011
\end{array}$$

(by underlining, we denote the occurrences of symbols rewritten in the next step).

Lemma 3.7. Let Q be a left-extended queue grammar. Then, there exists a scattered context grammar, $G = (K, T, P, S)$, such that $L(Q) = L(G)$, whereas P contains

$$(1, 2, 0, 3, 0, 2, 1) \rightarrow (2, \varepsilon, \varepsilon, \varepsilon, \varepsilon, \varepsilon, 2)$$

as its only non-context-free production.

Proof:
Without any loss of generality, assume that $Q = (V, T, W, F, s, R)$ is a left-extended queue grammar in normal form 2, H is an equivalent scattered context grammar constructed by the proof of Lemma 3.5 from Q, and let $D = \{0, 1, 2, 3\}$ such that $D \cap (V \cup W) = \emptyset$. Let us introduce

$$X_n = \{103\}^+\{1\}\{10\}^+ \cap \{x \colon x \in \{1, 0, 3\}^*, \text{occur}(x, 1) = n\}$$

and an injection ι from $(V - T)(W - F)$ to X_n so that ι remains an injection when its domain is extended to $((V - T)(W - F))^*$ in the standard way, where n is a positive integer great enough to allow us to introduce these notions in this way (a proof that such a constant necessarily exists is simple and left to the reader).

Notice that for some $a \in V - T$ and $q \in W - F$, $\iota(aq)$ is a string of the form

$$103103 \cdots 10311010 \cdots 10$$

in which there are n occurrences of 1 and precisely one occurrence of substring 11.

Then, define the homomorphism from $\{0, 1\}^*$ to $\{0, 1, 3\}^*$ by $\beta(0) = 30$ and $\beta(1) = 1$. Set

$$Y_n = \{301\}^+\{0301\}\{01\}^+ \cap \{x \colon x \in \{1, 0, 3\}^*, \text{occur}(x, 1) = n\}$$

and define the injection κ from $(V - T)(W - F)$ to Y_n by

$$\kappa(aq) = z10301w \quad \text{such that} \quad \begin{array}{l} \iota(aq) = u11v, a \in V - T, q \in W - F, \\ z = \beta(\text{Reverse}(v)) \text{ and } w = \text{Reverse}(\text{Erase}(u, \{3\}))\} \end{array}$$

Notice that every $\kappa(aq)$ is a string of the form

$$301301 \cdots 30103010101 \cdots 01$$

in which there are n occurrences of 1 and precisely one occurrence of 030.

Construction. Introduce the scattered context grammar $G = (U \cup D, T, P, S)$ with $P = P' \cup P''$ constructed from $H = (U \cup N \cup M, T, P_1 \cup P_2 \cup P_3, S)$ in the following way.
To construct P', perform (i) through (v), given next.

(i) For every production from P_1 of the form $S \to \lceil \underline{a_0}, q \rceil \langle q_0, 1 \rangle \lfloor a, \underline{q_0} \rfloor$, where $a_0, a \in V - T$ and $q_0, q \in W - F$,
add $S \to 1\iota(a_0q)\langle q_0, 1 \rangle \kappa(aq_0)1$ into P';

(ii) For every production from P_1 of the form $\langle q, 1 \rangle \to \lceil \underline{a_1}, q_1 \rceil \lceil \underline{a_2}, q_2 \rceil \cdots \lceil \underline{a_m}, q_m \rceil \langle p, 1 \rangle \lfloor b, \underline{p} \rfloor$, where $a_1, a_2, \ldots, a_m, b \in V - T$ and $q, q_1, q_2, \ldots, q_m \in W - F$, $m \geq 0$,
add $\langle q, 1 \rangle \to \iota(a_1q_1)\iota(a_2q_2) \cdots \iota(a_mq_m)\langle p, 1 \rangle \kappa(bp)$ into P';

(iii) For every production from P_1 of the form $\langle q, 1 \rangle \to \bar{L}\langle q, 2 \rangle$, where $q \in W - F$,
add $\langle q, 1 \rangle \to 20\langle q, 2 \rangle$ into P';

(iv) For every production from P_1 of the form $\langle q, 2\rangle \to y\langle p, 2\rangle\lfloor b, \underline{p}\rfloor$, where $b \in V - T$ and $q, p \in W - F$, $y \in T^*$,
add $\langle q, 2\rangle \to y\langle p, 2\rangle\kappa(bp)$ into P';

(v) For every production from P_1 of the form $\langle q, 2\rangle \to y\bar{R}$, where $q \in W - F$, $y \in T^*$,
add $\langle q, 2\rangle \to y302$ into P'.

Set

$$P'' = \{(1, 2, 0, 3, 0, 2, 1) \to (2, \varepsilon, \varepsilon, \varepsilon, \varepsilon, \varepsilon, 2),\ 2 \to \varepsilon\}$$

and, hereafter, denote $(1, 2, 0, 3, 0, 2, 1) \to (2, \varepsilon, \varepsilon, \varepsilon, \varepsilon, \varepsilon, 2)$ by π.

Basic Idea. To see why $L(G) = L(Q)$, we demonstrate that for any $y \in T^*$,

$$S \Rightarrow^* y \text{ in } G \text{ if and only if } \#s \Rightarrow^* w\#yf \text{ in } Q$$

with $s = a_0q_0$, $w \in (V - T)^*$, and $f \in F$ just like we did in the proof of Lemma 3.5.

More precisely, we need to show that $S \Rightarrow^* y$ in G if and only if for some $m \geq 1$, Q makes $\#a_0q_0 \Rightarrow^* a_0 \cdots a_m\#yf$ according to (a_0, q_0, z_0, q_1) through (a_m, q_m, z_m, q_{m+1}), where $q_{m+1} = f$. To see why this equivalence holds true, take any $S \Rightarrow^* y$ with $y \in L(G)$. Examine the construction of P to see that $S \Rightarrow^* y$ in G has, in a greater detail, the form

$$S \Rightarrow^* 1\nu_1 20y302\nu_2 1 \Rightarrow^* y$$

with $\nu_1, \nu_2 \in D^*$, $\nu_1 = \iota(a_0q_0) \cdots \iota(a_\ell q_\ell)$ for some $q_0, \ldots, q_\ell \in W - F$, $\nu_2 = \kappa(a_kq_k) \cdots \kappa(a_0q_0)$ for some $a_0, \ldots, a_k \in V - T$, where $\ell, k \geq 1$

Consequently, we see that proving the equivalence requires a demonstration that in the derivation of the above form in G,

(I) $m = \ell = k$;

(II) $\nu_1 = \iota(a_0q_0) \cdots \iota(a_\ell q_\ell)$ for some $q_0, \ldots, q_\ell \in W - F$ with $a_0, \ldots, a_\ell \in V - T$;

(III) $\nu_2 = \kappa(a_kq_k) \cdots \kappa(a_0q_0)$ for some $a_0, \ldots, a_k \in V - T$ with $q_0, \ldots, q_k \in W - F$.

Consider (II) above. Observe that ν_1 encodes the prefix of all the front queue symbols (including the erased symbols) rewritten during the generation of y. This is the reason why we assume that Q is a left-extended queue grammar, which records this prefix as opposed to any ordinary queue grammar, which throws it away.

From the definition of ι, it follows that

$$\begin{aligned}1\nu_1 20 \quad = \quad & 1103103 \cdots 103\underline{11}0 \cdots 1010 \cdots 103103 \cdots 103\underline{11}0 \cdots 101010 \cdots \\ & \cdots 103103 \cdots 103\underline{11}0 \cdots 1020\end{aligned}$$

Counting from the right to the left, we refer to the ith underlined occurrence of 11 as the ith *left turn*, $1 \leq i \leq n$. From the definition of κ, it follows that

$$\begin{aligned}302\nu_2 1 \quad = \quad & 302301 \cdots 301\underline{030}101 \cdots 0101 \cdots 301301301 \cdots 301\underline{030}101 \cdots \\ & \cdots 0101 \cdots 301301 \cdots 301\underline{030}101 \cdots 01011\end{aligned}$$

Counting from the left to the right, we refer to the ith underlined occurrence of 030 as the ith *right turn*.

Let us examine $1\nu_1 20y302\nu_2 1 \Rightarrow^* y$ in a greater detail. The first 1 and the last 1 are produced by a production from step (i) in the construction. Furthermore, in front of y, 20 is made by a production from (iii), and behind y, 302 is produced by a production from (v). Observe that all the left and right nonterminals can be removed only by π and $2 \to \varepsilon$. In $1\nu_1 20y302\nu_2 1$, there exist two occurrences of 2. Production π is applicable as soon as two occurrences of 2 appear in the rewritten string, and its application does not change the number of these occurrences. Consequently, during $1\nu_1 20y302\nu_2 1 \Rightarrow^* y$, $2 \to \varepsilon$ is applied only during the last two steps while all the preceding steps are made by using π. The first 2 is always a left nonterminal and the other occurs always as a right nonterminal. Considering these observations and π, we see that if a string contains 1 somewhere in between the left 2 and the right 2, then G cannot derive a terminal string from it. Consequently, during $1\nu_1 20y302\nu_2 1 \Rightarrow^* y$, G eliminates 1s in an inside-out way so that it always rewrites the rightmost occurrence within the left 1s and, simultaneously, the leftmost occurrence among the right 1s. Unless a string contains 0, 3, 0 in this order scattered somewhere in between the left 2 and the right 2, then G cannot apply π and derive a terminal string. More specifically, every successful derivation in G is of the form

$$S \Rightarrow^* 1\nu_1 20y302\nu_2 1 \Rightarrow^* 1203y021 \Rightarrow 2y2 \Rightarrow^2 y$$

Let $1v_3yv_41 \Rightarrow 1v_3'yv_4'1$ be a direct derivation step in $1\nu_1 20y302\nu_2 1 \Rightarrow^* 1203y021$. Then, in a greater detail, this step has one of these five forms

a) $1v_5\underline{1}0\underline{20}y\underline{302}30\underline{1}v_61 \Rightarrow 1v_520y302v_61$ with $v_51020 = v_3$ and $302301v_6 = v_4$;

b) $1v_5\underline{120}y\underline{302}030\underline{1}v_61 \Rightarrow 1v_52y0302v_61$ with $v_5120 = v_3$ and $3020301v_6 = v_4$;

c) $1v_5\underline{1}03\underline{2}y\underline{0302}0\underline{1}v_61 \Rightarrow 1v_5203y02v_61$ with $v_51032 = v_3$ and $030201v_6 = v_4$;

d) $1v_5\underline{1}03\underline{203}y\underline{02}0\underline{1}v_61 \Rightarrow 1v_5203y02v_61$ with $v_5103203 = v_3$ and $0201v_6 = v_4$;

e) $1v_5\underline{1}0\underline{203}y\underline{02}30\underline{1}v_61 \Rightarrow 1v_520y302v_61$ with $v_510203 = v_3$ and $02301v_6 = v_4$.

Hereafter, to point out that a derivation step $u \Rightarrow v\ [\pi]$ satisfies one specific form of the five previous forms ($X \in \{a, b, c, d, e\}$), we write $u \Rightarrow_{X)} v$.

Suppose that the leftmost right turn occurs closer to y than the rightmost left turn does. For instance,

$$1103103\cdots1031031101010101020y3020301010101\cdots01011$$

From this string, G performs these steps

$$\begin{array}{rrcl}
 & 1103103\cdots103103110101010\underline{1}0\underline{20} & y & \underline{302}030\underline{1}010101\cdots01011 \\
\Rightarrow & 1103103\cdots1031031101010\underline{1}0\underline{2}0 & y & \underline{0302}0\underline{1}0101\cdots01011 \\
\Rightarrow & 1103103\cdots1031031101010200 & y & 020101\cdots01011
\end{array}$$

Observe that in between the two 2s, no 3 occurs, so G cannot derive a terminal string from it.

Suppose that the leftmost right turn occurs farther from y than the rightmost left turn does. For instance,

$$1103103\cdots10310311020y302301301301301030101010 1\cdots01011$$

From this string, G performs these steps

$$\begin{array}{rrcl}
 & 1103103\cdots 1031031\underline{1}0\underline{20} & y & \underline{302}30\underline{1}301301301030101010\cdots 01011 \\
\Rightarrow & 1103103\cdots 103103\underline{120} & y & \underline{302}30\underline{1}3013010301010101\cdots 01011 \\
\Rightarrow & 1103103\cdots 1031032 & y & 30230130103010101\cdots 01011
\end{array}$$

Observe that in between the two 2s, only one 0 occurs, so G cannot derive a terminal string from it.

Next, we give an example for some $a \in V - T$ and $q \in W - F$, where the right and left turns match.

$$1103103\cdots 103110101010\cdots 101020y302301301\cdots 30103010101\cdots 01011$$

where

$$\iota(aq) = 103103\cdots 103110101010\cdots 1010$$

and

$$301301\cdots 30103010101\cdots 0101 = \kappa(aq)$$

Consequently, G always simultaneously eliminates the ith left turn and the ith right turn in the way sketched next

$$\begin{array}{rrcl}
 & 1103103\cdots 103110101010\cdots 10\underline{1}0\underline{20} & y & \underline{302}30\underline{1}301\cdots 30103010101\cdots 01011 \\
\Rightarrow_{a)} & 1103103\cdots 103110101010\cdots \underline{1}0\underline{20} & y & \underline{302}30\underline{1}\cdots 30103010101\cdots 01011 \\
\Rightarrow_{a)} & 1103103\cdots 103110101010\cdots 20 & y & 302\cdots 30103010101\cdots 01011 \\
\Rightarrow & & & \\
 & & \vdots & \\
\Rightarrow & 1103103\cdots 1031031\underline{1}0\underline{20} & y & \underline{302}30\underline{1}0301010101\cdots 01011 \\
\Rightarrow_{a)} & 1103103\cdots 103103\underline{120} & y & \underline{302}030\underline{1}010101\cdots 01011 \\
\Rightarrow_{b)} & 1103103\cdots 103\underline{1}03\underline{2} & y & \underline{0302}0\underline{1}0101\cdots 01011 \\
\Rightarrow_{c)} & 1103103\cdots 103203 & y & 020101\cdots 01011
\end{array}$$

In this way, G simulates the successful derivation of y performed by Q so

$$S \Rightarrow^* 1\nu_1 20y302\nu_2 1 \Rightarrow^* y$$

with $\nu_1, \nu_2 \in D^*$, $\nu_1 = \iota(a_0q_0)\cdots\iota(a_mq_m)$ for some $q_0, \dots, q_m \in W - F$ with $a_0, \dots, a_m \in V - T$, and $\nu_2 = \kappa(a_mq_m)\cdots\kappa(a_0q_0)$ for some $a_0, \dots, a_m \in V - T$ with $q_0, \dots, q_m \in W - F$. Thus, $L(Q) = L(G)$.

In the following example, we finish Example 3.6 but with nonterminals from N and M encoded into D^* using ι and κ, respectively.

Example 3.8. According to the construction, the injection ι should handle $\mathrm{card}((V - T) \times (W - F))$ elements. Thus, for simple coding, n should be at least $\mathrm{card}((V - T) \times (W - F)) + 2$. In the second phase of this example, we need to code six pairs—(S, s), (S, p), (S, q), (X, s), (X, p), and (X, q).

Take $n = 8$. Next, we introduce ι for these pairs—that is,

$$\begin{array}{rr}
\iota(Ss) = 1031(10)^6 = \lceil \underline{S}, s\rceil & \iota(Xs) = (103)^4 1(10)^3 = \lceil \underline{X}, s\rceil \\
\iota(Sp) = (103)^2 1(10)^5 = \lceil \underline{S}, p\rceil & \iota(Xp) = (103)^5 1(10)^2 = \lceil \underline{X}, p\rceil \\
\iota(Sq) = (103)^3 1(10)^4 = \lceil \underline{S}, q\rceil & \iota(Xq) = (103)^6 110 = \lceil \underline{X}, q\rceil
\end{array}$$

and $\lfloor S, \underline{s}\rfloor = (301)^6 030101$, $\lfloor S, \underline{q}\rfloor = (301)^4 0301(01)^3$, and $\lfloor X, \underline{p}\rfloor = (301)^2 0301(01)^5$.

Now we illustrate an erasure in the second phase of the derivation such that

$$1\lceil \underline{S}, s\rceil\lceil \underline{X}, p\rceil\lceil \underline{S}, q\rceil 20y302\lfloor S, \underline{q}\rfloor\lfloor X, \underline{p}\rfloor\lfloor S, \underline{s}\rfloor 1 \Rightarrow^* y$$

with $y = aabb$ in G:

$$\begin{array}{lrcl}
 & 1\lceil \underline{S}, s\rceil\,\lceil \underline{X}, p\rceil\,(103)^3 1101010\underline{1}0\,\underline{20} & y & \underline{302}\,30\underline{1}3013013010301(01)^3\,\lfloor X, \underline{p}\rfloor\,\lfloor S, \underline{s}\rfloor 1 \\
\Rightarrow_{a)} & 1\lceil \underline{S}, s\rceil\,\lceil \underline{X}, p\rceil\,10310310311010\underline{1}0\underline{20} & y & \underline{302}30\underline{1}3013010301010101\,\lfloor X, \underline{p}\rfloor\,\lfloor S, \underline{s}\rfloor 1 \\
\Rightarrow_{a)} & 1\lceil \underline{S}, s\rceil\,\lceil \underline{X}, p\rceil\,103103103110\underline{1}0\underline{20} & y & \underline{302}30\underline{1}3010301010101\,\lfloor X, \underline{p}\rfloor\,\lfloor S, \underline{s}\rfloor 1 \\
\Rightarrow_{a)} & 1\lceil \underline{S}, s\rceil\,\lceil \underline{X}, p\rceil\,1031031031\underline{1}0\underline{20} & y & \underline{302}30\underline{1}030\underline{1}010101\,\lfloor X, \underline{p}\rfloor\,\lfloor S, \underline{s}\rfloor 1 \\
\Rightarrow_{a)} & 1\lceil \underline{S}, s\rceil\,\lceil \underline{X}, p\rceil\,103103103\underline{120} & y & \underline{302}030\underline{1}010101\,\lfloor X, \underline{p}\rfloor\,\lfloor S, \underline{s}\rfloor 1 \\
\Rightarrow_{b)} & 1\lceil \underline{S}, s\rceil\,\lceil \underline{X}, p\rceil\,103103\underline{1}03\underline{2} & y & \underline{0302}0\underline{1}0101\,\lfloor X, \underline{p}\rfloor\,\lfloor S, \underline{s}\rfloor 1 \\
\Rightarrow_{c)} & 1\lceil \underline{S}, s\rceil\,\lceil \underline{X}, p\rceil\,103\underline{1}03\underline{203} & y & \underline{02}0\underline{1}01\,\lfloor X, \underline{p}\rfloor\,\lfloor S, \underline{s}\rfloor 1 \\
\Rightarrow_{d)} & 1\lceil \underline{S}, s\rceil\,\lceil \underline{X}, p\rceil\,\underline{1}03\underline{203} & y & \underline{02}0\underline{1}\,\lfloor X, \underline{p}\rfloor\,\lfloor S, \underline{s}\rfloor 1 \\
\Rightarrow_{d)} & 1\lceil \underline{S}, s\rceil\,103103103103103110\underline{1}0\,\underline{203} & y & \underline{02}\,30\underline{1}30103010101010101\,\lfloor S, \underline{s}\rfloor 1 \\
\Rightarrow_{e)} & 1\lceil \underline{S}, s\rceil\,1031031031031031\underline{1}0\,\underline{20} & y & \underline{302}\,30\underline{1}03010101010101\,\lfloor S, \underline{s}\rfloor 1 \\
\Rightarrow_{a)} & 1\lceil \underline{S}, s\rceil\,103103103103103\underline{1}\,\underline{20} & y & \underline{302}\,030\underline{1}0101010101\,\lfloor S, \underline{s}\rfloor 1 \\
\Rightarrow_{b)} & 1\lceil \underline{S}, s\rceil\,103103103103\underline{1}03\,\underline{2} & y & \underline{0302}\,0\underline{1}01010101\,\lfloor S, \underline{s}\rfloor 1 \\
\Rightarrow_{c)} & 1\lceil \underline{S}, s\rceil\,1031031\underline{1}0\underline{20} & y & \underline{302}30\underline{1}03010101\,\lfloor S, \underline{s}\rfloor 1 \\
\Rightarrow_{a)} & 1\lceil \underline{S}, s\rceil\,103103\underline{120} & y & \underline{302}030\underline{1}0101\,\lfloor S, \underline{s}\rfloor 1 \\
\Rightarrow_{b)} & 1\lceil \underline{S}, s\rceil\,103\underline{1}03\underline{2} & y & \underline{0302}0\underline{1}01\,\lfloor S, \underline{s}\rfloor 1 \\
\Rightarrow_{c)} & 1\lceil \underline{S}, s\rceil\,\underline{1}03\underline{203} & y & \underline{02}0\underline{1}\,\lfloor S, \underline{s}\rfloor 1 \\
\Rightarrow_{d)} & 1\,10311010101010\underline{1}0\,\underline{203} & y & \underline{02}\,30\underline{1}301301301301030101\,1 \\
\Rightarrow_{e)} & 1\,103110101010\underline{1}0\,\underline{20} & y & \underline{302}\,30\underline{1}301301301030101\,1 \\
\Rightarrow_{a)} & 1\,1031101010\underline{1}0\,\underline{20} & y & \underline{302}\,30\underline{1}301301030101\,1 \\
\Rightarrow_{a)} & 1\,10311010\underline{1}0\,\underline{20} & y & \underline{302}\,30\underline{1}301030101\,1 \\
\Rightarrow_{a)} & 1\,103110\underline{1}0\,\underline{20} & y & \underline{302}\,30\underline{1}030101\,1 \\
\Rightarrow_{a)} & 1\,1031\underline{1}0\,\underline{20} & y & \underline{302}\,30\underline{1}030101\,1 \\
\Rightarrow_{a)} & 1\,103\underline{120} & y & \underline{302}030\underline{1}01\,1 \\
\Rightarrow_{b)} & 1\,\underline{1}03\underline{2} & y & \underline{0302}0\underline{1}\,1 \\
\Rightarrow_{c)} & \underline{1203} & y & \underline{021} \\
\Rightarrow & 2 & y & 2 \\
\Rightarrow^2 & & y &
\end{array}$$

Formal Proof. For brevity and readability, the following rigorous proof omits some obvious details, which the reader can easily fill in.

Define the function Θ from X_n^* to Y_n^* recursively as follows

1. $\Theta(\varepsilon) = \varepsilon$
2. If $\Theta(x) = y, i \in \{1, \ldots, n{-}2\}, u \in X_n, u = 103103 \cdots 1031(10)^i, v \in Y_n, v = (301)^i030101 \cdots 0101$, then $\Theta(ux) = yv$.

To illustrate, assume that $10310311010 \in X_n$ and $1031101010 \in X_n$; then,

$$\Theta(10310311010\,1031101010) = 301301301030101\,30130103010101$$

To make the formal specification of encoded sentential forms in G easier to follow, define two functions, $\nu\colon (V-T)^* \to 2^{X_n^*}$ and $\mu\colon Q^* \to 2^{Y_n^*}$, which encode all possible strings of symbols and sequences of states, respectively. For $k \geq 0$,

$$\nu(a_1a_2\cdots a_k) = \{\iota(a_1q_1a_2q_2\cdots a_kq_k)\colon q_1, q_2, \ldots, q_k \in Q\} \text{ where } a_1, a_2, \ldots, a_k \in V-T$$

and

$$\mu(q_1q_2\cdots q_k) = \{\kappa(a_1q_1a_2q_2\cdots a_kq_k)\colon a_1, a_2, \ldots, a_k \in V-T\} \text{ where } q_1, q_2, \ldots, q_k \in Q.$$

Claim 3.9, proved next, establishes a derivation form by which G can generate each member of $L(G)$. This claim fulfills a crucial role in the demonstration that $L(G) \subseteq L(Q)$, given later in this proof (see Claim 3.11).

Claim 3.9. G constructed in Lemma 3.7 can generate every $h \in L(G)$ in this way

$$\begin{aligned}
&S\\
&\Rightarrow 1g_0\langle q_0, 1\rangle t_0 1 \Rightarrow 1g_1\langle q_1, 1\rangle t_1 1 \Rightarrow \ldots\\
&\Rightarrow 1g_k\langle q_k, 1\rangle t_k 1 \Rightarrow 1g_k 20\langle q_k, 2\rangle t_k 1\\
&\Rightarrow 1g_k 20 y_1\langle q_{k+1}, 2\rangle t_{k+1} 1 \Rightarrow 1g_k 20 y_1 y_2\langle q_{k+2}, 2\rangle t_{k+2} 1 \Rightarrow \ldots\\
&\Rightarrow 1g_k 20 y_1 y_2 \cdots y_{m-1}\langle q_{k+m-1}, 2\rangle t_{k+m-1} 1\\
&\Rightarrow 1g_k 20 y_1 y_2 \cdots y_{m-1} y_m 302 t_{k+m-1} 1\\
&\Rightarrow 1u_1 y_1 y_2 \cdots y_{m-1} y_m v_1 1 \Rightarrow 1u_2 y_1 y_2 \cdots y_{m-1} y_m v_2 1 \Rightarrow \ldots\\
&\Rightarrow 1u_\ell y_1 y_2 \cdots y_{m-1} y_m v_\ell 1 \Rightarrow 2 y_1 y_2 \cdots y_{m-1} y_m 2 \Rightarrow^2 y_1 y_2 \cdots y_{m-1} y_m = h
\end{aligned}$$

in G, where $k, \ell, m \geq 1$; $q_0, q_1, \ldots, q_{k+m-1} \in W - F$; $y_1, \ldots, y_m \in T^*$; $t_i \in \mu(q_i \cdots q_1 q_0)$ for $i = 0, 1, \ldots, k+m-1$; $g_j \in \nu(d_0 d_1 \cdots d_j)$ with $d_0 = a_0 \in V-T$ where $s = a_0 q_0$ and $d_1, \ldots, d_j \in (V-T)^*$ for $j = 0, 1, \ldots, k$; $d_0 d_1 \cdots d_k = a_0 a_1 \cdots a_{k+m-1}$ with $a_1, \ldots, a_{k+m-1} \in V-T$ (that is, $g_k \in \nu(a_0 a_1 \cdots a_{k+m-1})$); $1u_i y_1 y_2 \cdots y_{m-1} y_m v_i 1$ satisfies either (a) $1020 \in \text{suffix}(u_i)$ and $302301 \in \text{prefix}(v_i)$ or (b) $120 \in \text{suffix}(u_i)$ and $3020301 \in \text{prefix}(v_i)$ or (c) $1032 \in \text{suffix}(u_i)$ and $030201 \in \text{prefix}(v_i)$ or (d) $103203 \in \text{suffix}(u_i)$ and $0201 \in \text{prefix}(v_i)$ or (e) $10203 \in \text{suffix}(u_i)$ and $02301 \in \text{prefix}(v_i)$ so during $1u_i y_1 y_2 \cdots y_{m-1} y_m v_i 1 \Rightarrow 1u_{i+1} y_1 y_2 \cdots y_{m-1} y_m v_{i+1} 1$, G simultaneously rewrites these prefixes and suffixes by π for $0 \leq i \leq \ell - 1$, where $u_0 = g_k 20$, $v_0 = 302 t_{k+m-1}$, $u_\ell = 203$, $v_\ell = 02$; $\Theta(g_k) = t_{k+m-1}$.

Proof:
Examine the construction of P in G. Consider P'. Observe that every derivation begins with an application of a production having S on its left-hand side. Set

$$P'_1 = \{p\colon p \in P' \text{ and } \mathrm{lhs}(p) = \langle p, 1\rangle\}$$
$$P'_2 = \{p\colon p \in P' \text{ and } \mathrm{lhs}(p) = \langle p, 2\rangle\}$$

Consider any successful derivation that generates $h \in L(G)$. Let us make some observations. All applications of productions from P'_1 precede the applications of productions from P'_2. After applications of productions from P'_1, the current sentential form contains precisely two occurrences of 2. Furthermore, an application of

$$\pi = (1, 2, 0, 3, 0, 2, 1) \to (2, \varepsilon, \varepsilon, \varepsilon, \varepsilon, \varepsilon, 2)$$

requires the occurrence of 3 in the sentential form, and this occurrence is produced only by a production constructed in step (v) of the construction of P'. After a production from step (v) is applied, no production from P' can be applied throughout the rest of the derivation, so only productions from P'' can be used during this rest. An application of $2 \to \varepsilon$ eliminates 2. After this elimination, π is inapplicable. Thus, G applies $2 \to \varepsilon$ during the last two steps of the derivation. Taking these observations into account, we see that the generation of $h \in L(G)$ can be expressed as

$$\begin{aligned}
&S\\
\Rightarrow\ & 1g_0\langle q_0, 1\rangle t_0 1 \Rightarrow 1g_1\langle q_1, 1\rangle t_1 1 \Rightarrow \dots\\
\Rightarrow\ & 1g_k\langle q_k, 1\rangle t_k 1 \Rightarrow 1g_k 20\langle q_k, 2\rangle t_k 1\\
\Rightarrow\ & 1g_k 20 y_1\langle q_{k+1}, 2\rangle t_{k+1} 1 \Rightarrow 1g_k 20 y_1 y_2\langle q_{k+2}, 2\rangle t_{k+2} 1 \Rightarrow \dots\\
\Rightarrow\ & 1g_k 20 y_1 y_2 \cdots y_{m-1}\langle q_{k+m-1}, 2\rangle t_{k+m-1} 1\\
\Rightarrow\ & 1g_k 20 y_1 y_2 \cdots y_{m-1} y_m 302 t_{k+m-1} 1\\
\Rightarrow^*\ & 2y_1 y_2 \cdots y_{m-1} y_m 2 \Rightarrow^2 y_1 y_2 \cdots y_{m-1} y_m = h
\end{aligned}$$

in G, π is the only production applied during $1g_k 20 y_1 y_2 \cdots y_{m-1} y_m 302 t_{k+m-1} 1 \Rightarrow^* 2y_1 y_2 \cdots y_{m-1}$ $y_m 2$, and all the other involved symbols satisfy what is stated in Claim 3.9 (these symbols include gs, and ts).

Before going further, let us consider any three strings of the form $103103 \cdots 1031(10)^i \in X_n$, 20302, $(301)^j 030101 \cdots 0101 \in Y_n$, where $i, j \in \{1, \dots, n-2\}$ (see the definition of X_n for n) and study how to erase the concatenation

$$103103 \cdots 1031(10)^i 20302(301)^j 030101 \cdots 0101$$

by repeatedly applying π. We intend to demonstrate that this erasure can be performed provided that $i = j$. First of all, notice that an occurrence of 1 between the two occurrences of 2 implies that this occurrence of 1 cannot be removed by π. Thus, π is always applied so the nearest possible pair of 1s that encloses 2s are rewritten. Specifically, the two underlined 1s are changed to 2s in

$$103103 \cdots 1031(10)^{i-1}\underline{1}020302301\!\!\!\underline{}(301)^{j-1}030101 \cdots 0101$$

by using π. Next, we show that if $i \neq j$, then G cannot erase $103103 \cdots 1031(10)^i 20302(301)^j 030101 \cdots 0101$.

Let $i < j$. As G always rewrites the nearest possible pair of 1s that encloses the two 2s, it obtains

$$103103 \cdots 103\underline{23}02(301)^{j-i-1}030101 \cdots 0101$$

after $i + 1$ derivation steps. As between 2 and 3 appears no 0, π is inapplicable (see the underlined symbols), which rules out the erasure.

Let $j < i$. After making j steps, G obtains

$$103103 \cdots 1031(10)^{i-j}20302030101 \cdots 0101,$$

from which G directly derives $103103 \cdots 1031(10)^{i-j-1}20030201 \cdots 0101$. After the next change of the closest pair of 1s to 2s, G obtains a string with no 3 occurring between the two 2s. As a result, π is inapplicable, and the erasure is ruled out. Consequently, $i = j$.

Return to

$$1g_k 20 y_1 y_2 \cdots y_{m-1} y_m 302 t_{k+m-1} 1 \Rightarrow^* 2 y_1 y_2 \cdots y_{m-1} y_m 2$$

with $g_k \in \nu(a_0 a_1 \cdots a_{k+m-1})$ and $t_{k+m-1} \in \mu(q_{k+m-1} \cdots q_1 q_0)$. We have demonstrated that the erasure of $103103 \cdots 1031(10)^i 20302(301)^j 030101 \cdots 0101$ implies $i = j$. Considering this implication together with the definitions of ν, μ and Θ, we see that $1g_k 20302 t_{k+m} 1 \Rightarrow^* 22$ with $\Theta(g_k) = t_{k+m-1}$. Consequently, to express $1g_k 20 y_1 y_2 \cdots y_{m-1} y_m 302 t_{k+m-1} 1 \Rightarrow^* 2 y_1 y_2 \cdots y_{m-1} y_m 2$ in a step-by-step way, we have

$$1g_k 20 y_1 y_2 \cdots y_{m-1} y_m 302 t_{k+m-1} 1 \Rightarrow 1 u_1 y_1 y_2 \cdots y_{m-1} y_m v_1 1 \Rightarrow 1 u_2 y_1 y_2 \cdots$$
$$y_{m-1} y_m v_2 1 \Rightarrow \ldots \Rightarrow 1 u_\ell y_1 y_2 \cdots y_{m-1} y_m v_\ell 1 \Rightarrow 2 y_1 y_2 \cdots y_{m-1} y_m 2$$

in G, where $1 u_i y_1 y_2 \cdots y_{m-1} y_m v_i 1$ satisfies either

(a) $1020 \in \text{suffix}(u_i)$ and $302301 \in \text{prefix}(v_i)$ or

(b) $120 \in \text{suffix}(u_i)$ and $3020301 \in \text{prefix}(v_i)$ or

(c) $1032 \in \text{suffix}(u_i)$ and $030201 \in \text{prefix}(v_i)$ or

(d) $103203 \in \text{suffix}(u_i)$ and $0201 \in \text{prefix}(v_i)$ or

(e) $10203 \in \text{suffix}(u_i)$ and $02301 \in \text{prefix}(v_i)$

so during $1 u_i y_1 y_2 \cdots y_{m-1} y_m v_i 1 \Rightarrow 1 u_{i+1} y_1 y_2 \cdots y_{m-1} y_m v_{i+1} 1$, G simultaneously rewrites these prefixes and suffixes by π for $0 \leq i \leq \ell - 1$, where $u_0 = g_k 20, v_0 = 302 t_{k+m-1}$, and $\Theta(g_k) = t_{k+m-1}$. Of course,

$$2 y_1 y_2 \cdots y_{m-1} y_m 2 \Rightarrow^2 y_1 y_2 \cdots y_{m-1} y_m$$

is performed by applying $2 \to \varepsilon$ twice.

Putting all these partial derivations and their properties together, we obtain Claim 3.9. $\square$

Claim 3.10. Q generates every $h \in L(Q)$ in this way

$$\begin{array}{ll}
\quad \#a_0q_0 & \\
\Rightarrow a_0\#x_0q_1 & [(a_0, q_0, z_0, q_1)] \\
\Rightarrow a_0a_1\#x_1q_2 & [(a_1, q_1, z_1, q_2)] \\
\quad \vdots & \\
\Rightarrow a_0a_1\cdots a_k\#x_kq_{k+1} & [(a_k, q_k, z_k, q_{k+1})] \\
\Rightarrow a_0a_1\cdots a_ka_{k+1}\#x_{k+1}y_1q_{k+2} & [(a_{k+1}, q_{k+1}, y_1, q_{k+2})] \\
\quad \vdots & \\
\Rightarrow a_0a_1\cdots a_ka_{k+1}\cdots a_{k+m-1}\#x_{k+m-1}y_1\cdots & \\
\quad y_{m-1}q_{k+m} & [(a_{k+m-1}, q_{k+m-1}, y_{m-1}, q_{k+m})] \\
\Rightarrow a_0a_1\cdots a_ka_{k+1}\cdots a_{k+m}\#y_1\cdots y_mq_{k+m+1} & [(a_{k+m}, q_{k+m}, y_m, q_{k+m+1})]
\end{array}$$

where $k \geq 0$, $m \geq 1$, $a_i \in V - T$ for $i = 0, \ldots, k+m$, $x_j \in (V-T)^*$ for $j = 1, \ldots, k+m-1$, $s = a_0q_0; a_{j'}x_{j'} = x_{j'-1}z_{j'}$ for $j' = 1, \ldots, k$, $a_1\cdots a_kx_{k+1} = z_0\cdots z_k$, $a_{k+1}\cdots a_{k+m} = x_k$, $q_0, q_1, \ldots, q_{k+m} \in W - F$ and $q_{k+m+1} \in F$, $z_1, \ldots, z_k \in (V-T)^*$, $y_1, \ldots, y_m \in T^*$, $h = y_1y_2 \cdots y_{m-1}y_m$.

Proof:
Recall that Q is in normal form 2. Considering the properties of normal form 2 (see Definition 3.3), we see that Claim 3.10 holds true. □

Claim 3.11. Let G generate $h \in L(G)$ in the way described in Claim 3.9; then, $h \in L(Q)$.

Proof:
Let $h \in L(G)$. Consider the generation of h as described in Claim 3.9. Examine the construction of P to see that at this point R contains $(a_0, q_0, z_0, q_1), \ldots, (a_k, q_k, z_k, q_{k+1})$, $(a_{k+1}, q_{k+1}, y_1, q_{k+2})$, $\ldots, (a_{k+m-1}, q_{k+m-1}, y_{m-1}, q_{k+m})$, $(a_{k+m}, q_{k+m}, y_m, q_{k+m+1})$, where $z_1, \ldots, z_k \in (V-T)^*$, and $y_1, \ldots, y_m \in T^*$. Then, Q derives h in the way described in Claim 3.10. Thus, $h \in L(Q)$. □

Claim 3.12. Let Q generate $h \in L(Q)$ in the way described in Claim 3.10; then, $h \in L(G)$.

Proof:
Let $h \in L(Q)$. Consider the generation of h as described in Claim 3.10. Examine the construction of P. Among other productions, this construction makes

1. $S \rightarrow 1\nu_0\langle q_0, 1\rangle\mu_0$ by step (i) where $\nu_0 \in \nu(a_0)$;
2. $\langle q_{j-1}, 1\rangle \rightarrow \nu_j\langle q_j, 1\rangle\mu_j$ by step (ii) where $j = 1, \ldots, k$;
3. $\langle q_k, 1\rangle \rightarrow 20\langle q_k, 2\rangle$ by step (iii);
4. $\langle q_{k+j'-1}, 2\rangle \rightarrow y_{j'}\langle q_{k+j'}, 2\rangle\mu_{k+j'+1}$ by step (iv) where $j' = 1, \ldots, m-1$;

5. $\langle q_{k+m-1}, 2\rangle \to y_m302$ by step (v);

where $\nu_i \in \nu(d_i)$, $d_i \in (V-T)^*$ for $i = 1, \ldots, k$, $\mu_{i'} \in \mu(q_{i'})$ for $i' = 0, 1, \ldots, k+m-1$. Then, by additional applications of π followed by two applications of $2 \to \varepsilon$, G derives h in the way described in Claim 3.9. Thus, $h \in L(G)$. □

Claims 3.9 through 3.12 imply that $L(Q) = L(G)$. Furthermore, $(1, 2, 0, 3, 0, 2, 1) \to (2, \varepsilon, \varepsilon, \varepsilon, \varepsilon, \varepsilon, 2)$ is the only non-context-free production in P. Therefore, Lemma 3.7 holds. □

Theorem 3.13. For every recursively enumerable language, L, there exists a scattered context grammar, $G = (K, T, P, S)$, such that $L = L(G)$ and P contains a single non-context-free production.

Proof:
By Theorem 4.3 in [13], for every recursively enumerable language, L, there exists a left-extended queue grammar that generates L. Thus, this theorem follows from Lemma 3.7. □

As its main result, the present paper demonstrated the computational completeness resulting from scattered context grammars with a single non-context-free production (see Theorem 3.13). Can this result be improved so it simultaneously reduces the number of nonterminals?

Acknowledgements

The authors thank Jakub Martiško for his useful comments and all anonymous referees for their invaluable comments and suggestions. This work was supported by The Ministry of Education, Youth and Sports of the Czech Republic from the National Programme of Sustainability (NPU II), project IT4Innovations excellence in science – LQ1602 and by the Ministry of Education, Youth and Sports of Czech Republic project ERC.CZ no. LL1908.

This paper is dedicated to the memory of Ivana.

References

[1] Rozenberg G, Salomaa A (eds.). Handbook of Formal Languages, Vol. 1: Word, Language, Grammar. Springer, New York, 1997.

[2] Greibach S, Hopcroft J. Scattered Context Grammars. *Journal of Computer and System Sciences*, 1969. **3**:233–247.

[3] Meduna A, Techet J. Scattered Context Grammars and their Applications. WIT Press, UK. WIT Press, 2010. ISBN 978-1-84564-426-0. URL http://www.fit.vutbr.cz/research/view_pub.php?id=8997.

[4] Masopust T. On the descriptional complexity of scattered context grammars. *Theor. Comput. Sci.*, 2009. **410**(1):108–112. doi:10.1016/j.tcs.2008.10.017. URL https://doi.org/10.1016/j.tcs.2008.10.017.

[5] Masopust T. Bounded Number of Parallel Productions in Scattered Context Grammars with Three Nonterminals. *Fundam. Inform.*, 2010. **99**(4):473–480. doi:10.3233/FI-2010-258. URL https://doi.org/10.3233/FI-2010-258.

[6] Křivka Z, Martiško J, Meduna A. CD Grammar Systems with Two Propagating Scattered Context Components Characterize the Family of Context Sensitive Languages. *International Journal of Foundations of Computer Science*, in press. URL `https://www.fit.vut.cz/research/publication/11604`.

[7] Csuhaj-Varjú E, Vaszil G. Scattered context grammars generate any recursively enumerable languages with two nonterminals. *Information Processing Letters*, 2010. **110**(20):902–907.

[8] Meduna A. Generative power of three-nonterminal scattered context grammars. *Theoretical Computer Science*, 2000. **246**(1):279 – 284. doi:https://doi.org/10.1016/S0304-3975(00)00153-5. URL `http://www.sciencedirect.com/science/article/pii/S0304397500001535`.

[9] Meduna A. Automata and Languages: Theory and Applications. Springer, London, 2000.

[10] Meduna A, Zemek P. Regulated Grammars and Automata. Springer, 2014. ISBN 978-1-4939-0368-9. URL `http://www.fit.vutbr.cz/research/view_pub.php?id=10498`.

[11] Kleijn HCM, Rozenberg G. On the generative power of regular pattern grammars. *Acta Informatica*, 1983. **20**(4):391–411. doi:10.1007/BF00264281. URL `https://doi.org/10.1007/BF00264281`.

[12] Kolář D, Meduna A. Regulated Pushdown Automata. *Acta Cybernetica*, 2000. **2000**(4):653–664. URL `https://www.fit.vut.cz/research/publication/6020`.

[13] Meduna A, Horáček P, Tomko M. Handbook of Mathematical Models for Languages and Computation. The Institution of Engineering and Technology, 2020. ISBN 978-1-78561-659-4.