BRNO UNIVERSITY OF TECHNOLOGY FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2016
http://excel.fit.vutbr.cz

On the Generative Power of CD Grammar Systems With Scattered Context Components

Jakub Martiško*, Alexander Meduna**

Abstract
There is a long time open problem regarding the generative power of scattered context grammars without erasing rules. This problem tries to compare their generative power with context-sensitive grammars. In this paper, modified version of this problem will be presented. This modified version combines scattered context grammars with another model — grammar systems. It will be shown that when considering this modified version of the problem, those two models have indeed same generative power. Furthermore, it will be shown that this property holds even for grammar systems with only two components where each component contains only scattered context rules with degree of two or less. Even though this does not solve the original problem, this modified version can be useful when solving the standard variant of the problem.

Keywords: Context-Sensitive Grammars — Scattered Context Grammars — Grammar Systems — Formal Languages — Generative Power

Supplementary Material: *N/A*

*imartisko@fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*
**meduna@fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Introduction

Scattered context grammars (SCG) were first introduced in the late sixties by S. Greibach and J. Hopcroft in [1]. Scattered context grammar works as parallel rewriting model that during each derivation step rewrites finite number of noterminals in the current sentential form. Each of these rewritings is done in a context-free way. This model may thus serve as a compromise between relative simplicity of context-free grammars and generative power of more complex models.

Even though the scattered context grammars are quite old, there is still open problem[1] regarding their generative power [2]. It is known that SCGs with ε-rules are as powerful as Turing Machines. As for the SCGs without ε-rules, it is known that the family of languages generated by them is a subset of family of context-sensitive languages. It is however not known whether this subset is proper. In this paper, we will propose a modification of this problem which uses combination of SCGs and grammar systems. We will show, that this combined model has the same generative power as context-sensitive languages.

Furthermore, we will show that this property holds even when limiting the number of components of grammar system to two. Similarly, the scattered context degree of all of the rules of those two components need to be of degree of no more than two.

2. Preliminaries

We assume that the reader is familiar with formal language theory [2].

A *Scattered context grammar (SCG)* is a quadruple $G = (N, T, P, S)$, where N and T are alphabets of nonterminal and terminal symbols respectively, where $N \cap T = \emptyset$, further let $V = N \cup T$. $S \in N$ is starting symbol. P is a nonempty finite set of rules of the form $(A_1, \dots, A_n) \rightarrow (\alpha_1, \dots \alpha_n)$, where $A_i \in N$, $\alpha_i \in V^*, 1 \leq i \leq n, n \geq 1$. Let $u, v \in V^*$, relation of

[1]This problem is known as $\mathscr{L}(PSCG) = \mathscr{L}(CS)$ problem.

[2]See [3] for details.

direct derivation, denoted as $u \Rightarrow v$, is defined such that following holds:

1. $u = u_1A_1u_2A_2u_3 \dots u_nA_nu_{n+1}$
2. $v = u_1\alpha_1u_2\alpha_2u_3 \dots u_n\alpha_nu_{n+1}$
3. $(A_1, A_2, A_3, \dots, A_n) \rightarrow (\alpha_1, \alpha_2, \alpha_3, \dots, \alpha_n) \in P$

where $u_i \in V^*$ for $1 \leq i \leq n+1$. Language generated by SCG G is defined as $L(G) = \{x : S \Rightarrow^* x, x \in T^*\}$, where $\Rightarrow^*$ denotes transitive and reflexive closure and $\Rightarrow^+$ transitive closure of $\Rightarrow$. A SCG grammar is said to be *propagating (PSCG)* iff each rule $p \in P$ satisfies $(A_1, \dots, A_n) \rightarrow (\alpha_1, \dots \alpha_n)\alpha_i \neq \varepsilon, \forall i : 1 \leq i \leq n$. By $\mathscr{L}(SCG)$ and $\mathscr{L}(PSCG)$, the families of languages generated by SCGs and PSCGs respectively is denoted.

A *Context-sensitive grammar (CSG)* is a quadruple $G = (N, T, P, S)$, where N and T are alphabets of nonterminal and terminal symbols respectively, where $N \cap T = \emptyset$, further let $V = N \cup T$. $S \in N$ is starting symbol. P is a nonempty finite set of rules of the form $\alpha A \beta \rightarrow \alpha\gamma\beta$, where $A \in N$, $\alpha, \beta, \gamma \in V^*$ and $\gamma \neq \varepsilon$. Let $u, v \in V^*$, relation of direct derivation, denoted as $u \Rightarrow v$, is defined such that following holds:

1. $u = u_1\alpha A\beta u_2$
2. $v = u_1\alpha\gamma\beta u_2$
3. $\alpha A\beta \rightarrow \alpha\gamma\beta \in P$

where $u_1, u_2, \alpha, \beta, \gamma \in V^*, A \in N$, $\gamma \neq \varepsilon$. The language generated by CSG G is defined as $L(G) = \{x : S \Rightarrow^* x, x \in T^*\}$, where $\Rightarrow^*$ denotes transitive and reflexive closure and $\Rightarrow^+$ transitive closure of $\Rightarrow$. Language generated by CSG $G = (N, T, P, S)$ is defind as $L(G) = \{x : S \Rightarrow^* x, x \in T^*\}$. By $\mathscr{L}(CS)$ the family of languages generated by CSGs is denoted.

A CSG is in *Kuroda Normal form*, if the set P contains only rules of one of the following forms:

1. $AB \rightarrow CD$
2. $A \rightarrow CD$
3. $A \rightarrow C$
4. $A \rightarrow a$

where $A, B, C, D \in N$ and $a \in T$. It has been proven [3] that every CSG can be transformed into an equivalent grammar in Kuroda normal form. Without loss of generality, further on, we will assume only CSGs that are in this form.

A *cooperating distributed grammar system (CDGS)* of degree n is n+3 tuple $G = (N, T, S, P_1, P_2, \dots, P_n)$, where N and T are alphabets of nonterminal and terminal symbols respectively, where $N \cap T = \emptyset$, further let $V = N \cup T$. $S \in N$ is starting symbol. $P_i, 1 \leq i \leq n$ are nonempty finite sets (called components) of rewriting rules over V. For a CDGS $G = (N, T, S, P_1, P_2, \dots, P_n)$, the terminating (t) derivation by the i-th component, denoted as $\Rightarrow_{P_i}{}^t$ is defined as $u \Rightarrow_{P_i}{}^t v$ iff $u \Rightarrow_{P_i}^* v$ and there is no $z \in V^*$ such that $v \Rightarrow_{P_i} z$. The language generated by CDGS $G = (N, T, S, P_1, P_2, \dots, P_n)$ working in t mode is defined as $L(G) = \{x : S \Rightarrow_{P_{i_1}}{}^t x_1 \Rightarrow_{P_{i_2}}{}^t x_2 \dots \Rightarrow_{P_{i_m}}{}^t x, m \geq 1, 1 \leq i_j \leq n, 1 \leq j \leq m, x \in T^*\}$.

In this paper, CDGS with propagating scattered context rules (SCGS) and CDGS with context-sensitive rules will be considered.

3. Main Results

In this section, the equality of $\mathscr{L}(SCGS)$ and $\mathscr{L}(CS)$ will be shown.

Lemma 1. $\mathscr{L}(SCGS) \subseteq \mathscr{L}(CS)$

Proof. [4] shows, that any scattered context grammar can be simulated by context-sensitive grammar. [5] further shows that any CDGS with context-sensitive components working in t mode can be transformed to equivalent CSG. Based on those two facts, it is easy to show that any SCGS can be simulated by CSG. □

Lemma 2. $\mathscr{L}(CS) \subseteq \mathscr{L}(SCGS)$

Suppose CSG $G = (N, T, P, S)$ satisfying Kuroda normal form. An equivalent SCGS $\Gamma = (N_{GS}, T, {}^+S, P_1, P_2)$ can be created using the following constructions. First, let N_{GS} be union of $N \cup \{!\}$ $(! \notin N \cup T)$ and following sets:

$$\begin{aligned}
N_T &= \{a' : \forall a \in T\} \\
N_+ &= \{{}^+X : \forall X \in N \cup N_T\} \\
N_- &= \{{}^-X : \forall X \in N \cup N_T\} \\
N_* &= \{{}^*X : \forall X \in N \cup N_T\} \\
N_{first} &= N_+ \cup N_- \cup N_* \\
N_{CF} &= \{{}_|X_| : \forall X \in N \cup N_{first}\} \\
N_{CS} &= \{{}_|X_< : \forall X \in N \cup N_{first}\} \cup \{{}_>X_| : \forall X \in N\} \\
N_{cur} &= \{X^\wedge_< : \forall X \in N \cup N_{first}\} \cup \{X^\wedge_| : \forall X \in N \cup N_{first}\}.
\end{aligned}$$

Further on, we call subsets of N_{CF}, N_{CS} and N_{cur} constructed using the set N_+ as N_{CF+}, N_{CS+} and N_{cur+}, respectively. We will use similar naming convention for subsets constructed using N_- and N_*.

Based on these sets, we can now proceed with the construction of set P_1 which is defined as an union of following sets:

$$\begin{aligned}
P_T^1 = &\{({}^+X) \rightarrow ({}^*X) : \forall X \in N \cup N_T\} \\
&\cup \{({}^*X, a') \rightarrow ({}^*X, a) : \\
&\quad \forall X \in N \cup N_T, \forall a' \in N_T\}
\end{aligned}$$

$$\begin{aligned}
&\cup\{({}^{*}a^{'}) \rightarrow (a) : \forall a^{'} \in N_T\}\\
P^1_{AtoBC} =&\{({}^{+}X, A) \rightarrow ({}^{-}X_{|}, {}_{|}B_{|}{}_{|}C_{|}) :\\
&\forall X \in N_T \cup N, \forall p \in P, p = A \rightarrow BC\}\\
&\cup\{({}^{+}A) \rightarrow ({}^{-}B_{|}{}_{|}C_{|}) : \forall p \in P, p = A \rightarrow BC\}\\
P^1_{AtoB} =&\{({}^{+}X, A) \rightarrow ({}^{-}X_{|}, {}_{|}B_{|}) :\\
&\forall X \in N_T \cup N, \forall p \in P, p = A \rightarrow B\}\\
&\cup\{({}^{+}A) \rightarrow ({}^{-}B_{|}) : \forall p \in P, p = A \rightarrow B\}\\
P^1_{Atoa} =&\{({}^{+}X, A) \rightarrow ({}^{-}X_{|}, {}_{|}a^{'}_{|}) :\\
&\forall X \in N_T \cup N, \forall p \in P, p = A \rightarrow a\}\\
&\cup\{({}^{+}A) \rightarrow ({}^{-}_{|}a^{'}_{|}) : \forall p \in P, p = A \rightarrow B\}\\
P^1_{ABtoCD} =&\{({}^{+}X, A, B) \rightarrow ({}^{-}X_{|}, {}_{|}C_{<}, {}_{>}D_{|}) :\\
&\forall X \in N_T \cup N, \forall p \in P, p = AB \rightarrow CD\}\\
&\cup\{({}^{+}A, B) \rightarrow ({}^{-}C_{<}, {}_{>}D_{|}) :\\
&\forall p \in P, p = AB \rightarrow CD\}\\
P^1_{phase2} =&\{(X, B) \rightarrow (X, {}_{|}B_{|}) :\\
&\forall B \in N \cup N_T, X \in N_{CS-} \cup N_{CF-}\}
\end{aligned}$$

Last set that has to be defined is P_2. Again, this set is created as an union of several subsets:

$$\begin{aligned}
P^2_{init} =&\{({}^{-}X_{|}) \rightarrow ({}^{+}X^{\wedge}_{|}) : \forall X \in N_T \cup N\}\\
&\cup\{({}^{-}X_{<}) \rightarrow ({}^{+}X^{\wedge}_{<}) : \forall X \in N_T \cup N\}\\
P^2_{check} =&\{(A^{\wedge}_{|}, {}_{|}B_{|}) \rightarrow (A, B^{\wedge}_{|}) : \forall X, A, B \in N_T \cup N\}\\
&\cup\{(A^{\wedge}_{<}, {}_{>}B_{|}) \rightarrow (A, B^{\wedge}_{|}) : \forall X, A, B \in N_T \cup N\}\\
P^2_{checkf} =&\{({}^{+}A^{\wedge}_{|}, {}_{|}B_{|}) \rightarrow ({}^{+}A, B^{\wedge}_{|}) : \forall A, B \in N_T \cup N\}\\
&\cup\{({}^{+}A^{\wedge}_{<}, {}_{>}B_{|}) \rightarrow ({}^{+}A, B^{\wedge}_{|}) : \forall A, B \in N_T \cup N\}\\
P^2_{end} =&\{({}^{+}A^{\wedge}) \rightarrow ({}^{+}A) : \forall A \in N_T \cup N\}\\
&\cup\{({}^{+}A, B^{\wedge}_{|}) \rightarrow ({}^{+}A, B) : \forall A, B \in N_T \cup N\}\\
&\cup\{({}^{+}A^{\wedge}_{|}) \rightarrow ({}^{+}A) : \forall A \in N_T \cup N\}\\
P^2_{block} =&\{({}_{|}X_{|}) \rightarrow (!) : \forall X \in N_T \cup N\}
\end{aligned}$$

We will now briefly describe, how the resulting SCGS Γ simulates the input CSG G. The system consists of two components, both working in t mode. The computation of Γ consists of two phases. During the first one, all terminals are represented by a nonterminal variant of themselves. During the second phase, all nonterminals are rewritten to their terminal variant. The simulation itself takes place during the first phase.

The simulation in Γ of each application of one rule of G consists of two parts. Firstly, the first component applies the selected rule using the modified nonterminals contained in the sets N_{CS} and N_{CF}. Symbols of the type ${}_{|}X_{<}$ denote, that the rewriting is done in a context-sensitive way and that the remaining symbol on the right hand side of the rule should appear immediately right of the symbol. Similarly ${}_{>}X_{|}$ denotes, that the rest of the right hand side of the rule should appear immediately left of the symbol. Symbols of the form ${}_{|}X_{|}$ then represent cotext free rewriting. After the application of the rule, the first component rewrites all remaining symbols to their context-free variant and then deactivates. This is done using the rules of the set P^1_{phase2}. The fact, that only one of the rules was applied is checked using the first symbol of the sentential form. This symbol is of the form ${}^{+}X$ or ${}^{-}X$ (plus the context-sensitive and context-free versions), where the signs $+$ and $-$ indicate, whether input rule should be applied, or remaining symbols should be rewritten.

The second component then checks, whether the first component applied the rule correctly. This is done using the special $\wedge$ mark. This symbol indicates, which symbol is currently checked, we will call this symbol *current symbol*. Symbols are checked in pairs, where the first symbol of the pair is the current symbol and the second symbol is some symbol right of the first one. During this check, the special marks $(|, <, >)$ on the adjacent sides of those symbols are checked and removed and the $\wedge$ mark is moved to the other symbol of the pair. Since the first symbol of the pair is always the current symbol, the $\wedge$ moves from the left side of the sentential form to the right, with no way of returning back left. When all of the symbols are checked, the second component is deactivated and the first one simulates new rule. Since the components have scattered context rules, it is not guaranteed that adjacent symbols are always checked by the second component. Because of this, set of rules P^2_{block} is created. When some of the symbols is skipped during the checking phase, these rules will block the generation of sentence by Γ.

The final phase, which rewrites all nonterminals to terminals, is started by rewriting of the first symbol ${}^{+}X$ to ${}^{*}X$. Then for each symbol $a^{'}$, there is a rule of the form $({}^{*}X, a^{'}) \rightarrow ({}^{*}X, a)$, where a is corresponding terminal symbol. Finally, the leftmost symbol itself is rewritten to its terminal form. Since all the rules of all components always check the first symbol, after this step no further rewriting can be done and all nonterminals that remain in the sentential form cannot be removed. This phase is represented by set P^1_T.

Example 1. *Suppose CSG $G = (\{A, B, C, D, E\}, \{b, c, d, e\}, P, A)$ with rules $P = \{A \rightarrow BC, C \rightarrow CD, BD \rightarrow DB, CD \rightarrow ED, B \rightarrow b, C \rightarrow c, D \rightarrow d, E \rightarrow e\}$. Observe that there is no sentential form that could be* *generated by grammar G where the rule $BD \to DB$ could be aplied.*

Based on the described constructions, equivalent SCGS $\Gamma = (N_{GS}, T, {}^{+}A, P_1, P_2)$ can be created. We will now try to show, how would Γ simulate G. Because the amount of rules and symbols created by the transformation algorithm is quite large, we will not list elements of these sets.

The only rule of G that has starting symbol on its left hand side is $A \to BC$. Similarly, only rule applicable on BC (we will ignore rules with terminals) is rule $C \to CD$ Derivation $A \Rightarrow^{*} BCD$ would be simulated using following sequence of derivation steps:

$$\begin{aligned}
{}^{+}A &\Rightarrow {}^{-}B_{|}{}_{|}C_{|} && [({}^{+}A) \to ({}^{-}B_{|}{}_{|}C_{|}) \in P^{1}_{AtoBC}] \\
&\Rightarrow {}^{+}B^{\wedge}_{|}{}_{|}C_{|} && [({}^{-}B_{|}) \to ({}^{+}B^{\wedge}_{|}) \in P^{2}_{init}] \\
&\Rightarrow {}^{+}B\,C^{\wedge}_{|} && [({}^{+}B^{\wedge}_{|}, {}_{|}C_{|}) \to ({}^{+}B, C^{\wedge}_{|}) \in P^{2}_{checkf}] \\
&\Rightarrow {}^{+}BC && [({}^{+}B, C^{\wedge}_{|}) \to ({}^{+}B, C) \in P^{2}_{end}]
\end{aligned}$$

This way, the first rule is simulated. It is important to note that since Γ works in t mode, rules from set P_2 are all applied "together". The derivation would continue using the following rules:

$$\begin{aligned}
{}^{+}BC &\Rightarrow {}^{-}B_{|}{}_{|}C_{|}{}_{|}D_{|} && [({}^{+}B, C) \to ({}^{-}B_{|}, {}_{|}C_{|}{}_{|}D_{|}) \in P^{1}_{AtoBC}] \\
&\Rightarrow {}^{+}B^{\wedge}_{|}{}_{|}C_{|}{}_{|}D_{|} && [({}^{-}B_{|}) \to ({}^{+}B^{\wedge}_{|}) \in P^{2}_{init}] \\
&\Rightarrow {}^{+}B\,C^{\wedge}_{|}{}_{|}D_{|} && [({}^{+}B^{\wedge}_{|}, {}_{|}C_{|}) \to ({}^{+}B, C^{\wedge}_{|}) \in P^{2}_{checkf}] \\
&\Rightarrow {}^{+}BC\,D^{\wedge}_{|} && [(C^{\wedge}_{|}, {}_{|}D_{|}) \to (C, D^{\wedge}_{|}) \in P^{2}_{check}] \\
&\Rightarrow {}^{+}BCD && [({}^{+}B, D^{\wedge}_{|}) \to ({}^{+}B, D) \in P^{2}_{end}]
\end{aligned}$$

As was mentioned before, rule $BD \to DB$ can in fact never be applied by the grammar G. Suppose sentential form ${}^{+}BCD$ of the Γ. Simulation of this rule would lead to the following derivation:

$$\begin{aligned}
{}^{+}BCD &\Rightarrow {}^{-}D_{<}C_{>}B_{|} && [({}^{+}B, D) \to ({}^{-}D_{<}, {}_{>}B_{|}) \in P^{1}_{ABtoCD}] \\
&\Rightarrow {}^{-}D_{<}{}_{|}C_{|}{}_{>}B_{|} && [({}^{-}D_{<}, C) \to ({}^{-}D_{<}, {}_{|}C_{|}) \in P^{1}_{phase2}] \\
&\Rightarrow {}^{+}D^{\wedge}_{<}{}_{|}C_{|}{}_{>}B_{|} && [({}^{-}D_{<}) \to ({}^{+}D^{\wedge}_{<}) \in P^{2}_{init}] \\
&\Rightarrow {}^{+}D\,{}_{|}C_{|}\,B^{\wedge}_{|} && [({}^{+}D^{\wedge}_{<}, {}_{>}B_{|}) \to ({}^{+}D, B^{\wedge}_{|}) \in P^{2}_{checkf}] \\
&\Rightarrow {}^{+}B\,{}_{|}C_{|}D && [({}^{+}B, D^{\wedge}_{|}) \to ({}^{+}B, D) \in P^{2}_{end}] \\
&\Rightarrow {}^{+}B!D && [(B) \to (!) \in P^{2}_{block}]
\end{aligned}$$

Again, each component of Γ works in t mode. This ensures, that any symbols skipped during the checking phase, will be replaced by blocking symbols (!) before the second component of Γ deactivates.

On the other hand, rule $CD \to ED$ can be applied. The simulation of this rule works as follows:

$$\begin{aligned}
{}^{+}BCD &\Rightarrow {}^{-}B_{|}{}_{|}E_{<>}D_{|} && [({}^{+}B, C, D) \to ({}^{-}B_{|}, {}_{|}E_{<}, {}_{>}D_{|}) \in P^{1}_{ABtoCD}] \\
&\Rightarrow {}^{+}B^{\wedge}_{|}{}_{|}E_{<>}D_{|} && [({}^{-}B_{|}) \to ({}^{+}B^{\wedge}_{|}) \in P^{2}_{init}] \\
&\Rightarrow {}^{+}B\,E^{\wedge}_{<>}D_{|} && [({}^{+}B^{\wedge}_{|}, {}_{|}E_{<}) \to ({}^{+}B, E^{\wedge}_{<}) \in P^{2}_{checkf}] \\
&\Rightarrow {}^{+}BE\,D^{\wedge}_{|} && [(E^{\wedge}_{<}, {}_{>}D_{|}) \to (E, D^{\wedge}_{|}) \in P^{2}_{check}] \\
&\Rightarrow {}^{+}BED && [({}^{+}B, D^{\wedge}_{|}) \to ({}^{+}B, D) \in P^{2}_{end}]
\end{aligned}$$

We will now sketch the proof, that the algorithm described before really creates SCGS equivalent to the input CSG.

Claim 1. *In any sentential form, there is always at most one symbol marked with any of $*, -, +$.*

Proof. Observe, that no rule contains more than one symbol marked with any of the $*, -, +$ on the right hand side. Further observe that if any marked symbol does appear on the right hand side of a rule, there is also a marked symbol on the left hand side of the same rule. Thus no new marked symbols can be introduced into the sentential form. □

Claim 2. *Any derivation that generates a sentence ends with a sequence of rules of the form $p_1 p_{2_1} \dots p_{2_n} p_3$, where $p_1, p_{2_i}, p_3 \in P^1_T, 1 \le i \le n, n \ge 0$, where p_1, p_{2_i} and p_3 are from the first, second and third subset of P^1_T, respectively. No rule from the set P^1_T is applied before this sequence.*

Proof. Since the set P^1_T is the only set which contains terminals on the right hand side of the rules, it is obvious that rules of this set have to be used during any successful derivation. Since the rules in the second and third subset of P^1_T require the presence of symbol ${}^{*}X$, where $X \in N \cup N_T$ and this symbol can be introduced only by the rules of the first subset of P^1_T it is obvious that any rules from P^1_T must be applied after the application of some p_1. Similarly, some p_3 replaces the symbol ${}^{*}X$ with corresponding terminal, thus removing $*$ mark from the sentential form. Therefore, after the application of p_3 no other rule from P^1_T can be applied.

Suppose the application of sequence of rules of the form $p_1 p_{2_1} \dots p_{2_m}$. The sentential form after this application must be of the from ${}^{*}XX_1 \dots X_k$, where none of the symbols $X_i, 1 \le i \le k$ is marked with any of $+, -, *$ (claim 1). Only sets of rules, that do not require any marked symbol on the left hand side of its rules are sets P^2_{check} and P^2_{block}. Usage of any from the set P^2_{block} leads to introduction of blocking symbol and the system will thus not generate any sentence. Even though the rules from P^2_{check} do not require any of the marks $+,-,*$ in the sentential form they require the $\wedge$ mark to be present. This mark can only apear during the activation of the second component and its introduction requires symbol marked with $-$. Furthermore, the $\wedge$ mark is removed from the sentential form during the same activation of the second component (this will be shown in claim 5). Thus only the rules from P_T^1 can be applied after the application of p_1. $\square$

Since we have just shown that the rules of the subset P_T^1 must be applied right before the end of the simulation and no sooner, we ignore this subset in further proofs.

Claim 3. *The first component of Γ rewrites sentential forms of the form ${}^{+}X\alpha$ to a string of one of the following forms*

1. ${}^{-}Y_{|}\beta$
2. ${}^{-}Y_{<}\gamma$

where $X, Y \in N \cup N_T$, $\alpha \in (N \cup N_T)^*$, $\beta, \gamma \in (N_{CS} \cup N_{CF})^*$ (such that claim 1 holds) where one of the following is true:

(a) $\beta \in (N_{CF})^*$
(b) $\beta = Y_0 \ldots {}_{|}U_{<} \ldots {}_{>}V_{|} \ldots Y_n$*, where* $Y_i \in N_{CF}, 0 \leq i \leq n$ *and* ${}_{|}U_{<}, {}_{>}V_{|} \in N_{CS}$

and $\gamma = Y_0 \ldots {}_{>}V_{|} \ldots Y_n$, where $Y_i \in N_{CF}, 0 \leq i \leq n$ and ${}_{>}V_{|} \in N_{CS}$.

Proof. Suppose sentential form ${}^{+}X\alpha$ defined as above, where $\alpha = X_0 \ldots X_n$. Since there is no symbol from the alphabet $N_{cur} \cup N_{-}$ only rules of the first component can be used.

In ${}^{+}X\alpha$ one of the following derivation steps has to be applied in Γ (each derivation corresponds to one subset of the rules of the first component of Γ):

1. ${}^{+}XX_0 \ldots X_{i-1}AX_{i+1} \ldots X_n$
   $\Rightarrow {}^{-}X_{|}X_0 \ldots X_{i-1}\,{}_{|}B_{|}\,{}_{|}C_{|}X_{i+1} \ldots X_n$
2. ${}^{+}AX_0 \ldots X_n$
   $\Rightarrow {}^{-}B_{|}\,{}_{|}C_{|}X_0 \ldots X_n$
3. ${}^{+}XX_0 \ldots X_{i-1}AX_{i+1} \ldots X_n$
   $\Rightarrow {}^{-}X_{|}X_0 \ldots X_{i-1}\,{}_{|}B_{|}X_{i+1} \ldots X_n$
4. ${}^{+}AX_0 \ldots X_n$
   $\Rightarrow {}^{-}B_{|}X_0 \ldots X_n$
5. ${}^{+}XX_0 \ldots X_{i-1}AX_{i+1} \ldots X_n$
   $\Rightarrow {}^{-}X_{|}X_0 \ldots X_{i-1}\,{}_{|}a'_{|}X_{i+1} \ldots X_n$
6. ${}^{+}AX_0 \ldots X_n$
   $\Rightarrow {}^{-}_{|}a'_{|}X_0 \ldots X_n$
7. ${}^{+}XX_0 \ldots X_{i-1}A\delta BX_{i+1} \ldots X_n$
   $\Rightarrow {}^{-}X_{|}X_0 \ldots X_{i-1}\,{}_{|}C_{<}\delta\,{}_{>}D_{|}X_{i+1} \ldots X_n$
8. ${}^{+}A\delta BX_0 \ldots X_n$
   $\Rightarrow {}^{-}C_{<}\delta\,{}_{>}D_{|}X_0 \ldots X_n$

where $\delta \in (N \cup N_T)^*$. Observe that each of the generated string is in one of the following forms:

- ${}^{-}X_{|}\delta_1 B\delta_2 C\delta_3$ (1-7)
- ${}^{-}X_{<}\delta_1\,{}_{>}D_{|}\delta_2$ (8)

where $\delta_1, \delta_2, \delta_3 \in (N \cup N_T)^*$, $D \in N_{CS}$ (the second subset) and either $B, C \in N_{CF}$ or $B \in N_{CS}$ (first subset) and $C \in N_{CS}$ (the second subset).

After this first rule is applied, the sentential form contains symbol marked with $-$. Since the components of Γ work in t mode, rules of the first component have to be applied as long as there are some symbols that can be rewritten. This means that the rules from the set P^1_{phase2} have to be used now. Since the $\delta_1, \delta_2, \delta_3 \in (N \cup N_T)^*$ and left hand sides of second components of rules from set P^1_{phase2} are defined for all symbols in $N \cup N_T$. Substring $\delta_1 = Z_0 \ldots Z_n$, $Z_i \in N \cup N_T$ is rewritten to $\delta_1' = {}_{|}Z_{0|} \ldots {}_{|}Z_{n|}$, $Z_i \in N \cup N_T, 0 \leq i \leq n$. The same applies to δ_2, δ_3. Using the P^1_{phase2} we thus obtain one of the following sentential forms:

$$
\begin{gathered}
{}^{-}X_{|}\delta_1 B\delta_2 C\delta_3 \Rightarrow^* {}^{-}Y_{|}\beta \\
{}^{-}X_{<}\delta_1\,{}_{>}D_{|}\delta_2 \Rightarrow^* {}^{-}Y_{<}\gamma
\end{gathered}
$$

$\square$

Claim 4. *During its activation, the first component applies at most a single rule of the simulated CSG. This follows from claim 3 and its proof.*

Claim 5. *The second component of Γ rewrites any sentential form of the form ${}^{-}X_{|}\,{}_{|}X_{0|} \ldots {}_{|}X_{n|}$, where $X_i \in N \cup N_T \forall i : 0 \leq i \leq n$ to string of the form ${}^{+}XX_1 \ldots X_n$.*

Basic Idea 1 (Claim 5)**.** *Observe that the input string contains no symbols marked with $+$ or $\wedge$. Only some rule from the set P^2_{init} can be initially used. Furthermore, this rule must be from the first subset of this set (the first symbol of the sentential form is "context-free variant"). Application of this rule rewrites the $-$ mark of the first symbol to $+$ and introduces $\wedge$ mark to the sentential form, specifically to its first symbol. Since this rule rewrites $-$ to $+$, no rule from this set can be used during this activation of the component. From now on, there will always be exactly one symbol marked with $\wedge$ in the sentential form until the deactivation of this component. This is based on the fact, that no rule contains more than single $\wedge$ on the right hand side of the rule and furthermore, all rules that contain $\wedge$ on the right hand side also contain this mark on the* *left hand side. This means that the mark can never be "duplicated" and can only be passed from symbol to symbol or erased. Observe that this "passing" of the $\wedge$ mark can occur only from a symbol to some symbol right of it, never to the left of it. During this passing, $|$ marks are gradually removed from symbols marked with $\wedge$. Because $\wedge$ moves always to the right, together with the fact, that no new $\wedge$ mark can be introduced to the sentential form, any symbols skipped during this phase will remain in their "context-free" form. When the $\wedge$ reaches the last symbol of the sentential form, the mark is removed. Since the component works in t mode, all possible rules must be applied before the deactivation. This means that any symbol still in context-free form is rewritten to the blocking symbol by a rule from P^2_{block}.*

Claim 6. *The second component of Γ rewrites any string of the form ${}^{-}X_{<}\,{}_{|}X_{0|}\cdots{}_{|}X_{j-1|}\,{}_{>}X_{j|}\cdots{}_{|}X_{n|}$, where $X_i \in N \cup N_T, \forall i : 0 \leq i \leq n$ to string of the form ${}^{+}XX_1 \ldots X_n$ if and only if ${}_{|}X_{0|}\cdots{}_{|}X_{j-1|} = \varepsilon$, otherwise introduces blocking symbols.*

Claim 7. *The second component of Γ rewrites any string of the form ${}^{-}X_{|}\,{}_{|}X_{0|}\cdots{}_{|}X_{j<}\,{}_{|}X_{j+1|}\cdots{}_{|}X_{k-1|}\,{}_{>}X_{k|}\cdots{}_{|}X_{n|}$, where $X_i \in N \cup N_T \forall i : 0 \leq i \leq n$ to string of the form ${}^{+}XX_1 \ldots X_n$ if and only if ${}_{|}X_{j+1|}\cdots{}_{|}X_{k-1|} = \varepsilon$,otherwise introduces blocking symbols.*

Proof. Proof of claims 6 and 7 are similar to proof of claim 5 and they are left to the reader. □

Based on the previous claims, it is easy to show that each simulation of a rule of G consists of a single activation of the first component followed by a single activation of the second component of Γ. If the simulated context-sensitive rule is applied in a scattered way, blocking symbols are introduced to the sentential form, otherwise the sentential form is prepared for the simulation of a next rule. In the end, all non-blocking symbols are rewritten to terminals thus creating a sentence of the simulated language. Therefore, $L(G) = L(\Gamma)$.

Theorem 1. $\mathscr{L}(SCGS) = \mathscr{L}(CS)$

Proof. This is implied by lemmas 1 and 2. □

Theorem 2. *Any context-sensitive language can be generated by SCGS, where each scattered context rule has at most two components.*

Basic Idea 2. *Obviously, the only set that does not meet this criterion is P^1_{ABtoCD} and specifically its first subset. Rules of this subset can be simulated by introduction of some auxiliary rules and symbols. Suppose rule $({}^{+}X, A, B) \rightarrow ({}^{-}X_{|}, {}_{|}A_{<}, {}_{>}B_{|})$ and sentential form ${}^{+}XAB$. This rule can be simulated using those auxiliary rules in a following way:*

$$ {}^{+}XAB \Rightarrow {}^{\sim}_{|}X_{|}\,{}^{1}_{|}C_{<}B \Rightarrow {}^{\sim}_{|}X_{|}\,{}_{|}C_{<}\,{}^{2}_{>}D_{|} \Rightarrow {}^{-}X_{|}\,{}_{|}C_{<}\,{}_{>}D_{|}, $$

where always pairs of symbols are rewritten during each derivation step. Formal proof is left to the reader.

4. Conclusions

The modified version of $\mathscr{L}(PSCG) = \mathscr{L}(CS)$ problem was discussed in this paper. This modification deals with combination of CD grammar systems with propagating scattered context components and compares their generative power with context-sensitive grammars. The algorithm that constructs grammar system which simulates given context-sensitive grammar has been described. Based on this algorithm, it is shown that those two models have the same generative power. Furthermore it is shown that this property holds even for the most simple variant of these grammar systems — those using only two components, where each scattered context rule is of degree of at most two. While this modified version does not answer the original $\mathscr{L}(PSCG) = \mathscr{L}(CS)$ problem, it might be useful when solving it.

Acknowledgements

Authors would like to thank Zbyněk Křivka for his useful comments.

References

[1] Sheila Greibach and John Hopcroft. Scattered context grammars. *Journal of Computer and System Sciences*, 3(3):233 – 247, 1969.

[2] A. Meduna and P. Zemek. *Regulated Grammars and Automata*. Springer New York, 2014.

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

[4] J. Dassow and G. Păun. *Regulated rewriting in formal language theory*. EATCS monographs on theoretical computer science. Springer, 1989.

[5] Erzsebet Csuhaj-Varju, Josef Kelemen, Gheorghe Paun, and Jurgen Dassow. Grammar systems: A grammatical approach to distribution and cooperation. *Inc., Newark, NJ*, 1994.