SCATTERED CONTEXT GRAMMARS GENERATING DERIVATION TREES

Stanislav Židek
Doctoral Degree Programme (2), FIT BUT
E-mail: xzidek05@stud.fit.vutbr.cz

Supervised by: Alexander Meduna
E-mail: meduna@fit.vutbr.cz

Abstract: Propagating scattered context grammars are used to generate the strings corresponding to the derivation trees. It is proved that for every language defined by scattered context grammar, there exists a propagating scattered context grammar whose language consists of the strings representing derivation trees of the original language.

Keywords: propagating scattered context grammar, derivation tree, left-bracketted representation

1 INTRODUCTION

Scattered context grammars are one of the most intuitive yet very powerful types of paralell grammars. They deserve our attention for example as a suitable model of parallel compilation, which is being intensively studied last years (see [1]).

In this paper, we use their propagating version, which contains no erasing productions, to generate the strings representing the derivation trees of languages characterized by scattered context grammars. We demonstrate that for every scattered context grammar G, there exists a propagating scattered context grammar that generates the strings representing the derivation trees of $L(G)$ according to the grammar G. This characterization is of some interest, because the family of languages generated by *propagating* scattered context grammars is *properly included* in the family of languages generated by scattered context grammars.

In Section 2, we state the preliminaries and define the key notions of our article. In Section 3, we present our result, i.e. the algorithm taking the scattered context grammar and constructing the propagating scattered context grammar that generates the strings corresponding to derivation trees. Furthermore, the proof of the algorithm correctness is given. In Section 4, we make some final notes and suggestions regarding the future investigation.

2 PRELIMINARIES

We assume a reader is familiar with formal language theory (for further reference, see [2]).

A scattered context grammar (SCG, see [3], [4]) is a quadruple, $G = (V,T,P,S)$, where V is a total alphabet, $T \subset V$ is a finite set of terminal symbols (terminals), $S \in V - T$ is the starting symbol and P is a finite set of productions of the form $(A_1,\ldots,A_n) \rightarrow (x_1,\ldots,x_n)$, where $A_i \in V - T$ and $x_i \in V^*$ for all $i : 1 \leq i \leq n$.

A propagating SCG is a SCG $G = (V,T,P,S)$ in which every $(A_1,\ldots,A_n) \rightarrow (x_1,\ldots,x_n) \in P$ satisfies $x_i \in V^+$ for all $i : 1 \leq i \leq n$.

Let $G = (V,T,P,S)$ be a (propagating) SCG, $y = u_1A_1u_2\ldots u_nA_nu_{n+1}$, $z = u_1x_1u_2\ldots u_nx_nu_{n+1}$, $y,z \in V^*$, $p = (A_1,\ldots,A_n) \rightarrow (x_1,\ldots,x_n) \in P$. Then y directly derives z in SCG G according to the pro-

duction p, $y \Rightarrow_G z$ $[p]$ (or simply $y \Rightarrow_G z$). Let $\Rightarrow_G^+$ and $\Rightarrow_G^*$ denote transitive and reflexive-transitive closure of $\Rightarrow_G$, respectively.

Let $G = (V, T, P, S)$ be a (propagating) SCG. Language generated by G is denoted by $L(G)$ and defined as $L(G) = \{w : w \in T^*, S \Rightarrow_G^* w\}$.

Usually it is very obvious which grammar we are talking about, so we will sometimes abbreviate $\Rightarrow_G$ to $\Rightarrow$.

We also assume a reader is familiar with graph theory. By *tree*, we will mean an *labeled ordered tree*. Let Υ be a tree, Θ be a set of nodes in Υ, $\theta \in \Theta$, n be a nonnegative integer. Then $\mathrm{root}(\Upsilon)$ denotes the root node of the tree, $\mathrm{child}(\theta)$ denotes an n-tuple of node's child nodes (zero-tuple for leaf nodes), $\mathrm{lab}(\theta)$ is a label of the node θ. Sometimes we will generalize the notion of lab to tuples – $\mathrm{lab}((\theta_1, \ldots, \theta_n)) = (\mathrm{lab}(\theta_1), \ldots, \mathrm{lab}(\theta_n))$

Let $G = (V, T, P, S)$ be a SCG and $p = (A_1, \ldots, A_i, \ldots, A_n) \to (x_1, \ldots, x_i, \ldots, x_n) \in P$, $x_i = a_1 \ldots a_m$, $m \geq 0$, $a_j \in V$ for all $j : 0 \leq j \leq m$. Production tree of the i-th component of the production p, denoted by $\mathrm{pt}(p, i)$, is a labeled elementary tree Υ such that $\mathrm{lab}(\mathrm{root}(\Upsilon)) = A_i$ and

$$\mathrm{lab}(\mathrm{child}(\mathrm{root}(\Upsilon))) = \begin{cases} \lambda & \text{if } m = 0 \\ (a_1, \ldots, a_m) & \text{otherwise} \end{cases}$$

Let us have a SCG $G = (V, T, P, S)$ and a derivation $S \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_m$. The *derivation tree* corresponding to this derivation, denoted by $S \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_m [\![\Upsilon]\!]$, is a labeled tree Υ constructed as follows:

1. Create a root node, $\mathrm{lab}(\mathrm{root}(\Upsilon)) = S$.
2. Set $j = 0$.
3. Repeat until $j = m$:
   (a) Let $w_j = u_1 A_1 \ldots A_n u_{n+1}$, $w_{j+1} = u_1 x_1 \ldots x_n u_{n+1}$, $A_i \in V - T$, $x_i \in V^*$ for all $i : 1 \leq i \leq n$, $p = (A_1, \ldots, A_n) \to (x_1, \ldots, x_n) \in P$, $\theta_1, \ldots \theta_n$ be a leaf nodes of Υ in this order (considering inorder tree traversal), $\mathrm{lab}(\theta_i) = A_i$. Add child nodes to θ_1 through θ_n so that it holds that subtree rooted at θ_j is a $\mathrm{pt}(p, i)$.
   (b) Increment j.

Let $\mathrm{dt}(G)$ denote a set of all derivation trees corresponding to successful derivations of grammar G.

A left-bracketed representation (see [5]) of a derivation tree Υ, denoted by $\mathrm{lbr}(\Upsilon)$, is defined recursively as follows:

1. If $\mathrm{root}(\Upsilon) = ()$, then $\mathrm{lbr}(\Upsilon) = \mathrm{lab}(\mathrm{root}(\Upsilon))$.
2. If $\mathrm{child}(\mathrm{root}(\Upsilon)) = (\theta_1, \ldots, \theta_n)$, then $\mathrm{lbr}(T) = \mathrm{lab}(\mathrm{root}(\Upsilon))\langle \mathrm{lbr}(\Upsilon_1) \ldots \mathrm{lbr}(\Upsilon_n) \rangle$, where Υ_i is a subtree rooted at θ_i for all i: $1 \leq i \leq n$.

Definition 2.1. Let $G = (V, T, P, S)$ be a SCG. Then $\triangle_G$ denotes a set of *tree SCGs* corresponding to SCG G, that is

$$\triangle_G = \left\{ G_\triangle : L(G_\triangle) = \{\mathrm{lbr}(\Upsilon) : S \Rightarrow_G^* w \, [\![\Upsilon]\!], \, w \in T^*\} \right\}.$$

3 RESULTS

In this section, we present our algorithm.

Algorithm 3.1. Construction of tree SCG $G_\triangle \in \triangle_G$
Input: a SCG $G = (V,T,P,S)$
Output: a *propagating* SCG $G_\triangle = (V_\triangle, T_\triangle, P_\triangle, S_\triangle) \in \triangle_G$
Method:

1. Let $T_\triangle = V \cup \{\langle, \rangle, \lambda\}$, $N_\triangle = \{A_\triangle : A \in (V-T)\}$, $N_\triangle \cap (V-T) = \emptyset$, and $V_\triangle = T_\triangle \cup N_\triangle$.
2. Let h be a coding $h : V^* \to (T \cup (V_\triangle - T_\triangle))^*$ such that $h(a) = a$ for every $a \in T$ and $h(A) = A_\triangle$ for every $A \in (V-T)$.
   Let g be a function $g : V^* \to (T \cup (V_\triangle - T_\triangle) \cup \{\lambda\})^*$ such that:
$$g(x) = \begin{cases} \lambda & \text{if } x = \varepsilon \\ h(x) & \text{otherwise} \end{cases}$$
3. Initially set $P_\triangle = \emptyset$.
4. For each production $(A_1,\ldots,A_n) \to (x_1,\ldots,x_n) \in P$ add $(A_{1_\triangle},\ldots,A_{n_\triangle}) \to (A_1\langle g(x_1)\rangle,\ldots,A_n\langle g(x_n)\rangle)$ to $P_\triangle$.

Lemma 3.1. $L(G_\triangle) = \mathrm{dt}(G)$

Proof idea We will show that $G_\triangle$ simulates construction of proper derivation tree of G. Whenever a production of G would be applied to a sentential form, $G_\triangle$ adds child nodes to the leaves and marks former leaves as inner nodes.

Formal proof

Proof. Let us have a successful derivation $S = w_0 \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_m$, $w_m \in T^*$, in SCG G. Then there exists a derivation $S_\triangle = t_0 \Rightarrow t_1 \Rightarrow \ldots \Rightarrow t_m$, $t_m \in T_\triangle$, in $G_\triangle$, such that t_i represents an unfinished derivation tree corresponding to the sequence of derivations $S = w_0 \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_i$.

Definition 3.2. Let us have a derivation tree Υ, with set of nodes Θ. Auxilliary derivation tree to a derivation tree Υ of grammar G, denoted by $\Upsilon^\triangle$, is a derivation tree isomorphic to Υ under isomorphism f, such that

$$\mathrm{lab}(f(\theta)) = \begin{cases} A_\triangle & \text{if } \mathrm{lab}(\theta) = A, A \in V-T \wedge \mathrm{child}(\theta) = () \\ \mathrm{lab}(\theta) & \text{otherwise} \end{cases}$$

Put simply, the labelling of nodes of Υ differs only in case when some leaf node is labelled by nonterminal of grammar G – if the leaf is labeled by a nonterminal A in Υ, then it is assigned a nonterminal $A_\triangle$ in $\Upsilon^\triangle$ instead.

Claim 3.3.
$$S \Rightarrow_G^m w\ [\![\Upsilon]\!], w \in V^*, \textit{ if and only if } S_\triangle \Rightarrow_{G_\triangle}^m \mathrm{lbr}(\Upsilon^\triangle)$$

Equivalence estabilished in Claim 3.3 can be proven in two steps, one for every direction:

Only if ($\Rightarrow$): This implication can be proven by an induction on the length m of derivations.

- Let $m = 0$. Then $S \Rightarrow_G^0 S\ [\![\Upsilon]\!]$, and $S_\triangle \Rightarrow_{G_\triangle}^0 S_\triangle$. Clearly $S_\triangle = \mathrm{lbr}(\Upsilon^\triangle)$, because the only node of Υ is the leaf, so it is labeled by $S_\triangle$ instead of S.
- Let us suppose that the implication holds for all derivations of length at most $m : m \geq 0$ and consider a derivation of length $m+1$. By induction hypothesis, if $S \Rightarrow_G^m y\ [\![\Upsilon]\!]$, then $S_\triangle \Rightarrow_{G_\triangle}^m s = \mathrm{lbr}(\Upsilon^\triangle)$.

  We have to prove that if $S \Rightarrow_G^m y \Rightarrow_G z\ [\![\Upsilon']\!]$, then $S_\triangle \Rightarrow_{G_\triangle}^m s \Rightarrow_{G_\triangle} t = \mathrm{lbr}(\Upsilon'^\triangle)$. If $y \in T^*$, then the derivation step $y \Rightarrow_G z\ [p]$ is not possible, so the implication holds. Otherwise, $y = u_1A_1u_2 \ldots u_nA_nu_{n+1}$, $z = u_1x_1u_2 \ldots u_nx_nu_{n+1}$, $p = (A_1,\ldots,A_n) \to (x_1,\ldots,x_n) \in P$, $s = \mathrm{lbr}(\Upsilon^\triangle) = v_1A_{1_\triangle}v_2 \ldots v_nA_{n_\triangle}n_{n+1}$.

  According to the algorithm 3.1 there exists a production $p_\triangle \in P_\triangle$, $p_\triangle = (A_{1_\triangle},\ldots,A_{n_\triangle}) \to (A_1\langle r_1\rangle,\ldots,A_n\langle r_n\rangle)$, $r_i \in (T \cup (V_\triangle - T_\triangle))^* \cup \{\lambda\}$. It directly follows that there exists a derivation $s \Rightarrow_{G_\triangle} t\ [p_\triangle]$, $t = v_1A_1\langle r_1\rangle v_2 \ldots v_nA_n\langle r_n\rangle n_{n+1}$. We see that this derivation step exactly simmulates one step of construction of auxilliary derivation tree, because for all $i : 1 \leq i \leq n$, $A_i\langle r_i\rangle = \mathrm{lbr}(\Upsilon_i^\triangle)$, $\Upsilon_i = \mathrm{pt}(p_\triangle, i)$. So it immediately follows that $t = \mathrm{lbr}(\Upsilon'^\triangle)$.

If ($\Leftarrow$): This part can be proven by a very similar induction.

- Let $m = 0$. Then $S \Rightarrow_G^0 S\ [\![\Upsilon]\!]$ and $S_\triangle \Rightarrow_{G_\triangle}^0 S_\triangle$. Obviously $S_\triangle = \mathrm{lbr}(\Upsilon^\triangle)$.
- Let us suppose that the implication holds for all derivations of length at most $m : m \geq 0$, i.e. $S \Rightarrow_G^m y\ [\![\Upsilon]\!]$ and $S_\triangle \Rightarrow_{G_\triangle}^m s = \mathrm{lbr}(\Upsilon^\triangle)$ (induction hypothesis). We have to prove that if $S_\triangle \Rightarrow_{G_\triangle}^m s \Rightarrow_{G_\triangle} t = \mathrm{lbr}(\Upsilon'^\triangle)$, then $S \Rightarrow_G^m y \Rightarrow_G z\ [\![\Upsilon']\!]$.

  If $s \in T_\triangle^*$, then derivation step $s \Rightarrow_{G_\triangle} t\ [p_\triangle]$ is not possible, otherwise $s = \mathrm{lbr}(\Upsilon^\triangle) = v_1A_{1_\triangle}v_2 \ldots v_nA_{n_\triangle}n_{n+1}$, $t = v_1A_1\langle r_1\rangle v_2 \ldots v_nA_n\langle r_n\rangle n_{n+1}$, $p_\triangle = (A_{1_\triangle},\ldots,A_{n_\triangle}) \to (A_1\langle r_1\rangle,\ldots,A_n\langle r_n\rangle)$, $r_i \in (T \cup (V_\triangle - T_\triangle))^* \cup \{\lambda\}$, $v_i \in V_\triangle^*$, $A_{1_\triangle} \in V_\triangle - T_\triangle$.

  According to the algorithm 3.1 there exists a production $p = (A_1,\ldots,A_n) \to (x_1,\ldots,x_n) \in P$. There must exist a derivation $y \Rightarrow_G z\ [p]$, where $y = u_1A_1u_2 \ldots u_nA_nu_{n+1}$, $z = u_1x_1u_2 \ldots u_nx_nu_{n+1}$. This derivation is precisely the derivation step simmulated by applying the production $p_\triangle$, since for all $i : 1 \leq i \leq n$, $A_i\langle r_i\rangle = \mathrm{lbr}(\Upsilon_i^\triangle) = \mathrm{lbr}(\mathrm{pt}(p_\triangle, i))$ and rewriting $A_{i_\triangle}$ to $A_i\langle r_i\rangle$ exactly follows the definition of constructing the auxiliary derivation tree – nodes that become nonleaf change their labelling from $A_{i_\triangle}$ to A_i.

We see Claim 3.3 holds.

Claim 3.4. *Let $S \Rightarrow_G^m w\ [\![\Upsilon]\!]$. Then $w \in T^*$ if and only if* $\mathrm{lbr}(\Upsilon) = \mathrm{lbr}(\Upsilon^\triangle)$.

Proof of Claim 3.4 is very straightforward.

Only if ($\Leftarrow$): Proof by contradiction. Assume the implication is not true, that is $w = a_1 \ldots a_n \in T \wedge \mathrm{lbr}(\Upsilon) \neq \mathrm{lbr}(\Upsilon^\triangle)$, $n \geq 0$. Since both trees are isomorphic (under isomorphism f), difference must be in labelling function, so there exists a node θ such that $\mathrm{lab}(\theta) \neq \mathrm{lab}(f(\theta))$. By definition 3.2, this is possible only if θ is the leaf labeled by nonterminal, therefore $|w| = n > 0$. Then there must exist $i : 1 \leq i \leq n$, such that $a_i \in V - T$. But this is a contradiction, because $w \in T^*$. (Note: If $w = \varepsilon$, then for every leaf θ, $\mathrm{lab}(\theta) = \lambda = \mathrm{lab}(f(\theta))$ – also a contradiction.)

If ($\Leftarrow$): Proof by contradiction. Assume the implication is not true, that is $\mathrm{lbr}(\Upsilon) = \mathrm{lbr}(\Upsilon^{\triangle}) \wedge w = a_1 \ldots a_n \notin T^*$, $n \geq 0$. Let Θ be a set of nodes of Υ and f be a isomorphism according to definition 3.2. For every leaf node $\theta \in \Theta$, $\mathrm{lab}(\theta) \in T \cup \{\lambda\}$, otherwise $\mathrm{lab}(\theta) \neq \mathrm{lab}(f(\theta))$ and $\mathrm{lbr}(\Upsilon) \neq \mathrm{lbr}(\Upsilon^{\triangle})$. If $\mathrm{lab}(\theta) \in T$ for every leaf node θ, then for every $i : 1 \leq i \leq n$, $a_i \in T \cup \{\lambda\}$, that is $w \in T^*$. Either way, we get a contradiction.

□

Theorem 3.5. *For every SCG G, there exists a* propagating *SCG* $G_{\triangle} \in \triangle_G$.

Proof. Since the Algorithm 3.1 is correct (follows from Lemma 3.1) and it always terminates (due to the finiteness of the set of productions of the original grammar), it directly follows that this theorem holds. □

4 CONCLUSIONS

We showed that for every SCG G, there exists a propagating SCG generating derivation trees of G. This observation is of some interest, because we are able to characterize the family of languages generated by SCGs by propagating SCGs, which are strictly weaker generation mechanisms. Also, the derivation tree represents potentially very valuable information, particularly in terms of compilation.

In the future, we would like to focus on the case of generating not only the derivation tree, but also the sentence of the original language. Furthermore, we want to find the practical applications, possibly in the compiler based on (the subset of) scattered context grammars. In this case, the derivation tree could be useful for example in the phase right after the parsing of the input sentence, such as interpretation or generation of three-address code. As we can see, further research of this topic is neccessary.

ACKNOWLEDGMENT

This work was supported by the Research Plan No. MSM, 0021630528 – Security-Oriented Research in Information Technology.

REFERENCES

[1] Gao, G.R., Pollock, L., Cavazos, J., Li, X. (Eds.): Languages and Compilers for Parallel Computing. Berlin, DE, Springer 2009, ISBN 3-642-13373-8

[2] Meduna, A.: Automata and Languages: Theory and Applications, London, GB, Springer 2005, ISBN 1-85233-074-0

[3] Greibach, S., Hopcroft, J.: Scattered Context Grammars. Journal of Computer and System Sciences, Volume 3, Number 3, 2004, pp. 233–247

[4] Meduna, A., Techet, J.: Scattered Context Grammars and their Applications, Ashurst, GB, WIT Press 2010, ISBN 1845644263

[5] Aho, A.V., Ullman, J.D.: The Theory of Parsing, Translation, and Compiling. Upper Saddle River, US, Prentice Hall 1972, ISBN 0-13-914556-7