Scattered Context Generators of Sentences With Their Parses

Scattered Context Grammar (*SCG*)

Scattered context grammar G = (V, P, S, T)

V is a finite alphabet

T is a set of terminals, $T \subset V$

S is a starting symbol, $S \in (V - T)$

P is a finite set of productions of the form: $(A_1, \dots, A_n) \rightarrow (x_1, \dots, x_n)$; $A_1, \dots, A_n \in (V - T)$; $x_1, \dots, x_n \in V^*$

Propagating scattered context grammar (*PSCG*)

- special case of *SCG*
- every $(A_1, \dots, A_n) \rightarrow (x_1, \dots, x_n)$ satisfies $x_1, \dots, x_n \in V^+$

Derivation step

Derivation step

if $(A_1, \ldots, A_n) \rightarrow (x_1, \ldots, x_n) \in P$

$u = u_1 A_1 \ldots u_n A_n u_{n+1}$

$v = u_1 x_1 \ldots u_n x_n u_{n+1}$

then $u \Rightarrow v \, [(A_1, \ldots, A_n) \rightarrow (x_1, \ldots, x_n)]$

- $alph(w)$ denotes the set of all symbols occuring in w

Example

$alph(bacaab) = \{a, b, c\}$

Leftmost derivation step

every $A_i \notin alph(u_i)$ for all $1 \leq i \leq n$

Generated language

Generated language

$L(G) = \{x \mid x \in T^*, S \Rightarrow^* x\}$

- if every step in every generation of $x \in T^*$ is leftmost, then G generates $L(G)$ in a leftmost way

Generative power

- $\mathcal{L}_{SCG} = \mathcal{L}_{RE}$
- $\mathcal{L}_{CF} \subset \mathcal{L}_{PSCG} \subseteq \mathcal{L}_{CS}$

Production Labels

- for every grammar, G, there is a set of production labels
- we denote them $lab(G)$
- every $p \in lab(G)$ uniquely identifies one production
- we write $p : (A_1, \ldots, A_n) \rightarrow (x_1, \ldots, x_n)$

Example

$G_1 = (\{S, A, B, C, a, b, c\}, P_1, S, \{a, b, c\})$
$lab(G_1) = \{1, 2, 3\}$
$P_1 = \{\ 1 : (S) \rightarrow (ABC),$
$2 : (A, B, C) \rightarrow (aA, bB, cC),$
$3 : (A, B, C) \rightarrow (\epsilon, \epsilon, \epsilon)\}$

$L(G_1) = \{a^n b^n c^n \mid n \geq 0\}$
G_1 generates $L(G_1)$ in a leftmost way

Production Labels (cont.)

- to express that $x \Rightarrow y$ by $p : (A_1, \ldots, A_n) \rightarrow (x_1, \ldots, x_n)$, we write $x \Rightarrow y\,[p]$

Example

$S \Rightarrow ABC\,[1] \Rightarrow aAbBcC\,[2] \Rightarrow aaAbbBccC\,[2] \Rightarrow aabbcc\,[3]$ in G_1

- to express that $x \Rightarrow^* y$ by productions labeled with $p_1, \ldots, p_n$, we write $x \Rightarrow^* y\,[p_1 \ldots p_n]$
- $p_1 \ldots p_n \in lab(G)^*$

Example

$S \Rightarrow^* aabbcc\,[1223]$ in G_1

$1223 \in lab(G_1)^*$

Proper Generator of its Sentences with Their Parses

Parse

If $S \Rightarrow^* x\,[\rho], x \in T^*, \rho \in lab(G)^*$, then x is a sentence generated by G according to *parse* ρ

Example

aabbcc is a sentence generated according to parse 1223 in G_1

Proper generator of its sentences with their parses

- G is a proper generator of its sentences with their parses if $L(G) = \{x \,|\, x = y\rho, y \in (T - lab(G))^*, \rho \in lab(G)^*, S \Rightarrow^* x\,[\rho]\}$
- if G generates $L(G)$ in a leftmost way, G is a proper leftmost generator of its sentences with their parses

Proper Generator of its Sentences with Their Parses (cont.)

Example

$G_2 = (\{S, A, B, C, a, b, c, 1, 2, 3, 4, \$\}, P_2, S, \{a, b, c, 1, 2, 3, 4\})$

$lab(G_2) = \{1, 2, 3, 4\}$

$$
\begin{aligned}
P_2 = \{\ & 1 : (S) \rightarrow (ABC1\$), \\
& 2 : (A, B, C, \$) \rightarrow (AA, BB, CC, 2\$), \\
& 3 : (A, B, C, \$) \rightarrow (a, b, c, 3\$), \\
& 4 : (A, B, C, \$) \rightarrow (\epsilon, \epsilon, \epsilon, 4)\}
\end{aligned}
$$

$$
\begin{aligned}
S \Rightarrow\ & ABC1\$\,[1] \Rightarrow AABBCC12\$\,[2] \Rightarrow AabBCc123\$\,[3] \Rightarrow \\
& AAabBBCCc1232\$\,[2] \Rightarrow aAabBbcCc12323\$\,[3] \Rightarrow \\
& aabbcc123234\$\,[4]
\end{aligned}
$$

$S \Rightarrow^* aabbcc123234\,[123234]$

$L(G_2) = \{a^n b^n c^n \rho \mid n \geq 0, S \Rightarrow^* a^n b^n c^n \rho\,[\rho]\}$

G_2 is a proper generator of its sentences with their parses

G_2 is not a proper leftmost generator of its sentences with their parses

Main Result

- let $G = (V, P, S, T)$ be a proper generator of its sentences with their parses
- we define the weak identity π from V^* to $(V - lab(G))^*$ as
  - $\pi(a) = a$ for every $a \in (V - lab(G))$
  - $\pi(p) = \epsilon$ for every $p \in lab(G)$

Theorem

For every recursively enumerable language, L, there exists a PSCG, G, such that G is a proper generator of its sentences with their parses and $L = \pi(L(G))$.

Theorem

For every recursively enumerable language, L, there exists a PSCG, G, such that G contains no more than six nonterminals, G is a proper leftmost generator of its sentences with their parses and $L = \pi(L(G))$.

Queue Grammar (QG)

- we represent the recursively enumerable language by a queue grammar

Queue grammar G = (V, T, W, F, s, P)

V is a finite alphabet of symbols

T is a set of terminals, $T \subset V$

W is a finite alphabet of states

s is a starting configuration, $s \in (V - T)(W - F)$

F is a set of final states, $F \subseteq W$

P is a finite set of productions of the form: (a, b, x, c)

- a $\in V$
- b $\in (W - F)$
- x $\in V^*$
- c $\in W$

Derivation Step

Derivation step

if $u = arb$, $v = rxc$, $a \in V$, $r, x \in V^*$, $b, c \in W$, and $(a, b, x, c) \in P$, then

$$u \Rightarrow v\,[(a, b, x, c)]$$

Generated language

$L(G) = \{w \mid s \Rightarrow^* wf, w \in T^*, f \in F\}$

Example

$G = (V, T, W, F, s, P), \{(a, 1, bFc, 2), (B, 2, AA, 2)\} \subseteq P$, then

$aBB1 \Rightarrow BBbFc2\,[(a, 1, bFc, 2)] \Rightarrow BbFcAA2\,[(B, 2, AA, 2)] \Rightarrow bFcAAAA2\,[(B, 2, AA, 2)]$ in G

Generative Power

Generative power

$\mathcal{L}_{QG} = \mathcal{L}_{RE}$

- for every queue grammar there exists an equivalent queue grammar which first generates only words from $(V - T)^*$, and then only words from T^+

Proof Sketch

Basic idea

1. represent the recursively enumerable language by a *QG*
2. initiate the derivation
3. simulate *QG* by *PSCG*
    1. simulate generation of words from $(V - T)^*$
    2. simulate generation of words from T^+
4. check if the simulation was correct
5. complete the derivation

- every production has to add its label to the sentential form to create the parse in the correct order
- generated sentence must precede this parse

Proof

- $Q = (V, T, W, F, s, R)$, $L(Q) = L$
- α: injection from $lab(Q)$ to $\{\bar{0}\}^*\{\bar{1}\}$
- β: injection from T to $\{0\}^*\{1\}$
- $f(a) = \{\alpha(r) \,|\, r : (a, b, x, c) \in R\}$ for all $a \in V$
- $g(b) = \{\alpha(r) \,|\, r : (a, b, x, c) \in R\}$ for all $b \in W$

Constructed *PSCG*

$G = (\{S, A, B, \#, \bar{0}, \bar{1}\}, P, S, \{0, 1\} \cup lab(G))$

- the construction of P and $lab(G)$ is demonstrated on the following slides

Productions

Step 1 (initialization)

For every $\bar{a}_0 \in f(a_0)$, $\bar{q}_0 \in g(q_0)$ such that $s = a_0 q_0$, add

$\lfloor 1\bar{a}_0\bar{q}_0 \rfloor : (S) \rightarrow (A\lfloor 1\bar{a}_0\bar{q}_0 \rfloor AA\bar{q}_0 A\bar{a}_0 AB)$ to P

Step 2 (simulation of Q's productions generating words over V-T)

For every $r : (a, b, c_1 \ldots c_n, d) \in R$, $c_1, \ldots, c_n \in (V - T)$ for some $n \geq 0$ and $d \in (W - F)$, $\bar{c}_1 \in f(c_1), \ldots, \bar{c}_n \in f(c_n)$, $\bar{d} \in g(d)$, add

$$\begin{aligned} \lfloor 2r\bar{c}_1 \ldots \bar{c}_n\bar{d} \rfloor &: (A, A, A, A, A, B) \rightarrow \\ &: (A, \lfloor 2r\bar{c}_1 \ldots \bar{c}_n\bar{d} \rfloor A, \alpha(r)A, \bar{d}A, \bar{c}_1 \ldots \bar{c}_n A, B) \text{ to } P \end{aligned}$$

Step 3 (separation between steps 2 and 4)

Add $\lfloor 3 \rfloor : (A, A, A, A, A, B) \rightarrow (A, \lfloor 3 \rfloor A, A, A, B, A)$ to P

Productions (cont.)

Step 4 (simulation of Q's productions generating words over T)

For every $r : (a, b, c_1 \dots c_n, d) \in R$, $c_1, \dots, c_n \in T$ for some $n \geq 0$ and $d \in (W - F)$, $\bar{d} \in g(d)$, add

$\lfloor 4r\bar{d} \rfloor : (A, A, A, A, B, A) \rightarrow (\beta(c_1) \dots \beta(c_n)A, \lfloor 4r\bar{d} \rfloor A, \alpha(r)A, \bar{d}A, B, A)$ to P

Step 5 (simulation of Q's final step)

For every $r : (a, b, c_1 \dots c_n, d) \in R$, $c_1, \dots, c_n \in T$ for some $n \geq 0$ and $d \in F$, add

$\lfloor 5r \rfloor : (A, A, A, A, B, A) \rightarrow (\beta(c_1) \dots \beta(c_n), \lfloor 5r \rfloor A, \alpha(r)A, A, B, AA)$ to P

Productions (cont.)

Step 6 (simulation verification)

Add

$\lfloor 6 \rfloor : (A, \bar{0}, A, \bar{0}, A, \bar{0}, B, A, A) \rightarrow (\lfloor 6 \rfloor, A, \#, A, \#, A, B, A, A)$, and
$\lfloor 7 \rfloor : (A, \bar{1}, A, \bar{1}, A, \bar{1}, B, A, A) \rightarrow (\lfloor 7 \rfloor, A, \#, A, \#, A, B, A, A)$ to P;

Step 7 (finishing the derivation)

Add

$\lfloor 8 \rfloor : (A, A, A, B, A, A) \rightarrow (\lfloor 8 \rfloor B, \#, \#, \#, \#, \#)$,
$\lfloor 9 \rfloor : (B, \#) \rightarrow (\lfloor 9 \rfloor, B)$, and
$\lfloor 10 \rfloor : (B) \rightarrow (\lfloor 10 \rfloor)$ to P.

Future Investigation

- which other grammars can be used as proper generators of their sentences with their parses?
  - grammar systems seem to be appropriate candidates
- is it possible to generate sentences together with other useful information (e.g. derivation trees)?