Информационные процессы, Том 10, № 3, 2010, стр. 292–302.

Эквивалентные преобразования контекстно-свободных грамматик

С.Ю.Соловьев

МГУ имени М.В.Ломоносова, факультет ВМК, Москва, Россия
Поступила в редколлегию 25.09.2010

Аннотация—В работе описывается и обосновывается эквивалентное преобразование контекстно-свободных грамматик, позволяющее удалять общие префиксы и суффиксы в терминальных реализациях нетерминальных символов. Кроме того, в работе предлагается универсальный метод совместного применения нескольких эквивалентных преобразований грамматик.

1. ВВЕДЕНИЕ

В теории формальных языков известно большое количество эквивалентных преобразований контекстно-свободных грамматик (КС-грамматик). Помимо прочего это означает, что один и тот же КС-язык может порождаться весьма непохожими грамматиками. В настоящей работе эквивалентные преобразования грамматик рассматриваются с точки зрения удаления конструкций "несущественных" для порождаемого языка.

Контекстно-свободной грамматикой [1] называется четверка G $=$ $<$ N, Σ, P, S $>$, где
N – алфавит нетерминальных символов (нетерминалов);
Σ – непересекающийся с N алфавит терминальных символов (терминалов);
P – конечное множество правил вывода вида A$\rightarrow$ α, где A $\in$ N, α - цепочка символов из N $\cup$ Σ;
S – выделенный символ из N, именуемый начальным символом.

В последующих выкладках будем полагать, что действуют следующие соглашения:

- A, B, C, D – нетерминальные символы; S - начальный символ;
- a, b, c, d – терминальные символы;
- $\alpha, \beta, \gamma, \sigma$ – цепочки символов из $N \cup \Sigma$;
- x, y, z – цепочки символов (предложения) из Σ;
- e – пустая цепочка нулевой длины;
- запись A $\rightarrow$ α_1 | ... | α_n, означает множество правил { A$\rightarrow$ α_1, ..., A$\rightarrow$ α_n };
- запись α $=>_G$ β означает, что $\alpha = \alpha_1 A \alpha_2$, $\beta = \alpha_1 \gamma \alpha_2$ и $A \rightarrow \gamma \in P$; в этом случае будем говорить, что цепочка β непосредственно выводима из цепочки α в грамматике G;
- $L(G) = \{\, x \mid S \Rightarrow^*_G x \,\}$ – язык, порождаемый грамматикой G.

Принятые соглашения позволяют, в частности, задавать КС-грамматики простым перечислением правил вывода.

2. КС#ГРАММАТИКИ

Каждая контекстно-свободная грамматика $G =< N, \Sigma, P, S >$ порождает семейство грамматик $G(A) =< N, \Sigma, P, A >$, где $A \in N$. Для начального символа S имеем: $G(S) = G$ и $L(G(S)) = L(G)$. В общем случае КС-язык $L(G(A))$ есть реализация нетерминала A в классе терминальных цепочек.

В дальнейшем изложении будем рассматривать только такие КС-грамматики $< N, \Sigma, P, S >$, в которых:

- отсутствуют правила с пустой правой частью; и
- имеется правило $S \to \#$, причем терминальный символ $\#$ в других правилах не встречается.

Для КС-грамматик, удовлетворяющих перечисленным свойствам, будем использовать обозначение КС#грамматики.

С точки зрения порождаемых языков приведенные ограничения не являются существенными. Любой КС-язык L может быть получен из КС#языка $L_\#$ одним из двух способов: либо $L = L_\# \backslash \{\#\}$, либо $L = (L_\# \backslash \{\#\}) \cup \{e\}$; соответствующая КС-грамматика получается либо посредством удаления правила $S \to \#$, либо посредством его замены на правило $S \to e$. Вместе с тем, использование дополнительного символа $\#$ позволяет естественным образом вывести начальный символ S из-под действия многих эквивалентных преобразований.

Остановимся на одном из преобразований КС-грамматик, связанным с изменением языков L(G(A)), A $\neq$ S. В качестве неформально введения в задачу рассмотрим цепочки abcg, abcfg и abdcdg. Эти цепочки имеют общий префикс ab и общий суффикс g. В более изощренных случаях необходимо точно оговорить, что считать общим префиксом и/или суффиксом:
– для { a, ab, abb } префикс и суффикс отсутствуют;
– для { abc, abcc, abccc } префикс = ab, суффикс отсутствует;
– для { abc, abbc, abbbc } префикс = ab, суффикс отсутствует и пр.

В общем случае, после удаления общих префикса и суффикса не должна возникать пустая цепочка. Кроме того, будем исходить из того, что сначала определяется префикс, а затем – суффикс.

Если язык L(G(A)) имеет непустой общий префикс x,
то будем говорить, что нетерминал A имеет префикс x.

Если язык L(G(A)) имеет непустой общий суффикс z,
то будем говорить, что нетерминал A имеет суффикс z.

Продолжая неформальное введение, рассмотрим КС-грамматику

$$G: \quad \begin{array}{lll} S & \to & A + A \;\;|\;\; gb \\ A & \to & abcg \;\;|\;\; abcfg \;\;|\;\; abddg \end{array}$$

Здесь язык $L(G(A)) = \{abcg, abcfg, abdcdg\}$ задан в явном виде. У нетерминала A префикс ab и суффикс g можно изъять и передать “вышестоящему” нетерминалу S, при этом грамматика примет следующий вид

$$G': \quad \begin{array}{lll} S & \to & abA'g + abA'g \;\;|\;\; gb \\ A' & \to & c \;\;|\;\; cf \;\;|\;\; dd \end{array}$$

В грамматике G' нетерминал A' не имеет ни префикса ни суффикса. Понятно, что грамматики G и G' эквивалентны. Возникает вопрос *о возможности изъятия префиксов и суффиксов нетермналов для КС-грамматик общего вида.* Отметим, что в данном случае изъятие рассматривается как эквивалентное преобразование грамматик.

Нетерминалы, у которых можно забирать префиксы и суффиксы, удовлетворяют естественным условиям:

$$L(G(A)) = \text{конкатенация}(a, L) \qquad (L)$$

и/или $$L(G(A)) = \text{конкатенация}(L, a), \qquad (R)$$

где L – некоторый язык, $e \notin L$.

Однако приведенные условия являются слишком общими; можно показать алгоритмическую неразрешимость задачи изъятия префиксов и суффиксов у всех нетерминалов, удовлетворяющих условиям (L) и (R). Пойдем по пути уточнения класса нетерминалов, у которых можно "безнаказанно" забирать префиксы и суффиксы.

3. LD-ПРЕОБРАЗОВАНИЕ КС#ГРАММАТИК

Подмножество нетерминалов КС#грамматики G = < N, Σ, P, S >
будем называть делимым-слева относительно терминала a, и
будем его обозначать LD(a),
если множество правил вывода P образуют правила трех типов:

$$\begin{array}{ll} \text{тип 1: } A \to a\alpha, & \text{где } A \in LD(a) \text{ и } \alpha \neq e; \\ \text{тип 2: } A \to A_0\alpha, & \text{где } A \in LD(a) \text{ и } A_0 \in LD(a) \\ \text{тип 3: } B \to \alpha, & \text{где } B \notin LD(a) \end{array}$$

Понятно, что
если $A \in LD(a)$, то $L(G(A))$ обязательно удовлетворяет условию (L);
если $B \notin LD(a)$, то $L(G(B))$ может удовлетворять или не удовлетворять условию (L).
Например:

$$\begin{array}{lccrcl} G_T: & S & \to & A+A & | & Bg \\ & A & \to & ab & | & Aa \\ & B & \to & Dd & | & abd \\ & D & \to & a & | & aD \end{array}$$

∀ x ∈ { a, b, d, g, + }   D ∉ LD(x), и, следовательно, B ∉ LD(x).
LD(a) = { A },
правила типа 1: A → ab;
правила типа 2: A → Aa;
правила типа 3: S → A+A, S → Bg, B → Dd, B → abd, D → a, D → aD.
$A \in LD(a)$;   L(G_T(A)) = { aba^n | n ≥ 0 } = { ax | x = ba^n, n≥ 0 } .
$B \notin LD(a)$;   L(G_T(B)) = { a^nd | n ≥ 1 } ∪ { abd } = { ax | x = a^nd / bd, n≥ 2 }.
Предложения из $L(G_T(D))$ не сводимы к виду ax, где $x \neq e$.
Конец примера.

Каждому делимому-слева подмножеству нетерминалов $LD(a) = \{A_1, A_2, \ldots\}$ поставим во взаимнооднозначное соответствие подмножество ранее не использовавшихся нетерминальных символов $LD'(a) = \{A'_1, A'_2, \ldots\}2$.

Если в контексте некоторой КС-грамматики известно множество LD(a), то можно рассматривать преобразование цепочек W, заключающееся в выполнении всевозможных подстановок aA'_i *вместо* A_i. Цепочка W(α) получается из цепочки α посредством замены всех символов A из LD(a) на цепочки из двух символов aA', где A' ∈ LD'(a). Обратное преобразование W^{-1} заменяет в заданной цепочке все вхождения aA'_i на соответствующие нетерминалы A_i. Обратное преобразование полностью удаляет из заданной цепочки α все символы LD'(a) только в том случае, когда непосредственно перед символом из LD'(a) располагается символ a.

Например, в грамматике G_T для LD(a) = { A } имеем:

$$\begin{array}{l} W(Aa + A + aBg) = aA'a + aA' + aBg, \\ \qquad\qquad W^{-1}(\,aA'a + bA' + aBg) = Aa + bA' + aBg, \\ \qquad\qquad W^{-1}(\,aA'a + aA') = Aa + A \end{array}$$

Пусть LD(a) – подмножество делимых-слева нетерминалов некоторой КС#грамматики G = < N, Σ, P, S >, определим грамматику G_W = < N_W, Σ, P_W, S > следующим образом:
$N_W = (LD'(a) \cup N) \backslash LD(a)$, а
P_W построено так:

$$\begin{array}{lll} \text{если } A \to a\alpha & \text{есть правило типа 1 грамматики } G, & \text{то } A' \to W(\alpha) \in P_W; \\ \text{если } A \to A_0\alpha & \text{есть правило типа 2 грамматики } G, & \text{то } A' \to A'_0 W(\alpha) \in P_W; \\ \text{если } B \to \alpha & \text{есть правило типа 3 грамматики } G, & \text{то } B \to W(\alpha) \in P_W. \end{array}$$

Например, для множества LD(a) = { A } грамматики

$$\begin{array}{lllllll} G: & S & \to & A & | & A+A & | & \# \\ & A & \to & ab & | & A+A & | & A-A \end{array}$$

имеем:

$$\begin{array}{lllllll} G_W: & S & \to & aA' & | & aA'+aA' & | & \# \\ & A' & \to & b & | & A'+aA' & | & A'-aA' \end{array}$$

Заметим, что преобразование грамматики G в грамматику G_W (LD-преобразование) возможно только в том случае, когда в G найдется хотя бы одно непустое множество нетерминалов LD(a), более того, LD(a) является существенным аргументом LD-преобразования.

Утверждение 1. *L(G)* $\subseteq$ *L(G_W) для КС#грамматики G = < N, Σ, P, S >.*

Доказательство. Рассмотрим произвольное предложение x из L(G) и некоторый левый вывод x в грамматике G.

$$S = \sigma_0 =>_G \sigma_1 =>_G ... =>_G \sigma_{k-1} =>_G \sigma_k = x.$$

Докажем по индукции, что последовательность цепочек $W(\sigma_0), ..., W(\sigma_k)$ есть левый вывод предложения x в грамматике G_W.

Из-за наличия правила $S \to \#$ основной символ S не может входить ни в одно множество делимых-слева нетерминалов. Поэтому $W(\sigma_0) = W(S) = S$, то есть $W(\sigma_0)$ – цепочка, выводимая в G_W.

Предположим, что для некоторого $i, i < k$, установлено, что

$$S = W(\sigma_0) =>_{G_W} W(\sigma_1) =>_{G_W} ... =>_{G_W} W(\sigma_i)$$

Покажем, что в этом случае $W(\sigma_i) =>_{G_W} W(\sigma_{i+1})$.

Рассмотрим i+1-й этап левого вывода предложения x в грамматике G: $\sigma_i =>_G \sigma_{i+1}$. По определению левого вывода:

$$\sigma_i = zC\gamma =>_G z\beta\gamma = \sigma_{i+1} \text{ причем } C \to \beta \text{ есть правило грамматики } G.$$

Из-за наличия в грамматике G множества делимых-слева нетерминалов LD(a) правило $C \to \beta$ может быть:

▷ *либо* типа 1: $A \to a\alpha$ и тогда

$$\begin{aligned} W(\sigma_i) &= W(zA\gamma) = zaA'W(\gamma), \\ W(\sigma_{i+1}) &= W(za\alpha\gamma) = zaW(\alpha)W(\gamma), \end{aligned}$$

то есть $zaA'W(\gamma) =>_{G_W} zaW(\alpha)W(\gamma)$ посредством правила $A' \to W(\alpha)$ из P_W;

$\triangleright$ *либо* типа 2: $A \to A_0\alpha$ и тогда

$$\begin{aligned} W(\sigma_i) \quad &= W(zA\gamma) \quad = zaA'W(\gamma), \\ W(\sigma_{i+1}) &= W(zA_0\alpha\gamma) = zaA'_0W(\alpha)W(\gamma), \end{aligned}$$

то есть $zaA'W(\gamma) =>_{G_W} zaA'_0W(\alpha)W(\gamma)$ посредством правила $A' \to A'_0W(\alpha)$ из P_W;

$\triangleright$ *либо* типа 3: $B \to \alpha$ и тогда

$$\begin{aligned} W(\sigma_i) \quad &= W(zB\gamma) = zBW(\gamma), \\ W(\sigma_{i+1}) &= W(z\alpha\gamma) = zW(\alpha)W(\gamma), \end{aligned}$$

то есть $zBW(\gamma) =>_{G_W} W(\alpha)W(\gamma)$ посредством правила $B \to W(\alpha)$ из P_W.

Во всех трех случаях $W(\sigma_i) =>_{G_W} W(\sigma_{i+1})$ и, следовательно, предложение $x = W(\sigma_k)$ выводимо в грамматике G_W, а значит $L(G) \subseteq L(G_W)$. Утверждение 1 доказано.

Утверждение 2. *$L(G) \supseteq L(G_W)$ для КС#грамматики G.*

Доказательство проведем индукцией по длине левого вывода некоторого произвольного предложения x из $L(G_W)$.

$$S = \sigma_0 =>_{G_W} \sigma_1 =>_{G_W} ... =>_{G_W} \sigma_{k-1} =>_{G_W} \sigma_k = x$$

Покажем, что:

1и) в цепочках $\sigma_0, \sigma_1, \ldots, \sigma_{k-1}, \sigma_k$ перед каждым символом из N'_A размещается символ a; и
2и) последовательность цепочек $W^{-1}(\sigma_0), W^{-1}(\sigma_1), \ldots, W^{-1}(\sigma_{k-1}), W^{-1}(\sigma_k)$ есть левый вывод предложения x.

Цепочка $W^{-1}(\sigma_0)$ состоит из единственного символа S и поэтому она удовлетворяет условиям 1и) и 2и).

Предположим, что для некоторого $i, i < k$, установлено, что:

1п) в цепочках $\sigma_0, \sigma_1, \ldots, \sigma_i$ перед каждым символом из N'_A размещается символ a; и
2п) последовательность цепочек $S = W^{-1}(\sigma_0) =>_G W^{-1}(\sigma_1) =>_G \ldots =>_G W^{-1}(\sigma_i)$ есть левый вывод предложения $W^{-1}(\sigma_i)$.

Рассмотрим переход $\sigma_i =>_{G_W} \sigma_{i+1}$ в левом выводе предложения x. По определению левого вывода $\sigma_i = zC\gamma, \quad \sigma_{i+1} = z\beta\gamma$ и $C \to \beta$ есть правило грамматики G_W. Отметим два обстоятельства.

Во-первых, в цепочке γ перед каждым символом из N'(a) размещается символ a. Это следует из того, что

– в $zC\gamma$ по индуктивному предположению 1п) перед каждым символом из N'_A размещается символ a; и
– в $zC\gamma$ цепочка γ располагается непосредственно за нетерминальным символом C и поэтому не может начинаться символом из $N'(a)$.

Во-вторых, по построению грамматики G_W правило $C \to \beta$ может быть

$\triangleright$ *либо* $A' \to \alpha$, если оно получено из правила первого типа $A \to aW^{-1}(\alpha)$ из G, и тогда $\sigma_i = zC\gamma = zA'\gamma = yaA'\gamma$ на основании индуктивного предположения 1п),

$$\begin{aligned} W^{-1}(\sigma_i) \quad &= W^{-1}(yaA'\gamma) = W^{-1}(y)W^{-1}(aA')W^{-1}(\gamma) = yAW^{-1}(\gamma), \\ W^{-1}(\sigma_{+1}) &= W^{-1}(ya\alpha\gamma) \ = W^{-1}(y)W^{-1}(a\alpha)W^{-1}(\gamma) \ = yaW^{-1}(\alpha)W^{-1}(\gamma), \end{aligned}$$

то есть $yAW^{-1}(\gamma) =>_G yaW^{-1}(\alpha)W^{-1}(\gamma)$ посредством правила $A \to aW^{-1}(\alpha)$;

▷ *либо* $A' \to A'_0\alpha$, если оно получено из правила второго типа $A \to A_0W^{-1}(\alpha)$ из G, и тогда $\sigma_i = zC\gamma = zA'\gamma = yaA'\gamma$ на основании индуктивного предположения 1п),

$$\begin{aligned} W^{-1}(\sigma_i) &= W^{-1}(yaA'\gamma) &&= W^{-1}(y)W^{-1}(aA')W^{-1}(\gamma) &&= yAW^{-1}(\gamma), \\ W^{-1}(\sigma_{+1}) &= W^{-1}(yaA'_0\alpha\gamma) &&= W^{-1}(y)W^{-1}(aA'_0\alpha)W^{-1}(\gamma) &&= yA_0W^{-1}(\alpha)W^{-1}(\gamma), \end{aligned}$$

то есть $yAW^{-1}(\gamma) =>_G yA_0W^{-1}(\alpha)W^{-1}(\gamma)$ посредством правила $A \to A_0W^{-1}(\alpha)$;

▷ *либо* $B \to \alpha$, если оно получено из правила третьего типа $B \to W^{-1}(\alpha)$ из G, и тогда $\sigma_i = zC\gamma = zB\gamma$,

$$\begin{aligned} W^{-1}(\sigma_i) &= W^{-1}(yB\gamma) = W^{-1}(y)W^{-1}(B)W^{-1}(\gamma) = yBW^{-1}(\gamma), \\ W^{-1}(\sigma_{i+1}) &= W^{-1}(y\alpha\gamma) = W^{-1}(y)W^{-1}(\alpha)W^{-1}(\gamma) = yW^{-1}(\alpha)W^{-1}(\gamma), \end{aligned}$$

то есть $yBW^{-1}(\gamma) =>_G yW^{-1}(\alpha)W^{-1}(\gamma)$ посредством правила $B \to W^{-1}(\alpha)$.

Во всех трех случаях $W(\sigma_i) =>_G W(\sigma_{i+1})$ и, следовательно, предложение $x = W(\sigma_k)$ выводимо в грамматике G, а значит $L(G) \supseteq L(G_W)$. Утверждение 2 доказано.

Окончательно имеем следующее

Утверждение 3. $L(G) = L(G_W)$ *для КС#грамматики* G.

Другими словами, LD-преобразование грамматик является эквивалентным преобразованием.

Если КС#грамматика G одновременно является LL(1)-грамматикой [1], то в LD-множества могут попасть только простые[1] нетерминалы. Отсюда следует, что LD-преобразование сохраняет класс LL(1)-грамматик.

4. RD-ПРЕОБРАЗОВАНИЕ КС#ГРАММАТИК

Аналогично LD-преобразованию вводится RD-преобразование КС#грамматик, основанное на множестве RD-нетерминалов делимых-справа.

Подмножество нетерминалов КС#грамматики G = < N, Σ, P, S >
будем называть делимым-справа относительно терминала a, и
будем его обозначать RD(a),
если множество правил вывода P образуют правила трех типов:

$$\begin{array}{lll} \text{тип } 1'\text{:} & A \to \alpha a, & \text{где } A \in RD(a) \text{ и } \alpha \neq e; \\ \text{тип } 2'\text{:} & A \to \alpha A_0, & \text{где } A \in RD(a) \text{ и } A_0 \in RD(a) \\ \text{тип } 3'\text{:} & B \to \alpha, & \text{где } B \notin RD(a) \end{array}$$

Если в КС-грамматике зафиксировано некоторое подмножество делимых-справа нетерминалов $RD(a)$, то в такой грамматике можно рассматривать преобразование цепочек V. Цепочка $V(\alpha)$ получается из цепочки α посредством замены всех символов A из $RD(a)$ на цепочки из двух символов $A'a$, где $A' \in RD'(a)$.

Пусть RD(a) – подмножество делимых-справа нетерминальных символов КС#грамматики $G =< N, \Sigma, P, S >$, определим грамматику $G_V =< N_V, \Sigma, P_V, S >$ следующим образом:

[1] Простым называется нетерминал A, для которого в грамматике имеется ровно одно правило $A \to \alpha$ (A-правило).

$N_V = (RD'(a) \cup N) \backslash RD(a)$, а
P_V построено так:

$$\begin{array}{lll} \text{если } A \to \alpha a & \text{есть правило типа } 1' \text{ грамматики } G, & \text{то } A' \to V(\alpha) \in P_V; \\ \text{если } A \to \alpha A_0 & \text{есть правило типа } 2' \text{ грамматики } G, & \text{то } A' \to V(\alpha) A'_0 \in P_V; \\ \text{если } B \to \alpha & \text{есть правило типа } 3' \text{ грамматики } G, & \text{то } B \to V(\alpha) \in P_V. \end{array}$$

Можно показать, что $L(G) = L(G_V)$, то есть RD-преобразование КС#грамматики G в КС#грамматику G_V относительно некоторого непустого множества RD-нетерминалов является эквивалентным преобразованием.

5. РЕАЛИЗАЦИЯ ЭКВИВАЛЕНТНЫХ ПРЕОБРАЗОВАНИЙ КС-ГРАММАТИК

Разнообразие преобразований КС-грамматик, а также необходимость их многократного повторения порождает задачу совместного использования эквивалентных преобразований. Как организовать процесс трансформации заданной грамматики с тем, чтобы результирующая грамматика уже не допускала ни одного заданного преобразования? Сложность состоит в том, что одно преобразование может удалять из грамматики А-конструкции и пополнять грамматику Б-конструкциями, а другое преобразование может поступать ровно наоборот.

Прежде всего, отметим, что с алгоритмической точки зрения каждое преобразование можно представить в виде, приведенном на рис. 1.

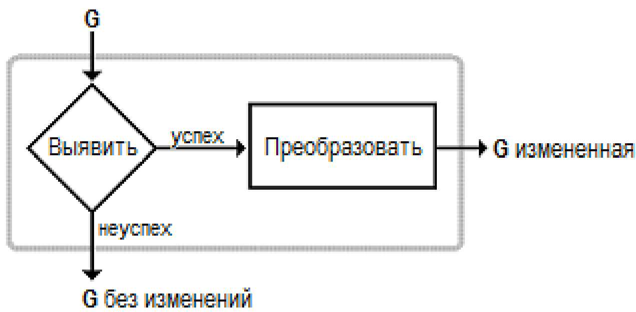

Рис. 1. Схема преобразования КС-грамматики

Например, в LD-преобразовании

– этап “Выявить” состоит в нахождении некоторого непустого множества LD(a);
– этап “Преобразовать” состоит в построении грамматики G_W относительно найденного множества LD(a).

Фактически по успешной ветке может передаваться некоторая информация, существенная для преобразования.

Не вдаваясь в подробности этапов, будем изображать преобразование в виде прямоугольника со сглаженными углами (рис. 2), в который:

– сверху входит стрелка, соответствующая исходной грамматике;
– направо выходит стрелка, соответствующая измененной грамматике; и
– вниз выходит стрелка, соответствующая исходной грамматике, оставшейся без изменений.

Рис. 2. LD-преобразование

С использованием принятых обозначений универсальная схема управления преобразованиями грамматик выглядит так как изображено на рис.3.

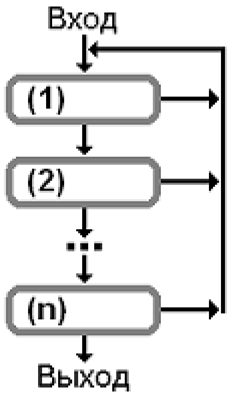

Рис. 3. Универсальная схема преобразования КС-грамматик;
(1), (2), ... (n) – конкретные преобразования КС-грамматик.

Нетрудно видеть, что универсальная схема

- полностью определяется последовательностью эквивалентных преобразований;
- имеет определенные достоинства; и
- порождает некоторые проблемы.

Достоинства. Универсальная схема гарантирует, что в результирующей грамматике более нельзя выполнить ни одного эквивалентного преобразования (1), (2), ... (n).

Проблемы. Для каждого набора преобразований необходимо доказывать корректность универсальной схемы, то есть необходимо доказывать конечность последовательности преобразований. В отдельных, но важных случаях на доказательстве корректности можно "сэкономить". В этих случаях конечность процесса эквивалентных преобразований основывается на том, что каждое преобразование уменьшает численную характеристику h исходной грамматики [2].

По определению величина h КС-грамматики $G = < N, \Sigma, P, S >$ есть

$$h(G) = \sum_{A \in N} min\{\text{ длина}(x) \mid x \in L(G(A))\}$$

Например, для рассмотренной ранее грамматики G_T имеем:
$min\{$ длина$(x) \mid x \in L(G_T(D))\} = 1;$
$min\{$ длина$(x) \mid x \in L(G_T(B))\} = 2;$
$min\{$ длина$(x) \mid x \in L(G_T(A))\} = 2;$
$min\{$ длина$(x) \mid x \in L(G_T(S))\} = 3;$
и окончательно имеем: $h(G) = 1 + 2 + 2 + 3 = 8$.

Зачастую по результатам эквивалентного преобразования

1 одна часть нетерминалов сохраняет свои реализации неизменными, в том числе, начальный символ S, а
2 другая (непустая) часть нетерминалов:
  2.1 либо вообще ликвидируется,
  2.2 либо изменяет свою реализацию с L(G(A)) на L, где
    $L = \{ y \mid xy \in L(G(A)\}$ или
    $L = \{ y \mid yz \in L(G(A)\}$ для некоторых непустых x и z.

Преобразование, удовлетворяющее свойствам 1–2, уменьшает величину h. Факт уменьшения величины h будем обозначать h-.

Например, LD-преобразование КС#грамматик относительно некоторого LD(a) подпадает под случай 1–2.2 при x = a, z = e. Здесь первую часть нетерминалов составляют $N\backslash\ LD(a)$, а вторую – $LD(a)$. Нетрудно показать, что $h(G) = h(G_W) - R$, где R – количество нетерминалов в LD(a).

Рассмотрим подвид универсальных схем эквивалентных преобразований, представленный на рис.4. Здесь:

– преобразование (1) – необязательное устранение бесполезных[2] символов, в том числе:
  – – нетерминалов, которые не могут порождать терминальные цепочки; и
  – – правил вывода, содержащих недостижимые символы;
– преобразование (2) – необязательное устранение цепных[3] правил;
– преобразования (3)..(n) – такие преобразования, которые уменьшают характеристику h.

Преобразования (1) и (2) давно и хорошо изучены, в общем случае они не изменяют характеристику грамматики h, однако их использование совместно или порознь не способно привести к зацикливанию. Что касается остальных преобразований, то их выполнение порождает монотонно убывающую последовательность положительных чисел $h_1, h_2, h_3, \ldots$. Такая последовательность не может быть бесконечной, а значит корректность подвида универсальных схем эквивалентных преобразований установлена.

К преобразованиям (3)..(n), в частности, относятся:

– устранение простых нетерминалов - случай 1+2.1 (h-);
– устранение нерекурсивных[4] нетерминалов - случай 1+2.1 (h-);
– устранение избыточных нетерминалов [2] - случай 1+2.1 (h-);

[2] Бесполезным [1] в грамматике $G =< N, \Sigma, P, S >$ называется символ $X \in N \cup \Sigma$, для которого в грамматике нет вывода вида $S => *_G\, yXz => *_G\, yxz$.
[3] Цепным [1] называется правило вывода вида $A \rightarrow B$.
[4] Нерекурсивным называется нетерминал A, для которого не существует выводов вида A $=>^+ \alpha$Aβ.

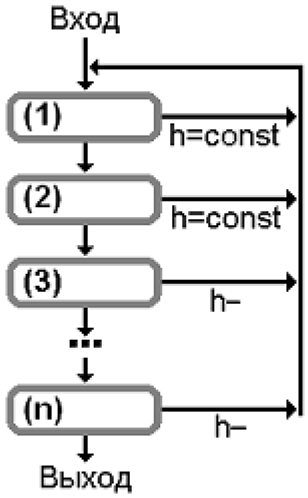

Рис. 4. Частный случай эквивалентных преобразований

– ЛНФ- и ПНФ-преобразования[5] - случай 1+2.2 (h-);
– LD- и RD- преобразования - случай 1+2.2 (h-).

Особого разговора заслуживает порядок размещения эквивалентных преобразований в универсальной схеме. С точки зрения свойств конечного результата порядок не играет роли, однако иногда частные преобразования имеет смысл выполнять раньше общих преобразований.

Так, в некоторых случаях ЛНФ-преобразования [2] можно рассматривать как частный случай LD-преобразований. Если, например A-правила грамматики имеют вид

$$\mathrm{A} \to \mathrm{aB} \mid \mathrm{aCc} \mid \mathrm{aBAd},$$

то ЛНФ-преобразование терминала A совпадает с LD-преобразованием относительно LD(a) = { A }. Вместе с тем ЛНФ-преобразование способно обрабатывать случаи. когда все правые части A-правил начинаются одним и тем же нетерминалом. При этом ЛНФ-преобразование действует вполне "разумно", преобразуя

$A \to Bb \mid BCc \mid BaAd$ в $A' \to b \mid Cc \mid aBA'd.$

LD-преобразование в этом случае действует более "топорно", преобразуя

$A \to Bb \mid BCc \mid BaAd$ в $A' \to B'b \mid B'Cc \mid B'abA'd.$

6. ЗАКЛЮЧЕНИЕ

Настоящая работа носит исключительно теоретический характер, ее главная цель – расширить спектр возможностей при выдвижении гипотез о строении неизвестной грамматики, породившей некоторые известные предложения. Если, например, известно, что LL(1) грамматика породила два предложения abcabdd и abcbcddd, то с определенными оговорками можно считать, что оба эти предложения в искомой грамматике имеют общую сентенциальную форму abcA"dd. В самом деле:

[5] ЛНФ-преобразование (ПНФ-преобразование) [2] заключается в устранении явно указанных общих префиксов (суффиксов) в правых частях нетерминалов.

- быть LL(1) означает наличие общей формы abcA, где A→ abdd | bcddd, в общем виде известной как “дерево суффиксов” [3];
- доказанная допустимость RD-преобразований фактически означает наличие общей формы abcA”dd, где A”→ ab | bcd.

Упомянутые оговорки будут раскрыты в следующих работах.

СПИСОК ЛИТЕРАТУРЫ

1. Ахо А., Ульман Дж. *Теория синтаксического анализа, перевода и компиляции.* М.: Мир, 1978, тт. 1,2.
2. Соловьев С.Ю. Нормализация контекстно-свободных грамматик для целей грамматического вывода. *XII национальная конференция по искусственному интеллекту с международным участием КИИ-2010. Труды конференции.* М: Физматлит, 2010, том 1, стр.218-224. *http : //www.park.glossary.ru/serios/read_09.php*
3. Смит Б. *Методы и алгоритмы вычислений на строках.*. М: Вильямс, 2006.