Глава 1. Элементы теории формальных языков и грамматик

1.6. Эквивалентные преобразования грамматик

При построении грамматик часто возникает необходимость в их эквивалентных преобразованиях для того, чтобы они удовлетворяли определенным критериям, но при этом не изменялся порождаемый язык. Рассмотрим некоторые простые, но наиболее часто применяемые и важные приемы преобразований КС-грамматик: удаление бесполезных символов, замена вхождений, устранение леворекурсивных продукций и цикла, факторизация, удаление ε- и цепных продукций.

1.6.1. Удаление бесполезных символов

Нетерминал X называется *производящим* (*продуктивным*), если $X \overset{*}{\Rightarrow} w$, $w \in V_T^*$, т. е. из нетерминала X можно вывести какую-нибудь терминальную строку. Нетерминал называется *непроизводящим* (*бесплодным*), если он не порождает ни одной терминальной строки. Очевидно, что если все символы правой части продукции производящие, то производящим является и нетерминал в левой части. На этом свойстве основана процедура выявления непроизводящих нетерминалов:

1. Составить список нетерминалов, для которых существует хотя бы одна продукция, правая часть которой не содержит нетерминалов.

2. Если найдена такая продукция, что все нетерминалы, стоящие в ее правой части, уже занесены в список, то добавить в список нетерминал из левой части.

Если на шаге 2 список больше не пополняется, то получен список всех производящих нетерминалов, а все не попавшие в него нетерминалы – непроизводящие.

Формально данное преобразование можно представить алгоритмом 1.1, который исходную КС-грамматику $G = (V_T, V_N, P, S)$ преобразует в эквивалентную $G' = (V_T, V'_N, P', S)$, не содержащую непроизводящих символов. Сначала определяется множество V'_N производящих нетерминалов путем рекурсивного построения множеств производящих нетерминалов $N_0, N_1, N_2,\ldots$ Затем в множество P' продукций включаются только те продукции из P, которые содержат символы из $V'_N \cup V_T$. Логическая переменная b служит для реализации выхода из цикла после завершения вычисления множества производящих нетерминалов.

Алгоритм 1.1. Удаление непроизводящих символов

Вход: КС-грамматика $G = (V_T, V_N, P, S)$

Выход: $G' = (V_T, V'_N, P', S)$ – эквивалентная грамматика, не содержащая непроизводящих символов

$N_0 := \varnothing;\ i := 1;\ b := \mathbf{true}$

$\mathbf{while}\ b\ \mathbf{do}\ \begin{cases} N_i := N_{i-1} \cup \{A \mid A \to \alpha \in P,\ \alpha \in (N_{i-1} \cup V_T)^*\} \\ \mathbf{if}\ \ N_i \ne N_{i-1}\ \ \mathbf{then}\ i := i+1\ \mathbf{else}\ b := \mathbf{false} \end{cases}$

$V'_N := N_i$ $\quad // V'_N$ – *множество производящих нетерминалов*

$P' := \{A \to \alpha \mid A \to \alpha \in P,\ A \in V'_N,\ \alpha \in (V'_N \cup V_T)^*\}$

Символ грамматики $X \in V_T \cup V_N$ (терминал или нетерминал) называется *достижимым*, если существует вывод $S \overset{*}{\Rightarrow} \alpha X \beta$ для некоторых $\alpha, \beta \in (V_T \cup V_N)^*$, т. е. символ появляется хотя бы в одной сентенциальной форме грамматики. В противном случае символ грамматики называется *недостижимым*. Очевидно, что если нетерминал левой части продукции является достижимым, то и все символы правой части достижимы. На этом свойстве основана процедура выявления недостижимых символов, которую можно представить следующим образом:

1. Образовать одноэлементный список, состоящий из начального символа грамматики.

2. Если найдена продукция, левая часть которой уже имеется в списке, то включить в список все символы, содержащиеся в ее правой части.

Если на шаге 2 список не пополняется новыми символами, то получен список всех достижимых символов, а символы, не попавшие в список, являются недостижимыми.

Формально данное преобразование можно представить алгоритмом 1.2, который исходную КС-грамматику $G = (V_T, V_N, P, S)$ преобразует в эквивалентную $G' = (V'_T, V'_N, P', S)$, не содержащую недостижимых символов. Сначала определяется множество V_i достижимых символов путем рекурсивного построения множеств $V_0, V_1, V_2,\ldots$ Затем в множество P' продукций включаются только те продукции из P, которые содержат символы из V_i. Логическая переменная b служит для реализации выхода из цикла после завершения вычисления множества достижимых символов.

Алгоритм 1.2. Удаление недостижимых символов
Вход: КС-грамматика $G = (V_T, V_N, P, S)$
Выход: $G' = (V'_T, V'_N, P', S)$ – эквивалентная грамматика, не содержащая недостижимых символов

$V_0 := \{S\};\ i := 1;\ b := \mathbf{true}$

$$\mathbf{while}\ b\ \mathbf{do}\ \begin{cases} V_i := V_{i-1} \cup \{X \mid X \in V_T \cup V_N,\ \ A \to \alpha X \beta \in P, \\ \qquad\qquad A \in V_{i-1},\ \ \alpha, \beta \in (V_T \cup V_N)^*\} \\ \mathbf{if}\ \ V_i \ne V_{i-1}\ \ \mathbf{then}\ i := i+1\ \ \mathbf{else}\ b := \mathbf{false} \end{cases}$$

// *построено множество* V_i *достижимых символов*
$V'_N := V_i \cap V_N$ // V'_N – *множество достижимых нетерминалов*
$V'_T := V_i \cap V_T$ // V'_T – *множество достижимых терминалов*
$P' := \{A \to \alpha \mid A \to \alpha \in P,\ \ A \in V_i,\ \ \alpha \in V_i^*\}$

Символы, которые являются непроизводящими или недостижимыми, называются *бесполезными*. Бесполезные символы грамматики исключаются из соответствующих множеств, и продукции, содержащие эти символы, удаляются. Исключение выполняется в следующем порядке:

1. Удалить из грамматики $G = (V_T, V_N, P, S)$ непроизводящие символы и получить грамматику $G' = (V_T, V'_N, P', S)$.

2. Удалить из грамматики $G' = (V_T, V'_N, P', S)$ недостижимые символы и получить грамматику $G'' = (V''_T, V''_N, P'', S)$.

Рассмотрим грамматику $G = (\{a, b, c\}, \{S, A, B, C\}, P, S)$ с продукциями

$S \to aSb \mid cAc \mid a \qquad B \to aa$

$A \to ABa \mid cAb \qquad C \to ac$

Непроизводящим является нетерминал A, после его исключения из V_N и удаления продукций $S \to cAc$ и $A \to ABa \mid cAb$ получится грамматика $G' = (\{a, b, c\}, \{S, B, C\}, P', S)$ с продукциями

$S \to aSb \mid a$

$B \to aa$

$C \to ac$

Недостижимыми являются символы B, C и c. После их исключения получим грамматику $G'' = (\{a, b\}, \{S\}, P'', S)$ с множеством продукций $S \to aSb \mid a$.

Грамматику, не содержащую бесполезные символы, будем называть *приведенной*.

Следует обратить внимание на то, что действия по удалению бесполезных символов должны выполняться именно в указанном порядке. В противном случае результатом не всегда будет приведенная грамматика.

Проиллюстрируем это на рассмотренной выше грамматике $G = (\{a, b, c\}, \{S, A, B, C\}, P, S)$ с продукциями

$S \to aSb \mid cAc \mid a \qquad B \to aa$

$A \to ABa \mid cAb \qquad C \to ac$

Удалим недостижимый нетерминал C. В результате получим грамматику $G' = (\{a, b, c\}, \{S, A, B\}, P', S)$ с продукциями

$S \to aSb \mid cAc \mid a$

$A \to ABa \mid cAb$

$B \to aa$

Удалим непроизводящий нетерминал A. Полученная грамматика $G'' = (\{a, b, c\}, \{S, B\}, P'', S)$ с продукциями

$S \to aSb \mid a$

$B \to aa$

не является приведенной, поскольку нетерминал B и терминал c в результате удаления нетерминала A стали недостижимыми.

В дальнейшем будем рассматривать только приведенные КС-грамматики.

1.6.2. Замена вхождений

Данное преобразование позволяет сократить число нетерминалов в грамматике и состоит в том, что если левая часть продукции входит в правую часть продукции R, то замена данного вхождения приведет просто к замене продукции R другой продукцией. Если такую замену выполнить для всех продукций с данным нетерминалом в левой части, то этот нетерминал можно исключить из грамматики. Исключение – случай, когда правая часть продукции содержит нетерминал левой части, например $S \to aSb \mid a$. Тогда удаление нетерминала из грамматики невозможно.

Формально данное преобразование можно представить следующим образом. Если $A \to \alpha_1 B \alpha_2$ – продукция грамматики и $B \to \beta_1 \mid \beta_2 \mid \ldots \mid \beta_k$ – все B-продукции этой грамматики, тогда продукции вида $A \to \alpha_1 B \alpha_2$ можно поставить в соответствие продукции вида $A \to \alpha_1 \beta_1 \alpha_2 \mid \alpha_1 \beta_2 \alpha_2 \mid \ldots \mid \alpha_1 \beta_k \alpha_2$.

Пусть дана грамматика с множеством продукций

$S \to AB \mid Bb \mid Ba$

$A \to a$

$B \to aSb \mid b$

Выполнив замену вхождений нетерминала *B*, получим

$S \to AaSb \mid Ab \mid aSbb \mid bb \mid aSba \mid ba$

$A \to a$

Замена вхождения нетерминала *A* приведет к следующему множеству продукций:

$S \to aaSb \mid ab \mid aSbb \mid bb \mid aSba \mid ba$

В общем случае замена вхождений приводит к увеличению числа продукций. Исключение представляет случай, когда заменяемый нетерминал является левой частью единственной продукции (в примере – нетерминал *A*).

Можно использовать и обратное преобразование, когда некоторая подстрока заменяется новым нетерминалом для сокращения числа продукций.

1.6.3. Устранение леворекрсивных продукций

Продукция вида $A \to A\alpha$, где $A \in V_N$, $\alpha \in (V_T \cup V_N)^*$, называется *леворекурсивной* (содержит *прямую левую рекурсию*), а продукция вида $A \to \alpha A$ – *праворекурсивной*. Для любой КС-грамматики, содержащей леворекурсивные продукции, можно построить эквивалентную КС-грамматику, не содержащую леворекурсивных продукций.

Преобразование заключается в следующем. Пусть множество продукций грамматики содержит леворекурсивные продукции $A \to A\alpha_1 | A\alpha_2 | \ldots | A\alpha_m$ и все остальные A-продукции $A \to \beta_1 | \beta_2 | \ldots | \beta_n$, не являющиеся леворекурсивными. Тогда новая эквивалентная грамматика может быть построена добавлением нового нетерминала A' и заменой леворекурсивных продукций продукциями вида

$A \to \beta_1 | \beta_2 | \ldots | \beta_n | \beta_1 A' | \beta_2 A' | \ldots | \beta_n A'$

$A' \to \alpha_1 | \alpha_2 | \ldots | \alpha_m | \alpha_1 A' | \alpha_2 A' | \ldots | \alpha_m A'$

Таким образом, рассмотренное преобразование заменяет левую рекурсию на правую и может быть выполнено для любой КС-грамматики.

Для иллюстрации техники преобразования рассмотрим грамматику (E – начальный символ):

$E \to E + T \mid T$

$T \to T \times F \mid F$

$F \to (E) \mid i$

В соответствии с рассмотренным выше правилом продукции $E \to E + T \mid T$ преобразуются в продукции $E \to T \mid TE'$ и $E' \to +T \mid +TE'$, аналогично продукции $T \to T \times F \mid F$ преобразуются в продукции $T \to F \mid FT'$ и $T' \to \times F \mid \times FT'$. В результате получается грамматика без леворекурсивных продукций:

$E \to T \mid TE'$

$E' \to +T \mid +TE'$

$T \to F \mid FT'$

$T' \to \times F \mid \times FT'$

$F \to (E) \mid i$

1.6.4. Устранение леворекурсивного цикла

Говорят, что грамматика имеет *леворекурсивный цикл* (*косвенную левую рекурсию*), если в грамматике имеется нетерминал A такой, что $A \overset{+}{\Rightarrow} A\alpha$, т. е. из нетерминала A можно вывести строку, начинающуюся с A. Отметим, что понятие леворекурсивной продукции есть частный случай общего понятия леворекурсивного цикла.

Грамматику, содержащую леворекурсивный цикл, можно достаточно просто преобразовать в грамматику, содержащую только леворекурсивные продукции (прямую левую рекурсию), и далее исключить леворекурсивные продукции, преобразовав их в праворекурсивные. Замена леворекурсивного цикла на прямую левую рекурсию представлена алгоритмом 1.3.

Алгоритм 1.3. Замена леворекурсивного цикла на прямую левую рекурсию
Вход: КС-грамматика $G = (V_T, V_N, P, S)$ с леворекурсивным циклом
Выход: КС-грамматика $G' = (V_T, V_N, P', S)$ с прямой левой рекурсией

Шаг 1. Упорядочить нетерминалы грамматики, начиная с начального, в порядке их появления в продукциях грамматики, т. е. $V_N = \{A_1, A_2, \ldots, A_m\}$, $m \geq 1$, где A_1 соответствует начальному нетерминалу. В результате продукции, у которых правая часть начинается с нетерминала, примут вид $A_i \to A_j\alpha$, где $\alpha \in (V_T \cup V_N)^*$. Если у всех продукций $i < j$, левая рекурсия в грамматике отсутствует, если $i = j$, данная продукция является леворекурсивной, если $i > j$, имеет место леворекурсивный цикл.

Шаг 2. Для продукций вида $A_i \to A_j\alpha$, где $i > j$, производится замена вхождений нетерминала A_j. Такая последовательность замен повторяется до тех пор, пока не получится продукция, для которой $i = j$, т. е. пока не сведется к прямой левой рекурсии.

В качестве примера такого преобразования рассмотрим следующую грамматику:

$S \to AS | AB$

$A \to BS | a$

$B \to SA | b$

Прежде всего упорядочим нетерминалы грамматики. Для этого переименуем их следующим образом: S – в A_1, A – в A_2 и B – в A_3. В результате продукции будут иметь вид

$A_1 \to A_2A_1 | A_2A_3$

$A_2 \to A_3A_1 | a$

$A_3 \to A_1A_2 | b$

Продукция $A_3 \to A_1A_2$ показывает, что в грамматике имеется леворекурсивный цикл. Произведем замену вхождений нетерминала A_1 следующим образом: $A_3 \to A_2A_1A_2 | A_2A_3A_2$. Поскольку условие $i = j$ еще не выполнено, продолжим процесс замены вхождений A_2: $A_3 \to A_3A_1A_1A_2 | aA_1A_2 | A_3A_1A_3A_2 | aA_3A_2$. Получили леворекурсивные продукции, процесс замен прекращается. В результате получена грамматика, в которой вместо леворекурсивного цикла имеется прямая левая рекурсия

$S \to AS | AB$

$A \to BS | a$

$B \to BSSA | aSA | BSBA | aBA | b$

1.6.5. Факторизация

Если в грамматике имеются продукции вида

$$A \to \alpha\beta_1 \mid \alpha\beta_2 \mid \ldots \mid \alpha\beta_k,\ \alpha \in (V_T \cup V_N)^+,\ \beta_i \in (V_T \cup V_N)^*,\ 1 \le i \le k$$

с нетерминалом A в левой части, то эти продукции можно заменить, добавив новый нетерминал X, на следующие:

$A \to \alpha X$

$X \to \beta_1 \mid \beta_2 \mid \ldots \mid \beta_k$

Такое преобразование называется *левой факторизацией*. Левую факторизацию можно рассматривать как вынос за скобки общего префикса α, а то, что осталось в скобках, заменяется новым нетерминалом X. Для наглядности в качестве промежуточной можно использовать форму записи $A \to \alpha(\beta_1 \mid \beta_2 \mid \ldots \mid \beta_k)$.

Например, пусть дана грамматика с продукциями

$S \to aSb \mid aSc \mid d$

Факторизации преобразует их в следующие продукции:

$S \to aSX \mid d$

$X \to b \mid c$

Аналогично можно определить правую факторизацию.

1.6.6. Удаление ε-продукций

Продукция вида $A \to \varepsilon$ называется ε-*продукцией* (*аннулирующей продукцией*). Наличие в грамматике ε-продукций может усложнить реализацию некоторых методов синтаксического анализа. Поэтому на практике удаление ε-продукций позволяет упростить процесс построения синтаксического анализатора, хотя и может привести к существенному росту числа продукций. В случаях, когда при преобразованиях грамматики к виду, необходимому для реализации ряда методов синтаксического анализа, напротив, могут появиться ε-продукции, обычно они проблем не вызывают.

Следует отметить, что если пустая строка принадлежит языку, то избавиться от ε-продукции в ее грамматике, не изменяя порождаемого языка, невозможно. Самое большее, чего можно достигнуть, – это преобразовать грамматику G таким образом, чтобы полученная грамматика G' не содержала ε-продукции и выполнялось условие $L(G') = L(G) - \{\varepsilon\}$.

Грамматика $G = (V_T, V_N, P, S)$ называется ε-*свободной* (или *неукорачивающей*):

а) либо она не имеет ε-продукций (в случае, когда $\varepsilon \notin L(G)$;

б) либо имеется только одна ε-продукция $S \to \varepsilon$ и S не встречается в правых частях всех продукций грамматики (в случае, когда $\varepsilon \in L(G)$.

Нетерминал $A \in V_N$ называется ε-*порождающим*, если $A \overset{*}{\Rightarrow} \varepsilon$, т. е. из A выводима пустая строка. Построение множества $N_\varepsilon \subseteq V_N$ ε-порождающих нетерминалов для заданной КС-грамматики $G = (V_T, V_N, P, S)$ можно представить алгоритмом 1.4. Множество N_ε определяется путем рекурсивного построения множеств $N_0, N_1, N_2, \ldots$ Логическая переменная b служит для реализации выхода из цикла после завершения вычисления множества N_ε.

Алгоритм 1.4. Построение множества ε-порождающих нетерминалов
Вход: КС-грамматика $G = (V_T, V_N, P, S)$
Выход: N_ε – множество ε-порождающих нетерминалов

$$N_0 := \{A \mid A \to \varepsilon \in P\};\ \ i := 1;\ \ b := \mathbf{true}$$

$$\mathbf{while}\ b\ \ \mathbf{do}\ \ \begin{cases} N_i := N_{i-1} \cup \{A \mid A \to \alpha \in P,\ \ \alpha \in N_{i-1}^{+}\} \\ \mathbf{if}\ \ N_i \neq N_{i-1}\ \ \mathbf{then}\ i := i+1\ \ \mathbf{else}\ b := \mathbf{false} \end{cases}$$

$$N_\varepsilon := N_i$$

Для любой КС-грамматики $G = (V_T, V_N, P, S)$, содержащей ε-продукции, можно построить эквивалентную ε-свободную КС-грамматику $G' = (V_T, V'_N, P', S')$ в соответствии с алгоритмом 1.5.

Алгоритм 1.5. Удаление ε-продукций
Вход: КС-грамматика $G = (V_T, V_N, P, S)$
Выход: ε-свободная КС-грамматика $G' = (V_T, V'_N, P', S')$

Шаг 1. Построить множество $N_\varepsilon \subseteq V_N$ ε-порождающих нетерминалов в соответствии с алгоритмом 1.4.

Шаг 2. $P' := P - \{A \to \varepsilon \in P$ для всех $A \in V_N\}$.

Шаг 3. Для каждой продукции $p \in P'$ вида

$$A \to \alpha_0 B_1 \alpha_1 B_2 \alpha_1 \ldots B_k \alpha_k \in P,$$

где $k \geq 0$, $B_j \in N_\varepsilon$ $(1 \leq j \leq k)$ и ни один из символов строк $\alpha_i \in (V_T \cup V_N)^*$ $(0 \leq i \leq k)$ не содержит нетерминалы из N_ε, включить в P' все продукции вида $A \to \alpha_0 X_1 \alpha_1 X_2 \alpha_1 \ldots X_k \alpha_k$, где $X_j = B_j$ или $X_j = \varepsilon$. Если все строки $\alpha_i = \varepsilon$, то продукцию $A \to \varepsilon$ не включать в P'.

Шаг 4. Если $S \in N_\varepsilon$, то $V'_N := \{S'\} \cup V_N$ и добавить в P' продукции $S' \to \varepsilon \mid S$, где S' – новый начальный символ грамматики. В противном случае положить $V'_N = V_N$ и $S' = S$. Замечание: если $S \in N_\varepsilon$ и S не встречается в правых частях продукций, то достаточно добавить в P' продукцию $S \to \varepsilon$, тогда $V'_N = V_N$ и $S' = S$.

На первом шаге алгоритма 1.5 строится множество $N_\varepsilon \subseteq V_N$ ε-порождающих нетерминалов. На втором шаге в множество P' объединяются все имеющиеся в P продукции, за исключением ε-продукций. На третьем шаге каждой продукции $p \in P'$, у которой в правой части содержатся ε-порождающие нетерминалы, ставятся в соответствие такие продукции, что в их правых частях опущены (по сравнению с продукцией p) все возможные комбинации ε-порождающих нетерминалов из множества N_ε, полученные продукции присоединяются к множеству P'. Другими словами, необходимо во все продукции грамматики выполнить все возможные подстановки пустой строки вместо ε-порождающего нетерминала. Четвертый шаг выполняется только для случая $\varepsilon \in L(G)$, т. е. пустая строка принадлежит языку.

Проиллюстрируем преобразование на примере грамматики

$S \to ASB \mid \varepsilon$

$A \to aA \mid \varepsilon$

$B \to bB \mid b$

Нетерминалы S и A являются ε-порождающими. Выполнение всех возможных замен этих нетерминалов пустой строкой приведет к следующим продукциям:

$S \to ASB \mid SB \mid AB \mid B$

$A \to aA \mid a$

$B \to bB \mid b$

Поскольку пустая строка принадлежит языку, а начальный нетерминал S входит в правые части некоторых продукций, добавим продукции $S' \to \varepsilon$ и $S' \to S$. В результате получается следующая грамматика:

$S' \to \varepsilon \mid S$

$S \to ASB \mid SB \mid AB \mid B$

$A \to aA \mid a$

$B \to bB \mid b$

1.6.7. Удаление цепных продукций

Цепной (*сингулярной*) *продукцией* (*цепным правилом*) называется продукция вида $A \to B$, где $A, B \in V_N$, т. е. правая часть представляет собой один нетерминал. Цепные продукции могут быть удобны для построения грамматик (повышают наглядность грамматики), но приводят к излишним шагам при выводе.

Продукция вида $A \to A$ также относится к цепным продукциям. Если такая продукция принадлежит множеству продукций грамматики или появляется при каких-либо преобразованиях, то она должна быть просто исключена из множества продукций.

Пусть в грамматике наряду с другими продукциями имеются цепные продукции $A \to B$, $B \to C$, $C \to D$. Тогда существует вывод нетерминала D из нетерминала A, в котором на всех шагах применяются только цепные правила (*цепной вывод*):

$A \Rightarrow B \Rightarrow C \Rightarrow D$.

Поскольку грамматика приведенная (в ней нет бесполезных символов), обязательно должна существовать нецепная продукция $D \to \alpha$, т. е имеется схема вывода строки α:

$A \Rightarrow B \Rightarrow C \Rightarrow D \Rightarrow \alpha$.

Строку α можно вывести из нетерминала A за один шаг, если добавить в грамматику продукцию $A \to \alpha$. Аналогично из нетерминалов B и C можно вывести строку α за один шаг, добавив в грамматику продукции $B \to \alpha$, $C \to \alpha$. Другими словами, выполняется замена вхождений нетерминалов A, B и C на α. На этом и основано удаление цепных правил.

Формально данное преобразование можно представить алгоритмом 1.6.

Алгоритм 1.6. Удаление цепных продукций
Вход: ε-свободная КС-грамматика $G = (V_T, V_N, P, S)$
Выход: ε-свободная КС-грамматика $G' = (V_T, V_N, P', S)$ без цепных продукций

Шаг 1. Для каждого нетерминала $A \in V_N$ построить множество $N_A = \{B \mid A \overset{*}{\Rightarrow} B\}$, т. е. множество нетерминалов, для которых существует цепной вывод из нетерминала A, следующим образом (рекуррентное вычисление каждого N_A):

$$\textbf{for}\ \ A \in V_N\ \ \textbf{do}\ \begin{cases} N_0 := \{A\};\ \ i := 1;\ \ b := \textbf{true} \\ \textbf{while}\ b\ \ \textbf{do} \\ \ \ \begin{cases} N_i := N_{i-1} \cup \{C \mid C \in V_N,\ \ B \to C \in P,\ \ B \in N_{i-1} \\ \textbf{if}\ \ N_i \ne N_{i-1}\ \ \textbf{then}\ i := i+1\ \ \textbf{else}\ b := \textbf{false} \end{cases} \\ N_A := N_i \end{cases}$$

Шаг 2. Построить множество продукций P': если продукция $B \to \alpha \in P$ и не является цепной продукцией, то включить в P' продукцию $A \to \alpha$ для всех таких нетерминалов A, что $B \in N_A$.

Рассмотрим данное преобразование на примере следующей грамматики:

$E \to E + T \mid T$

$T \to T \times F \mid F$

$F \to (E) \mid i$

Для всех нетерминалов $A \in V_N$ вычислим соответствующие множества: $N_E = \{E, T, F\}$, $N_T = \{T, F\}$, $N_F = \{F\}$. Построим новое множество продукций P':

$E \to E + T \mid T \times F \mid (E) \mid i$

$T \to T \times F \mid (E) \mid i$

$F \to (E) \mid i$

Получена эквивалентная грамматика без цепных продукций.

Как видно из примера, увеличилось число продукций, что усложняет синтаксический анализ. Например, при использовании табличных методов синтаксического анализа это приводит к увеличению размеров таблиц разбора.