Регулярные языки

Правосторонние грамматики

Регулярный язык – это язык, который может быть задан *регулярной* (правосторонней или левосторонней) *грамматикой*.

Правосторонняя грамматика имеет правила двух видов:

$$A \to \alpha$$

$$A \to \alpha B,$$

где A и B – нетерминалы, а α – цепочка терминалов, возможно, пустая. Другими словами, *в правой части правил может быть не более одного нетерминала и, если он есть, то занимает самую крайнюю правую позицию.* Очевидно, что правосторонняя грамматика является подклассом КС-грамматик.

Грамматика, которая может имеет правила двух видов:

$$A \to tB$$

$$A \to \varepsilon\,,$$

где A и B – нетерминалы, а t – терминал, называется *автоматной правосторонней* или *правосторонней грамматикой специального вида*.

Любую правостороннюю грамматику можно преобразовать в автоматную правостороннюю.

Алгоритм преобразования правосторонней грамматики в автоматную правостороннюю.

1. Если имеются правила вида $A \to \alpha$, α – непустая цепочка терминалов, то ввести новый нетерминал E и добавить правило $E \to \varepsilon$.

Затем каждое из правил вида $A \to \alpha$ заменить правилом $A \to \alpha E$.

2. Пока в грамматике есть правила вида $A \to \alpha B$, где α – цепочка более чем из одного терминала, выполнять:

правило $A \to \alpha B$ заменить правилами $A \to tC$ и $C \to \beta B$,

где t – первый терминал в цепочке α ,

C – новый нетерминал,

β - цепочка α без первого терминала.

3. Устранить цепные правила.

Пример.

Привести правостороннюю грамматику к автоматному виду.

1. S→aA	4. A→abS
2. S→b	5. A→cA
3. S→A	6. A→ε

1. В грамматике есть правило 2 вида $A \to \alpha$, поэтому введём новый нетерминал E , добавим правило 7 и изменим правило 2 :

1. S→aA	4. A→abS
2. S→bE	5. A→cA
3. S→A	6. A→ε
	7. E→ε

2. Пребразуем правило 4, обозначив цепочку bS ннетерминалом B :

1. S→aA	4.2. B→bS
2. S→bE	5. A→cA
3. S→A	6. A→ε
4.1. A→aB	7. E→ε

3. Устраним цепное правило 3 :

1. S→aA	4.1. A→aB
2. S→bE	4.2. B→bS
3.1. S→aB	5. A→cA
3.1. S→cA	6. A→ε
3.1. S→ε	7. E→ε

Автоматная правосторонняя грамматика получена.

Левосторонние грамматики

Регулярный язык может быть задан левосторонней грамматикой. *Левосторонняя* грамматика имеет правила двух видов:

$$A \to \alpha$$

$$A \to B\alpha,$$

где A и B – нетерминалы,

α - цепочка терминалов, возможно, пустая.

Грамматика, которая имеет правила двух видов:

$$A \to Bt$$

$$A \to \varepsilon,$$

где A и B – нетерминалы, а t – терминал, назвается *автоматной левосторонней* или *левосторонней грамматикой специального вида*.

Левостороннюю грамматику можно преобразовать в автоматную левостороннюю.

Алгоритм преобразования левосторонней грамматики в автоматную левостороннюю.

1. Если имеются правила вида $A \to \alpha$, α – непустая цепочка терминалов, то ввести новый нетерминал E и добавить правило $E \to \varepsilon$. Затем каждое из правил вида $A \to \alpha$ заменить правилом $A \to E\alpha$.

2. Пока в грамматике есть правила вида $A \to B\alpha$, где α – цепочка более чем из одного терминала, выполнять:

правило $A \to B\alpha$ заменить правилами $A \to Ct$ и $C \to B\beta$,

где t – последний терминал в цепочке α ,

C – новый нетерминал,

β – цепочка α без последнего терминала.

3. Устранить цепные правила.

Пример.

Привести левостороннюю грамматику к автоматному виду.

1. S→Aa	4. A→Sab
2. S→b	5. A→Ac
3. S→A	6. A→ε

1. В грамматике есть правило 2 вида $A \to \alpha$, поэтому введём новый нетерминал E , добавим правило 7 и изменим правило 2 :

1. S→Aa	4. A→Sab
2. S→Eb	5. A→Ac
3. S→A	6. A→ε
	7. E→ε

2. Выполним выделение нетерминала в правиле 4, обозначив цепочку Sa как нетерминал B :

1. S→Aa	4.2. B→Sa
2. S→Eb	5. A→Ac
3. S→A	6. A→ε
4.1. A→Bb	7. E→ε

3. Устраним цепное правило 3 :

1. S→Aa	4.1. A→Bb
2. S→Eb	4.2. B→Sb
3.1. S→Bb	5. A→Ac
3.1. S→Ac	6. A→ε
3.1. S→ε	7. E→ε

Автоматная левосторонняя грамматика получена.

Связь правосторонних и левосторонних грамматик

Класс левосторонних грамматик эквивалентен классу правосторонних грамматик, т.е. любой язык, заданный правосторонней грамматикой, может быть задан левосторонней грамматикой и любой язык, заданный левосторонней грамматикой, может быть задан правосторонней грамматикой.

Покажем, что любую левостороннюю грамматику можно преобразовать в эквивалентную ей правостороннюю грамматику и любую правостороннюю грамматику можно преобразовать в эквивалентную ей левостороннюю грамматику.

Будем считать, что в исходной грамматике исключены цепные и ε-правила выполнением соответствующих преобразований.

Алгоритм преобразования левосторонней грамматики в правостороннюю.

1. Устранить левую рекурсию.

2. Выполнить замену края.

3. В каждом i-ом правиле вида $A\to\alpha B\beta$, где α и β – непустые цепочки терминалов, выделить новый нетерминал C_i, определяемый правилом $C_i\to B\beta$, а правило $A\to\alpha B\beta$ заменить на $A\to\alpha\, C_i$.

4. Выполнить замену края.

5. В правых частях правил грамматики подцепочки, представляющие собой правую часть правила, определяющего нетерминал C_i в правилах, полученных в п.3, заменить на C_i .

В результате устранения левой рекурсии (п.1 алгоритма) можем получить правила вида:

1. $A \to B\alpha_1$ - в правой части нетерминал и цепочка терминалов;
2. $A \to \alpha_2$ - в правой части цепочка терминалов;
3. $B \to \alpha_3 B'$ - в правой части цепочка терминалов и новый нетерминал;
4. $B' \to \alpha_4 B'$
5. $B' \to \varepsilon$

Правила вида 1-3 определяют нетерминалы исходной грамматики, а правила вида 4 и 5 определяют нетерминалы, введённые в грамматику при устранении левой рекурсии. Среди получаемых правил только правила вида 1 не соответствуют правилам правосторонней грамматики. В соответствии с п.2 алгоритма выполним замену края в правилах вида 1. В результате замены края можем получить правила вида 2 и правила вида:

6. $A \to \alpha_3 B'\alpha_1$

Правило вида 6 не соответствует правилам правосторонней грамматики. По п.3 алгоритма выполним выделение нетерминала С и получим правила вида:

7. $A \to \alpha_3 C$
8. $C \to B'\alpha_1$

В правилах вида 8 выполним замену края (п.4 алгоритма), используя правила вида 4 и 5. В результате вместо правил вида 8 получим правила вида:

9. $C \to \alpha_4 B'\alpha_1$
10. $C \to \alpha_1$

В правилах вида 9 есть подцепочка $B'\alpha_1$, соответствующая правой части правила вида 8, и, заменяя её на левую часть правила вида 8 (п.5 алгоритма), получаем правостороннюю грамматику.

Пример.
Преобразовать левостороннюю грамматику в правостороннюю.

1. $A \to Ba$
2. $A \to ab$
3. $B \to Bca$
4. $B \to Ac$
5. $B \to b$

1. В результате устранения левой рекурсии получим:

1. $A \to Ba$
2. $A \to ab$
3. $B \to abcB'$
4. $B \to bB'$
5. $B' \to caB'$
6. $B' \to acB'$
7. $B' \to \varepsilon$

2. В результате замены края в правиле 1 получим:

1. $A \to abcB'a$
2. $A \to bB'a$
3. $A \to ab$
4. $B' \to caB'$
5. $B' \to acB'$
6. $B' \to \varepsilon$

3. В результате выделения нетерминала C в правилах 1 и 2 получим:

1. $A \to abcC$
2. $A \to bC$
3. $A \to ab$
4. $B' \to caB'$
5. $B' \to acB'$
6. $B' \to \varepsilon$
7. $C \to B'a$

4. В результате замены края в правиле 7 получим:

1. $A \to abcC$
2. $A \to bC$
3. $A \to ab$
4. $B' \to caB'$
5. $B' \to acB'$
6. $B' \to \varepsilon$
7. $C \to caB'a$
8. $C \to acB'a$
9. $C \to a$

5. В результате замены подцепочки $B'a$ в правилах 7 и 8 на нетерминал C в соответствии с правилом *7.* $C \to B'a$ п.3 получим правостороннюю грамматику:

1. $A \to abcC$
2. $A \to bC$
3. $A \to ab$
4. $B' \to caB'$
5. $B' \to acB'$
6. $B' \to \varepsilon$
7. $C \to caC$
8. $C \to acC$
9. $C \to a$

При преобразовании **правосторонней грамматики в левостороннюю** будем использовать операцию обращения языка.

Если язык L задан порождающей грамматикой G , то язык L^{-1} – обращение языка L , можно задать грамматикой G’ , полученной из грамматики G обращением правой части каждого правила (изменения порядка следования символов в правой части правила на обратный).

Если язык L задан правосторонней грамматикой G , то его обращение – язык L^{-1} , можно задать левосторонней грамматикой G’ . Если же язык L задан левосторонней грамматикой G , то его обращение можно задать правосторонней грамматикой G’ . Естественно, что обращение обращения языка L есть язык L, т.е. $L= L^{-1-1}$. Имея алгоритм преобразования левосторонней грамматики в правостороннюю и учитывая вышесказанное, получаем простой алгоритм преобразования правосторонней грамматики в левостороннюю.

Алгоритм преобразования правосторонней грамматики в левостороннюю.

1. Получить левостороннюю грамматику G’ путём обращения правых частей правил грамматики G. Грамматика G’ порождает обращение языка, заданного исходной грамматикой.

2. Преобразовать левостороннюю грамматику G’ в эквивалентную ей правостороннюю грамматику G’’ .

3. Получить правостороннюю грамматику G’’’ путём обращения правых частей правил грамматики G’’. Грамматика G’’’ порождает обращение языка, заданного грамматикой G’’, т.е. исходный язык.

Пример.

Преобразовать правостороннюю грамматику в левостороннюю.

1. $A \to aB$	3. $B \to acB$
2. $A \to ba$	4. $B \to cA$
	5. $B \to b$

1. Получим левостороннюю грамматику путём обращения правых частей правил исходной грамматики:

1. $A \to Ba$	3. $B \to Bca$
2. $A \to ab$	4. $B \to Ac$
	5. $B \to b$

2. В результате преобразования левосторонней грамматики в правостороннюю (процесс преобразования рассмотрен в предыдущем примере) получим:

1. $A \to abcC$	5. $B' \to acB'$
2. $A \to bC$	6. $B' \to \varepsilon$
3. $A \to ab$	7. $C \to caC$
4. $B' \to caB'$	8. $C \to acC$
	9. $C \to a$

3. В результате обращения правых частей правил получим левостороннюю грамматику, эквивалентную исходной правосторонней:

1. $A \to Ccba$	5. $B' \to B'ca$
2. $A \to Cb$	6. $B' \to \varepsilon$
3. $A \to ba$	7. $C \to Cac$
4. $B' \to B'ac$	8. $C \to Cca$
	9. $C \to a$

Для преобразования правосторонней грамматики в левостороннюю можно использовать алгоритм, аналогичный алгоритму преобразования левосторонней грамматики в правостороннюю.

Алгоритм преобразования правосторонней грамматики в левостороннюю.

1. Устранить правую рекурсию.
2. Выполнить замену нетерминалов, занимающих самую правую позицию в правых частях правил.
3. В каждом i-ом правиле вида $A \to \alpha B \beta$, где α и β - непустые цепочки терминалов, выделить новый нетерминал C_i , определяемый правилом $C_i \to \alpha B$, а правило $A \to \alpha B \beta$ заменить на $A \to C_i \beta$.
4. Выполнить п.2.
5. В правых частях правил грамматики подцепочки, представляющие собой правую часть правила, определяющего нетерминал C_i в правилах, полученных в п.3, заменить на C_i .

В результате устранения правой рекурсии (п.1 алгоритма) можем получить правила вида:

1. $A \to \alpha_1 B$ - в правой части цепочка терминалов и нетерминал;
2. $A \to \alpha_2$ - в правой части цепочка терминалов;
3. $B \to B' \alpha_3$ - в правой части новый нетерминал и цепочка терминалов;
4. $B' \to B' \alpha_4$
5. $B' \to \varepsilon$

Правила вида 1-3 определяют нетерминалы исходной грамматики, а правила вида 4 и 5 определяют нетерминалы, введённые в грамматику при устранении правой рекурсии. Среди получаемых правил только правила вида 1 не соответствуют правилам левосторонней грамматики. В соответствии с п.2 алгоритма выполним замену нетерминалов, занимающих самую правую позицию в правых частях правил в правилах вида 1. В результате замены можем получить правила вида 2 и правила вида:

6. $A \to \alpha_1 B' \alpha_3$

Правило вида 6 не соответствует правилам левосторонней грамматики. По п.3 алгоритма выполним выделение нетерминала С и получим правила вида:

7. $A \to C\ \alpha_3$
8. $C \to \alpha_1 B'$

В правилах вида 8 выполним замену нетерминала (п.4 алгоритма), используя правила вида 4 и 5. В результате вместо правил вида 8 получим правила вида:

9. $C \to \alpha_1 B' \alpha_4$
10. $C \to \alpha_1$

В правилах вида 9 есть подцепочка $\alpha_1 B'$, соответствующая правой части правила вида 8, и, заменяя её на левую часть правила вида 8 (п.5 алгоритма), получаем левостороннюю грамматику.

Пример.

Преобразовать правостороннюю грамматику в левостороннюю.

1. $A \to aB$	3. $B \to acB$
2. $A \to ba$	4. $B \to cA$
	5. $B \to b$

1. В результате устранения правой рекурсии получим:

1. $A \to aB$	5. $B' \to B'ac$
2. $A \to ba$	6. $B' \to B'ca$
3. $B \to B'cba$	7. $B' \to \varepsilon$
4. $B \to B'b$	

2. В результате замены нетерминала в правиле 1 получим:

1. $A \to aB'cba$	4. $B' \to B'ac$
2. $A \to aB'b$	5. $B' \to B'ca$
3. $A \to ba$	6. $B' \to \varepsilon$

3. В результате выделения нетерминала C в правилах 1 и 2 получим:

1. $A \to Ccba$	4. $B' \to B'ac$
2. $A \to Cb$	5. $B' \to B'ca$
3. $A \to ba$	6. $B' \to \varepsilon$
	7. $C \to aB'$

4. В результате замены нетерминала в правиле 7 получим:

1. $A \to Ccba$	5. $B' \to B'ca$
2. $A \to Cb$	6. $B' \to \varepsilon$
3. $A \to ba$	7. $C \to aB'ac$
4. $B' \to B'ac$	8. $C \to aB'ca$
	9. $C \to a$

5. В результате замены подцепочки $a\,B'$ в правилах 7 и 8 на нетерминал C в соответствии с правилом *7.* $C \to aB'$ п.3 получим левостороннюю грамматику:

1. $A \to Ccba$	5. $B' \to B'ca$
2. $A \to Cb$	6. $B' \to \varepsilon$
3. $A \to ba$	7. $C \to Cac$
4. $B' \to B'ac$	8. $C \to Cca$
	9. $C \to a$

Преобразование КС-грамматики произвольного вида в правостороннюю

Регулярный язык может быть задан КС-грамматикой произвольного вида, т.е. без ограничений на правую часть правил. В этом случае исходную грамматику можно преобразовать в регулярную, например в правостороннюю.

Если же исходная грамматика порождает нерегулярный язык, то её нельзя преобразовать в правостороннюю с конечным числом правил.

Для преобразования КС-грамматики произвольного вида в правостороннюю, представим её в такой форме, что правая часть каждого правила начинается терминалом (см.’Нормальная форма Грейбах’). Такая грамматика может содержать правила вида $A\to\alpha B$ и $A\to\alpha$, где A и B – нетерминалы, α – последовательность терминалов, удовлетворяющие правосторонней грамматике, и правила вида $A\to\alpha\beta$, где β - последовательность терминалов и нетерминалов, начинающаяся нетерминалом, содержащая более одного символа, которые неудовлетворяют правосторонней грамматике.

Рассмотрим возможную последовательность действий, позволяющую выразить правило вида $A\to\alpha\beta$ множеством правил, удовлетворяющих правосторонней грамматике.

Пусть в грамматике есть правило $A\to\alpha_1\beta_1$. На первом шаге заменим его правилом, удовлетворяющим правосторонней грамматике $A\to\alpha_1N_1$, а нетерминал N_1 определим правилом $N_1\to\beta_1$. Это правило неудовлетворяет правосторонней грамматике, т.к. правая часть (край) начинается нетерминалом, за которым следуют другие символы (по условию). Теперь выполним замену края. При этом край заменяется правыми частями всех правил, его определяющих. Пусть среди них будет правило с номером i_1 . В результате замены края по правилу с номером i_1 можем получить либо правило $N_1\to\alpha_2\beta_2$, либо правило $N_1\to\alpha_2B$, удовлетворяющее правосторонней грамматике. Допустим получено правило $N_1\to\alpha_2\beta_2$.

На втором шаге выполним аналогичные действия: заменим $N_1\to\alpha_2\beta_2$ правилом $N_1\to\alpha_2N_2$, определим $N_2\to\beta_2$, выполним замену края и применяя правило с номером i_2 , получим правило $N_2\to\alpha_3\beta_3$ и т.д.

На n-ом шаге при замене края в правиле $N_n\to\beta_n$ по правилу с номером i_n получим правило $N_n\to\alpha_{n+1}\beta_{n+1}$. Далее заменим это правило на $N_n\to\alpha_{n+1}N_{n+1}$ и определим $N_{n+1}\to\beta_{n+1}$.

Если $\beta_{n+1}=\beta_j$, $1\le j \le n$, то правило $N_n\to\alpha_{n+1}\beta_{n+1}$ можно заменить на $N_n\to\alpha_{n+1}N_j$ и новый нетерминал N_{n+1} и правило $N_{n+1}\to\beta_{n+1}$ вводить не нужно. Т.о. процесс добавления новых нетерминалов и правил в данном направлении останавливается. Но, т.к. замена края выполняется по всем правилам, определяющим край, процесс добавления новых нетерминалов и правил может продолжаться по другим направлениям. Если этот процесс остановится по всем направлениям, то будет получена правосторонняя грамматика.

Если же β_{n+1} и β_j , $1\le j \le n$, имеют общий префикс и β_{n+1} представляет собой β_j , дополненную цепочкой χ , то в процессе преобразований можно снова применить последовательность правил $i_j, i_{j+1},\ldots, i_n$, в результате чего получим правило $N_n\to\beta_k$ и β_k будет иметь общий префикс с β_j и содержать ещё одну цепочку χ . Повторное применение последовательности правил $i_j, i_{j+1},\ldots, i_n$ приводит к добавлению цепочки χ в β_j . Это требует введения нового правила и процесс преобразования зацикливается, получить правостороннюю грамматику с конечным числом правил в этом случае нельзя.

Для обнаружения зацикливания будем строить и анализировать множество деревьев, вершины которых соответствуют вводимым правилам вида $N\to\beta$. Если в процессе преобразований в результате замены края в правиле $N_j\to\beta_j$ по i-ому правилу получаем $N_j\to\alpha_k\beta_k$, которое заменяем на $N_j\to\alpha_k N_k$ и вводим правило $N_k\to\beta_k$, то в дереве из вершины с правилом $N_j\to\beta_j$ проводим дугу в вершину с правилом $N_k\to\beta_k$.

Если в дереве существует путь от вершины с правилом $N_j\to\beta_j$ до вершины с правилом $N_m\to\beta_m$, цепочки β_j и β_m имеют общий префикс и цепочка β_m получена из цепочки β_j добавлением (вставкой) непустой цепочки χ , то процесс преобразования зацикливается и получить правостороннюю грамматику с конечным числом правил нельзя.

Алгоритм преобразования КС-грамматики, правые части правил которой начинаются терминалом, в правостороннюю.

Вход: G – исходная грамматика.

Выход: G' – правосторонняя грамматика.

1. Последовательно просмотреть все правила исходной грамматики. Правила вида $A\to\alpha B$ и $A\to\alpha$, удовлетворяющие правосторонней грамматике, включить в множество правил грамматики G'. Правила вида $A\to\alpha\beta$, неудовлетворяющие правосторонней грамматике, обработать в соответствии с п.2.

2. Обработка правила вида $A\to\alpha\beta$.

Если в деревеьях есть вершина с правилом $N_m\to\beta$, то правило $A\to\alpha N_m$ включить в множество правил грамматики G' , конец п.2.

Если в деревьях нет вершины с правилом $N_m\to\beta$, то создать вершину (корень нового дерева) с правилом $N_m\to\beta$, правило $A\to\alpha N_m$ включить в множество правил грамматики G' и обработать вершины дерева (листья) в порядке их создания в соответствии с п.3.

3. Обработка вершины с правилом $N_m\to\beta$.

Выполнить замену края. В результате получим множество правил, определяющих нетерминал N_m : $N_m \to \alpha_i \beta_i$. Каждое правило этого множества обработать в соответствии с п.4.

4. Обработка правила $N_m \to \alpha_i \beta_i$, полученного в п.3.

Если в деревеьях есть вершина с правилом $N_k \to \beta_i$, то правило $N_m \to \alpha_i N_m$ включить в множество правил грамматики G' , конец п.4.

Если в деревеьях нет вершины с правилом $N_k \to \beta_i$, то создать вершину с правилом $N_k \to \beta_i$, провести в неё дугу из вершины с правилом $N_m \to \beta$ (см. п.3) и правило $N_m \to \alpha_i N_m$ включить в множество правил грамматики G'. Если на пути отсозданной вершины (с правилом $N_k \to \beta_i$) к корню дерева есть вершина с правилом $N_p \to \beta_j$, таким, что β_j и β_i имеют общий префикс и цепочка β_i получена из цепочки β_j добавлением (вставкой) непустой цепочки χ , то получить правостороннюю грамматику с конечным числом правил нельзя, конец алгоритма.

Ниже приведены примеры преобразования. Рядом с вершинами деревьев записаны множества правил, получаемых из правил вершин выполнением замены края.

Пример 1.

Исходная грамматика	Деревья правил		Правосторонняя грамматика
$S \to aBc$	$N_1 \to Bc$	$N_1 \to cASc$	$S \to aN_1$
$S \to b$	↓	$N_1 \to aBc$	$N_1 \to cN_2$
$B \to cAS$			$N_1 \to aN_1$
$B \to aB$	$N_2 \to ASc$	$N_2 \to bBSc$	$N_2 \to bN_3$
$A \to bB$	↓	$N_2 \to dSc$	……….
$A \to d$			……….
	$N_3 \to BSc$		

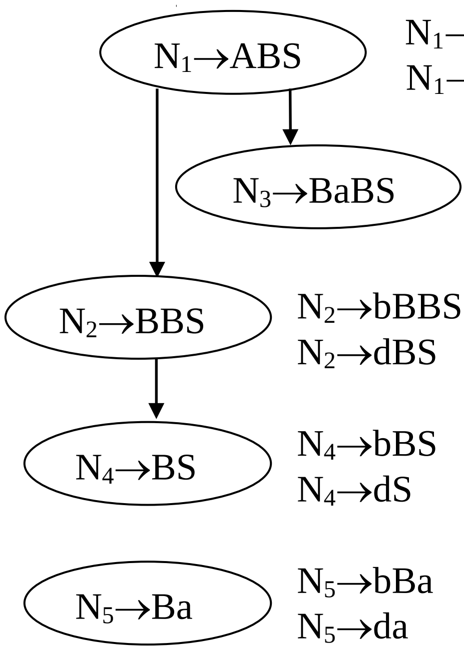
Правые части правил $N_1 \to Bc$ (корень) и $N_3 \to BSc$ (лист) имеют общий префикс (B) и правая часть правила $N_3 \to BSc$ представляет собой правую часть правила $N_1 \to Bc$, дополненную символом S . Дальнейшие преобразования будут добавлять символ S в цепочку BSc , в результате чего будем получать бесконечное число правил вида $N \to BS^*c$. Правостороннюю грамматику с конечным числом правил получить нельзя.

Пример 2.

Исходная грамматика

$S \to aABS$
$S \to bB$
$A \to aB$
$A \to cBa$
$B \to bB$
$B \to d$

Деревья правил



$N_1 \to aBBS$
$N_1 \to cBaBS$

$N_3 \to bBaBS$
$N_3 \to daBS$

$N_2 \to bBBS$
$N_2 \to dBS$

$N_4 \to bBS$
$N_4 \to dS$

$N_5 \to bBa$
$N_5 \to da$

Правосторонняя грамматика

$S \to aN_1$
$N_1 \to aN_2$
$N_1 \to cN_3$
$N_2 \to bN_2$
$N_2 \to dN_4$
$N_3 \to bN_3$
$N_3 \to daN_4$
$N_4 \to bN_4$
$N_4 \to dS$
$S \to bB$
$A \to aB$
$A \to cN_5$
$N_5 \to bN_5$
$N_5 \to da$
$B \to bB$
$B \to d$

Характерной особенностью КС-грамматик, порождающих нерегулярный язык, является наличие самовложенных нетерминалов. Нетерминал A называется *самовложенным*, если существует вывод $A \Rightarrow^* \alpha A \beta$, где α и β - непустые цепочки терминалов и нетерминалов. КС-грамматика произвольного вида, несодержащая самовложенных нетерминалов, порождает регулярный язык и применение к ней рассмотренного выше алгоритма всегда приводит к получению правосторонней грамматики с конечным числом правил.

Для нахождения самовложенных нетерминалов грамматики G, правые части правил которой начинаются терминалом, будем анализировать грамматику G', включающую в себя множество *псевдоправил, соответствующих правилам грамматики G, содержащим хотя бы один нетерминал в правой части.*

Псевдоправило в правой части содержит один нетерминал, который может быть окружён символами α.

Каждому вхождению нетерминала в правую часть правила $A \rightarrow \chi$ грамматики G соответствует псевдоправило.

Пусть правая часть правила $A \rightarrow \chi$ содержит нетерминал B.

Если нетерминал B самый правый символ в цепочке χ, то ему соответствует псевдоправило $A \rightarrow \alpha B$.

Если нетерминал B самый левый символ в цепочке χ, то ему соответствует псевдоправило $A \rightarrow B\alpha$.

Если нетерминал B в цепочке χ окружён непустыми цепочками символов и справа и слева, то ему соответствует псевдоправило $A \rightarrow \alpha B \alpha$.

Символ α в псевдоправиле заменяет всю непустую цепочку символов, расположенную перед или после рассматриваемого нетерминала в исходном правиле.

В грамматике G' из каждого нетерминала можно вывести конечное множество (выводимых) псевдоцепочек, состоящих из нетерминала и не более двух, не стоящих рядом, символов α. Определим *множество псевдоцепочек, выводимых из нетерминала A*, следующим образом:

– псевдоцепочка, состоящая из нетерминала A – выводимая;

– если псевдоцепочка β выводима из нетерминала A, то псевдоцепочка δ, полученная из псевдоцепочки β заменой входящего в неё нетерминала правой частью псевдоправила, содержащего в левой части этот нетерминал, и заменой каждой последовательности из двух символов α одним символом α – выводимая.

Если псевдоцепочка $\alpha A\alpha$ выводима в грамматике G' из нетерминала A, то нетерминал A в грамматике G является самовложенным.

Для нахождения множества псевдоцкпочек, выводимых из нетерминала A, будем строить дерево, корень которого содержит нетерминал A, остальные вершины – псевдоцепочки, выводимые из нетерминала A, причём из вершины, содержащей псевдоцепочку β, идёт дуга в вершину, содержащую псевдоцепочку δ, если δ непосредственно выводима из β.

Алгоритм построения дерева выводимых псевдоцепочек из нетерминала A.

1. Создать корень дерева с нетерминалом A. Он же является листом. Обрабатывать листья дерева в порядке их создания в соответствии с п.2.

2. Пусть лист содержит псевдоцепочку β с нетерминалом B и в грамматике G' n псевдоправил, определяющих нетерминал B. Получить все псевдоцепочки δ_i, $i=1,n$, непосредственно выводимые из псевдоцепочки β. Если в дереве нет вершины, содержащей псевдоцепочку δ_i, то создать вершину (лист) с псевдоцепочкой δ_i и провести в неё дугу из обрабатываемой вершины с псевдоцепочкой β.

Алгоритм нахождения самовложенных нетерминалов грамматики G,

1. Получить грамматику G', содержащую псевдоправила, соответствующие правилам грамматики G.

2. Последовательно обрабатывать нетерминалы грамматики G' в соответствии с п.3.

3. Пусть A – обрабатываемый нетерминал. Построить дерево выводимых псевдоцепочек из нетерминала A (см. алгоритм выше). Если дерево содержит вершину с псевдоцепочкой $\alpha A \alpha$, то нетерминал A – самовложенный. (Если в процессе построения дерева на некотором шаге получена псевдоцепочка $\alpha A \alpha$, то дерево дальше не строить.)

Пример 1.

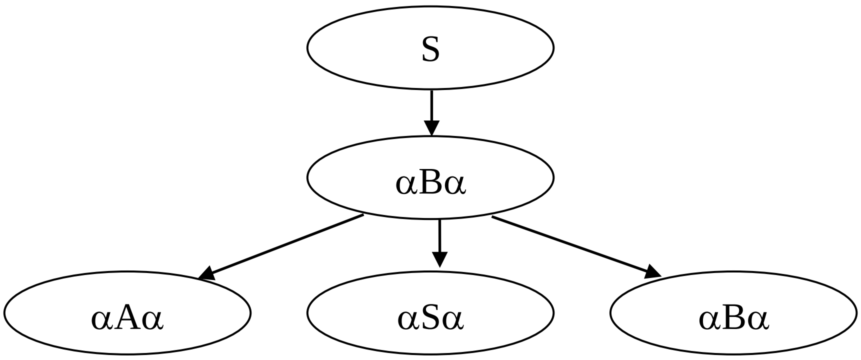
Исходная грамматика G

S→aBc
S→b
B→cAS
B→aB
A→bB
A→d

Грамматика G’

S→αBα
B→αAα
B→αS
B→αB
A→αB

Дерево выводимых псевдоцепочек из нетерминала S

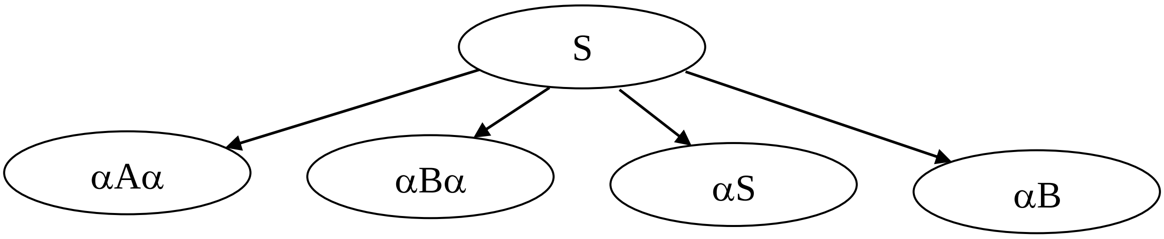


Дерево выводимых псевдоцепочек из нетерминала S содержит псевдоцепочку αSα , следовательно, нетерминал S самовложенный.

Пример 2.

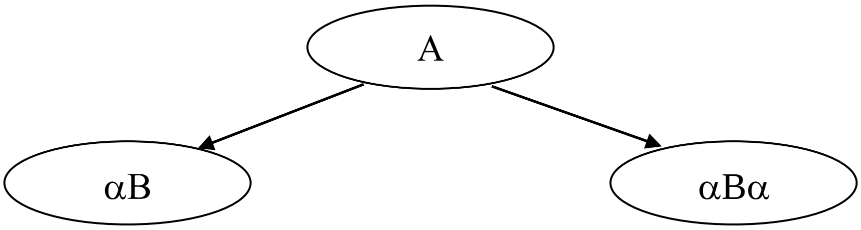
Исходная грамматика G	Грамматика G’
$S \to aABS$	$S \to \alpha A \alpha$
$S \to bB$	$S \to \alpha B \alpha$
$A \to aB$	$S \to \alpha S$
$A \to cBa$	$S \to \alpha B$
$B \to bB$	$A \to \alpha B$
$B \to d$	$A \to \alpha B \alpha$
	$B \to \alpha B$

Дерево выводимых псевдоцепочек из нетерминала S



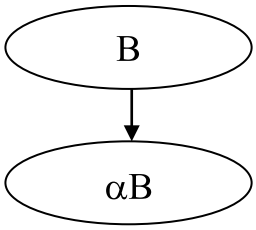
Дерево выводимых псевдоцепочек из нетерминала S не содержит псевдоцепочку $\alpha S \alpha$, следовательно, нетерминал S несамовложенный.

Дерево выводимых псевдоцепочек из нетерминала A



Дерево выводимых псевдоцепочек из нетерминала A не содержит псевдоцепочку $\alpha A \alpha$, следовательно, нетерминал A несамовложенный.

Дерево выводимых псевдоцепочек из нетерминала B



Дерево выводимых псевдоцепочек из нетерминала B не содержит псевдоцепочку $\alpha A \alpha$, следовательно, нетерминал B несамовложенный.

В грамматике G нет самовложенных нетерминалов, поэтому она порождает регулярный язык и может быть преобразована в правостороннюю.

КС-грамматика произвольного вида, имеющая самовложенные нетерминалы, порождает регулярный язык в том случае, если исключение некоторых правил приводит к отсутствию самовложенных нетерминалов и не изменяет порождаемого языка, но проверить это, в общем случае, невозможно в силу алгоритмической неразрешимости проблемы эквивалентности КС-языков.

Пример.

Грамматика G1	Грамматика G2
$A \to aAc$	$A \to b$
$A \to b$	$A \to aA$
$A \to aA$	$A \to bB$
$A \to bB$	$B \to cB$
$B \to cB$	$B \to c$
$B \to c$	

Грамматика G1 содержит самовложенный нетерминал A . Грамматика G2 получена из грамматики G1 исключением правила $A \to aAc$ и не имеет самовложенных нетерминалов. Обе грамматики порождают один и тот же регулярный язык $L=\{a^n bc^m \mid n \geq 0 \text{ и } m \geq 0\}$.

ЗАДАНИЕ

Преобразовать КС-грамматику, определяющую язык знаковых вещественных констант, в которых может отсутствовать дробная или целая чась, но не обе сразу, в правостороннюю.

S → +B
S → –B
S → B
B → Ц.Ц
B → Ц.
B → .Ц
Ц → цЦ
Ц → ц

Терминалы; {+, –, ., ц}
Пояснения {плюс, минус, точка, ц – терминал, обозначающий любую цифру}

Нетерминалы {S, B, Ц}
S – начальный нетерминал, обозначает вещественную знаковую константу
B – нетерминал, обозначает вещественную беззнаковую константу
Ц – нетерминал, обозначает целое число (непустая последовательность цифр)