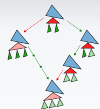


# Леммы о накачке для контекстно-свободных языков. Нормальная форма Грейбах

---

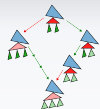
Теория формальных языков  
*2021 г.*



## Высота вывода слова в CFL

### Вопрос

Дана КС-грамматика  $G$  в CNF (н.ф. Хомского). Дерево разбора какой высоты может соответствовать непустому слову длины  $w \in L(G)$ ?



## Высота вывода слова в CFL

### Вопрос

Дана КС-грамматика  $G$  в CNF (н.ф. Хомского). Дерево разбора какой высоты может соответствовать непустому слову длины  $w \in L(G)$ ?

- Максимум:  $|w|$  (каждое нефинальное правило увеличивает длину слова хотя бы на 1);



## Высота вывода слова в CFL

### Вопрос

Дана КС-грамматика  $G$  в CNF (н.ф. Хомского). Дерево разбора какой высоты может соответствовать непустому слову длины  $w \in L(G)$ ?

- Максимум:  $|w|$  (каждое нефинальное правило увеличивает длину слова хотя бы на 1);
- Минимум:  $\lceil \log_2 w \rceil + 1$  (если вывод высоты  $k$  порождает максимум слова длины  $s$ , тогда вывод высоты  $k + 1$  породит максимум слово длины  $2 * s$ ).



## Лемма о накачке КС-языков

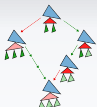
### Лемма о накачке (разрастании)

Пусть  $G$  — КС-грамматика в форме Хомского. Тогда существует  $p \in \mathbb{N}$  такое, что любое слово  $w \in L(G)$  длины не меньше  $p$  имеет представление вида  $x_1 u_1 z u_2 x_2$ , где  $|u_1 u_2| \geq 1$ ,  $|u_1 z u_2| \leq p$ , и все слова вида  $x_1 u_1^k z u_2^k x_2$  также принадлежат  $L(G)$ .



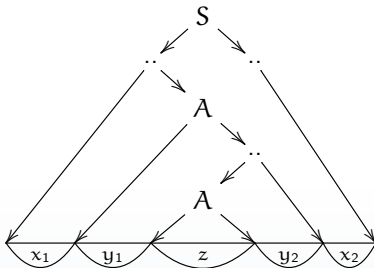
## Лемма о накачке КС-языков

Пусть в н.ф. Хомского  $G$   $n$  нетерминалов. Возьмём  $p = 2^n$ . Его вывод будет иметь минимум высоту  $n + 1 \Rightarrow$  в нём будет существовать путь, содержащий два одинаковых нетерминала  $A$ .



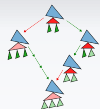
## Лемма о накачке КС-языков

Пусть в н.ф. Хомского  $G$   $n$  нетерминалов. Возьмём  $p = 2^n$ . Его вывод будет иметь минимум высоту  $n + 1 \Rightarrow$  в нём будет существовать путь, содержащий два одинаковых нетерминала  $A$ .



Выберем самые нижние два одинаковых нетерминала  $\Rightarrow$  высота поддерева от первого из них не больше  $n + 1 \Rightarrow$  длина выводимого слова  $y_1zy_2 \leq 2^n$  (т.е.  $\leq p$ ).





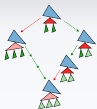
## Пример применения

### Парсинг в Python

Проанализировать язык

$$\{a^n z_1 a^n z_2 a^n | n \geq 1, |z_i|_a = 0, |z_i| \geq 1\}.$$





## Пример применения

### Парсинг в Python

Проанализировать язык

$$\{a^n z_1 a^n z_2 a^n | n \geq 1, |z_i|_a = 0, |z_i| \geq 1\}.$$

Пусть длина накачки есть  $p$ . Рассмотрим слово  $a^p b a^p b a^p$ . Заметим, что если  $y_1 z y_2 = a^i b a^j$  (где  $i$  и  $j$  могут быть равны 0), тогда  $|y_1 y_2|_b = 0$ . Действительно, иначе нулевая накачка породит слово  $a^m b a^p$ , которое не принадлежит языку.

Значит,  $y_1 = a^j$ ,  $y_2 = a^i$ . Однако слова  $a^{p+i+j} b a^p b a^p$ ,  $a^p b a^{p+i+j} b a^p$ ,  $a^p b a^p b a^{p+i+j}$ ,  $a^{p+j} b a^{p+i} b a^p$ ,  $a^p b a^{p+j} b a^{p+i}$  ни одно не принадлежат требуемому языку  $\Rightarrow$  он не контекстно-свободен.



## Накачиваемые языки

Некоторые не КС-языки тоже накачиваются, например,  
 $\{a^m b^n c^n d^n \mid m > 0\} \cup \{b^i c^j d^k\}$ .



## Накачиваемые языки

Некоторые не КС-языки тоже накачиваются, например,  
 $\{a^m b^n c^n d^n \mid m > 0\} \cup \{b^i c^j d^k\}$ .

Действительно, если слово языка содержит буквы  $a$ , тогда мы можем взять  $y_1 y_2 = a^i$ . Иначе накачку можно выбрать произвольно.



## Накачиваемые языки

Некоторые не КС-языки тоже накачиваются, например,  $\{a^m b^n c^n d^n \mid m > 0\} \cup \{b^i c^j d^k\}$ .

Действительно, если слово языка содержит буквы  $a$ , тогда мы можем взять  $y_1 y_2 = a^i$ . Иначе накачку можно выбрать произвольно.

То, что этот язык — не КС, можно понять по тому факту, что его пересечение с регулярным языком  $ab^*c^*d^*$  не контекстно-свободно.



## Лемма Огдена

### Теорема

Пусть  $L$  — КС-язык. Тогда существует такое число  $n$ , что в любом слове  $w$ ,  $|w| \geq n$ , можно отметить  $n$  или более букв так, что  $w$  представляется в виде  $x_1 y_1 z y_2 x_2$ , причём либо во всех трех из  $x_1$ ,  $y_1$ ,  $z$  есть отмеченные буквы, либо они есть во всех трех из  $z$ ,  $y_2$ ,  $x_2$ , но в слове  $y_1 z y_2$  отмечено не более  $n$  букв, и все слова вида  $x_1 y_1^k z y_2^k x_2$  также входят в  $L$ .



## Много разных лемм о накачке

Теорема Турчина плюс нормальная форма Грейбах (а также обращение н.ф. Грейбах) позволяют выводить неограниченное количество лемм о накачке. Можно отсечь произвольный префикс (суффикс) слова и накачивать то, что осталось. Можно находить несколько точек накачки.

Рассмотрим

$\{a^n b^m \mid n \neq m\} \cup \{a^n b^n \mid n \text{ — простое число}\}$ . Этот язык накачивается любыми леммами Огдена! Но множественный анализ накачек его берёт.



## Теоретико-игровая интерпретация

Достаточное условие непринадлежности языка  $L$  к КС по лемме о накачке:  
 $\forall p \exists w \in L (|w| > p \ \& \ \forall x_i, y_i, z (w = x_1 y_1 z y_2 x_2 \ \& \ |y_1 z y_2| < p \Rightarrow \exists i (x_1 y_1^i z y_2^i x_2 \notin L)))$ .

В пренексной форме этого условия кванторы образуют последовательность:  $\forall \exists \forall \exists$ . Эта последовательность задаёт правила игры, где каждый квантор  $\exists$  — ход протагониста, квантор  $\forall$  — ход антагониста. Ходы антагониста назначают неопределённые параметры. Ходы протагониста дают выбор известной вам структуры, зависящей от ходов антагониста. В случае леммы о накачке это выглядит так.

- Антагонист выбирает длину накачки  $p$ .
- Зная  $p$ , протагонист выбирает  $w$ .
- Антагонист выбирает разбиение  $w$  на пять подстрок.
- Возможно, в зависимости от этого разбиения, протагонист предъявляет  $i$ , для которого накачка не выполняется.

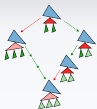


## Теоретико-игровая интерпретация

- Антагонист выбирает длину накачки  $p$ .
- Зная  $p$ , протагонист выбирает  $w$ .
- Антагонист выбирает разбиение  $w$  на пять подстрок.
- Возможно, в зависимости от этого разбиения, протагонист предъявляет  $i$ , для которого накачка не выполняется.

Иногда такая система анализа свойств, записанных в виде формул с чередующимися кванторами, также называется игрой Элоизы и Абеляра (по буквам, образующим кванторы  $\exists$  и  $\forall$ ).





## Техника применения

### Сужение перебора

Если в язык  $L$  входят под слова произвольной формы из  $\Sigma^+$ , где  $|\Sigma| > 1$ , тогда, скорее всего, потребуется пересечь  $L$  с регулярным языком, чтобы облегчить поиск свидетельства о ненакачиваемости. Пример: язык  $\{w_1 w_1 w_2 \mid |w_1|_a = |w_2|_a\}$ . Пересечение этого языка с  $ba^* bba^* bba^*$  гораздо легче поддаётся анализу, поскольку такие слова разбиваются на подходящие  $w_1$  и  $w_2$  однозначно.

- Начальная буква  $b$  вынуждает  $w_1$  содержать ровно две буквы  $b$ . Действительно, если  $|w_1|_b = 1$ , тогда второе вхождение  $w_1$  должно будет начинаться с  $b^2$ , что противоречит выбору  $w_1$ .
- Последняя буква  $b$  навязывает позицию начала  $w_2$ .



## Техника применения

### Работа с отрицанием

Если характеристическая функция  $L$  содержит предикат отрицания, связывающий две структуры неопределённого размера, в некоторых случаях это приводит к невозможности применения леммы о накачке. В других можно попробовать воспользоваться приёмом «всё включено». Поскольку мы знаем, что длина накачиваемого фрагмента  $y_1zy_2$  меньше  $p$ , то выберем  $w$  так, чтобы в нём нашлись всевозможные фрагменты такой длины, удовлетворяющие желательному свойству.



## Техника применения

Покажем, что язык  $L = \{w \mid w \neq a^{n^2} \ \& \ w \in \{a, b\}^*\}$  не является КС. Для начала заметим, что слова  $L$  содержат произвольные подслова в  $\{a, b\}^*$ , и пересечём  $L$  с  $a^*$ . Получим  $L' = \{a^k \mid k \neq n^2\}$  — если он не КС, то исходный язык также не КС.

- Антагонист выбирает  $p$ .
- Наша задача — подобрать такое  $k$ , что  $\forall p' \exists i, m(p' < p \Rightarrow k + p' * i = m^2)$ . То есть включить возможность взятия любого такого  $p'$  в наше значение  $k$  как конструктивного элемента для построения квадрата числа.
- Возьмём  $k = (p!)^2 + 1$ . Тогда при любом значении  $p'$ , меньшем  $p$ , можно взять  $i = \frac{p!}{p'} * 2$ , и получим  $k + p' * i = (p! + 1)^2$ .



## Н.ф. Хомского и левосторонний вывод

- Могут быть непродуктивные левосторонние цепочки:  
 $A \rightarrow AB \rightarrow \dots AB^n \rightarrow \dots$
- Есть гарантия роста слова при развёртке, но нет определённости, по какому префиксу.



## Нормальная форма Грейбах

### Определение

Грамматика  $G$  ( $\varepsilon \notin L(G)$ ) находится в GNF (н.ф. Грейбах)  $\Leftrightarrow$  каждое её правило имеет вид  $A_i \rightarrow a_j \alpha$ , где  $A_i \in N$ ,  $\alpha \in N^*$ ,  $a_j \in \Sigma$ .

- Левосторонний разбор по грамматике в GNF на каждом шагу переписывания порождает терминальный символ.



## Нормальная форма Грейбах

### Определение

Грамматика  $G$  ( $\varepsilon \notin L(G)$ ) находится в **GNF** (н.ф. Грейбах)  $\Leftrightarrow$  каждое её правило имеет вид  $A_i \rightarrow a_j \alpha$ , где  $A_i \in N$ ,  $\alpha \in N^*$ ,  $a_j \in \Sigma$ .

- Левосторонний разбор по грамматике в GNF на каждом шагу переписывания порождает терминальный символ.
- Для приведения к GNF нужно «вытащить из рекурсии» возможные first-терминалы, порождаемые нетерминалами грамматики.



## Нормальная форма Грейбах

### Определение

Грамматика  $G$  ( $\varepsilon \notin L(G)$ ) находится в GNF (н.ф. Грейбах)  $\Leftrightarrow$  каждое её правило имеет вид  $A_i \rightarrow a_j \alpha$ , где  $A_i \in N$ ,  $\alpha \in N^*$ ,  $a_j \in \Sigma$ .

- Левосторонний разбор по грамматике в GNF на каждом шагу переписывания порождает терминальный символ.
- Для приведения к GNF нужно «вытащить из рекурсии» возможные first-терминалы, порождаемые нетерминалами грамматики.
  - явно найти все завершающиеся цепочки вывода;



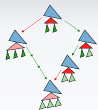
## Нормальная форма Грейбах

### Определение

Грамматика  $G$  ( $\varepsilon \notin L(G)$ ) находится в **GNF** (н.ф. Грейбах)  $\Leftrightarrow$  каждое её правило имеет вид  $A_i \rightarrow a_j \alpha$ , где  $A_i \in N$ ,  $\alpha \in N^*$ ,  $a_j \in \Sigma$ .

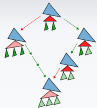
- Левосторонний разбор по грамматике в GNF на каждом шагу переписывания порождает терминальный символ.
- Для приведения к GNF нужно «вытащить из рекурсии» возможные first-терминалы, порождаемые нетерминалами грамматики.
  - явно найти все завершающиеся цепочки вывода;
  - рассмотреть язык-реверс сентенциальных форм.
- По умолчанию считаем, что к GNF приводится CNF (н.ф. Хомского).





## Первый способ приведения к GNF

- 1 Пронумеровать нетерминалы в правых частях правил в порядке их вхождения;
- 2 (по исчерпанию) Если имеется правило вида  $A_i \rightarrow B_j \beta$ , где  $j < i$ , тогда подставить вместо  $B_j$  все правые части  $\alpha_k$  правил вида  $B_j \rightarrow \alpha_k$ .
- 3 Если после этого все правила имеют вид либо  $A_i \rightarrow a\alpha$ ,  $a \in \Sigma$ , либо  $A_i \rightarrow B_j \beta$ , причём  $i < j$ , тогда GNF получается последовательной развёрткой  $B_j$ .



## Первый способ приведения к GNF

- 1 Пронумеровать нетерминалы в правых частях правил в порядке их вхождения;
- 2 (по исчерпанию) Если имеется правило вида  $A_i \rightarrow B_j \beta$ , где  $j < i$ , тогда подставить вместо  $B_j$  все правые части  $\alpha_k$  правил вида  $B_j \rightarrow \alpha_k$ .
- 3 Если после этого все правила имеют вид либо  $A_i \rightarrow a\alpha$ ,  $a \in \Sigma$ , либо  $A_i \rightarrow B_j \beta$ , причём  $i < j$ , тогда GNF получается последовательной развёрткой  $B_j$ . Существует лексикографический порядок на функциональных символах из  $N$ .



## Первый способ приведения к GNF

- 1 Пронумеровать нетерминалы в правых частях правил в порядке их вхождения;
- 2 (по исчерпанию) Если имеется правило вида  $A_i \rightarrow B_j \beta$ , где  $j < i$ , тогда подставить вместо  $B_j$  все правые части  $\alpha_k$  правил вида  $B_j \rightarrow \alpha_k$ .
- 3 Если после этого все правила имеют вид либо  $A_i \rightarrow a\alpha$ ,  $a \in \Sigma$ , либо  $A_i \rightarrow B_j \beta$ , причём  $i < j$ , тогда GNF получается последовательной развёрткой  $B_j$ . Существует лексикографический порядок на функциональных символах из  $N$ .
- 4 Если есть правила вида  $A_i \rightarrow A_i \alpha$ , тогда устраним левую рекурсию.



## Устранение левой рекурсии

- 1 Предположим, для  $A_i$  нашлось  $n$  леворекурсивных правил и  $m$  упорядоченных лексикографически:

$$A_i \rightarrow A_i \alpha_1$$

$$A_i \rightarrow \beta_1$$

...

...

$$A_i \rightarrow A_i \alpha_n$$

$$A_i \rightarrow \beta_m$$

- 2 Вводим новый нетерминал  $A'_i$  такой, что его вес меньше всех прочих, и заменяем правила на:

$$A'_i \rightarrow \alpha_1 A'_i \mid \alpha_1$$

$$A_i \rightarrow \beta_1 \mid \beta_1 A'_i$$

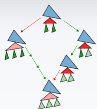
...

...

$$A'_i \rightarrow \alpha_n A'_i \mid \alpha_n$$

$$A_i \rightarrow \beta_m \mid \beta_m A'_i$$

- 3 После всех таких замен грамматика лексикографически упорядочена по левому разбору, и GNF получается последовательной левой развёрткой.

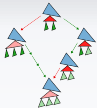


## Второй способ приведения к GNF

Алгоритм Блюма–Коха (1999).

### Неформальное описание

- Рассмотрим язык сентенциальных форм с переписыванием только по левому разбору. Он регулярен, и в конечное состояние его NFA ведут стрелки, помеченные терминалами.
- Для такого языка легко построить инверсный  $\Rightarrow$  множество терминалов-префиксов, которые может породить данный нетерминал.



## Второй способ: порождение NFA

- 1 По каждому нетерминалу  $B$  строим автомат  $M_B = \langle N_B \cup \{S_B\}, \Sigma \cup N, B_B, \{S_B\}, \delta \rangle$  ( $S_B$  — новое состояние,  $N_B$  — множество нетерминалов CFG, индексированное нетерминалом  $B$ ). Правила перехода  $\delta$ :
  - $\langle C_B, E, M \rangle \Leftrightarrow M = \{D_B \mid C \rightarrow DE \in P\};$
  - $\langle C_B, a, \{S_B\} \rangle \Leftrightarrow C \rightarrow a \in P.$
- 2 Строим реверс к  $M_B$ , получаем NFA  $M_B^R$ .
- 3 Строим грамматику  $G'_B = \langle N_B \cup \{S_B\}, \Sigma \cup N, R'_B, S_B \rangle$  для  $M_B^R$  с правилами переписывания:
  - $S_B \rightarrow aC_B \Leftrightarrow \langle S_B, a, C_B \rangle \in \delta^R$  и  $C_B \neq B_B$  либо из  $B_B$  есть стрелки в  $M_B^R$ ;
  - $S_B \rightarrow a \Leftrightarrow \langle S_B, a, B_B \rangle \in \delta^R$ ;
  - $D_B \rightarrow EC_B \Leftrightarrow \langle S_B, E, C_B \rangle \in \delta^R$  и  $C_B \neq B_B$  либо из  $B_B$  есть стрелки в  $M_B^R$ ;
  - $C_B \rightarrow E \Leftrightarrow \langle C_B, E, B_B \rangle \in \delta^R.$



## Окончание конструкции

### Построение GNf

Теперь по всем  $G'_i$  строим окончательный вариант грамматики  $G_B = \langle N_B \cup \{S_B\}, \Sigma, R_B, S_B \rangle$  с правилами:

- $S_B \rightarrow \alpha C_B, S_B \rightarrow \alpha C_B \in R'_B$ ;
- $S_B \rightarrow \alpha S_B \rightarrow \alpha \in R'_B$ ;
- $D_B \rightarrow \alpha C_B \Leftrightarrow D_B \rightarrow E C_B \in R'_B \ \& \ S_E \rightarrow \alpha$  (по всем таким  $\alpha$  и  $E$ );
- $D_B \rightarrow \alpha \Leftrightarrow D_B \rightarrow E \in R'_B \ \& \ S_E \rightarrow \alpha$  (по всем таким  $\alpha$  и  $E$ ).

Грамматика  $\bigcup_{i \in N} G_i$  со стартовым символом  $S_S$  — это  
искомая GNf для исходной грамматики  $G$ .



## Пример преобразования грамматики по Блему–Коху

Привести к GNF грамматику некорректных сумм двоичных чисел (почему некорректных?)

$$S \rightarrow S + S \mid D \quad D \rightarrow D0 \mid D1 \mid 1 \mid (S)$$





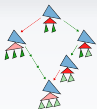
## Пример преобразования грамматики по Блему–Коху

Привести к GNF грамматику некорректных сумм двоичных чисел (почему некорректных?)

$$S \rightarrow S + S \mid D \quad D \rightarrow D0 \mid D1 \mid 1 \mid (S)$$

Сначала избавляемся от цепного правила  $S \rightarrow D$ . Потом строим порождающую структуру  $A_V$  сентенциальных форм по левостороннему разбору с финальным состоянием  $N_V$  и стартовым  $V_V$ . Каждому нетерминалу  $V$  соответствует своя структура.

$$\begin{array}{llllll} \text{Для } A_S : & S_S \xrightarrow{+S} S_S & S_S \xrightarrow{0} D_S & S_S \xrightarrow{1} D_S & S_S \xrightarrow{1} N_S \\ & S_S \xrightarrow{(S)} N_S & D_S \xrightarrow{0} D_S & D_S \xrightarrow{1} D_S & D_S \xrightarrow{1} N_S & D_S \xrightarrow{(S)} N_S \end{array}$$



## Пример преобразования грамматики по Блему–Коху

Привести к GNF грамматику некорректных сумм двоичных чисел (почему некорректных?)

$$S \rightarrow S + S \mid D \quad D \rightarrow D0 \mid D1 \mid 1 \mid (S)$$

Сначала избавляемся от цепного правила  $S \rightarrow D$ . Потом строим порождающую структуру  $A_V$  сентенциальных форм по левостороннему разбору с финальным состоянием  $N_V$  и стартовым  $V_V$ . Каждому нетерминалу  $V$  соответствует своя структура.

$$\begin{array}{llllll} \text{Для } A_S : & S_S \xrightarrow{+S} S_S & S_S \xrightarrow{0} D_S & S_S \xrightarrow{1} D_S & S_S \xrightarrow{1} N_S \\ & S_S \xrightarrow{(S)} N_S & D_S \xrightarrow{0} D_S & D_S \xrightarrow{1} D_S & D_S \xrightarrow{1} N_S & D_S \xrightarrow{(S)} N_S \end{array}$$

$$\text{Для } A_D : \quad D_D \xrightarrow{0} D_D \quad D_D \xrightarrow{1} D_D \quad D_D \xrightarrow{1} N_D \quad D_D \xrightarrow{(S)} N_D$$



## Пример преобразования грамматики по Блуму–Коху

Превращаем структуры в праволинейные (меняя местами нетерминалы левых и правых частей правил и стартовые состояния с финальными):

$$\begin{array}{llllll} \text{Для } A_S : & S_S \xrightarrow{+S} S_S & D_S \xrightarrow{0} S_S & D_S \xrightarrow{1} D_S & N_S \xrightarrow{1} S_S \\ N_S \xrightarrow{(S)} S_S & D_S \xrightarrow{0} D_S & D_S \xrightarrow{1} D_S & N_S \xrightarrow{1} D_S & N_S \xrightarrow{(S)} D_S \end{array}$$

$$\text{Для } A_D : \quad D_D \xrightarrow{0} D_D \quad D_D \xrightarrow{1} D_D \quad N_D \xrightarrow{1} D_D \quad N_D \xrightarrow{(S)} D_D$$

Извлекаем праволинейные (почти) грамматики. В правой части правила может не быть нетерминалов, если там стоял нетерминал  $V_V$ .

$$\begin{array}{lllll}
 \text{q-RLG } G_S : & S_S \rightarrow +SS_S & D_S \rightarrow 0S_S & D_S \rightarrow 1D_S & N_S \rightarrow 1S_S \\
 & N_S \rightarrow (S)S_S & D_S \rightarrow 0D_S & D_S \rightarrow 1D_S & N_S \rightarrow (S)D_S \\
 & S_S \rightarrow +S & D_S \rightarrow 0 & D_S \rightarrow 1 & N_S \rightarrow (S)
 \end{array}$$

Извлекаем праволинейные (почти) грамматики. В правой части правила может не быть нетерминалов, если там стоял нетерминал  $V_V$ .

$$\begin{array}{lllll} \text{q-RLG } G_S : & S_S \rightarrow +SS_S & D_S \rightarrow 0S_S & D_S \rightarrow 1D_S & N_S \rightarrow 1S_S \\ & N_S \rightarrow (S)S_S & D_S \rightarrow 0D_S & D_S \rightarrow 1D_S & N_S \rightarrow 1D_S & N_S \rightarrow (S)D_S \\ & S_S \rightarrow +S & D_S \rightarrow 0 & D_S \rightarrow 1 & N_S \rightarrow 1 & N_S \rightarrow (S) \end{array}$$

$$\begin{array}{lllll} \text{q-RLG } G_D : & D_D \rightarrow 0D_D & D_D \rightarrow 1D_D & N_D \rightarrow 1D_D & N_D \rightarrow (S)D_D \\ & D_D \rightarrow 0 & D_D \rightarrow 1 & N_D \rightarrow 1 & N_D \rightarrow (S) \end{array}$$

Извлекаем праволинейные (почти) грамматики. В правой части правила может не быть нетерминалов, если там стоял нетерминал  $V_V$ .

$$\begin{array}{lllll} \text{q-RLG } G_S : & S_S \rightarrow +SS_S & D_S \rightarrow 0S_S & D_S \rightarrow 1D_S & N_S \rightarrow 1S_S \\ & N_S \rightarrow (S)S_S & D_S \rightarrow 0D_S & D_S \rightarrow 1D_S & N_S \rightarrow (S)D_S \\ & S_S \rightarrow +S & D_S \rightarrow 0 & D_S \rightarrow 1 & N_S \rightarrow 1 \\ & & & N_S \rightarrow 1 & N_S \rightarrow (S) \end{array}$$

$$\begin{array}{lllll} \text{q-RLG } G_D : & D_D \rightarrow 0D_D & D_D \rightarrow 1D_D & N_D \rightarrow 1D_D & N_D \rightarrow (S)D_D \\ & D_D \rightarrow 0 & D_D \rightarrow 1 & N_D \rightarrow 1 & N_D \rightarrow (S) \end{array}$$

Заменяем неразмеченные нетерминальные символы  $V$  исходной грамматики на  $N_V$ . В данном случае нет правил, в которых неразмеченные нетерминалы стояли бы первыми в правых частях, поэтому достаточно просто заменить их на  $N_V$ . Иначе пришлось бы заменять их на все возможные правые части  $\alpha$  правил вида  $N_V \rightarrow \alpha$ . Стартовый символ —  $N_S$ . GNF почти построена!

Извлекаем праволинейные (почти) грамматики. В правой части правила может не быть нетерминалов, если там стоял нетерминал  $V_V$ .

Заменяем неразмеченные нетерминальные символы  $V$  исходной грамматики на  $N_V$ . В данном случае нет правил, в которых неразмеченные нетерминалы стояли бы первыми в правых частях, поэтому достаточно просто заменить их на  $N_V$ . Иначе пришлось бы заменять их на все возможные правые части  $\alpha$  правил вида  $N_V \rightarrow \alpha$ . Стартовый символ —  $N_S$ . GNF почти построена!

q-GNF для  $G$  :

$S_S \rightarrow +N_S S_S$	$D_S \rightarrow 0S_S$	$D_S \rightarrow 1D_S$	$N_S \rightarrow 1S_S$	
$N_S \rightarrow (N_S)S_S$	$D_S \rightarrow 0D_S$	$D_S \rightarrow 1D_S$	$N_S \rightarrow 1D_S$	$N_S \rightarrow (N_S)D_S$
$S_S \rightarrow +N_S$	$D_S \rightarrow 0$	$D_S \rightarrow 1$	$N_S \rightarrow 1$	$N_S \rightarrow (N_S)$

Извлекаем праволинейные (почти) грамматики. В правой части правила может не быть нетерминалов, если там стоял нетерминал  $V_V$ .

Заменяем неразмеченные нетерминальные символы  $V$  исходной грамматики на  $N_V$ . В данном случае нет правил, в которых неразмеченные нетерминалы стояли бы первыми в правых частях, поэтому достаточно просто заменить их на  $N_V$ . Иначе пришлось бы заменять их на все возможные правые части  $\alpha$  правил вида  $N_V \rightarrow \alpha$ . Стартовый символ —  $N_S$ . GNF почти построена!

q-GNF для  $G$  :

$S_S \rightarrow +N_S S_S$	$D_S \rightarrow 0S_S$	$D_S \rightarrow 1D_S$	$N_S \rightarrow 1S_S$	
$N_S \rightarrow (N_S)S_S$	$D_S \rightarrow 0D_S$	$D_S \rightarrow 1D_S$	$N_S \rightarrow 1D_S$	$N_S \rightarrow (N_S)D_S$
$S_S \rightarrow +N_S$	$D_S \rightarrow 0$	$D_S \rightarrow 1$	$N_S \rightarrow 1$	$N_S \rightarrow (N_S)$

Осталось обернуть в delay-нетерминалы терминальные символы правых частей правил, кроме первого. Здесь это символ  $)$ .