

# ТРЯП. Домашнее задание № 8

Шарапов Денис, Б05-005

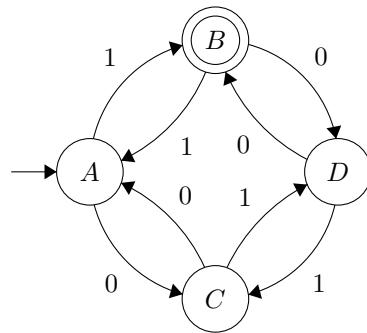
## Задача 1

$L$  — язык, состоящий из всех слов в алфавите  $\{0, 1\}$ , которые содержат чётное число нулей и нечётное число единиц. Выполнить следующие задания:

1. Построить эквивалентную праволинейную грамматику. Будет ли она однозначной?
2. Построить регулярное выражение для языка  $L^R$ .

**Решение.**

1. Построим автомат, распознающий язык всех слов, которые содержат чётное число нулей. Далее построим автомат, распознающий язык всех слов, которые содержат нечётное число единиц. После чего пересечём эти автоматы и получим следующий автомат (задача 5 из домашнего задания № 2):



Теперь воспользуемся алгоритмом перевода НКА  $\rightarrow$  ПГ. Пусть  $G$  — искомая праволинейная грамматика. Обозначим  $G = (N, T, P, S)$ , где:

- $N = \{A, B, C, D\}$  — множество нетерминалов;
- $S = A$  — начальный символ грамматики;
- $T = \{0, 1\}$  — алфавит;
- $P$  — множество правил вывода.

Алгоритм:

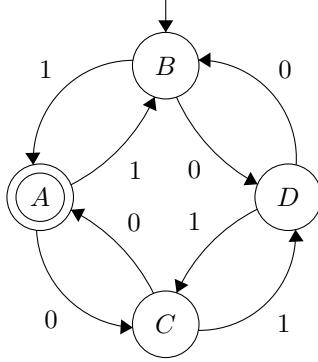
- (a)  $(A \rightarrow \sigma B) \in P$ , если  $\delta(A, \sigma) = \{B\}$ ,  $\sigma \in T$ ;
- (b)  $(A \rightarrow \sigma) \in P$ , если  $\delta(A, \sigma) = \{B\}$ ,  $\sigma \in T$  и  $B \in F$  — принимающее;
- (c)  $(S \rightarrow \varepsilon) \in P$ , если  $q_0 \in F$ .

Тогда грамматика задаётся правилами:

$$\begin{aligned} A &\rightarrow 0C \mid 1, \\ B &\rightarrow 0D \mid 1A, \\ C &\rightarrow 0A \mid 1D, \\ D &\rightarrow 0 \mid 1C. \end{aligned}$$

Алгоритм построения грамматики строит  $G$  буквально по переходам в автомате. Заметим, что автомат — детерминированный конечный, т. е. в нём каждый переход определён однозначно (с помощью перехода можно получить конкретное слово и только его). Поэтому данная грамматика является однозначной.

2. Построим РВ для языка  $L^R$ . Для этого воспользуемся алгоритмом перевода автомата, распознающего язык  $L$ , в автомат, распознающий язык  $L^R$ :



Заметим, что  $L = L^R$ .

Составляем систему:

$$\begin{cases} R_A = 0R_C + 1R_B + \varepsilon, \\ R_B = 0R_D + 1R_A, \\ R_C = 0R_A + 1R_D, \\ R_D = 0R_B + 1R_C. \end{cases} \quad \begin{cases} R_A = 0(0R_A + 1R_D) + 1(0R_D + 1R_A) + \varepsilon, \\ R_B = 0R_D + 1R_A, \\ R_C = 0R_A + 1R_D, \\ R_D = 0R_B + 1R_C. \end{cases}$$

$$\begin{cases} R_A = (00 + 11)^*((01 + 10)R_D + \varepsilon), \\ R_B = 0R_D + 1R_A, \\ R_C = 0R_A + 1R_D, \\ R_D = 0R_B + 1R_C. \end{cases} \quad \begin{cases} R_A = (00 + 11)^*((01 + 10)R_D + \varepsilon), \\ R_B = 0R_D + 1R_A, \\ R_C = 0R_A + 1R_D, \\ R_D = 0(0R_D + 1R_A) + 1(0R_A + 1R_D). \end{cases}$$

$$\begin{cases} R_A = (00 + 11)^*((01 + 10)R_D + \varepsilon), \\ R_B = 0R_D + 1R_A, \\ R_C = 0R_A + 1R_D, \\ R_D = (00 + 11 + (01 + 10)(00 + 11)^*(01 + 10))^*(01(00 + 11)^* + 10(00 + 11)^*). \end{cases}$$

Откуда и получаем искомое РВ. □

## Задача 2

Покажите индукцией по длине слова, что КС-грамматика с правилами

$$S \rightarrow SS \mid aSb \mid bSa \mid \varepsilon$$

порождает язык всех слов с равным числом символов  $a$  и  $b$ .

**Решение.**

Пусть  $L$  — язык всех слов с равным числом символов  $a$  и  $b$ ,  $L(G)$  — язык, порождаемый грамматикой  $G$  из условия.

**Утверждение.**  $L(G) \subseteq L$ .

**Доказательство.** Выберем произвольное слово  $w \in L(G)$ . Докажем по индукции числа  $n$  букв в слове  $w$ .

База индукции:

- 1)  $n = 0 : w = \varepsilon \in L$ , выведенное из правила  $S \rightarrow \varepsilon$
- 2)  $n = 2 : w = ab \in L$  или  $w = ba \in L$ , выведенные цепочками  $S \rightarrow aSb \rightarrow ab$  или  $S \rightarrow bSa \rightarrow ba$  соответственно.

Далее будем использовать натуральную константу  $k \geq 2$ .

**Случай 1.** Индукция для правила вывода  $S \rightarrow aSb$ .

Предположение индукции  $n = 2k$ :

Пусть существует цепочка, выводящая слово  $w \in L$  длиной  $|w| = n$  применением правила  $S \rightarrow \varepsilon$ .

Переход индукции  $n = 2k + 2$ :

Применим  $k$  раз правило вывода  $S \rightarrow aSb$ . По предположению индукции найдётся цепочка (просто предъявим её) такая, что из неё можно вывести слово длиной  $n = 2k$ :

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow \dots \Rightarrow a^k Sb^k.$$

Применяем выбранное правило ещё раз:

$$S \Rightarrow^* a^{k+1} Sb^{k+1} \Rightarrow a^{k+1} b^{k+1} \in L.$$

**Случай 2.** Индукция для правила вывода  $S \rightarrow bSa$  — аналогично.

**Случай 3.** Индукция для комбинации правил вывода  $S \rightarrow aSb$  и  $S \rightarrow bSa$ .

Предположение индукции  $n = 2k$ :

Пусть существует цепочка, выводящая слово  $w \in L$  длиной  $|w| = 2k$  применением правила  $S \rightarrow \varepsilon$ .

Переход индукции  $n = 2k + 2$ :

Воспользуемся уже доказанными утверждениями (случаи 1 и 2). Пусть без ограничения общности правила были применены в таком порядке (для другого порядка рассуждения аналогичны): сначала  $m_1 \geq 1$  раз применили правило  $S \rightarrow aSb$ , затем  $m_2 \geq 1$  раз применили правило  $S \rightarrow bSa$ , после чего  $m_3 \geq 0$  раз применили правило  $S \rightarrow aSb$  и т. д. по индукции числа чередований правил. Случай 1 утверждает, что после применения  $m_1 \geq 1$  раз первого правила выводится слово  $v \in L$ . Далее второе правило  $S \rightarrow bSa$  изменит количество символов  $a$  и  $b$  в слове  $v$  (текущее на данный момент) на одинаковое количество. Далее применяем случай 2 и по индукции числа чередований правил получим слово  $w \in L$ .

**Случай 4.** Индукция для правила вывода  $S \rightarrow SS$ .

Предположение индукции  $n = 2^k$ :

Пусть существует цепочка, выводящая слово  $w \in L$  длиной  $|w| = 2^k$  применением правила  $S \rightarrow aSb \mid bSa \mid \varepsilon$ .

Переход индукции  $n = 2^{k+1}$ :

Применим  $k$  раз правило вывода  $S \rightarrow SS$ :

$$S \Rightarrow SS \Rightarrow SSSS \Rightarrow^* S^{2^k}.$$

Делаем ещё один переход:

$$S \Rightarrow^* S^{2^{k+1}}.$$

По предложению индукции найдётся цепочка (предъявим её) выводящая слово  $w \in L$  длиной  $|w|_n$ : используя доказанные случаи 1, 2, 3, 4, подставляем всевозможные комбинации правил 1 и 2 в вывод

$$S \Rightarrow^* S^{2^k}$$

и получаем слово длиной  $2^k$  при последней замене  $S \rightarrow \varepsilon$ . Теперь представим, что последней замены не было. Применяем правило  $S \rightarrow SS$ , после чего применяем комбинации правил 1 и 2, затем применяем правило  $S \rightarrow \varepsilon$  и получаем слово длиной  $n = 2^{k+1}$ , т. к. при замене каждого терминального символа количество букв в слове изменяется ровно на единицу («удаляется»  $S$ , подставляется  $ab$  или  $ba$ ), а при последнем переходе индукции было использовано правило  $S \rightarrow SS$ , которое дало четное число терминалов  $S$ .

**Случай 5.** Произвольная комбинация правил  $S \rightarrow SS \mid aSb \mid bSa \mid \varepsilon$ . Используются доказанные случаи 1, 2, 3, 4.  $\square$

**Утверждение.**  $L \subseteq L(G)$ .

**Доказательство.** Требуется доказать утверждение

$$\forall w \in L \hookrightarrow w \in L(G).$$

Докажем по индукции числа  $n$  букв в слове в  $w \in L$ .

База индукции:

- 1)  $n = 0 : w = \varepsilon \in L(G)$ , выведенное из правила  $S \rightarrow \varepsilon$
- 2)  $n = 2 : w = ab \in L(G)$  или  $w = ba \in L(G)$ , выведенные цепочками  $S \rightarrow aSb \rightarrow ab$  или  $S \rightarrow bSa \rightarrow ba$  соответственно.

Далее будем использовать натуральную константу  $k \geq 2$ .

Предположение индукции  $n = 2k$ : пусть верно утверждение

$$\forall w \in L : |w| = 2k \hookrightarrow w \in L(G).$$

Переход индукции  $n = 2k + 2$ :

Используем утверждение индукции. Если произвольное слово  $w \in L$  длиной  $|w| = 2k$  лежит в языке  $L(G)$ , то его можно вывести последовательным применением правил. Пусть в цепочке вывода было сделано  $m - 1$  шагов так, что если сделать ещё один шаг  $(m - 1) \rightarrow m$ , то можно вывести слово  $w$ . Тогда на шаге  $m$  применим одно из правил вывода (кроме  $S \rightarrow \varepsilon$ ). Т. к. мы рассматриваем произвольное слово  $w \in L$ , то будут учтены всевозможные комбинации подстановок правил (потому что в предположении индукции можно вывести произвольное слово из  $L$  длины  $2k$ ). После применения одного из правил и дальнейшего применения правила  $S \rightarrow \varepsilon$  получим слово длиной  $n = 2k + 2, 2k + 4, 2k + 6, 2k + 8 \dots$  в зависимости от предыстории (например, в выводе была длинная последовательность  $SS$ , рассмотренная в прошлом утверждении). Используя доказанные случаи прошлого утверждения, получаем, что произвольное слово, выбранное в предположении индукции, не могло покинуть множество  $L$  после применения правил. Переход индукции доказан.  $\square$

Замечание:

*Утверждение, данное в условии задачи, удобнее доказывать следующим образом: доказать сначала по индукции (того, что удобнее выбрать) включение  $L \subseteq L(G)$ , а затем, используя уже доказанное утверждение, по контрапозиции доказать обратное включение.*

### Задача 3

1. Показать, что язык палиндромов в произвольном алфавите является КС-языком.
2. Показать, что дополнение к языку палиндромов также является КС-языком.

### Решение.

1. Покажем, что язык палиндромов в произвольном алфавите является КС-языком. Для этого достаточно привести грамматику  $G = (N, T, P, S)$  в качестве примера. Пусть

- $N = \{S\}$  — множество нетерминалов;
- $T = \{a_i\}_{i=1}^I$  — конечный заданный алфавит;
- $P = \{S \rightarrow a_i, S \rightarrow a_i Sa_i S \rightarrow \varepsilon\} \quad \forall i \in \overline{1, I}$ .

Тогда  $G$  порождает язык PAL.

Следующие утверждения будем доказывать для  $I = 2$  (для краткости). Тогда грамматика  $G$  принимает вид

- $N = \{S\}$ ;
- $T = \{a, b\}$ ;
- $P : S \rightarrow aSa \mid bSb \mid a \mid b \mid \varepsilon$ .

**Утверждение.**  $L(G) \subseteq \text{PAL}$ .

**Доказательство.** Докажем по индукции числа  $n$  букв в произвольно выбранном слове  $w \in L(G)$ .

База индукции:

- 1)  $n = 0: w = \varepsilon \in \text{PAL}$ ;
- 2)  $n = 1: w = a \in \text{PAL}$  или  $w = b \in \text{PAL}$ ;
- 3)  $n = 2: w = aa \in \text{PAL}$  или  $w = bb \in \text{PAL}$ ;

Далее используем натуральную константу  $k > 2$ .

Предположение индукции  $n = k$ :

Пусть верно утверждение

$$\forall w \in L(G) : |w| = k \hookrightarrow w \in \text{PAL}.$$

Переход индукции  $n = k + 1$ :

Рассмотрим произвольное слово  $w \in L(G)$  длиной  $|w| = k$ . Пусть без ограничения общности на предпоследнем шаге вывод слова  $w$  имеет вид

$$S \Rightarrow^* ava,$$

где  $v$  содержит в себе последовательность применения правил  $S \rightarrow aSa$  или  $S \rightarrow bSb$ . Далее применяется одно из правил, не содержащих справа терминала и по предположению индукции получается слово  $w \in \text{PAL}$ . Представим, что последнего перехода не было. Из рассматриваемого предпоследнего шага делаем переход по одному из правил, содержащих справа терминал. Затем делаем переход по правилу, не содержащему справа терминала ( $a$  или  $b$ ), и в силу того, что по предположению индукции на предыдущем шаге был получен палиндром, переход по данному правилу выведет слово  $u \in \text{PAL}$ .

Переход индукции  $n = k + 2$ :

Аналогичные рассуждения, но только последние два перехода имеют вид  $S \rightarrow aSa \rightarrow aa$ .  $\square$

**Утверждение.**  $\text{PAL} \subseteq L(G)$ .

**Доказательство.** Приведём конструктивное доказательство. Построим грамматики  $G'$  и  $G''$  по произвольным словам из языка PAL. Для этого рассмотрим два случая.

**Случай 1.** Рассмотрим произвольное слово  $w \in \text{PAL}$  нечётной длины. Тогда будем действовать следующему алгоритму:

- (a) Смотрим на первый и последний символы слова  $w$  (они совпадают) и добавляем их в определенный «класс», содержащий терминальный символ (например, слово  $abba$  добавляем в класс  $aSa$ , затем  $bbb$  в класс  $bSb$  и т. д.).
- (b) Повторяем процедуру, пока не получим на конце единственный символ (например,  $abba - bbb - b$ ).

Получили некоторый «разбор» слова  $w$ , который соответствует правилам

$$S \rightarrow aSa \mid bSb \mid a \mid b$$

грамматики  $G'$ .

**Случай 2.** Рассмотрим произвольное слово  $w \in \text{PAL}$  чётной длины. Строим аналогичный алгоритм, как в случае 1, но только в пункте (b) получим на конце пустое слово. Тогда правила грамматики  $G''$  имеют вид

$$S \rightarrow aSa \mid bSb \mid \varepsilon.$$

«Объединяя»  $G'$  и  $G''$ , получим грамматику  $G$ . Утверждение доказано.  $\square$

Осталось заменить в рассуждениях символы  $a$  и  $b$  на символы  $a_i$  и получить утверждение для произвольного конечного алфавита.

2. Пусть  $R$  — дополнение языка палиндромов. Приведём КС-грамматику  $G$  для произвольного конечного алфавита  $T$ :

- $N = \{S, A\}$ ;
- $T = \{a_i\}_{i=2}^I$ ;
- $S$  — начальный терминал;
- $P$  :

$$\begin{aligned} S &\rightarrow a_i Sa_i \mid a_i Ba_j \quad \forall i, j \in \overline{2, I} : i \neq j, \\ B &\rightarrow a_i Ba_j \mid a_i \mid \varepsilon \quad \forall i, j \in \overline{2, I} : i \neq j. \end{aligned}$$

**Утверждение.**  $L(G) \subseteq R$ .

**Доказательство.** Рассмотрим вывод произвольного слова  $w \in L(G)$ . Слово  $w$  на концах имеет либо разные символы (правило  $S \rightarrow a_i Ba_j$ ), либо одинаковые (правило  $S \rightarrow a_i Sa_i$ ), но в выводе обязательно встретится последовательность символов, которая читается с конца и с начала по-разному, т. к. если слово  $w$  было выведено, то существует конечная последовательность применения правил вывода (цепочка), следовательно должны применяться правила  $B \rightarrow a_i Ba_j \mid a_i \mid \varepsilon$ , которые и нарушают палиндромность.  $\square$

**Утверждение.**  $R \subseteq L(G)$ .

**Доказательство.** Рассмотрим произвольное слово  $w \in R$ . Докажем по индукции  $n$  длины вывода слова  $w$ .

База индукции  $n = 2$ :

$$w = a_i a_i a_j \in L(G) \text{ или } a_i a_j \in L(G), \text{ где } i \neq j$$

Предположение индукции  $n = k$  ( $k > 2$  — натуральное):

Пусть существует вывод произвольного слова  $v \in R$  из группы слов одинаковой длины за  $k$  шагов.

Переход индукции  $n = k + 1$ :

Докажем, что слово  $w$ , выведенное за  $k + 1$  шагов, принадлежит языку  $R$ . Возможно два случая.

**Случай 1.** Слово  $w$  может иметь вид  $a_iua_i$ , где  $u \in R$ . Тогда по предположению индукции существует вывод слова  $v \in R$  за  $k$  шагов:

$$S \Rightarrow^k v,$$

следовательно, возьмём вывод  $v$ , сделаем ещё два шага и получим вывод слова  $w$  (без ограничения общности была выбрана именно такая последовательность):

$$S \Rightarrow^{k-1} xBy \Rightarrow xa_iBa_jy \Rightarrow xa_ia_jy = w \in R,$$

где  $x, y$  — некоторые слова, получившиеся при выводе, удовлетворяющие условиям случая 1.

**Замечание.** При выводе было известно, что после  $k - 1$  шага при подстановки вместо нетерминала  $B$  любого символа (отличного от нетерминального) получалось слово  $v \in R$ . Поэтому на  $k$  шаге вместо нетерминала  $B$  было подставлено правило, содержащее терминал  $B$ , а дальше был подставлен произвольный символ (отличный от терминального).

**Случай 2.** Слово  $w$  может иметь вид  $a_iua_j$ , где  $u$  — слово, получившееся при выводе. Аналогично случаю 1 возьмём вывод слова  $v$  из предложения индукции, затем сделаем ещё два шага и получим искомое слово  $w \in R$ .

Таким образом, утверждение доказано.  $\square$

## Задача 4

Покажите, что дополнение языка  $U = \{a^n b^n c^n \mid n = 0, 1, 2, \dots\}$  является КС-языком.

## Задача 5

Построить ДМП-автомат, распознающий язык всех слов, содержащих одинаковое количество символов  $a$  и  $b$ .

**Решение.**

В задаче 2 было доказано, что КС-грамматика  $G$  с правилами

$$S \rightarrow SS \mid aSb \mid bSa \mid \varepsilon$$

порождает язык  $L$  всех слов, содержащих одинаковое количество символов  $a$  и  $b$ . Поэтому воспользуемся алгоритмом перевода  $G \rightarrow M$ , где  $M$  — МП автомат, допускающий язык  $L$  опустошением автомата.

Описание грамматики  $G = \langle N, T, P, S \rangle$ :

- $N = \{S\}$ ;
- $T = \{a, b\}$ ;
- $S$  — аксиома грамматики;
- $P : S \rightarrow SS \mid aSb \mid bSa \mid \varepsilon$

Описание МП-автомата  $M = \langle Q, T, \Gamma, \delta, q_0, Z_0, F \rangle$ :

- $Q = \{q\}$ ;
- $T = \{a, b\}$ ;
- $\Gamma = N \cup T$ ;

- $q_0 = q$ ;
- $Z_0 = S$ ;
- $F = \emptyset$ ;

Описание функции  $\delta : Q \times (T \cup \{\varepsilon\}) \times \Gamma \rightarrow 2^{Q \times \Gamma^*}$ :

$$(S \rightarrow SS) \in P \Rightarrow (q, SS) \in \delta(q, \varepsilon, S),$$

$$(S \rightarrow aSb) \in P \Rightarrow (q, asb) \in \delta(q, \varepsilon, S),$$

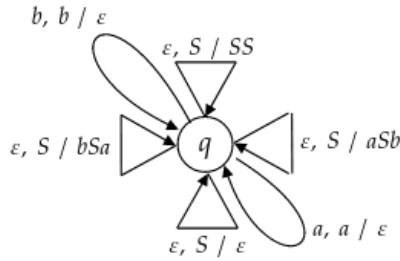
$$(S \rightarrow bSa) \in P \Rightarrow (q, bSa) \in \delta(q, \varepsilon, S),$$

$$(S \rightarrow \varepsilon) \in P \Rightarrow (q, \varepsilon) \in \delta(q, \varepsilon, S),$$

$$\delta(q, a, a) = \{(q, \varepsilon)\},$$

$$\delta(q, b, b) = \{(q, \varepsilon)\}.$$

Диаграмма:



Теперь воспользуемся алгоритмом перевода между МП-автоматами: допускающий язык по пустому стеку  $M \rightarrow$  допускающий язык по принимающему состоянию  $M'$ .

Описание автомата  $M' = \langle Q', T, \Gamma', \delta', q'_0, Z'_0, F \rangle$ :

- $Q' = Q \cup \{q'_0, q_f\}$ ;
- $\Gamma' = \Gamma \cup \{Z'_0\}$ ;
- $F = \{q_f\}$ ;

Описание функции  $\delta'$ :

$$\delta'(q'_0, \varepsilon, Z'_0) = \{(q, Z_0 Z'_0)\};$$

$$\delta'(q, \sigma, Z) = \delta(q, \sigma, Z) \quad \forall q \in Q, \forall \sigma \in T \cup \{\varepsilon\}, \forall Z \in \Gamma;$$

$$(q_f, \varepsilon) \in \delta'(q, \varepsilon, Z'_0) \quad \forall q \in Q.$$

Теперь, опираясь на работу приведенного МП-автомата  $M'$ , приведём построение (и параллельно с ним конструктивное доказательство) построение ДМП-автомата  $\mathcal{M}$ , распознающего язык из условия.

Требуется с помощью стека проверить принадлежит ли слово  $w$  языку  $L$ , т. е. в нём количество символов  $a$  должно быть равным количеству символов  $b$ . Начнём обрабатывать произвольное слово  $w$ . Положим на вершину стека нетерминальный символ  $Z_0$ . Если встретим какой-нибудь символ  $s_1$ , то положим его в стек. Далее на вход поступает следующий символ  $s_2$ , и если он не совпадает с символом  $s_1$ , то сдвигаем головку автомата на следующий символ  $s_3$  и удаляем из стека  $s_2$ . Если же  $s_2 == s_1$ , то добавляем  $s_2$  в стек. Продолжаем процедуру. В результате, если количество символов  $a$  равно количеству символов  $b$ , то в стеке останется только нетерминальный символ  $Z_0$ . Получившийся автомат по построению принимает слова из языка  $L$  и только их.

Диаграмма  $M'$ :

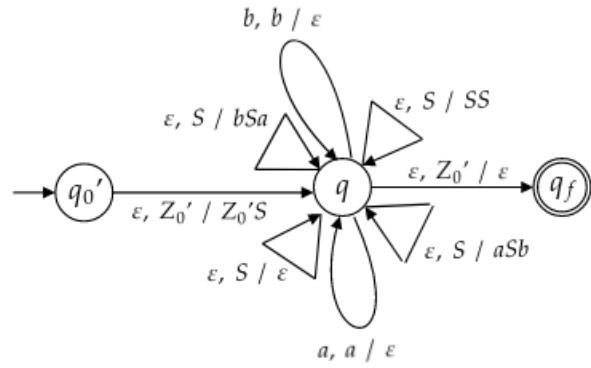
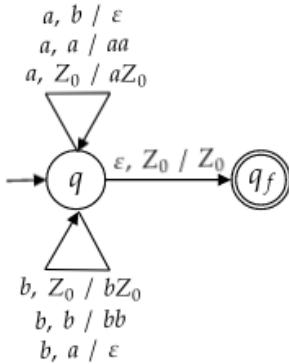


Диаграмма  $\mathcal{M}$ :



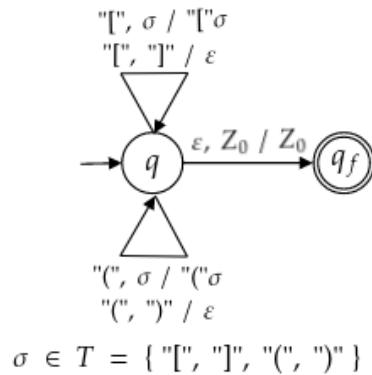
□

## Задача 6

Построить ДМП-автомат, распознающий язык  $D_2$ , порожденный грамматикой:

$$S \rightarrow SS \mid (S) \mid [S] \mid \varepsilon.$$

**Решение.**



$$\sigma \in T = \{ "[", "]", "(", ")" \}$$

□

Приведём конструктивное доказательство.

Данная грамматика задаёт язык слов, являющихся правильными скобочными последовательностями. Поэтому воспользуемся алгоритмом из курса «Основные алгоритмы»:

*Каждый раз, когда встречаем открывающую скобку, добавляем её в стек. Когда встречаем закрывающую скобку, проверяем верхушку стека: если в ней лежит открывающая скобка того же типа («{» и «}» одного типа), то снимаем открывающую с вершины и переводим головку автомата на следующий символ; если в вершине стека лежит закрывающая скобка, то выводим ошибку (автомат не примет такое слово). Продолжаем процедуру.*

Теперь модифицируем алгоритм, переведя его в ДМП-автомат  $\mathcal{M}$ . На первом шаге добавим терминальный символ  $Z_0$ . При считывании очередного символа  $s_k$  смотрим на вершину, если там лежит закрывающая скобка того же типа, то снимаем её с вершины и переводим головку на символ  $s_{k+1}$ . Если же там другой символ  $\sigma$ , то снимаем его и добавляем в вершину последовательность  $s_k\sigma$  (теперь на вершине лежит  $s_k$ ). Продолжаем действия. Если слово действительно является правильной скобочной последовательностью, то после его обработки в стеке останется только терминальный символ  $Z_0$ . Остается сделать последний переход в принимающее состояние.  $\square$