Задание 7

Коновалов Андрей, 074

0	1	2	3	4	5	6	σ

Задача 2

Алгоритм построения грамматики следующий. Модифицируем D так, что бы он стал детерминированным N-автоматом, принимающем язык $L_\$$, отличающийся от L тем, что в конец каждого слова приписана буква \$. Затем по полученному автомату алгоритмом из теории построим однозначную грамматику, принимающую $L_\$$. Допишем в полученную грамматику правило $\$ \to \varepsilon$. Полученная грамматика будет однозначной и принимать L.

Единственный неочевидный шаг - это построение детерминированного N-автомата. Алгоритм построения следующий. Добавим новое состояние q_s и сделаем его начальным. Единственным переходом из него будет переход вида $(\varepsilon, z_0, z_0\$)$ в бывшее начальное состояние исходного автомата, дописывающий после начального символа стека знак \$. Добавим новое состояние q_f. Из каждого финального состояния исходного автомата добавим переход вида $(\$, ?, \varepsilon)$ в q_f. Из q_f будет переход сам в себя вида $(\varepsilon, ?, \varepsilon)$, опустошающий стек. Каждое из финальных состояний исходного автомата сделаем нефинальным.

Заметим, что построенный автомат будет детерминированным N-автоматом. При этом он будет принимать любое слово, которое принималось исходным автоматом, поскольку после обработки любого такого слова, автомат окажется в одном из бывших финальных состояний, при в его стеке будет находится хотя бы 1 символ (\$), а затем перейдет в состояние q_f, где и опустошит стек. Новых принимаемых слов не добавится, поскольку слова, не принимаемые исходным автоматом, не переведут итоговый автомат в финальное состояние.

Задача 3

(i) Очевидно включение $L(G_3) \subseteq L_1$. Докажем $L_1 \subseteq L(G_3)$ индукцией по длине n слова.

База. При $n = 0$, $\varepsilon \in L(G_3)$.

Переход. Пусть для всех слов длины меньше n утверждение выполняется. Посмотрим на слово $w \in L_1$ длины n.

Заведем счетчик, равный 0 изначально, который уменьшает свое значение на 1, если встречает букву b и увеличивает на 1, если встречает букву a. Будем идти по слову w пока счетчик не обнулится в первый раз (кроме изначального).

Если после этого мы не дошли до конца слова w, то w представимо ввиде xy, где $x \in L_1$, $y \in L_1$, а значит, используя правило $S \to SS$ и выводы слов x и y, можно вывести слово w в G_3.

Если счетчик обнулился после обработки последнего символа в первый раз, то последняя буква слова w должна отличаться от первой. А значит слово w представимо ввиде $w = axb$ или $w = bxa$, где $x \in L_1$. А значит, используя вывод слова x и правило $S \to aSb$ или $S \to bSa$, можно вывести слово w.

(ii) Приведем два правых вывода слова ab в G_3:

$$S \to SS \to S\varepsilon \to aSb \to ab$$
$$S \to SS \to SaSb \to Sa\varepsilon b \to ab$$

(iii) Допустим, что язык не обладает префиксным свойством, при этом для него удалось построить детерминированный N-автомат. Если язык не обладает префиксным свойством, то в нем есть два слова вида: x и xy, причем $|y| > 0$. Поскольку автомат детерминированный, то слово x может быть обработано единственным способом. Поскольку x принимается автоматом, то после обработки слова x стек пуст, но тогда следующий такт невозможен и слово xy не может быть принято. Противоречие, следовательно N-автомат построить нельзя.

Язык L_1 не обладает префиксным свойством, т.к. содержит слова ab и $abab$. Следовательно для него нельзя построить N-автомат.

(iv) Да, поскольку из любого состояния все переходы различны как пары (l, Z), где l - буква перехода, Z - верхний символ стека, а так же для любого состояния, из которого есть переход вида (ε, Z), нет переходов вида $(?, Z)$.

(v) Нет. P_1 удовлетворяет как определению CA, так и определению расширенного CA.

(vi) Докажем, что $L(P_1) = L_1$. Пронумеруем состояния: начальное - q_0, верхнее - q_b, нижнее - q_a.

Докажем, что q_0 "собирает" слова, имеющие одинаковое количество букв a и b, причем в стеке записан только символ z. q_a - слова, содержащие больше либо равно букв a, чем b, причем в стеке записано столько букв a, насколько их содержание превосходит содержание букв b в слове, а затем символ z. q_b аналогично q_a, только оно считает превосходство букв b над a.

Изначально автомат находится в состоянии q_0. Поскольку обработанное слово ε содержит одинаковое количество букв a и b, а в стеке записан символ z, то утверждение выполняется.

Пусть после обработки префикса слова автомат находится в состоянии q_0, а значит префикс содержит равное количество букв a и b. Если слово было обработано полностью, то оно будет принято. Если следующая буква слова это a, то автомат перейдет в состояние q_a, а в стек допишется буква a. Аналогично, если следующая буква b - в состояние q_b, допишется b. При

этом количество букв a (b) в обработанном префиксе будет на 1 превосходить количество b (a). Утверждение выполняется.

Пусть после обработки префикса слова автомат находится в состоянии q_a. Если на вершине стека находится буква a, а следующая буква слова это a, то она допишется в стек. Если следующая буква это b, то из стека сотрется 1 буква a. Если на вершине стека находится символ z, то это означает, что количество букв a равно количеству букв b в обработанном префиксе, а автомат перейдет по ε-переходу в q_0, поскольку других переходов по символу стека z нет. Во всех случаях утверждение выполняется.

Аналогично разбираются переходы из состояния q_b.

(vii) Построим детерминированный N-автомат, эквивалентный данному. Добавим новое состояние q_s и сделаем его начальным. Добавим переход $(\varepsilon, z, z\$)$ в q_0. Добавим новое состояние q_f. Из состояния q_0 добавим переходы $(\$, a, \varepsilon)$, $(\$, b, \varepsilon)$, $(\$, z, \varepsilon)$, $(\$, \$, \varepsilon)$ в q_f. Из q_f добавим переходы $(\varepsilon, a, \varepsilon)$, $(\varepsilon, b, \varepsilon)$, $(\varepsilon, z, \varepsilon)$, $(\varepsilon, \$, \varepsilon)$ в q_f.

Теперь по полученному автомату, в соответствии с алгоритмом из теории, построим однозначную грамматику, и добавим в нее правило $\$ \to \varepsilon$.

Задача 4

Докажем, что данный язык не удовлетворяет лемме о разрастании, а значит не является КС-языком.

Для $\forall C > 0$, возьмем слово $w = a^c b^c a^c b^c$. Посмотрим на любое его разбиение $w = uvzxy$, такое что $|vx| > 0$, $|vzx| < C$.

Будем говорить, что слово w состоит из четырех частей: первая (a^c), вторая (b^c), третья (a^c) и четвертая (b^c).

Если v или x содержит a и b, то при $i = 2$, слово $uv^i zx^i y \notin L$, т.к. при последовательном проходе по его буквам "переходов" между a и b будет больше, чем 3.

Если одно из слов v или x состоит только из a, а второе - только из b, то при $i = 2$, слово $uv^i zx^i y \notin L$, т.к. не сохранится баланс букв между первой и третьей частью, поскольку слово, состоящее только из a является подсловом либо первой, либо третьей части.

Пусть оба слова v и x состоят только из букв a или b. Тогда они оба являются подсловом одной и той же части, т.к. иначе не выполнится $|vzx| < C$, поскольку z является всем, что стоит между словами v и x в слове w. Но в этом случае при $i = 2$ так же не выполняется баланс букв a или b, в зависимости от того из каких букв состоят слова v и x.

Задача 5

(i) В данной грамматике нет бесполезных символов. Покажем, что все символы не бесполезные:

$$A : S \to ABC \to Aab \to ab$$
$$B : S \to ABC \to \varepsilon Bb \to ab$$
$$C : S \to ABC \to \varepsilon aC \to ab$$

(ii) В данной грамматике нет циклов. Любой вывод из нетерминала C будет содержать хотя бы один из терминалов a или b, а значит цикла не будет. Любой вывод из B либо будет содержать терминал a, либо в некоторый момент будет иметь вид CC, а значит при дальнейшем выводе будет содержать a или b. Любой вывод из A будет либо ε, либо в некоторый момент будет иметь вид B, а значит при дальнейшем выводе будет содержать a или b. При выводе из S цикла не получится, поскольку S не содержится в правой части никакого правила.

(iii) Избавимся от ε-правила $A \to \varepsilon$, добавив для каждого правила, содержащему в правой части A, его копию, заменив в ней A на ε:

$$\begin{aligned} S &\to ABC|BC \\ A &\to B \\ B &\to CC|a \\ C &\to AAa|Aa|a|b \end{aligned}$$

(iv) Приведем грамматику к бинарной форме в несколько шагов, используя эквивалентные преобразования. Шаг 1 уже указан в пункте (iii).

Шаг 2. Избавляемся от правила $A \to B$.

$$\begin{aligned} S &\to BBC|BC \\ B &\to CC|a \\ C &\to BBa|Ba|a|b \end{aligned}$$

Шаг 3. Делаем так, что бы в правой части любого правила не стояли терминалы с нетерминалами.

$$\begin{aligned} S &\to BBC|BC \\ B &\to CC|a \\ D &\to a \\ C &\to BBD|BD|a|b \end{aligned}$$

Шаг 4. Разбиваем правила в тремя нетерминалами с правой части на два.

$$\begin{aligned} E &\to BC \\ S &\to BE|BC \\ B &\to CC|a \\ D &\to a \\ F &\to BD \\ C &\to BF|BD|a|b \end{aligned}$$