

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО
ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ ТЕХНИКИ
НАПРАВЛЕНИЕ ПРОГРАММНАЯ ИНЖЕНЕРИЯ
ОБРАЗОВАТЕЛЬНАЯ ПРОГРАММА СИСТЕМНОЕ И ПРИКЛАДНОЕ
ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ
СПЕЦИАЛИЗАЦИЯ СИСТЕМНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

ОТЧЕТ ПО ДОМАШНЕЙ РАБОТЕ № 3
курса «Разработка компиляторов»

**по теме: «Конструирование LL(1) анализатора для
КС-грамматики»**

Вариант № 9

Выполнил студент:
Тюрин Иван Николаевич
группа: Р33102

Преподаватель:
Лаздин А. В.

Санкт-Петербург, 2024 г.

Содержание

Домашняя работа № 3. Конструирование LL(1) анализатора для КС-грамматики	2
1. Задание варианта № 9	2
2. Вывод	8

Домашняя работа № 3

Конструирование LL(1)

анализатора для

КС-грамматики

1. Задание варианта № 9

Исходный вариант:

$$\begin{aligned}S &\rightarrow ABCC \\C &\rightarrow cccA \mid ccBB \mid cC \mid c \\B &\rightarrow BBb \mid BBa \mid b \\A &\rightarrow aAa \mid c\end{aligned}$$

Упрощение (убраны все подряд идущие нетерминалы, которые раскрываются в бесконечные последовательности):

$$\begin{aligned}S &\rightarrow ABC \\C &\rightarrow cccA \mid ccB \mid cC \mid c \\B &\rightarrow Bb \mid Ba \mid b \\A &\rightarrow aAa \mid c\end{aligned}$$

Факторизация C:

$$C \rightarrow cccA \mid ccB \mid cC \mid c$$

$$\begin{aligned}
C &\rightarrow cC_1 \\
C_1 &\rightarrow ccA \mid cB \mid cC_1 \mid \varepsilon
\end{aligned}$$

$$\begin{aligned}
C &\rightarrow cC_1 \\
C_1 &\rightarrow cC_2 \mid \varepsilon \\
C_2 &\rightarrow cA \mid B \mid C_1
\end{aligned}$$

$$\begin{aligned}
C &\rightarrow cC_1 \\
C_1 &\rightarrow cC_2 \mid \varepsilon \\
C_2 &\rightarrow cA \mid B \mid cC_2 \mid \varepsilon
\end{aligned}$$

$$\begin{aligned}
C &\rightarrow cC_1 \\
C_1 &\rightarrow cC_2 \mid \varepsilon \\
C_2 &\rightarrow cC_3 \mid B \mid \varepsilon \\
C_3 &\rightarrow A \mid C_2
\end{aligned}$$

$$\begin{aligned}
C &\rightarrow cC_1 \\
C_1 &\rightarrow cC_2 \mid \varepsilon \\
C_2 &\rightarrow cC_3 \mid B \mid \varepsilon \\
C_3 &\rightarrow A \mid cC_3 \mid B \mid \varepsilon
\end{aligned}$$

$$\begin{aligned}
C &\rightarrow cC_1 \\
C_1 &\rightarrow cC_2 \mid \varepsilon \\
C_2 &\rightarrow cC_3 \mid B \mid \varepsilon \\
C_3 &\rightarrow c \mid aAa \mid cC_3 \mid B \mid \varepsilon
\end{aligned}$$

$$\begin{aligned}
C &\rightarrow cC_1 \\
C_1 &\rightarrow cC_2 \mid \varepsilon \\
C_2 &\rightarrow cC_3 \mid B \mid \varepsilon \\
C_3 &\rightarrow aAa \mid cC_4 \mid B \mid \varepsilon \\
C_4 &\rightarrow C_3 \mid \varepsilon
\end{aligned}$$

Видно, что $C_4 = C_3$, поэтому уберем это правило:

$$\begin{aligned}
C &\rightarrow cC_1 \\
C_1 &\rightarrow cC_2 \mid \varepsilon \\
C_2 &\rightarrow cC_3 \mid B \mid \varepsilon \\
C_3 &\rightarrow aAa \mid cC_3 \mid B \mid \varepsilon
\end{aligned}$$

Факторизация В:

$$B \rightarrow Bb \mid Ba \mid b$$

$$\begin{aligned}
B &\rightarrow BB_1 \mid b \\
B_1 &\rightarrow b \mid a
\end{aligned}$$

Устраняем левую рекурсию В:

$$\begin{aligned}
B &\rightarrow bB_2 \\
B_1 &\rightarrow b \mid a \\
B_2 &\rightarrow B_1B_2 \mid \varepsilon
\end{aligned}$$

Итоговая грамматика:

$$\begin{aligned}
S &\rightarrow ABC \\
C &\rightarrow cC_1 \\
C_1 &\rightarrow cC_2 \mid \varepsilon \\
C_2 &\rightarrow cC_3 \mid B \mid \varepsilon \\
C_3 &\rightarrow A \mid cC_3 \mid B \mid \varepsilon \\
B &\rightarrow bB_2 \\
B_1 &\rightarrow b \mid a \\
B_2 &\rightarrow B_1B_2 \mid \varepsilon \\
A &\rightarrow aAa \mid c
\end{aligned}$$

FIRST	FOLLOW	Name	c	a	b	\$
$\{a, c\}$	$\{\$ \}$	S	$S \rightarrow ABC$	$S \rightarrow ABC$		
$\{c\}$	$\{\$ \}$	C	$C \rightarrow cC_1$			
$\{c, \varepsilon\}$	$\{\$ \}$	C_1	$C_1 \rightarrow cC_2$			$C_1 \rightarrow \varepsilon$
$\{c, \varepsilon, b\}$	$\{\$ \}$	C_2	$C_2 \rightarrow cC_3$		$C_2 \rightarrow B$	$C_2 \rightarrow \varepsilon$
$\{a, c, \varepsilon, b\}$	$\{\$ \}$	C_3	$C_3 \rightarrow cC_3$	$C_3 \rightarrow aAa$	$C_3 \rightarrow B$	$C_3 \rightarrow \varepsilon$
$\{b\}$	$\{c, \$ \}$	B			$B \rightarrow bB_2$	
$\{b, a\}$	$\{b, a, c, \$ \}$	B_1		$B_1 \rightarrow a$	$B_1 \rightarrow b$	
$\{b, a, \varepsilon\}$	$\{c, \$ \}$	B_2	$B_2 \rightarrow$	$B_2 \rightarrow B_1B_2$	$B_2 \rightarrow B_1B_2$	$B_2 \rightarrow \varepsilon$
$\{a, c\}$	$\{b, a\}$	A	$A \rightarrow c$	$A \rightarrow aAa$		

Таблица 1.1: Сконструированный СА

Для сконструированного СА разработаем программный код выполняющий аналогичную функцию, см. 1.1. Программный код составляет код на языке DOT для визуализации в среде Graphviz.

```

1 class SA:
2     id = 0
3     s: str = None
4     idx = 1
5     cur_char: chr = None
6
7
8     def __init__(self, string: str):
9         self.s = string
10        self.cur_char = self.s[0]
11
12
13    def get_next_char(self):
14        self.s += "$"
15        self.idx += 1
16        return self.s[self.idx - 1]
17
18

```

```

19     def preambula(self, from_id, name):
20         local_id = self.id
21         print(f'{{local_id}} [label = "{{name}}"];')
22         self.id += 1
23         print(f"{{from_id}} -> {{local_id}};")
24
25         return local_id
26
27
28     def B1(self, from_id):
29         self.preambula(from_id, "B1")
30
31         if self.cur_char in ["b", "a"]:
32             self.cur_char = self.get_next_char()
33         else:
34             raise ValueError
35
36
37     def B2(self, from_id):
38         local_id = self.preambula(from_id, "B2")
39
40         if self.cur_char in ["c", "$"]:
41             pass
42         elif self.cur_char in ["b", "a"]:
43             self.B1(local_id)
44             self.B2(local_id)
45         else:
46             raise ValueError
47
48
49     def B(self, from_id):
50         local_id = self.preambula(from_id, "B")
51
52         if self.cur_char != "b":
53             raise ValueError
54
55         self.cur_char = self.get_next_char()
56         self.B2(local_id)
57
58
59     def A(self, from_id):
60         local_id = self.preambula(from_id, "A")
61
62         if self.cur_char == "c":
63             self.cur_char = self.get_next_char()
64         elif self.cur_char == "a":
65             self.cur_char = self.get_next_char()
66             self.A(local_id)
67             if self.cur_char != "a":
68                 raise ValueError
69             self.cur_char = self.get_next_char()
70         else:
71             raise ValueError
72
73
74     def C3(self, from_id):
75         local_id = self.preambula(from_id, "C3")
76
77         if self.cur_char == "c":
78             self.cur_char = self.get_next_char()

```

```

79         self.C3(local_id)
80     elif self.cur_char == "b":
81         self.B(local_id)
82     elif self.cur_char == "a":
83         self.cur_char = self.get_next_char()
84         self.A(local_id)
85         if self.cur_char != "a":
86             raise ValueError
87         self.cur_char = self.get_next_char()
88     elif self.cur_char == "$":
89         pass
90     else:
91         raise ValueError
92
93
94     def C2(self, from_id):
95         local_id = self.preamble(from_id, "C2")
96
97         if self.cur_char == "c":
98             self.cur_char = self.get_next_char()
99             self.C3(local_id)
100     elif self.cur_char == "b":
101         self.B(local_id)
102     elif self.cur_char == "$":
103         pass
104     else:
105         raise ValueError
106
107
108     def C1(self, from_id):
109         local_id = self.preamble(from_id, "C1")
110
111         if self.cur_char == "c":
112             self.cur_char = self.get_next_char()
113             self.C2(local_id)
114     elif self.cur_char == "$":
115         pass
116     else:
117         raise ValueError
118
119
120     def C(self, from_id):
121         local_id = self.preamble(from_id, "C")
122
123         if self.cur_char == "c":
124             self.cur_char = self.get_next_char()
125             self.C1(local_id)
126     else:
127         raise ValueError
128
129
130     def S(self):
131         local_id = self.id
132         print(f'[{local_id}] [label = "S"];')
133         self.id += 1
134         self.A(local_id)
135         self.B(local_id)
136         self.C(local_id)
137
138 if __name__ == "__main__":

```



```

139 sa = SA(input())
140 print("digraph {")
141 sa.S()
142 print("}")

```

Листинг 1.1: Python-код синтаксического анализатора

Пример работы программы для строки **acabbс** можно видеть на листинге 1.2. И получившийся граф можно видеть на рисунке 1.1.

```

1 digraph {
2   0 [label = "S"];
3   1 [label = "A"];
4   0 -> 1;
5   2 [label = "A"];
6   1 -> 2;
7   3 [label = "B"];
8   0 -> 3;
9   4 [label = "B2"];
10  3 -> 4;
11  5 [label = "B1"];
12  4 -> 5;
13  6 [label = "B2"];
14  4 -> 6;
15  7 [label = "C"];
16  0 -> 7;
17  8 [label = "C1"];
18  7 -> 8;
19 }

```

Листинг 1.2: Пример вывода программы для строки **acabbс**

2. Вывод

Изучили принципы построения синтаксических анализаторов, способы устранения левой рекурсии и выполнения левой факторизации. Построили синтаксический анализатор для грамматики выданной по варианту и реализовали его поведение в программном коде.

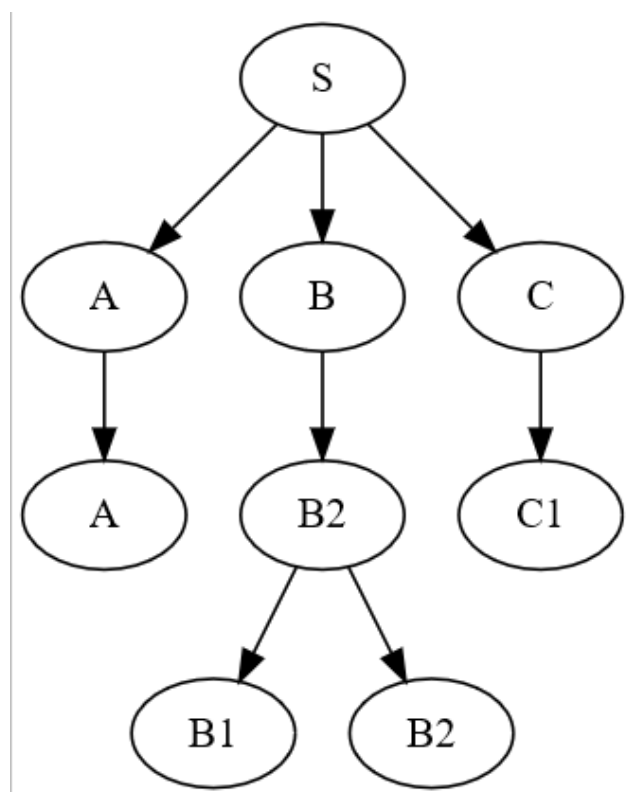


Рис. 1.1: Получившийся граф разбора для строки asabbs