МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

# «БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА» (БГТУ им. В.Г. Шухова)

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

# Лабораторная работа №3

по дисциплине: Теория автоматов и формальных языков тема: «Регулярные языки и конечные распознаватели»

Выполнил: ст. группы ПВ-223 Дмитриев Андрей Александрович

Проверил:

Рязанов Юрий Дмитриевич

Белгород 2024 г.

**Цель работы:** изучить основные способы задания регулярных языков, способы построения, алгоритмы преобразования, анализа и реализации конечных распознавателей.

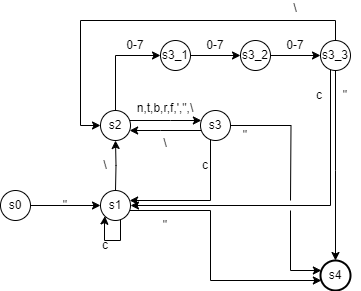
Вариант 2: (по старому пособию)

Язык строковых констант.

Строковая константа — последовательность символов, заключённая в кавычки. Последовательности символов \n , \t , \b , \r , \f , \’ , \” , \\ и символ \ , за которым следует трёхразрядное восьмеричное число, считаются одним символом, а одиночный символ \ недопустим.

**Задание 1. (по старому пособию)** Построить минимальный детерминированный конечный распознаватель заданного языка (см.варианты заданий).

Граф распознавателя:



Распознаватель:

A = <S,X,s0,o,F>

S = {s0, s1, s2, s3, s3\_1, s3\_2, s3\_3, s4}

X = {“, \, n, t, b, r, f, ’, 0-7} + другие символы

s0 = s0

F = {s4}

Распознаватель в виде таблицы:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  | 1 |
|  | s0 | s1 | s2 | s3 | s3\_1 | s3\_2 | s3\_3 | s4 |
| c` |  | s1 |  | s1 |  |  | s1 |  |
| “ | s1 | s4 |  | s4 |  |  | s4 |  |
| \ |  | s2 |  | s2 |  |  | s2 |  |
| {“,\,n,t,b,r,f,’} |  |  | s3 |  |  |  |  |  |
| 0-7 |  |  | s3\_1 |  | s3\_2 | s3\_3 |  |  |

с` – все символы, кроме ‘, “ и \

Классы 0-эквивалентых преобразований:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | К1 | | | | | | | | К2 |
|  | s0 | s1 | s2 | s3 | s3\_1 | s3\_2 | s3\_3 |  | s4 |
| c` | К1 | К1 s1 | К1 | К1 s1 | К1 | К1 | К1 s1 | К1 | К1 |
| “ | К1 s1 | К2 s4 | К1 | К2 s4 | К1 | К1 | К2 s4 | К1 | К1 |
| \ | К1 | К1 s2 | К1 | К1 s2 | К1 | К1 | К1 s2 | К1 | К1 |
| {“,\,n,t,b,r,f,’} | К1 | К1 | К1 s3 | К1 | К1 | К1 | К1 | К1 | К1 |
| 0-7 | К1 | К1 | К1 s3\_1 | К1 | К1 s3\_2 | К1 s3\_3 | К1 | К1 | К1 |

Классы 1-эквивалентых преобразований:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | К7 | К6 | К5 | К4 | К1 | К2 | | | К3 |
|  | s0 | s2 | s3\_1 | s3\_2 |  | s1 | s3 | s3\_3 | s4 |
| c` | К1 | К1 | К1 | К1 | К1 | К2 s1 | К2 s1 | К2 s1 | К1 |
| “ | К2 s1 | К1 | К1 | К1 | К1 | К3 s4 | К3 s4 | К3 s4 | К1 |
| \ | К1 | К1 | К1 | К1 | К1 | К1 s2 | К1 s2 | К1 s2 | К1 |
| {“,\,n,t,b,r,f,’} | К1 | К2 s3 | К1 | К1 | К1 | К1 | К1 | К1 | К1 |
| 0-7 | К1 | К5 s3\_1 | К4 s3\_2 | К2 s3\_3 | К1 | К1 | К1 | К1 | К1 |

Получим распознаватель:

A = <S,X,s0,o,F>

S = {s0, s1, s2, s3\_1, s3\_2, s4}

X = {“, \, n, t, b, r, f, ’, 0-7} + другие символы

s0 = s0

F = {s4}

Окончательная таблица переходов минимального распознавателя:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | 1 |
|  | s0 | s1 | s2 | s3\_1 | s3\_2 | s4 |
| c` |  | s1 |  |  |  |  |
| “ | s1 | s4 |  |  |  |  |
| \ |  | s2 |  |  |  |  |
| {“,\,n,t,b,r,f,’} |  |  | s1 |  |  |  |
| 0-7 |  |  | s3\_1 | s3\_2 | s1 |  |

**Задание 1.** Язык L1 в алфавите {a,b}, представляющий собой множество цепочек, в

которых символ а встречается не менее одного раза, а символ b — не более

одного раза, задан грамматикой:

S → aAbA

S → baA

A → Aa

A → ε

Построить детерминированный конечный распознаватель языка L1.

1. Приведём грамматику к НФГ:

S → aAS1

S1 → bA

S → bS2

S2 → aA

A → aA

A → ε

2. Преобразуем в регулярную правостороннюю грамматику и получим искомую грамматику:

S → aN1

S1 → bA

S → bS2

S2 → aA

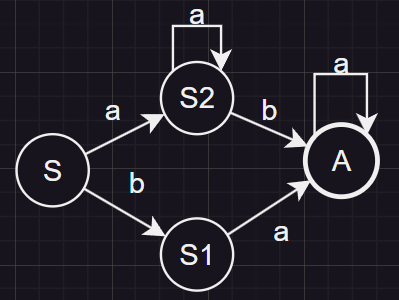
A → aA

A → ε

N1 → bA

N1 → aN1

Схема распознавателя:



**Задание 2.** Язык L2 в алфавите {a,b}, представляющий собой множество цепочек, в

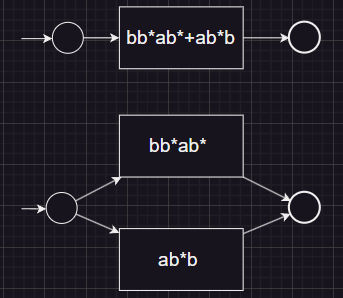
которых символ b встречается не менее одного раза, а символ а — не более

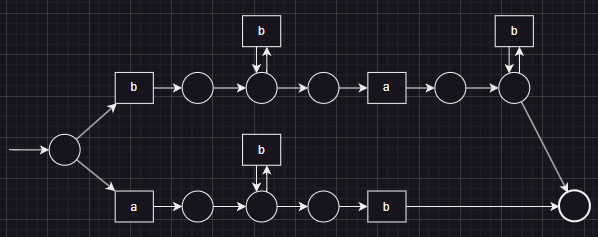
одного раза, задан регулярным выражением:

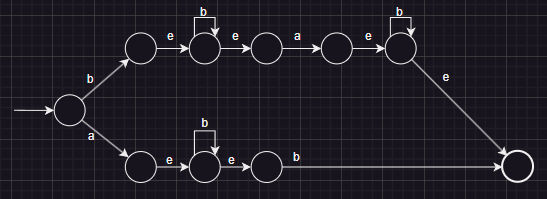
bb\*ab\*+ab\*b

Построить детерминированный конечный распознаватель языка L2.

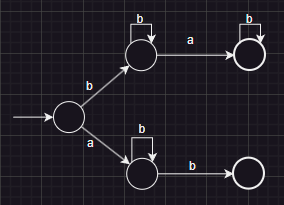
Вывод недетерминированного распознавателя по шагам:







Полученный детерминированный распознаватель:



**Задание 3.** Построить минимальный детерминированный конечный распознаватель

языка L3 в алфавите {a,b}, представляющий собой множество цепочек, в

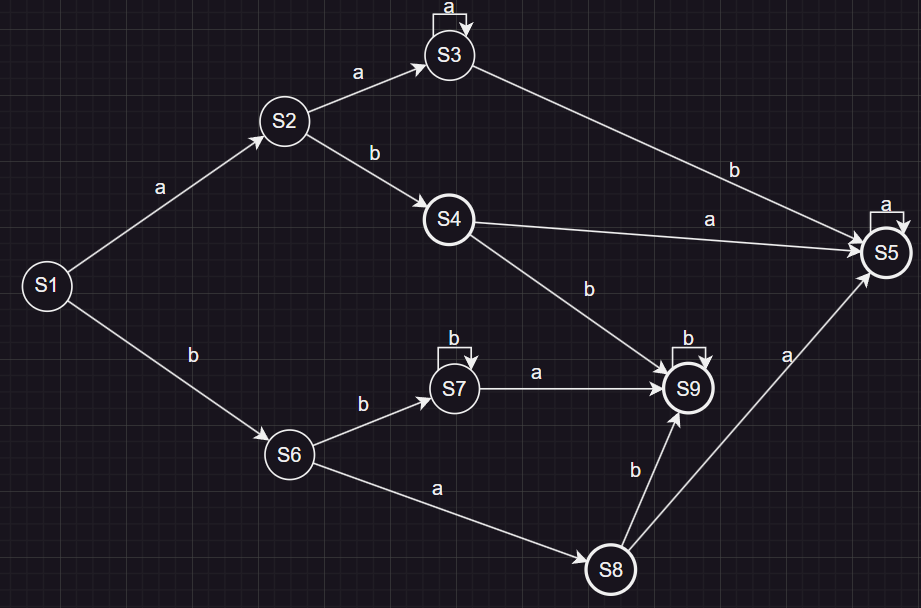
которых символ а встречается не менее одного раза, а символ b — не более

одного раза, или символ b встречается не менее одного раза, а символ а — не

более одного раза.

Детерминированный распознаватель имеет вид:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **↓** |  |  | 1 | 1 |  |  | 1 | 1 |
|  | S1 | S2 | S3 | S4 | S5 | S6 | S7 | S8 | S9 |
| a | S2 | S3 | S3 | S5 | S5 | S8 | S9 | S5 |  |
| b | S6 | S4 | S5 | S9 |  | S7 | S7 | S9 | S9 |



Минимизируем, используя эквивалентные преобразования:

Классы 0-эквивалентности:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | К1 | | | | | | | | К2 | |
|  | S1 | S2 | S3 | S4 | S8 | S6 | S7 |  | S5 | S9 |
| a | K1 S2 | K1 S3 | K1 S3 | K2 S5 | K2 S5 | K1 S8 | K2 S9 | К1 | К2 S5 |  |
| b | K1 S6 | K1 S4 | K2 S5 | K2 S9 | K2 S9 | K1 S7 | K1 S7 | К1 |  | К2 S9 |

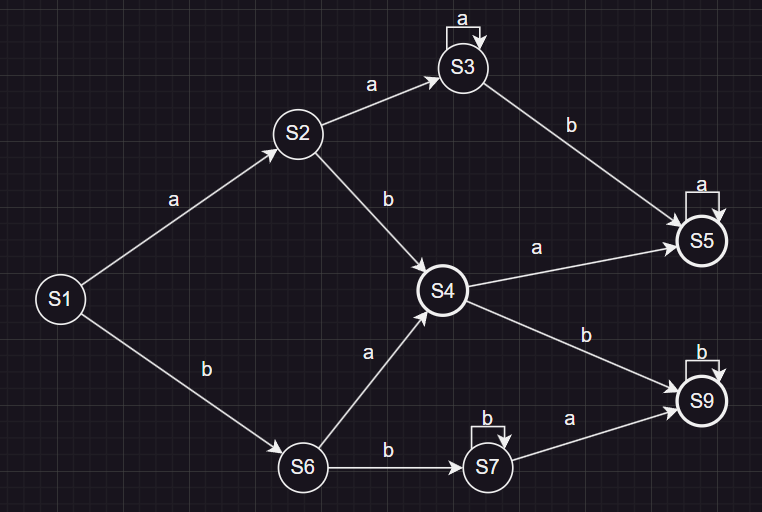
Классы 1-эквивалентности:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | К1 | | | | | | К3 | | К2 | |
|  | S1 | S2 | S3 | S6 | S7 |  | S4 | S8 | S5 | S9 |
| a | K1 S2 | K1 S3 | K1 S3 | K3 S8 | K2 S9 | К1 | K2 S5 | K2 S5 | К2 S5 | K1 |
| b | K1 S6 | K3 S4 | K2 S5 | K1 S7 | K1 S7 | К1 | K2 S9 | K2 S9 | K1 | К2 S9 |

Классы 2-эквивалентности:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | К1 | K4 | K5 | K6 | K7 | K8 | К3 | | К2 | |
|  | S1 | S2 | S3 | S6 | S7 |  | S4 | S8 | S5 | S9 |
| a | K4 S2 | K5 S3 | K5 S3 | K3 S8 | K2 S9 | К8 | K2 S5 | K2 S5 | К2 S5 | K8 |
| b | K6 S6 | K3 S4 | K2 S5 | K7 S7 | K7 S7 | К8 | K2 S9 | K2 S9 | K8 | К2 S9 |

Получим минимизированный детерминированный распознаватель:



|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **↓** |  |  | 1 | 1 |  |  | 1 |
|  | S1 | S2 | S3 | S4 | S5 | S6 | S7 | S9 |
| a | S2 | S3 | S3 | S5 | S5 | S4 | S9 |  |
| b | S6 | S4 | S5 | S9 |  | S7 | S7 | S9 |

**Задание 4.** Написать программу компиляционного типа для реализации

минимального детерминированного конечного распознавателя языка L3.

Код программы:

MESSAGES = {  
 -1: "Отвергнуть. Последовательность пуста",  
 -2: "Отвергнуть. Входной символ невалидный",  
 -3: "Отвергнуть. Символ b должен быть введён не более 1 раза",  
 -4: "Отвергнуть. Символ а должен быть введён не более 1 раза",  
 0: "Допустить."  
}  
  
  
def S1(input):  
 if len(input) == 0:  
 return -1  
  
 if input[0] == 'a':  
 return S2(input[1:])  
 elif input[0] == 'b':  
 return S6(input[1:])  
 else:  
 return -2  
  
  
def S2(input):  
 if len(input) == 0:  
 return -1  
  
 if input[0] == 'a':  
 return S3(input[1:])  
 elif input[0] == 'b':  
 return S4(input[1:])  
 else:  
 return -2  
  
  
def S3(input):  
 \_input = input  
 while True:  
 if len(\_input) == 0:  
 return -1  
  
 if \_input[0] == 'a':  
 \_input = \_input[1:]  
 elif \_input[0] == 'b':  
 return S5(\_input[1:])  
 else:  
 return -2  
  
  
def S5(input):  
 \_input = input  
 while True:  
 if len(\_input) == 0:  
 return 0  
  
 if \_input[0] == 'a':  
 \_input = \_input[1:]  
 elif \_input[0] == 'b':  
 return -3  
 else:  
 return -2  
  
  
def S4(input):  
 if len(input) == 0:  
 return 0  
  
 if input[0] == 'a':  
 return S5(input[1:])  
 elif input[0] == 'b':  
 return S9(input[1:])  
 else:  
 return -2  
  
  
def S6(input):  
 if len(input) == 0:  
 return -1  
  
 if input[0] == 'a':  
 return S4(input[1:])  
 elif input[0] == 'b':  
 return S7(input[1:])  
 else:  
 return -2  
  
  
def S7(input):  
 \_input = input  
 while True:  
 if len(\_input) == 0:  
 return -1  
  
 if \_input[0] == 'a':  
 return S9(\_input[1:])  
 elif \_input[0] == 'b':  
 \_input = \_input[1:]  
 else:  
 return -2  
  
  
def S9(input):  
 \_input = input  
 while True:  
 if len(\_input) == 0:  
 return 0  
  
 if \_input[0] == 'a':  
 return -4  
 elif \_input[0] == 'b':  
 \_input = \_input[1:]  
 else:  
 return -2  
  
  
def L3validator(input):  
 result = S1(input)  
 return result

**Задание 5.** Написать программу интерпретационного типа для реализации

минимального детерминированного конечного распознавателя языка L3.

Код программы:

CONVERT = {  
 2: -5,  
 3: -5,  
 6: -6,  
 7: -6,  
 -1: -1,  
 -2: -2,  
 -3: -3,  
 -4: -4,  
 -5: -5,  
 -6: -6,  
}  
  
MESSAGES = {  
 -1: "Отвергнуть. Последовательность пуста",  
 -2: "Отвергнуть. Входной символ невалидный",  
 -3: "Отвергнуть. Символ b должен быть введён не более 1 раза",  
 -4: "Отвергнуть. Символ а должен быть введён не более 1 раза",  
 -5: "Отвергнуть. Символ b должен быть введён хотя бы 1 раз",  
 -6: "Отвергнуть. Символ a должен быть введён хотя бы 1 раз",  
 0: "Допустить",  
}  
  
PERMITTING = [4, 5, 8]  
  
MATRIX = {  
 'a': [2, 3, 3, 5, 5, 4, 8, -4],  
 'b': [6, 4, 5, 8, -3, 7, 7, 8],  
}  
  
  
def L3validator(input):  
 if len(input) == 0:  
 return -1  
  
 S = 1  
 while len(input) > 0 and S > 0:  
 if not input[0] in MATRIX.keys():  
 return -2  
 S = MATRIX[input[0]][S - 1]  
 input = input[1:]  
  
 if S in PERMITTING:  
 return 0  
  
 return S

**Задание** **6.** Подобрать наборы тестовых данных так, чтобы в процессе тестирования

сработал каждый переход конечного распознавателя.

Тестовые данные:

|  |  |
| --- | --- |
| "aaab" | Допустить |
| "aab" | Допустить |
| "aaba" | Допустить |
| "ab" | Допустить |
| "aba" | Допустить |
| "abaa" | Допустить |
| "ba" | Допустить |
| "bab" | Допустить |
| "babb" | Допустить |
| "bbba" | Допустить |
| "bba" | Допустить |
| "bbbab" | Допустить |
| "bbab" | Допустить |

**Задание** **7.** Подобрать наборы тестовых данных так, чтобы в процессе тестирования

распознаватель закончил обработку цепочек в каждом состоянии конечного

распознавателя.

Тестовые данные:

|  |  |  |
| --- | --- | --- |
| "" | S1 | Отвергнуть |
| "c" | S1 | Отвергнуть |
| "a" | S2 | Отвергнуть |
| "aaa" | S3 | Отвергнуть |
| "ab" | S4 | Допустить |
| "ba" | S4 | Допустить |
| "aab" | S5 | Допустить |
| "aba" | S5 | Допустить |
| "baa" | S5 | Допустить |
| "aabb" | S5 | Отвергнуть |
| "b" | S6 | Отвергнуть |
| "bb" | S7 | Отвергнуть |
| "bba" | S9 | Допустить |
| "bab" | S9 | Допустить |
| "abb" | S9 | Допустить |
| "bbaa" | S9 | Отвергнуть |

**Задание** **8.** Выполнить тестирование программ для реализации минимального

детерминированного конечного распознавателя языка L3.

check\_set = [  
 ("aaab", 1),  
 ("aab", 1),  
 ("aaba", 1),  
 ("ab", 1),  
 ("aba", 1),  
 ("abaa", 1),  
 ("ba", 1),  
 ("bab", 1),  
 ("babb", 1),  
 ("bbba", 1),  
 ("bba", 1),  
 ("bbbab", 1),  
 ("bbab", 1),  
 ("", 0), # 1  
 ("c", 0), # 1  
 ("a", 0), # 2  
 ("aaa", 0), # 3  
 ("ab", 1), # 4  
 ("ba", 1), # 4  
 ("aab", 1), # 5  
 ("aba", 1), # 5  
 ("baa", 1), # 5  
 ("aabb", 0), # 5  
 ("b", 0), # 6  
 ("bb", 0), # 7  
 ("bba", 1), # 9  
 ("bab", 1), # 9  
 ("abb", 1), # 9  
 ("bbaa", 0) # 9  
]  
  
for check in check\_set:  
 res = L3validator(check[0])  
 if check[1]:  
 print(check[0], res >= 0, MESSAGES[res], sep='\t')  
 else:  
 print(check[0], res < 0, MESSAGES[res], sep='\t')

Вывод программы:

aaab True Допустить.

aab True Допустить.

aaba True Допустить.

ab True Допустить.

aba True Допустить.

abaa True Допустить.

ba True Допустить.

bab True Допустить.

babb True Допустить.

bbba True Допустить.

bba True Допустить.

bbbab True Допустить.

bbab True Допустить.

True Отвергнуть. Последовательность пуста

c True Отвергнуть. Входной символ невалидный

a True Отвергнуть. Символ b должен быть введён хотя бы 1 раз

aaa True Отвергнуть. Символ b должен быть введён хотя бы 1 раз

ab True Допустить.

ba True Допустить.

aab True Допустить.

aba True Допустить.

baa True Допустить.

aabb True Отвергнуть. Символ b должен быть введён не более 1 раза

b True Отвергнуть. Символ a должен быть введён хотя бы 1 раз

bb True Отвергнуть. Символ a должен быть введён хотя бы 1 раз

bba True Допустить.

bab True Допустить.

abb True Допустить.

bbaa True Отвергнуть. Символ а должен быть введён не более 1 раза

**Вывод:** в ходе работы изучены основные способы задания регулярных языков, способы построения, алгоритмы преобразования, анализа и реализации конечных распознавателей.