**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)**



ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И УПРАВЛЯЮЩИХ СИСТЕМ

**Лабораторная работа №3**

по дисциплине: Теория автоматов и формальных языков

тема: «Регулярные языки и конечные распознаватели»

Выполнил: ст. группы ПВ-223

Степанов Дмитрий Сергеевич

Проверили:

Рязанов Юрий Дмитриевич

Белгород 2024 г.

**Лабораторная работа №3  
Регулярные языки и конечные распознаватели  
Вариант 1**

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

1. Язык L1 в алфавите {a,b,c}, представляющий собой множество цепочек, которые содержат хотя бы один символ а, задан грамматикой:

S→RA

R→Rb

R→Rc

R→ε

A→aP

P→aP

P→bP

P→cP

P→ ε

Построить детерминированный конечный распознаватель языка L1.

Чтобы получить распознаватель необходимо преобразовать грамматику в автоматную правую.

Удалим e-правила:

S→RA

S→A

R→Rb

R→b

R→Rc

R→c

A→aP

A→a

P→aP

P→a

P→bP

P→b

P→cP

P→c

Избавимся от цепных правил:

S→RA

S→ aP

S→a

R→Rb

R→Rc

R→b

R→c

A→aP

A→a

P→aP

P→a

P→bP

P→b

P→cP

P→c

Избавимся от левой рекурсии:

S→RA

S→ aP

S→a

R→bB1

R→cB1

A→aP

A→a

P→aP

P→a

P→bP

P→b

P→cP

P→c

B1→bB1

B1→cB1

B1→ ε

Избавимся от e-правил:

S→RA

S→ aP

S→a

R→bB1

R→b

R→cB1

R→c

A→aP

A→a

P→aP

P→a

P→bP

P→b

P→cP

P→c

B1→bB1

B1→cB1

B1→b

B1→c

Преобразуем грамматику к виду, когда все правые части правил начинаются с терминала:

S→ bB1A

S→ cB1A

S→ bA

S→ cA

S→ aP

S→a

R→bB1

R→b

R→cB1

R→c

A→aP

A→a

P→aP

P→a

P→bP

P→b

P→cP

P→c

B1→bB1

B1→cB1

B1→b

B1→c

Удалим недостижимые символы:

S→ bB1A

S→ cB1A

S→ bA

S→ cA

S→ aP

S→a

A→aP

A→a

P→aP

P→a

P→bP

P→b

P→cP

P→c

B1→bB1

B1→cB1

B1→b

B1→c

Приведём грамматику к правосторонней грамматике:

N1 → bB1A, то же что и N1 → bN1

N1 → cB1A, то же что и N1 → cN1

N1 → bA

N1 → cA

Привели грамматику к правосторонней:

S→ bN1

S→ cN1

S→ aP

S→a

S→ bA

S→ cA

A→aP

A→a

P→aP

P→a

P→bP

P→b

P→cP

P→c

B1→bB1

B1→cB1

B1→b

B1→c

N1 → bN1

N1 → cN1

N1 → bA

N1 → cA

Приведём грамматику к правосторонней автоматной. Введём правило N2 → ε:

S→ bN1

S→ cN1

S→ aP

S→aN2

S→ bA

S→ cA

A→aP

A→aN2

P→aP

P→aN2

P→bP

P→bN2

P→cP

P→cN2

B1→bB1

B1→cB1

B1→bN2

B1→cN2

N1 → bN1

N1 → cN1

N1 → bA

N1 → cA

N2 → ε

Теперь можем построить распознаватель:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | ↓ |  |  |  |  | 1 |
|  | S | A | P | B1 | N1 | N2 |
| a | P, N2 | P, N2 | P, N2 |  |  |  |
| b | N1, A |  | P, N2 | B1, N2 | N1, A |  |
| c | N1, A |  | P, N2 | B1, N2 | N1, A |  |

Приведём недетерминированный распознаватель к детерминированному:

|  |  |  |  |
| --- | --- | --- | --- |
|  | ↓ | 1 |  |
|  | {S} | {P, N2} | {N1, A} |
| a | {P, N2} | {P, N2} | {P, N2} |
| b | {N1, A} | {P, N2} | {N1, A} |
| c | {N1, A} | {P, N2} | {N1, A} |

|  |  |  |  |
| --- | --- | --- | --- |
|  | ↓ | 1 |  |
|  | S1 | S2 | S3 |
| a | S2 | S2 | S2 |
| b | S3 | S2 | S3 |
| c | S3 | S2 | S3 |

Получили детерминированный распознаватель для языка L1:



1. Язык L2 в алфавите {a,b,c}, представляющий собой множество цепочек, которые содержат хотя бы один символ b, задан регулярным выражением: (a+c)\*b(a+b+c)\*



Избавимся от e-переходов:

e(S1) = {S1, S2, S3}

e(S2) = {S2, S3}

e(S3) = {S3}

e(S4) = {S4, S5, S6}

e(S5) = {S5, S6}

e(S6) = {S6}

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | ↓ | ↓ | ↓ | 1 | 1 | 1 |
|  | e(S1) {S1, S2, S3} | e(S2) {S2, S3} | e(S3)  {S3} | e(S4)  {S4, S5, S6} | e(S5)  {S5, S6} | e(S6)  {S6} |
| a | e(S2) | e(S2) |  | e(S5) | e(S5) |  |
| b | e(S4) | e(S4) | e(S4) | e(S5) | e(S5) |  |
| c | e(S2) | e(S2) |  | e(S5) | e(S5) |  |

Удалим недостижимый символ S6

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | ↓ | ↓ | ↓ | 1 | 1 |
|  | S1 | S2 | S3 | S4 | S5 |
| a | S2 | S2 |  | S5 | S5 |
| b | S4 | S4 | S4 | S5 | S5 |
| c | S2 | S2 |  | S5 | S5 |

Преобразуем недетерминированный распознаватель в детерминированный

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | ↓ |  | 1 | 1 |
|  | {S1, S2, S3} | {S2} | {S4} | {S5} |
| a | {S2} | {S2} | {S5} | {S5} |
| b | {S4} | {S4} | {S5} | {S5} |
| c | {S2} | {S2} | {S5} | {S5} |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | ↓ |  | 1 | 1 |
|  | S1 | S2 | S3 | S4 |
| a | S2 | S2 | S4 | S4 |
| b | S3 | S3 | S4 | S4 |
| c | S2 | S2 | S4 | S4 |

Получили детерминированный распознаватель для языка L2



1. Построить минимальный детерминированный конечный распознаватель языка L3 в алфавите {a,b,c}, представляющий собой множество цепочек, которые содержат хотя бы один символ а или хотя бы один символ b.



Преобразуем недетерминированный распознаватель в детерминированный:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | ↓ | 1 | 1 |  | 1 | 1 | 1 |
|  | {S1, S’1} | {S2, S’2} | {S3, S’3} | {S3, S’2} | {S2, S’3} | {S2, S’4} | {S3, S’4} |
| a | {S2, S’2} | {S2, S’2} | {S2, S’4} | {S2, S’2} | {S2, S’4} | {S2, S’4} | {S2, S’4} |
| b | {S3, S’3} | {S2, S’3} | {S3, S’4} | {S3, S’3} | {S2, S’4} | {S2, S’4} | {S3, S’4} |
| c | {S3, S’2} | {S2, S’2} | {S3, S’4} | {S3, S’2} | {S2, S’4} | {S2, S’4} | {S3, S’4} |

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

Выполним минимизацию:

K1 = {S1, S4, e}

K2 = {S2, S3, S5, S6, S7}

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | K1 | | | K2 | | | | |
|  | S1 | S4 | e | S2 | S3 | S5 | S6 | S7 |
| a | K2 | K2 | K1 | K2 | K2 | K2 | K2 | K2 |
| b | K2 | K2 | K1 | K2 | K2 | K2 | K2 | K2 |
| c | K1 | K1 | K1 | K2 | K2 | K2 | K2 | K2 |

Получили таблицу переходов в классы 0-эквивалентных состояний. На основе этой

таблицы можем построить таблицу переходов в классы 1-эквивалентных состояний.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | K1 | | K3 | K2 | | | | |
|  | S1 | S4 | e | S2 | S3 | S5 | S6 | S7 |
| a | K2 | K2 | K3 | K2 | K2 | K2 | K2 | K2 |
| b | K2 | K2 | K3 | K2 | K2 | K2 | K2 | K2 |
| c | K1 | K1 | K3 | K2 | K2 | K2 | K2 | K2 |

По таблице видно, что классы 2-эквивалентных состояний совпадают с классами 1-эквивалентных состояний, следовательно, классы 1-эквивалентных состояний представляют собой классы эквивалентных состояний. Переобозначим K1=S1, K2=S2. Таблица переходов минимального распознавателя:

|  |  |  |
| --- | --- | --- |
|  | ↓ | 1 |
|  | S1 | S2 |
| a | S2 | S2 |
| b | S2 | S2 |
| c | S1 | S2 |

Получили минимальный детерминированный распознаватель для языка L3.



1. Написать программу компиляционного типа для реализации минимального детерминированного конечного распознавателя языка L3.

MESSAGES = {  
 -1: "Отвергнуть, невалидный входной символ",  
 0: "Отвергнуть, цепочка не содержит a или b",  
 1: "Допустить",  
}  
  
  
def validate(input):  
 original\_input = input  
 s = 0  
 while len(input) > 0 and s >= 0:  
 current\_symbol = input[0]  
 if s == 0:  
 if current\_symbol == "c":  
 s = 0  
 elif current\_symbol == "a" or current\_symbol == "b":  
 s = 1  
 else:  
 s = -1  
 break  
 elif s == 1:  
 if current\_symbol == "a" or current\_symbol == "b" or current\_symbol == "c":  
 s = 1  
 else:  
 s = -1  
 break  
  
 input = input[1:]  
  
 print(original\_input, MESSAGES[s])  
 return s

1. Написать программу интерпретационного типа для реализации минимального детерминированного конечного распознавателя языка L3.

MESSAGES = {  
 -1: "Отвергнуть, невалидный входной символ",  
 0: "Отвергнуть, цепочка не содержит a или b",  
 1: "Допустить",  
}  
  
PERMITTING = [1]  
  
MATRIX = {  
 "a": [1, 1],  
 "b": [1, 1],  
 "c": [0, 1]  
}  
  
  
def validate(input):  
 original\_input = input  
 s = 0  
 while len(input) > 0 and s >= 0:  
 current\_symbol = input[0]  
 if current\_symbol in MATRIX:  
 s = MATRIX[input[0]][s]  
 else:  
 s = -1  
 break  
  
 input = input[1:]  
  
 if s in PERMITTING:  
 s = 1  
  
 print(original\_input, MESSAGES[s])  
 return s

1. Подобрать наборы тестовых данных так, чтобы в процессе тестирования сработал каждый переход конечного распознавателя.
   1. caabc
   2. cbabc
2. Подобрать наборы тестовых данных так, чтобы в процессе тестирования распознаватель закончил обработку цепочек в каждом состоянии конечного распознавателя.
   1. ccc
   2. a
3. Выполнить тестирование программ для реализации минимального детерминированного конечного распознавателя языка L3.

Результат выполнения программ:

caabc Допустить

cbabc Допустить

ccc Отвергнуть, цепочка не содержит a или b

a Допустить

caabc Допустить

cbabc Допустить

ccc Отвергнуть, цепочка не содержит a или b

a Допустить

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