Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

Лабораторная работа 13

По дисциплине «Конструирование программного обеспечения»

На тему «Формальные языки и формальные грамматики, порождающие их»

Выполнил:

Студент 1 курса 10 группы, 1 подгруппы

Редько Павел Валерьевич

2024, Минск

Вариант 12



3. Приведите 7 примеров цепочек символов, принадлежащих регулярному множеству.

p – proc

r - □

b – start

c – cos

y

s – sin

x

e - end

1. prre
2. prbcre
3. prbcyre
4. prbsxre
5. prbsxxre
6. prbcyyyre
7. prrbsxxxbcyrre

4. Выберите любую из цепочек, сформированных в предыдущем задании, и постройте для нее диаграмму мгновенных состояний конечного автомата.

Цепочка: 1) prre

* начальное состояние s0;
* считывание символов proc, переход в состояние s1;
* считывание символов start, переход в состояние s2;
* считывание символов cosy, переход в состояние s3;
* считывание символов sin, переход в состояние s4;
* считывание символов end, переход в конечное состояние s5;

s0 → p → s1 → b→ s2 → c → s3 → s → s4 → e →s5;

5. Постройте граф конечного автомата, распознающего цепочки, описанные регулярным выражением.

a – proc; b - □; c – start; d – cos; e – y; f – sin; g – x; h – end;

Ответы на вопросы:

1. I – алфавит входных символов.
2. I+ – множество всех цепочек, состоящих из символов алфавита I, исключая пустую цепочку (λ).

I\* – множество всех цепочек, состоящих из символов алфавита I, включая пустую цепочку.

λ - пустая цепочка.

1. Языком L(I) над алфавитом I называется произвольное подмножество цепочек из I\*.
2. Формальная грамматика – это способ задания языка и система, которая состоит из T(множества терминальных символов), N(множества нетерминальных символов), P(множества правил грамматики), S(начального символа грамматики).
3. Цепочка b выводится из цепочки a

В первом случае число шагов вывода n = 0;

Во втором случае число шагов вывода b из a n >= 0.

1. Язык L, порождаемый грамматикой G – это множество всех цепочек, которые можно получить из начального символа S с помощью правила P.
2. БНФ представляет собой систему описания синтаксиса языков программирования и других формальных языков
3. Регулярная грамматика — это тип формальной грамматики, который используется для описания регулярных языков.
4. Регулярное множество - множество цепочек описанных регулярным выражением.
5. Регулярный язык – это тип формального языка, который может быть определен регулярным выражением и распознан конечным автоматом.
6. Лексический анализ – первая (наиболее простая) фаза трансляции.
7. Лексический анализатор — это программа или часть транслятора, которая преобразует исходный код в набор лексем (токенов).
8. Входной информацией для лексического анализатора является исходный код программы, который он преобразует в последовательность лексем.

Выходной информацией лексического анализатора является таблица лексем (или токенов), которая содержит классифицированные лексемы, и таблица идентификаторов, где перечислены все идентификаторы и их атрибуты.

1. Последовательное взаимодействие:
2. Лексический анализатор сначала обрабатывает весь исходный код и создает таблицу лексем.
3. После этого синтаксический анализатор начинает свою работу, используя данные из таблицы лексем.
4. Этот процесс является линейным: сначала полностью выполняется один шаг, затем следующий.

Параллельное взаимодействие:

1. Лексический и синтаксический анализаторы работают одновременно.
2. Лексический анализатор передает лексемы синтаксическому анализатору по мере их обнаружения.
3. Это позволяет синтаксическому анализатору начать работу до того, как лексический анализатор закончит обработку всего кода.
4. Регулярное выражение над алфавитом ( I ) — это формальный способ описания множества строк (цепочек символов), составленных из символов алфавита ( I ).

Регулярные выражения над алфавитом I и языки, представляемые ими, рекурсивно определяются следующим образом:

1) ∅ – регулярное выражение, представляет пустое множество;

2) λ – регулярное выражение, представляет множество {λ} ;

3) для каждого a∈I символ a является регулярным выражением и представляет множество {a} ;

4) если p – регулярное выражение, представляющее множество P и q – регулярное выражение, представляющее множество Q , то p + q, pq, q \* являются регулярными выражениями и представляют множества:

a) P ∪ Q (объединение),

b) PQ (конкатенация множеств),

c) P\* (итерация) соответственно.

5) + pp\* = p

1. КА это пятерка M = S, I, δ, s0, F, где:
2. S – конечное множество состояний устройства управления;
3. I – алфавит входных символов;
4. δ – функция переходов, отображающая S × (I ∪{λ}) во множество подмножеств S : δ(s,i) ⊂ S,s∈S,i∈ I ;
5. s0 ∈ S - начальное состояние устройства управления;
6. F ⊆ S - множество заключительных (допускающих) состояний устройства управления.
7. Отличие между детерминированным (ДКА) и недетерминированным (НКА) конечным автоматом заключается в следующем:

Детерминированный Конечный Автомат (ДКА):

1. Имеет строго один переход для каждого состояния и символа алфавита.
2. Не содержит λ-переходов (переходов по пустой строке).
3. Функция переходов однозначно определяет следующее состояние.

Недетерминированный Конечный Автомат (НКА):

1. Может иметь несколько переходов для одного состояния и символа.
2. Может содержать λ-переходы.
3. Функция переходов может отображать в несколько возможных состояний.
4. Мгновенное состояние конечного автомата - это пара, состоящая из текущего состояния автомата и оставшейся необработанной части входной цепочки. В контексте данной страницы:
5. Текущее состояние (s): Это состояние из конечного множества состояний ( S ), в котором находится автомат в данный момент.
6. Входная цепочка (w): Это часть входной цепочки символов алфавита ( I ), которая еще не была обработана автоматом.
7. (s,aw) > (s ′ ,w)

* Это обозначение описывает один шаг работы конечного автомата. Если конечный автомат находится в состоянии s и входная цепочка начинается с символа a, за которым следует строка w, то автомат может перейти в состояние s' и продолжить работу со строкой w. Здесь > означает “непосредственно следует”.

(si, wi) > (sk, wk)

* (s\_i) и (s\_k) - это состояния конечного автомата. (s\_i) - начальное состояние, а (s\_k) - конечное состояние после перехода.
* (w\_i) и (w\_k) - это части входной строки (цепочки символов), которые еще не были обработаны. (w\_i) - это входная строка до перехода, а (w\_k) - оставшаяся часть строки после того, как автомат совершил переход и возможно обработал один или несколько символов.

1. Соотношение
   1. язык является регулярным множеством тогда и только тогда, когда он задан регулярной грамматикой;
   2. язык может быть задан регулярной грамматикой (левосторонней или правосторонней) тогда и только тогда, когда язык является регулярным множеством;
   3. язык является регулярным множеством тогда и только тогда, когда он задан конечным автоматом;
   4. язык распознается с помощью конечного автомата тогда и только тогда, когда он является регулярным множеством.