Московский государственный университет

Факультет вычислительной математики и кибернетики

Отчет по программе практикума

Повжик Юрий

325 группа

Москва 2023

Содержание

Постановка задачи

По заданному тексту *регулярной* (леволинейной или праволинейной) грамматики построить соответствующий детерминированный конечный автомат (*диаграмму состояний*).

Грамматика задается как конечный набор правил, состоящих из левой и правой частей, например: T=aN|bN. Нетерминальные символы грамматики записываются большими латинскими буквами, а терминальные – маленькими.

Построенный автомат представляет собой ориентированный и помеченный граф: вершины графа соответствуют состояниям автомата и помечены нетерминальными символами грамматики. Ребра графа соответствуют переходам между состояниями автомата и помечены терминальными символами грамматики. Граф записывается в виде списка входящих в него ребер, каждое ребро представлено трехэлементным списком вида (метка_вершины метка_ребра метка_вершины).

При записи грамматики нужно заключить в круглые скобки каждое ее правило, а также левые и правые части правил – тем самым получаем лисповский список правил. Для запаси терминальных символов и символа | использовать символы Лиспа, например:

(((S) = (#\a N I #\b N)) ((N) = (#\c N I #\d))).

В этом списке терминальные и нетерминальные символы разделены пробелами.

Реализация

1) Построение автомата

На вход может быть подана как леволинейная грамматика, так и праволинейная:

(((S) = (B #\b I C #\c)) ((B) = (C #\b I #\a)) ((C) = (B #\b I #\a)))

(((H) = (#\a A I #\a B)) ((A) = (#\a S)) ((B) = (#\b S)))

При первой вершине равной S мы считаем грамматику леволинейной, иначе – праволинейной.

По праволинейной грамматике автормат строит функция CrAut, по леволинейной – CrAut_2.

Как это делается:

Мы берем поочередно правила, из примера выше:

((H) = (#\a A I #\a B))

((A) = (#\a S))

((B) = (#\b S))

Для каждого запоминаем текущую вершину: H, A, B

Далее формируем тройки из текущей вершины, символа перехода и новой вершины. Если встречаем разделитель I, продолжаем создавать тройки с текущей вершиной. Если список заканчивается, то переходим к следующей вершине.

Если у нас нет следующей вершины, то мы добавляем конечную или начальную в зависимости от типа грамматики.

Все тройки мы сохраняем в параметр res.

Для примера в итоге получим: ((H #\a A) (H #\a B) (A #\a S) (B #\b S))

2) Алгоритм детерминизации

Его суть проста:

Пусть p – текущая вершина, c – множество букв(переходов).

Мы хотим найти для каждой буквы из c множество вершин q, в которые мы можем перейти из p.

Каждое q мы называем новой вершиной и повторяем для нее алгоритм.

Пример: ((H #\a A) (H #\a B) (A #\a S) (B #\b S))

Сначала p = H;

C = a => q = (A B)

C = b => q = nil

Далее p = (A B)

C = a => q = q(a A) + a(a B) = (S)

C = b => q = q(b A) + a(b B) = (S)

Какие функции за это отвечают?

F3 – находит q для одного p и c. Добавляет в res вершину, если такая есть.

F2 – находит все q для всех p и одного c. Т. е. для всех c вызывает f1 и сохраняет получившиеся вершины.

D2 - находит все q для всех p и всех c. Т. е. для всех c вызывает f2 и, если полученный список не пуст, то объединяет в тройку текущую вершину, букву перехода и полученное множество.

В нашем примере он последовательно вернет:

((H #\a A) (H #\a B) (A #\a S) (B #\b S))

(((H) #\a (B A)))

(((B A) #\a (S)) ((B A) #\b (S)))

NIL

Функция unt объединяет три элемента в тройку.

Функция help_1 делает вызов d2 короче

3) Сборка

Функции find_1, eq_1, eq_2 – нужны для проверки вхождения списка вершин в список из списков вершин.

Функция task_5 принимает грамматику.

Task_55 создает список букв при помощи set_letters и стартовую вершину при помощи set_St. Затем вызывает M1.

M1 – ищет все тройки для текущей вершины(это может быть список из одной или нескольких вершин, в начале это стартовая вершина) и вызывает M2

M2 – дублирует полученное множество и вызывает M22

M22 – дополняет множество P вершин нового автомата(те, которые мы получили, но еще не посмотрели, куда они ведут) и множество Q просмотренных вершин

M3 – создает новые тройки и, если у нас еще есть не рассмотренные вершины(P), то вызывает M1 от (car P)

Тесты

Пример 1 :

Грамматика: (((H) = (#\a A)) ((A) = (#\a S)))

Построенный автомат: ((H #\a A) (A #\a S))

(((A) #\a (S)) ((H) #\a (A)))

Детерминированный автомат: (((A) #\a (S)) ((H) #\a (A)))

Пример 2:

Грамматика: (((H) = (#\a A | #\a B)) ((A) = (#\a S)) ((B) = (#\b S)))

Построенный автомат: ((H #\a A) (H #\a B) (A #\a S) (B #\b S))

Детерминированный автомат:(((B A) #\b (S)) ((B A) #\a (S)) ((H) #\a (B A)))

Пример 3:

Грамматика: (((S) = (B #\b | C #\c)) ((B) = (C #\b | #\a)) ((C) = (B #\b | #\a))))

Построенный автомат: ((H #\a C) (B #\b C) (H #\a B) (C #\b B) (C #\c S) (B #\b S))

Детерминированный автомат: (((B S C) #\c (S)) ((B S C) #\b (B S C)) ((B C) #\c (S)) ((B C) #\b (B S C)) ((H) #\a (B C)))