МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ ДНР

Донецкий Национальный Технический Университет

КАФЕДРА ПРОГРАМНОЙ ИНЖЕНЕРИИ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту по дисциплине

«Теория алгоритмов и функциональных языков»

на тему: «Построение аналитических моделей алгоритмов и

оценка их сложности»

|  |  |
| --- | --- |
| Руководитель:  кафедры ПИ | Выполнил:  Студент гр. ПИ18б  Куркурин Н.Л. |

Донецк – 2019

2. РАЗРАБОТКА АНАЛИТИЧЕСКОЙ И ПРОГРАММНОЙ МОДЕЛИ АЛГОРИТМА ДЛЯ РАСПОЗНАЮЩЕЙ МАШИНЫ ТЬЮРИНГА

Рассмотрим реализацию машины Тьюринга для языка L = { wwRw∈{a, b, c}\*}, способ реализации в программе, построим график временной сложности. Пустая ячейка на ленте обозначается ‘L’.

2.1 Формальное определение машины, распознающей Тьюринга

На первую ленту подается входное слово.

Суть машины заключается в том, что нужно проверить входное значение на соответствие условию: входное слово, состоящее из алфавита ‘a’, ‘b’, ‘c’ нужно разделить на два равных по длине слова, причем второе слово должно представлять собой реверсию первого. В случае если входные данные удовлетворяют условию задачи программа выводит: ‘1’, иначе ‘0’.

Для реализации данного алгоритма составлена функциональная таблица, представленная на рисунке 2.1.

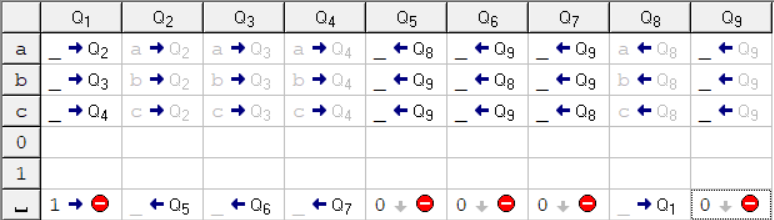


Рисунок 2.1 – Функциональная таблица для алгоритма проверки второго слова на реверс первого

2.2 Протоколы работы машины Тьюринга на двух лентах

Рассмотрим контрольные примеры для машины Тьюринга, реализовывающей работу с данным алфавитом.

Протоколы работы над словами, принадлежащими языку: “aa”, “сbbс” (Табл. 2.2 –Табл. 2.3).

Таблица 2.2

|  |  |
| --- | --- |
| K0 = | LaaL |
| L |
| K1 = | LLaL |
| L |
| K2 = | LLLL |
| L |
| K3 = | LL1L |
| 1 |

Таблица 2.3

|  |  |  |  |
| --- | --- | --- | --- |
| К0= | LcbbcL | K4= | LLLLLL |
| L | L |
| К1= | LLbbcL | K5= | LLL1LL |
| L | 1 |
| К2= | LLbbLL |  | |
| L |
| К3= | LLLbLL |
| L |

Рассмотрим протоколы работы над словами, не принадлежащими языку: “a”, “bc” (Табл. 2.4 –Табл. 2.5).

Таблица 2.4

|  |  |
| --- | --- |
| K0 = | LaL |
| L |
| K1 = | LLL |
| L |
| K2 = | L0L |
| 0 |

Таблица 2.5

|  |  |
| --- | --- |
| K0 = | LabL |
| L |
| K1 = | LLbL |
| L |
| K2 = | LLLL |
| L |
| K3 = | L0LL |
| 0 |

2.3 ПРОГРАММНАЯ МОДЕЛЬ МАШИНЫ ТЬЮРИНГА

В программе две ленты заданы строками, создан абстрактный класс mt от которого наследуется основная машина. В классе основной машины есть переменные:

* letter, в которую записан элемент, на который указывает головка;
* direction, в каком направлении двигаться машине;
* state, в каком состоянии машина Тьюринга;
* cursor, ссылка на ячейку, на которой стоит головка.

В машине Тьюринга есть метод heart, в котором прописана логика работы машины, а именно, это бесконечный цикл, мы переходим в определенное состояние, где меняем или не меняем будущее состояние, букву на которой находимся, направление движения.

Преимущество данной реализации в красоте и понятности кода, а также в производительности и быстродействии.

Для того, чтобы иметь возможность лицезреть график, он спроектирован в отдельном окне с помощью отдельного потока, генератор также сделан в отдельном потоке, с целью не утруждать основной поток в генерации большого кол-ва тестовых слов.

2.4 ПРОТОКОЛЫ РАБОТЫ МАШИНЫ ТЬЮРИНГА, ПОСТРОЕННЫЕ ПРОГРАММНО (НА ДВУХ СЛОВАХ ЯЗЫКА И ДВУХ СЛОВАХ, НЕ ПРИНАДЛЕЖАЩИХ ЯЗЫКУ)

Для вывода листинга пошаговой работы программы был использован файл logging.txt. Рассмотрим пример логирования входного слова abba, удовлетворяющего условию (см. рис. 2.2).

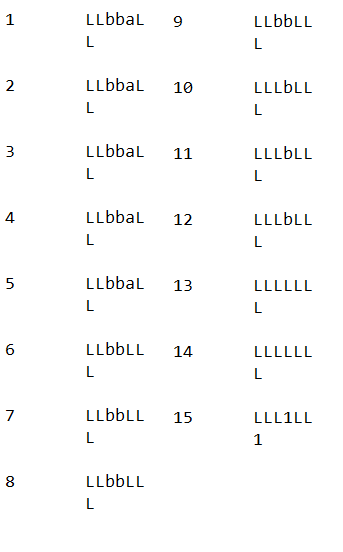


Рисунок 2.2 – Логгинг работы программы над словом “abba”

Рассмотрим слово, не удовлетворяющее условию, слово “cb” (см. рис. 2.3).

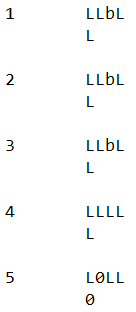


Рисунок 2.3 – Логгинг работы программы над словом “cb”

Рассмотрим слово, удовлетворяющее условию “abccba”. (см. рис. 2.4)

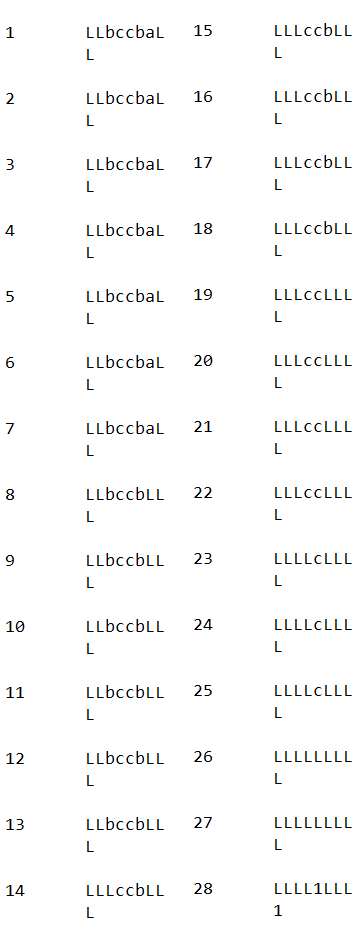


Рисунок 2.4 – Логгинг работы программы над словом “abccba”

Как можно увидеть, алгоритм максимально минимизирован, и выдает либо 1 если слово удовлетворяет условию, либо 0 если слово не удовлетворяет условию.

2.5 РАСЧЕТ ВРЕМЕННОЙ СЛОЖНОСТИ (ГРАФИК ФУНКЦИИ ВРЕМЕННОЙ СЛОЖНОСТИ)

Так же при запуске программы создается два дополнительных потока: один поток выделен для построения графика временной сложности, второй – для генерации бесконечного множества значений, удовлетворяющих условию. Работа графика представлена на рис. 2.5.

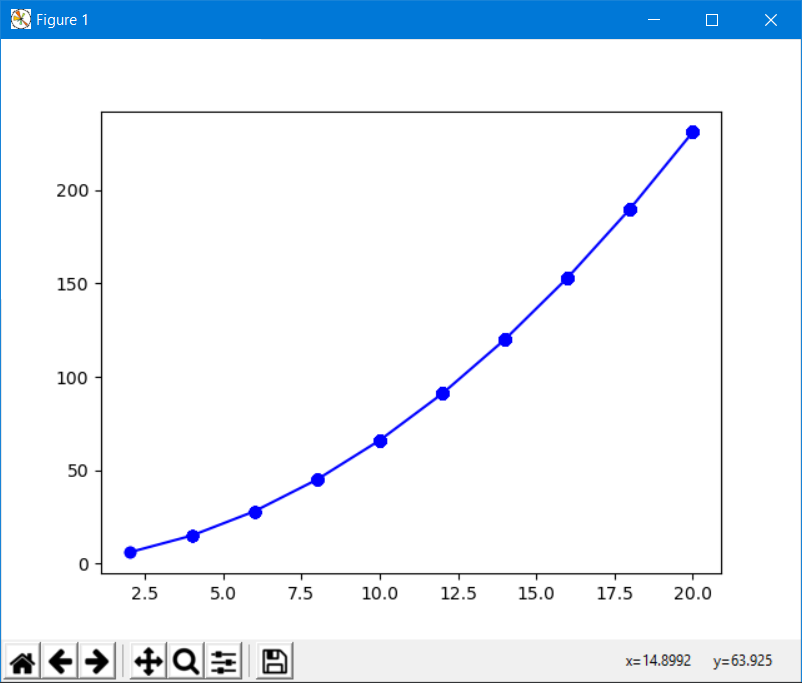


Рисунок 2.5 – График временной сложности