Лабораторная работа № 4 по курсу дискретного анализа: Поиск образца в строке

Выполнил студент группы М8О-208Б-21 *Армишев Кирилл.*

Условие

Кратко описывается задача:

1. Необходимо реализовать один из стандартных алгоритмов поиска образцов для указанного алфавита.
2. *Вариант алгоритма:* Поиск одного образца при помощи алгоритма Апостолико-Джанкарло.
3. *Вариант алфавита:* Слова не более 16 знаков латинского алфавита (регистронезависимые).
4. Запрещается реализовывать алгоритмы на алфавитах меньшей размерности, чем указано в задании.

Метод решения

Алгоритм Апостолико-Джанкарло - это учучшенный алгоритм Бойера-Мура. Он включает в себя все ивристики, которые содержатся в Бойере-Муре, однако он может предвидить результат многих сравнений, которые делает метод Бойера-Мура, и избежать их, за счет внедрения массива M. Как и наивный алгоритм, алгоритм Бойера-Мура последовательно прикладывает паттерн к тексту и проверяет совпадение символов Паттерна с прилежащими символами текста. Когда проверка завершается, P сдвигается вправо точно так же, как в наивном алгоритме. Однако алгоритм Бойера-Мура использует три здравые идеи, которых нет в наивном алгоритме: просмотр справа налево, правило сдвига по плохому символу и правило сдвига по хорошему суффиксу.

Описание программы

Разобьем алгоритм на следующие шаги:

1. Реализация вычисления Z - функции: Z,(S) — это длина наибольшего префикса $S[i..|S|]$, совпадающего с префиксом S
2. Реализация вычисления N - функции x: Реализуется разворотом обратной Z-функции
3. Реализация правила улучшенного правила плохого символа: Проходим по паттерну справа налево и запоминаем индексы вхождения каждого элемента в словарь map

4. Реализация улучшенного правила хорошего суффика: $L(i)$ определяет позицию правого конца крайней правой копии $[i..n]$, которая сама не является суффиксом , а $L'(i)$ — такую же позицию с дополнительным усиливающим условием, что предшествующий символ не равен $P(i-1)$.

5. Реализация поиска подстроки в строке при помощи M-массива с сдвигов, вычисленных через правила плохого символа и хорошего суффика: В начале каждой фазы просматривается элемент вектора M под номером текущей позиции в тексте при поиске. Путём сравнения значений вектора M со значениями N-функции, а также изменением элементов из M при частичном совпадении паттерна с текстом, некоторые сравнения пропускаются. В случае обнаружения (не)совпадения паттерн сдвигается по правилам из алгоритма Бойера-Мура

Весь код содержится в файле **main.c**: в нем реализованы Z-функция, улучшенное правило плохого символа и хорошего суффикса. В int main() происходят сами сравнения элементов паттерна с текстом, используя подсчет сдвига через выше указанные функции и на каждой фазе просматривается элемент вектора M, чтобы не выполнять лишних сравнений. Все функции реализованы согласно определениям и алгоритмам, описанным на лекциях.

Дневник отладки

1. 15 май 2023, 02:13:53 WA на 13 тесте.

   Причина: неправильно высчитал начальную позицию вхождения паттерна в текст, изменил вычисление данной позиции.

Тест производительности

Реализованный алгоритм Апостолико-Джанкарло сравнил с наивным алгоритмом поиска. Замеры производятся на тестах из 10^3 слов, 10^4 или 10^5. Длина образца в первом тесте - 10, во втором - 25, в третьем - 100

```
[armishev src] ./main < tests/01.t
==============START============
Naive: 0.721 ms
Apostol-Janc: 0.033 ms
==============END==============
[armishev src] ./main < tests/02.t
==============START============
Naive: 5.238 ms
Apostol-Janc: 0.631 ms
==============END==============
[armishev src] ./main < tests/03.t
```

```
==============START============
Naive: 23.321 ms
Apostol-Janc: 2.321 ms
==============END==============
```

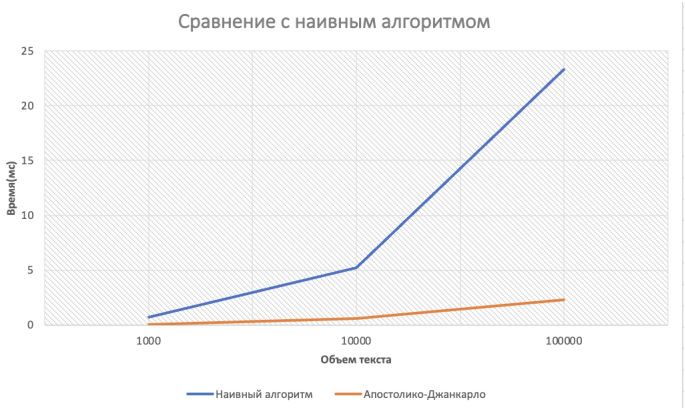

Рис. 1: Сравнение Апостолико-Джанкарло с наивным алгоритмом

Видно, что наивный алгоритм поиска работает намного медленнее чем алгоритм Апостолико-Джанкарло. Классический алгоритм допускает лишние сравнения на этапе поиска образца в тексте, а алгоритм Апостолико-Джанкарло нет. Обработка таких сравнений длиться дольше, чем предпроцессинг для функций, использующихся в Апостолико-Джанкарло.

Выводы

Проделав данную лабораторную работу, я изначально реализовал алгоритм Бойера-Мура, отладил его, а уже потом реализовал алгоритм Апостолико-Джанкарло. При выполнении работы, кроме того, что реализовал классные алгоритмы, я научился лучше отлаживать свой код. Например, открыл для себя возможность сравнить скорость и результаты своего алгоритма с наивным его вариантом, обычно работающим значительно медленнее, но в реализации которого сложнее налажать.