Київський національний університет імені Тараса Шевченка

Факультет комп`ютерних наук та кібернетики

Кафедра інтелектуальних інформаційних систем

Алгоритми та складність

Лабораторна робота №7

“Узагальнений метод Рабіна-Карпа”

Виконав студент 2-го курсу

Групи К-28

Гуща Дмитро Сергійович

2020

**Завдання**

Узагальніть метод Рабіна-Карпа пошуку зразка в текстовому рядку так, щоб він дозволив розв’язати задачу пошуку заданого зразка розміром m на m у символьному масиві розміром n на n. Зразок можна рухати по горизонталі та вертикалі, але не обертати.

**Теорія**

Алгоритм Рабіна-Карпа, який виконує пошук підрядка в рядку узагальнений до пошуку підматриці m x m в матриці символів n x n. Цей алгоритм використовує спосіб хешування. За допомогою математичних операцій, зокрема ділення з остачею знаходяться збіги, і далі порівнянням перевіряється істинність розв’язку. Алгоритм в середньому працює за Θ(n2), оскільки працює з матрицями {\displaystyle O(N^{3})=O(N^{\log \_{2}8})}

**Алгоритм**

Спочатку розбиваємо матрицю m x m на рядочки. Потім додаємо ці рядочки до словника, тоді, використовуючи хешування, ділення з остачею та інші математичні операції виконуємо пошук з перекриттям. Далі маємо викреслити зі списку знайдених ті, чиї позиції не починаються у вузлах підсітки m x m, накладеної на сітку n x n( в першому стовпчику). Далі необхіднопереконатися, що залишилося рівно m рядочків у вузлах з кроком n, перевіряємо розв’язки на істинність і маємо результат

**Складність**

Алгоритм працює з матрицями, тому має середню складність Θ(n2)= Θ(n+1)2, фактично ускладнивши вдвічі звичайний алгоритм Рабіна-Карпа

**Мова програмування**

С++

**Модулі програми**

void swap(int\* a, int\* b)  
//Змінює елементи місцями  
int partition(int\* arr, int low, int high)  
//Поділ масиву на підмасиви для сортування  
void quick\_sort(int\* arr, int low, int high)  
//Швидке сортування  
int rand\_int(int min, int max)  
//Генерація випадкових даних  
void create\_source(int size)  
//Заповнення випадковими даними наших матриць  
int array\_sum(const int\* array)  
//Сума елементів масиву  
int max\_index(const int\* array)  
//Індекс максимального елемента  
int min\_index(const int\* array)

//Індекс мінімального елемента  
int min\_positive\_index(const int\* array)  
//Індекс мінімального додатнього елемента  
bool all\_int\_max(const int\* array)  
//Перевіряє, чи всі елементи максимальні  
struct Node  
//Вузол зв’язного списку  
void polyphase\_sort()  
//Поліфазне сортування у файлах

**Інтерфейс користувача**

Вхідні дані, згенеровані за допомогою функції rand() із 100 елементів беруться із файлу, результат роботи алгоритму записується в файл

**Тестові приклади**

|  |  |
| --- | --- |
| **input** | **output** |
| Дві символьні матриці  Шаблон:  G K  M N  Зразок:  M R H E  G K T V  M N W Q  G K T N | M R H E  **G K** T V  **M N** W Q  G K T N |

Підматриця розбивається на рядки GK та MN, далі дані хешуються і виконується пошук, як у звичайному алгоритмі Рабіна-Карпа. У першому рядку не знаходимо нічого, зате знаходимо GK у другому та четвертому рядках.

Далі маємо перевірити відповідно 3 і 5 рядки, але оскільки 5 немає, то лише 3 рядок. У 3 рядку знаходимо MN, і бачимо, що їх позиції по стовпчиках відповідають позиціям по стовпчикам 0, 1 для GK у 2 рядку. Маємо потенційне входження, розкодовуємо символи, звіряємо і переконуємося, що ми дійсно знайшли входження шаблону в матриці.

**Висновки**

Алгоритм Рабіна-Карпа узагальнено для матриць і він працює в 2 рази довше за звичайний, тобто за Θ(n2), що цілком нормально для роботи з матрицями

**Література**

* Лекція №10
* <https://www.cyberforum.ru/cpp-beginners/thread1252591.html>
* <https://uk.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%A0%D0%B0%D0%B1%D1%96%D0%BD%D0%B0_%E2%80%94_%D0%9A%D0%B0%D1%80%D0%BF%D0%B0>