Київський національний університет імені Тараса Шевченка

Факультет комп`ютерних наук та кібернетики

Кафедра інтелектуальних інформаційних систем

Алгоритми та складність

Завдання №1

“ Багатофазне сортування злиттям”

Виконав студент 2-го курсу

Групи К-28

Самойлич Євгеній Костянтинович

2020

**Завдання**:

Реалізуйте багатофазне сортування злиттям.**Теорія**

Реалізуємо багатофазне сортування злиттям.

**Алгоритм**

Алгоритм сортування злиттям складається з трьох кроків:

• поділ: послідовність, що сортується і складається з n елементів, розбивається на дві менші по n/2 елементів кожна. При цьому рекурсія закінчується, коли послідовність містить єдиний (і очевидно відсортований) елемент

• підкорення: сортування обох допоміжних підпослідовностей шляхом злиття

• комбінування: злиття двох відсортованих підпослідовностей для отримання кінцевого результату

**Складність**

На кожному кроці алгоритму знаходиться задача розбивається на дві підзадачі меншого розміру. Найгірший, кращий та середній випадки будуть виконуватись за однаковий час, так як ми в будь-якому разі розбиваємо задачу на дві рівні підзадачі. Складність буде *O(n∙*log *n)* та *O(n)* затрати на додаткову пам’ять.

**Мова програмування**

С++

**Модулі програми**

* void merge\_sort(const string& file\_name)

Рекурсивна функція, яка сортує дані з файлу file\_name. На вхід приймає файл file\_name, який розбивається на два підфайли з однаковою кількістю елементів, для кожного з яких рекурсивно викликається ця ж функція

* void split\_file\_in\_two(const string& sourse, const string& filename1, const string& filename2)

Функція, яка розділяє масив даних з файлу *source* порівну між файлами *filename1* та *filename2*

* bool check\_file(const string& file\_name)

Функція, яка перевіряє чи наявно у файлі більше 1 елементу, тобто чи потрібно продовжувати розподіл.

* void merge(const string& sourse, const string& filename1, const string& filename2)

Функція, яка виконує злиття даних з файлів filename1 та filename2 у файл sourse. На кожному кроці ми беремо менший з двох елементів та записуємо його у результуючий файл.

**Інтерфейс користувача**

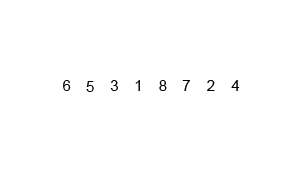
Вхідні дані вводяться з текстового файлу і виводяться також в текстовий файл.

**Тестові приклади**

|  |  |
| --- | --- |
| **input** | **output** |
| Number of elements: 8  Array: 6 5 3 1 8 7 2 4 | Sorted array: 1 2 3 4 5 6 7 8 |

Нехай масив ={6, 5, 3, 1, 8, 7, 2, 4}. Спочатку розбиваємо масив на два підмасиви {6, 3, 8, 2} та {5, 1, 7, 4}, потім розбиваємо ці два масиви на ще чотири {6, 8}, {3, 2}, {5, 7} та {1, 4} після ще однієї ітерації отримуємо вісім підмасивів {6},{8},{3},{2},{5},{7},{1},{4}. Потім виконуємо злиття {6, 8},{2, 3},{5, 7},{1, 4}, потім зливаємо у {2, 3, 6, 8}, {1, 4, 5, 7} після останнього злиття врешті решт отримуємо відсортований масив {1, 2, 3, 4, 5, 6, 7, 8}.

Gif-анімація прикладу сортування злиттям з WikipediA



**Висновки**

При практичному застосуванні алгоритму складність, буде *O(n∙*log *n)* та буде використовувати *O(n)* додаткової пам’яті.

**Література**

* Лекція №3
* <http://algolist.manual.ru/sort/faq/q13.php>
* https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0\_%D1%81%D0%BB%D0%B8%D1%8F%D0%BD%D0%B8%D0%B5%D0%BC