Лабораторна робота № 5

Тема: програмування та відлагодження алгоритму внутрішнього сортування методом злиття. Визначення складності алгоритму та часу його виконання.
Мета: проаналізувати та дослідити алгоритм сортування шляхом злиття.

Основні теоретичні відомості

1 Сортування методом злиття

Існує багато стандартних прийомів, які використовуються під час побудови алгоритмів. Наприклад, сортування вставками застосовує алгоритм, що діє покроково: ми додаємо елементи один за одним до відсортованої частини масиву. У цій лабораторній роботі розглянемо інший підхід, який називають „*розділяй і володарюй”*, і побудуємо за його допомогою ***значно швидший алгоритм сортування***.

Принцип „розділяй і володарюй”. Багато алгоритмів за своєю природі рекурсивні, тобто розв’язуючи деяку задачу, вони викликають самих себе для розв’язання її підзадач. Ідея принципу „*розділяй і володарюй*” полягає саме в цьому. Спочатку задача розбивається на декілька підзадач меншого розміру. Потім ці задачі розв’язуються (за допомогою рекурсивного виклику або безпосередньо, якщо розмір досить малий). Нарешті, їх розв’язки комбінуються і ми отримуємо шуканий розв’язок вихідної задачі [1 – 4].

Для задачі сортування ці три етапи виглядають так. Спочатку ми розбиваємо масив на дві половини меншого розміру. Потім ми сортуємо кожну з половин окремо. Після цього нам залишається з’єднати два впорядкованих масиви половинного розміру в один. Рекурсивне розбивання задачі на менші відбувається доти, поки розмір масиву не стане одиничним (будь-який масив розміром 1 можна вважати впорядкованим) [1].

Нетривіальним етапом є з’єднання двох впорядкованих масивів в один. Воно виконується за допомогою допоміжної процедури $\mathrm{MERGE}(A,p,q,r)$. Параметрами цієї процедури є масив A і числа p, r, q, що вказують границі ділянок, які зливаються. Процедура передбачає, що $p \le q < r$, а ділянки $A[p...q]$ та $A[q+1...r]$ вже відсортовані, і зливає їх в одну ділянку $A[p...r]$.

Зрозуміло, що час роботи процедури MERGE є $\theta(n)$, де n – загальна довжина ділянок, що зливаються $n = r - p + 1$. Це легко пояснити на звичайних гральних картах. Нехай ми маємо дві стопки карт (сорочкою

вниз), і у кожній карти йдуть зверху вниз у порядку зростання. Для об'єднання цих стопок в одну на кожному кроці ми беремо меншу із двох верхніх карт і кладемо її (сорочкою вверх) у результуючу стопку. Коли одна з вихідних стопок стає порожньою – ми додаємо всі карти, що залишилися другої стопки до результуючої стопки. Отже, кожен крок вимагає обмеженого числа дій, і загальне число дій становить $\theta(n)$ [1].

Тепер напишемо процедуру сортування злиттям MERGE-SORT(A,p,r), яка сортує ділянку $A[p...r]$ масиву A, не змінюючи іншої його частини. При $p \geq r$ ділянка містить максимум один елемент, і, отже є відсортованою. У протилежному випадку ми шукаємо число q, що ділить ділянку на дві приблизно рівні частини $A[p...r]$ (містить $\lceil n/2 \rceil$ елементів) і $A[q+1...r]$ (містить $\lfloor n/2 \rfloor$ елементів). Нагадаємо, що через $\lfloor x \rfloor$ ми позначаємо цілу частину x (найбільше ціле число, яке менше або дорівнює x), а через $\lceil x \rceil$ – найменше ціле число, яке > або = x.

MERGE-SORT (A,p,r)
1. **if** $p < r$
2. **then** $q \leftarrow \lfloor (p+r)/2 \rfloor$
3. MERGE-SORT (A,p,q)
4. MERGE-SORT $(A,q+1,r)$
5. MERGE (A,p,q,r)

Весь масив тепер можна відсортувати за допомогою виклику MERGESORT $(A,1,length[A])$. Якщо довжина масиву $n=length[A]$ є степенем двійки, то у процесі сортування відбудеться злиття пар елементів у відсортовані ділянки довжиною *2*, потім злиття пар таких ділянок у відсортовані ділянки довжиною *4* і так далі до n (на останньому кроці з'єднуються дві відсортованих ділянки довжиною $n/2$). Цей процес наведено на рисунку 1.

Аналіз алгоритмів типу „розділяй і володарюй". Для того, щоб оцінити час роботи рекурсивного алгоритму ми повинні враховувати час, який витрачається на рекурсивні виклики. Отже, отримаємо деяке *рекурентне співвідношення*, виходячи з якого й можемо оцінити час роботи алгоритму.

Припустимо, що алгоритм розбиває задачу розміром n на a підзадач, кожна з яких має в b разів менший розмір. Будемо вважати, що розбиття потребує часу $D(n)$, а з'єднання отриманих рішень – часу $C(n)$. Тоді

одержуємо співвідношення для часу роботи *T(n)* на задачах розміром *n* (у найгіршому випадку):

$$T(n) = aT(n/b) + D(n) + C(n).$$

Це співвідношення виконується для достатньо великих *n*, коли є смисл розбивати задачу на підзадачі. Для малих *n*, коли таке розбивання неможливе або не потрібне, застосовується деякий прямий метод розв'язання задачі. Оскільки *n* обмежене – час роботи також не перевищує деякої константи.

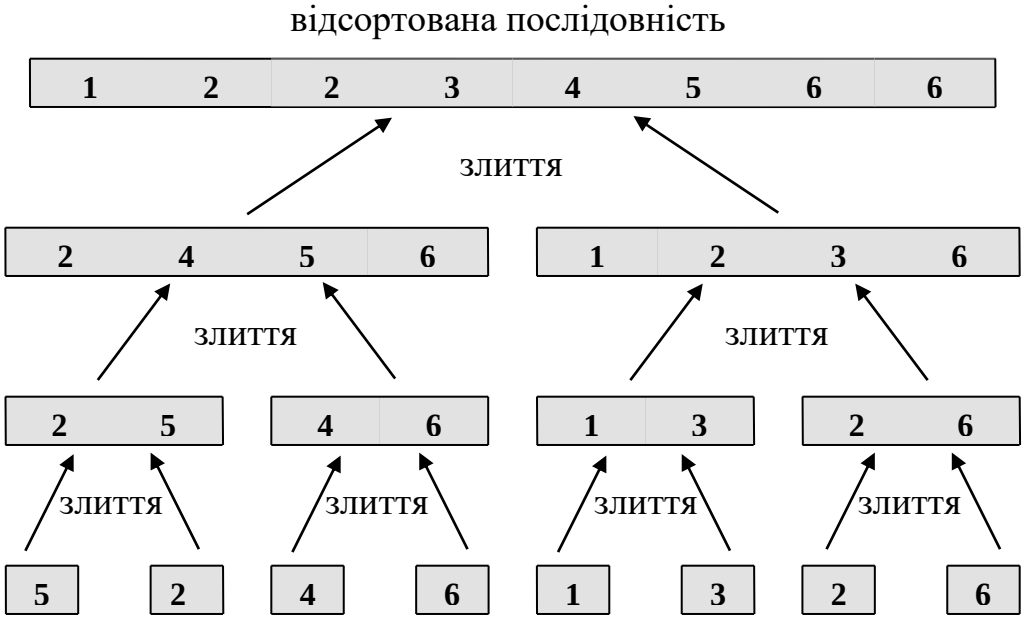

Рисунок 1 – Сортування злиттям для масиву $A\{5,2,4,6,1,3,2,6\}$

Аналіз алгоритму сортування злиттям. Для простоти припустимо, що розмір масиву є степінь двійки. Тоді на кожному кроці ділянка, що підлягає сортуванню ділиться на дві рівні половини. Розбиття на частини (обчислення границі) вимагає часу $\theta(1)$, а злиття – часу $\theta(n)$. Таким чином, одержуємо співвідношення

$$T(n)\begin{cases}\theta(1), & \text{якщо } n = 1;\\ 2T(n/2) + \theta(n), & \text{якщо } n > 1.\end{cases}$$

Отже, розв'язуючи дане співвідношення (детальний аналіз співвідношень та методів їх розв'язання наводиться у наступній лабораторній роботі) приходимо до такої оцінки: $T(n)=\theta(n\cdot log\, n)$, де через *log* ми позначаємо двійковий логарифм (основа логарифмів не відіграє ролі, тому що приводить лише до зміни константи). Тому для більших *n* сортування злиттям ефективніше сортування вставками, яке вимагає часу $\theta(n^2)$.

Контрольні питання

1. Поясніть чому задача сортування елементів є однією з найцікавіших та показових задач для курсу теорії алгоритмів.
2. Поясніть, що таке стійкість алгоритму сортування.
3. За якими критеріями на Ваш погляд можна класифікувати алгоритми сортування?
4. Наведіть класифікацію алгоритмів сортування.
5. Перерахуйте та порівняйте відомі Вам алгоритми сортування за псевдолінійний час.
6. Поясніть чому при оцінці складності алгоритму нас частіше за все цікавить робота у найгіршому випадку.
7. Виконайте детальний аналіз алгоритму сортування методом злиття.
8. Наведіть переваги та недоліки алгоритму сортування методом злиття.

Звіт повинен містити

1. Тему, мету та порядок виконання роботи.
2. Ідея відповідного алгоритму сортування.
3. Власний приклад роботи відповідного алгоритму сортування на масиві з 10 чисел.
4. Алгоритм сортування у графічному вигляді.
5. Теоретична оцінка складності алгоритму для трьох випадків.
   5.1 для найкращого випадку.
   5.2 для найгіршого випадку.
   5.3 для середнього випадку.
6. Сирець алгоритму відповідного сортування.
7. Практична оцінка складності відповідного алгоритму в трьох випадках для масивів різного розміру. Навести таблиці та відповідні графіки часу виконання програми для трьох випадків. Кількість значень часу має бути не менше семи.
   Увага! Мінімальне та максимальне значення часу ***кожним студентом підбирається індивідуально****, залежно від потужностей Вашого*

ПК. При цьому врахуйте, що мінімальне значення часу повинно бути порядка кількох секунд (до 10), а максимальне – не більше 5 – 10 хвилин (за Вашим бажанням останній час може бути збільшено).

7.1 Таблиця та графік для найкращого випадку.

7.2 Таблиця та графік для найгіршого випадку.

7.3 Таблиця та графік для середнього випадку.

8. Порівняльний аналіз теоретично та практично отриманих графіків залежностей часів виконання програми від розмірів входу.
9. Переваги та недоліки дослідженого Вами алгоритму сортування та Ваші міркування.
10. Розширені висновки з роботи.