Київський національний університет імені Тараса Шевченка

Факультет комп`ютерних наук та кібернетики

Кафедра інтелектуальних інформаційних систем

Алгоритми та складність

Завдання № 2

**Болти і гайки**

Виконала студентка 2-го курсу

Групи К-28

Панченко Тетяна Андріївна

2020

**Завдання**

Нехай є n болтів різного розміру та n відповідних гайок. Можна порівнювати, чи підходять гайка і болт одне до одного, або гайка більша чи менша. Порівнювати між собою дві гайки чи два болти не можна. Розробіть і реалізуйте алгоритм розбивки всіх гайок і болтів на відповідні пари за час Θ(n log n).

**Теорія**

Зафіксуємо гайку і розділимо всі болти на три групи:

а) болт, що підходить гайці (саме цю пару ми перенесемо в розв’язок.)

б) болти, менші за розміром.

в) болти, більші за гайку.

Візьмемо цей болт і порівняємо з наступною гайкою. Потім так само знайдемо болт для іншої гайки.

Для отримання заданої складності Θ(nlogn) використаємо швидке сортування.

**Алгоритм**

Швидке сортування належить до алгоритмів типу «розподіляй та владарюй»

Поділ:

Масив A[p..r] розбивається шляхом перерозподілу елементів на (можливо порожні) підмасиви A[p..(q–1)] та A[(q+1).. r]. При цьому кожен елемент A[p..(q–1)] не перевищує A[q], а кожен елемент A[(q+1)..r] не менше A[q]. Індекс q обчислюється в ході розбиття.

Підкорення:

Рекурсивно сортуються масиви A[p..(q–1)] та A[(q+1)..r].

Комбінування:

Підмасиви сортуються на місці, об’єднання не потрібне: весь масив A[p..r] буде відсортований.

**Складність**

На кожному кроці алгоритму знаходиться одна пара (болт, гайка), і задача розбивається на дві підзадачі меншого розміру (логарифм)

Найгірший випадок відбудеться, коли масив гайок буде спадний. В такому випадку складність буде О(n2).

В найкращому і середньому випадку складність буде Θ(n log n)

**Мова програмування**

С++

**Модулі програми**

Функція сортування масиву В:

template<typename T1, typename T2>

void quick\_sort(T1\* A, T2\* B, int down, int up)

Функція перерозподілу:

int partition(T1\* A, T2\* B, int down, int up)

Генерація масивів:

template<typename T1, typename T2>  
 void init(Node<T1>\* A, Node<T2>\* B, int size)   
  
 int rand\_int(int min, int max) - допоміжна ф-я для init(…), що генерує випадкове значення в межах [min; max]

Виведення в консоль:  
 template<typename T>  
 void print\_array(T\* A, int size)

**Інтерфейс користувача**

Вхідні дані вводяться генеруються випадковим чином та виводяться в консоль.

**Тестові приклади**

Дано:

Болти: 2 5 4 1 3

Гайки: 1 5 3 2 4

На першому кроці алгоритму гайки розміщуються так, щоб усі елементи, лівіші за деякий p будуть менші за 2(перший болт), правіше будуть розміщені ті, що більші або рівні.

Отримаємо {Гайки: 1 2 3 4 5}, p = 1. Рекурсивно застосовуємо алгоритм для утворених підмасивів [0…1] і [2...4]...

Результат:

Болти: 1 2 3 4 5

Гайки: 1 2 3 4 5

**Висновки**

При реалізації алгоритма були враховані усі умови і отримана задана складність О(n log n)

**Література**

1. Лекція 5
2. <https://en.wikipedia.org/wiki/Quicksort>
3. https://uk.wikipedia.org/wiki/%D0%A8%D0%B2%D0%B8%D0%B4%D0%BA%D0%B5\_%D1%81%D0%BE%D1%80%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F