Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

**Лабораторна робота *№2***

із дисципліни ***«Теорія алгоритмів»***

Тема: ***«Метод декомпозиції. Пошук інверсій»***

**Виконали:**

Студенти групи ІА-34

Янович Марія,

Ковальчук Станіслав,

Ястремський Богдан,

Сухоручкін Гліб

**Перевірив:**

Степанов Андрій Сергійович

Київ — 2024

**Завдання:**

Існує веб сервіс, який надає своїм користувачам можливість перегляду фільмів онлайн. Періодично система надає нові рекомендації користувачам — які фільми, що їх користувач ще не дивився, можливо будуть йому або їй цікаві.

В основі рекомендаційного алгоритму лежить ідея, що користувачі, які подивились однакові фільми та також оцінили їх схожим чином, мають схожі смаки. Наприклад, нехай є два користувача: Аліса та Богдан. Обидва вони переглянули наступні фільми: “Зоряні війни”, “Гравітація”, “Пірати карибського моря”, “Володар перснів”, “Матриця”.

Спочатку система просить користувачів оцінити ці фільми і розташувати їх у порядку вподобання, іншими словами — створити власний хіт-парад. Так Аліса розташувала вказані фільми у порядку від найбільш до найменш вподобаного: “Пірати карибського моря”, “Володар перснів”, “Матриця”, “Гравітація”, “Зоряні війни”. Хіт-парад Богдана: “Зоряні війни”, “Володар перснів”, “Гравітація”, “Матриця”, “Пірати карибського моря”.

Після цього система може надати кількісну оцінку наскільки схожими є смаки двох користувачів. Для цього використовується алгоритм підрахунку інверсій поміж двома масивами.

Нехай A[1..n] — масив з n чисел. Якщо i<j та A[i]>A[j], то пара (i, j) — інверсія в A.

Нехай A[1..n] — масив з n чисел. Якщо i<j та A[i]>A[j], то пара (i, j) — інверсія в A. Щоб звести задачу порівняння двох хіт-парадів до задачі підрахунку інверсій у нашому прикладі, побудуємо два масиви A та B. Масив A = [1, 2, 3, 4, 5]. Масив B будується наступним чином: елементом B[j] є число, яке відповідає позиції фільму в хітпараді Богдана, який в хіт-параді Аліси посідав місце j. Наприклад, j = 1 у хіт-параді Аліси відповідає фільму “Пірати карибського моря”. Цей фільм в списку Богдана стоїть на позиції 5, тому B[1] = 5. Загалом отримуємо масив B = [5, 2, 4, 3, 1].

Масив B = [5, 2, 4, 3, 1] має наступні інверсії (вказуються індекси елементів, а не їх значення): (1,2), (1,3), (1,4), (1,5), (2,5), (3,4), (3,5), (4,5). Загалом 8 інверсій. І це число вказує наскільки сильно відрізняється список вподобань Аліси від списку вподобань Богдана. Ми порахували віддаленість списку Аліси від списку Богдана. Якщо порахувати цю відстань в іншому напрямку, то чи буде вона такою самою? Тобто визначити кількість інверсій в списку Аліси по відношенню до списку Богдана.

Сервіс перегляду фільмів онлайн має базу даних D вподобань користувачів. Ця база є матрицею.

Рядки цієї матриці відповідають користувачам, а стовпці — фільмам. Її розмірність u\*m, де u — це кількість користувачів, m — кількість фільмів. Кожний елемент матриці D[i, j] вказує на позицію фільму j в списку вподобань користувача i. Для спрощення припускаємо, що всі користувачі переглянули всі фільми.

Тепер щоб визначити наскільки подібні смаки деякого користувача x до смаків інших користувачів, система попарно порівнює списки вподобань x та всіх інших користувачів i не дорівнює x: за вказаним вище принципом підраховується кількість інверсій у масиві D[x] відносно масиву D[i].

Визначене число інверсій буде кількісною оцінкою наскільки смаки x є близькими до смаків кожного i — чим менше значення цього числа, тим більш подібними є смаки двох користувачів.

**Формальна постановка задачі:**

За допомогою методу декомпозиції розробити алгоритм, який буде розв’язувати наступну задачу.

Вхідні дані. Матриця D натуральних чисел розмірності u\*m, де u — ці кількість користувачів, m — кількість фільмів. Кожний елемент матриці D[i, j] вказує на позицію фільму j в списку вподобань користувача i. Іншим вхідним елементом є x — номер користувача, з яким будуть порівнюватись всі інші користувачі.

Вихідні дані. Список з впорядкованих за зростанням другого елементу пар (i, c), де i — номер користувача, c — число, яке вказує на степінь схожості вподобань користувачів x та c (кількість інверсій).

**Формат вхідних/вихідних даних:**

Розроблена програма повинна зчитувати вхідні дані з файлу заданого формату та записувати дані у файл заданого формату. У вхідному файлі зберігається матриця вподобань всіх користувачів D.

Номер користувача Х, з яким відбувається порівняння всіх інших користувачів, передається аргументом виклику програми через командний рядок.

Вхідний файл представляє собою текстовий файл із U+1 рядків. Перший рядок містить два числа: U та M, де U — кількість користувачів, M — кількість фільмів. Кожен наступний рядок представляє список вподобань (хіт-парад) фільмів відповідних користувачів і містить M+1 число, розділених пробілом. Перше число в рядку є номером користувача (від 1 до U). Решта M чисел є номерами фільмів 1 ,..., M у хіт-параді відповідного користувача.

Вихідний файл представляє також текстовий файл із U рядків. Перший рядок містить одне число — номер користувача Х, з яким відбувалось порівняння всіх інших користувачів. Далі йде U-1 рядків, кожен з яких містить два числа через пробіл: номер користувача i та число c, яке визначає степінь подібності списків вподобань користувачів x та і. Рядки з парами i та c впорядковані за значенням елементу c.

До документу завдання також додаються приклади вхідних і вихідних файлів різної розмірності.

Нижче наведені приклади вхідного та вихідного файлу для U = 10 та M = 5 і користувача X = 6.

|  |  |
| --- | --- |
| Вхідний файл | Вихідний файл |
| 10 5 | 6 |
| 1 5 2 1 3 4 | 3 3 |
| 2 3 2 4 1 5 | 2 4 |
| 3 4 5 3 2 1 | 7 4 |
| 4 5 1 4 3 2 | 9 4 |
| 5 1 2 5 4 3 | 5 5 |
| 6 2 5 4 1 3 | 1 7 |
| 7 2 4 5 3 1 | 4 7 |
| 8 5 3 1 4 2 | 8 7 |
| 9 4 5 2 3 1 | 10 8 |
| 10 3 1 2 4 5 | 6 |

**Вимоги до програмного забезпечення:**

Розробляти програму можна на одній з наступних мов програмування: C/C++ (версія C++11), C# (версія C# 5.0), Java (версія Java SE 8), Python.

Програма повинна розміщуватись в окремому висхідному файлі, без використання додаткових нестандартних зовнішніх модулів.

Не дозволяється використовувати будь-які нестандартні бібліотеки та розширення. Програма не повинна залежати від операційної системи.

Не реалізуйте жодного інтерфейсу користувача (окрім командного рядку). Програма не повинна запитувати через пристрій вводу в користувача жодної додаткової інформації. Вашу програму будуть використовувати виключно у вигляді “чорного ящику”.

Назва висхідного файлу вашої програми повинна задовольняти наступному формату: НомерГрупи\_ПрізвищеСтудента\_НомерЗавдання.Розширення, де НомерГрупи — це один з рядків is01, is02, is03; ПрізвищеСтудента — прізвище студента записане латинськими літерами; НомерЗавдання — двозначний номер завдання (01, 02, ...);

Розширення — розширення файлу, відповідно до мови програмування (.c, .cpp, .cs, .java, .py). Приклад назви висхідного файлу: is31\_ivanenko\_01.cs.

Розроблена програма повинна зчитувати з командного рядку назву вхідного файлу та записувати результат у вихідний файл. При запуску першим і єдиним аргументом командного рядку повинна бути назва вхідного файлу (наприклад, input\_10.txt). Назва вихідного файлу повинна складатись із назви файлу самої програми разом із суфіксом “\_output” і мати розширення .txt. Приклад назви вихідного файлу: is01\_ivanenko\_01\_output.txt.

**Нараховані бали:**

За успішне виконання даного завдання нараховується 5 балів.

ЯКЩО ПРОГРАМИ І ВХІДНІ ДАНІ БУДУТЬ ПОВТОРЮВАТИСЬ У ДЕКІЛЬКОХ СТУДЕНТІВ – бали нараховуються тільки першому з них, іншому 0 балів.

Увага! В даному завданні вимагається використання методу декомпозиції для розробки алгоритму. У разі не виконання цієї вимоги завдання не буде враховуватись виконаним.

Можна отримати ще один додатковий бал за ефективне алгоритмічне рішення: коли алгоритм є компактним та елегантним. Цей бал враховується тільки у випадку вчасної здачі програми (коли ще не нараховуються штрафні бали), першим 6 студентам.

**Виконання завдання:**

1. Псевдокод алгоритму:

FUNCTION find\_diff(input, x)

b = {}

result = {}

FOR i FROM 0 TO LENGTH(input[x-1]) - 1

b[i] = input[x-1][i] - 1

FOR i FROM 0 TO LENGTH(input) - 1

IF i + 1 EQUALS x THEN

CONTINUE

END IF

current = {}

FOR ii IN SORTED(KEYS(b))

current[b[ii]] = input[i][ii]

END FOR

current = SORTED\_VALUES(current)

l = LENGTH(current)

c = 0

FOR ii FROM 0 TO l - 1

FOR iii FROM ii + 1 TO l - 1

IF current[ii] > current[iii] THEN

c = c + 1

END IF

END FOR

END FOR

result[i + 1] = c

END FOR

RETURN SORTED(result)

END FUNCTION

1. Посилання на код:

<https://github.com/KPI-IA-34-Team-8/TA_2ndLab/blob/main/ia34_sukhoruchkin_yanovych.py>

**Висновки:**

При виконанні даної лабораторної роботи ми проаналізували запропоновані нам алгоритми і методи. У ході аналізу було написано код, використовуючи мову програмування Python, для реалізації можливостей певних алгоритмів та методів. Ми створили рейтинг людей на основі їх вподобань: ми мали визначити, наскільки їх вподобання схожі (їх було задано задано за допомогою числової матриці. Після роботи з такими алгоритмами як: методу декомпозиції та алгоритму сортування злиттям, можемо зробити деякі висновки. Алгоритм сортування злиттям використовується для ефективного підрахунку інверсій у масиві чисел та має асимптотичну складність O(n \* log n), де n - кількість елементів у масиві. Метод декомпозиції використовується для вирішення одного великого завдання рішенням серії менших завдань, нехай і взаємопов'язаних, але більш простих. Отже, після закінчення виконання роботи ми здобули нові навички та знання у сфері теорії алгоритмів, а саме навчились користуватись алгоритмом сортування злиттям та методом декомпозиції для сортування одиниць списку у порядку за заданим параметром. Ми мали змогу відпрацювати ці вміння та застосувати нову інформацію як на практиці, так і в теорії.