Звіт

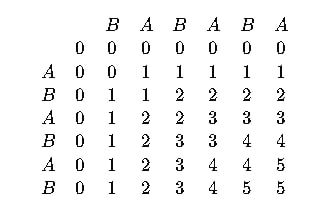
Практична робота №7

Варіант 9

Тема: Алгоритми на рядках

Постановка завдань: Маємо дві короткі послідовності символів: «ABABAB» і «BABABA». Знайти найдовшу спільну підпослідовності символів, використовуючи алгоритм динамічного програмування

Розвязок



dp[6][6] = 5 - символ A з обох рядків

dp[5][5] = 4 - символ B з обох рядків

dp[4][4] = 3 - символ A з обох рядків

dp[3][3] = 2 - символ B з обох рядків

dp[2][2] = 1 - символ A з обох рядків

Відповідь ABABA

Відповідь на контрольні питання

1. Задача знаходження найдовшої спільної підпослідовності (LCS):

Полягає у знаходженні найдовшої послідовності символів, яка є спільною для двох заданих рядків, зберігаючи порядок символів, але не обов'язково послідовність.

2. Головні методи для знаходження LCS:

- Алгоритм динамічного програмування.

- Алгоритм Хаббарда.

- Рекурсивні методи з мемоізацією.

- Методи з використанням хвильових ліній.

3. Алгоритм динамічного програмування для LCS:

- Створюється матриця розміром (n+1) x (m+1) для двох рядків довжиною n і m.

- Заповнюється матриця, де dp[i][j] представляє довжину LCS для підрядків рядків до i і j.

- Якщо символи збігаються, значення збільшується на 1, якщо ні - береться максимум з лівої або верхньої комірки.

- Відновлюється LCS, рухаючись від правої нижньої комірки до лівої верхньої, додаючи спільні символи до результату.

4. Алгоритм Хаббарда для LCS:

- Використовує ідею хвильових ліній, зокрема для скорочення кількості порівнянь.

- Працює з групами символів, оптимізуючи пошук спільних послідовностей.

- Алгоритм менш поширений і складніший у реалізації порівняно з динамічним програмуванням.

5. Переваги та недоліки алгоритмів для LCS:

- Динамічне програмування:

- Переваги: Простота реалізації, передбачуваність часу виконання.

- Недоліки: Високе споживання пам'яті (O(n\*m)).

- Алгоритм Хаббарда:

- Переваги: Може бути швидшим і ефективнішим у певних випадках.

- Недоліки: Складність реалізації, потребує додаткових оптимізацій.

6. Практичні застосування для задачі LCS:

- Біоінформатика: Порівняння генетичних послідовностей.

- Текстове порівняння: Порівняння різних версій документів.

- Комп'ютерна лінгвістика: Аналіз схожості текстів.

- Файлові системи: Визначення відмінностей між файлами.

- Стиснення даних: Виявлення шаблонів у даних для оптимізації стиснення.