Студент: Шкандюк Денис Олегович
Группа: М4140
Дата: 17 марта 2024 г.

Условия

Дан массив из 1000 чисел. На любом подотрезке этого массива определена функция $f(left, right)$, которая считается за $O(1)$. Предложите алгоритм разбиения массива на непересекающиеся отрезки (l_i, r_i), сумма значений $f(l_i, r_i)$ для которых максимальна

Решение:

1. Инициализация:
   - Инициализируем массив $dp[0...n-1]$, установив для всех элементов значение 0, где n - длина массива. Этот массив будет содержать максимальную сумму значений функций для отрезков, заканчивающихся на каждом индексе.
   - Инициализируем вспомогательный массив $prev[0...n-1]$, чтобы сохранить начальный индекс отрезка, который заканчивается на i, для последующего восстановления решения.
2. Заполняем таблицу DP:
   - Для каждого индекса i от 1 до $n-1$ (включительно) выполним итерацию по всем возможным начальным точкам отрезка, заканчивающегося на i, обозначаемого как j (где j колеблется от 0 до i).
   - Для каждой пары (j, i) вычисляем сумму текущего значения отрезка $f(j, i)$ и максимальную сумму, полученную для отрезков, заканчивающихся перед j, которая равна $dp[j-1]$ (для простоты рассмотрим $dp[-1] = 0$).
   - Обновим $dp[i]$ максимальным значением, полученным из всех возможных j для текущего i, т.е. нашей функцией оптимизации выгляжит так: $dp[i] = max(dp[i], dp[j-1] + f(j, i))$.
3. Обратная трассировка:
   - После заполнения массива dp максимальную сумму можно найти по адресу $dp[n-1]$.
   - Чтобы найти фактические отрезки, начинаем с последнего элемента массива и используя массив $prev$ для отслеживания начальных точек каждого отрезка, пока не будет достигнуто начало массива.
3. Верните результат:

   Конечным результатом является максимальная сумма значений f по непересекающимся сегментам, которая сохраняется в $dp[n-1]$, и отрезки могут быть восстановлены с помощью шага обратной трассировки.

Корректность

База: заключается в том, что для каждого элемента $dp[i]$ мы рассматриваем все возможные отрезки, заканчивающиеся в этой позиции, и выбираем максимально возможное значение. Так как $dp[i]$ включает в себя оптимальные значения для всех предыдущих сегментов, добавление нового сегмента на основе $f(l, r)$ будет также оптимальным.

Индуктивный шаг предполагает, что если для всех $k < i$ $dp[k]$ содержит максимальную сумму значений функции для отрезков, заканчивающихся на k, то при добавлении следующего отрезка $dp[i]$ также будет содержать максимальную сумму значений функции для сегментов, заканчивающихся на i. Это следует из того, что для каждого i мы исследуем все возможные "старты"отрезков и выбираем наилучший вариант.

Т.о., по принципу динамического программирования, когда каждый шаг оптимизации строится на предыдущем оптимальном решении, алгоритм гарантирует нахождение оптимального решения для всей задачи.

Анализ асимптотической сложности

Асимптотическая сложность алгоритма зависит от двух вложенных циклов, каждый из которых итерируется по n элементам массива, где ndрпа - длина массива. Внутренний цикл выполняет функцию f, которая, согласно условию задачи, имеет константную сложность O(1).

Общая асимптотическая сложность алгоритма составляет $O(n^2)$, так как для каждого из n элементов массива выполняется n операций, каждая из которых, включая вычисление f, занимает константное время.

Это доказательство подчеркивает, что алгоритм является эффективным с точки зрения достижения оптимального решения при условии, что функция f вычисляется за константное время. Если функция f имеет более высокую сложность, общая асимптотика алгоритма изменится в соответствии со сложностью f.