Forward Techniques

Introduction

In the class, we explored several techniques to optimize array operations, particularly focusing on computing cumulative sums efficiently using prefix sums and handling multiple queries with improved time complexity. We also explored the concept of subarrays and the carry forward technique.

Prefix Sum

Concept

- A **Prefix Sum** array is a powerful tool used to answer range sum queries efficiently. It allows you to calculate the sum of any subarray in constant time once the prefix sum array is constructed【4:16†transcript.txt】.

Creation of Prefix Sum Array

- **Definition:** The prefix sum array `pf[i]` is defined as the sum of elements from the start up to the ith index of the original array. Mathematically, `pf[i] = pf[i-1] + A[i]` .
- **Algorithm:**

```
pf[0] = A[0];
for(int i = 1; i < N; i++){
    pf[i] = pf[i-1] + A[i];
}
```

Example

Given the array `[2, 5, -1, 7, 1]` , the prefix sum array would be `[2, 7, 6, 13, 14]` .

Application in Query Solutions

Subarrays

Definition

- A **subarray** is a contiguous part of an array. For any given start index `i`, the number of valid subarrays is calculated as the length of the array (`N`) minus `i`.

Representation

- A subarray can be represented using its start and end indices.

Example

Given the array `[3, 10, 8, 10]`, all possible subarrays are `[3]`, `[3, 10]`, `[3, 10, 8]`, `[3, 10, 8, 10]`, `[10]`, `[10, 8]`, `[10, 8, 10]`, `[8]`, `[8, 10]`, and `[10]`.

Calculating Subarrays

To print all subarrays, use nested loops to iterate through all possible start and end indices .

Carry Forward Technique

Concept

- **Carry Forward** is a technique where results from previous steps are used to compute results for subsequent steps, reducing the need for repeated calculations .

Example: Counting Specific Pair Occurrences

- **Problem:** Counting the number of 'a,g' pairs where 'a' appears before 'g'.

of 'a's to the result.

- **Algorithm:**

```
int count_ag(string str) {
    int result = 0;
    int count_a = 0;
    for(char c : str) {
        if (c == 'a') {
            count_a++;
        } else if (c == 'g') {
            result += count_a;
        }
    }
    return result;
}
```

- **Time Complexity:** O(n), since it only requires a single pass through the array .

Summary

In this class, we discussed:

1. The efficient use of the prefix sum arrays for range queries.
2. The definition and calculation of subarrays.
3. The carry forward technique to reduce unnecessary calculations in iteratively processed data structures like arrays.

These methods are fundamental in optimizing array-related operations and are widely applicable in algorithm design 【4:0†transcript.txt】【4:6†transcript.txt】.