

Deque Applications

Ulises Mdz, Ulises Tirado Zatarain

Algorist Weekly Talks

ulisesmdzmtz@gmail.com

ulises.tirado@cimat.mx

July 12, 2016

Overview

- 1 Finding Maximum
 - Example
- 2 Deque Review
 - C++ STL Deque
- 3 Deque & BFS
 - BFS 0/1

Finding Maximum

Description

Given an array **nums**, there is a *sliding window* of size **k** which is moving from the very left of the array to the very right. You can only see the **k** numbers in the window. Each time the sliding window moves right by one position, can you calculate the maximum element each time the window moves?.^a

^aSource: <https://leetcode.com/problems/sliding-window-maximum/>

Example/Simulation

For example, Given **nums** = [1, 3, -1, -3, 5, 3, 6, 7], and **k** = 3.

Window position								Max
[1	3	-1]	-3	5	3	6	7	3
1	[3	-1	-3]	5	3	6	7	3
1	3	[-1	-3	5]	3	6	7	3
1	3	-1	[-3	5	3]	6	7	5
1	3	-1	-3	[5	3	6]	7	6
1	3	-1	-3	5	[3	6	7]	7

Therefore, return the max sliding window as [3, 3, 5, 5, 6, 7].

Could you solve it in linear time?

C++ STL Deque Review

```
std::deque
```

```
template < class T, class Alloc = allocator<T> > class deque;
```

Double ended queue

deque (usually pronounced like "deck") is an irregular acronym of double-ended queue. Double-ended queues are sequence containers with dynamic sizes that can be expanded or contracted on both ends (either its front or its back). Specific libraries may implement deques in different ways, generally as some form of dynamic array. But in any case, they allow for the individual elements to be accessed directly through random access iterators, with storage handled automatically by expanding and contracting the container as needed. Therefore, they provide a functionality similar to vectors, but with efficient insertion and deletion of elements also at the beginning of the sequence, and not only at its end.

Some things to notice

- What happens when we insert an element?
- What happens when we remove an element?
- What kind of data structure do we need?
- Do we need to store all elements from window?
- Useful elements

Solution using queue: C++

Example (C++ Implementation)

```
void printKMax(int arr[], int n, int k) {
    deque<int> Qi(k);
    for(int i=0; i<k; ++i) {
        while ( (!Qi.empty()) && arr[i] >= arr[Qi.back()])
            Qi.pop_back();
        Qi.push_back(i);
    }
    for(int i=k; i<n; ++i) {
        cout << arr[Qi.front()] << " ";
        while((!Qi.empty()) && Qi.front() <= (i-k))
            Qi.pop_front();
        while((!Qi.empty()) && arr[i] >= arr[Qi.back()])
            Qi.pop_back();
        Qi.push_back(i);
    }
    cout << arr[Qi.front()];
}
```


Ocean Currents (Difficult)

At each location, the current flows in some direction. The captain can choose to either go with the flow of the current, using no energy, or to move one square in any other direction, at the cost of one energy unit. The boat always moves in one of the following eight directions: north, south, east, west, north-east, North-west, south-east, south-west. The boat cannot leave the boundary of the lake.

You are to help him devise a strategy to reach the destination with the minimum energy consumption.

References

- <http://www.cplusplus.com/reference/queue/>
- <https://en.wikipedia.org/>
- <http://www.geeksforgeeks.org/maximum-of-all-subarrays-of-size-k/>
- <https://chococontest.wordpress.com/category/bfs-01/>
- <http://ideone.com/PvCked>