Paradigmas de Resolução de Problemas

Programação Dinâmica – *Max Range Sum*: Exercícios Resolvidos

---

Prof. Edson Alves - UnB/FGA

2020

Sumário

Timus 1146 – Maximum Sum

Problema

Given a 2-dimensional array of positive and negative integers, find the sub-rectangle with the largest sum. The sum of a rectangle is the sum of all the elements in that rectangle. In this problem the sub-rectangle with the largest sum is referred to as the maximal sub-rectangle. A sub-rectangle is any contiguous sub-array of size 1×1 or greater located within the whole array.

As an example, the maximal sub-rectangle of the array:

```
 0  -2  -7   0
 9   2  -6   2
-4   1  -4   1
-1   8   0  -2
```

is in the lower-left-hand corner and has the sum of 15.

Entrada e saída

Input

The input consists of an $N \times N$ array of integers. The input begins with a single positive integer N on a line by itself indicating the size of the square two dimensional array. This is followed by N^2 integers separated by white-space (newlines and spaces). These N^2 integers make up the array in row-major order (i.e., all numbers on the first row, left-to-right, then all numbers on the second row, left-to-right, etc.). N may be as large as 100. The numbers in the array will be in the range $[-127, 127]$.

Output

The output is the sum of the maximal sub-rectangle.

Exemplo de entradas e saídas

Sample Input

```
4
0 -2 -7 0
9 2 -6 2
-4 1 -4 1
-1 8 0 -2
```

Sample Output

```
15
```

Solução $O(N^3)$

- Uma solução de força bruta computaria a soma todas as N^4 submatrizes, sendo que cada soma é feita em $O(N^2)$, de modo que a solução teria complexidade $O(N^6)$
- Contudo, o uso de combinado de somas prefixadas e o algoritmo de Kadane permite identificar a submatriz de soma máxima com complexidade $O(N^3)$
- Para cada par de colunas (i, j), deve ser computado, por meio do algoritmo de Kadane nas somas $p_k(i, j)$, para $1 \leq k \leq N$, o intervalo de maior soma, onde

$$p_k(i, j) = \sum_{t=i}^{j} a_{kt}$$

- Veja que, dados os limites do problema, mesmo nos casos extremos a soma máxima ainda pode ser armazenada em variáveis inteiras

Solução $O(N^3)$

```cpp
#include <bits/stdc++.h>

using namespace std;

const int oo { 1000000010 };

int kadane(int N, const vector<int>& as)
{
    vector<int> s(N + 1);
    s[1] = as[1];

    for (size_t i = 2; i < as.size(); ++i)
        s[i] = max(as[i], s[i - 1] + as[i]);

    return *max_element(s.begin() + 1, s.end());
}

int solve(int N, const vector<vector<int>>& A)
{
    vector<vector<int>> p(N + 1, vector<int>(N + 1, 0));
    int ans = -oo;
```

Solução $O(N^3)$

```
22
23      for (int i = 1; i <= N; ++i)
24      {
25          vector<int> r(N + 1, 0);
26
27          for (int j = i; j <= N; ++j)
28          {
29              for (int k = 1; k <= N; ++k)
30                  r[k] += A[k][j];
31
32              ans = max(ans, kadane(N, r));
33          }
34      }
35
36      return ans;
37  }
38
39  int main()
40  {
41      ios::sync_with_stdio(false);
42
```

Solução $O(N^3)$

```
43      int N;
44      cin >> N;
45
46      vector<vector<int>> A(N + 1, vector<int>(N + 1));
47
48      for (int i = 1; i <= N; ++i)
49          for (int j = 1; j <= N; ++j)
50              cin >> A[i][j];
51
52      auto ans = solve(N, A);
53
54      cout << ans << endl;
55
56      return 0;
57  }
```

OJ 13095 – Tobby and Query

Problema

In his free time Tobby is always searching for interesting things. This time Tobby created the following problem: given a sequence of n integer numbers, Tobby would like to know how many different numbers are in the range $[l, r]$ $(r \geq l)$.

Entrada e saída

Input

The input has several test cases. The first line of each test case contains an integer n $(1 \leq n \leq 10^5)$, the size of the sequence of numbers. The next line contains n values a_i $(0 \leq a_i \leq 9)$, the numbers in the sequence. The next line contains an integer q $(1 \leq q \leq 10^4)$, the amount of queries. Then there are q lines, each line contains a query: two integers l and r $(1 \leq l, r \leq n)$.

Output

For each test case print q integers, representing the amount of different numbers in the range $[l, r]$ for each query in the input.

Exemplo de entradas e saídas

Sample Input

```
7
0 2 3 3 7 5 2
3
1 1
2 4
2 7
5
7 7 7 7 7
2
4 5
1 5
```

Sample Output

```
1
2
4
1
1
```

Solução $O(N+Q)$

- Uma forma de responder rapidamente (em $O(1)$) cada uma das consultas é calcular as somas dos prefixos p_d, onde d representa os 10 dígitos decimais (pois $0 \leq a_i \leq 9$)
- Estas somas podem ser computadas em $O(N)$
- Assim, a consulta para o intervalo $[L, R]$ pode ser respondida por meio da $RSQ(L, R)$ para cada um dos 10 vetores de prefixos:

$$q(L,R) = \sum_{d=0}^{9} \delta(p_d[R] - p_d[L-1]),$$

onde

$$\delta(x) = \begin{cases} 1, & \text{se } x > 0 \\ 0, & \text{caso contrário} \end{cases}$$

Solução $O(N + Q)$

```cpp
#include <bits/stdc++.h>

using namespace std;
using ii = pair<int, int>;

vector<int>
solve(int N, const vector<int>& xs, vector<ii>& qs)
{
    vector<vector<int>> ps(10, vector<int>(N + 1, 0));

    for (int i = 1; i <= N; ++i)
    {
        for (int d = 0; d <= 9; ++d)
            ps[d][i] += ps[d][i - 1];

        ps[xs[i]][i] += 1;
    }

    vector<int> ans;

```

Solução $O(N + Q)$

```
21      for (auto [L, R] : qs)
22      {
23          int res = 0;
24
25          for (int d = 0; d <= 9; ++d)
26              res += (ps[d][R] - ps[d][L - 1] > 0 ? 1 : 0);
27
28          ans.push_back(res);
29      }
30
31      return ans;
32  }
33
34  int main()
35  {
36      ios::sync_with_stdio(false);
37
38      int N;
39
40      while (cin >> N)
41      {
```

Solução $O(N + Q)$

```
42          vector<int> xs(N + 1);
43
44          for (int i = 1; i <= N; ++i)
45              cin >> xs[i];
46
47          int Q;
48          cin >> Q;
49
50          vector<ii> qs(Q);
51
52          for (int i = 0; i < Q; ++i)
53              cin >> qs[i].first >> qs[i].second;
54
55          auto ans = solve(N, xs, qs);
56
57          for (auto x : ans)
58              cout << x << '\n';
59      }
60
61      return 0;
62  }
```

Referências

1. Timus 1146 – Maximum Sum
2. OJ 13095 – Tobby and Query