HUST

ĐẠI HỌC BÁCH KHOA HÀ NỘI

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.

ĐẠI HỌC BÁCH KHOA HÀ NỘI

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

THUẬT TOÁN ỨNG DỤNG

CẤU TRÚC DỮ LIỆU VÀ KỸ THUẬT NÂNG CAO

Mảng cộng dồn, kỹ thuật 2 trỏ

ONE LOVE. ONE FUTURE.

NỘI DUNG

- Mảng cộng dồn
- Kỹ thuật 2 con trỏ

Mảng cộng dồn

- **Bài tập minh họa (P.02.02.01).** Cho dãy số a_1, a_2, ..., a_n. Thực hiện Q truy vấn, mỗi truy vấn được đặc trưng bởi cặp chỉ số (i, j) trong đó ta cần tính tổng $a_i + a_{i+1} + \ldots + a_j$.
- Thuật toán trực tiếp:
  - Với mỗi truy vấn, ta duyệt dãy từ a_i đến a_j để thực hiện tính tổng các phần tử của dãy con
  - Độ phức tạp trong tình huống tồi nhất của mỗi truy vấn là $O(n)$
  - Độ phức tạp trong tình huống tồi nhất của Q truy vấn là $O(Qn)$

```
sum(i, j){
    T = 0;
    for k = i to j do
      T = T + a_k;
    return T;
}
```

Mảng cộng dồn

- **Bài tập minh họa (P.02.02.01)**. Cho dãy số $a_1, a_2, \ldots, a_n$. Thực hiện Q truy vấn, mỗi truy vấn được đặc trưng bởi cặp chỉ số (i, j) trong đó ta cần tính tổng $a_i + a_{i+1} + \ldots + a_j$.
- Thuật toán sử dụng mảng cộng dồn:
  - Tính mảng cộng dồn $S_k = a_1 + a_2 + \ldots + a_k$ ($k = 1, 2, \ldots, n$), $S_0 = 0$
    - Độ phức tạp O(n)
  - Khi đó truy vấn (i, j) có giá trị bằng $S_j - S_{i-1}$
    - Độ phức tạp mỗi truy vấn là O(1)
    - Độ phức tạp của toàn bộ chương trình: O(n) + O(Q)

```
Input a₁, a₂, …, aₙ and Q;
S₀ = 0;
for k = 1 to n do
   Sₖ = Sₖ₋₁ + aₖ;

for q = 1 to Q do {
   Input i, j;
   res = Sⱼ - Sᵢ₋₁;
   output(res);
}
```


Mảng cộng dồn

- Cho mảng 2 chiều $a[1..n, 1..m]$, mảng cộng dồn $S[1..n, 1..m]$ được định nghĩa như sau:
  - $S[0, j] = 0$, $S[i, 0] = 0$, $i = 0, 1, \ldots, n$ và $j = 0, 1, \ldots, m$
  - $S[i, j] = \sum_{k=1}^{i} \sum_{q=1}^{j} a[k, q]$, $i = 1, \ldots, n$ và $j = 1, \ldots, m$
  - Công thức truy hồi: $S[i, j] = S[i-1, j] + S[i, j-1] - S[i-1, j-1] + a[i,j]$
- Thuật toán tính mảng cộng dồn
  - Độ phức tạp O(nm)

```
for i = 0 to n do S[i,0] = 0;
for j = 0 to m do S[0,j] = 0;
for i = 1 to n do {
   for j = 1 to n do {
       S[i,j] = S[i-1,j] + S[i,j-1] -
             S[i-1,j-1] + a[i,j];
   }
}
```


Mảng cộng dồn

- **Bài tập minh họa (P.02.02.02)**. Cho mảng 2 chiều $a[1..n, 1..m]$. Ta cần thực hiện Q truy vấn, mỗi truy vấn có dạng được đặc trưng bởi bộ chỉ số (a, b, c, d) và được định nghĩa như sau:

$$\text{query}[a, b, c, d] = \sum_{k=a}^{c} \sum_{q=b}^{d} a[k, q]$$

- Thuật toán trực tiếp:
  - Với mỗi truy vấn, ta thực hiện 2 vòng lặp lồng nhau để thực hiện duyệt qua tất cả các phần tử và tính tổng
  - Độ phức tạp của mỗi truy vấn trong tình huống tồi nhất O(nm)

Mảng cộng dồn

- **Bài tập minh họa (P.02.02.02)**. Cho mảng 2 chiều $a[1..n, 1..m]$. Ta cần thực hiện Q truy vấn, mỗi truy vấn có dạng được đặc trưng bởi bộ chỉ số (a, b, c, d) và được định nghĩa như sau:

$$\text{query}[a, b, c, d] = \sum_{k=a}^{c} \sum_{q=b}^{d} a[k,\ q]$$

- Thuật toán sử dụng mảng cộng dồn:
  - Công thức: query[a, b, c, d] = S[c, d] – S[c, b-1] – S[a-1, d] + S[a-1, b-1]
  - Độ phức tạp của mỗi truy vấn trong tình huống tồi nhất O(1)

Kỹ thuật 2 trỏ

- Trong nhiều tình huống, chúng ta phải thực hiện duyệt một dãy $a_1, a_2, \ldots, a_n$ để tìm kiếm các đối tượng được đặc trưng bởi 2 chỉ số (*i, j*) trên dãy (ví dụ: dãy con gồm các phần tử liên tiếp đứng cạnh nhau hoặc cặp 2 phần tử của dãy) có tính chất nào đó.
  - Dùng 2 vòng lặp lồng nhau để duyệt qua tất cả các cặp 2 chỉ số (i, j): độ phức tạp O(n^2)
  - Dùng 2 con trỏ tiến theo 1 chiều hoặc tiến theo 2 chiều ngược nhau: độ phức tạp O(n)

Kỹ thuật 2 trỏ

- **Bài tập minh họa 2.1 (P.02.02.03).** Cho dãy số a[1], a[2], . . ., a[n] được sắp xếp theo thứ tự tăng dần (các phần tử đôi một khác nhau). Cho trước giá trị Q, hãy đếm số cặp 2 chỉ số i và j sao cho a[i] + a[j] = Q.

- Thuật toán trực tiếp
  - Dùng 2 vòng lặp lồng nhau để duyệt qua tất cả các cặp (i, j) và kiểm tra điều kiện a[i] + a[j] = Q
  - Độ phức tạp O(n^2)

```
res = 0;
for i = 1 to n do {
  for j = i+1 to n do {
     if a[i] + a[j] = Q then
        res = res + 1;
  }
}
Output(res);
```


Kỹ thuật 2 trỏ

- **Bài tập minh họa 2.1 (P.02.02.03).** Cho dãy số a[1], a[2], . . ., a[n] được sắp xếp theo thứ tự tăng dần (các phần tử đôi một khác nhau). Cho trước giá trị Q, hãy đếm số cặp 2 chỉ số i và j sao cho a[i] + a[j] = Q.

- Thuật toán sử dụng 2 trỏ
  - Biến i di chuyển từ trái qua phải; biến j chạy từ phải qua trái trên dãy
  - Độ phức tạp O(n)

```
res = 0;
i = 1; j = n;
while i < j do {
  if a[i] + a[j] = Q then {
     res = res + 1; i = i + 1; j = j - 1;
  }else if a[i] + a[j] < Q then
     i = i + 1;
  else
     j = j - 1;
}
Output(res);
```


Kỹ thuật 2 trỏ

- **Bài tập minh họa 2.2 (P.02.02.04).** Cho dãy số không âm a[1], a[2], . . ., a[n]. Cho trước giá trị Q, hãy tìm dãy con (gồm một số phần tử đứng liên tiếp cạnh nhau) dài nhất mà có tổng nhỏ hơn hoặc bằng Q.

- Thuật toán trực tiếp
  - Dùng 2 vòng lặp lồng nhau để xét tất cả các vị trí bắt đầu và kết thúc của 1 dãy con và kiểm tra điều kiện tổng có nhỏ hơn hoặc bằng Q hay không?
  - Độ phức tạp O(n^2)

```
res = 0;
for i = 1 to n do {
  S = 0;
  for j = i to n do {
    S = S + a[j];
    if S <= Q then {
       res = max(res, j - i + 1);
    }
  }
}
Output(res);
```

Kỹ thuật 2 trỏ

- **Bài tập minh họa 2.2 (P.02.02.04)**. Cho dãy số không âm a[1], a[2], . . ., a[n]. Cho trước giá trị Q, hãy tìm dãy con (gồm một số phần tử đứng liên tiếp cạnh nhau) dài nhất mà có tổng nhỏ hơn hoặc bằng Q.

- Thuật toán sử dụng 2 trỏ
  - Biến L di chuyển từ trái qua phải; biến R chạy từ trái qua phải trên dãy
  - Độ phức tạp O(*n*)

```
res = 0; S = 0;
L = 1;
for R = 1 to n do {
   S = S + a[R];
   while S > Q do {
      S = S - a[L]; L = L + 1;
   }
   res = max(res, R - L + 1);
}
Output(res);
```


HUST

THANK YOU !