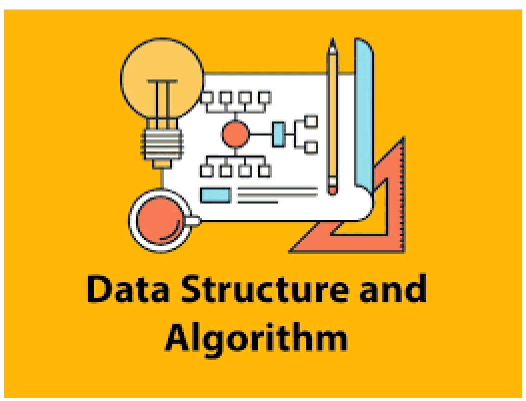
ĐỘI TUYỂN OLP Tin học PTIT 2025

CHỮA BÀI TUẦN 1 (Sep 29 – Oct 5)



Contest cá nhân – Thứ 7

- Bài 1: Mảng prefix/số học
- Bài 2: CTDL
- Bài 3: DSU
- Bài 4: thuật toán Dijkstra

Bài 1: Tính giá trị biểu thức

- F(x) = 1 * (1! + x) + 2 * (2! + x) + ... + x * (x! + x)
  = 1*1! + 2*2! + 3*3! + … + x*x! + x*(1+2+…+x)
  = (1*1! + 2*2! + 3*3! + … + x*x!) + x*x*(x+1)/2
- Có thể sử dụng công thức này hoặc sử dụng tổng tiền tố

$$k \cdot k! = (k+1)! - k!$$

$$\sum_{k=1}^{x} k \cdot k! = \sum_{k=1}^{x} [(k+1)! - k!] \Rightarrow \sum_{k=1}^{x} k \cdot k! = (x+1)! - 1!$$

- Nếu x+1 >= M thì (x+1)! chia hết cho M
- Tính nhanh F(a[i]) trong O(1).

Bài 2: Tráo đổi phần tử dãy số

- Bài toán kiểu quản lý dãy số động, xử lý bằng xây dựng dãy ảo
- Gán mỗi phần tử A[i] ở vị trí BASE+i (chọn BASE lớn hơn M)
- → Dãy A[1], A[2], …, A[N] đặt ở vị trí BASE+1, BASE+2, …, BASE+N
- → Với L = BASE+1, R = BASE+N để quản lý đoạn
- Đánh dấu trên cây IT/BIT là tree[pos[A[i]]] = 1

Các thao tác:

- Update(pos, val): val = 1 → thêm phần tử vào vị trí pos, val = 0 → xóa

Bài 2: Tráo đổi phần tử dãy số

- Dãy A[1], A[2], …, A[N] đặt ở vị trí BASE+1, BASE+2, …, BASE+N
- Với L = BASE+1, R = BASE+N để quản lý đoạn
- Đánh dấu trên cây IT/BIT là tree[pos[A[i]] = 1

Các thao tác:

- Với số x ở vị trí pos[x], tổng các phần tử là N
  - Khoảng cách bên trái: Dleft = getsum(1, pos[x]-1)
  - Khoảng cách bên phải: Dright = N – getsum(1, pos[x])
  - Chi phí bằng min(Dleft, Dright)
- Di chuyển phần tử:
  - Xóa phần tử cũ: update(pos[x], 0)
  - Truy vấn bên trái: đặt pos[x] = --L; update(pos[x], 1)
  - Truy vấn bên phải: đặt pos[x] = ++R; update(pos[x], 1)

Bài 3: Chọn quân bài

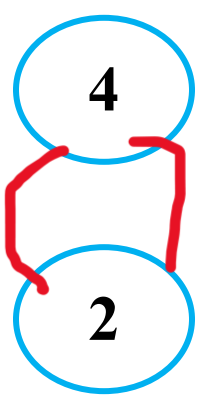
Tóm tắt

- Đưa về bài toán đồ thị, với lá bài i, nối cạnh (P[i], Q[i]).
- Bài toán quy về: có bao nhiêu subset trên tập cạnh E sao cho nó là một phủ cạnh (mỗi đỉnh được che phủ bởi ít nhất 1 cạnh)

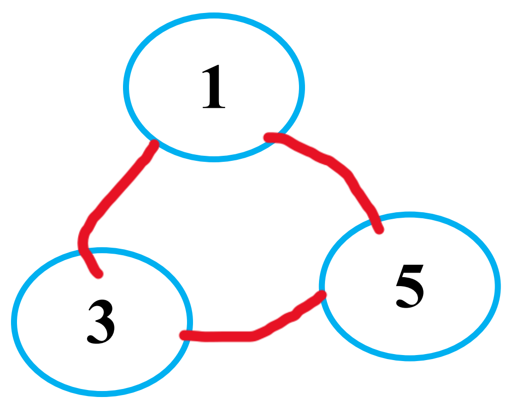
Bài 3: Chọn quân bài

Lời giải

- Đưa về bài toán đồ thị, với lá bài i, nối cạnh (P[i], Q[i]).
- Bài toán quy về: có bao nhiêu subset trên tập cạnh E sao cho nó là một phủ cạnh (mỗi đỉnh được che phủ bởi ít nhất 1 cạnh)
- Vì P[], Q[] đều là hoán vị của(1,…,N), nên bậc của mỗi đỉnh là 2, đồ thị là một tập hợp các chu trình. Do đó, chỉ cần giải bài toán cho một chu trình là đủ. Answer = tích đáp án của từng chu trình



3 cách



4 cách

Test 2 có 12 cách

Bài 3: Chọn quân bài

Lời giải

- Gọi F(M) là số cách chọn subset trong tập {1, 2, …, M} sao cho trong 2 số liên tiếp, có ít nhất 1 số được chọn.
- Tính F(M)
  - Nếu chọn số M, M-1 số còn lại là tùy ý, hay chính là F(M-1).
  - Nếu không chọn M, buộc phải chọn M-1, do vậy M-2 số còn lại, ta chọn tùy ý, hay chính là F(M-2).
  - Vậy F(M) = F(M-1) + F(M-2) với F(1) = 2 và F(2) = 3
  - Đây chính là dãy Fibonacci, F(M) = fibo(M+2)

Bài 3: Chọn quân bài

Lời giải

- Gọi G(M) là số cách chọn subset cạnh trong tập cạnh {1 vs 2, 2 vs 3, …, M-1 vs M, M vs 1} sao cho mỗi đỉnh được che phủ bởi ít nhất 1 cạnh.
- Tính G(M)
  - Nếu chọn cạnh (1, M), M-1 cạnh còn lại giống như bài toán phía trước, chọn tập cạnh sao cho trong 2 cạnh liên tiếp phải chọn ít nhất 1 cạnh, đáp án là F(M-1).
  - Nếu không chọn cạnh (1, M), ta buộc phải chọn cạnh (1, 2) và (M-1 vs M) để che phủ đỉnh 1 và đỉnh M. Tập cạnh còn lại có độ dài M-3, và đáp án dựa trên bài toán phía trước là F(M-3).
  - Ta có G(M) = F(M-1) + F(M-3) = fibo(M+1) + fibo(M-1)
  - Đây chính là dãy số Lucus:

  L(1) = 1, L(2) = 3, L(3) = 4, L(M) = L(M-1) + L(M-2)

Bài 4: Kim cương

Tóm tắt đề

- Cho đồ thị n đỉnh và m cạnh **một chiều** có trọng số**.** Ban đầu có S đồng và D viên kim cương. Mỗi đỉnh chứa giá trị P[i]
- Để đi qua cạnh j cần trả c[j] đồng. Tại đỉnh i có thể đổi một số viên kim cương thành tiền với giá P[i] đồng mỗi viên.
- Cần tìm cách để đi từ đỉnh 1 đến đỉnh N và giữ được nhiều kim cương nhất

Bài 4: Kim cương

Giải

- Sử dụng Dijkstra với 3 giá trị: Số kim cương, số đồng tiền và đỉnh i có P[i] lớn nhất đã đi qua. Dijkstra sẽ ưu tiên các trạng thái với số kim cương lớn hơn, nếu không thì ưu tiên trạng thái có tổng giá trị nếu quy đổi hết kim cương thành tiền lớn hơn.
- Tạo một hàm F[i][j] lưu lại tổng giá trị tiền lớn nhất khi đi đến đỉnh i và đỉnh có giá trị P lớn nhất là đỉnh j. Nếu bất kỳ đường đi nào đến đỉnh này sau thì sẽ không tối ưu.
- Độ phức tạp O(n* n + m * log(m + n * n))

2025 ICPC Asia Taiwan Online Programming Contest

J: Gas station - Dương

K: Move stone – thầy Kiên

106084J – Gas Station

Tóm tắt đề bài:

Cho một cây **n** đỉnh có trọng số. Cần đặt **k** trạm lên các đỉnh của cây sao cho với hai trạm xăng đặt tại **u** và **v** sao cho không có trạm nào khác nằm trên đường đi (**u**, **v**), tổng trọng số từ **u** tới **v** không vượt quá **d**.

Tìm **d** nhỏ nhất để có thể đặt k trạm thỏa mãn điều kiện trên.

106084J – Gas Station

Nhận xét: d càng nhỏ -> số trạm có thể đặt để thỏa mãn điều kiện của đề bài càng lớn

⇒ Chặt nhị phân d

Giả sử d = m, ta cần tính cnt là số trạm lớn nhất có thể đặt. Nếu cnt <= k thì điều kiện của đề bài được thỏa mãn

106084J – Gas Station

Với mỗi đỉnh u, gọi dfs(u) là khoảng cách lớn nhất từ u tới một đỉnh v nào đó thuộc cây con gốc u mà đường đi từ u tới v không có trạm nào được đặt.

106084J – Gas Station

Với mỗi đỉnh u, gọi dfs(u) là khoảng cách lớn nhất từ u tới một đỉnh v nào đó thuộc cây con gốc u mà đường đi từ u tới v không có trạm nào được đặt.

Xét một đỉnh u nào đó trên cây, có các con trực tiếp là (v1, w1), (v2, w2), …, (vl, wl).

Đặt mx1 = max_i(dfs(vi) + wi),

mx2 = max_(j != i)(dfs(vj) + wj)

Nếu mx1 + mx2 > m, ta cần đặt một trạm tại u

Gọi (p, w) là cha trực tiếp của u. Nếu w + mx1 > m ta cũng phải đặt một trạm tại u.

106084J – Gas Station

Nếu ta phải đặt một trạm tại u, tăng cnt lên 1 và đồng thời trả về dfs(u) = 0

Nếu không đặt trạm nào tại u thì dfs(u) = mx1

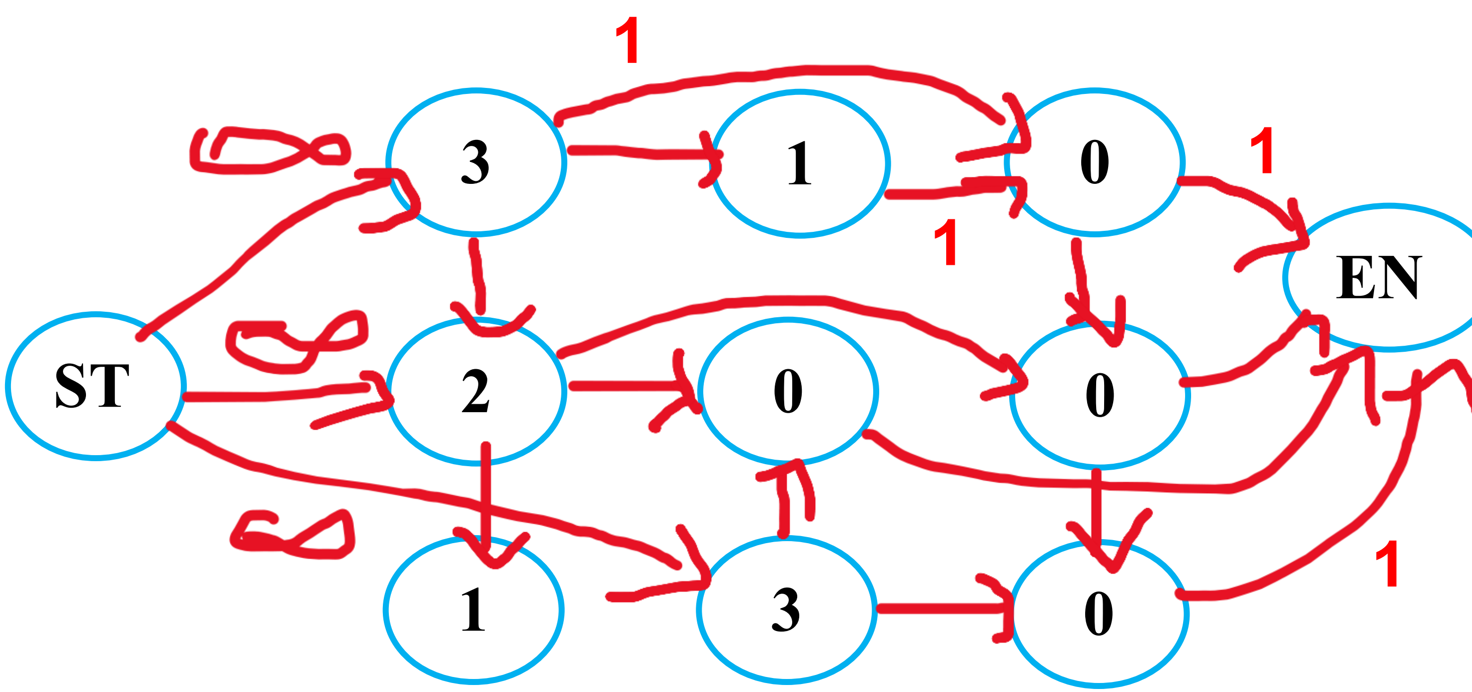
Độ phức tạp: O(log(tổng trọng số) * n).

106084K - Move stone

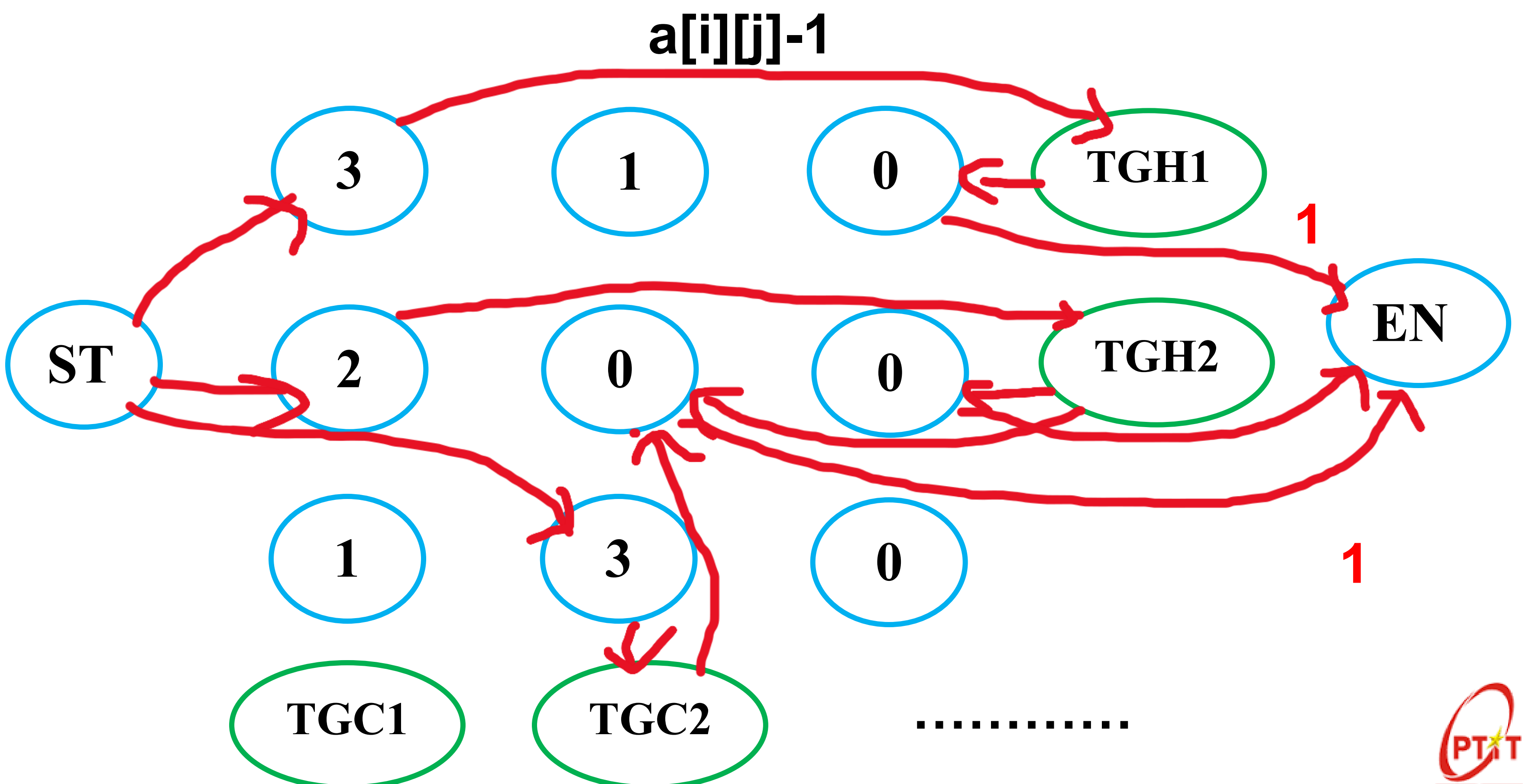
Đề bài

- Cho bảng n x n, mỗi ô có một số viên sỏi, tổng số viên sỏi đúng bằng n^2.
- Mỗi bước, có thể di chuyển 1 viên sỏi sang 1 ô cùng hàng hoặc cột.
- Cần di chuyển ít bước nhất sao cho thu được bảng với mỗi ô có đúng 1 viên sỏi.

106084K - Move stone



106084K - Move stone



106084K - Move stone

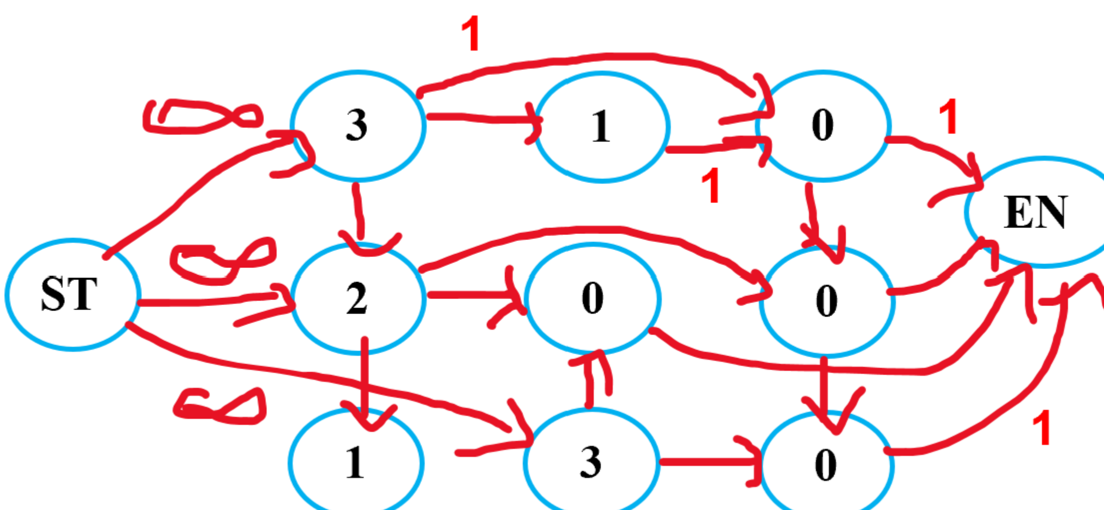
Lời giải

- Sử dụng thuật toán luồng cực đại.
- Các ô có nhiều hơn 1 viên, san sẻ, di chuyển bớt sỏi sang các ô khác, đóng vai trò là các đỉnh phát.
- Các ô không có viên nào, đóng vai trò là các đỉnh thu.
- Ta xây dựng 1 đỉnh phát tổng ST và 1 đỉnh thu tổng EN.

106084K - Move stone

Lời giải

- Đỉnh phát nối tới ô a[i][j] > 1, thông lượng INF.
- Với mỗi a[i][j] > 1, nối các cạnh sang tất cả ô cùng hàng, cùng cột, thông lượng bằng 1
- Các ô có a[i][j] = 0, nối cạnh tới đỉnh thu EN, thông lượng = 1
- Đáp số bằng 2 * số ô (a[i][j] == 0) - maxFlow



106084K - Move stone

Lời giải

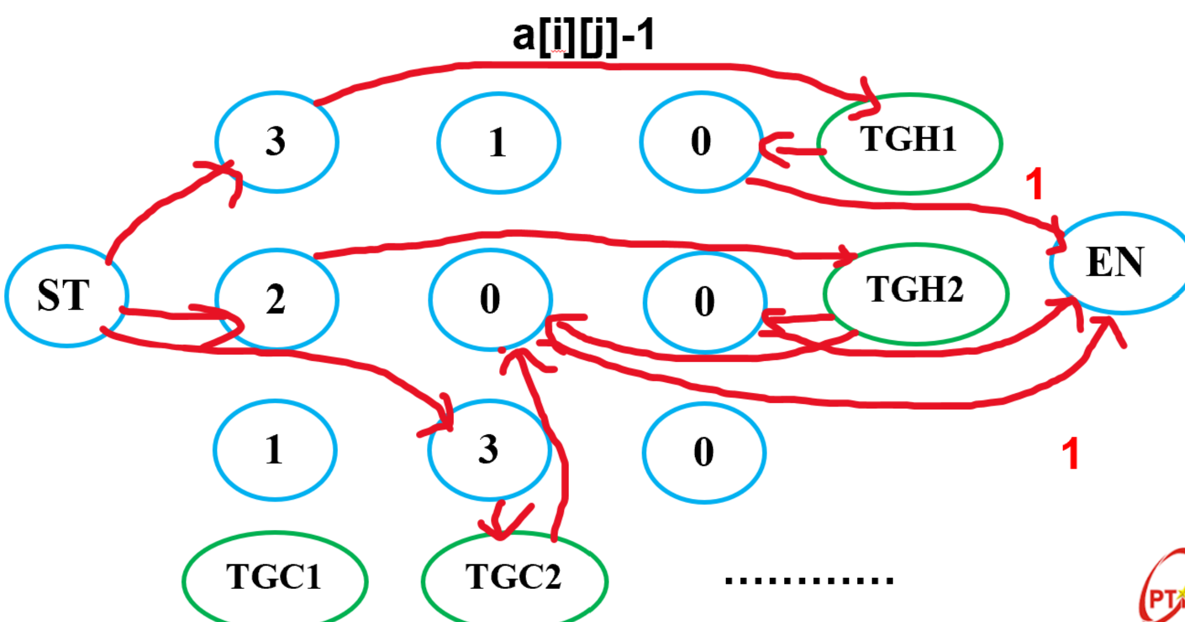
- Đỉnh phát nối tới ô a[i][j] > 1, thông lượng INF.
- Với mỗi a[i][j] > 1, nối các cạnh sang tất cả ô cùng hàng, cùng cột, thông lượng bằng 1
- Các ô có a[i][j] = 0, nối cạnh tới đỉnh thu EN, thông lượng = 1
- Đáp số bằng 2 * số ô (a[i][j] == 0) – maxflow

→ Cách làm này đúng nhưng bị TLE do số cạnh của đồ thị là N^3

106084K - Move stone

Lời giải

- Tạo đỉnh trung gian TGH thể hiện viên sỏi có thể đi từ a[i][j] > 1, thông lượng là a[i][j]-1 tới bất cứ vị trí nào, thay vì cố định qua vị trí a[i][k] với mọi k khác j.
- Tương tự, tạo đỉnh TGC (trung gian cột) → số cạnh giảm còn N^2



The 2024 ICPC Asia Yokohama Regional Contest

D: Gas station – Tiến Khôi

L: - Hoàng Minh Đức

F: - thầy Kiên

105633D – Tree Generators

1

Đề bài: Cho hai biểu thức, mỗi biểu thức định nghĩa một quy trình ngẫu nhiên để sinh ra một cây n đỉnh. n <= 7e5

2 **Yêu cầu:**

- Đếm số lượng cây khác nhau có thể được sinh ra từ cả hai biểu thức đã cho. Kết quả lấy modulo 998 244 353.

3 **Cú Pháp Biểu Thức (E):**

Một biểu thức E có thể có một trong hai dạng:

- '1': Biểu thức này sinh ra một cây duy nhất chỉ có một đỉnh, được gán nhãn là 1.
- '(E1 E2)': Biểu thức này kết hợp hai cây được sinh ra từ hai biểu thức con là E1 và E2.

Quy Tắc Sinh Cây từ (E1 E2):

Giả sử:

- E1 sinh ra cây T1 có n1 đỉnh (được gán nhãn từ 1 đến n1).
- E2 sinh ra cây T2 có n2 đỉnh (được gán nhãn từ 1 đến n2).

Quá trình tạo cây mới diễn ra như sau:

1. Đánh lại nhãn: Cộng n1 vào nhãn của tất cả các đỉnh trong cây T2. Các đỉnh của T2 giờ sẽ có nhãn từ n1 + 1 đến n1 + n2.
2. Thêm cạnh ngẫu nhiên: Chọn ngẫu nhiên một đỉnh từ T1 và một đỉnh từ T2 (đã được đánh lại nhãn), rồi thêm một cạnh nối chúng lại.
3. Kết quả: Cây mới được tạo thành là một cây duy nhất có n1 + n2 đỉnh.

105633D – Tree Generators

1 **Nhận xét:**

- Ứng với mỗi một cặp ngoặc trong biểu thức sẽ tương ứng với việc ta merge 2 cây E1, E2 lại thành 1 cây mới
- Hay nói cách khác với mỗi một cặp ngoặc sẽ tương ứng với việc nối 1 cạnh giữa cây gồm các đỉnh từ [L, Mid] đến cây gồm các đỉnh [Mid + 1, R]

105633D – Tree Generators

2 **Giải thuật:**

- Ứng với mỗi 1 biểu thức ta sẽ dựng 1 cây dựa vào nhận xét trên như sau, khi kết thúc một cặp ngoặc ta sẽ tạo 1 cạnh nối giữa đỉnh [L, R] đến 2 đỉnh con là [L, Mid] và [Mid + 1, R]
- Khi đó bài toàn trở thành bài toán QHĐ trên cây, khi duyệt đến đỉnh [L, R] trên cây 1 thì :
  - Gọi [U, V] là đoạn trên cây thứ 2 sao cho [U, V] có 2 nút con là [U, Mid] và [Mid + 1, V]
  - Khi đó: F[L, R] = F[L, Mid] * F[Mid + 1][R] * Min(Mid - U + 1, Mid - L + 1) * Min(V - Mid, R - Mid)
- Mỗi một nút [L, R] trên cây 2 sẽ có 2 nút con là [L, Mid] và [Mid + 1, R] nên khi đó mình có thể dùng map để lưu lại Range[Mid] = {L, R} để khi cần tìm [U, V] là đoạn trên cây thứ 2 sao cho [U, V] có 2 nút con là [U, Mid] và [Mid + 1, V] thì chỉ cần gọi ra Range[Mid]
- ĐPT: O(N * Log2(N))

105633L: Peculiar Protocol

Tóm tắt đề

- Cho dãy số không âm A độ dài N <= 500 và 2 số D, R
- Được thực hiện thao tác sau không giới hạn số lần:
  - Chọn đoạn con liên tiếp sao cho A[l] + A[l+1] … + A[r] = k * D + R với k là một số nguyên không âm. Sau thao tác này nhận được số điểm là k
  - Xoá đoạn A[l], A[l+1], …, A[r] khỏi dãy số.
  - Dãy trở thành dãy A[1], A[2], …A[l – 1], A[r + 1], …, A[n] và được đánh số lại
- Yêu cầu: Tìm tổng điểm lớn nhất có thể đạt được

105633L: Peculiar Protocol

Giải:

- Đọc đề
- Gọi S[i] là tổng cộng dồn
- Nếu xoá được một dãy (u, v) dùng T thao tác thì nhận được số điểm là (S[v] – S[u – 1] – T * R) / d (điều kiện tử chia hết cho mẫu)
- Gọi f[u][v] là tổng điểm lớn nhất đạt được nếu chỉ áp dụng thao tác bên trong đoạn (u, v)
- Gọi g[u][v] là số thao tác dùng được nhiều nhất nếu chỉ áp dụng thao tác bên trong đoạn (u, v)

105633L: Peculiar Protocol

Giải:

- TH 1: Kết quả đoạn (u, v) là hợp của 2 đoạn
  - f[u][v] = max(f[u][m] + f[m + 1][v]) với m = u, u + 1, …, v - 1
  - g[u][v] = max(g[u][m] + g[m + 1][v]) với m = u, u + 1, …, v - 1
- TH 2: Áp dụng thao tác lên đoạn (u, v)
  - Nếu có cách sử dụng k thao tác lên một đoạn thì có cách sử dụng k – 1 thao tác lên đoạn đó.
  - f[u][v] = max(f[u][v], (S[v] – S[u – 1] – (k + 1) * r) / d) với k nhỏ nhất thoả mãn k <= g[u][v] trong TH 1 và tử chia hết cho mẫu.
  - g[u][v] = max(g[u][v], k + 1) với k nhỏ nhất thoả mãn k <= g[u][v] trong TH1 và tử chia hết cho mẫu.
- Kết quả bài toán f[1][n].
- Độ phức tạp: O(n^3)

105633F: Điểm xa nhất

Tóm tắt:

- Cho hình hộp chữ nhật kích thước a x b x c
- Hãy tìm điểm xa gốc tọa độ O nhất

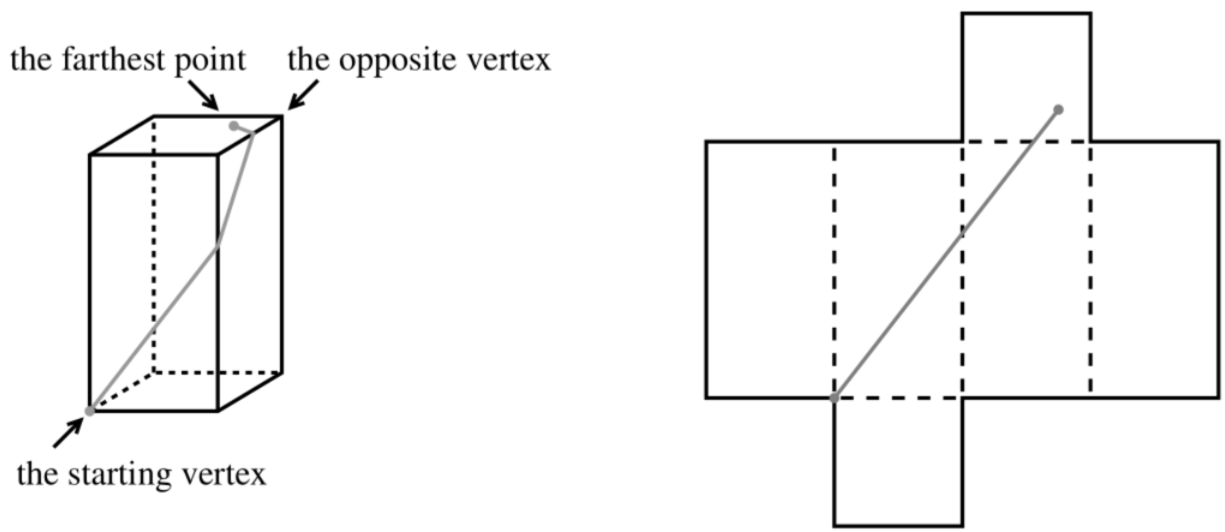
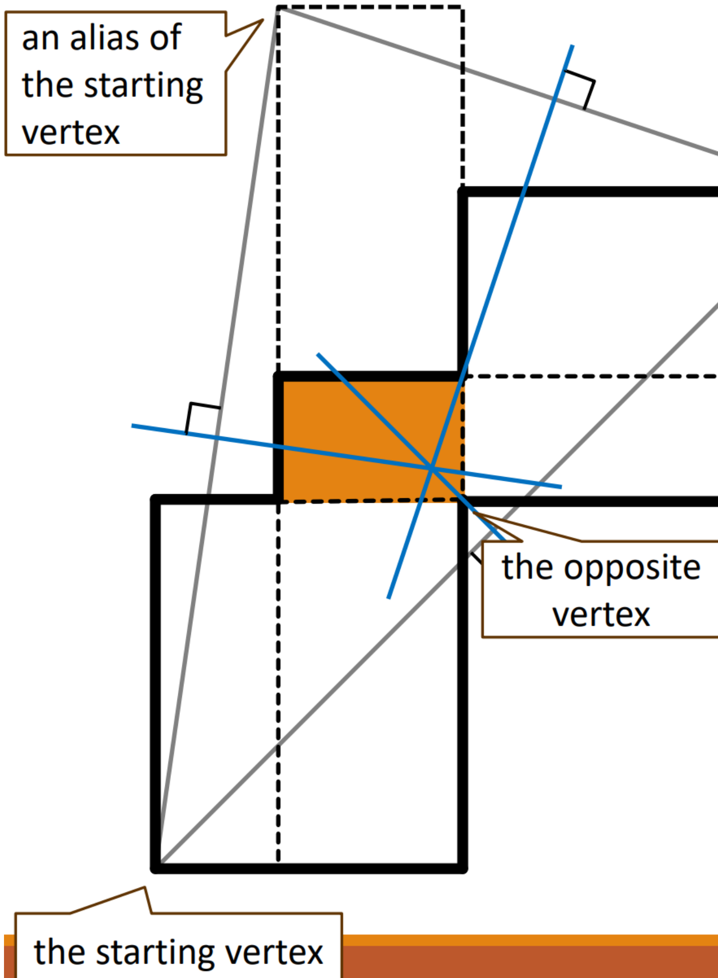


Figure F-1: Rectangular cuboid of size $1 \times 1 \times 2$, and its net

105633F: Điểm xa nhất

Core Idea



Any path from the farthest point to the starting vertex has the same distance.

The farthest point is the circumcenter of the triangle consists of the starting vertex and its two aliases on a net.

Calculate the length of the longest path to the candidate among sufficient net settings.

105633F: Điểm xa nhất

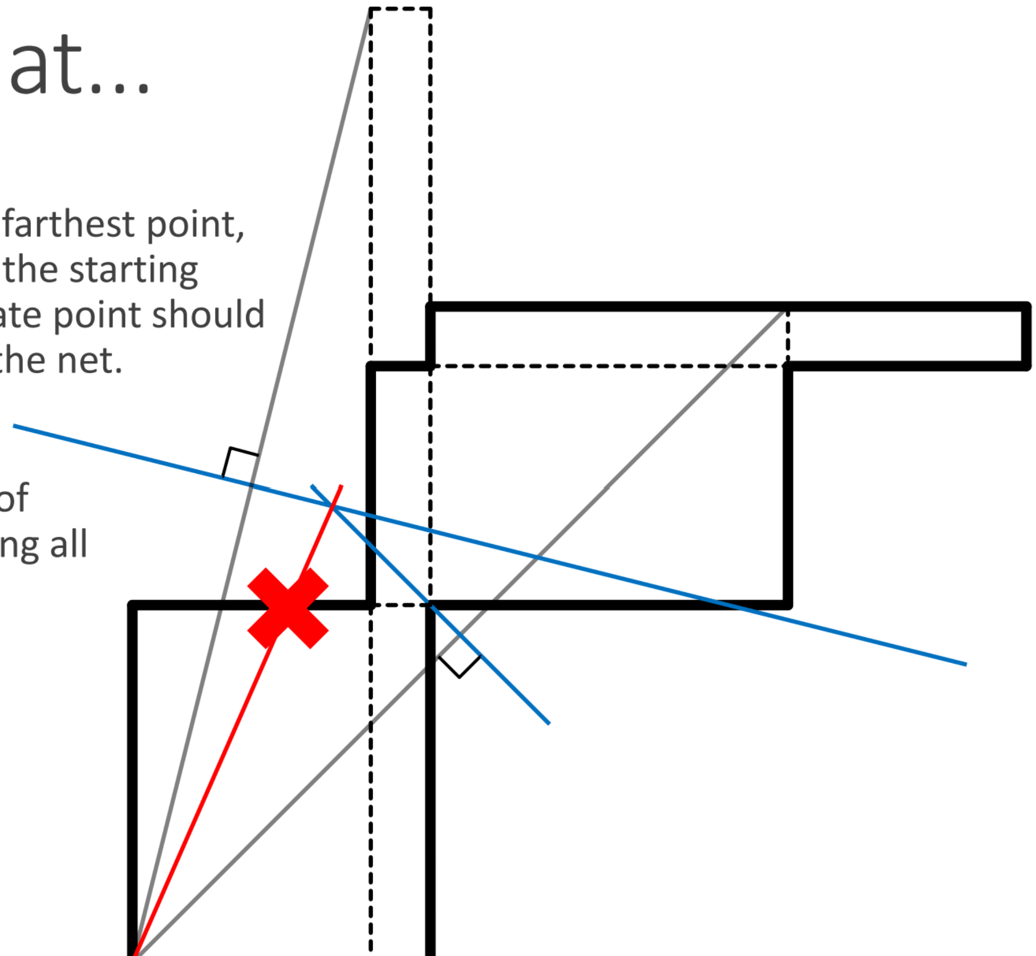
Ngoại lệ

Note that...

For a candidate of the farthest point, the segment between the starting vertex and the candidate point should not cross any edge of the net.

Distance is the length of the shortest path among all possible paths.

The distance from the starting point is not a convex function.



105633F: Điểm xa nhất

Lời giải

- Cho hình hộp chữ nhật kích thước a x b x c
- Sắp xếp a <= b <= c
- Điểm đối diện gốc tọa độ O là

d_opp = sqrt((a+b)^2 + c^2)

- Đường tròn ngoại tiếp tam giác tạo bởi O và 2 điểm copy của O khi bung hình hộp ra:

R = sqrt((a+b-x)^2 + (c+x)^2)

- ans = max(d_opp, R)

105633F: Điểm xa nhất

Lời giải

- Cho hình hộp chữ nhật kích thước a x b x c
- Sắp xếp a <= b <= c
- Điểm đối diện gốc tọa độ O là

d_opp = sqrt((a+b)^2 + c^2)

- Đường tròn ngoại tiếp tam giác tạo bởi O và 2 điểm copy của O khi bung hình hộp ra:

R = sqrt((a+b-x)^2 + (c+x)^2)

- ans = max(d_opp, R)

- **Chứng minh x = a/2 * (1 – b/c) ???**