SUMMER CAMP 2022

Trường Chuyên Phan Bội Châu – Nghệ An

presented by Đỗ Phan Thuận
dophanthuan@gmail.com

Khoa Học Máy Tính
Đại học Bách Khoa Hà Nội

Ngày 17 tháng 5 năm 2022

Bài 2. TOWER

Bài 3. TELMOV

Bài 2. TOWER

Bài 3. TELMOV

TOWER

- n loại hình hộp chữ nhật có kích thước (x_i, y_i, z_i) có số lượng tùy ý và có thể xoay hoặc lật trong không gian 3 chiều.
  - i.e. $(x_i, y_i, z_i) \rightarrow (y_i, z_i, x_i)$
- Một tòa tháp $t^{(1)}, t^{(2)}, \ldots$ được xây mỗi tầng là một hình hộp sao cho mặt sàn tầng dưới lớn hơn chặt mặt sàn tầng trên:
  - $t_x^{(i)} > t_x^{(i+1)}, t_y^{(i)} > t_y^{(i+1)}$
- Mục tiêu: Xây tòa tháp cao nhất có thể.

$$\sum_{i=1} t_z^{(i)} \rightarrow max$$

Nhận xét

- $t_x^{(i)} > t_x^{(i+1)}, t_y^{(i)} > t_y^{(i+1)} \rightarrow$ bỏ khả năng xoay và lật của hình hộp thì mỗi hình hộp chỉ được sử dụng một lần.
- có tối đa 6 cách xoay cho mỗi hình hộp
- $\rightarrow$ sinh ra $6 * n$ hình hộp, mỗi hình hộp dùng một lần.
- sắp xếp lại các hình hộp theo độ lớn giảm dần của $x_i \rightarrow y_i \rightarrow z_i$.
- $\rightarrow$ với mỗi hình hộp, đảm bảo hình hộp đứng sau không lớn hơn hình hộp đứng trước.
- $\rightarrow$ chia bài toán thành $6 * n$ bài toán nhỏ, bài toán i ứng với xây tòa tháp độ cao lớn nhất sử dụng các hình hộp từ $1...i \rightarrow$ quy hoạch động.

Thuật toán

- $dp[i]$ chiều cao tòa tháp cao nhất sử dụng hình hộp i làm chóp.
- $$dp[i] = \max_{j \in [0..i-1], x[i] < x[j], y[i] < y[j]} (dp[j] + z[i])$$
- kết quả $\max_{i \in [1, 6*n]}(dp[i])$

Bài 2. TOWER

Bài 3. TELMOV

TELMOV

Cô kỹ sư Alice đang sống ở trong thiên hà VNOI2020. Trong thiên hà này có N hành tinh khác nhau và M kênh vận chuyển hai chiều dạng (x, y, t) cho phép bạn di chuyển từ hành tinh x đến hành tinh y (hoặc ngược lại) trong t giây.

Nhưng Alice nhận thấy phương pháp vận chuyển này rất kém hiệu quả nên đã phát triển một thiết bị cho phép bạn dịch chuyển từ hành tinh x đến bất kỳ hành tinh y nào khác trong P giây với điều kiện bạn có thể đến hành tinh y đó từ hành tinh x chỉ sử dụng tối đa L kênh vận chuyển.

Thiết bị này hiện mới là bản thử nghiệm nên không thể được sử dụng quá K lần. Alice đang ở hành tinh 1 và muốn biết thời gian tối thiểu để đến hành tinh N.

Yêu cầu: Viết chương trình tính thời gian tối thiểu cần thiết để đến được hành tinh N bắt đầu từ hành tinh 1.

Thuật toán 24 điểm

Đúng với các test thoả mãn điều kiện không được sử dụng phép dịch chuyển và tất cả các kênh vận chuyển đều có thời gian $= 1$. Do đó ta chỉ cần sử dụng thuật toán loang BFS đơn giản để giải quyết.

Thuật toán 50 điểm

Đúng với các test thoả mãn điều kiện không sử dụng phép dịch chuyển. Do đó bài toán chính là bài toán tìm đường đi ngắn nhất trên đồ thị có trọng số và ta chỉ cần sử dụng thuật toán Dijkstra đơn giản để giải quyết.

Thuật toán 50 điểm

Đúng với các test thoả mãn điều kiện không sử dụng phép dịch chuyển. Do đó bài toán chính là bài toán tìm đường đi ngắn nhất trên đồ thị có trọng số và ta chỉ cần sử dụng thuật toán Dijkstra kết hợp với Hàng đợi ưu tiên PQ để giải quyết.

Thuật toán 66 điểm

Đúng với các test thoả mãn điều kiện số lượng hành tinh nhỏ hơn hoặc bằng 300. Do đó có thể chia bài toán thành hai bước

1. Xây dựng lại đồ thị bằng cách bổ sung thêm các cạnh đi được bởi cách dịch chuyển thoả mãn các ràng buộc về số lần sử dụng kênh vận chuyển.
2. Áp dụng thuật toán tìm đường đi ngắn nhất trên đồ thị mới xây dựng.

Thuật toán 100 điểm

- Quan sát thấy các điều kiện hạn chế $L, K \leq 10$ là rất nhỏ. Vì vậy ta có thể lưu lại được các trạng thái dịch chuyển trong quá trình xây dựng đường đi ngắn nhất bởi thuật toán Dijkstra vào mảng ba chiều $10000 \times 10 \times 10$.
- Sửa hàm đánh giá tối ưu của thuật toán Dijkstra thành $F(x, y, z)$, nghĩa là thời gian đến x nhỏ nhất sử dụng y lần dịch chuyển và qua z cạnh tính từ lần dịch chuyển gần nhất. Chuyển trạng thái dựa vào điều kiện không quá K lần dịch chuyển giới hạn cho tham số y và sử dụng tối đa L kênh vận chuyển giới hạn cho tham số z.