Miles To Chicago

Link submit: https://www.urionlinejudge.com.br/judge/en/problems/view/1655

Solution:

C++	http://ideone.com/KFItT5
Java	https://ideone.com/60LJqL
Python	https://ideone.com/rlPORF

Tóm tắt đề:

Bạn được cho danh sách gồm N thành phố và M con đường 2 chiều. Mỗi con đường 2 chiều sẽ nối 2 thành phố u và v với nhau và được quy định một xác suất là c, là xác suất bạn có thể đi qua được đường (u v). Bạn hãy tìm xác suất lớn nhất để có thể đi được từ thành phố 1 đến thành phố n.

Input:

Input gồm nhiều test case, test case sẽ kết thúc nếu như số N nhập vào = 0. Các test case được tổ chức như sau:

- Dòng đầu gồm hai số nguyên dương n và m cách nhau bởi một khoảng trắng ($1 \le n \le 100$, $1 \le m \le n * (n - 1) / 2$)
- m dòng sau, dòng thứ i gồm ba số nguyên dương u , v , c với u v là chỉ số của hai thành phố mà con đường đó nối, c là tỉ lệ phần trăm con đường đó có thể đi qua được.

Output:

Với mỗi test case, bạn xuất ra một số thực được làm tròn 6 chữ số sau phần thập phân là phần trăm xác suất mình có thể đi được từ 1 đến N.

Ví dụ:

<table>
<tr>
<td><pre>5 7
5 2 100
3 5 80
2 3 70
2 1 50
3 4 90
4 1 85
3 1 70
0</pre></td>
<td><pre>61.200000 percent</pre></td>
</tr>
</table>

Hướng dẫn giải:

Ta sẽ quy các giá trị phần trăm có thể đi được về xác suất. Tức trọng số của cạnh (u v) từ c thành c / 100.0 . Như vậy, để tìm được phần trăm xác suất có thể đi qua được từ 1 đến N, ta tính xác suất lớn nhất khi đi từ 1 đến N. Như vậy, ta có thể quy về bài toán tìm TÍCH lớn nhất các trọng số trên đường đi từ đỉnh 1 đến đỉnh N. Để giải bài này, ta sử dụng thuật toán Bellman-Ford với đoạn tối ưu như sau:

- Gọi prob[i] là xác suất lớn nhất để đi từ đỉnh 1 đến đỉnh i.
- Ta gán prob[1] = 1.0 , prob[i] = -1 (i > 1)
- Với mỗi cạnh (u v), ta cập nhật: prob[u] = max(prob[u], prob[v] * c) và prob[v] = max(prob[v], prob[u] * c).

Kết quả: prob[n] * 100.0 (làm tròn 6 chữ số sau phần thập phân).

Độ phức tạp: O(T * V * E) với T là số lượng bộ test, O(V * E) là độ phức tạp của Bellman-Ford.