Bài Hướng Dẫn Thực Hành

Tìm đường đi ngắn nhất với Dijkstra

1.Thuật toán Dijkstra

Cho G=(X,E) là một đồ thị có trọng không âm gồm n đỉnh. Thuật toán Dijkstra được dùng để tìm đường đi ngắn nhất từ đỉnh i đến j cho trước.

Gọi L là ma trận trọng lượng (với qui ước $L_{hk} = +\infty$ nếu không có cạnh nối từ đỉnh h đến đỉnh k).

Ta sử dụng thêm hai mảng để lưu vết của quá trình tìm đường đi:

- Length[…] : lưu độ dài từ đỉnh đầu i đến các đỉnh trong đồ thị.
- Last[…] : lưu đỉnh liền trước nó trên đường đi.

Bước 1: Gán T:= X và gán các nhãn:

Length[i] = 0;

Length[k] = $+\infty$ với $\forall k \in X \setminus \{i\}$;

Last[k] = -1; $\forall k \in X$;

Bước 2: Nếu j $\notin$ T thì dừng và giá trị Length[j] là độ dài đường đi ngắn nhất từ i đến j và Last[j] lưu đỉnh nằm ngay trước j trên đường đi đó.

Bước 3: Chọn đỉnh v $\in$ T sao cho Length[v] nhỏ nhất và gán T:= T\{v}.

Bước 4: $\forall k \in T$ và có cạnh nối từ v đến k:

Nếu Length[k] > Length[v] + L_{vk} thì

Length[k] = Length[v] + L_{vk};

Last[k] = v;

Cuối nếu

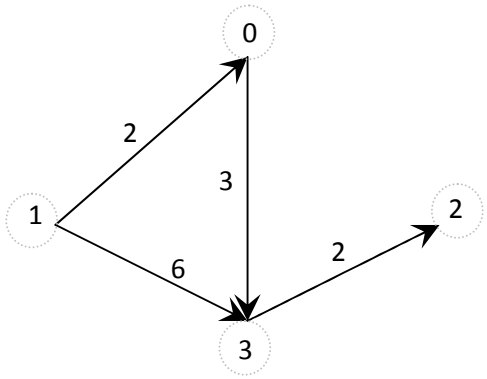
Cuối với mọi

Trở về bước 2

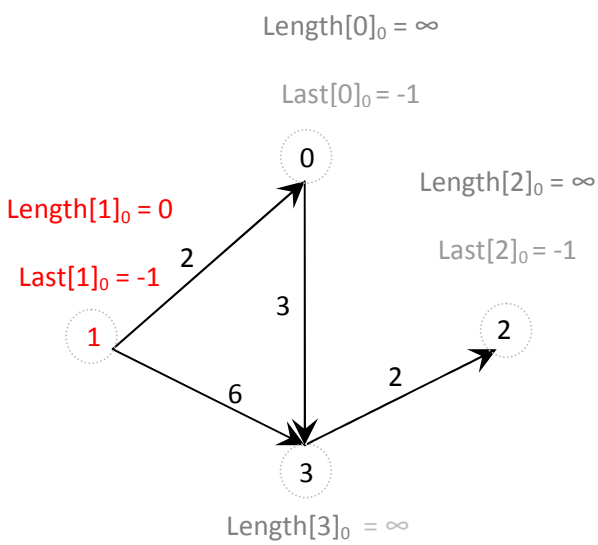
Chú ý: Khi thuật toán dừng, nếu Length[j] = +∞ thì không tồn tại đường đi từ i đến j, nếu ngược lại thì Length[j] là độ dài đường đi ngắn nhất.

2. Ví dụ Dijkstra

Cho đồ thị sau:



Tìm đường đi ngắn nhất từ đỉnh 1 đến đỉnh 2 trong đồ thị.



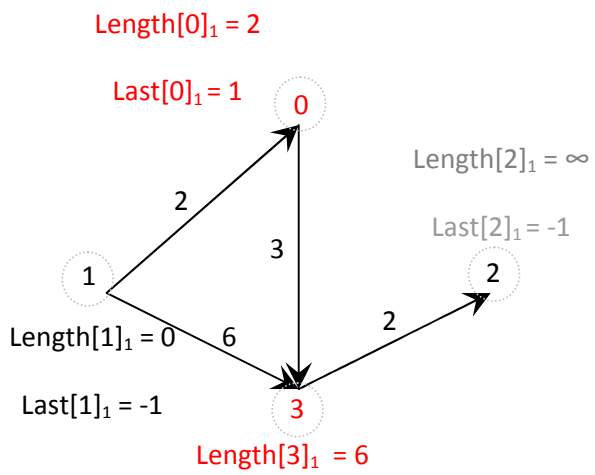
<u>Bước 1</u>: khởi tạo

Khởi tạo đỉnh 1 với độ dài min hiện tại là 0, và nhãn đỉnh trước là chính nó (hoặc -1, điều này không quan trọng). Các đỉnh còn lại đều được gán độ dài min là ∞. Dưới đây là bảng mô tả

T[]\step	0
0	0
1	1
2	2
3	3

Length[]\step	0
0	∞
1	0
2	∞
3	∞

Last[]\step	0
0	-1
1	-1
2	-1
3	-1



<u>Bước 2</u>: đỉnh 2 vẫn thuộc T ta sang bước kế tiếp.

<u>Bước 3</u>: chọn đỉnh có độ dài nó nhỏ nhất. Ở đây là đỉnh 1, ta loại đỉnh này ra.

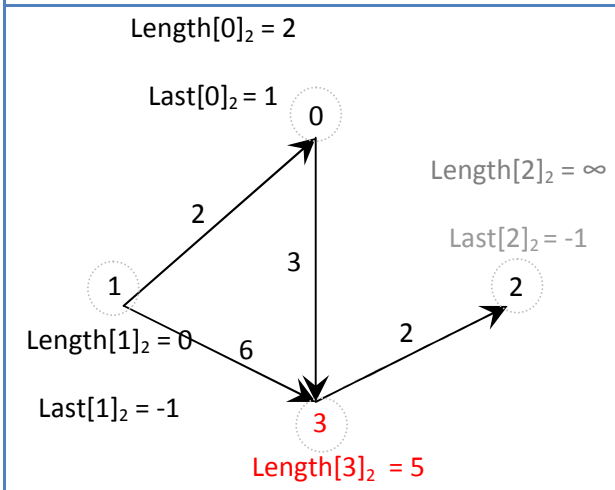
T[]\step	0	1
0	0	0
1	1	~~1~~
2	2	2
3	3	3

<u>Bước 4</u>: Các đỉnh còn lại đều có Length[] cực đại nên ta cập nhật lại như sau

Length[]\step	0	1
0	∞	2
1	0	0
2	∞	∞
3	∞	6

Last[]\step	0	1
0	-1	1
1	-1	-1
2	-1	-1

3	-1	1	



<u>Bước 2 (lần 2)</u>: đỉnh 2 vẫn thuộc T ta sang bước kế tiếp.

<u>Bước 3 (lần 2)</u>: trong **T**, chọn đỉnh có độ dài nhỏ nhất. Ở đây là đỉnh 0, nên:

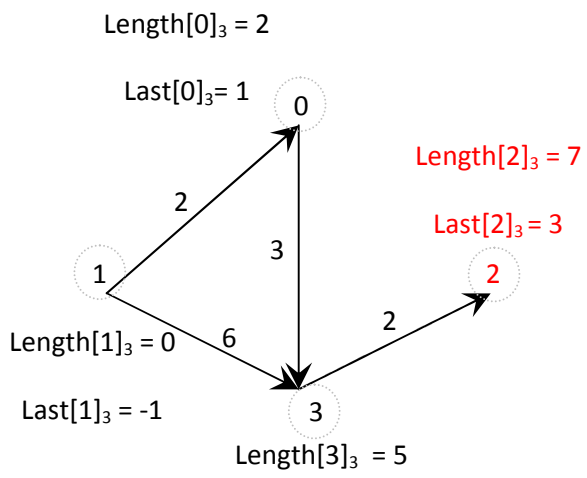
T[]\step	0	1	2
0	0	0	~~0~~
1	1		
2	2	2	2
3	3	3	3

<u>Bước 4 (lần 2)</u>: tính độ dài từ đỉnh 0 vừa xét ở trên đến các đỉnh còn lại trong T.

Đỉnh số 3 có chi phí mới là 2+3=5 nhỏ hơn chi phí cũ (6), vì vậy ta cập nhật lại đỉnh này.

Length[]\step	0	1	2
0	∞	2	2
1	0	0	0
2	∞	∞	∞
3	∞	6	5

Last[]\step	0	1	2
0	-1	1	1
1	-1	-1	-1
2	-1	-1	-1
3	-1	1	0



Bước 2 (lần 3): đỉnh 2 vẫn thuộc T ta sang bước kế tiếp.

Bước 3 (lần 3): lấy đỉnh 3 ra khỏi **T** vì có độ dài nhỏ nhất:

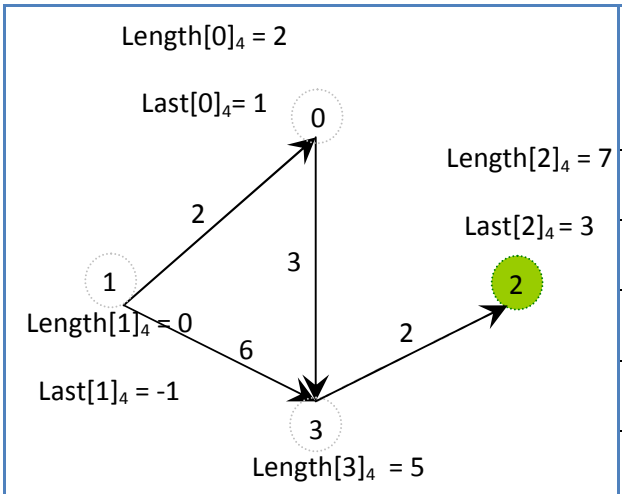
T[]\step	0	1	2	3
0	0	0		
1	1			
2	2	2	2	2
3	3	3	3	~~3~~

Bước 4 (lần 3): tương tự ta cũng tính độ dài từ đỉnh 3 đến đỉnh còn lại trong T.

Đỉnh số 2 có chi phí mới là 5+2=7 < ∞.

Length[]\step	0	1	2	3
0	∞	2	2	2
1	0	0	0	0
2	∞	∞	∞	7
3	∞	6	5	5

Last[]\step	0	1	2	3
0	-1	1	1	1
1	-1	-1	-1	-1
2	-1	-1	-1	3
3	-1	1	0	0



Bước 2 (lần 4): đỉnh 2 vẫn thuộc T.

Bước 3 (lần 4): lấy đỉnh 2 ra khỏi **T** vì có độ dài nhỏ nhất, lúc này

T[]\step	0	1	2	3	4
0	0	0			
1	1				
2	2	2	2	2	~~2~~
3	3	3	3		

Lúc này đỉnh 2 không còn nằm trong T nữa, chúng ta kết thúc thuật toán.

Bài Tập Thực Hành

Thuật toán Dijkstra

I. Quy định:

- Thời gian làm bài: 2 tuần (deadline xem trên moodle)
- Loại bài tập: cá nhân
- Cấu trúc bên trong thư mục **MSSV** bao gồm các thư mục
  - **Source**: thư mục chứa toàn bộ source code
  - **Document**: chứa báo cáo (**MSSV.doc** hoặc **MSSV.docx**)
  - **Release**: thư mục chứa file thực thi (**MSSV.exe**).
- Nén toàn bộ thư mục thành file **MSSV.zip** hoặc **MSSV.rar**
- Nộp bài lên moodle.

Lưu ý: tất cả các bài làm sai quy định sẽ không được chấm tức là 0 điểm.

II. Đề bài:

1. Yêu cầu bắt buộc:

Sử dụng thuật toán Dijkstra tìm đường đi ngắn nhất giữa 2 đỉnh cho trước của đồ thị đơn có hướng và có trọng số dương. Chương trình được viết ở dạng Console với 3 tham số dòng lệnh lần lượt là đường dẫn tập tin đầu vào và đường dẫn tập tin đầu ra:
Ví dụ tham số dòng lệnh:

MSSV.exe input.txt output.txt

Hoặc

MSSV.exe C:\input.txt D:\output.txt

Lưu ý: đường dẫn/ tên tập tin đầu vào và đầu ra có thể thay đổi (không cố định)

Cấu trúc dữ liệu đầu vào:

- Dòng đầu tiên: **số đỉnh đồ thị (N).**
- N dòng tiếp theo: **ma trận kề** của đồ thị với quy ước:
  - A[i][j] > 0: trọng số cạnh nối trực tiếp i và j.
  - A[i][j] = 0: không có cạnh nối trực tiếp i và j.
- Dòng cuối cùng: chỉ số đỉnh bắt đầu s và đỉnh kết thúc g.

Cấu trúc dữ liệu đầu ra:

Nếu không có đường đi ngắn nhất từ s đến g thì xuất -1. Ngược lại xuất các thông tin sau:

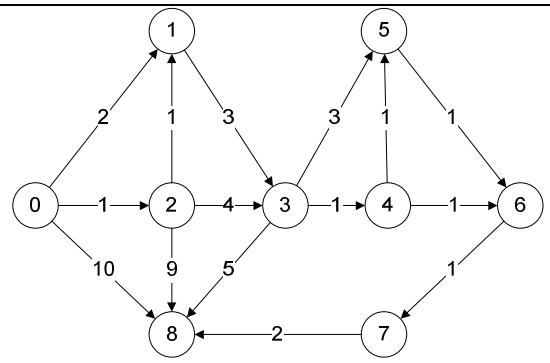
- Dòng đầu tiên: chi phí đi từ s đến g (tổng trọng số các cạnh phải đi qua).
- Dòng thứ hai: đường đi ngắn nhất từ s đến g.

- Trong trường hợp có nhiều đường đi ngắn nhất từ s đến g thì chỉ cần xuất 1 đường đi tùy ý.

2. Yêu cầu nâng cao (cộng điểm):

- Cải tiến thuật toán **Dijkstra** để tìm **tất cả** đường đi ngắn nhất từ s đến g.
- Xuất các đường đi theo thứ tự tăng dần với định nghĩa "*nhỏ hơn*" như sau:
  A < B ⇔
  - Hoặc Số đỉnh (A) < Số đỉnh (B)
  - Hoặc Số đỉnh (A) = Số đỉnh(B) , A(k) < B(k) với k = min(i | A(i) !=B(i))

Ví dụ:

Tập tin đầu vào	*Đồ thị*	*Tập tin kết quả*
9 0 2 1 0 0 0 0 0 10 0 0 0 3 0 0 0 0 0 0 1 0 4 0 0 0 0 9 0 0 0 0 1 3 0 0 5 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 8		10 0 8 0 2 8 0 1 3 8 0 2 3 8 0 2 1 3 8 0 1 3 4 6 7 8 0 2 3 4 6 7 8 0 2 1 3 4 6 7 8

III. Nội dung báo cáo:

Báo cáo cần thể hiện rõ các nội dung sau:

- Ý tưởng thuật toán đã cài đặt (thuật toán Dijkstra và phần cải tiến nếu có).
- Cách tổ chức thiết kế chương trình.