Thuật toán tìm đường đi ngắn nhất Dijkstra

Với bài toán tìm đường đi ngắn nhất, chúng ta có một số thuật toán nổi tiếng giải quyết nó như: Dijkstra hay Floyd.

Thuật toán Dijkstra có 2 loại là:

- Tìm đường đi ngắn nhất từ 1 đỉnh nguồn tới 1 đỉnh đích.
- Tìm đường đi ngắn nhất từ 1 đỉnh nguồn tới các đỉnh còn lại của đồ thị.

Dưới đây là code minh họa cho loại thứ 1 của thuật toán Dijkstra.

```
#include<iostream>

#include<conio.h>

using namespace std;

#define MAX 50

#define TRUE 1

#define FALSE 0

#define VOCUNG 10000000

int n; //s đỉnh của đồ thị.

int s; //đỉnh đầu.

int t; //đỉnh cuối

char chon;

int truoc[MAX]; //mảng đánh dấu đường đi.
```

```
int d[MAX]; //mảng đánh dấu khoảng cách.

int Matrix[MAX][MAX]; //ma trận trọng số

int chuaxet[MAX]; //mảng đánh dấu đỉnh đã được gán nhãn.

void Init(void) {

  freopen("DIJKSTRA.IN", "r", stdin);

  cin >> n;

  cout << "So dinh : " << n << endl;

  cin >> s >> t; //nhập đỉnh đầu và đỉnh cuối của đồ thị.

  //nhập ma trận của đồ thị.

  for (int i = 1; i <= n; i++) {

    for (int j = 1; j <= n; j++) {

      cin >> Matrix[i][j];

      if (Matrix[i][j] == 0) Matrix[i][j] = VOCUNG;

    }

  }

}

void Result(void) {

  cout << "Duong di ngan nhat tu " << (char)(s + 'A' - 1) << " den " << (char)(t + 'A' - 1) <<
" la" << endl;

  cout << (char)(t + 'A' - 1) << "<="; //in đỉnh cuối dưới dạng char.
```

```
  int i = truoc[t];
  while (i != s) {
    cout << (char)(i + 'A' - 1) << "<="; //in ra kết quả dưới dạng char.
    i = truoc[i];
  }
  cout << (char)(s + 'A' - 1); //in đỉnh đầu dưới dạng char.
  cout << endl << "Do dai duong di la : " << d[t];
}
void Dijkstra(void) {
  int u, minp;
  //khởi tạo nhãn tạm thời cho các đỉnh.
  for (int v = 1; v <= n; v++) {
    d[v] = Matrix[s][v];
    truoc[v] = s;
    chuaxet[v] = FALSE;
  }
  truoc[s] = 0;
  d[s] = 0;
  chuaxet[s] = TRUE;
  //bước lặp
```

```
while (!chuaxet[t]) {

  minp = VOCUNG;

  //tìm đ□nh u sao cho d[u] là nh□ nh□t

  for (int v = 1; v <= n; v++) {

    if ((!chuaxet[v]) && (minp > d[v])) {

      u = v;

      minp = d[v];

    }

  }

  chuaxet[u] = TRUE; // u la dinh co nhan tam thoi nho nhat

  if (!chuaxet[t]) {

    //gán nhãn lại cho các đ□nh.

    for (int v = 1; v <= n; v++) {

      if ((!chuaxet[v]) && (d[u] + Matrix[u][v] < d[v])) {

        d[v] = d[u] + Matrix[u][v];

        truoc[v] = u;

      }

    }

  }

}
```

```
}

void main(void) {

  Init();

  Dijkstra();

  Result();

  getch();

}
```

Input:

```
9
1 9
0 6 5 0 8 0 0 0 0
6 0 4 0 0 7 0 0 0
5 4 0 4 0 0 0 0 0
0 0 4 0 4 0 6 0 0
8 0 0 4 0 0 0 6 0
0 7 0 0 0 0 5 0 6
0 0 0 6 0 5 0 4 3
0 0 0 0 6 0 4 0 5
0 0 0 0 0 6 3 5 0

dinh nghia.

1 tuong ung voi dinh A
2 tuong ung voi dinh B
3 tuong ung voi dinh C
4 tuong ung voi dinh D
5 tuong ung voi dinh E
6 tuong ung voi dinh F
7 tuong ung voi dinh G
8 tuong ung voi dinh H
9 tuong ung voi dinh I
```

Output:

```
So dinh: 9

Duong di ngan nhat tu A den I la

I<=G<=D<=C<=A

Do dai duong di la: 18
```

Revision #1
Created 6 May 2023 04:04:37 by Laptrinh.vn
Updated 27 June 2023 09:20:43 by Laptrinh.vn