**Buổi 07 Tìm đường đi ngắn nhất - Giải thuật Dijkstra, Floyd**

1. **Tìm đường đi ngắn nhất từ một đỉnh đến các đỉnh còn lại**

***Giải thuật Dịkstra***

Gọi s là đỉnh xuất phát (với đồ thị ma trận kề)

dist[] là array với d[v] lưu độ dài ngắn nhất từ s 🡪 v

pre[] là array với pre[v] lưu đỉnh nằm ngay trước đỉnh v trên đường đi ngắn nhất từ s 🡪 v

T là tập đỉnh, ban đầu : T = V (tập đỉnh của đồ thị)

Giải thuật :

void Dijkstra()

{ For(v **∈** V)

{ dist[v] = a[s,v] ;

pre[v] = s ;

}

dist[s] = 0 ; T = V; T = V \ {s} ;

while(T != ∅)

{ Tìm u **∈** T sao cho : dist[u] = min { d[z] , mọi z **∈** T }

T = T \ {u};

for(v **∈** T)

Nếu ( dist[v] > dist[u] + a[u,v])

{

dist[v] = dist[u] + a[u,v];

pre[v] = u ;

}

}

}

Các bước thực hiện thuật toán được minh họa như sau :

Chọn s = 0 là đỉnh xuất phát

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 |
| 0 | ∞ | 1 | ∞ | ∞ | 7 |
| 1 | ∞ | ∞ | 1 | 4 | 8 |
| 2 | ∞ | ∞ | ∞ | 2 | 4 |
| 3 | ∞ | ∞ | 1 | ∞ | ∞ |
| 4 | ∞ | ∞ | ∞ | 4 | ∞ |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| T | dist[1] pre[1] | dist [2] pre[2] | dist [3] pre[3] | dist [4] pre[4] |  |
| 1 , 2 , 3 , 4 | 1 0 | **∞** 0 | **∞** 0 | 7 0 |
| 2 , 3 , 4 |  | 2 1 | 5 1 | 7 0 |
| 3 , 4 |  |  | 4 2 | 6 2 |
| 4 |  |  |  | 6 2 |
| ∅ | 1 0 | 2 1 | 4 2 | 6 2 |

Cài đặt

|  |  |  |
| --- | --- | --- |
| Đồ thị(có hướng, có trọng số) | Graph.txt | Output : xuất lên màn hình |
| 1    1  8 4 4 2 1  7  4 | 5  0 1 0 0 7  0 0 1 4 8  0 0 0 2 4  0 0 1 0 0  0 0 0 4 0 | Nhập đỉnh xuất phát x = ? (giả sử x = 0)  Đỉnh : 0 1 2 3 4  pre[] : 0 0 1 2 2  dist[] : 0 1 2 4 6 |

* Tạo file : Graph.txt với nội dung như trên

Sử dụng class WeightMatrix.cs đã làm bài trong buổi 1 (đồ thị biểu diễn theo ma trận kề), trong đó viết các phương thức :

public void Dijkstra(int s)

{

// Khởi tạo pre : Lưu đỉnh nằm trước trên đường từ s đi qua

// Khởi tạo dist : Lưu độ dài ngán nhất từ s đến các đỉnh còn lại

// Khởi tạo visited : Đánh dấu đỉnh đã đi qua

// Khởi gán giá trị ban đầu cho dist[], pre[], visited[]

for (int i = 0; i < n; i++)

{

dist[i] = int.MaxValue;

visited[i] = false;

}

// cho : dist[s] = 0;

// Duyệt các đỉnh của đồ thị

for (int i = 0; i < n; i++)

{

// Gọi u là đỉnh sao cho dist[u] nhỏ nhất (u = dmin())

// Đánh dấu u

// Duyệt các đỉnh v thuộc Kề(u) (tồn tại cạnh (u, v) )

{

// Nếu (v chưa đánh dấu) và (tồn tại cạnh (u,v))

// Nếu (dist[u] đã xác định) và (dist[u] + a[u, v] < dist[v])

{

// Đặt dist[v] = dist[u] + a[u, v];

// Đặt đỉnh u trước v trên đường đi

}

}

}

// Kết quả : xác định giá trị các phần tử của dist[] và pre[] 🡪 xuất lên màn hình

}

// Tìm đỉnh minIndex có dist[minIndex] là nhỏ nhất

public int dmin()

{

}

// Xuất kết quả dist[] và pre[]

public void PrintDijkstra(int s)

{

}

1. **Bài tập**

*Bài 1* : Tìm đường đi ngắn nhất từ đỉnh x đến đỉnh y theo thuật toán Dijkstra

|  |  |  |
| --- | --- | --- |
| Đồ thị | MatrixBai1.txt | Output |
|  | 7  0 3 18 6 9 15 0  3 0 12 2 0 0 0  18 12 0 7 0 0 4  6 2 7 0 3 8 0  9 0 0 3 0 4 0  15 0 0 8 4 0 2  0 0 4 0 0 2 0 | (giả sử nhập x=1 và y=5)  Nhập đỉnh xuất phát x = 1  Nhập đỉnh đích y = 5  Đường đi ngắn nhất từ 1 đến 5 :  1 -> 3 -> 4 -> 5 -> độ dài : 9 |

Giải thuật :

public void MinRouteXY(int x, int y)

{

Gọi Dijkstra(x) với đỉnh xuất phát x 🡪 kết quả có dist[] và pre[]

🡪 xuất đường đi và độ dài ngắn nhất từ x đến y

}

*Bài 2* : Đường đi ngắn nhất từ đỉnh x đến đỉnh z qua đỉnh trung gian y

* Sử dụng đồ thị bài 1 (MatrixBai1.txt)
* Giải thuật : Đường đi ngắn nhất (x 🡪 y 🡪 z) =

đường đi ngắn nhất (x 🡪 y) + đường đi ngắn nhất (y 🡪 z)

public void MinRouteXYZ(int x, int y, int z)

{

Gọi Dijkstra(x) với đỉnh xuất phát x 🡪 kết quả có dist[] và pre[]

🡪 Lưu đường đi ngắn nhất (x 🡪 y) vào stack và d1 = dist[y]

Gọi Dijkstra(y) với đỉnh xuất phát y 🡪 kết quả có dist[] và pre[]

🡪 Lưu đường đi ngắn nhất (y 🡪 z) vào stack và d2 = dist[z]

Từ Stack : Xuất độ dài và đường đi : x 🡪 ... 🡪 y 🡪 ... 🡪 z và

Độ dài ngắn nhất : d1 + d2

}

*Bài 3* : Đường đi ngắn nhất giữa các cặp đỉnh – giải thuật **Floyd**

|  |  |  |
| --- | --- | --- |
| Đồ thị | MatrixBai3.txt | Output : xuất 2 ma trận d và p |
|  | 4  0 3 6 9  3 0 2 0  6 2 0 2  9 0 2 0 | dist :  0 3 5 7  3 0 2 4  5 2 0 2  7 4 2 0  pre :  0 0 1 2  1 1 1 2  1 2 2 2  1 2 3 3 |

Giải thuật:

Ma trận d[i,j] : lưu độ dài ngắn nhất từ đỉnh i đến đỉnh j.

Ma trận p[i,j] : lưu đỉnh nằm ngay trước j trên đường đi ngắn nhất từ i đến j

* + Bước 1 (Khởi tạo)
  + Bước 2 (Lặp)

for (int k=0; k<n; k++)

for (int i=0; i<n; i++)

for (int j=0; j<n; j++)

if (d[i,j]>d[i,k]+d[k,j])

{

d[i,j] = d[i,k]+d[k,j];

p[i,j] = p[k,j];

}

Bài tập : Viết Dijkstra với đồ thị danh sách kề