Mục lục

Lời Nói Đầu …………………………………………………………………3

Chương 1 .Lí thuyết Về thuật toán Dijkstra………………………………….4

I.Lịch sử hình thành………………………………………………………….4

II.Thuật toán Dijkstra………………………………………………………...4

1. Cây đường đi ngắn nhất……………………………………………………4

2. Biểu diễn mạng bằng đồ thị………………………………………………..4

3. Thuật toán tìm đường đi ngắn nhất Dijkstra……………………………….5

4. Các bước tìm đường đi ngắn nhất sử dụng thuật toán Dijkstra với ví dụ trên…………………………………………………………………………….6

III.Ứng dụng của thuật toán Dijkstra………………………………………….8

1. Khám phá các láng giềng và học các địa chỉ mạng của chúng……………..8

2. Đo độ trễ (delay), hay giá (cost) tới các láng giềng………………………..8

3. Xây dựng một gói tin báo cho các trạng thái thông tin vừa học……………8

4. Gửi gói tin cập nhật đến tất cả các routers khác……………………………9

5. Tính đường dẫn ngắn nhất cho từng routers………………………………..9

6. Đánh giá vê Link-State……………………………………………………10

Chương 2 .Mô phỏng định tuyến thuật toán trên Scilab…………………….11

I .Phân tích yêu cầu………………………………………………………….11

II. Sử dụng modul hỗ trợ việc định tuyến Dijkstra………………………….11

III.Mô phỏng định tuyến Dijkstra trên Scilab……………………………….11

1. Các hàm cơ bản ban đầu giúp việc chuyển dữ liệu từ file

input.txt để xử lí………………………………………………………11

1. Các hàm đóng vai trò định tuyến……………......................................13
2. Hàm tạo đồ thị trên giao diện………………………………………....17
3. Hàm Main……………………………………………………………..18
4. File Input……………………………………………………………...18
5. Kết quả trên giao diện………………………………………………...18

Chương 3.Kết luận………………………………………………………......23

Tài liệu tham khảo…………………………………………………………...24

Lời Nói Đầu

Ngày nay với sự ra đời của Internet đã làm thay đổi cuộc sống của chúng ta, sô lượng người sử dụng mạng Internet tăng lên không ngừng, vấn đề định tuyến cho dữ liệu trên đường truyền Internet là vô cùng quan trọng, nó giúp cho hệ thống Internet hoạt động hiệu quả, phòng tránh tắc nghẽn, đảm bảo đường truyền tin cậy cho người dùng Internet.

Trong quá trình học tập và nghiên cứu về vấn đề định tuyến cho mạng Internet nhóm chúng em quyết định:” Tìm hiểu về thuật toán Dijkstra và mô phỏng thuật toán trên môi trường Scilab” để giúp chúng em tìm hiểu, và hoàn thành môn học Project 2.

Do thời gian thực hiện đề tài tương đối rộng cũng như sự hiểu biết của chúng em có hạn và cũng do có nhiều nguyên nhân khách quan và chủ quan khác nên trong quá trình làm đề tài cũng không thể tránh được các sai sót mong cô bỏ qua cho chúng em.

Để hoàn thành được đề tài này chúng em xin gửi lời cảm chân thành đến giáo viên hướng dẫn là cô Trương Thị Diệu Linh –giảng viên Viện Công Nghệ Thông Tin và Truyền Thông, thầy đã tận tình giúp đỡ chúng em trong quá trình học tập cũng như tìm hiểu về đề tài này.

Chương 1: Lí thuyết về thuật toán Dijkstra

1. Lịch sử hình thành

Thuật toán Dijkstra mang tên nhà khoa học máy tính Edsger Dijkstra vào năm 1956 và được ấn bản vào năm 1959. Thuật toán Dijkstra được biết đến là một trong những thuật toán giải quyết bài toán tìm đường đi ngắn nhất giữa hai nút trong đồ thị có hướng không có cạnh mang trọng số âm.

Thuật toán có nhiều ứng dụng trong nhiều lĩnh vực đặc biệt là lĩnh vực truyền thông trong mạng Internet.

Sử dụng thuật toán Dijkstra trong việc xây dựng các giao thức định tuyến gói tin trong mạng internet ở tầng 3 trong mô hình mạng OSI.

1. Thuật toán Dijkstra
2. *Cây đường đi ngắn nhất*

y

x

w

u

z

u

y

x

w

v

z

2

2

1

3

1

1

2

5

3

5

v

-Các cạnh xuất phát từ nút gốc tới nút lá

-Đường đi duy nhất từ nút gốc tới nút v, là đường đi ngắn nhất giữa nút gốc và nút v

-Mỗi nút sẽ có 1 cây đường đi ngắn nhất giữa riêng nút đó.

1. *Biểu diễn mạng bằng đồ thị*

-Đồ thị với các nút (bộ định tuyến) và các cạnh (các liên kết)

-Chi phí cho việc thực hiện các liên kết c(x,y)

+ Băng thông, độ trễ, chi phí, mức độ tắc nghẽn, …

-Giải thuật chọn đường, xác định đường đi ngắn nhất giữa hai nút tùy ý

u

y

x

w

v

z

2

2

1

3

1

1

2

5

3

5

1. *Thuật toán tìm đường đi ngắn nhất Dijkstra*

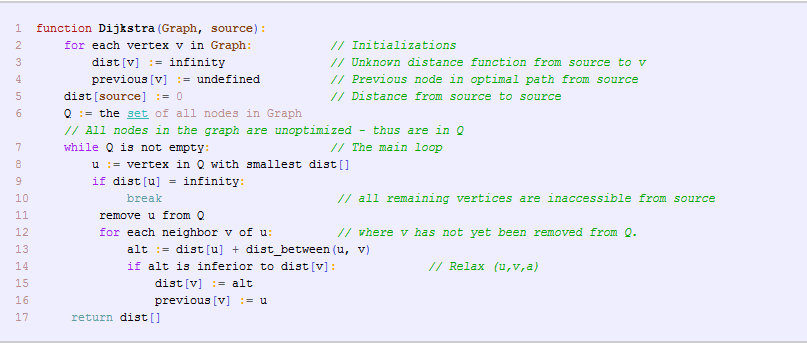
-Thuật toán tìm đường đi ngắn nhất giữa 1 đỉnh tới tất cả các đỉnh còn lại trong đồ thị có trọng số không âm

-Độ phức tạp thính toán của đồ thị là: O(n2 +m)

Trong đó m là số cạnh, n là số đỉnh của đồ thị đang xét.

-Để minh họa các giải thuật tìm đường thông thường ta thường kí hiệu N là số nodes trong mạng và i j là các nhãn của các nodes trong mạng

\* Đối với Thuật toán Dijkstra



Ta kí hiệu:

G = (V, E):Đồ thị với tập đỉnh *V* và tập cạnh *E*

C(x,y):chi phí của liên kết *x* tới *y*; = ∞ nếu không

phải 2 nút kế nhau

d(v):chi phí hiện thời của đường đi từ nút nguồn tới

nút đích v.

p(v):nút ngay trước nút *v* trên đường đi từ nguồn tới

đích.

T:Tập các nút mà đường đi ngắn nhất đã được xác

định.

\*Các hàm sử dụng trong mã giả

-Init

Với mỗi nút v, *d[v]* = ∞, *p[v]* = NIL *d[s]* = 0

-Update (u,v) trong đó u, v là một cạnh nào đó của G

if *d[v] > d[u] + c(u,v)* then

*d[v] = d[u] + c(u,v) p[v] = u*

\*Viết lại code theo mã giả:

1.Init() ;

2. *T* = Φ*;*

3. Repeat

4. *u:* u ∈ *T* | *d(u)* là bé nhất ;

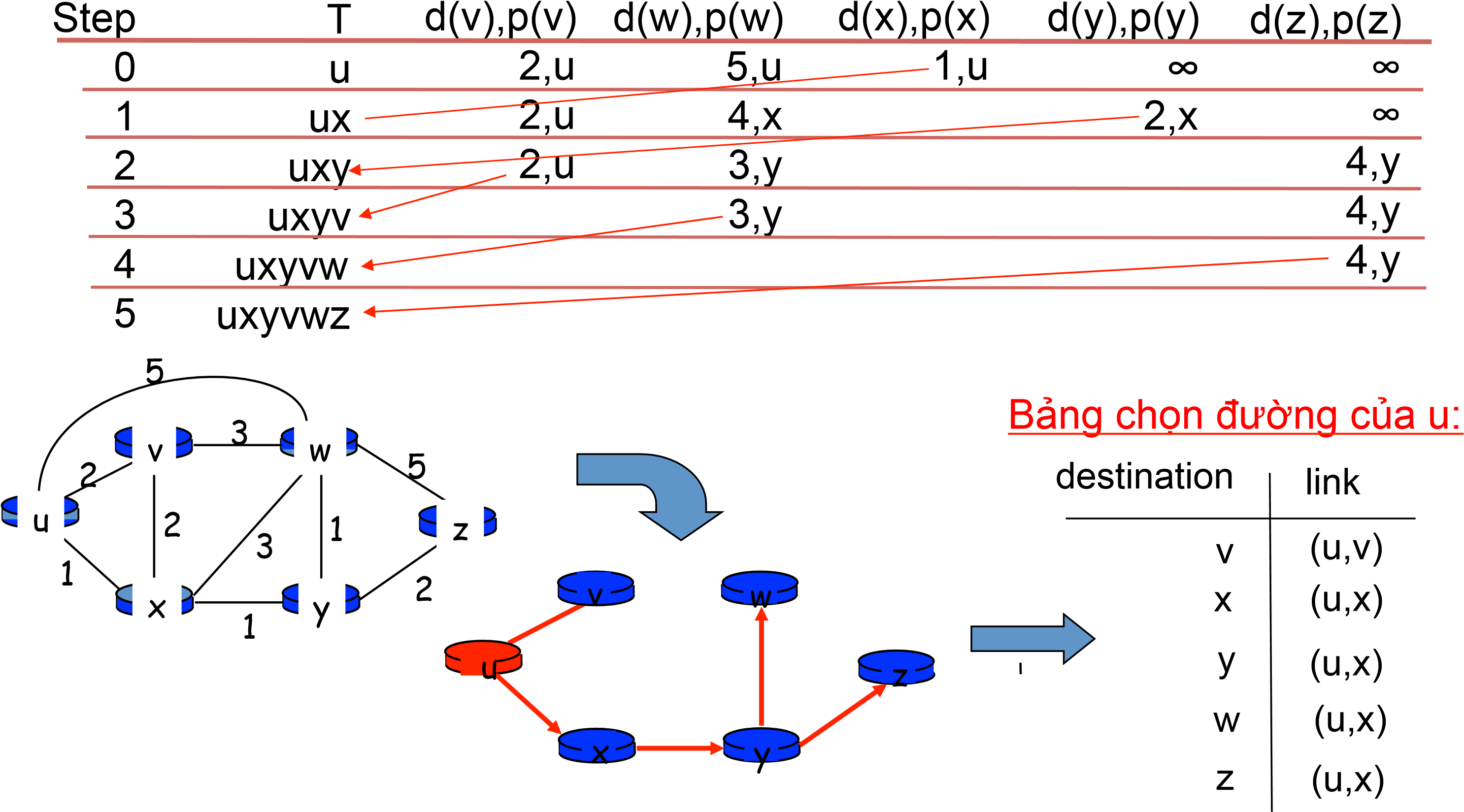
5. *T = T* ∪ *{u};*

6. for all *v* ∈ neighbor(u) và *v* ∉*T*

7. *update(u,v)* ;

8. Until *T = V*

1. *Các bước tìm đường đi ngắn nhất sử dụng thuật toán Dijkstra với ví dụ trên*



\*Giải thuật chọn đường đi ngắn nhất





1. Ứng dụng của thuật toán Dijkstra

\**Tìm đường Link-State*

1. Khám phá các láng giềng và học các địa chỉ mạng của chúng

- Khám phá các routers láng giềng bằng cách gửi 1 gói tin HELLO trên mỗi đường dẫn

-Khi 2 hay nhiều routers kết nối bởi một LAN, tình huống sẽ phức tạp



Router được nối qua 1 mạng Lan, mạng Lan được coi như là một nút ảo.

1. Đo độ trễ (delay), hay giá (cost) tới các láng giềng

-Các routers phải có sự ước lượng về các đường dẫn tới các routers láng giếng để làm trọng số cho giải thuật định tuyến.



1. Xây dựng một gói tin báo cho các trạng thái thông tin vừa học:

-Gói tin bắt đầu với định danh của máy gửi, theo sau là thứ tự trạng thái, tuổi (age) và một danh sách các láng giềng. Thông thường các gói tin trạng thái được xây dựng một cách định kỳ.



Ví dụ cho các gói tin trạng thái cho subnet

1. Gửi gói tin cập nhật đến tất cả các routers khác:

- Sử dụng thuật toán ngập lụt (flooding) để gửi các gói tin trạng thái,

-Các gói tin chứa thông tin về tuổi (age) để tránh trùng lặp và cập nhật thông tin. Khi bộ đếm tuổi quay về zero, thông tin về routers đấy sẽ bị hủy.

- Trường tuổi cũng giảm theo từng routers trong quá trình ngập lụt để đảm bảo không có gói tin nào có thể tồn tại vô hạn,

-Các gói tin trạng thái thường được lưu vào bộ nhớ đệm để xử l. tuần tự, nếu có trùng lặp sẽ bị loại bỏ.



Ví dụ về buffer đệm lưu trữ gói tin trạng thái của router B

1. Tính đường dẫn ngắn nhất cho từng routers:

-Sau khi các routers có đầy đủ thông tin trạng thái các đường dẫn sẽ sử dụng thuật toán Dijkstra để tính toán xây dựng đường dẫn ngắn nhất cho mọi nơi đến có thể.

-Chọn đường trạng thái liên kiết (link-state) được dùng nhiều trong các mạng hiện nay,

-Các giao thức chọn đường trạng thái liên kết phổ biến là OSPF (Open Shortest Path First) và IS-IS (Intermediate System-Intermediate System).

1. Đánh giá vê Link-State

- Bộ định tuyến dùng nhiều bộ nhớ và thực thi nhiều hơn so với các giao thức định tuyến theo vectơ khoảng cách.

- Lí do cần thiết phải lưu trữ thông tin của tất cả các láng giềng, cơ sở dữ liệu mạng đến từ các nơi khác và việc thực thi các thuật toán định tuyến trạng thái.

- Các nhu cầu về băng thông cần phải tiêu tốn để khởi động sự phát tán gói trạng thái

Chương 2. Mô phỏng định tuyến thuật toán trên Scilab

I .Phân tích yêu cầu

Đề tài: “Tìm hiều về thuật toán Dijkstra và mô phỏng định tuyến thuật toán trên Scilab “

Yêu cầu:

* Phần mềm Scilab
* Tạo ra giao diện định tuyến giữa các nodes
* In ra đường dẫn giữa các nodes khi định tuyến
* Có thể định tuyến giữa hai nodes bất kì trong danh sách các nodes hoặc giữa 1 điểm và tất cả các điểm còn lại

Khó Khăn

* Scilab chưa hỗ trợ code cho thuật toán Scilab
* Scilab chưa có những hàm toán học hỗ trợ cho việc tạo giao diện của các nodes

II.Sử dụng modul hỗ trợ việc định tuyến Dijkstra

Để giải quyết 2 khó khăn trong khi mô phỏng thuật toán Dijkstra chúng em đã sử dụng 2 modul mở rộng của Scilab để thực hiện quá trình mô phỏng

* Modul mở rộng Metanet : modul này hỗ trợ chúng em tạo ra 1 cây đồ thị trong chế độ GUI
* Modul mở rộng Network Topology Generator (NTG): Modul này hỗ trợ việc định tuyến theo thuật toán Dijkstra

Cách hướng dẫn cài đặt hai Modul này:

Sau khi cài đặt xong Scilab, khởi động ứng dụng Scilab chọn vào Application trên thanh TarBar rồi chọn vào Module Manager ATOMS. Sau khi chọn nó hộp thoại Module Manage sẽ xuất hiện chọn vào mục Educaton sau đó chọn Network Topology Generator và sau đó phần bên phải của hộp thoại sẽ xuất hiện mô tả cho gói module này,sau đó chọn download , chờ đợi cho đến khi hoàn thành, khởi động lại Scilab thì trong Scilab đã có cả 2 module ở trên do module NTG luôn cần module metanet để hoạt động

III.Mô phỏng định tuyến Dijkstra trên Scilab

1.Các hàm cơ bản ban đầu giúp việc chuyển dữ liệu từ file input.txt để xử lí

Các hàm này viết trong file add\_value.sci

*//Hàm này làm nhiệm vụ lấy giá trị từ 1 biến và add vào 1 list*

function {B}=add\_value(A,i)

[r,c]=size(A);

A(1,c+1)=i;

B=A;

endfunction

*//Hàm này đọc data từ đường dẫn cho trước và trả lại 2 list là điểm đầu và điểm cuối của 1 list*

function [**ta2**, **he2**]=convert\_matrix\_to\_list(**pathfile**, **k**)

f=read(**pathfile**,-1,**k**)

[r,c]=size(f); *//kich cớ 2 chiều của matrix*

ta1=[];

he1=[];

for j=1:r

for i=1:c

temp=f(i,j);

if(temp <> 0) then

ta1=add\_value(ta1,i);

he1=add\_value(he1,j);

end

end

end

**ta2**=ta1;

**he2**=he1;

endfunction

endfunction

*// Lấy 2 giá tri đã có và copy vào 1 vector*

function [**x**, **y**]=add\_list(**i1**, **j1**, **x1**, **y1**)

[r1,c1]=size(**x1**)

[r2,c2]=size(**y1**)

**x1**(1,c1+1)=**i1**

**y1**(1,c2+1)=**j1**

**x**=**x1**

**y**=**y1**

endfunction

*//Tạo ra các tọa độ của đồ thị bất kì đầu vào là số điểm cần tạo*

function [**x**, **y**]=creat\_point(**k**)

x1=[]

y1=[]

for m=1:**k**

[i,j]=Random\_i\_j(800) *//Lấy bất kì 2 số i, j trong khoảng nhỏ hơn 800*

[x1,y1]=add\_list(i,j,x1,y1)

end

**x**=x1;

**y**=y1;

endfunction

*//Ham đọc từ file mục đích lấy ra ma trận kề*

function [**M**]=Read\_file(**pathfile**, **k**)

f=read(**pathfile**,-1,**k**)

**M**=f

endfunction

2.Hàm các hàm đóng vai trò định tuyến dijkstra

a.Hàm dijskstra

Mục đích viết hàm này : Sử dụng giúp định tuyến từ 1 điểm s tới điểm d trên matrix kề A, hàm trả về đường dẫn từ d tới s là matrix L và có độ dài cost là e

function [**e**, **L**]=dijkstra(**A**, **s**, **d**)

*// Output variables initialisation (not found in input variables)*

**e**=[];

**L**=[];

*// Display mode*

mode(0);

*// Display warning for floating point exception*

ieee(1);

*//---------------------------------------------------*

*// [cost route] = dijkstra(Graph, source, destination)*

*//*

*// example*

*// G = [0 3 9 0 0 0 0;*

*// 0 0 0 7 1 0 0;*

*// 0 2 0 7 0 0 0;*

*// 0 0 0 0 0 2 8;*

*// 0 0 4 5 0 9 0;*

*// 0 0 0 0 0 0 4;*

*// 0 0 0 0 0 0 0;*

*// ];*

*// [e L] = dijkstra(G,1,7)*

*//Đây là ví dụ em test hàm dijkstra*

*//---------------------------------------------------*

if mtlb\_logic(mtlb\_double(**s**),"==",mtlb\_double(**d**)) then

**e** = 0;

**L** = **s**;

else

**A** = setupgraph(**A**,%inf,1);

*//Lấy tất cả các giá tri =0 chuyển thành infinity*

if mtlb\_logic(mtlb\_double(**d**),"==",1) then

**d** = **s**;

end;

*//Kiểm tra nếu d=1 thì đổi d=s*

**A** = exchangenode(**A**,1,**s**);*//Hoán đồi vị trí hàng s với hàng 1 và cột s với*

*//cột 1*

lengthA = size(mtlb\_double(**A**),1);

W = zeros(lengthA,lengthA); *// tạo matrix 0 có kích cỡ của A*

for i = 2:lengthA

W(1,i) = i; *//tất cả các phần tử hàng 1 của W thành các điểm*

W(2,i) = **A**(1,i); *//tất cả phần tử hàng 2 thành hàng 1 của matrix A*

end;

for i = 1:lengthA

D(i,1) = **A**(1,i);

D(i,2) = i;

end;

D2 = D(2:max(size(D)),:);

**L** = 2;

while **L**<=(size(mtlb\_double(W),1)-1)

**L** = **L**+1;

D2 = mtlb\_sortrows(D2,1);

k = D2(1,2);

W(**L**,1) = k;

D2(1,:) = [];

for i = 1:size(D2,1)

if mtlb\_logic(D(D2(i,2),1),">",D(k,1)+mtlb\_double(**A**(k,D2(i,2)))) then

D(D2(i,2),1) = D(k,1)+mtlb\_double(**A**(k,D2(i,2)));

D2(i,1) = D(D2(i,2),1);

end;

end;

for i = 2:max(size(mtlb\_double(**A**)))

W(**L**,i) = D(i,1);

end;

end;

if mtlb\_logic(mtlb\_double(**d**),"==",mtlb\_double(**s**)) then

**L** = 1;

else

**L** = **d**;

end;

**e** = W(size(mtlb\_double(W),1),**d**);

**L** = listdijkstra(**L**,W,**s**,**d**);

end;

endfunction

*b. Hàm setupgraph*

Mục đích sử dụng hàm này : Giúp cho việc chuyển tất cả các cạnh trên matrix kề thành infininy

function [**G**]=setupgraph(**G**, **b**, **s**)

*// Display mode*

mode(0);

*// Display warning for floating point exception*

ieee(1);

if mtlb\_logic(mtlb\_double(**s**),"==",1) then

for i = 1:size(mtlb\_double(**G**),1)

for j = 1:size(mtlb\_double(**G**),1)

if mtlb\_double(**G**(i,j))==0 then

**G**(i,j) = **b**;

end;

end;

end;

end;

if mtlb\_logic(mtlb\_double(**s**),"==",2) then

for i = 1:size(mtlb\_double(**G**),1)

for j = 1:size(mtlb\_double(**G**),1)

if mtlb\_logic(mtlb\_double(**G**(i,j)),"==",mtlb\_double(**b**)) then

**G**(i,j) = 0;

end;

end;

end;

end;

endfunction

*c.Hàm Exchangenode*

Mục đích sử dụng hàm này: Hoán đổi vị trí cảu cột thứ a với cột thứ b và hàng thứ a với hàng thứ b của matrix kề

function [**G**]=exchangenode(**G**, **a**, **b**)

*// Display mode*

mode(0);

*// Display warning for floating point exception*

ieee(1);

*//Hoán vị cột a với cột b*

buffer = **G**(:,**a**);

**G**(:,**a**) = **G**(:,**b**);

**G**(:,**b**) = buffer;

*//Hoán vị hàng a với hàng b*

buffer = **G**(**a**,:);

**G**(**a**,:) = **G**(**b**,:);

**G**(**b**,:) = buffer;

endfunction

*d.Hàm ListDijkstra*

Mục đích sử dụng hàm : Hỗ trợ với hàm Dijkstra để đưa ra đường dẫn định tuyến

function [**L**]=listdijkstra(**L**, **W**, **s**, **d**)

*// Display mode*

mode(0);

*// Display warning for floating point exception*

ieee(1);

index = size(mtlb\_double(**W**),1);

while index>0

if mtlb\_double(**W**(2,**d**))==mtlb\_double(**W**(size(mtlb\_double(**W**),1),**d**)) then

**L** = [**L**,**s**];

index = 0;

else

index2 = size(mtlb\_double(**W**),1);

while index2>0

if mtlb\_logic(mtlb\_double(**W**(index2,**d**)),"<",mtlb\_double(**W**(index2-1,**d**))) then

**L** = [**L**,**W**(index2,1)];

**L** = listdijkstra(**L**,**W**,**s**,**W**(index2,1));

index2 = 0;

else

index2 = index2-1;

end;

index = 0;

end;

end;

end;

endfunction

*3. Hàm tạo ra đồ thị trên giao diện*

*a. Hàm tạo đồ thị trên giao diện*

load('C:\Users\Nam\Documents\Scilab\Convert\input.txt')

function [**g3**]=creat\_graph(**k**)

*// load('C:\Users\Nam\Documents\Scilab\Project Update\add\_value.sci')*

[ta,he]=convert\_matrix\_to\_list('C:\Users\Nam\Documents\Scilab\Convert\input.txt',**k**)

*//Hàm trên được viết trong file add\_value.sci*

[g2]=make\_graph("Graph",1,**k**,ta,he)

*//Tạo 1 đồ thị trên giao diện với k điểm và số cạnh là size của 2 vector ta , he*

*//Hai vector này , ta là 1 vector ghi lại những giá trị điểm cuối của vector chứa cạnh trong đồ thị*

*//Vector he này là 1 vector ghi lại những giá trị điểm đầu của vector chứa cạnh trong đồ thị*

*//kích cỡ của te và he bằng nhau và đó là số cạnh của đồ thị ( đồ thị có hướng)*

[x1,y1]=creat\_point(**k**)

*//Hàm trên được viết trong file add\_value.sci*

g2.nodes.graphics.x=x1

g2.nodes.graphics.y=y1

g2.nodes.graphics.name=string(1:**k**) *//Đặt các nút từ 1 đến k*

g2.nodes.graphics.display="name" *// Hiện tên các điểm lên dồ thị*

**g3**=g2

endfunction

*b.Hàm giúp định tuyến từ 1 điểm và hiển thị các điểm lên trên đồ thị(giao diện đồ họa)*

function [**g2**, **path1**, **M**]=Routing(**k1**, **s**, **d**, **g1**)

[M1] =Read\_file('C:\Users\Nam\Documents\Scilab\Convert\input.txt',**k1**)

*//Hàm trên được viết trong file add\_value ,đọc ra matrix kề từ file*

[cost path] = dijkstra(M1,**s**,**d**)

*//Hàm này giúp định tuyến giữa 2 điểm đầu và điểm đích của 1 matrix kề và trả lại cost và path từ điểm đích tới điểm nguồn*

**g1**.nodes.graphics.colors(2,path)=color('red')*//tô đậm tọa dộ các node tham gia định tuyến*

**path1**=path *//Lấy đầu ra là đường dẫn từ điểm đích tới nguồn*

**g2**=**g1**

**M**=M1 *//Lấy giá trị của matrix kề*

endfunction

*4.Hàm Main*

*//Load file input*

load('C:\Users\Nam\Documents\Scilab\Convert\input.txt')

k=5 *// số điểm trong đồ thị*

[i,j]=Random\_i\_j(k) *//Lấy 2 điểm i j bất kì trên đồ thị để định tuyến*

[g4]=creat\_graph(k) *//Tạo ra cây đồ thị*

show\_graph(g4)

[g2,path,M2]=Routing(k,i,j,g4) *//Định tuyên trên matrix kề*

path

show\_graph(g2)

*5.File input.txt*

0 5 0 3 4

0 0 3 3 4

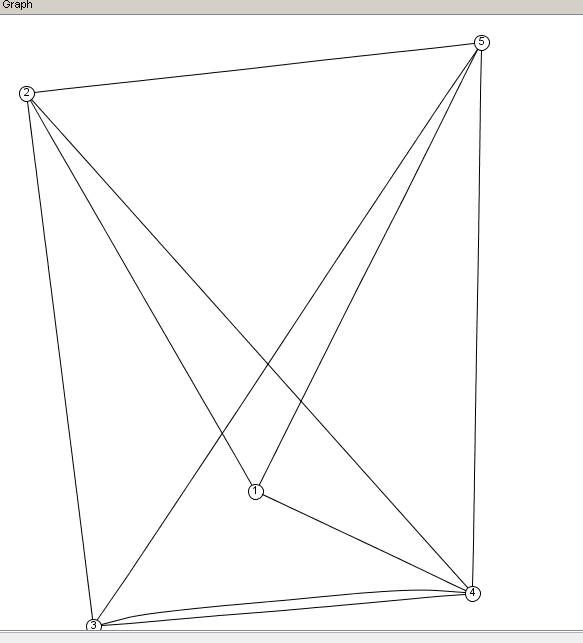
0 0 0 2 1

0 0 2 0 4

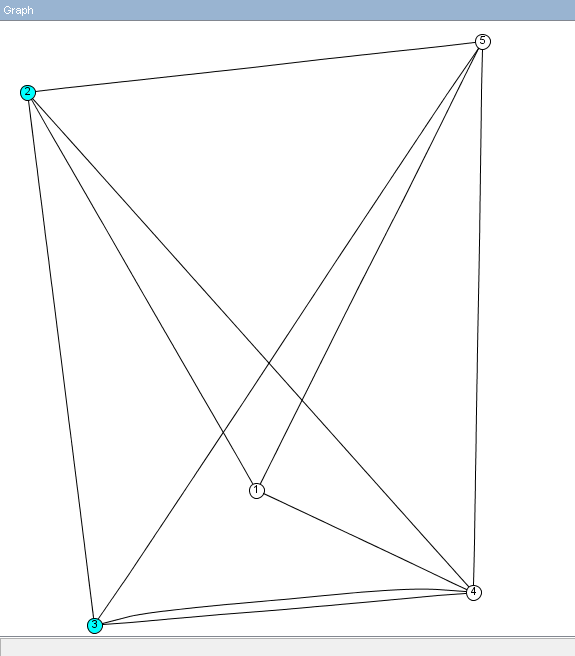
1. 0 0 0 0

*6.Kết quả trên giao diện*

**Đồ thị tạo ra trên giao diện**



**Kết quả định tuyến từ 3->2**



**Kết quả trên Console**

>k=5

k =

5.

-->[i,j]=Random\_i\_j(k)

j =

2.

i =

3.

-->[g4]=creat\_graph(k)

g4 =

graph name : Graph

version : 5.0.1

oriented : yes

number of nodes : 5

number of edges : 10

-->show\_graph(g4)

ans =

1.

-->[g2,path,M2]=Routing(k,i,j,g4)

M2 =

0. 5. 0. 3. 4.

0. 0. 3. 3. 4.

0. 0. 0. 2. 1.

0. 0. 2. 0. 4.

0. 0. 0. 0. 0.

path =

2. 3.

g2 =

graph name : Graph

version : 5.0.1

oriented : yes

number of nodes : 5

number of edges : 10

-->path

path =

2. 3. **🡪 Đường dẫn từ điền đích tới nguồn**

-->show\_graph(g2)

ans =

2.

Chương 3. Kết Luận

* Dijkstra là một trong những thuật toán cho phép tìm đường đi ngắn nhất giữa các node trong đồ thị có hướng không có cạnh mang trọng số âm
* Sau khi tìm hiểu và nghiên cứu đề tài này giúp cho nhóm chúng em có thêm nhiều kinh nghiêm trong học tập.

+ Hiểu biết sâu rộng hơn về thuật toán Dijkstra

+ Biết cách trình bày vấn đề theo cách hiểu của mỗi thành viên

+ Tự tìm tòi kiến thức và tổng hợp lại, giải quyết bất đồng giữa các thành viên trong quá trình tìm hiểu đề tài, từ đó xây dựng hoàn chỉnh đè tài

+ Biết cách sử dụng phần mềm Scilab trong mô phỏng thuật toán

+ Giúp cho các thành viên trở lên thân thiết, gắn bó nâng cao khả năng làm việc nhóm, hỗ trợ nhau trong học tập

**Tài liệu tham khảo**

* Scilab\_beginners-vietnamese <http://www.scilab.org/resources/documentation/tutorials>
* Link giúp đỡ sử dụng các hàm trong Scilab

<https://help.scilab.org/docs/5.5.2/en_US/index.html>

* Một số cuốn sách hướng dẫn

http://www.scilab.org/resources/documentation/books