

**Sarvajanik College of Engineering & Technology**

A Report on AAD OEP:-

**Hierarchical Routing**

Under subject of

# Algorithm Analysis and Design

# (2150703)

B.E.III, Semester – V

Computer Engineering (Shift-1)

Submitted By:

Group: 4

|  |  |  |
| --- | --- | --- |
| **Sr.** | **Name of Student** | **Enrolment No.** |
| 1. | Francy Dudhwala | 140420107014 |
| 2 | Yazad Dumasia | 140420107015 |
| 3. | Karan Gajjar | 140420107016 |
| 4. | Vatsal Gotawala | 140420107018 |

Submitted To:

Prof. Rachna Oza

Prof. Urvashi m

Academic Year

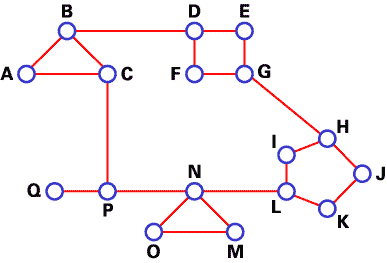
(2016-2017)

**Aim:-**

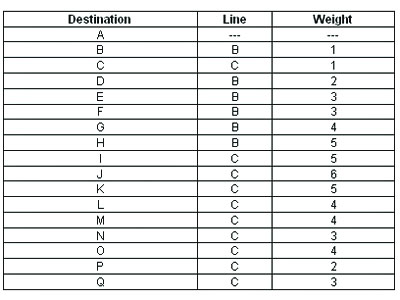
To create routing table of a network for all the nodes contained by the network using Hierarchical Routing.

**Description:-**

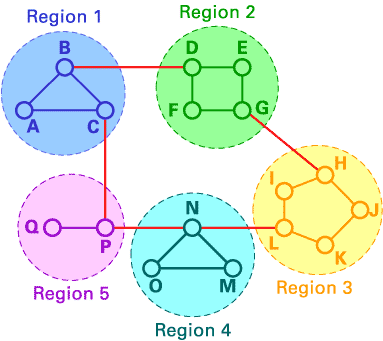
Consider this arrangement of network:



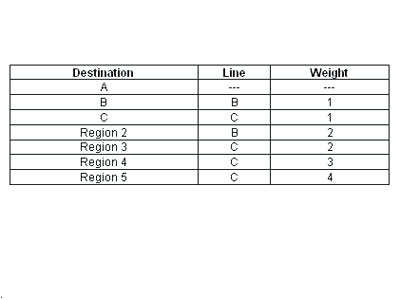
We use DV algorithms to find best routes between nodes. In the situation depicted below, every node of the network has to save a routing table with 17 records. Here is a typical graph and routing table for A:



In hierarchical routing, routers are classified in groups known as **regions**. Each router has only the information about the routers in its own region and has no information about routers in other regions. So routers just save one record in their table for every other region. In this example, we have classified our network into five regions (see below).



If A wants to send packets to any router in region 2 (D, E, F or G), it sends them to B, and so on. As you can see, in this type of routing, the tables can be summarized, so network efficiency improves. The above example shows two-level hierarchical routing. We can also use three- or four-level hierarchical routing.



In three-level hierarchical routing, the network is classified into a number of **clusters**. Each cluster is made up of a number of regions, and each region contains a number or routers. Hierarchical routing is widely used in Internet routing and makes use of several routing protocols.

**Code:-**

#include<iostream>

#include<sys/time.h>

const int size=15;

using namespace std;

struct node

{

char name;

unsigned dist[size];

unsigned from[size];

unsigned region;

}rt[size];

/\*

void quick(int l,int u,int \*temp,int \*s)

{

int p,i,j,t;

if(l<u)

{

p=temp[l];

i=l;

j=u;

while(i<j)

{

while(temp[i]<=p && i<j )

i++;

while(temp[j]>p && i<=j )

j--;

if(i<=j)

{

t=temp[i];

temp[i]=temp[j];

temp[j]=t;

t=s[i];

s[i]=s[j];

s[j]=t;

}

}

t=temp[j];

temp[j]=temp[l];

temp[l]=t;

t=s[j];

s[j]=s[l];

s[l]=t;

quick(l,j-1,temp,s);

quick(j+1,u,temp,s);

}

}

\*/

void sort(int a[], int s[], int n)

{

int i, j, temp[size], t;

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

temp[i]=a[i];

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

for(j=i+1; j<n; j++)

if(temp[i]>temp[j])

{

t=temp[i];

temp[i]=temp[j];

temp[j]=t;

t=s[i];

s[i]=s[j];

s[j]=t;

}

}

int main()

{

struct timeval tv1,tv2;

unsigned least=65535;

int i, j, k, l, r[size], n=12, conn[size], temp[size], regi=0, sorted[size], flag=0, count;

int costmat[12][12]={{0,1,1,65535,65535,65535,65535,65535,65535,65535,65535,65535},

{1,0,1,1,65535,65535,65535,65535,65535,65535,65535,65535},

{1,1,0,65535,65535,65535,65535,65535,65535,65535,65535,1},

{65535,1,65535,0,1,1,65535,65535,65535,65535,65535,65535},

{65535,65535,65535,1,0,65535,1,65535,65535,65535,65535,65535},

{65535,65535,65535,1,65535,0,1,65535,65535,65535,65535,65535},

{65535,65535,65535,65535,1,1,0,1,65535,65535,65535,65535},

{65535,65535,65535,65535,65535,65535,1,0,1,1,65535,65535},

{65535,65535,65535,65535,65535,65535,65535,1,0,65535,65535,1},

{65535,65535,65535,65535,65535,65535,65535,1,65535,0,1,65535},

{65535,65535,65535,65535,65535,65535,65535,65535,65535,1,0,1},

{65535,65535,1,65535,65535,65535,65535,65535,1,65535,1,0},

};

char x='A';

/\*cout<<"Enter the number of nodes: ";

cin>>n;

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

{

cout<<"Assign a character to node "<<i+1<<" :\t";

cin>>rt[i].name;

rt[i].dist[i]=0;

rt[i].region=0;

}

cout<<"\nEnter the cost matrix :\n";\*/

gettimeofday(&tv1,NULL);

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

{

rt[i].name=x++;

rt[i].dist[i]=0;

rt[i].region=0;

}

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

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

{

rt[i].dist[j]=costmat[i][j]; //Initialise the distance equal to cost matrix

rt[i].from[j]=j;

}

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

for(conn[i]=0, j=0; j<n; j++)

if(rt[i].dist[j]==1)

conn[i]++;

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

{

temp[i]=conn[i];

}

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

sorted[i]=i;

//quick(0,n,temp,sorted);

sort(conn,sorted,n);

for(i=0; i<n ;i++) //Assigning region to the nodes

{

if(rt[sorted[i]].region==0)

{

for(flag=0, j=0; j<n; j++)

{

if(rt[sorted[i]].dist[j]==1 && rt[j].region!=0)

{

flag=1;

break;

}

}

if(flag==1)

rt[sorted[i]].region=rt[j].region;

else

{

rt[sorted[i]].region=++regi;

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

if(rt[sorted[i]].dist[j]==1)

rt[j].region=regi;

}

}

}

do

{

count=0;

for(i=0;i<n;i++) //We choose arbitary vertex k and we calculate the direct distance from

for(j=0;j<n;j++) //the node i to k using the cost matrix and add the distance from k to node j

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

if(rt[i].dist[j]>costmat[i][k]+rt[k].dist[j])

{ //We calculate the minimum distance

rt[i].dist[j]=rt[i].dist[k]+rt[k].dist[j];

rt[i].from[j]=k;

count++;

}

}while(count!=0);

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

{

cout<<"\n\n For node "<<rt[i].name<<endl;

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

{

if( rt[i].region==rt[j].region)

cout<<"\t\nTo node "<<rt[j].name<<" via "<<rt[rt[i].from[j]].name<<" Distance is "<<rt[i].dist[j];

}

for(l=1; l<=regi; l++)

{

if(rt[i].region!=l)

{

for(least=65535, j=0; j<n; j++)

{

if(rt[j].region==l)

{

if(rt[i].dist[j]<least)

{

least=rt[i].dist[j];

r[l]=rt[i].from[j];

}

}

}

cout<<"\t\nTo region "<<l<<" via "<<rt[r[l]].name<<" Distance is "<<least;

}

}

}

gettimeofday(&tv2,NULL);

cout<<endl<<tv2.tv\_usec<<endl<<tv1.tv\_usec;

cout<<"\nTotal time taken : "<<(double)(tv2.tv\_usec-tv1.tv\_usec);

return 0;

}

**Output:-**

For node A

To node A via A Distance is 0

To node B via B Distance is 1

To node C via C Distance is 1

To region 2 via B Distance is 2

To region 3 via C Distance is 2

For node B

To node A via A Distance is 1

To node B via B Distance is 0

To node C via C Distance is 1

To region 2 via D Distance is 1

To region 3 via C Distance is 2

For node C

To node A via A Distance is 1

To node B via B Distance is 1

To node C via C Distance is 0

To region 2 via B Distance is 2

To region 3 via L Distance is 1

For node D

To node D via D Distance is 0

To node E via E Distance is 1

To node F via F Distance is 1

To node G via E Distance is 2

To region 1 via B Distance is 1

To region 3 via E Distance is 3

For node E

To node D via D Distance is 1

To node E via E Distance is 0

To node F via D Distance is 2

To node G via G Distance is 1

To region 1 via D Distance is 2

To region 3 via G Distance is 2

For node F

To node D via D Distance is 1

To node E via D Distance is 2

To node F via F Distance is 0

To node G via G Distance is 1

To region 1 via D Distance is 2

To region 3 via G Distance is 2

For node G

To node D via E Distance is 2

To node E via E Distance is 1

To node F via F Distance is 1

To node G via G Distance is 0

To region 1 via E Distance is 3

To region 3 via H Distance is 1

For node H

To node H via H Distance is 0

To node I via I Distance is 1

To node J via J Distance is 1

To node K via J Distance is 2

To node L via I Distance is 2

To region 1 via I Distance is 3

To region 2 via G Distance is 1

For node I

To node H via H Distance is 1

To node I via I Distance is 0

To node J via H Distance is 2

To node K via L Distance is 2

To node L via L Distance is 1

To region 1 via L Distance is 2

To region 2 via H Distance is 2

For node J

To node H via H Distance is 1

To node I via H Distance is 2

To node J via J Distance is 0

To node K via K Distance is 1

To node L via K Distance is 2

To region 1 via K Distance is 3

To region 2 via H Distance is 2

For node K

To node H via J Distance is 2

To node I via L Distance is 2

To node J via J Distance is 1

To node K via K Distance is 0

To node L via L Distance is 1

To region 1 via L Distance is 2

To region 2 via J Distance is 3

For node L

To node H via I Distance is 2

To node I via I Distance is 1

To node J via K Distance is 2

To node K via K Distance is 1

To node L via L Distance is 0

To region 1 via C Distance is 1

To region 2 via C Distance is 3

933312

795290

Total time taken : 138022 (in microseconds)

--------------------------------

Process exited after 0.1777 seconds with return value 0

Press any key to continue . . .

Note:-

Total time taken by the Dijkstra’s Algorithm to create the routing table for the same network is 260000 microsecs (on average).

**Conclusion:-**

Thus, time taken by Hierarchical routing is nearly “half” than the time taken by Dijkstra’s algorithm for some particular network having 10 to 15 routers (as for above case having 12 routers). Hierarchical routing becomes more and more favourable and efficient as the network size (i.e. the number of router) increases.