Volume 15 Issue 05 Pages 6135-6145 (2024) ISSN: 0975-0290

Routing Methods For Data Transmission In Large Networks: A Review

Sujit K. Bose
S.N. Bose National Center for Basic Sciences, Salt Lake City, Kolkata 700106, India.
E-mail: sujitkbose1@gmail.com.
https://orcid.org/0000-0003-3867-0969

ABSTRACT. Data networks are increasingly becoming very large, spanning even continents, in tune with the development in hardware technologies capable of long haul transmission by carrier radio/microwave and photonic devices. Such special devices are however driven by special softwares based on algorithms based on specially designed methods. The literature on algorithms is however voluminous, but the principal methods of transmission of data in practice are quite limited. A review of these methods is presented in this article and the corresponding algorithms presented in the form of cogent pseudo codes for easy coding in a suitable language. This survey also includes some algorithms recently developed by this author in respect of networks on a geographical scale. Special modifications may be incorporated in the currently used algorithms as suggested here, for meeting specific purposes without unnecessary latency and error .

Keywords - Metro/megacity networks, network methods and algorithms,, routing, search, wide area networks.

Date of Submission: 1 January, 2024 Date of Acceptance: 4 February, 2024

I. INTRODUCTION

A network in abstract mathematical terms, consists of *nodes* (capable of performing certain tasks in a physical scenario), lablelled by the elements 1, 2, $3, \cdots, n$, connected together by *links* or *edges* defined by the set of ordered pairs $\{i, j\}$. Associated with each link generally speaking, is some kind of flow starting from a particular node called the *source node* and end at another called the *destination node*. These are usually taken as nodes 1 and n respectively. A *path* from source to destination may in general, pass through several intermediatary nodes of the network called *routers* in the context of of daa transmission. A path that connects a source node to itself is called a *loop* and is undesirable in a network taxonomy. Associated with each link is some kind of flow between the two connecting nodes, that are are assumed to be capable of *bidirectional* in nature, possessing some other attributes such as the geographical distance, bandwidth capacity, queueing constraints etc. For large networks, the geographical distance $d(i, j)$ of a link $\{i, j\}$ is especially of some concern. For theoretical purpose all the nodes can be assumed to be directly connected (or fully connected network topology) having a distance metric (or weight, and sometimes as cost)$d(i, j)$ with the stipulation that if i and j are not connected at all then $d(i, j)$ is set to infinity. In this way, all of the prevalent topologies, such as *local area mesh topology, bus topology, star topology* and their modifications are covered by a sigle definition. It may be noted that the number of links in a fully connected network with $d(i, j) \neq \infty$ is $C_2^n = n(n-1)/2$ which is $O(n^2)$ and is prohibitively large for large networks. As a result methods of treatment of a network must involve the least possible number of nodes as possible. Most of the methods developed are termed *greedy* for exact optimality is not established.

In practice networks have different size, which are broadly divided in to three main categories of medium sizes: Local Area Network (LAN), Metropolitan/Megacity Area Network, and Wide Area Network (WAN). These three nomenclatures are of common use, and suggestive of the installation of appropriate hardware and driving software technologies. While LAN is local and small, MAN is of medium size. On the other hand, WAN can be of huge size as in the case of the public internet, which is a completely connected network of computers and digital devices distributed worldwide, that includes many private and public networks, through common nodes called *gateways*. The physical links of a network may be *wired* or *wireless*. A wired link is, in general, a fiber-optic cable carrying light wave signal generated by a *light emitting diode* (LED) or a*laser diode*. In the former case, the carrier light wave is incoherent having a band of frequencies, while in the latter case it is coherent and highly directional in contrast and monochromatic. Data is transmitted by modulation of the carrier light wave signal. In the case of wireless as the name signifies, the carrier wave is a radio wave in thw frequency range of 500 kHz - 1000 MHz or higher frequency microwaves in the 1 GHz -300 GHz range. In the terrestrial context the overground links are obviously of the wireless type, while the links in some cases, may be underground as well with fiber-optic connectivity. The latter option is preferred for high fidelity of transmitted data because of almost no interference from any external source. In terrestrial networking, the nodes are obviously fixed to the ground and are called *Base Stations*. Of late however, with the launching of *Low Earth Orbit* (LEO) satellites, total wireless communication networks are being developed. To an extent wireless communication has gained importance because of mobile telephonic hand sets and computers of various sizes. In view of this *Mobile Ad-hoc Network* (MANET) using fixed cellular ground stations are already in use. Details of the above stated technologies are indeed very intricate, known only to the professionals of the field. The actual networks briefly sketched above is based on various websites of the internet where additional information is available.

The present article has a the limited objective of presenting the basics of the methods and the algorithms employed in the software for driving actual networks. A medium sized network of base stations linked together in some taxanomy, as also very large networks on global scale are envisaged to look for an optimal or otherwise at least a *greedy path* requiring shortest length for connecting a source node to a destination. In practice there may be several destination nodes for transmitting the same data set, but the focus here is on only one of them. The destination node may be of changing position due to mobility of the device holder, and so the ensuing discussion is divided in to two parts. Firstly, for locating the destination base station followed by optimal/greedy path selection called *routing* for connecting the source with the chosen destination. Sections II and III deal with these two aspects respectively. The algorithms for the methods developed are then presented in the form of pseudo codes for easy implementation in to programs in a suitable language, with some modification necessary for practical implementation.

II. SEARCH IN A PLANE

(a) Tree search

Given a network of n nodes in a plane, the search graph from the source node 1 to to destination node n must not contain any loop and so the search graph must form a (loopless) tree graph whose root node is 1 connected to its nearest neighbour *child* nodes that are in turn connected to their own child nodes and so on till the destination node is reached. Indeed it will shown in section 4, that any network can be spanned by a tree graph for connectivity. The search procedure can take place in two ways (Cormen et. al.[1]) : (1) along succeeding child nodes, or (2) by following lineage of the succeeding generations of child nodes such that no node is visited more than once. The first method is called the *Breadth First Search* (BFS) and the second one the *Depth First Search* (DFS). These methods are in fact employed in searching a destination by *Flood-*

ing the network in which the source computer sends the search address to all of its neighbours. The neighbours then sends it to to all of their neighbours and so on. The methods are also used in *Web Crawl* of the internet. The algorithms for the two methods are given in the following. The important point to note is that in BFS a *queue* of visited nodes is created and is easy to develop, while DFS creates a *stack* of such nodes, following lineages of the generations. A list of elements is called a queue when it follows the *First In First Out* (FIFO) principle, while a stack is a list following the *Last In First Out* (LIFO) principle.

Algorithm 1. Breadth First Search (BFS).

1. Input: n \\\\ n=number of nodes.

2. Output: n_T \\\\ n_T = target node of the search.

3. (Initialise)
for $\leftarrow$ 1 to n
queue[i] $\leftarrow$ 0 \\\\ empty child queue.
for $j \leftarrow 1$ to n
print i, j; $l[i,\, j] \leftarrow 0$ \\\\ flag for linked nodes.
if($i \neq j$ are directly linked) $l[i,\, j] \leftarrow 1$
end for; end for

4. $i \leftarrow 1$; kount $\leftarrow$ 0

5. for $j \leftarrow i+1$ to n
if($l[i,\, j] = 1$ & queue[j] $= 0$) then
kount $\leftarrow$ kount+1; q[kount]$\leftarrow j$
if(**target found**) then
$n_T \leftarrow j$
print n_T; STOP
end if
end for
end if

6. if($i = n_T$) then
print i; STOP
end if
$i \leftarrow i+1$; go to Step 5

7. end

Algorithm 2. Depth First Search (DFS).

1. Input: n \\\\ n = number of nodes.

2. Output: n_T \\\\ n_T = target node of the search.

3. (Initialise)
for $i \leftarrow 1$ to n
list[i]$\leftarrow$ 0 \\\\ DFS linked stack list.

for $j \leftarrow 1$ to n
print i, j; $l[i, j] \leftarrow 0$ \\ flag for linked nodes.
if($i \neq j$ are directly linked) $l[i, j] \leftarrow 1$
end for
end for

4. $i \leftarrow 1$; list[1]$\leftarrow$ 1; kount$\leftarrow$ 0

5. for $j \leftarrow$ list[i] to n
if($l([i, j] = 1)$ then
kount$\leftarrow$kount+1; list[kount]$\leftarrow j$
if(**target found**) then
$n_T \leftarrow j$
print n_T; STOP
end if

6. $i \leftarrow i + 1$
if(list[i]=0) $i \leftarrow i + kount$
go to Step 5
end if

7. end

(b) Directional search

In the case of a large network like MAN, a cellular search was suggested by this author [2]. In that method, since the base stations are geographically located over a large, nearly plane area with known latitude and longitude from GPS data, the search is carried out over circular sectors around the source station bounded by rays, say 1^o apart from the eastward direction. If a node A_k with latitude, longitude (ϕ_k, λ_k) under search subtends an angle β_k at the source node 1 with latitude, longitude (ϕ_1, λ_1) measured from the eastward direction, as in figure 1(b) and if ψ_k is the deviation of A_k from the north, i.e. $\psi_k = \angle A_k A_1 N$, then in the plane

$$\beta_k = \frac{\pi}{2} - \psi_k \tag{1}$$

Now, referring to the spherical triangle NA_1A_k in that figure

$$\sin\psi_k = \frac{\cos\phi_k\,\sin(\lambda_k - \lambda_1)}{\sin r} \tag{2}$$

where

$$\cos r = \sin\phi_1 \sin\phi_k + \cos\phi_1 \cos\phi_k \cos\lambda_k - \lambda_1) \tag{3}$$

(Abramowitz and Stegun [3]). The above formulation leads to the next algorithm in which if the target is found in a cell, then the method is stopped. Otherwise the iterations are carried over the cells for maximum number of time i_{max}.

Algorithm 3. Cellular search in a plane.

1. Input. ϕ_1, λ_1 \\ Latitude, Longitude of source node.

2. Out[ut. ϕ_n, λ_n Latitude, longitude of destination node.

3. $i \leftarrow 1$; $j \leftarrow 1$

4. $\beta_0 \leftarrow 0$; $\beta_1 \leftarrow \pi/180$ \\ Bounds of sectors of 1^o.

5. $k \leftarrow 1$

6. $r \leftarrow \arccos[\sin\phi_1 \sin\phi[k] + \cos\phi_1 \cos\phi[k] \cos(\lambda[k] - \lambda_1)]$

$\psi \leftarrow \arcsin\left[\frac{\cos\phi[k]\sin(\lambda[k]-\lambda_1)}{\sin r}\right]$ \\\\ $\phi[k]$, $\lambda[k]$ $\leftarrow$Latitude, Longitude of a station k in the sector bounded by $(i,\, j)$.

$\beta[k] \leftarrow \pi/2-\psi$

7. do while$(\beta_0 \leq \beta[k]-\beta[j] \leq \beta_1)$
{if(**target found** then
$\phi_n \leftarrow \phi[k]$; $\lambda_n \leftarrow \lambda[k]$
print ϕ_n, λ_n
end if}
end do

8. if$(k < k_{max})$ then
$k \leftarrow k+1$; go to Step 6
end if

10. $i \leftarrow i+1$
if$(i < i_{max})$ then
go to Step 4
else
STOP
end if

11. end

(c) Search on a globe

If the network is geographically spread over the globe as in WAN, connecting nodes over possibly different continents, the sphericity of the earth come in to play in the network layout. This aspect was examined by this author [4], by parsimonious method of selective flooding over cells on the earth's surface having the source node A_1 as the pole. In this case, the spherical surface is first divided in to "latitudinal" strips of 1^o, the angle subtended at the center of the sphere. The search is cellular starting from the innermost strip outwards, until the target destination is detected. For further restricting the search area, the strips are bounded on the two sides by "longitudinal" great circles of vertical angle $\pi/4$. This means that the polar region is divided in to equal sectors.

With A_1 as pole, a node A_k having latitude - longitude $(\phi_k,\, \lambda_k)$ in the cell of search is supposed bounded by the inner and outer circles of "colatitude" $(\psi_0,\, \psi_1)$, then as in figure 1(b) it can be shown that

$$\phi_1+\psi_0 \leq \phi_k \leq \phi_1+\psi_1 \tag{4}$$

where $(\phi_1,\, \lambda_1)$ are the latitude-longitude of the source node A_1. For restricting the "longitudinal" boundaries of the cells, let the geographical meridian through A_1 be taken as the reference circle. If χ_1 be the vertical angle at A_1 subtended by the node A_k with the reference circle, and ψ_k its "colatitude", then referring to figure 1(b), the angle χ_k is given by the equation

$$\sin\chi_k = \frac{\cos\phi_k \sin(\lambda_k-\lambda_1)}{\sin\psi_k} \tag{5}$$

where

$$\cos\psi_k = \sin\phi_1\sin\phi_k + \cos\phi_1\cos\phi_k\cos(\lambda_k-\lambda_1) \tag{6}$$

The angle χ_k must then lie in the octant of search. This formulation leads to the following:

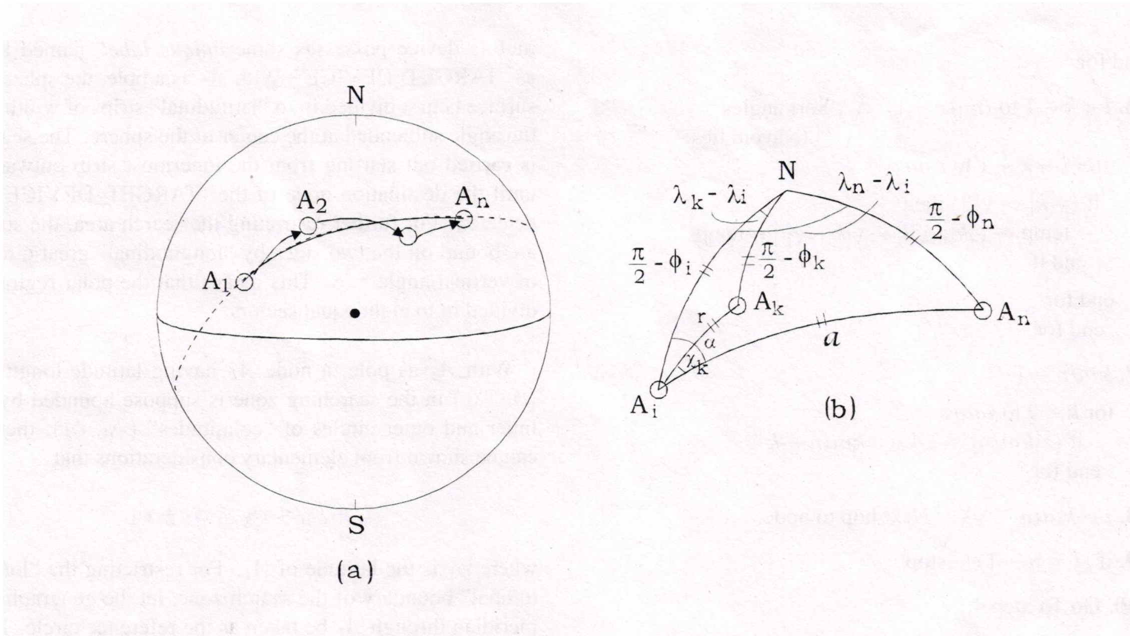

Figure 1: (a) Path $A_1 A_2 \cdots A_n$ on spherical earth. (b) Configuration of nodes A_i, A_k, andA_n.

Algorithm 4. Cellular search on the globe.

1. Input: ϕ_1, λ_1 \\ Latitude-Longitude of source node.

2. Output: ϕ_n, λ_n \\ Latitude-Longitude of destination node.

3. $\psi_0 \leftarrow 0$

4. for $i \leftarrow 1$ to 180
$\psi_1 \leftarrow i \times \pi/180$
for $j \leftarrow 1$ to 8
$\chi_0 \leftarrow 0$; $\chi_1 \leftarrow \chi_0 + \pi/4$
for $k \leftarrow 1$ to k_{max} \\ $k_{max} \leftarrow$ max. number of nodes in the cell of search.
$\phi[k]$, $\lambda[k] \leftarrow$ Latitude-Longitude of the stations of k^{th} cell of search.
$\psi k \leftarrow \arccos\{\sin\phi_1 \sin\phi[k] + \cos\phi_1 \cos\phi[k] \cos(\lambda[k] - \lambda_1)\}$
$\chi k \leftarrow \arcsin\{\cos(\phi[k]) \times \sin(\lambda[k] - \lambda_1)\}/\sin\psi k$
do while $(\psi_0 \le \phi[k] - \phi_1 \le \psi_1$ & $\chi_0 < \chi < \chi_1)$
if(**target found**) then
$\phi_n \leftarrow \phi[k]$, $\lambda_n \leftarrow \lambda[k]$; STOP
end if
end for
$\chi_0 \leftarrow \chi_1$; $\chi_1 \leftarrow \chi_0 + \pi/4$

5. end

Having located the destination node, the next objective that remains is transmission of data from source to destination in efficient way, reducing the load on the network.

III. ROUTING PATHS

(a) On a plane

Data networks essentially consist of router ground stations as nodes linked by radio/microwave transmitters and/or by fiber-optic cables layed along the ground/sea bed. The essential function of the network is to deliver packets of data from a source node to a destination of the network, preferably along a fixed path for economy of task as well as to minimise congestion in the network as a very large number of sources are usually active simultaneously in the network.

The commonly used routing protocols at present are of three types: *distance-vector routing, link-state routing* and *path-vector routing* developed by CISCO. In distance-vector routing, each router multicasts its knowledge to its neighbours covering a large area of the whole network. The procedure is dynamic and kept in the form of an updated table in each router. A table contains an assigned cost of each link of the router and the least-cost path between any two nodes calculated by the optimal Bellman-Ford algorithm (Bellman [5], Ford [6], Corman et. al. [1]). This information enables dispatch of data packets from source to destination along the least-cost path. RIP (Routing Information Protocol) and IGRP (Interior Gateway Routing Protocol) are protocols of this type (CISCO systems [7] and [8] embedded in TCP/IP environment. In the link-state procedure, developed to mitigate some of the shortcomings, initially a router defines the cost of each link and broadcast its information to other nodes of the network as well. Data forwarding from source to destination is carried out by the least-cost path determined by Dijkstra's algorithm (Dijkstra [9], Bose [10], Corman et. al. [1]). Any change occurring in the routing table in the path is notified to other nodes as an update. OSPF (Open Shortest Path First) is an elaborate algorithm that carries out this basic feature of the routing (CISCO systems [11]). The two systems path-vector and link-state are essentially suitable for intra-domain autonomous networks, in contrast to inter-domain routing. For the latter type BGP (Border Gateway Protocol, CISCO systems [12]) is employed, which also uses path-vector routing. Here in this section the Bellman-Ford and Dijkstra's algorithms are presented in the following.

In the Bellman-Ford method, the network is assumed to be completely connected with a distance metric $d(i,\, j)$ between two nodes i and j. If i and j are not directly linked, $d(i,\, j)$ is assumed to be infinite, so that $0 < d(i,\, j) \leq \infty$ where the matrix $[d(i,\, j]$ need not be symmetric. For connecting source node 1 to destination node n, the number of possible paths increases rapidly with increasing value of n and so a direct enumeration to find the shortest path becomes prohibitive. For that purpose a *dynamic programming* formulation (Corman et.al. [1]) is employed. Accordingly, let

$$f_i = \text{the total distance from node } i \text{ to node } n \; (i = 1,\, 2,\, 3, \cdots,\, n-1) \text{ using an optimal policy,} \tag{7}$$

with $f_n = 0$. This means that f_i satisfy the nonlinear system of equations

$$\begin{aligned} f_i &= \min_{j \neq i} \{d(i,\, j)+f_i\}, \quad i = 1,\, 2,\, 3, \cdots,\, n-1 \\ f_n &= 0 \end{aligned} \tag{8}$$

The system of Eqs. (8) is solved by applying the method of successive approximations as in Bellman [5], starting with some initial approximation $\{f_i^{(0)}\}$ and use the iterations

$$\begin{aligned} f_i^{(k+1)} &= \min_{j \neq i} \{d(i,\, j)+f_i\},\; i = 1,\, 2,\, 3, \cdots,\, n-1 \\ f_n^{(k+1)} &= 0 \end{aligned} \tag{9}$$

for $k = 0,\, 1,\, 2,\, 3, \cdots$. For the initial approximation to start the iteration, Bellman [5] has proposed the form

$$\begin{aligned} f_i^{(0)} &= \min_{j \neq i} d(i,\, j), \quad i = 1,\, 2,\, 3, \cdots,\, n-1 \\ f_n^{(0)} &= 0 \end{aligned} \tag{10}$$

which can be shown to be monotone convergent. For although $f_i^{(k+1)} > f_i^k$, $f_i^0 \leq f_i$ from Eqs. (10) and (8). So proceeding inductively,

$$f_i^{(k+1)} = \min_{j \neq i} \{d(i,\, j)+f_i^{(k)}\} \leq \min_{j \neq i} \{d(i,\, j)+f_i\} = f_i \tag{11}$$

Thus the sequence $f_i^{(k)}$ converges to f_i as $k \to \infty$. The foregoing scheme leads to the algorithm given in the following:

Algorithm 5. (Bellman-Ford) Finding shortest path.

1. Input: n, $d[\,]$, $f[\,]$ \\\\ $n \leftarrow$ number of nodes.
\\\\ $d[i, j] \leftarrow$ direct distance between nodes i, $j = 1, 2, 3, \cdots, n$.
\\\\ $f[i] \leftarrow$ shortest distance between nodes i and destination n.

2. Output: i, $f[i]$ $(i = 1, 2, 3, \cdots, n)$.

3. $f[n] \leftarrow 0$; huge $\leftarrow 10^6$ (one million)
for $i \leftarrow 1$ to $n-1$
for $j \leftarrow 1$ to $n-1$
$d[i, j] \leftarrow$ huge; if$(i = j)$ $d[i, j] \leftarrow 0$
print i, j; pause
if$(i \neq j$ & i directly linked to $j)$ read $d[i, j]$
end for; end for

4. $i \leftarrow 1$

5. $d_{min} \leftarrow$ huge
for $j \leftarrow 1$ to $n-1$
if$(d[i, j] < d_{min})$ $d_{min} \leftarrow d[i, j]$
end for
$f[i] \leftarrow d_{min}$; small $\leftarrow$ huge

6. for $j \leftarrow 1$ to n
if$(j \neq i)$ $f[j] \leftarrow f[i] + d[i, j]$
if$(f[j] <$ small$)$ then
small $\leftarrow f[j]$; $i_{next} \leftarrow j$
end if
end for

7. $i \leftarrow i_{next}$; $f[i] \leftarrow f[i_{next}]$
print i, $f[i]$; if$(i = n-1)$ STOP
go to Step 5

8. end

In the Dijkstra's method the iterations are carried out in a slightly different manner (Corman et. al. [1], Bose [10]). Here the shortest path is searched starting from the source node 1, hoping through intermediate nodes connected by shortest links between successive nodes, ignoring nodes that were already visited. The method is easily understandable by using combinatoric labelling method as in Bose [10]. Thus, let

g_i = the total shortest distance path from source node 1 to node i. (12)
with $g_1 = 0$. This means that at the node i the next hop to j is given by

$$g_j = \min_{j \neq \mathrm{nv}[i]} \{g(i) + d(i, j)\} \tag{13}$$

where nv$[i]$ is a list of nodes already visited to reach i. Following Eq. (13) the algorithm for the method becomes:

Algorithm 6. (Dijkstra). Finding shortest path.

1. Input: n, nv[ ] \\\\ $n \leftarrow$ number of nodes.
\\\\ nv$\leftarrow$ distance between two directly linked nodes. $i,\ j = 1,\ 2,\ 3, \cdots,\ n$.

2. Output: i, $g[i]$ \\\\ $i \leftarrow$ sequential label of nodes from source 1 to destination n.
\\\\ $g[i] \leftarrow$ shortest distance of node i from source node 1.

3. huge $\leftarrow 10^6$ \\\\ One million.

4. for $i \leftarrow 1$ to n
$d[i,\ j] \leftarrow$ huge
print i, j; pause
if(i directly linked to j) read $d[i,\ j]$
end for; end for

5. $g(1) \leftarrow 0$; nv[1]$\leftarrow 0$
for $i \leftarrow 2$ to n
$g[i] \leftarrow$ huge; nv[i]$\leftarrow$0
end for

6. $i \leftarrow 2$; kount $\leftarrow 1$

7. for $j \leftarrow 2$ to n
for $k \leftarrow 1$ to kount
if($i \neq$ nv[k] & $d[i,\ j]$ <huge) $g[j] \leftarrow g[i] + d[i,\ j]$
small $\leftarrow g[j]$
end for; end for

8. for ȷ $\leftarrow 2$ to n
if($g[j]$ = small) $i_{next} \leftarrow j$
end for
$i \leftarrow i_{next}$; $g[i] \leftarrow g[i_{next}]$; kount $\leftarrow$ kount+1; nv[kount] $\leftarrow i$
print i, $g[i]$; if($i = n$) STOP
go to Step 7

9. end

The above two methods though relate to planar graphs, are also applicable to an extent when geographical sphericity of the network design is also to be taken into account. In that case, if $(\phi_i,\ \lambda_i)$ and $(\phi_j,\ \lambda_j)$ are the latitude-longitude of the nodes i and j, then the geographical distance $d(i,\ j)$ between the nodes is to be replaced by the great circular arc of length

$$d(i,\ j) \leftarrow \arccos[\sin\phi_i \sin\phi_j + \cos\phi_i \cos\phi_j \cos(\lambda_j - \lambda_i)] \tag{14}$$

(b) On a sphere

In a very large network of long data haul from a geographical point of view, the sphericity of the globe comes in to play for selecting a path from source to destination as shown in figure 1(a). Accordingly a greedy algorithm was proposed in [4] to account for such a topology of the network. According to the nomenclature introduced earlier, A_1 and A_n represent the source and destination nodes respectively having latitude-longitude $(\phi_1,\ \lambda_1)$ and $(\phi_n,\ \lambda_n)$. Let $(\phi_i,\ \lambda_i)$, $(i = 2,\ 3, \cdots,\ n-1)$ be a node on the desired path having nearest neighbour base stations $A_k\,(\phi_k,\,\lambda_k)$ (Figure 1(b)), then for a greedy path $\angle A_kA_iA_n$ must be least possible with a favourable waiting time W_k. Now, in the spherical triangle A_iA_kN, where N is the north pole of the earth, A_kNA_i = difference of longitude between A_k and $A_i = \lambda_k - \lambda_i$, and arc A_iN = colatitude of $A_i = \pi/2 - \phi_i$, arc A_kN =colatitude of $A_k = \pi/2 - \phi_k$. Let $r :=$ arcA_iA_k, $a :=$ arcA_iA_n and $\alpha := \angle NA_iA_n$, then from the spherical triangle NA_iA_k (Abramowitz and Stegun [3], p.79)

$$\sin(\alpha - \chi_k) = \frac{\cos\phi_k \sin(\lambda_k - \lambda_i)}{\sin r} \tag{15}$$

where

$$\cos r = \sin\phi_i \sin\phi_k + \cos\phi_i \cos\phi_k \cos(\lambda_k - \lambda_i) \tag{16}$$

Similarly, from the spherical triangle NA_iA_n

$$\sin\alpha = \frac{\cos\phi_n \sin(\lambda_n - \lambda_i)}{\sin a} \tag{17}$$

where

$$\cos a = \sin\phi_i \sin\phi_n + \cos_i \cos\phi_n \cos(\lambda_n - \lambda_i) \tag{18}$$

Thus, Eqs. (15) - (18) determine the angle χ_k which is the deviation from the geodesic path between A_i and A_n. For a greedy path, the objective function

$$z = |\chi_k| \tag{19}$$

must be a minimum. The objective function can be modified so as to take in to account the waiting time for the path as in Bose [4]. This method leads to the following algorithm:

Algorithm 7. Data transmission on the globe.

1. Input: $\phi_1,\,\lambda_1,\quad \phi_n,\,\lambda_n$ \\ Latitude-Longitude of source and destination.

2. Output: $\phi[\;],\,\lambda[\;]$ \\ Latitude, Longitude of intermediate nodes on the path,
\\ $(i = 2,\,3,\cdots,\,n-1)$.

3. $i \leftarrow 1$
$\phi[1] \leftarrow \phi1$; $\lambda[1] \leftarrow \lambda1$

4. $i_{max} \leftarrow$ Number of stations in the neighbourhood of node i.
$a \leftarrow \arccos\{\sin\phi[i]\sin\phi_n + \cos\phi[i]\cos\phi_n\cos(\lambda_n - \lambda[i])\}$

5. for $k \leftarrow 1$ to i_{max}
$\phi[k],\,\lambda[k] \leftarrow$ (Latitude, Longitude of nearest neighbour stations of node i.
$r \leftarrow \arccos\{\sin\phi[i]\sin\phi[k] + \cos\phi[i]\cos\phi[k]\cos(\lambda[k] - \lambda[i])\}$
$\chi[k] \leftarrow \arcsin\{\cos\phi_n \sin(\lambda_n - \lambda[i])/\sin a$ $-\arcsin\{\cos\phi[k]\sin(\lambda[k] - \lambda[i])\}$
$z[k] \leftarrow |\chi[k]|$
end for

6. for $k \leftarrow 1$ to $i_{max} - 1$ \\ Sort angles $\chi[k]$ to avoid ties.
for $\leftarrow k+1$ to i_{max}
if($\chi[k] > \chi[l]$) then
temp $\leftarrow \chi[k]$; $\chi[l] \leftarrow \chi[k]$; $\chi[l] \leftarrow$ temp
end if
end for; end for

7. $k_{min} \leftarrow 1$
for $k \leftarrow 2$ to i_{max}
if($z[k_{min}] > z[k]$) $k_{min} \leftarrow k$

end for

8. $i \leftarrow k_{min}$ \\ Next hop to node.

9. if$(i = n - 1)$) STOP

10. go to Step 4

11. end

IV. CONCLUSION

A survey of the main methods of routing of data from a source to destination is presented here in the form of pseudo codes for easy implementation in a suitable programming language, beginning with a brief sketch of the networks in use, delineating potentially newer approaches for long haul over very large networks that have been proposed in recent years, that take in to account the sphericity of the earth. The problem of despatch is divided in to two parts: firstly searching the destination, particularly when it is mobile, and secondly sending the data by shortest possible path (routing) to avoid latency and congestion in the network. These two aspects are covered in sections 2 and 3 respectively. For planer graphs such as in MAN, the BFS and DFS methods of search are quite well known, so also are the Bellman-Ford and the Dijkstra methods for shortest path transmission. But because of vastness of networks in this age, a new perspective, that of geographical sphericity of the earth using spherical trigonometry, has been introduced lately. These are presented as subsections in the two sections. Finally, the problem of designing a network, having shortest total path legth is treated in section 4.

V. DECLARATIONS

Disclosures. There are no conflicts of interest involved in the reported research of this paper.

Data availability. No data were generated or analyzed in the presented research.

REFERENCES

[1] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, McGraw-Hill (2009).

[2] S.K. Bose, A routing algorithm of data in networks of metro and mega cities, Int. J. Innov. Res. in Tech. **8**, 185-187.

[3] M. Abramowitz, I.A. Stegun, Handbook of Mathematical Functions, Dover Publications, New York (1972).

[4] S.K. Bose, A routing algorithm in networks on the globe, Informatica, **45**, 273-278 (2021).

[5] R. bellman, On a routing problem, Quart. App. Math. **16**, 87-90 (1958).

[6] L.R. Ford, Network flow theory, RAND Corp. USA, Paper P-923 (1956).

[7] CISCO Systems, IOS IP Routing, RIP Configuration Guide (2011).

[8] CISCO Systems, An Introduction to IGRP (2005).

[9] E.W. Dijkstra, A note on two problems in connection with graphs, Numerische Mathematik, **1**, 269-271 (1959).

[10] S.K. Bose, Operations Research Methods, Narosa Publishing, New Delhi.

[11] CISCO Systems, IOS IP Routing, OSPF Command Reference (2011).

[12] CISCO Systems, IP Routing: BGP Configuration Guide (20019).