INTERNATIONAL JOURNAL OF NATURAL AND APPLIED SCIENCES (IJNAS) VOL. 4, NOS.1& 2 (2009); P. 125 – 134, 1 TABLE, 9 FIGS.

Basics of network routing

O. O. Elechi [1]

ABSTRACT

With the fact that the Internet is a network of networks, it means that packets (Protocol Data Unit of the network layer) have several possible paths to pass through when leaving its source to get to its destination. This was actually the idea of the United States Department of Defence when they developed the first packet switching network known as Arpanet. Their main aim then was that if a node was bombed down by enemies in war times, the administrators in offices and soldiers on war fronts should still be able to communicate because the node that was bombed does not lie on the only access route between the office administrators and soldiers on site. Today in the big Internet, several considerations have to be taken by the network router designer on the rules to be embedded on the router for it to use when routing decisions have to be made on arriving packets. Such considerations include, the least cost to the end router if it has complete knowledge of the topology of the network, the least cost to the nearest neighbour, the least cost to any of its immediate neighbours, manually configured congested links, number of hops and so on. With these considerations several routing algorithms have been designed of which we look at their basic principles.

INTRODUCTION

When networks first came into being, only computers of the same manufacture could communicate together. In the 1970s the International Organisation for Standardization (ISO) introduced the OSI (Open System Interconnection) model to break this barrier (Clements, 2000). This was meant to help vendors develop interoperable network devices. The model is made up of seven layers which provides a set of guidelines that application developers can use to create and implement applications that run on a network. The top three layers (Application, Presentation and Session layers) define how the applications in the end stations of the network will communicate with each other and with users while the bottom four layers (Transport, Network, Data Link and Physical layers) define how data is transmitted from one end station to another end station (Lammle, 2000).

The network layer of the OSI model is responsible for moving packets (data from transport layer known as segment encapsulated with network control information known as header) from the source node, all through the network to the destination node. It is the lowest layer in the OSI model that deals with end-to-end transmission as well involves every host and router in the network. This makes the network layer protocols among the most challenging in the network. The data link layer is just responsible for moving frames (network layer packets encapsulated with data link headers) from one end of a wire/link to the to the other end. In order for the network layer to transfer these packets effectively, the network layer must know or at least have an idea about the topology of the communication subnet so as to be able to choose appropriate paths through it (Tanenbaum, 1996).

Exactly how the network layer determines these paths and the issues involved in resolving these paths is the subject of discussion here. Before delving into the details of the theory and implementation of the path determination, it will be worthwhile to brush through various issues concerned with the network layer such as functions and types of service models rendered.

NETWORK LAYER FUNCTIONS

The network layer has three main and important functions (Kurose and Ross 2001)

- Path determination
- Switching
- Call setup

Path determination

The network layer must decide on the path by which an arriving packet should follow to get to its destination. The routing algorithm is a path of the network layer software that determines the path a packet should flow through to its destination.

Switching

When a packet arrives at one of the routers port, after the header has been examined, it must be placed on the appropriate port leading to its final destination. This transfer of a packet from its incoming port to the appropriate outgoing port is the switching function being considered here. It is usually achieved by switching via memory, switching via bus or by switching via an interconnection network (crossbar).

Manuscript received by the Editor June 15, 2007; revised manuscript accepted September. 11, 2008

[1]Department of Computer Science , Ebonyi State University, Abakaliki , Nigeria.
kachyelechi@yahoo.com

Call Setup

Some network layer architecture, usually Asynchronous Transfer Mode (ATM), require that the routers along a chosen path from source to destination establish a path for a particular session running on the end systems before data begins to flow. When the session ends the virtual circuit created by the network layer is terminated usually referred to as "tear down". This does not happen in the network layer of the Internet architecture.

NETWORK SERVICE MODELS

The network layer renders two types of service models:

- virtual circuit
- datagram circuit

Virtual circuit (VC) model

This behaves much like the telephone network that uses real circuit and, as such, is referred to as connection-oriented service model in some text. It involves setting up and tearing down a connection-like entity and maintaining connection state information in the packet switches (Kurose and Ross, 2001). The network layer determines the path between the two end systems through which all packets for that VC session will pass through, and all packet switches' routing tables on that path are indexed to reflect the established circuit. The circuit is then terminated on ending the session.

Datagram service model

This is also known as connectionless service model and is much similar to routing a letter through ordinary postal mail. The network layer on receiving a packet attaches the addresses of the two end systems and pops the packet into the network. Any packet switch (router) on receiving the packet, forwards it in the direction of the destination address. The current internet architecture employs this model and is often referred to as best-effort service because as the network pops a packet into the network it give no assurance it will get to its destination.

ROUTING

In other to move packets from one host to another, through a network, the network layer must determine the path the packet should pass through. This is so regardless of the type of service offered by the network layer, virtual circuit or datagram services, and in order to do this job effectively, the router must at a minimum know the following.

- Destination address
- Possible path to all destinations
- Neighbouring routers from which learning processes occur
- Possible best path to all destination

If there is no router in the network then one is not routing (Lammle, 2000).. The router acquires the necessary knowledge for routing by learning from its neighbours about remote networks. It then builds its network from which it determines the possible path to a remote network/destination. Typically, a good path is one that has the least cost. These best paths, however, continually change in the network due to a lot of reasons to be discussed later; hence a best path to a network at time t_1 might not be the best path at a later time t_2. In the virtual circuit service the decision of the best path is made once on establishing the virtual circuit but for datagram service, this decision must be made anew for every arriving packet. This is the principal reason for the possibility of packets arriving out of order. One rule which is important to note here, known as the optimality principle is that in an optimal path between two nodes, say A and E, in a network, if router C lies on this path then the optimal path from A to C or C to E also lies from this path.

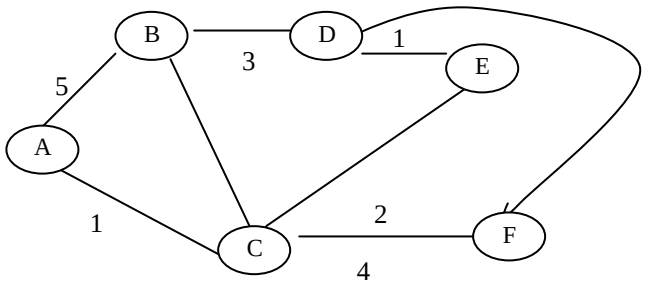


Fig. 1. Abstract model of a network

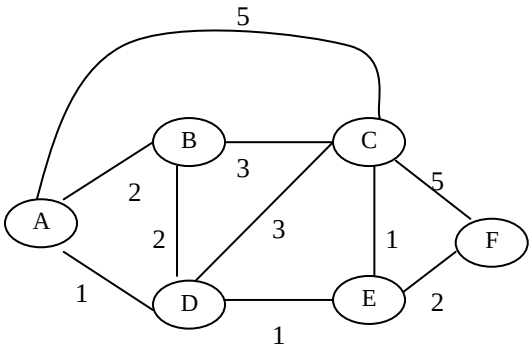
Fig.1 gives an abstraction of a small real network used to formulate routing algorithms. The lines joining the nodes represent the links with the figures on the lines representing the cost between the nodes. The cost is computed with some metric which might be any one or combination of bandwidth, level of congestion, physical distance, etc. The method of computation of these costs also lead to classification of routing algorithms as static or dynamic. In static routing, route changes do not occur often and the path measure is usually configured by the network administrator, but in dynamic routing the routing paths change in response to network topology or load changes. A more general way of classifying routing algorithms which is going to be used here is whether each router has a complete/global knowledge of the network or each router develops its knowledge in a propagating manner. Based on this criterion, there are two types of routing algorithms, namely:

- link state routing algorithm
- distance vector routing algorithm

In many texts, one will find these algorithms referred to by many names but the ones used here are the most common names.

LINK STATE ROUTING

In link state (LS) algorithm each router has a complete knowledge of the entire connectivity and costs between every node in the network and then uses it to build its routing table. In other words every node accepts as inputs to the link state algorithm, the network topology and all links' costs. This is achieved by each node first meeting each of its neighbours, learning its identity and cost to get to it from itself. It then constructs a link state packet (LSP) having these information and broadcasts it to all other nodes in the network. Each of these nodes on receiving these packets then starts to build its routing table (Perlman, 2000). The principal algorithm used here is based on Edsger Dijkstra algorithm, which is discussed using the network of Fig.2.



A	B	C	D	E	F
B/2 D/1 C/5	A/2 D/2 C/3	A/5 B/2 D/3 E/1 F/1	A/1 B/2 C/3 E/1	D/1 C/1 F/2	E/2 C/5

Fig 2.An abstraction of a network (after Kurose and Ross 2001) with corresponding LSP of each node

Consider a database DB as holding all nodes whose least cost path from source A for example is known. Upon initialisation, A having the least cost from A (of course it doesn't cost anything to get to itself) is added to DB and the cost to every other node is computed from A of which at this time, only A's neighbours' costs are known, hence the rest are set to infinity. In the second step, the node with the least cost is added to DB and the distance to all other nodes not in DB computed again. Again the node with the least cost is added to DB and so on until all nodes are in DB. The algorithm for this now follows, with the corresponding results of iteration from node A shown in Table1.

Initialisation: add self to DB for all nodes *n.*

If *n* is neighbour to self then cost = cost //known cost else cost = infinity//unknown cost ,while ([DB] < [LSPs]).

Find *n* not in DB with cost = minimum add *n* to DB for all *n*'s neighbours not in DB cost = minimum (cost from self to *n*'s neighbour, cost from self to *n* + cost rom *n* to *n*'s neighbour)

Table 1.Result of running LS algorithm on node A (after Kurose and Ross, 2001)

step	DB	B	C	D	E	F
0	A	2,A	5,A	1,A	∞	∞
1	AD	2,A	4,D		2,D	∞
2	ADE	2,A	3,E			4,E
3	ADEB		3,E			4,E
4	ADEBC					4,E
5	ADEBCF					

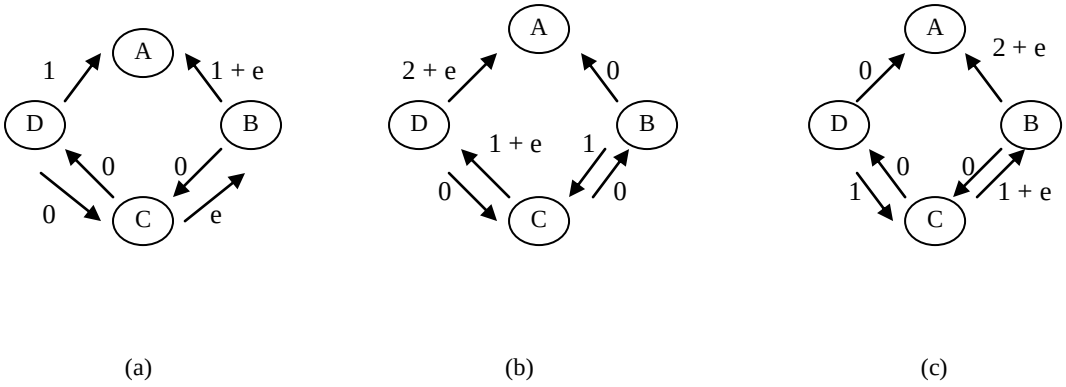
3,E stands for a total cost of 3 on arriving to the node through E from the host node – A in this case

You might have noticed that after step 1, nodes B and E are found to have the least costs. Here one node among the two is chosen arbitrarily.

Issues involved with link state routing

(a) Oscillation where link cost is dependent on amount of traffic

Consider the network shown in Fig.3, with nodes A, B, C and D. Here the link cost is not symmetrical i.e. A to B is not equal to B to A. If nodes D and B originate a unit of traffic each to A, the load from each of the nodes will go clockwise and anticlockwise respectively. If node C originates traffic equal to e destined for A, the link cost will appear as the diagram of fig 3a. When next the LS algorithm is run, B and C will find out that it is cheaper to route to A in a clockwise direction and the cost will appear as fig 3b. On running the LS algorithm again, B, C and D will again discover that it is cheaper to route to A in an anticlockwise direction and so on. In trying to solve this problem, making the link cost not dependent on the amount of traffic carried will not be an acceptable solution since one major goal in routing is avoiding congestion. One solution that might seem acceptable is having the routers run the LS algorithm asynchronously. Another solution will be to adopt the technique used in flow based routing whereby average flow over a period of time is used. Here, using the average flow and capacity of a given line, the mean packet delay is computed for that line, whence a flow - weighted average for the whole subnet is computed (Tanenbaum 1996).



(a) (b) (c)

Fig .3. Oscillations with link state routing (after Kurose and Ross, 2001).

(b) Sequence numbers/age schemes.

There are certain problems also witnessed in distributing LSPs Ordinarily, distributing information in a network will be done using the information in the routing database. But the routing database relies on LSPs and distributing LSPs as well relies on the routing database – recursion problem

How can a router know which of a particular neighbour's LSP that was generated more recently or not the same LSP that has traveled through a longer route thereby arriving late?

In solving the first problem, flooding can be used whereby every received LSP is disseminated on every link except the one that it arrived from hence leading to the second problem. Using sequence numbers whereby each LSP has a number and the router keeps track of its LSPs' numbers, a router receiving an LSP from say node B compares the sequence number with the sequence number of node B's LSP in its database and determines if it is a newer LSP, same LSP or an older LSP. Now, our new problem is with a sequence number field of finite size what happens on getting to its maximum value. One would expect the sequence to be reset. If a router stops receiving router B's LSPs for a while due to some fault/failure then suddenly starts receiving it, how does it determine if they (the arriving B's LSP and the B's LSP on its database) are of the same sequence or which is of newer sequence? If again due to some failure a router goes down and looses it track of sequence numbers, should it reset and start from zero? These problems can be solved by introducing a second field known as the age field. This starts at some value and is periodically decremented by routers as it is held in memory. Hence when an LSP's age reaches zero, an arriving LSP with a non-zero age is accepted as a newer LSP. Therefore, the elements of an LSP are as shown Fig.4 (Perlman. 2000).

Source
Sequence number
Age
List of neighbours

Fig .4 .Contents of a link state packet (LSP)

DISTANCE VECTOR ROUTING

The distance vector (DV) algorithm, also known as Bellman-Ford algorithm, is a distributed and iterative algorithm in that each node receives some information from its neighbours (computed minimum distance to all known destinations), performs some calculations, and then distributes an information back to its neighbours. The process is iterative and self-terminating from the fact that when a node receives information from its neighbours, it only forwards another information to its neighbours if only the received information brought about a change in any of its minimum paths to a destination.

Consider the network Fig.5 in which, for convenience, the link costs are all set to 1 making cost between nodes a measure of hop count.

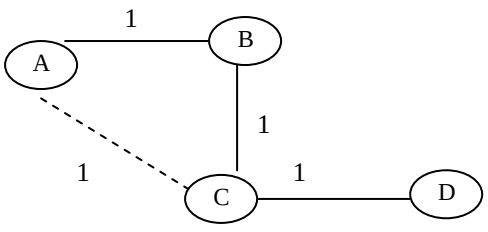


Fig .5. An example of distance vector (DV) routing network.

On running the algorithm, first ignoring the dashed line, node D sends to C its distance table entries having just C as destination and neighbour with cost to it equal to 1. At the same time, node C sends to

B and D its distance table entries having destinations and neighbours as B and D and cost equal to 1 in either case and so on.

Node D on receiving C's distance table entries finds out that it can get to B via C and with a total cost of 2. It then updates its table accordingly and resends the new information to its neighbours. This algorithm terminates (converges) when all nodes on receiving an information from its neighbour, there is no change brought about in any of its minimum costs to a destination, hence does not need to broadcast any information to its neighbours.

Suppose the link represented by the dashed line Fig.5 was down and suddenly comes up, A and C find out that they can get to each other directly with only a cost of 1 and no longer with a cost of 2 via B. They then update their entries on the distance vector table with their new neighbours and new minimum cost and broadcast to their neighbours. The listing of Fig.6 show the progression of the distance table in each of the nodes with the circled entries as the minimum cost. Notice that a node, on receiving an information from its neighbours, checks for a new cost to a destination and only informs its neighbours if it finds a change in its minimum cost to a destination. The DV algorithm is as follows:

Initialisation

for all neighbours *n*

cost to destination d via *n* = infinity

cost to *n* via *n* = cost //direct cost from node to neighbour

to all neighbours send minimum cost to all destinations

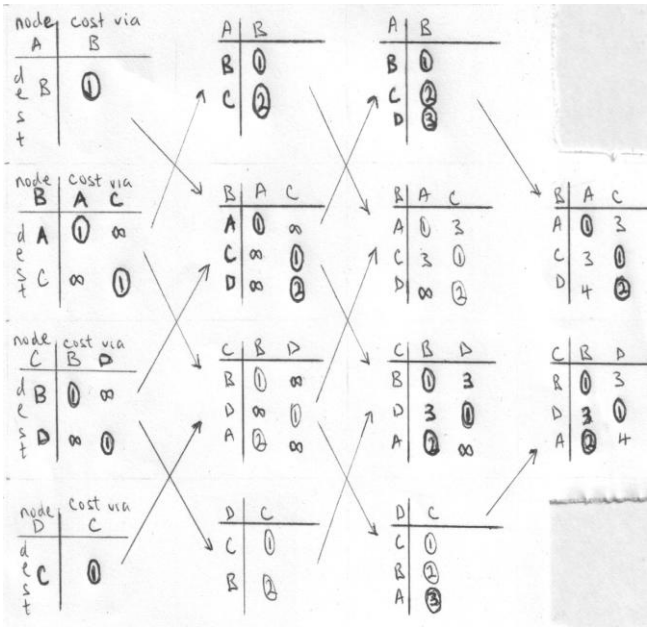
when link cost to neighbour *n* changes for all known costs to destinations via *n* new cost = old cost + change //change can be positive or negative if minimum cost to any destination has changed

send new minimum cost to all neighbours when cost from neighbour *n* to destination d changes //on receiving update from *n*

cost to d via *n* = minimum cost to *n* from self + new cost from *n* to d

if minimum cost to d has changed

send new minimum cost to all neighbours



and when the link represented by the dashed line comes up we have

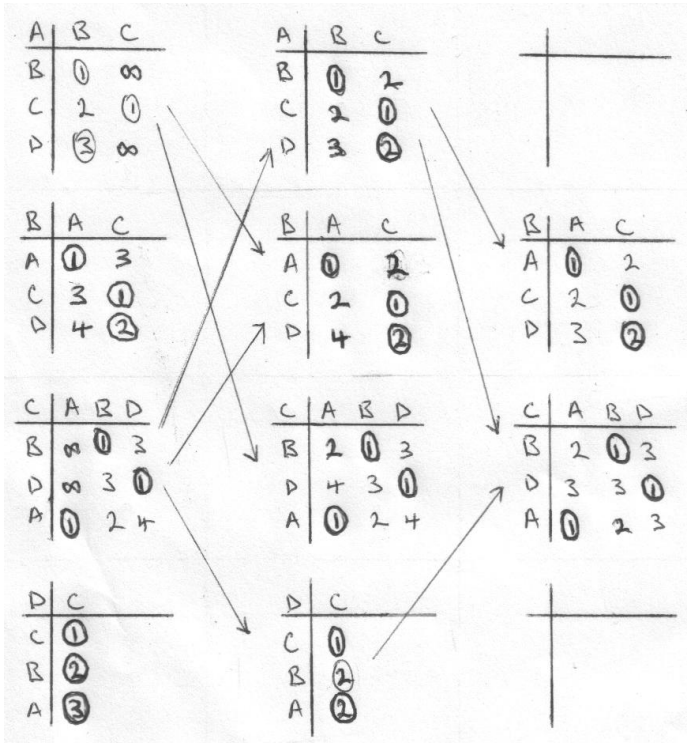


Fig. 6.Outcome on running DV algorithm

Issues involved with distance vector routing

(a) Routing loop/slow convergence

In the network Fig.5, the good news was that when there was a change (the link represented by the dashed line came up) only two iterations were needed for the algorithm to converge. Now consider the network of Fig. 7 in which the link cost between X and Y changes

from 4 units to 60 units. Here, we only examine what happens in cost to X from both Y and Z with the following steps (refer to fig.7):

i. Y notices a change in link cost from itself to X and makes a change. Its minimum cost to X changes and through the route through Z with a cost of 6. It then informs its neighbour of the change in its minimum cost from 4 to 6. We ourselves can see this is wrong only because we have a global view of the network.

ii. Z is informed of the change in cost from Y to X. It updates it's cost to X via Y to the new cost from Y to X which he has been told is 6, plus its cost to Y which is 1 all making a total of 7. Its minimum cost therefore has changed from 5 to 7 and it informs its neighbours of the new cost.

iii. Y is informed of the change in cost from Z to X. It also updates its cost to X via Z to new cost of Z to X which he's been told is 7, plus its cost to Z, which is 1, all making a total of 8. The process continues.

The process will actually persist for 48 iterations before Z's route to X, via Y, will be greater that 50 and Z concludes that its best route to X is via its direct connection with a cost of 50. It will even continue rising gradually again to 60 if link X to Z suddenly changes to a value above 60.

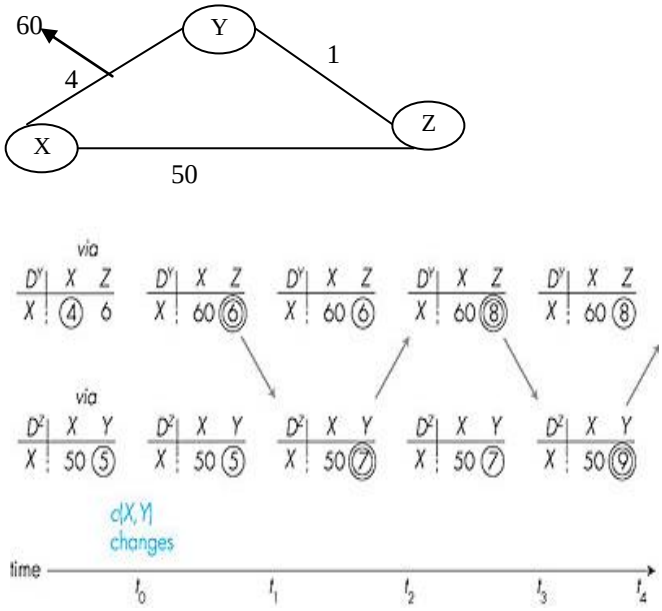


Fig .7.Network illustrating slow convergence (after Kurose and Ross 2001) .

(b) Count to infinity

Suppose that in the above problem there was no direct connection between X and Z and that the link between Y and X went down instead of rising to 60. In this case, Y and Z will continue counting to infinity and in the process packets for X get bounced between Y and Z until they die of old age (Perlman. 2000). In some texts, the two problems are referred to as the same.

(c) Hold down

The problems mentioned above can be curtailed using a technique known as hold down. In this technique, when a node's path to a neighbour n goes down, it waits for some time (hold down time) before switching to another path and during the same time, it broadcast its path to n as infinity to other neighbours. The idea behind this is that during the hold down time, the news of the gone down link will have circulated enough through the entire network for every node to have forgotten the path. However it has been confirmed that this method does not always prevent count to infinity problem and as well slows down convergence in many cases (Perlman. 2000).

Reporting the entire path

In this technique a node reports to its neighbours the cost to a destination and its path to the destination. The neighbours can then determine if the routing was done through them. One is confident with this solution but some say it is expensive.

Split horizon/poisoned reverse

In this technique, if X routes through Y to get to Z, then X tells Y a white lie that its cost to Z is infinity. With this, when link from Y to Z goes down, Y does not switch its route to Z to via X since X says it cant get to Z. However this solution is limited to three-node networks. Fig. 8 shows a larger network with the cost as hop count and the link from C to D goes down. Using the above technique, A and B broadcast cost to D as infinity. But, here, A sees that B to D has a cost of two and switches it route to D to via B and then broadcasts this to its neighbours, while B is as well doing the same thing. Node C then sees this route and sets its cost to D to four initiating the count to infinity problem.

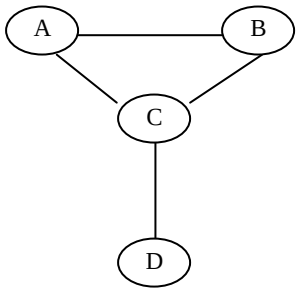


Fig .8. Failure in split horizon

The simple split horizon scheme omits routes learned from one neighbour in updates sent to that neighbour whereas split horizon with poisoned reverse includes such routes in updates, but tells the white lie that its cost to it is infinity (Hedrick 1998).

COMPARISON OF LINK STATE(LS) AND DISTANCE VECTOR (DV)ROUTING ALGORITHMS.

Message complexity and Bandwidth consumed

LS routing requires each node to know the cost of the entire links in the network which requires O(nE) messages to be sent, n being number of nodes and E the number of links. When there is a link cost change, DV routing will only propagate the result if there is a change in its minimum path and as well the message will only get to the nodes concerned while other nodes and links remain up with no routing control traffic. With LS routing a single change will be propagated through the entire network. But in DV routing, a single link change can cause multiple control packets transmission over a single link especially with count to infinity problem hence increasing the overall bandwidth consumed but with LS routing, even though the change is propagated through the entire network it only goes through one link just once. Though the issue deserves more careful and practical study, for now both algorithms are modest and the criterion should not be used in choosing any algorithm.

Robustness

In link state routing, a malfunctioning router can

(i) claim to have a link that does not exist or vice versa

(ii) generate wrong sequence numbers for its LSPs or even corrupt sequence numbers of others' LSPs

(iii) fail to forward others or its LSPs (Perlman. 2000).

In distance vector routing a malfunctioning router can advertise incorrect least cost paths to any/all destinations, which might cause that router to be flooded with packets from other routers (Kurose and Ross 2001). In 1997 a malfunctioning router in a small ISP provided national backbone routers with erroneous routing tables causing other routers to flood the malfunctioning router with traffic and as well large portions of the Internet to become disconnected for up to several hour (Kurose and Ross, 2001).

Memory

In an autonomous system (AS) routing (network in an internetwork running one routing protocol and is independent of other networks in the internetwork), to run DV routing, a router must keep a distance vector to each node n from each of its neighbours K making a memory requirement of O(K * n). In LS routing, each node's LSP contains information about each of its neighbours and each node is required to keep all nodes' LSP again making a total memory requirement of O(K * n). But in some inter-autonomous routing, where the inter-autonomous routers are more than the lower level autonomous systems, DV routing only needs to keep as destination entries, gateway routers, but LS routing need to have all the routers' LSPs which contains entries for all its neighbours.

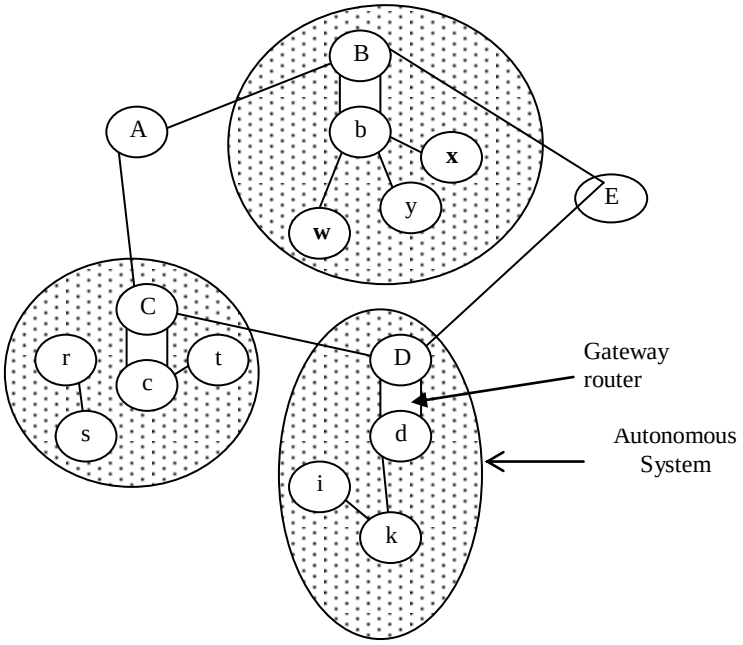


Fig .9. Abstraction of autonomous systems showing their gateway router paths.

In Fig.9, routers A, B, C, D and E are routers for running the inter-autonomous routing protocol but only three out of these routers B, E and D are gateway routers to autonomous systems X, Z and Y respectively. In this scenario using a DV for running the inter-autonomous routing protocol, the distance table entries need only to have as destination E, D & B where as with LS routing every router will have LSP with full information as previously described.

Functionality

LS routing provides more functionality in the following areas (Perlman. 2000).

- It is easier to discover the topology of a whole network by querying just a single router because each router contains information about the entire links and nodes in the network.
- Trouble shooting a network is easier because just by querying a router, one can discover broken links in a network in which the topology is known.
- Source routing is more powerful with LS routing. In times of security specific router can be avoided.

Speed of Convergence

LS routing actually converges more quickly than distance vector routing. We have seen that DV routing suffers from oscillations a lot and even if abated, a router on receiving a control packet runs its routing algorithm first before forwarding the packet whereas in LS routing, a router can just recognise a new LSP and forward it before recalculating routes. However one thing worth noting here is that not

all link changes bring about propagation of information in DV routing and as well different link changes brings about different amounts of propagation but with LS routing, no matter the change, the information must be propagated through the entire network though passing through each link just once.

OTHER ISSUES INVOLVED IN ROUTING

Hierarchical routing

Today's public Internet consists of millions of interconnected routers and more than two billion host. Having each of these routers keep routing table entries for each and every other router is not practicable. This will in fact leave the Internet with just distributing routing control information of which the routers will never converge. The problem of size is solved by hierarchical routing. It entails grouping the routers into regions and sub regions known as autonomous systems (AS). Each router thus knows all the details about how to route packet to destinations within its own region but knows nothing about the internal structure of other regions. An organisation can then have its own AS and run whatever routing protocol (intra-autonomous system routing protocol) it pleases like DV or LS algorithms. Some of the routers in the AS have the added task or routing packets outside the AS and are referred to as gateway routers. These gateway router then use a routing algorithm known as Inter-Autonomous System Routing Protocol to route packets among various ASs. Figure 9 shows what is being explained here with router B, C and D being the gateway routers running Inter AS routing protocol. However, the gain achieved in having each router hold smaller amount of information is not free. A penalty is paid is paid in the form of increased path cost. There are times when there exists a smaller cost path between two routers in different ASs, but because the larger cost path is a smaller cost path for most other routers in the two ASs, that larger cost path is which is a shorter cost path for most other routers is then accepted as the path for routing between the two ASs. (An AS can have multiple gateway routers). But again the increase in effective mean path cost caused by hierarchical routing is sufficiently small making it is usually acceptable (Tanenbaum 1996)

Load Splitting

The idea behind load splitting is to be able to forward packets to a destination in multiple directions rather than a single path (minimum path), which might lead to overloading the path. With distance vector routing, load splitting can be achieved relatively easier because its forwarding database consists of neighbours (unlike a single neighbour in LS routing) through that traffic for a particular destination can appropriately be sent through. In LS routing, the data structure is in the form of triples of the form (ID, path cost, forwarding direction). In other to achieve load splitting the triples will be replaced with soothing like (ID, path cost, {set of forwarding directions}). Thus when adding a forwarding direction, other paths with equally minimal or close to equally minimal costs are added. An algorithm is then devised for selecting among the choices when forwarding a data packet of which might be any of

- move through the list of neighbours one after the other
- choose a neighbour from the list in a random manner
- use any method above and as well include a ratio equal to path congestion making it possible for least congested links to be selected more
- make selection according to returned information on path congestion

Load splitting no doubt has increased network capacity and reduced oscillations, but most probably, load splitting will only be effective in a network where the rate of change path cost is relatively low. Where the rate of change of path cost is high then the dynamic routing algorithm should be taking care of load sharing in the network.

Link Cost

A lot of arguments still arise as whether the link cost should be a fixed quantity or should it vary with link utilisation. Some factors that should be noted are as follows

- Having link cost configured/assigned by network managers requires extra work and reduces the beauty achieved with plug-n-play devices
- Dynamic routing algorithms should be able to handle link costs/topology changes
- With dynamic routing, likelihood of the network being more in the unconverged state is more and most of the bandwidth will then be taken up by routing control packets.
- In static routing, link cost must not necessarily be configured by human. It could still be computed by routers on booting.

With these considerations, some authors suggest that there is actually little difference in network capacity achieved in trying to increase this capacity with costly algorithms. He is of the view that taking into account additional bandwidth consumed by routing control packets, the difference between simple routing and what can be gained by constantly monitoring delays on each link and adding control traffic to keep all routers informed is probably negligible.

Specific Routing Protocols

Here specific routing protocols are explored only in relation to their routing algorithms and tables.

Routing Information Protocol (RIP)

This is a distance vector protocol that operates in a manner very close to the idealized protocol examined earlier in section 3.2.2. It is an inter-AS routing protocol with the latest version, version 2 defined in RFC 1723. Each link has a cost of one and the maximum hop count set to 15. This limits the number of nodes in an AS to a maximum of 15 hops in diameter. Distance table are exchanged approximately every 30 seconds and timed out after 180 seconds as well triggered updates allowed. A triggered update is when a node notices a change in cost, advertises this message immediately even it is not time for its regular updates. The protocol also implements split horizon with poisoned reverse. RIP version 2 supports subnet masks but the routers must be aware of the subnet mask for a particular network number. RIPng for IPv6 specified in RFC 2080 is the RIP version for routing in next generation Internet Protocol or Internet Protocol version 6 (IPng or IPv6). The main consideration in this protocol is on the address specification, hence there is a modification on its message format and as well the algorithms have to accommodate longer addresses.

DECnet

This is also a distance vector protocol. Control information is sent reliably over point-to-point links. Distance vectors are advertised periodically usually of about 10 seconds. Distance vectors are not also timed out but a separate hello message is sent periodically to say a neighbour is still up. On occasion of a link cost change, only that change is announced and not the whole distance vector. However the comp0lete distance vectors form all neighbours are stored in all nodes like the distance vector table of fig 3.5. This is unlike RIP that only stores the next direction (neighbour) with minimum cost. Hence in DECnet when a neighbour goes down, the node has all information needed to compute the next best path.

Open Shortest Path First (OSPF)

This is a link state protocol with its latest version defined in RFC 2178. Individual link costs are configured by the network administrator and a router constructs a complete topological map of the entire AS. Routers continually maintain its neighbours by sending and receiving hello packets and these packets indicate how often a neighbour must be heard from in order to stay alive. Updates between routers are also authenticated to ensure only trusted routers can participate in the OSPF protocol. This makes it possible for undesired routers to be avoided. Each router maintains a database of the AS's topology known as the link state database, each router having an identical database (Moy 1997). Each router then constructs a shortest path tree form its self, and on existence of equal minimum paths, both paths are stored and traffic distributed equally among them. OSPF also allows sets of routers in an AS to be grouped together, each set known as an area. Each area's topology is unknown to other areas, thus enabling reduction of routing traffic. OSPF for IPv6 defined in RFC 2740 specifies the OSPF version of IPv6. One unusual thing in this protocol is that OSPF router IDs, area IDs and LSA link state IDs remains at the IPv4 size of 32 bits whereas one of IPv6 requirements is the increase of address spaces to accommodate more nodes. This may actually lead to a modification in feature. Most of the algorithms used in OSPF for IPv4 have been preserved in OSPF for IPv6 (Coltun et al, 1999**).**

Intermediate System to Intermediate System (IS-IS)

This is a link state intra domain routing protocol defined in RFC 1142. IS-IS was not originally designed for IP. It can support multiple layer 3 protocols of which modification were later made to support routing in TCP/IP defined in RFC 1195. The hello messages and LSP have a field to indicate which protocols they support in other to avoid a node receiving a packet in a foreign format. The address of the router's IP interface is also included in the format in other for a router to know the IP address of its neighbour. IS-IS also supports portioning and AS into areas like OSPF. There is also an IS-to-IS hello that enables routers to co-ordinate with their neighbours of which is a settable parameter to indicate frequency of occurrence. The IS-to-IS hellos are of two types, between LANs and between point-to-point links.

Border Gateway Protocol(BGP)

This is and inter AS routing protocol with version 4 specified in RFC 1771. It is the de facto standard inter-domain routing protocol in today's Internet Though BGP has some flavours of distance vector protocol, it is commonly referred to as path vector protocol because it propagates path information and not cost information. In updates lists of intermediate routers to each AS is sent to neighbours. Each router has its own desired metric used and it then uses it in selection of optimum path.

The BGP protocol has four types of messages.

- OPEN – used by peers to identify and authenticate each other and as well specify time period information.
- KEEPALIVE – used to let a peer know that a sender is live has no other information to send. It is also used to acknowledge an open message.
- UPDATE – used to advertise/withdraw paths to destinations.
- NOTIFICATION – used to notify error or terminate a BGP session.

BGP-4 also provides a new set of mechanisms for supporting classless inter-domain routing and aggregation of routers or AS paths.

CONCLUSION

We have learned about the basic routing principles and have digressed as well to a few other routing protocol types. Note that the other protocols discussed make use of the principles of either distance vector or link state routing. As can also be deduced above, you find out that the router has a lot of work to perform on the very busy and large Internet in routing thousands to millions of packets transversing it. However when you learn about classful addressing, a technique adopted to reduce the fast rate of depletion of Ipv4 address, you find out that a router might only need to look at the network address part of the whole address to know the direction to forward a packet through. As well with the Autonomous system (AS) partitions arranged in hierarchical structure, the scaling problem is largely reduced because within an AS, a router only need to know about the other routers within its AS and any incoming packet that it does not have knowledge of its destination address, it simply forwards it to the gate way router within its AS. To route packets among Ass, an inter-AS routing protocol such as Border Gateway Protocol (BGP) is used. Also with the existence of ASs, a network administrator can choose the best routing protocol to use that will best suit the various services that is network would render. Even with these techniques of scaling down the router's job, network designers still have their job to do on reducing the amount of work done by the routers. This can be achieved by reducing virtual circuits, fragmentation, and providing perhaps a best effort (like datagram) service.

REFERENCES

Clements, A. (2000). *The principles of computer hardware*, Oxford University Press Inc., London

Coltun, R., Ferguson, D. and Moy, J. (1999). Open shortest path first for IPv6: RFC 2740 (http://www.rfc.net/rfc2740.html)

Hedrick, C. (1998). Routing information protocol: RFC 1058 (http://www.rfc.net/rfc1058.html).

Kurose, J. F. and Ross, K. W. (2001). *Computer networking*: *A top-down approach featuring the internet,* Addison Wesley Longman, Inc, New York

Lammle, T. (2000). *Cisco certified network associate study guide.* SYBEX Inc.

Moy, J. (1997). Open shortest path first version 2: RFC2178 (http://www.rfc.net/rfc2178.html).

Perlman, R. (2000). *Interconnections* (Second Edition): *Bridges, routers, switches, and internetworking protocols.* Addison Wesley Longman, Inc., New York

Tanenbaum, A. S. (1996). *Computer networks*. Prentice-Hall, Inc., London