Brouter: The Transparent Bridge with Shortest Path in Interconnected LANs

Ying-Dar Lin and Mario Gerla
Computer Science Department
University of California, Los Angeles
Los Angeles, CA 90024

Abstract

Transparency is a popular feature in a distributed system where users can access any local or remote resources just as if they were local. As different units in the distributed environment can be integrated into a single unit by means of a distributed operating system, so can multiple LANs be connected by transparent bridges and appear as one LAN to upper layer protocols and end stations. End stations just transmit their packets on their local LANs and leave the forwarding task to bridges. Transparency of the interconnected LANs requires "self learning" at bridges. The main problem of self learning is packet looping. Thus, a "spanning tree" algorithm was developed to eliminate loops in networks. However, two main disadvantages of the spanning tree solution are non-shortest path routing and resource wastage. In this paper, a protocol is proposed which combines the features of transparency and shortest path. The result is a bridge which supports self learning and distributed routing computation. Since this bridge has all routing facilities as a router, it is called "brouter" (ie. bridge+router). Brouters are compared with current bridge schemes. Also, performance aspects of the protocol are examined.

1 Introduction

Both contention-based protocols like CSMA/CD and token-based protocols like Token Ring have their limitations: maximum number of stations, poor fault isolation, limited geographical extension, and throughput. Due to the limitations, LANs are routinely interconnected via conventional gateways like XNS, Sytek, and DNA. But those gateways require compatible protocols layered above the MAC protocol [1]. That is, in order to participate in the interconnected LAN, end stations need to implement consistent protocols above the MAC layer. End stations need to specify which gateways should forward these packets when they transmit packets on their local LANs, which means they need to implement a network layer protocol which is not necessary in the single LAN environment.

In fact, network layer may be null in the interconnected LAN environment. To overcome this problem, the bridge approach, which is transparent to the upper layer protocols and end stations, was proposed several years ago [2][3]. But the transparent bridge solution with spanning tree algorithm still suffers of a major limitation: non-shortest paths [4][5]. An alternative scheme, the source routing bridge, was later developed. This scheme includes a shortest path-finding algorithm; however, it violates transparency [6][7].

The overhead of source routing scheme on each host is overwheleming in a large interconnected LAN although this scheme possesses more sophisticated routing and network management facilities and reduces the complexity of the bridge [8]. The spanning tree bridge offers the valuable features of transparency for hosts and yet reasonable implementation complexity. However, it does not provide optimal path and flow control.

Should we go back to routers and gateways in order to obtain a sophisticated service and, at the same time, a reasonable complexity? Or should we stay with the bridges? The reason why the MAC-layer bridges are becoming increasingly popular is due to the low protocol processing time which is critical especially with the introduction of high-bandwidth fiber optics LANs and the increasing user bandwidth demand. The tradeoff here is that, on one hand, we are trying to keep the protocols as simple as possible; on the other hand, we

This research was supported in part by a MICRO grant from the State of California and Pacific Bell, and in part by a grant from Mitsubishi Electric.

desire a more sophisticated network service and operation. A new scheme, the brouter, is proposed here to keep the LAN interconnection protocol at the MAC layer while gaining the advanced features of routers without overwhelming complexity.

The brouter protocol with joint self learning and distributed routing algorithm combines the features of transparency and shortest path, where transparency comes from joint self learning and shortest path comes from distributed routing. In fact, it is a combination of bridge and router (hence the name "brouter").

Section 2 gives a general idea of how the brouter solution works. The detailed protocol is presented in section 3 with subsections on the major mechanisms of brouters and how they work in parallel. Since transparency and shortest path mean the burden of packet routing is left to the bridges, the bridges carry the overhead of not only self learning but also distributed routing. Section 4 considers the overhead of brouters and compares it with current schemes in various aspects.

2 Basic Approach of Brouter Solution

Before presenting the approach adopted in transparent brouter scheme, we need to identify the goals of this approach. Basically, there are three goals to achieve in this protocol:

- Transparency for end stations
- Optimal paths for internet traffic
- Reduction of routing table size at bridges

Transparency at end stations means that end stations just transmit frames on their local (adjacent) LANs without worrying about whether the destinations are on local or remote LANs. The job to forward frames for stations on remote LANs is left to brouters. Brouters must know where the destinations are, at least in which direction the targets are. This implies self learning by inspecting source address of each detected frame at brouters as in transparent spanning tree bridges. But this time the spanning tree topology is not enforced to resolve the problem of infinite frame looping. Instead, brouters perform joint self learning. By "joint self learning" with neighbor brouters on the same LAN, i.e. by means of information exchange and intersection, each brouter can determine the set of stations on this LAN and then broadcast it to the entire network.

More specifically, the brouters attached to the same LAN will work together to determine the set of station IDs on this LAN by intersecting their own source address data base with the data base broadcast by the neighbors. The determined set will be broadcast to all the other brouters. In simpler words, brouters know the stations on their local LANs by joint self learning with neighbor brouters and know the stations on remote LANs by just listening to the result of other groups' joint self learning. Thus, all stations are grouped into LANs.

However, knowing which LAN the destination belongs to is not enough to determine how to get there. To route the internet frames on shortest paths, a distributed routing algorithm similar to ARPAnet is used. Both internet and ARPAnet algorithms involve the exchange of delay tables. But many differences exist. For example, a frame detected by a brouter is not necessarily forwarded by that brouter. This occurs, for example, when the frame destination is on the same LAN where the frame is detected or some other brouter on this LAN has a shorter path to the destination. Thus, to avoid frame duplication or loss, this protocol needs to guarantee that one and only one among the brouters attached to this LAN will forward this frame. The way to do this is, for each target LAN, to associate a bit to disable or enable the port of a brouter. If the bit for the target LAN is disabled, the frame will be discarded.

The forwarding table in the spanning tree bridge and the path table in the source routing bridge can be extremely large in a real-world environment where there are thousands of hosts. By keeping an entry in the routing table for each destination LAN instead of each host, we are reducing the routing table size from number of hosts to number of LANs, which means from thousands to tens. The LAN routing tables combined with the station-to-LAN mapping enable us to consider a group of stations as one LAN in making routing decision.

3 Brouter Protocol

To provide a clear description of this protocol, we divide the protocol into a number of mechanisms and present them in the following subsections. The proposed brouter protocol is composed of four major mechanisms:

1. Determination of LAN IDs
2. Self learning and routing mechanism

3. Exchange and flooding of mapping vectors

4. Exchange of delay tables in the distributed routing algorithm

Part 2 and 3 support the feature of transparency. Part 1 and 4 reduce the size of the routing table. Optimal paths are obtained by part 4. Before presenting the above four components, the following subsection describes the major data structures required to support these mechanisms.

3.1 Main Data Structures

There are three major data structures, Mapping Table, Routing Table, and Neighbor Table, which must be maintained in brouters (see figure 1).

(a) Mapping Table

LAN ID	Station IDs
L3	S1 S2 S3 ...

(b) Routing Table

LANID	Delay	Out_Port	Next_Brouter	In_Port_Dis(1)	..	(n)
L3	20	P2	B4	0	..	1

(c) Neighbor Table

Port Number	Adjacent LAN ID	Adjacent Brouter IDs
P1	L3	B1 B2 B3 ...

Figure 1. Main data structures to maintain in brouters

(a) Mapping Table:

For the purpose of reducing routing table size and finding out where the destinations are, we need a Mapping Table. The normal usage of this table is to map the station ID to LAN ID. Each entry of the table is a Mapping Vector associated with a LAN. It contains the LAN ID and the IDs of the stations on this LAN.

(b) Routing Table:

For each target LAN, there is an entry containing the delay to it, the output port to forward the frame, the brouter offering the optimal path, and the enable/disable bit for each port. Note that the reason to have Next_Brouter is to be able to tell where the delay table comes from in the distributed routing algorithm. If the delay table is from the brouter offering current optimal path, this means there may be a change for this path. If it is from other brouters, this delay table may contain the information of inferior paths or new optimal path. They need to be treated differently.

As previously mentioned, we need the extra bits to enable/disable the ports in the interconnected LANs environment in order to guarantee one and only one delivery. In figure 1.(b), the In_Port_Dis(n) bit associated with port n is set to one in the entry for LAN L3, this means that the brouter, X, will discard the frame detected on port n with destination on LAN L3. Among the entries associated with LAN L3 in the brouters attached to the same LAN L, there should be one and only one entry setting its In_Port_Dis bit to zero (enabled) so that only one brouter will forward the frame to destination LAN L3.

(c) Neighbor Table:

For each port of the brouter, there is an entry composed of the ID of the adjacent LAN attached to this port and the IDs of neighbor brouters on this LAN. The neighbor brouters IDs on this adjacent LAN are needed in the determination of LAN ID. If all brouters on this LAN have announced their ID and port number since last announcement, the LAN ID is then determined for this cycle. The details will be discussed in the following subsections.

3.2 Determination of LAN IDs

- Periodically, each brouter announces its brouter ID and port number on this adjacent LAN by transmitting a HELLO message to all other brouters on this LAN. This process is repeated for all adjacent LANs.
- Upon receiving a HELLO message, the brouter determines the LAN ID by the following:
  If the brouter ID, X, in current LAN ID is larger than the brouter ID, Y, in the HELLO message, replace current LAN ID with [Y concatenated with port number from the HELLO message], otherwise, current LAN ID remains the same.
- If current LAN ID is changed, this LAN ID is broadcast to the entire network by the designated brouter so that each brouter in the whole network can modify or make an entry for this LAN ID in its Mapping Table.
- Upon receiving a HELLO message, the brouter also makes an entry in its Neighbor Table where each

entry identifies its port number associated with the adjacent LAN ID and brouter IDs on this LAN.

In the interconnected LANs, hosts on the same LAN can be treated as a group in making routing decision. Brouters' forwarding task is to forward the frame from source LAN to destination LAN. Then, the target host on the destination LAN will detect the frame itself. Thus, in this scheme, it is necessary for brouters to have the Mapping Table that partitions all of the hosts into groups in terms of LANs. In turn, we need unique LAN identifiers for all brouters so that they can exchange the partition information learned by themselves.

A LAN ID is determined as the smallest brouter ID of brouters on this LAN concatenated with the port number of that brouter for that LAN. The brouter with the smallest ID can be considered as the designated brouter on that LAN as well as the designated bridge in spanning tree scheme. Since it is very possible that some brouters may be down or new brouters may be installed during the network operation, LAN IDs may also need to be determined dynamically. By continuously exchanging local HELLO messages which contain brouters' IDs and port numbers, the LAN IDs can be kept up-to-date. A newly determined LAN ID will be broadcast by the designated brouter to the entire network so that all brouters have the same view about partitioning.

A positive side effect of partitioning is the reduction of routing table size, which has been mentioned earlier.

3.3 Self Learning and Routing Mechanism

- Upon detecting any frame on LAN LX, brouter X inspects the source of the frame. And insert the source station ID into the temporary Mapping Table for LAN LX.
- To determine which port to route the frame, map the destination station ID to destination LAN ID, L, using the Mapping Table. Then, map L to port number, P, using the Routing Table:

  1. If L = 0 (the destination station ID is not in Mapping Table) or port number for L in Routing Table is empty, flood the frame to the network.

  2. If (i) the incoming port for L is disabled in Routing Table, or (ii) L or the outgoing LAN attached to P is equal to incoming LAN LX, discard the frame.

  3. Otherwise, forward the frame to port P.

In the above algorithm, since each entry in Routing Table is for each target LAN, we need a two-stage mapping, namely: (1) from station ID to LAN ID via the Mapping Table, and; (2) from LAN ID to port number via the Routing Table. There may be a temporary incorrect knowledge about stations on the adjacent LANs because we always insert the source station ID learned into the Mapping Vector for this adjacent LAN even if the station does not belong to the LAN. But after intersection of the Mapping Vectors of all neighbor brouters on this LAN, the station IDs which do not belong to this LAN will be dropped. In fact, we may give higher priority to Mapping Vectors for remote LANs by consulting them first so that the temporary incorrect knowledge won't affect the routing decision. That is, if, in the Mapping Table of a brouter, some station ID happens to appear in both the vector for a remote LAN and the vector for a adjacent LAN, it is very likely that the one in the Vector for the adjacent LAN is a transient one except in the unstable status of station migration. Thus, if there is a conflict, the one in the vector for the remote LAN wins.

Routing in interconnected LANs is more complex than in WAN since the brouter need to decide first if the detected frame need to be forwarded. There are two reasons to discard the detected frame:

- The target LAN is the adjacent LAN where the frame is detected
- The target LAN is on the same side as the adjacent LAN where the frame is detected

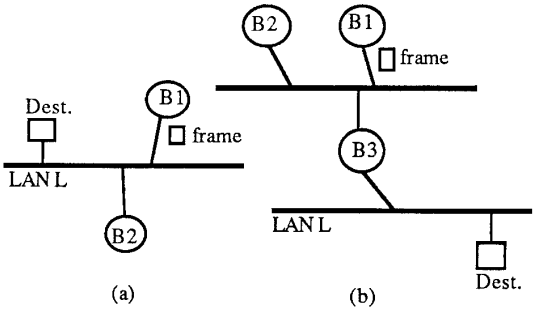

Figure 2. Two situations to drop the detected frame

Figure 2 illustrates two reasons to drop a detected frame at brouters. In both cases, B1 is the brouter detecting a frame and Dest. is the target end station. Case (a) is obvious. Case (b) means that some other brouter on this LAN, B3, can offer a better path than the current brouter. In both cases, the port will

be disabled with respect to the target LAN L, which means the brouter will ignore the frame detected on this port with destination on LAN L.

Note, in case 2 of the above algorithm, that (i) and (ii) are equivalent most of the time. If the target LAN is the adjacent LAN or on the same side as the adjacent LAN where the frame is detected, the incoming port will be disabled with respect to the target LAN. In fact, we may just delete (ii). But, during unstable state (before the mapping algorithm converges), those In_Port_Dis bits may not yet be set. Thus, to reduce frame duplication during unstable state, we may apply (ii) as the second check.

3.4 Exchange and Flooding of Mapping Vectors

- Periodically, each brouter transmits its Mapping Vector for the adjacent LAN L on L in order to decide the set of stations on L and then broadcast this result if the set is changed.
- Associate a timer with each Mapping Vector. Periodically, the brouter with smallest ID on LAN L will initiate a purge operation for L. Upon receiving a purge instruction on an adjacent LAN L, a brouter just purges the Mapping Vector for L.
- Upon receiving a Mapping Vector for an adjacent LAN L, brouter X applies intersection operation of this vector with its own vector for L. X will broadcast the result to the entire network if the following is true:

  1. X is the brouter with the smallest ID on L, and

  2. X has received Mapping Vectors for L from all other brouters on L since last broadcast, and the result of intersection is different from previous broadcast Vector.
- Upon receiving a Mapping Vector for a remote LAN, just change the Mapping Vector to this new one.

Figure 3 shows a LAN with three brouters attached to it. These three brouters will work together to figure out the set of stations on this LAN. And then the designated brouter, for example G1, is the one in charge to let the other brouters of the network know the set of stations on this LAN. The way joint self learning works is the following. Each one of these three brouters learns independently the set of stations in the direction of the port attached to this LAN by the algorithm in previous subsection. G1, G2, G3 will know the stations on this LAN and some other stations not on this LAN. Since G1, G2, G3 are the only brouters that this LAN is connected to, no station which is not on this LAN can be learned by all these three brouters as a station located in the direction of the port attached to this LAN. Thus, after applying intersection with the sets of stations received from the other two brouters, the set left in each one of these three brouters is the set of stations on this LAN. It is guaranteed that the set broadcast later will contains only the station IDs of this LAN.

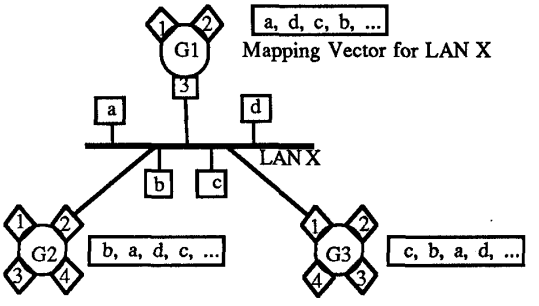

Figure 3. Joint self learning by intersection

However, this set may be only a partial list of station IDs of this LAN. Some stations keep silent and they will never be "learned". For those stations, flooding must be used every time there are frames for them. This is clearly very inefficient. One solution to this is to have end stations transmit IDENTIFIER messages periodically to announce themselves. But if this kind of messages are not used in the single LAN environment, this solution will violate the feature of transparency since end stations need to do something extra.

As in spanning tree bridges, stale Mapping Vectors associated with adjacent LANs will be purged in order to support station migration. In figure 3, the designated bouter G1 will initiate a purge operation.

3.5 Exchange of Delay Tables in Distributed Routing Algorithm

- Each brouter will initialize its Routing Table while determining an adjacent LAN ID:

  (i) Make an entry for this adjacent LAN and insert LAN ID

  (ii) Insert [delay to this LAN, current port number] into [Delay, Out_Port]

  (iii) Set In_Port_Dis bit for current port
- Periodically, each brouter transmits Delay Table on its adjacent LANs. Upon receiving a Delay Table from brouter Y, the brouter X with port P for this adjacent LAN LX modifies its own Routing Table if necessary:

For each entry (LAN L) in Routing Table:

1. If current delay is empty (infinite) and coming delay is not empty (Brouter X has no idea about how far LAN L is, but brouter Y has.), insert [coming delay from Y plus delay from X to Y, Y] to [Delay, Next_Brouter], disable incoming port P.

2. If Y $\neq$ Next_Brouter

   (i) if current delay = coming delay (Brouter Y has the same delay to LAN L as brouter X. A tie!), disable port P if X is not the brouter with smallest ID on LAN LX.

   (ii) if coming delay < current delay $\leq$ coming delay + delay from X to Y (Brouter Y can offer a shorter path to LAN L, but for LAN LX only.), disable port P.

   (iii) if current delay > coming delay + delay from X to Y (Brouter Y can offer a shorter path to LAN L for all X's adjacent LANs.), modify [Delay, Out_Port, Next_Brouter] to [coming delay + delay from X to Y, P, Y] enable all ports, disable port P.

3. If Y = Next_Brouter (The current optimal path to LAN L via Y may get longer or shorter.), modify Delay to [coming delay + delay from X to Y].

Based on these three mechanisms presented above, the distributed routing algorithm will converge to a min-delay path. With the determination of LAN IDs, routing entries are LAN-oriented rather than station-oriented. With joint self learning and Mapping Vector broadcast, each brouter knows which LAN the destination host belongs to. However, brouters still need to know where the destination LAN is. The problem is very similar to the routing problem in a wide area network. But, to avoid frame duplication, brouters need to decide if the detected frames needs to be forwarded first. Brouters will forward the frames only if they are on the shortest paths from source LANs to destination LANs.

3.6 A Picture in Mind

To have a better view of the operation of the brouter protocol, main processes in brouter are shown in figure 4. There are six main processes and two main data structures shown. All the processes can run either in a multiprogramming or a multiprocessor environment. Thus, we need concurrency control for the access of Mapping Table and Routing Table.

Once a frame is detected and goes up to the Receive process in the "Transparent Brouter Layer", it will be routed to Forwarding process if it is simply a data frame. The source station ID along with number of the port where the frame is detected is passed to the Joint Self Learning process. If it is a control frame, it will be routed to one of three handling process. Three types of control frame exist in this environment:

- Delay Table
- Mapping Vector
- HELLO message

Delay Table is passed to the Distributed Routing process which then recomputes the delay to each LAN and updates the Routing Table. Mapping Vector is passed to the Joint Self Learning process where current Mapping Vector is intersected with the incoming one if it is associated with adjacent LAN; if it is for remote LAN, just replace the old Mapping Table with new one. HELLO message is passed to the LAN ID Determining process which keeps the LAN ID up-to-date. Periodically, each of these three processes instructs the Message process to transmit a control frame to the adjacent LANs. Also, routing decision at Forwarding process depends on the content of Routing Table and Mapping Table. Frames may be forwarded or discarded depending on the enable/disable bits in Routing Table.

4 Performance Consideration

A set of considerations apply to the evaluation of LAN interconnection schemes. This may include throughput on each LAN and bridge, delays for intra-LAN and inter-LAN traffic, degree of transparency for end station, network management overhead at hosts and bridges, response to failures and changes, supported services, stability, etc. One may find that the performance issues in internetworking are much more complex than a single LAN or WAN environment. No scheme can dominate all others in all aspects. In general, there are four general performance criteria in an internet with bridges/routers/gateways as the interconnection facilities:

- Transparency for end stations
- Lightweight network management
- High throughput at bridge/gateway
- Low delay for internet traffic

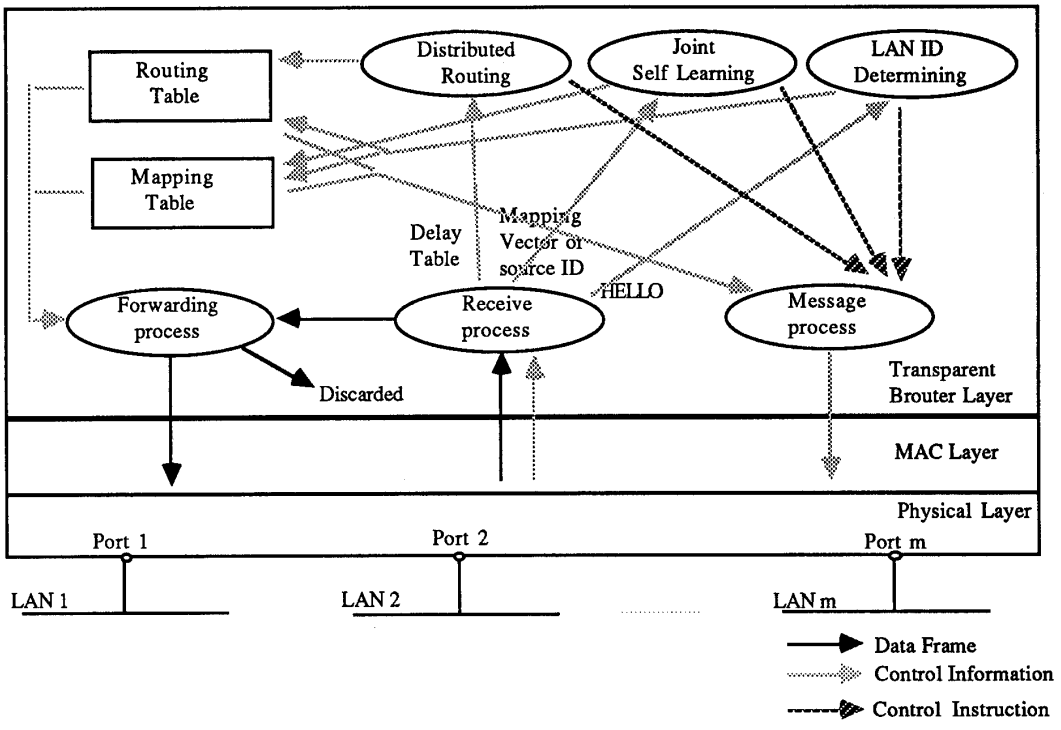

Figure 4. A conceptual view of brouter operation

It is an ideal goal that the internet can provide the end station with a total transparency such that no additional effort is needed to transmit data to a different network. The end station doesn't even know whether the destination is on the same network or not. Thus, the internetworking tasks are all shifted to interconnection facilities. This may cause a "thick" protocol in the interconnection switch, which may form a traffic bottleneck[9]. In order to increase the degree of transparency, the delay for internet traffic should be as low as intra-net traffic. It is more likely to have more internet traffic in interconnected LANs than in interconnected WANs because the former is owned by a single organization where traffic may be evenly distributed among all LANs. With the low delay requirement and large amount of internet traffic, a powerful switch is strongly desired to alleviate the bottleneck and achieve high throughput.

4.1 Brouter Overhead

	Joint Self Learning	Distributed Routing
Space	O(S*N) for Mapping Table	O(p*N) for Routing Table
CPU	O(S) for intersection	O(b*p*N) for processing Delay Table
Bandwidth	O(N) for broadcasting Mapping Table	O(b) for exchanging Delay Table

Figure 5. Major brouter overhead

In figure 4, there are six cooperating processes running in each brouter. Receive, Forwarding, Message, and LAN ID Determining processes are just basic operations to support higher level processes: Distributed

	Spanning Tree Bridge	Source Routing Bridge	Brouter
Transparency	transparent to end station	sources must cooperate with bridges	transparent to end station
Path	non-optimal	near optimal	optimal
Resource Utilization	only on the spanning tree	full utilization	full utilization
Load Sharing	no	yes	yes
Bridge Complexity	has to maintain spanning tree and forwarding table, grows linearly with number of LANs	simple	has to maintain mapping table and routing table, grows linearly with number of LANs
End Station Complexity	none	has to discover and store routes, grows exponentially with diameter of network topology	none
Bridge Delay	longer	short	longer
Start-up Delay	none	route discovery delay	none
Bandwidth Overhead	overhead for periodical exchange of HELLO message for maintaining spanning tree by each bridge	overhead for route discovery frames by each host	overhead for periodical exchange of HELLO message for determining LAN ID, mapping vector for joint self learning, delay table for distributed routing by each bridge
Failure and Change	fast response, bridges detect and reconfigure in the area	slow response, up to end station to detect and find a new route	fast response, bridges detect and reconfigure in the area
Flow Control	difficult	possible	possible

Figure 6. Comparisons of spanning tree bridge, source routing bridge, and brouter

Routing and Joint Self Learning. Thus, for each of these two algorithms, figure 5 lists its overhead where S is the average number of stations on a LAN, N is the number of LANs, p is the maximum number of ports in a brouter, and b is the maximum number of brouters on a LAN. Space and CPU overhead is counted for each brouter, while bandwidth overhead is for each LAN. CPU and bandwidth overhead is counted for each time slot, assuming tasks at brouters are executed once per time slot.

Among these entries, some deserve a detailed explanation. The bandwidth overhead of Joint Self Learning is caused by the exchange and broadcast of Mapping Vectors. However, the former is O(b) which is dominated by the latter, O(N). Since Routing Table is indexed by LAN ID, its space requirement is O(N) plus O(p*N) for In_Port_Dis bits. The total number of Delay Tables a brouter may receive in a time slot is O(b*p) where b*p is the maximum number of neighboring brouters. Because each Delay Table has N entries, the processing overhead of Distributed Routing is O(b*p*N).

4.2 Comparison with Previous Schemes

Generally, routers and gateways provide a higher level quality of service in their operation. For example, they can perform fragmentation and reassembly of

packets, choose light-loaded paths to have better overall throughput, tolerate failures in links or stations, interconnect dissimilar MAC layer LANs [1]. The problem here is that they have higher delay and may not be needed in homogeneous interconnected LANs.

Although spanning tree bridges offer transparency, the spanning tree topology makes optimal path selection impossible. Some bridges and links are idle while others are very busy. There is also a strong tendency to have congestion at the root bridge. This traffic concentration violates our principle of distributed network. On the other hand, source routing bridges fully utilise resources and have more sophisticated routing algorithm, but the large overhead at hosts and links for path discovery is a significiant disadvantage. There is no transparency. Besides, the greater routing effort still cannot guarantee path optimality because once an optimal route is discovered and stored, it will be used for a long time or even fixed. A currently optimal path may become non-optimal in the future due to congestion, changes or failures. On the other hand, frequent path discovery to keep the path optimal will make the overhead more serious.

Brouters seem to solve most of the above problems. But with the features of transparency and optimal path, the bridge complexity has been increased. Thus, we need to worry about the congestion at bridges. By granting higher LAN access priority to brouters which can be influenced by adjusting token holding time in Token Rings and collision resolution algorithm in Ethernets, this problem can be alleviated. In summary, figure 6 gives a list of comparisons between these three bridging schemes.

5 Conclusion

A new network interconnection device, the brouter, is proposed here to provide both transparency and sophisticated network services. Based on the "joint self learning" algorithm, a distributed routing algorithm is executed to obtain the optimal path. Without the enforcement of a spanning tree topology and the excessive overhead of the path-finding algorithm, the features of transparency and optimal path are successfully combined. Also, the routing table size is considerably reduced. However, there is still a tendency of congestion at brouters due to the processing delay and the large amount of traffic introduced by the high bandwidth of fiber optics. Thus, we need powerful bridges and grant them higher LAN access priority.

Recently, fiber optics technology has led to the development of high speed networks. On one hand, since SMDS (Switched Multi-Megabit Data Services) [10] is being deployed via MAN technology (e.g. FDDI and DQDB), we can expect that the scenario of MAN interconnection as a backbone for a distributed system over a metropolitan area will soon prevail. On the other hand, Ethernets, the current local interconnection media, are expected to be replaced by MAN technology like DQDBs as the distributed systems become more sophisticated and tightly coupled. Thus, the proposed brouter solution for current LAN interconnection can then be applied to MAN interconnection and future LAN interconnection.

References

[1] W. M. Seifert, *Bridges and Routers,* IEEE Network, vol. 2, No. 1, Jan. 1988.

[2] B. Hawe, A. Kirby, and B. Stewart, *Transparent Interconnection of Local Networks with Bridges,* Advances in Local Area Networks, K. Kummrele, J. O. Limb, and F. Tobagi, Eds., IEEE press, 1987.

[3] J. Lamont, J. Doak and M. Hui, *LAN Interconnection via Frame Relaying,* INFOCOM 1989.

[4] C. Cheng, I.A. Cimet, and Srikanta P.R. Kumar, *A Protocol to Maintain a Minimum Spanning Tree in a Dynamic Topology,* SIGCOMM 1988.

[5] Radia Perlman, *An Algorithm for Distributed Computation of a Spanning Tree in an Extended LAN,* Ninth Data Communications Symposium, Vancouver, 1985.

[6] D. A. Pitt, K. Sy, and R. A. Donnan, *Source Routing for Bridged Local Area Networks,* in Advances in Local Area Networks, K. Kummerle, J. O. Limb, and F. Tobagi, Eds., IEEE press, 1987.

[7] C. A. Sunshine, *Source Routing in Computer Networks,* ACM Computer. Commun. Rev., vol. 7, No. 1, Jan. 1977.

[8] J. W. Wong, A. J. Vernon, and J. A. Field, *Evaluation of a Path-finding Algorithm for Interconnected Local Area Networks,* IEEE J. Select. Areas Commun., vol. SAC-5, No. 9, Dec. 1987.

[9] Tse-yun Feng, *A Survey of Interconnection Networks,* IEEE Computer, Dec. 1981.

[10] Bell Communications Research, *Generic System Requirements in Support of Switched Multi-Megabit Data Service,* Bellcore Technical Advisory, TA-TSY-000772, October, 1989.