Integrated Routing Protocol for Multicast and Anycast Messages

Weijia Jia Gaochao Xu Wei Zhao[†]
Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong
[†]Department of Computer Science, Texas A & M University, College Station, USA
Email: wjia@cs.cityu.edu.hk

Abstract

A novel efficient and dynamic integrated routing protocol for multicast and anycast messages is presented. The contributions of the protocol differ from well-known shared-tree systems in two aspects: (1) Off-tree anycast configuration and routing: multicast sources use anycast routing to select a better path from the source to one router in the group in order to avoid congestion or any fault in the network. (2) On-tree router anycast configurations: The nodes in the shared-tree are formed into a virtual anycast group. The shared-tree approach is extended with capability of a group cores (anycast group). The simulation data demonstrates the efficiency of the protocol.

Keywords: Multicast, Anycast, Dynamic Routing, Flexibility, Shared-tree, Local Router Reconfiguration.

1. Introduction

Efficient Internet multicast communication are critical for applications that require fast and reliable services such as e-commerce and QoS multicast service arising from Information Service Providers.

Anycast has been defined as a standard service and an anycast message should be delivered to one member in a group of designated receivers as stated in the latest version of IPv6 specifications [13]. Using anycast communication services may considerably simplify some applications. For example, it is much easier for a client to find a best server when there are multiple servers for one kind of service in a network. Several anycast application layer protocols have been proposed [1,2].

Many well-known multicast routing protocols and techniques have been developed such as Distance-Vector Multicast Routing Protocol (DVMRP) [16], MOSPF [11, 12], PIM (Protocol Independent Multicast) [7], and CBT [3] and others [8, 15]. There are two popular multicast routing strategies: Source-based tree and Shared-tree routings. The former can achieve the short end-to-end average delay by applying the shortest path routing but does not scale well as it is a ONE-tree/per source routing. If many sources in the network, this approach may result in many source trees and high overhead is introduced to establish the routing tables for the routers. A shared-tree has advantages of scalability as it is a one tree per group approach, however, the traffic may concentrate on some nodes of the singletree trunk when many sources send their multicast packets to the tree center (core) simultaneously. Anycast routing may lead the traffic from different sources through different paths to the members in a group. Furthermore, the traffic from the same source can be transmitted along different paths to the members in the group by using dynamic anycast path selection. Thus, there is potential to improve the performance and reliability of multicast routing using anycast routing techniques. Based on this idea, this paper studies a novel integrated routing protocol for multicast and anycast messages, aiming to achieve the following objectives:

- Efficiency: Short end-to-end average delay should be achieved to satisfy application requirements.
- Scalability: Multicast routing should be scale to a large network without comprising the performance. Ideally, the overhead of the multicast routing protocol should be independent of the number of multicast sources or the size of the network.
- Integration: The multicast and anycast messages can be routed in an integrated way effectively.

The rest of the paper is structured as follows. Section 2 discusses the modeling of delay calculation. Section 3 presents the new routing algorithms that can dynamically respond to link or node alternatives in case of traffic concentrations. Section 4 considers performance analysis and we draw some conclusions in the final section.

2. Performance Notations and Modeling

Network is modeled as a graph $N(V, E)$ where V is a finite set of vertices in N, representing nodes (routers or switches) in the network; E is a finite set of edges, representing the links between the nodes. The participation of a host in a multicast group is reflected via its local router that is responsible for routing multicast/anycast packets and maintaining the group information for all hosts in a local area.

0190-3918/01 $10.00 © 2001 IEEE

Router/node. R is the next hop of another node (say R') if R can receive a packet directly from R' without going through any other router. The key data structure that a router uses for routing is a routing table. An entry in a routing table consists of fields for destination address, next hop and the distance etc. For an incoming packet, the router locates an entry in a routing table such that the destination address of the packet matches the destination address of the entry. The next hop field defines the next hop where the packet should be sent. The distance field contains the value of the total distance of a path that leads to the host with the destination address. In a router, once the next hop of a packet is determined, the packet will be sent to a proper output (out-going) interface where the packet will be transmitted into the associated output link, which, in turn, connects to the next hop. It is possible that a packet may have to be queued at the output interface because the transmission of previous packet(s) has not yet been finished. Obviously, if the network status is changed (e.g., some link fails, some router joins, etc), the routing tables may need to be updated. We say that a router is reconfigured if its routing table is updated (such as by ICMP protocol [6]). A router has a number of input and output interfaces. We also assume that there is a FIFO queue at each output interface[1]. Three quantities are of particular interest in characterizing the performance of multicast routing algorithms (see [5]):

Transmission-Delay (TD) is measured as the maximum time traversed by a packet from a source to all destinations. It is the upper bound of delay a multicast packet experienced in network. We modeled the delays that may be experienced by a packet as the summary of router transmission delay and path delay.

- Router transmission delay d_{ij} is defined as the time needed for router R_i to process the packet and sends it through jth interface (or jth output port). Assume that non-blocking gigabit router is used [14]. Since the packet has to go through the FIFO of a certain output interface, the packet may experience queuing delay. The queuing delay is calculated in terms of specific output interface of router R_i.
- Path delay: To avoid confusion, we use P_{ij} to denote the shortest path between R_i and R_j, i.e., packet transmission with static minimum delay. Routers in the network cooperatively decide a path for a packet and transmit the packet along the path. Formally, P_{ij} denotes a path from R_i to R_j. A sequence of nodes may be listed explicitly in a path. We use terms "route" and "path" interchangeably. A path P_{ij} is defined as the shortest series links transferring a packet from R_i to R_j. Denote P_{ij} as $\{R_i, R_{i+1}, \ldots, R_m, R_{m+1}, \ldots, R_{j-1}, R_j\}$. P_{ij} is a loop-free path if every R in P_{ij} is distinct. Path delay is defined as the sum of transmission delays $d_{i,i+1}, \ldots, d_{m,m+1}, \ldots, d_{j-1,j}$ which is

$$D_{ij} = \sum_{k=i}^{j-1} d_{k,k+1}.$$

Bandwidth-Consumption (BC) is measured as the total number of links used to deliver a packet from a source node to all receiver nodes.

Traffic-Concentration (TC) is measured as the number of packets transmitted across each link per unit time.

[1] In our opinion, this assumption reflects current high speed router such as 50-Gb/s router [14]. For example, if one of its output interface connected to a 100 Mb/s high-speed Ethernet LAN, the router's backplane speed is nearly 500 times faster than the network link speed. Therefore, the packets are most probably queued in an output FIFO.

3. The Algorithms

The major objective of this algorithm is to improve the performance of existing shared tree approaches such as CBT. To achieve efficient and robust multicast routing, our multicast routing protocol targets at selecting the best paths to achieve load balance and short end-to-end delay for reliable multicast packet routing. The routing algorithm takes advantage of anycast routing for path selection, which can do dynamic multiple path selection [10, 17]. In order to transfer a multicast packet to all members in a group, the information of *tree* routing table must be equipped for all routers on the tree.

3.1 Shared tree routing

3.1.1. Tree creation. The group formation procedure is similar to that of CBTv2 [4], aiming at establishing a shared multicast distribution tree that spans only those networks and links leading to interested receivers. To achieve this, a host must express its interest in joining a group by multicasting an IGMP host membership report across its attached link [6]. On receiving this report, a local (CBT) aware router invokes the tree joining process (unless it has already) by generating a JOIN_REQUEST message. The message is sent to the next hop on the path towards the group's core router.

The state created in routers by the sending or receiving of a JOIN_ACK is bi-directional data can flow either way along a tree "branch", and the state is group specific - it consists of the group address and a list of local interfaces over which join messages for the group have previously been acknowledged. In the shared tree, it is necessary to be able to distinguish the upstream interface from any downstream interfaces. These interfaces are known as the "parent" and "child" interfaces, respectively. A router is not considered "on-tree" until it has received a JOIN_ACK for a previously sent JOIN_REQUEST.

3.1.2. Tree maintenance is achieved by having each downstream router periodically send a *"keepalive"* message to its upstream neighbor. The *"keepalive"* mechanism may be implemented by means of ICMP echo-request message. The receipt of a *keepalive* message from a valid downstream router prompts a "response" message. The "response" mechanism also may be implemented by means of ICMP echo-reply message. If no response within a specific timeout, it means the router's upstream neighbor becomes unreachable, the router sends a quit-notification message upstream, and flushes all of its downstream branches by sending flush-tree messages, allowing them to rejoin individually if necessary. In the case that a member leaves the group, if the local router doesn't have any other directly attached members or downstream on-tree routers, the router sends a quit-notification message to its parent router on the tree and deletes the corresponding forwarding cache.

3.1.3. Data transmission. During the data transmission phase, data packets flow from any source to its parent and children. The parent router forwards packets to all the children other than the source and to its parent until data packets reach the core. Data packets are then sent down all the other branches, ensuring that all group members receive them. To accommodate the situation in which a sender is not on the multicast tree, the local router to which the sender is attached encapsulates the data packet and unicasts it to the core; when it reaches the tree, it is decapsulated and disseminated over the tree.

3.1.4. Problems associated with shared tree. We have observed that Bandwidth Consumption (BC) for shared tree routing algorithm is relatively constant because a packet must traverse over all links on the tree anyway in order to reach the destination members. But BC outside the tree can vary. It is apparent that Transmission Delay (TD) for a packet may increase due to Traffic Concentration (TC) on a link.

Initially, assume that network is interference-free, i.e., only the traffic related to group is considered, (we will relax the assumption later). Under such assumption, we can estimate the delay from router R_i to all the members of group G (denoted as $D_{i,G}$) as

$$\begin{aligned} D_{i,G} &= d_{ij} + D_{span}(R_j) \\ &= d_{ij} + max\{D_{jk} \mid R_j \in V(T) \wedge R_j \in P_{ic} \wedge \forall R_k \in V(T)\} \end{aligned} \quad (3\text{-}1)$$

Under this constraint, R_j is a first on-tree router in the path P_{ic} where c is the core. In the following, we use examples to illustrate that CBT routing algorithms may not be efficient in terms of metrics TD, BC and TC. An example is shown in Figure 1 and the labels denote the delay on each link. The member group is a set of nodes $G = \{R_1, R_2, R_3\}$, the shared-tree $T(V(T), E(T))$, here $V(T)=\{R_1, R_2, R_3, R_c\}$, $E(T) = \{(R_1, R_c), (R_2, R_c), (R_3, R_c)\}$, R_c is the core, R_4 is the sender. Using CBT algorithm, it can be seen that $D_{4,G}$ is not the shortest delay for a multicast packet traveling from router R_4 to all the receivers. R_4 sends a packet to R_c via the shortest path through R_3. In terms of formula (3-1), the transmission delay from sender R_4 to all members in G is calculated as $D_{4,G}=d_{4,3}+D_{span}(R_3)=2+6=8$. Obviously there is another path for R_4 to transmit packets with shorter TD, i.e., from R_4 to R_1, and then span over T from R_1, which is $D'_{4,G}=d_{4,1}+D_{span}(R_1)=1+6=7$.

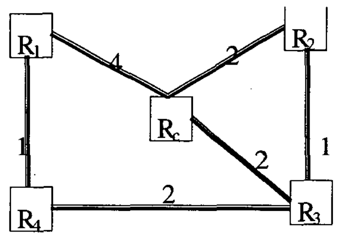

Fig. 1. CBT tree routing. Example 1

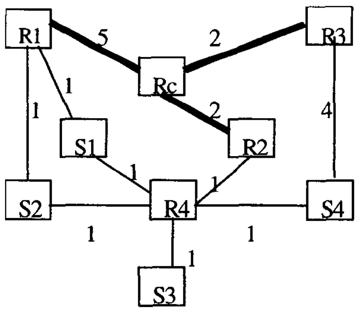

Fig. 2. Example 2

Now consider another example to illustrate Traffic Concentration. Because all packets from different sources will be transmitted towards the core according to CBT algorithm [3], traffic congestion may occur when many packets from different sources were transmitted along the same link.

See Fig. 2 for the example. Assume the multicast group is $G = \{R_1, R_2, R_3\}$, the shared-tree is $V(T) = \{R_1, R_2, R_3, R_c\}$ and $E(T) = \{(R_1, R_c), (R_2, R_c), (R_3, R_c)\}$ where R_c is the core. Nodes S_1, S_2, S_3, S_4 are multicast packet sources. In the CBT algorithm, $\{S_1, S_2, S_3, S_4\}$ will transmit their traffic to R_4 along the edge (R_4, R_2). If the bandwidth of this edge is relatively lower, the traffic congestion may occur. As discussed above, if both S_1 and S_2 transmit their multicast packets to R_1, the delay can be reduced and the traffic can be split along different paths. Like the previous example, using CBT routing, the maximum packet delay from S_1 and S_2 via R_4 is 9 but if the packets are transmitted via R_1, the total delay is 8.

Bandwidth Consumption: CBT routing may suffer a problem of extra bandwidth consumption. We still use Figure 2 as an example. If the multicast packets are forwarded from sources S_1 and S_2 may have to pass 5 links but if they go through R_1, only 4 links are used.

3.2 Algorithm using anycast routing

From the examples, we observe that the worse TD, BC and TC in CBT are due to the fact that the source nodes of multicast packets outside the shared-tree will choose the core for transmission of their multicast packets. Although some researchers try to alleviate this problem by selecting more cores [5], similar problems still remain.

In this section, more efficient algorithm is discussed for solving the problems. The algorithm aims at adaptively choosing appropriate on-tree routers for the multicast routing so as to balance the traffic load and to utilize less bandwidth. In the algorithm, sources are not restricted to forward their packets only to the core, instead, we allow each source independently to transmit their packets to ANY ONE router in $V(T)$ which results in shorter delay than that of CBT. The algorithm intends to achieve the minimum total delay among all the possibilities for a packet originated from source R_i to the tree, which is:

$$\begin{aligned} min\{D_{i,G}\} &= min\{d_{ij}+D_{span}(R_j)\} \\ &= min\{d_{ij}+max\{D_{jk}: R_j \in V(T) \wedge \forall R_k \in V(T)\}\} \quad (3\text{-}2) \end{aligned}$$

Note that in general, to enable more routing possibilities will introduce more cost over the routers in network. It is prohibitively expensive for the algorithm to test all the routes for each individual source since the dynamic distribution of sources cannot be predicted. To realize (3-2), i.e., to implement the adaptive any-one routing semantics, in the following sub-sections, we describe a novel adaptive and dynamic routing algorithm called *Anycast Routing* based on *CBT* (*Anycbt*).

3.2.1. On-tree Anycast Group Establishment. When the shared-tree is built, all on-tree routers (including the core) in $V(T)$ are selected to form an anycast group with anycast address G_A to replace the role of the core. G_A can be advertised to the network N (by broadcast). The format of anycast address may be considered as some "temporary" anycast address to denote the on-tree routers as long as G exists. Senders (hosts) outside G, whose attaching routers are not the on-tree routers, may assign G_A as an interface entry in the routing table and the routing table can be configured with $<G_A, G>$.

3.2.2. On-tree Router Span Delay Calculation. For any router R_i in $V(T)$, the longest path delay from itself to the farthest router in tree T is its span delay. For example, for a router $R_i \in V(T)$. The spanning delay $D_{span}(R_i)$ is

$$D_{span}(R_i) = max\{D_{ij}: R_j \in V(T)\} \qquad (3\text{-}3)$$

Consider all the routers in G. The span delay is denoted as $D_{span}(G) = \{D_{span}(R_i) | \forall R_i \in V(T)\}$ which is required by all off-tree routers related to G. To propagate this information efficiently, when each router in the shared-tree calculates its own span delay, the individual delay may be accumulated by the core and the core broadcasts $D_{span}(G)$ to the overall network once to reduce the number of flooding messages. Note that $D_{span}(G)$ is a set of delays, not just a single value. It is true that there is a non-trivial amount of data to send around, which scales with the number of routers in the multicast group. It also needs to be updated occasionally based on load or topology changes, or dynamic joins or leaves. This overhead is also considered by MOSPF [12].

3.2.3. Off-tree Router Anycast Group Configuration. Upon the reception of the spanning delays $D_{span}(G)$ flooded from the core, the off-tree routers that are interested in sending multicast packets to G will install an anycast routing table according to $D_{span}(G)$. The anycast routing table enables the routers to dynamically select an optimal path to reach the shared-tree among multiple paths even in the presence of link/next hop failure. The configuration procedure works as follows for an off-tree router R:

1. For each output interface j of R, $(1 \le j \le k)$, denoted as I_j, find the shortest path to group G_A. Assume that R_i is the first on-tree router in the path $P(I_j, R_i)$. Denote the delay for the path $P(I_j, R_i)$ as $D(I_j, R_i)$. The delay to multicast a packet starting from interface I_j (denoted as $D(I_j)$) till it is received by all destinations in G is calculated as $(1 \le j \le k)$:

$$D(I_j) = min\{D(I_j, R_i)+D_{span}(R_i): \forall R_i \in V(T)\} \; (3\text{-}4)$$

If there exists more than one such shortest paths then we select the on-tree router R_k with the smallest $D(I_j, R_k)$, otherwise the choice is arbitrary.

2. Once all $D(I_j)$s are computed, the next task is to establish the anycast routing table. Without losing generality, assume that there are k outgoing interfaces in an off-tree router R, each leading to an on-tree router and the corresponding values of $D(I_j)$ are different. The routing table is arranged in the order of $D(I_1)<D(I_2)<\ldots< D(I_k)$. Each row in the table is in the form of

Destination	Next hop	Delay	Dynamic Cost (option)	Connection Available (option)
a.b.c	*x.y.z1*	$D(I_1)$	C_1	OK_1
a.b.c	*x.y.z2*	$D(I_2)$	C_2	OK_2
...	...	...	...	...

Where OK_j is a Boolean control variable for fault-tolerance. If the path (channel) through interface I_j is connected then OK_j=1, otherwise, OK_j=0. C_j is the threshold of FIFO for interface I_j in R to decide if an incoming anycast packet should be forwarded through interface I_j or choose the next available interface. They are discussed in turn as below:

- Management of OK_j (initialized 1): our protocol applies the same fault-detection techniques as the current unicast routing protocol, i.e., R exchanges an "alive" message regularly with the next hop for a specific time interval. If R does not hear any response from the next hop on timeout, then OK_j=0. Otherwise OK_j=1.
- Set-up of C_j: Setting up of C_j is crucial for the dynamic anycast routing. We use a simple heuristic method for easy calculation of dynamic delay. For a multicast packet m_a at R, if R transmits m_A through interface I_j, the following inequality should hold (μ_j is the processing speed of interface I_j). Let Q_j be the queue attached with interface I_j and $|Q_j|$ is the number of packets in Q_j. Interface I_j is chosen to forward m_a if

$$D(I_j) + |Q_j|/\mu_j < D(I_{j+1}) + |Q_{j+1}|/\mu_{j+1} \quad (3\text{-}5)$$

i.e., the total delay of expected multicast transmission through interface j is shorter than the total delay through the next available interface j+1 where μ_{j+1} is the processing speed of interface j+1. To decide the threshold for the length of Q_i that a packet should be transmitted through next interface, assume that the next interface is idle, i.e., $|Q_{j+1}| = 0$. Thus we have the condition that R uses interface j for transmission of m_a if and only if

$$D(I_j) + |Q_j|/\mu_j < D(I_{j+1}) \quad (3\text{-}6)$$

Thus R chooses $(j+1)$th interface when

$$|Q_j| \geq \lfloor \mu_j(D(I_{j+1}) - D(I_j)) \rfloor \quad (3\text{-}7)$$

Considering hardware restriction for the maximum length of queue Q_j (denoted as *max_length_Q_j*), the value of threshold for the queue length is defined as

$$C_j = \min\{max\_length\_Q_j, \lfloor \mu_j(D(I_{j+1}) - D(I_j)) \rfloor\} (3\text{-}8)$$

Note that the anycast routing table is set on top of unicast routing mechanism. In our protocol, we just propose that the anycast address is available. Setting up of C_i and OK_i aims at achieving dynamic routing, taking into the dynamic traffic into considerations. The two items are considered optional as they may be omitted in case high overhead is required for maintaining the information in case of frequent change of network topology.

3.3. Dynamic Routing Algorithms

With the anycast routing tables, the dynamic multicast routing algorithm can be described below. Note that in the following algorithm, the routers cooperatively route the multicast messages by selecting anycast routing dynamically and we differentiate the packets originated from the nodes in *G* and that outside *G*.

Alg-1. Multicast packet m originating from a node in *G*:

1. When an on-tree router receives a packet m destined to group G from an attached sender host, it adds a multicast-header to m and forwards copies of m to all interfaces according to the routing table *<G, input-interface, output-interface>* triple except the incoming interface for m.
2. Any other on-tree routers, upon reception of *m*, read the multicast-header group id G, and forward m to the interface connecting to the next hop which matches the triple *<G, input-interface, output-interface>*. If the routers have hosts attached that are the members of *G*, they strip the multicast-header off m and transmit it to these hosts for delivery.

Alg-2. Multicast packet m originated from the node outside *G*:

1. If a host is not a member in *G*, but its attached router R is an on-tree router i.e. $R \in V(T)$, the routing algorithm is the same as that of Alg-1;
2. If the attached router R is an off-tree router, i.e., $R \notin V(T)$, upon reception of *m*, *R* adds an anycast header (G_A, G) to m, makes it into anycast packet m_A and executes the following dynamic anycast routing procedure:

```
for j:=1 to $K_R$ do
  // $K_R$ is the number of interfaces of R.
    if $OK_j$ & ($|Q_j| < C_j$) then
    begin
        Transmit $m_A$ to next hop via Interface-j;
          exit; //procedure terminates
     end;
```

3. If an off-tree router receives an anycast packet m_A, it reads the header G_A and uses the same routing approach as Step 2.
4. When an on-tree router receives an anycast packet, it strips the anycast header and adds a multicast-header to the packet destination group G and sends the packet copies to all output interfaces in terms of <*G, input-interface, output-interface*> triple. The rest steps are the same as Alg-1.

4. Performance Evaluation

4.1. Simulation Model

In this section, we will report performance results of our protocol introduced in this paper. To obtain the performance data, we use a discrete event simulation model to simulate data communication networks. The simulation program is written in C programming language and runs in a SUN SPARC-20 workstation. A twenty-one-nodes ARPANET is simulated as shown in Figure 3.

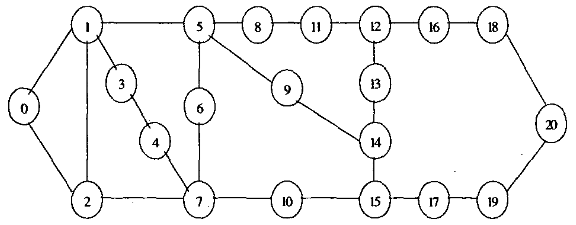

Fig. 3. ARPANET

In the network, assume that every router is attached by a sub-network, which in turn connects to hosts and all the links with uniform bandwidth. For simplification of the simulation, the bandwidth and distance between any two adjacent nodes are assumed to be uniform with one unit. During the simulation, 100 groups are randomly generated with average of 5 members for each group and 2 million multicast packets are randomly generated as a Poisson process and the average size of each packet is about several tens of Kbytes (20-30 Kb). In fact, the number of members affects the performance of the system. But in this section, we intend to analyze the performance between anycast to CBT and pure CBT in fault-free cases, therefore, the group size and distribution are assumed the same for execution of both systems and they are not particularly taken into account in the simulations. We are interested in the following performance metrics for the off-tree routers and links:

- Mean max delay: The delay of a packet at a router is defined as the summation of the routing delay, the queuing delay, and the transmission delay. The delay of a packet is the sum of the delays at all the routers through which the packet passes. The max delay of a multicast packet is the delay from its source to the last multicast group member reached. Decreasing the max delay is important for multicast communication. In a simulation session, the mean max delay is computed by dividing the sum of all packets' max delay by the number of packets all sources sent.
- Mean standard deviation of all off-tree-link utilization: In the simulation, suppose the total simulation time is T, and the busy time of link L_i is t_i in the period of T, then the utilization of L_i is defined as $\partial_i = t_i/T$. For links $L_1, L_2, \ldots, L_k$, suppose the set of utilization is denoted as $\partial_1, \partial_2, \ldots, \partial_k$. If $\Phi = (\Sigma^k_{i=1}\partial_i)/K$, then $\sqrt{(\sum_{i=1}^{k}(\partial_i - \Phi)^2)/K}$ is the standard deviation of link utilization. The lower the standard deviation is, the more balanced the loads of links are and the better capability to prevent the congestion that the system has. Because the multicast packets must pass all links on tree, so we only consider off-tree links. We will simulate integrated protocol (represented by "*anycbt*") and the original fault-free CBT protocol (cbt/nf).

4.2. Performance Observations

The bandwidth of each link is uniformly measured as 10Mbps. For each simulation, 2 millions of multicast packets are randomly generated as a Poisson process and the average size of each packet is about several tens of Kbytes (20-30 Kb). Simulation starts when the first multicast packet is generated and ends when all the packets have reached their destinations. We collected the statistics data when the systems were stable. The average delay is calculated by taking the multicasts from five randomly generated sources and a four members group.

Our multicast protocol has higher capability to prevent the congestion and to balance the traffic loads. Figure 4 gives the runtime comparison of the standard deviations of queue length under the cbt/nf and anycbt. Figure 5 shows the average delay changes with the arrival rate under *cbt/nf* and *anycbt*. Fig. 6 gives the system throughput changes with the arrival rate under *cbt/nf* and *anycbt*. From these data, we have the following observations:

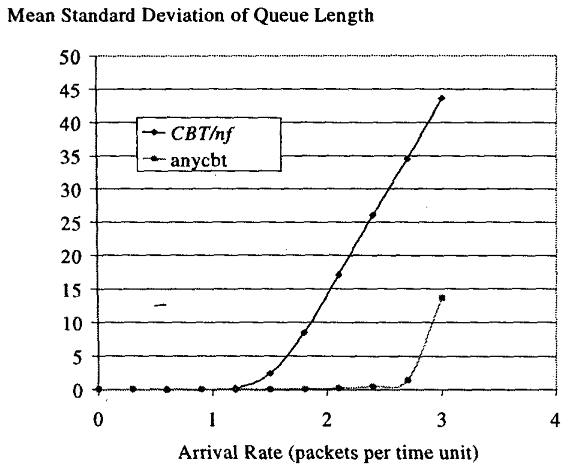

Fig. 4. Mean standard deviation of queue length comparison

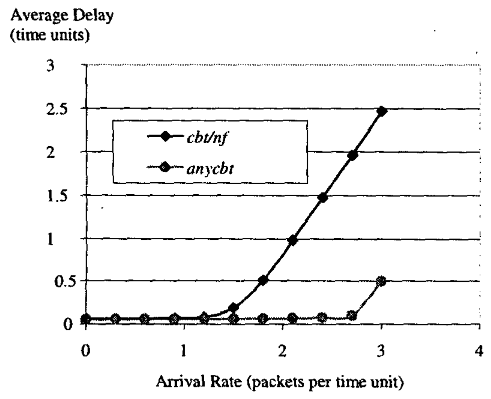

Fig. 5. Average delay comparison.

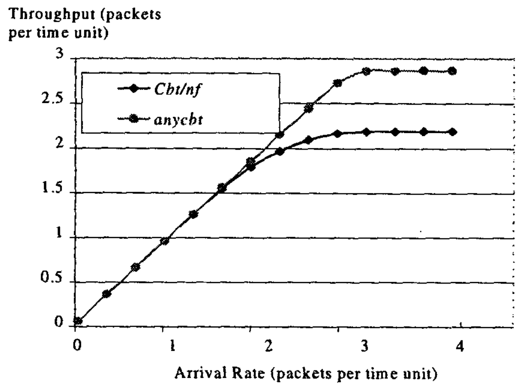

Fig. 6 Throughput comparison

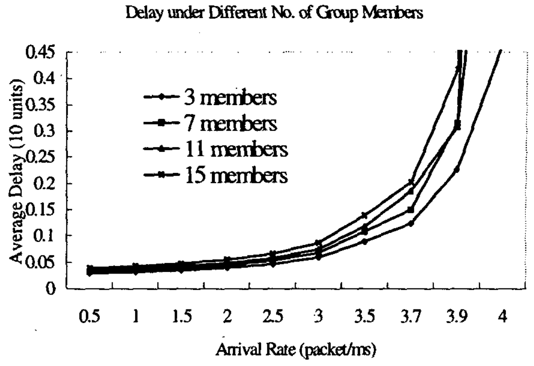

Fig. 7. Average delay comparison

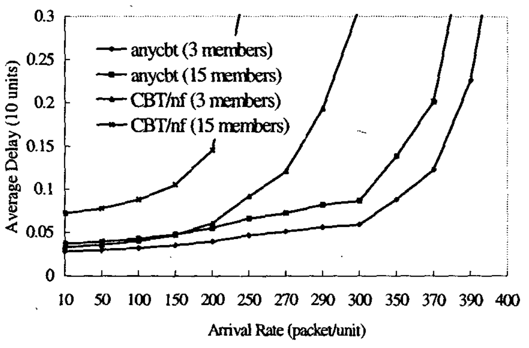

Fig. 8. Delay difference

1. Fig. 4 shows that the standard deviation of queue length is much lower under the *anycbt* than under the *cbt/nf*. It shows that the *anycbt* has the much higher capability to prevent the congestion and to balance the traffic loads than CBT, especial in the heavy traffic situation.
2. From Fig. 5 we can observe that as the arrival rate increases, the *cbt/nf* suffers traffic congestion problems and causes much longer delay. In *anycbt* approach, the traffic may be split dynamically; therefore, the delay is much shorter as compared with *cbt/nf* under the same data rate situation.
3. From Fig. 6, it can be seen that as the traffic load is higher, *anycbt* approach gets less congested and more balanced than that of *cbt/nf*. The *anycbt* achieves higher throughput than *cbt/nf* does. The figure shows that the throughput (saturation point) for *anycbt* is 2.8, which outperformed that of *cbt/nf* (which is 2.1).

4. From Fig. 7, we have tested the scalability of our algorithm and found that *anycbt* is quite scalable in terms of average delay and different size of multicast group.
5. Fig. 8 demonstrates that under the heavy loading situations in which every node is the message source and continues to about 200,000 packets. The approach *anycbt* achieves much better throughput and short average delay. It can be observed that our algorithm is able to achieve about 50% performance improvement in terms of throughput and packet delivery delay. The worse delay in CBT routing is due to the congestions near the core of the CBT tree.

5. Conclusions

We have proposed and analyzed a new dynamic integrated multicast/anycast protocol. Our protocol consists of two parts: router configurations and packet transmission. The algorithm uses an improved version of the original CBT protocol. While maintaining the same level of scalability, our improved CBT protocol has much better performance relative to CBT based algorithms because we have used anycast routing technology.

Several extensions are possible. By integration of dynamic anycast and multicast protocols, the anycast protocol delivers a packet to any one of the members in a recipient group. Transmitting a multicast packet from its source to the shared multicast tree can be considered as an anycast communication where the recipient group is made of all the nodes on the tree. We should study how to apply them here. Our protocol can also be extended to the applications where the messages have both dynamic and real-time requirements. The key issue here is to model the traffic on the shared multicast tree so that a delay bound can be derived.

6. Acknowledgments

This work was partially sponsored by the Strategic Grants City University of Hong Kong under nos. 7001189 and 7001060 and RGC Hong Kong Grant no. 9040352.

7. References

[1] S. Bhattacharjee, M. H. Ammar, E. W. Zegura, V. Shah, and Z. Fei, Application-Layer Anycasting, Proc. of IEEE INFOCOM'97, April 1997.

[2] E. Basturk, R. Engel, R. Haas, V. Peris and D. Saha, Using network layer anycast for load distribution in the Internet, IBM Research Rep., T.J. Watson, RC 20938.

[3] A. J. Ballardie, P. F. Francis and J. Crowcroft, Core based trees, Proc. ACM SIGCOMM, San Francisco, 1993, pp.85-95.

[4] A. Ballardie. Core Based Trees (CBT & CBT version 2) Multicast Routing Architecture. Sept. 1997, RFC2189.

[5] K. L. Calvert, E. W. Zegura and M. J. Donahoo, "Core Selection Methods for Multicast Routing", Proc. of ICCCN'95, Sept. 1995, pp. 638-642.

[6] S. Deering. IGMP Router Discovery Messages. RFC 1257, Sept., 1991.

[7] S. Deering, D. L. Estrin, D. Farinacci, Van Jacobson, C. Liu, and L. Wei. The PIM Architecture for Wide-Area Multicast Routing, IEEE/ACM Transactions on Networking, Vol. 4, No. 2, April 1996, pp. 153-162.

[8] C. Huitema, Routing in the Internet, Prentice-Hall, Inc., New Jersey, 1995.

[9] W. Jia, W. Zhao, D. Xuan and G. Xu "An Efficient Fault-Tolerant Multicast Routing Protocol with Core-Based Tree Techniques", IEEE Trans. on Parallel and Distributed Systems, 10(10), October 1999, pp.984-999.

[10] W. Jia, D. Xuan, W. Zhao, "Integrated Routing Algorithms for Anycast Messages" IEEE Communications Magazine, January 2000, pp.2-12.

[11] J. Moy. Multicast Extensions to OSPF. RFC 1584, March 1994.

[12] J, Moy, OSPF: Anatomy of an Internet Routing Protocol, Addison-Wesley, Reading Mass.1998.

[13] C. Partridge, T. Mendez, and W. Milliken. Host Anycasting Service. RFC1546, November 1993.

[14] C. Partridge et al., A 50-Gb/s IP Router, IEEE/ACM Transactions on Networking, Vol. 6, No. 3, June 1998, pp.237-248.

[15] M. Parsa and J. J. Garcia-Luna-Aceves. Scalable Internet Multicast Routing, Proceedings of ICCCN'95, pp. 162-166, September 1995.

[16] D. Waitzman, C. Partridge, and S. Deering, Distance Vector Multicast Routing Protocol, RFC 1075, 1988.

[17] D. Xuan, W. Jia, and W. Zhao, and H. Zhu, "Routing Protocols for Anycast Messages", IEEE Trans. on Parallel & Distributed Systems, IEEE Computer Society Press, 11(6), June 2000, pp571-588.