Computing and Informatics, Vol. 30, 2011, 335–355

A CLUSTERING SCHEME IN APPLICATION LAYER MULTICAST

Xinchang Zhang

Shandong Key Laboratory of Computer Networks
Shandong Computer Science Center
No. 19, Keyuan Road, Lixia Distric
Jinan 250014, P.R. China
e-mail: xinczhang@hotmail.com

Xiaodong Li, Wanming Luo, Baoping Yan

Computer Network Information Center
Chinese Academy of Sciences
4, South 4th Street, Zhongguancun
Beijing 100049, P.R. China
e-mail: {lee, luowanming}@cnnic.cn, ypb@cnic.cn

Abstract. In application layer multicast (ALM), member hosts lack direct knowledge of underlying network topology, which brings some performance penalty. This paper investigates an effective way to rapidly obtain some related topology knowledge, i.e. getting topology hints from existing IP registered resources – WHOIS database. We further propose a clustering scheme, which can be integrated into the existing ALM solutions. Our proposed scheme can cluster some nearby member hosts no matter when these hosts join the group. Therefore the scheme also solves the join sequences problem in some degree. We also present an application framework of the clustering scheme, and give an application example named HMTP-E that integrates the scheme into HMTP protocol. The experiment results show that the clustering scheme plays a positive role on improving the performance of existing ALM solutions.

Keywords: Multicast, application layer multicast, WHOIS searching, network topology, IP address

Mathematics Subject Classification 2000: 94A11, 94C99

1 INTRODUCTION

In group communication, multicast is the most efficient approach because it can save much bandwidth and greatly reduce the load of servers. Multicast functionality was originally implemented at the IP layer. However, IP multicast has not been deployed widely. C. Diot et al. [1] explain the main reasons for the non-ubiquitous deployment, such as dependence on the supports of network infrastructures and rapid resource-consuming of routers. As an alternative of IP Multicast, application layer multicast implements multicast functionality at the application layer instead of the IP layer. Therefore, application layer multicast approach needs no additional modification of the network infrastructure, and accelerates the deployment of multicast applications. Since member hosts only duplicate and forward the packets in end-networks and have no direct knowledge of underlying network topology, ALM has some unavoidable performance penalty.

Clearly, application layer multicast solutions can benefit from some knowledge of underlying network topology. With the knowledge, close-by nodes can be clustered. Clustering nearby nodes can (1) localize transmission of some member hosts, which reduces the number of data packets in backbone links, and (2) localize the recovery operations when some member hosts fail or leave. To some extent, some proposed positioning approaches (e.g. Global Network Positioning (GNP) [18], binning scheme [19] and PIC [20]) can provide the information on relative proximities among the hosts. However, by these approaches, it is difficult for a host to get the on-demand topology information quickly.

This paper investigates an approach to get some topology information from registered information on IP addresses. Through analyzing the real data of WHOIS searching, we validate the feasibility of the approach. Based on the topology hints, we propose a clustering scheme. The scheme can position a newcomer and form topology-aware clusters quickly. In these clusters, the hosts are close to each other, which improves the multicast performance.

In this paper, we further present an application framework of our proposed clustering scheme, which integrates the clustering scheme into some existing ALM protocols. In the integrated system, some nearby member hosts are clustered no matter when these hosts enter the system. Consequently, the scheme also solves the join sequences problem in some degree. In other words, the scheme can build good multicast trees when members join the group in different orders. As an application example, we integrated the scheme into an existing ALM protocol – HMTP ([3]). The experiment results show that our proposed scheme and application framework can enhance the performance of HMTP. We also believe that the clustering scheme can take on a positive role on most of existing ALM protocols.

The rest of the paper is organized as follows. In Section 2, we discuss some existing application layer multicast solutions and positioning approaches. Section 3 investigates the feasibility of getting some topology hints from registered information on IP addresses. A detailed clustering scheme and its application framework are proposed in Section 4. Section 4 also gives an application example (HMTP-E)

of our proposed clustering scheme. We evaluate the performance of HMTP-E by analyzing the simulation results in Section 5. Finally, we summarize this paper in Section 6.

2 RELATED WORK

In recent years, many application layer (or overlay) multicast solutions have been proposed, e.g. HMTP, NICE [4], NARADA [5], TBCP [6], ALMI [7], Scribe [8], Hostcast [9], Overcast [10], ZIGZAG [11], SAH [12], OMNI [13], TOMA [14] and PALM [15]. In [2], depending on the sequence of constructing overlays and building delivery trees, different proposed application layer multicast solutions are classified into three approaches, i.e. mesh-first approach (e.g. NARADA), tree-first approach (e.g. ALMI and HMTP), and implicit approach(e.g. NICE and Scribe).

Some protocols explicitly use clustering strategies to improve the multicast performance, e.g. NICE and ZIGZAG. NICE organizes the overlay into a hierarchy of clusters, and forms the multicast tree based on the hierarchy. The size of each cluster is between k and $3k-1$, which confines the scale of a cluster. Each NICE cluster has a head, which is the center of this cluster in the ideal situation. In NICE, cluster heads at the same level form the next level clusters initiating from the lowest level, until there is only a single cluster at the highest level. In ZIGZAG, the administrative organization represents the logical relationship among the members. ZIGZAG also uses a multi-layer hierarchy of clusters:

1. layer 0 contains all members (peers), and
2. members in layer $j < H-1$ are partitioned into clusters of sizes in $[k, 3k]$.

The height of the ZIGZAG multicast tree is at most $2log_kN+1$, and the worst-case control overhead of a node is $O(klog_kN)$, where N is the number of members. In HMTP, a candidate parent P becomes the parent of a newcomer if it (1) is closer to the newcomer than its existing children and (2) can accept one more child. Therefore HMTP also clusters nearby nodes.

In application layer multicast solutions, positioning a newcomer quickly and accurately is a prerequisite of organizing overlay networks and building delivery trees. Many ALM protocols (e.g. NICE, HMTP and Hostcast) use distributed depth-first searching (DFS) approach to position a host (seen in [16]). In DFS, a newcomer searches down the multicast tree by exploring the branches of some existing nodes, as conventional depth-first traversal does. The protocol based on DFS uses some search criterion to select the appropriate branch in the traversal. A typical criterion is to choose the nearest branch as the candidate parent. Since DFS searching approach is a progressive procedure, existing nodes have a heavy influence on positioning the newcomer.

As mentioned above, there are some positioning approaches that are implemented at the application layer. TAG [17] uses a topology-aware approach to position the hosts. In TAG, underlying network topology information can be obtained

from traceroute or OSPF/BGP routing table dumps. Additionally, there are some positioning solutions based on network coordinates (NCs), e.g. Global Network Positioning (GNP), binning scheme and PIC. With the help of some landmarks, NC-based approaches can position a host accurately. However, it is difficult for these positioning approaches to provide on-demand topology information quickly. Another worry is that the landmarks might become the bottleneck of ubiquitous deployment of application layer multicast.

3 TOPOLOGY HINTS OF REGISTERED RESOURCES

3.1 Topology Information in WHOIS Database

Currently, there are five Regional Internet Registries (RIRs) in the world, i.e. RIPE NCC, APNIC, ARIN, AfriNIC and LACNIC, which oversee the allocation and registration of Internet number resources (incluing IP address) in the particular regions in the world, respectively. Each RIR maintains a WHOIS database which records some information on IP addresses, autonomous system numbers, organizations or customers that are associated with registered resources. In addition, each RIR provides the WHOIS searching service for finding contact and registration information on these resources. In APNIC, there are five National Internet Registries (NIRs). The NIRs perform analogous functions to APNIC at a national level. WHOIS database provides much information on registered resources, which could be useful to enhance the performance of some network applications. In this paper, we will analyze the potential topology hints of the registered information on IP addresses, and propose a scheme to make use of these hints in application layer multicast.

An inetnum[1] object in WHOIS databases, of APNIC, AfriNIC and RIPE NCC, contains details of an allocation or assignment of IPv4 address space. The inet6num object plays the similar role in the IPv6 context. In this paper, we only discuss the inetnum object, because the attributes of the inet6num object are essentially similar to those of the inetnum object. In each database of these RIRs, the inetnum object contains some mandatory attributes, e.g. inetnum, netname, descr, admin-c and mnt-by. Additionally, inetnum also contains some optional attributes, such as mnt-lower and remarks.

In the databases of ARIN and LACNIC, an IP address space is also associated with a city attribute. Clearly, the city attribute shows some topology information at a city level, which is very helpful for application layer multicast solutions to organize overlay network and position the hosts. Unfortunately, WHOIS databases in APNIC, AfriNIC and RIPE NCC have no city attribute. Similarly, the country attribute, a mandatory attribute in each database of five RIRs, also gives coarse-grained positioning information. The netname attribute means the name of a range of IP address space. Netnames are associated with some ISPs or local networks

[1] We use the term inetnum as APNIC WHOIS database defines ([21]).

in most cases. For example, CNINFO-CN means Shenzhen Great Trend Securities Computer Information Company.

3.2 Netname Attribute of Inetnum Object

The netname attribute provides some topology information, because it is closely related to an ISP or local network in most cases. On the whole, clustering member hosts in terms of netnames can improve multicast performance. However, it is impractical to form initial clusters completely and directly according to the original netname information. First, clustering the hosts of the same netname brings heavy computation (compare operation) burden, because the number of netnames is very large. For example, there are more than 286,000 netnames in WHOIS database of APNIC. Second, the IP address ranges of most netnames are less than the size of 1C. Thus it is hard for a group member to find other members that have the same netname. Third, the IP address spaces of some netnames might span wide-across areas, which makes the above clustering approach ineffective or even harmful. Finally, there might be little netname information that is stale, useless even wrong. We will explore a scheme to address the above problems in this section and in Section 4.

In this paper, we introduce a location attribute. The location attribute means the city attribute in the databases of ARIN and LACNIC, but indicates the country attribute in the databases of other RIRs. Furthermore we use the couple ⟨netname, location⟩ (called netname-location couple) to mean the registered entity which binds netname to location. Since clustering in terms of netnames is infeasible, we define a new entity called allounit to contain an appropriate IP address allocation space. An allounit entity contains an address space whose size is equal to/larger than a min size threshold (called minsize). When a netname-location couple contains an address space whose size is not less than minsize, we say the couple is an allounit. In addition, we unite the netname-location couples with small address spaces to form some allounits by the following rules:

- If WHOIS database has city attribute, each city C forms an allounit $\langle \cdot, C \rangle$, which includes the $\langle *, C \rangle$ couples that have an address space of less than minsize. Note that symbol * means any netname.
- If WHOIS database has no city attribute, the netname-location couples, whose space size are less than minsize, are merged into some allounits. Each of these allounits is identified by ⟨M, location⟩, where M means the corresponding mnt-lower or mnt-by value (if mnt-lower is null).

We investigated the WHOIS database of APNIC according to the above approach, and discarded the data of some unrelated netnames such as IANA-BLOCK and APNIC-AP. Figures 1 and 2 show the distributions of the allounits in the APNIC area and CN area (where the country attribute is CN), respectively. Note that the two figures filter the allounits whose sizes are less than the minsize, which account for negligible part of whole space that we investigated. In each figure, the

label of horizontal axis represents the range of IP address, e.g. $8-9$ means the address space ranged from 2^8 to 2^9. To a given label $a-b$ of horizontal axis, the vertical axis indicates the allounit address space ratio (called ASR) defined as

$$ASR = \sum_{u \in \Gamma(a,b)} u \cdot s(u)) / \sum_{v \in \Gamma(0,Max)} u \cdot s(u), \tag{1}$$

where $\Gamma(a,b) = \{w | a < w \leq b, a, b, w \in I\}$, Max represents the max address space size of the allounits, and $s(u)$ means the number of the allounits that contain a given address space of u.

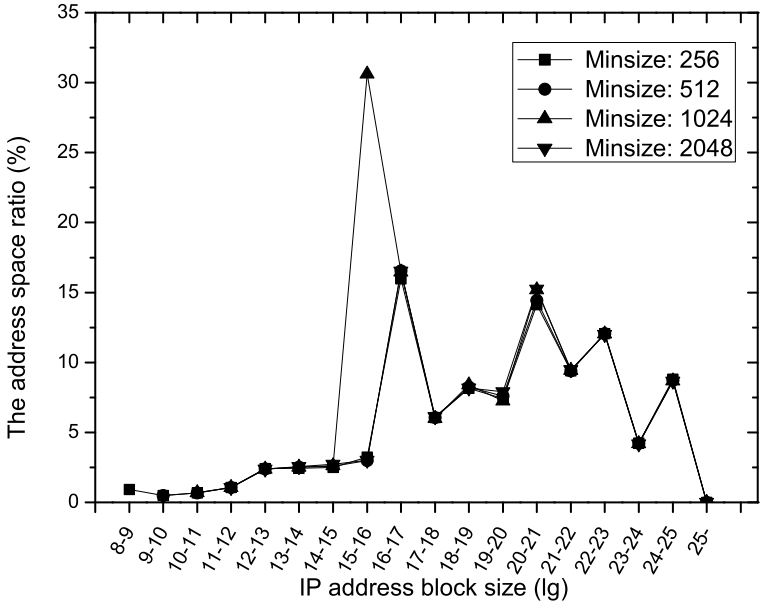

Fig. 1. Allounits in APNIC area

In this investigation, we used four *minsize* values, i.e. 256, 512, 1 024 and 2 048. From Figure 1 we can observe that 1 024 is the best threshold among the four values, because it filters the least addresses. We also see that the allounits (about 1 100 allounits), whose address space sizes are not less than 2^{16} (1 B), contain about 85 percent of the whole sample address space. As a result, most hosts belong to these big-sized allounits. Additionally, there are 15 percent of the whole address space which belong to many small-sized allounits. The probability of containing multiple group members in a small-sized allounit is relatively small, but the members can be clustered well in such allounit. Furthermore, the total number (as Table 1 shows) of

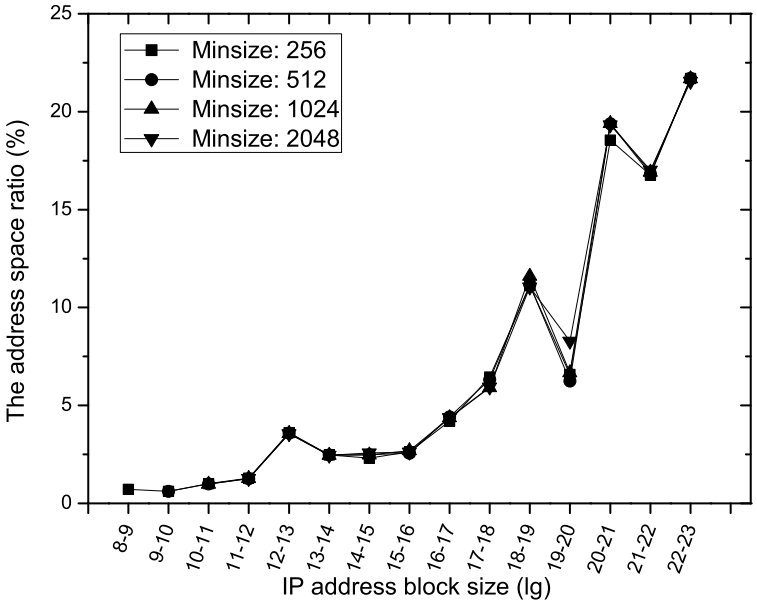

Fig. 2. Allounits with country attribute CN

allounits is small enough for the allounits to be retrieved quickly. We also studied registered resources whose country attribute is CN, and have similar results, as Figure 2 depicts. In addition, we find that most allounits are (geography-divided parts of) ISPs or local networks.

sample	minsize	allounitnum	sample	minsize	allounitnum
APNIC	256	22 724	CN	256	8 762
APNIC	512	11 139	CN	512	5 272
APNIC	1 024	8 205	CN	1 024	3 829
APNIC	2 048	6 218	CN	2 048	2 671

Table 1. The number of allounits (allounitnum) in two sample spaces

With allounit entities, the first two problems noted previously are addressed. In the next section, we will propose a clustering scheme, which can solve the last two problems.

4 CLUSTERING SCHEME AND APPLICATION FRAMEWORK

As noted previously, the allounits provide some topology hints, which are useful for an ALM solution to position the hosts and build topology-aware clusters. However, the heuristics must address the last two problems mentioned in Section 3.2. In this section, we will propose an application layer multicast clustering scheme based on the topology hints of IP address registered resources, to solve the two problems.

4.1 Clustering Scheme

In our proposed scheme, we use the term IPInfo to mean some structurized information on a given IP address, which consists of five parts—netname, city, country, mnt-by and mnt-lower. Note that the city, mnt-by and mnt-lower parts could be null because of different formats in RIRs. Each host stores its IPInfo, or timely gets the corresponding IPInfo by WHOIS searching. We use allounit table (called AT table) as a guideline of positioning hosts and forming clusters. An item of AT table consists of five fields—type, netname, location, mnt and address fields, as Table 2 describes. The type field means the type of the allounit that an item includes. Specifically, the value of type is assigned as follows:

1. the value is 0 if the allounit derives from the couple ⟨netname, location⟩,
2. the value is 1 if the allounit derives from the couple ⟨mnt, location⟩, where mnt means mnt-by or mnt-lower attribute, and
3. the value is 2 if the allounit means the couple ⟨·, city⟩.

The location field means city, or country if city value cannot be obtained. The address field memorizes one IP address which belongs to the corresponding allounit. We introduce a term home host to mean the host that the IP address in address field identifies. In this paper, we select 1024 as the minsize threshold. Thus the size of AT table is small enough for the table to be deployed at a common server or host (called AT server). An AT server provides retrieve service for a given multicast group. Additionally, in our scheme, there is a public-oriented function module (called transserver) that translates the IPInfos into candidate AT items in terms of the rule noted above. The format of the candidate AT item is coincident with that of the item in AT table, and the two items both potentially contain the allounit names. Note that the filtered IP addresses have no IPInfos that can be translated into valid candidate AT items. In this paper, we do not care about the implementation details of the above translation.

In our proposed scheme, each member host belongs to some cluster. Each cluster has a head, and if so, one or multiple cluster members, as Figure 3 shows. For example, cluster A1 has one head and four cluster members. The cluster head receives the packets from another cluster head or the multicast source, and forwards duplicated packets to its children. A cluster member receives the packets from the head or another member in its cluster.

type	netname	location	mnt	address
0	CNCGROUP-FJ	CN	–	···
1		CN	MAINT-xxx	···
2		FR	–	···
···	···	···	···	···

Table 2. The basic structure of AT table

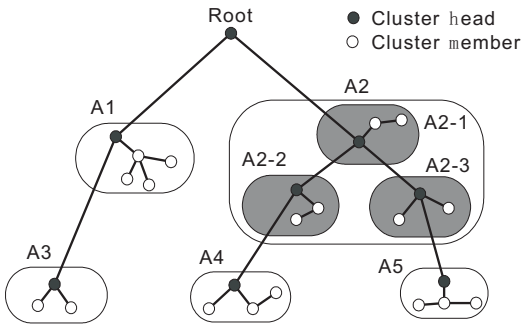

Fig. 3. Structure of our proposed clustering scheme

Similar to TBCP, we also use the term fanout to mean the maximum number of children one member is willing to accommodate in the multicast tree. We further define the term called remnant fanout to denote the current capacity of accepting new children, i.e. remnant fanout = fanout − the number of existed children. We can see from Figure 3 that the fanout of a cluster head is larger than 1. The hosts with fanout of less than 2 will be pushed down to the bottom of the multicast tree. In this paper, we do not discuss the details of the push procedure, and assume that each host can accept more than one child.

We define a cluster threshold λ such that the max distance between the head and the members in a cluster is less than λ. In addition, we use $d(n, c)$ to mean the distance of shortest unicast path from node n to node c. When a newcomer N wants to join a group, it first contacts the trans-server. The latter tries to transform N's IPInfo into a valid candidate AT item. If the trans-server finishes the translation successfully, it returns the candidate AT item to N. Otherwise, the trans-server notifies N of this. Once obtaining a candidate AT item, the newcomer does as follows:

- If there are some items (in AT table) which contain the allounit that $N's$ candidate AT item claims, the newcomer finds the closest home host C. Then N decides whether $d(C, N) < \lambda$ holds or not. If the inequality holds, N becomes a member of the cluster whose head is C. Otherwise, N joins the group as some ALM protocol does, and adds a new item to AT table with the allounit and N's address.

- If AT table does not contain any item with the same allounit that N's candidate AT item contains, the corresponding new item is added to AT table. Then N joins the group as some ALM protocol does.

Note that the newcomer joins the group as a cluster head when its candidate AT item is added to AT table. Thus a small-sized or medium-sized allounit topology space forms a cluster, and a large-sized allounit topology space is divided into multiple clusters. Since the cluster threshold can test the validity of the topology hints, it addresses the third challenge mentioned in Section 3.2. If the IPInfo of a newcomer cannot be translated into a valid candidate AT item, the newcomer joins the group as normal solution does. As noted above, the event happens with very small probability. Thus we can neglect the last problem noted in Section 3.2. Figure 3 shows an example of our proposed scheme. In Figure 3, the allounit space $A2$ is divided into three clusters – $A2-1$, $A2-2$ and $A2-3$.

In a given multicast topology space Ω, we use S_i to represent the space (called covering space) of a subset of Ω, which covers a local and medium-sized topology area. In addition, we suppose that there are m such covering spaces in Ω. In Ω, we use S'_i to mean the space of a cluster formed by our propose scheme, and assume that there are m' such cluster spaces. In addition, function N_i and N'_i mean the set of nodes in the space S_i and S'_i, respectively. We define a function $\Theta(i,j)$ as

$$\Theta(i,j) = \begin{cases} 1 & \text{if } \exists k(i,j \in N_k \wedge (i \in N'_l) \wedge (j \in N'_p)) \\ 0 & \text{otherwise} \end{cases}, \tag{2}$$

where i and j mean two different host nodes, and $l \neq p$. For obtaining topology-aware clusters, cluster threshold λ and area threshold μ should be found to minimize the following objective function $f_{obj}(\cdot)$:

$$f_{obj}(S_1, S_2, \cdots, S_m) = \sum_{i=1}^{m} \sum_{j \in N(S_i)} \sum_{\substack{k \in N(S_i) \\ j \neq k}} \Theta(j,k). \tag{3}$$

4.2 Scheme Application Framework

In this part, we further present an application framework of our proposed clustering scheme, which can integrate the clustering scheme into some existing ALM protocols, as Algorithm 1 shows. In Algorithm 1, when a newcomer N wants to join a group, it contacts the trans-server with its IPInfo. If the IPInfo can not be translated into a valid candidate AT item, the newcomer only joins the group as the chosen ALM protocol does. Otherwise, trans-server returns a candidate AT item to the newcomer. When AT server receives the searching request, it executes the Search procedure to seek the items containing informed allounit. If the Search procedure finds some items containing informed allounit, AT server returns the addresses in the items to the newcomer. Otherwise, AT server returns *null* to the newcomer. In addition, if

the result is null, a new item is added to AT table through the Add procedure. The Join(S, N) procedure means that N joins the group as the chosen ALM protocol does, except that cluster members are not returned as candidate parents. In the JoinCluster(C, N) procedure, N joins the group beginning from node C, and only C and the member nodes in the cluster can become N's candidate parents. In nature, the protocol that the JoinCluster(C, N) procedure uses is independent of the one that the JoinCluster(C, N) procedure adopts. In our proposed scheme, the member host without a valid candidate AT item is also considered as a cluster head. To summarize, there are two types of nodes in our proposed scheme, i.e. cluster members and cluster heads. Therefore, a cluster head might have two types of children, i.e. cluster children (which are the members of the cluster) and common children (which are heads of other clusters). The scheme coordinates the numbers of the children of different types by the following rules:

- If the remnant fanout of node H is larger than 0, H can accept a node as its child of any type.
- If the remnant fanout of node H is equal to 0, then do as follows:
  - Suppose that a node N wants to become a common child of node H and H has more than one cluster child, H will accept N as its child instead of an existing cluster child. Note that the replaced node will rejoin the group beginning from the original parent.
  - Assume that a node N wants to become a cluster child of node H, but H has no existing cluster child, N will become H's child instead of an existing common child.

In our proposed scheme, the member hosts of a group are divided into many clusters. The hosts in a cluster build a cluster tree, and all the cluster heads build the inter-cluster tree. In this paper, we do not propose new tree-building algorithm to form the cluster and inter-cluster trees.

4.3 Maintenance and Enhancement

When a host leaves a group gracefully, it will notify its parent and children of its leave. The parent simply deletes the node from its children list, but the host's children have to find new parents. When a head of a cluster leaves the group, it also informs the AT Server to delete the address of corresponding item in AT table. For rejoining the group, a node finds the closest active node j in its root path to initiate the rejoin procedure. Additionally, each host periodically sends live message to its parent to keep active. If a host finds that its parent has left without any notice, it actively initiates the rejoin procedure. Each head of a cluster also periodically sends live message to the AT Server to keep the corresponding item active.

In accord with the tree-building algorithm, the improvement of the multicast tree consists of three parts:

Algorithm 1 The newcomer N joins the group initiating from the root S

```
1: procedure JoinGroup(S, N)
2:     \\Suppose that N gets the candidate AT item (named CI)
3:     result ← Search(AT, CI).
4:     if result = null then
5:         Join(S, N);Add(AT, CI)                    ▷ N becomes a cluster head
6:     else
7:         find the closest returned home host C
8:         if d(C, N) < λ then
9:             JoinCluster(C, N)                     ▷ N becomes a cluster member
10:        else
11:            Join(S, N); Add(AT, CI)               ▷ N becomes a cluster head
12:        end if
13:    end if
14: end procedure
```

1. improvement of the inter-cluster tree,
2. improvement of cluster trees, and
3. cluster head reselection.

The first two parts depend on the specific algorithms that our proposed scheme uses, and we neglect the details. The last part is to select an appropriate approximate center of a given cluster, which can reduce the average (packet-receiving) latency of the nodes in the cluster. Currently, we also use some existing approach (i.e. the way used in NICE protocol) to find the center, except that the new center must have fanout of larger than a configurable parameter γ ($\gamma > 0$). If a new center c' of some cluster is selected, the old head c is replaced with the new center c' in address field in AT table, and the common children of c will rejoin the group starting from c'.

5 SIMULATION EXPERIMENTS

We used HMTP protocol to implement the Join and JoinCluster procedures in our proposed scheme, and called the integrated solution HMTP-E. We employed the GT-ITM Generator [22] to generate a 5200-node transit-stub graph as underlying network topology. Each node represented a router, and the average degree of router nodes was between 3 and 4. The server's location was located at a stub-domain node randomly. In these experiments, stub domains represented medium and small allounit areas, and transit domains indicated large allounit areas. In a given group, five percent of member hosts had not valid candidate AT items, twenty percent of member hosts were identified by stub domain IDs, and other hosts were identified by transit domain IDs. We used these IDs to mean the allounits in our proposed clustering scheme. Note that the above configurations are coincident with the investigation results in Section 3.2. We simulated HMTP and HMTP-E with

NS-2 [23], and analyzed the performance of HMTP and HMTP-E in two different scenarios.

5.1 Scenario 1: Dense Mode

In the dense mode, we generated 1 000 member nodes, and each member node was connected to a stub node with a delay between 1 ms and 20 ms. Furthermore, n different member nodes were connected to a certain router node with a probability of $\alpha^n(1-\alpha)^{s-n}$, where α is 0.3 and s is the group size. Additionally, the fanout of each member host was random between 2 and 5. We made experiments with 10 different group sizes from 100 to 1 000, increasing by 100.

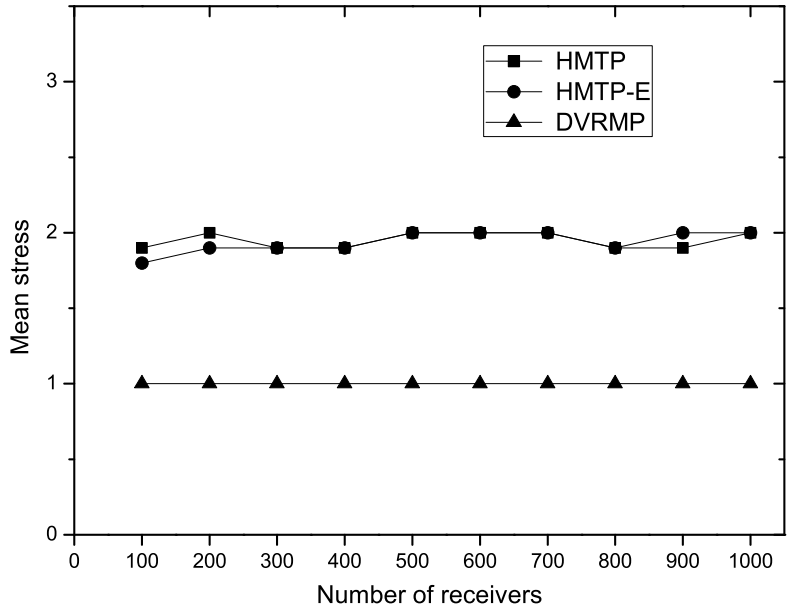

Fig. 4. Link load in the dense mode

1) Stress: Figure 4 plots the mean stresses in HMTP, HMTP-E and DVRMP in 10 different groups. As expected, the mean stress of DVRMP is 1 in each group. We can notice that the mean stress in HMTP and HMTP-E each keeps a low value. In HMTP-E, the cluster is topology-aware. In other words, the number of member hosts in a cluster is not confined. Therefore the mean stress in a cluster might increase, and the mean stress in HMTP-E might be higher than that in HMTP in some cases.

Most experimental studies have only investigated the mean stress in all related links, and ignored that in backbone links. However, it is more important to

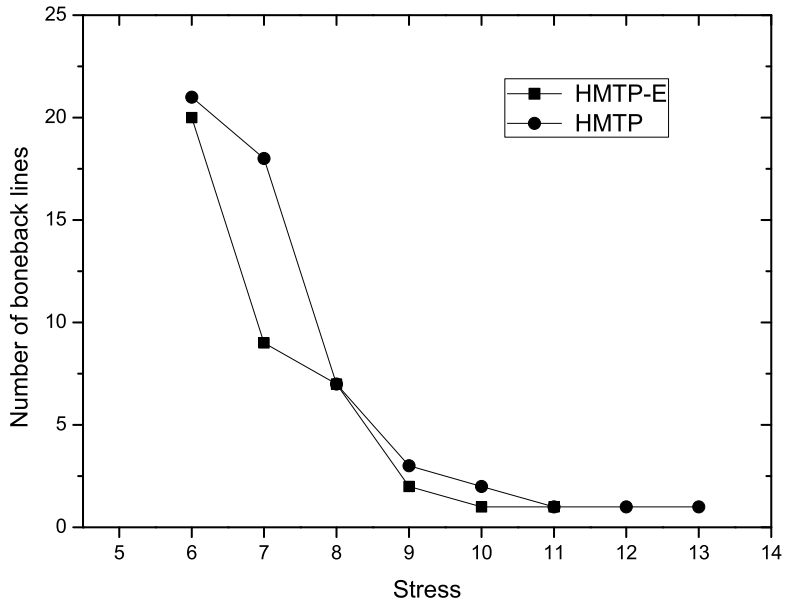

Fig. 5. Stress distributions in backbone links in a group containing 1 000 receivers

reduce the stress in backbone links, because backbone links bear more multicast application than other links. Therefore, we investigated the stress distribution in backbone links. In our experiments, a backbone link was the link that connected two different transit domain nodes. Figure 5 depicts stress distributions in backbone links in HMTP and HMTP-E. The horizontal axis represents stress, and the vertical axis indicates the number of physical backbone links with a given stress. Note that we only counted the numbers of backbone links whose stress values were more than 5, because these links were of heavy stress. We can notice that each of distribution curves has a heavier-tail. However, we also see that the number of the backbone links of heavy stress in HMTP-E is obviously lower than that in HMTP. We attribute the advantage of HMTP-E to the topology-aware clusters.

2) Tree cost: We use tree cost ratio (the ratio of the HMTP's or HMTP-E's tree cost to corresponding SPST's tree cost) to evaluate the performance, here SPST denotes the shortest path source tree. Figure 6 shows the tree cost ratios of HMTP and HMTP-E trees.

We can observe from the figure that HMTP-E has lower tree cost than HMTP in each group. In this scenario, we assigned low value to the cluster threshold. In other words, the nodes were tightly clustered. Tightly clustering is advantageous when the group size is relatively small and members are distributed intensively. However, as the group size grows, the number of clusters will rapidly increase,

which can weaken the role of our proposed clustering scheme, as Figure 6 describes. We can use a cluster threshold of higher value to further reduce the tree costs in large-scale groups, as the next part shows.

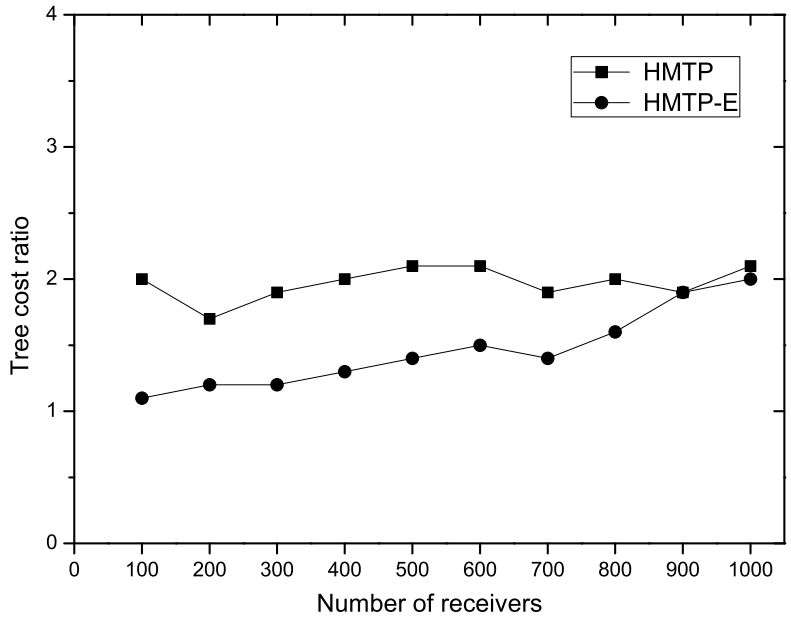

Fig. 6. Tree cost ratios in the dense mode

3) Join overhead: We define join overhead reducing ratio ($JORR(X)$) as $(O(X-E)-O(X))/O(X)$, where O(X) means the join overhead of X protocol, and $X-E$ represents the solution that integrates our proposed scheme into X protocol. In addition, we compute $O(X)$ by $\sum_{i=1}^{m} n(i,X)$, where $n(i,X)$ means the number of touched nodes before n_i becomes a group member by X protocol, and m is the number of receivers in a given group. Figure 7 shows join overhead reducing ratios in 10 different groups. We can clearly notice from the figure that our proposed scheme can greatly reduce the join overhead, especially in large and medium groups. According to the above definition, low overhead means that our proposed scheme can position the hosts quickly.

5.2 Scenario 2: Disperse Mode

In the disperse mode, we also generated 1 000 member nodes. However, in this mode n different member nodes were connected to a certain router node with a probability of $(0.1)^n(0.9)^{s-n}$, where s is the group size. In this part, the fanout of each host was random between 2 and 3. Additionally, we assigned a high value (1 600 ms) to

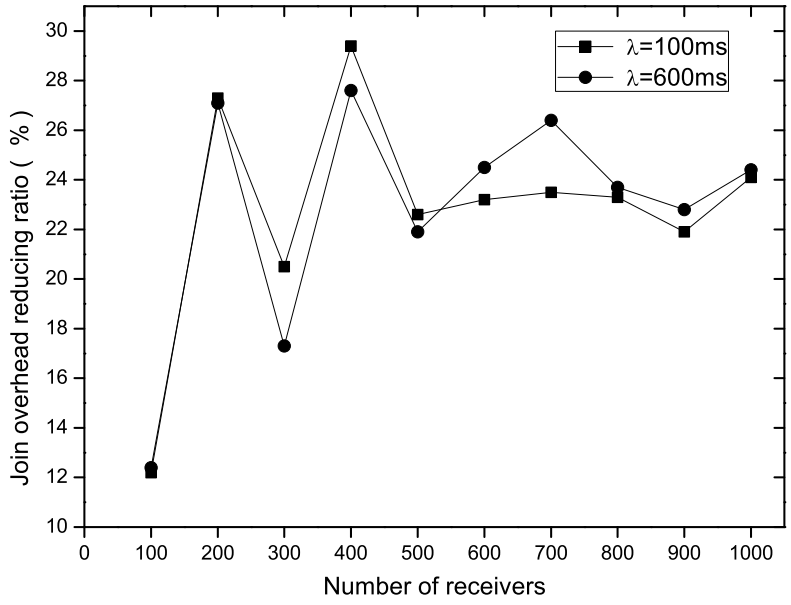

Fig. 7. Join overhead reducing ratios in the dense mode

the cluster threshold. Note that 1 600 ms was still lower than the average delay of backbone links.

1) Stress: Figure 8 shows the mean stresses in DVRMP, HMTP and HMTP-E. In this scenario, the mean stresses in HMTP and HMTP-E are slightly higher than those in the previous scenario, because the hosts in this mode are more disperse. Similar to Figure 4, the mean stresses in HMTP-E might be higher than those in HMTP in some groups.

2) Tree cost: In this scenario, the tree cost ratio of the HMTP-E tree is also lower than that of the HMTP tree in each group, as Figure 9 shows. Different from Figure 6, we can see that the tree cost ratios of HMTP-E trees are evidently lower than those of HMTP trees in large-scale groups, because the cluster threshold is relatively higher in this scenario.

3) Join overhead: On the whole, our proposed scheme can greatly reduce the join load, as Figure 10 shows. However, the join overhead might increase when the group is small, because few clusters can/should be formed in this case and the addition operations in our proposed scheme cannot get good gains.

6 CONCLUSION AND FUTURE WORK

This paper proposed a clustering scheme to improve the performance of application layer multicast. The scheme gets topology hints from the registered information on

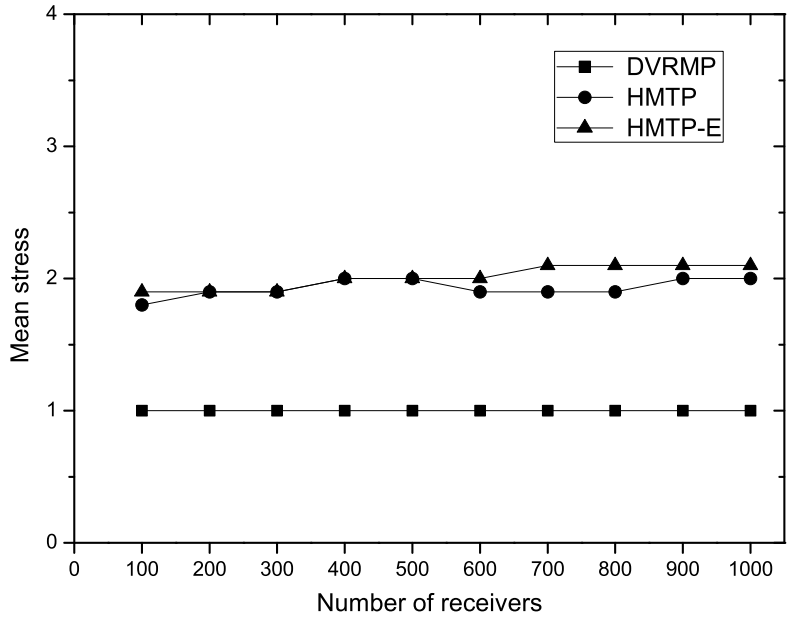

Fig. 8. Link load in the disperse mode

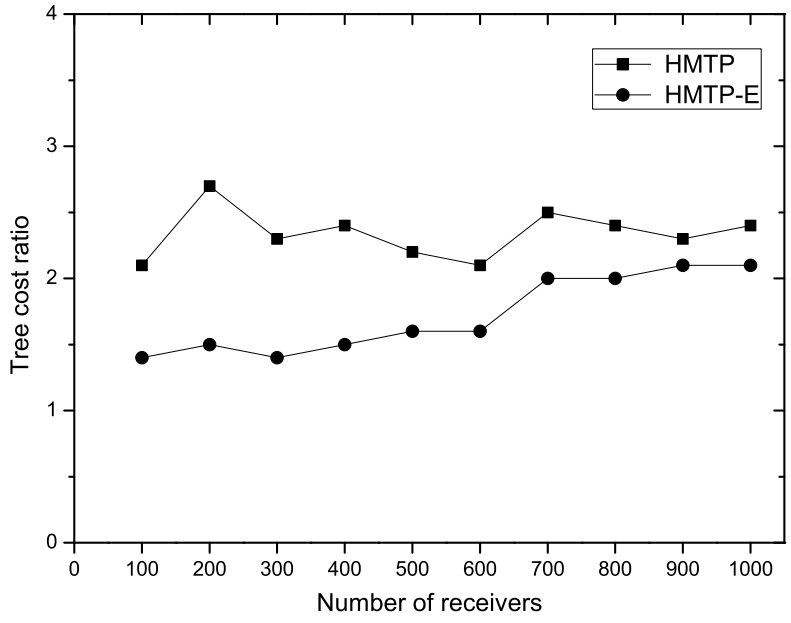

Fig. 9. Tree cost ratios in the disperse mode

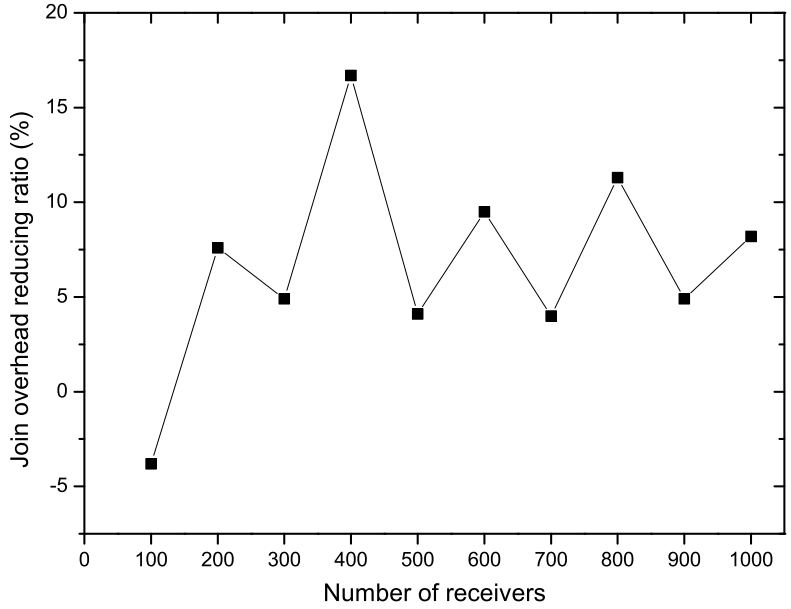

Fig. 10. Join overhead reducing ratios in the disperse mode

IP addresses, and forms topology-aware clusters according to the hints. In addition, the topology hints are useful for an ALM solution to position the hosts quickly. We also presented an application framework of the scheme, and gave an application example to improve HMTP protocol. We evaluated our proposed clustering scheme through simulating the example application with NS-2. The experiment results show that the scheme is desirable. One of our future works is to build a trans-server to provide positioning service, and test and verify the positioning results in the Internet environment. Meanwhile we plan to evaluate HMTP-E in the Internet environment. Another on-going work is to validate our proposed scheme widely through integrating the scheme into more proposed application layer multicast solutions.

Acknowledgments

This work was supported by the National Basic Research Program of China under contract number 2003CB314807, and Science Research Innovation Fund of Computer Network Information Center of Chinese Academy of Sciences under contract number CNIC-CX-08003.

REFERENCES

[1] Diot, C.—Levine, B.—Lyles, B.—Kassem, H.—Balensiefen, D.: Deployment Issues for the IP Multicast Service and Architecture. IEEE Network, Vol. 14, 2000, No. 1, pp. 78–88.

[2] Banerjee, S.—Bhattacharjee, B.: Comparative Study of Application Layer Multicast Protocols. `http://www.cs.umd.edu/projects/nice/papers/compare.ps.gz`.

[3] Zhang, B.—Jamin, S.—Zhang, L.: Host Multicast: A Framework for Delivering Multicast to End Users. In: Proceedings of IEEE Infocom, June 2002.

[4] Banerjee, S.—Bhattacharjee, B.—Kommareddy, C.: Scalable Application Layer Multicast. In: Proceedings of ACM Sigcomm, August 2002.

[5] Chu, Y. H.—Rao, S. G.—Zhang, H.: A Case for End System Multicast. In: Proceedings of ACM SIGMETRICS, June 2000.

[6] Mathy, L.—Canonico, R.—Hutchison, D.: An Overlay Tree Building Control Protocol. In: Proc. of Networked Group Communication, 2001.

[7] Pendarakis, D.—Shi, S.—Verma, D.—Waldvogel, M.: ALMI: An Application Level Multicast Infrastructure. In: Proceedings of 3rd Usenix Symposium on Internet Technologies & Systems, March 2001.

[8] Castro, M.—Druschel, P.—Kermarrec, A. M.—Rowstron, A.: A Large-Scale and Decentralized Application-Level Multicast Infrastructure. IEEE Journal on Selected Areas in communications (JSAC) 2002.

[9] Li, Z.—Mohapatra, P.: Hostcast: A New Overlay Multicasting Protocol. In: IEEE International Communications Conference, June 2003.

[10] Jannotti, J.—Giord, D. K.—Johnson, K. L.—Kaashoek, M. F.—O'Toole, J. W.: Overcast: Reliable Multicasting with an Overlay Network. In: Proc. of OSDI 2000, October 2000.

[11] Tran, D. A.—Hua, K. A.—Do, T. T.: ZIGZAG: An Efficient Peer-to-Peer Scheme for Media Streaming. In: IEEE INFOCOM, San Francisco, USA 2003.

[12] Mathy, L.—Canonico, R.—Simpson, S.—Hutchison, D.: Scalable Adaptive Hierarchical Clustering. IEEE Communication Letters, Vol. 6, 2002, No. 3, pp. 117–119.

[13] Banerjee, S.—Kommareddy, C.—Kar, K.—Bhattacharjee, B: Construction of an Efficient Overlay Multicast Infrastructure for Real-Time Applications. In: Proc. Joint Conf. IEEE Computer and Comm, Societies (INFOCOM), IEEE Press 2003.

[14] Lao, L.—Hong, J.—Gerla, M.—Chen, S.: A Scalable Overlay Multicast Architecture for Large-Scale Applications. IEEE Transaction on Parallel and Distributed Systems, Vol. 18, 2007, No. 4, pp. 449–459.

[15] Li, X.–Striegel, A. D.: A Case for Passive Application Layer Multicast. Computer Networks, Vol. 51, 2007, No. 11, pp. 3157–3171.

[16] Tan, S. W.—Waters, G.—Crawford, J.: A Performance Comparison of Self-Organising Application Layer Multicast Overlay Construction Techniques. Computer Communications 2006.

[17] Kwon, M.—Fahmy, S.: Topology-Aware Overlay Networks for Group Communication. In Proceedings of ACM NOSSDAV, May 2002, pp. 127–136.

[18] Ng, T.—Zhang, H.: Predicting Internet Network Distance With Coordinates-Based Approaches. In: Proc. of the INFOCOM 2002, New York, IEEE Communication Society 2002.

[19] Ratnasamy, S.—Handley, M.—Karp, R.—Shenker, S.: Topologically Aware Overlay Construction and Server Selection. In: Proc. IEEE INFOCOM, New York, NY, USA, June 2002.

[20] Costa, M.—Castro, M.—Rowstron, A.—Key, P.: PIC: Practical Internet Coordinates for Distance Estimation. In Proc. of ICDCS '04, Tokyo, Japan 2004.

[21] Asia Pacific Network Information Centre, `http://www.apnic.net/db/index.html`.

[22] Calvert, K.—Zegura, E.—Bhattacharjee, S.: How to Model an Internetwork. In: IEEE Infocom, 1996.

[23] The Network Simulator-ns2, `http://www.isi.edu/ns-nam/ns`.

Xinchang Zhang is working toward the Ph. D. degree at the Computer Network Information Center of Chinese Academy of Sciences. His research interests include network protocols and architectures, multicasting, and overlay networks.

Xiaodong Li is the Chief Technology Officer of CNNIC, and the Chair of EAI working group in application area of IETF. His research interests include performance optimization questions in ad-hoc and sensor networks, traffic engineering, congestion control and multicasting DNS-related technologies, network protocols and architectures.

Wanming Luo is Associate Professor at Computer Network Information Center, Chinese Academy of Sciences. His current research interests include computer networks, performance evaluation, network security analysis, and mobile computing.

Baoping Yan is Professor, Supervisor of Ph. D. students of Computer Network Information Center, Chinese Academy of Sciences (CAS); Board Member, The Internet Society (ISOC); Deputy Director, CODATA Chinese Committee. Her research interests include scientific databases, next generation networks, network addressing and network measurements.