Construct optimized overlay multicast via preferential random walk[1]

Xuan Zhang
Network Research Center, Tsinghua University, Beijing, China,100084
zhangx@cernet.edu.cn
Xing Li
Network Research Center, Tsinghua University, Beijing, China,100084
xing@cernet.edu.cn
Chongrong Li
Network Research Center, Tsinghua University, Beijing, China,100084
licr@cernet.edu.cn

Abstract

Tree depth and load balancing are two main metrics in overlay multicast network. Optimizing the two metrics with lightweight overhead is important for live media streaming. This paper proposes one scheme to construct optimized overlay multicast with short tree depth and load balancing via short random walk. The key idea is the preferential random walk based on fitness function in which the tree depth and load balancing metrics are defined as parameters with weighted coefficients. Simulations and experiments show that the fitness function is valid and optimized overlay network could be constructed via preferential random walk. We also find the local and global optimized results occur at some middle value of coefficient between 0 and 1, which is not consist with our intuitions that optimized result with single metric should occur at the boundary of coefficient(0 or 1.0).

Keywords: optimized overlay multicast, tree depth, preferential random walk, load balancing

1. Introduction

For delivering live media streaming from one sources to large amounts of users, native multicast is probably the most efficient way. But for some practical reasons as deployment and management cost, IP multicast is still not ubiquitous available on internet. Recently the ALM(Application Layer Multicast) or overlay multicast has played important roles on streaming media transmission for their easily deployment on internet. By utilizing the application layer users' resources, all participant nodes are organized into overlay network that can relay streaming data between themselves.

The traditional ALM algorithm is to construct application layer multicast tree such as NICE[1], ZigZag[2] and HMTP[3].This way is similar to network level multicast, is not steady on application level when peer users arbitrarily join and leave with frequent tree reconstruction. The other catalog is to construct P2P based overlay, including *structured overlay* as Chord[4],Pastry[5] and *unstructured overlay*[6][7].The structured overlay scheme constructs multicast tree by the DHT (distribution hash tables) method. The unstructured overlay scheme

[1] The work is supported by National Science Foundation of China under grant No. 60703052

employs gossiping or flooding mechanism for message exchanging [6]. In a typical gossiping-based overlay [7], a node send a newly generated message to a set of random selected nodes, the random choice of gossiping mechanism achieves resilience to peers' randomly leave and join with decentralized structure.

The algorithm described in this paper belongs to the *unstructured overlay*, there is no centralized server or equivalent mechanism to monitor the global state of the network and no structured network constructed.

To simply state our approach for transmitting streaming media via overlay multicast network, we consider only one streaming media source and large number of participant hosts that receive the streaming. Each participant node could do as client and server at same time, which receives streaming media from its parent node and transmits data to its children nodes.

During the construction of the overlay multicast network, when one node wants to join the overlay, it would choose a suitable overlay node as its parent node and make connections to receive streaming data from parent. Three key questions should be considered during the construction of overlay network: (1) Efficient: how to find the suitable parent node quickly with least cost, and keep the least maintaining information (2) Load balancing: No node is significantly more loaded than others. (3) Low overlay delay for media data transmission.

The common unstructured overlay network uses gossiping or flooding mechanism for message exchanging during overlay network construction, which may make it heavyweight overhead. In this paper, we design an algorithm based on short preferential random walk for message changes in overlay network construction. The random walk length is limited in O(logN) steps. The nodes need only maintain information of neighbor nodes. The short walk step length and small amount of maintaining information make it lightweight overhead and scalable. To achieve the low overlay delay and load balancing, the algorithm adopts a fitness function which considers both load balancing and delay for preferential random walk.

The rest of this paper is organized as followings. Section 2 describes related works, sections 3 describes the protocol details, section 4 about the performance metrics. Section 5 gives results on simulation and experiment with analysis on performance, section 6 ends with conclusions.

2. Related works

Many algorithms have been proposed for the media streaming data dissemination of overlay network. They could be divided into two classes: distributed complicated servers scheme and peer-to-peer based scheme. The former system disseminate media streaming data via distributed servers or application-level proxies that placed on the broad bandwidth location [8][9].This scheme is not scalable for the bandwidth or performance bottleneck of server itself, the number of hosts could be served is limited. The peer-to-peer based scheme doesn't rely on the limited resource of predisposed servers, but constructs one self-organized overlay network by utilizing the hosts' resources flexibly with well scalability[6][7][10]. Our work belongs to the second category.

Some pioneering works on load balancing overlay adopts the methodologies of selective polling, global random choice, etc [11][12].The later research work such as BONet [13] proposes the random walk to balance the node's computing/server load of mirror http server, in which degree-base random walk based are used, but the degree-map methodology is too complex to implement in practice. The scheme in this paper adopts the idea random walk and replaces the degree-based random walk of BONet with fitness based preferential random walk.

The theory on random graph and random walk in random network gives us illuminations [14][15] during our algorithm designing.

3. Protocol description

Here we present the protocol in details. First the simple random walk is reviewed and preferential random walk is introduced, then the protocol details with fitness based preferential random walk are described.

3.1 Simple random walk and preferential random walk

During the simple random walk in one undirected graph G=(V,E), the walk is simple and randomly. The transition probabilities from current position *u* to any of its neighbor *v* are set as the same values [16]. The transition probability *P(u,v)* could be expressed as :

$$P_{u,v} = \begin{cases} 1/d_u & : (u,v) \in E \\ 0 & : others \end{cases} \quad (1)$$

In which d_u means the degree (num of links) of vertex *u*。

We could regard the simple random walk as one Markov process. To one finite states and all states experienced Markov chains, the probability of walker's states would achieve a steady distribution π after enough amounts of steps, no matter what the walker starts from any primary states: $\pi * P = \pi$ [16].

In order to search or locate the desired nodes in the network via random walk, two questions should be considered: (1) what steady distributions would the walker's states trend to be after enough steps walk? (2) How many steps would the walker need to walk for reaching the node whose states close to the desired steady state?

To vertex $u \in V$, its degree is *d(u)*. The total degree of the network is $d_T = \Sigma_u d_u$. In simple random walk with enough steps walk, node *u*'s steady distribution is [16]

$$\pi_u = d_u / d_T \quad (2)$$

Formula (2) means in simple random walk the probability that node *u* could be walked is proportional to its degree. The nodes with large degree would be sampled much more than the nodes with small degree. But the simple random walk could only achieve the degree-based distribution for node sampling. It is not flexible when nodes' properties distribution does not depend on degree distribution.

To make the distribution of walker's steady state change to our expected steady distribution, one way is to modify the transition probability at each walk step. MCMC (Markov chain Mente Calo)[17] is one of these algorithms. The principal of this algorithm is that the transition probability is generated according to the current state and the expected steady distribution. The transition probability matrix is decided by the following formula [18][19]:

$$P_{u,v}^{(MH)} = \begin{cases} P_{u,v} \min(1, \dfrac{\pi_v P_{v,u}}{\pi_u P_{u,v}}) & : v \in V \\ 1 - \Sigma_{i \in V,(u,i) \in E} P_{u,i} \min(1, \dfrac{\pi_i P_{i,u}}{\pi_u P_{u,i}}) & : v = u \\ 0 & others \end{cases} \quad (3)$$

From formula (3) we can get

$$\pi_u P_{u,v}^{(MH)} = \pi_v P_{v,u}^{(MH)} \quad (u \neq v) \quad (4)$$

Equation (4) is named as Detailed Balance Equation [17].

In real network, nodes' some properties could be described and expressed as *fitness*. When our goal is sampling nodes according to node's *fitness* distribution via random walk, the probability of the walker's transition should be proportional to the node's *fitness*. On this condition, the steady state of node *u* is:

$$\pi_u = \frac{h_u}{\Sigma_{v \in V} h_u} \quad (5)$$

h(u) means *u* node's fitness. According formula (3)(5), the *fitness based* preferential random walk could be adopted. Its transition probability could be presented as following:

$$P_{u,v}^{(h)} = \begin{cases} \min(\dfrac{1}{d_u}, \dfrac{h_v}{h_u d_v}) & : v \in V, v \neq u \\ 1 - \Sigma_{i \in V,(u,i) \in E} \min(\dfrac{1}{d_u}, \dfrac{h_i}{h_u d_i}) & : v = u \\ 0 & others \end{cases} \quad (6)$$

Formula (6) indicates that in fitness based preferential random walk, one step transition probability only depends on the neighbor's fitness, it needs not know network's global fitness distribution. This character ensures the preferential random walk can achieve the node's steady state only using local information instead of global information. In the next section, we will define *fitness* in the construction of overlay multicast network and present the practical preferential random walk methodology.

3. 2 Construction of overlay multicast network

The key step in the construction of overlay multicast network is how to find and select optimal host as parent node to make connection when new node joins. The optimal node is gained via fitness based preferential random walk.

3.2.1. Definitions: The overlay multicast network could be defined as one directed graph *G(V,E)* as figure 1. Vertex (*src*) is defined as the source of media streaming. Other vertexes *(V)* in the graph present peer nodes which could receive media streaming from parent node and transmit them to children nodes. Edges *(E)* in the graph present real links with arrow direction

from parent nodes to children nodes. The overlay multicast tree is constructed through new nodes' joining and connecting to parent nodes.

In the overlay network, each peer node needs only maintain its neighbor nodes' information including parent node, children nodes and virtual parent nodes.The three types of neighbors is defined as following data structure:

```
Node {
        Node     parent;
        NodeList   children_list;
        NodeList   virtual_parent_list;
        ……
}
```

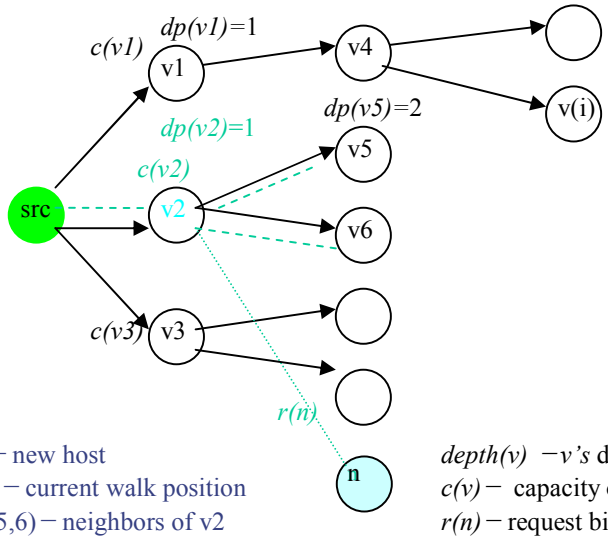


Figure 1. overlay network graph and one step preferential random walk

Peer node's main properties are described as following:

a) *dp(x)* or *depth(x)*denotes the host *x*'s depth in the overlay multicast tree. It presents the overlay hops from source node to host *x*, usually corresponds to overlay delay for media data transmission. In Figure 1, *dp*(src) = 0 ,*dp(v1)=dp(v2)= dp(v3)*= 1; *dp(v4)= dp(v5)* =2 .

*rdp(x)= dp (x)/(*log*N)* represents the host's relative depth in the overlay multicast tree. N represents the size of the overlay network (peer nodes' number).

b) *c(x)* and $c_0(x)$ represents the available resource or capacity of peer *x* that could be provided for children nodes. Here we define *c(x)* as the node's current available bandwidth. *c(x)* changes according to network state. We define $c_t(x)$ as *x*'s capacity at time *t* and $c_0(x)$ as the initial capacity before host joins.

c) $ld_t(x) = c_0(x) - c_t(x)$ represents peer *x*'s load cased by the children nodes at time *t*

$rld_t(x) = ld_t(x) / c_0(x) = 1 - c_t(x) / c_0(x)$ represents the relative load on peer *x* at time t to time 0, when t=0,*rld*=0.

d) *r(n)* : When new node *n* wishes to join overlay network, it requests *r(n)* bit rate bandwidth from parent node. *r(n)* denotes how much the parent node's *c(x)* would decrease when one new child node connects to it.

3.2.2. Nodes join: The new node joins the overlay network by connecting to a suitable parent node selected from a nodes set via multiple steps random walk.

1) One step random walk :

When node *n* joins overlay network, it selects one node in overlay network as its start position for walk. As show in *Figure 1*, node *n* selects node *v2* as walker's start position. Then it begins to walk and transits to next node.

Which node would be selected as the next step walk position is the key issue in the protocol. The walker prefers to walk to the neighbor node with lower *fitness*. Here we define function *h(v)* as the node's fitness in which both nodes' relative load and relative depth are considered.

$$h(v) = 1/(\alpha * rld(v) + \beta * rdp(v)) \qquad \alpha + \beta = 1,\ \alpha > 0,\ \beta > 0 \quad (7)$$

rld(v) denotes node *v*'s current relative load, *rdp(v)* denotes node *v*'s relative depth in overlay multicast tree.

The probability that which neighbor node would be selected as next walk position is proportional to nodes' *fitness*. Unlike *formula (6)*, the walker need not rest on its local position, at each step it must transit to a new position, and the one step transition probability depends on the neighbor's fitness *h(v)* and capability *c(v)* as following:

$$P_{u,v}^{(h)} = \begin{cases} \dfrac{h_v}{\Sigma_{v \in V,(u,v) \in E} h_v} & : v \in V, v \neq u \quad \exists_v c(v) > r \\ \dfrac{1}{d_v} & v \in V, (u,v) \in E \quad c(v) < r \\ 0 & (u,v) \notin E \quad or \quad u = v \end{cases} \quad (8)$$

In formula (8), line 1 means the transition probability from current node *u* to its neighbor node *v* is proportional to node *v*'s fitness *h(v)* if at least one neighbor's *c(v)>r(n)* . The largest *h(v)* nodes with *c(v)>r(n)* (if exist) would be added to the nodes set *S(n)*, which is defined as the possible parent nodes set.

Line 2 means when all neighbor's *c(v)<r(n)*, no neighbor is preferentially to be the next walk position. If no neighbor has enough resource provided for one child nodes, the walker need just choose one neighbor randomly as simple random walk does.

Line 3 means the walker won't rest on or walk to the unconnected nodes (Here virtual connected node is regard as neighbor). No neighbor node would be added to the node set *S(n)* in case of line 2 and line 3.

In Figure 1, node *v2* is the current walk position, *v5,v6* and *src* are its neighbors .The dashed lines in the figure represent the probable walk directions during one step walk.

2) Connection making

After several steps random walk (walk length *l*=O(log N), N is the size of overlay network),the walk ends and *S(n)* could be gained. The new node (*node_new*) would choose one optimal node from *S(n)* with maximal *h(x)* as its parent node (*node_parent*) and make connection to it. If possible, the joined node also holds several second optimal nodes from *S(n)* as backup parent nodes (virtual link). The new node's connecting to parent could be described as:

AddNodeToList(*node_parent.chilren_list, node_new*);

node_new.parent = node_parent

Once the connection is made, the parent node and the new joined node's *c(x)*/ *rld(x)* should be updated, the depth and relative depth of the new joined node are updated too.

$$c_{t+1}(node\_parent) = c_t(node\_parent) - r(n)$$
$$c_{t+1}(node\_new) = c_t(node\_new) - r(n)$$

node_new. depth = node_parent.depth + 1;

3.2.3. Nodes depart: When one peer node departs from the overlay normally, it sends leave message to its parent node and children nodes. If the node departs abnormally, the connected nodes would know the state of disconnecting via heat-beat mechanism. All its neighbors would update their connection information and resource information after the node departs.

The departed node's parent node would delete the node from its *children_list*, and update its *c(x)* and *rld(x)*

DeleteNodeFromList(*node_leave.parent.chilren_list, node_leave*)

$$c_{t+1}(node\_leave.parent) = c_t(node\_leave.parent) + r(node\_leave)$$

The departed node's children nodes would quickly switch to a backup parent node (virtual link) and build real link. When the backup nodes' number is below some threshold, a new random walk process would be launched to complement the number of backup parent nodes.

The update of node *c(x)* and *rld(x)* could reflect the overlay node's current real load state and do benefit for overlay load balancing. The backup parent nodes (*virtual_parent_list*) make the overlay network stable and robust. The quick switching to backup node does prevent the overlay from fluctuating when nodes depart arbitrarily.

4. Metrics

In overlay multicast network, one important overlay delay metric is the hops from peer nodes to source node. We define <*depth*> as the average depth or average hops in the overlay multicast dissemination tree. Obviously less the <*depth*> is, lower overlay delay would be.

$$< depth >= \frac{\Sigma_N depth(x)}{N}. \qquad \text{N : size of the overlay network}$$

The following describes the metrics on load balancing.

In an ideal load balancing overlay network with N peer nodes, each node should have the same relative load. The theoretical perfect average *rld* could be defined as *E(rld):*

$$E(rld) = \frac{\Sigma_N ld(x)}{\Sigma_N c_0(x)} \qquad (6)$$

ld(x) means node *x* 's actual load. When ideal load balancing is achieved, the variance of *rld* is zero.

In fact the variance of *rld*(x) could't be zero, the actual average *rld(x)* value could hardly equal to *E(rld)*. The actual average *rld(x)* could be computed as following:

$$< rld >= \frac{\Sigma_N rld(x)}{N}$$

One metric for load balancing is the deviation from actual average <*rld*> to ideal *E(rld):*

$\triangle <rld> = |<rld> - E(rld)|$

The other metric is the variance of the *rld(x)* represented as:

$\delta^2(rld) = < (rld(x) - < rld >)^2 >$

Less the <*rld*> deviation and variance are, better the performance on load balancing should be.

5. Simulations and analysis

We simulate the construction process of overlay multicast network with the simulator developed by us according to the protocol described at section 3. At beginning one node is designated as the streaming source node, then the nodes join the overlay network one by one by connecting to suitable parent node with different initial *c(x).*

After all N nodes join the overlay, the overlay multicast network construction ends. By changing parameters, different results would be gained. The following analysis focus on three aspects: (1) Overlay network performance in case of coefficient α=0 and β=0 respectively in preferential random walk (2) The influence of join sequencing on load balancing (3) The effect of coefficient α and β on overlay network performance.

5.1. Node's initial *c(x)* distribution of reciprocal law and power-law

Mostly the nodes' initial *c(x)* distribution is heterogeneous. Two typical distributions, reciprocal distribution and power-law distribution are used in our simulations.

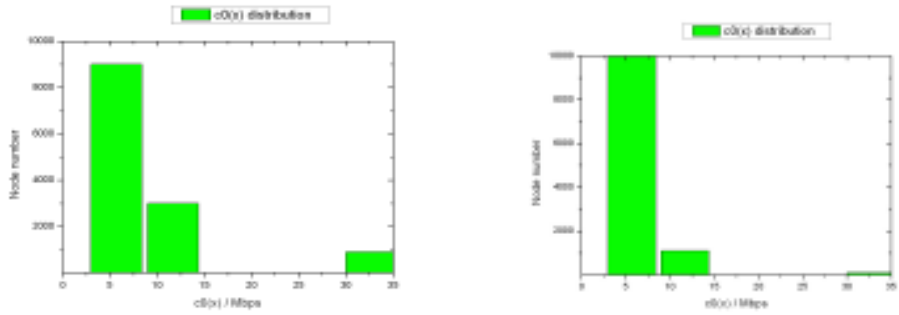
The reciprocal distribution of the node's initial capacity *c(x)* could be expressed as

$$P(c_i(x)) \sim k / c_i(x) \ , k \in R$$

Figure 2(a) shows one of the node's initial *c(x)* reciprocal distribution, three initial *c(x)* values: 3.0Mbps, 9.0Mbps and 30.0Mbps are set. The node number to each *c(x)* is inverse proportional to its *c(x)* value as figure 2(a). Here we set *r(n)*= 0.5 Mbps. For each node has only one actual connection to parent node, the average degree (in-degree plus out-degree) to the whole network would be *2*. And the average load of each node is 2**r(n)* = 1Mbps.

The power-law distribution could be expressed as following formula, figure 2(b) shows one of its distributions with network size 11200:

$$P(c_i(x)) \sim c_i(x)^{-\gamma}, \gamma \in (2,3)$$

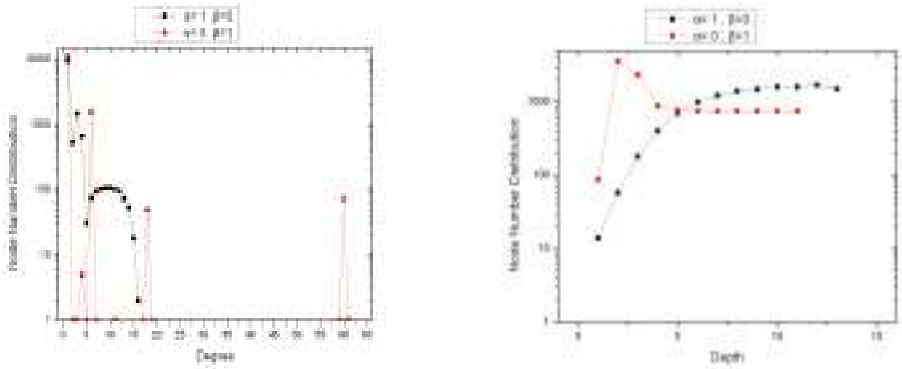


(a) reciprocal distribution, N=12900 (b) power-law distribution, N= 11200

Figure 2. Initial c(x) distribution with three c(x) values respectively 3.0Mbps,9.0Mbps and 30.0Mbps.

5.2. Performance comparison in case of α= 1 and α = 0

According to figure 2(a) initial distribution, we simulate the two extreme cases with α=1(β=0) and α=0(β=1). When α=1 the peer nodes' load balancing is the only goal in the fitness function during preferential random walk. When α=0, the lower average depth of dissemination tree is the only objective during the overlay construction. Figure3 (a) shows the nodes' degree (node load) distributions of the overlay network in case of α=1(β=0) and α=0(β=1), and figure 3(b) shows the nodes' depth distribution of the overlay network in the two cases. The nodes' initial *c(x)* distribution is corresponding to figure 2(a) with node number 12900, walk length is 12, and the nodes' join sequencing is larger *c(x)* nodes first.



(a) Degree distribution comparison (b) Depth distribution comparison

Figure 3 . Comparison of nodes' degree and depth distributions between α=0 (β=1) and α=1(β=0)

Table 1 shows the comparison on variance of relative load and the average depth on the whole overlay network.

Table 1. Performace contrast between α=0 (β=1) and α=1(β=0)

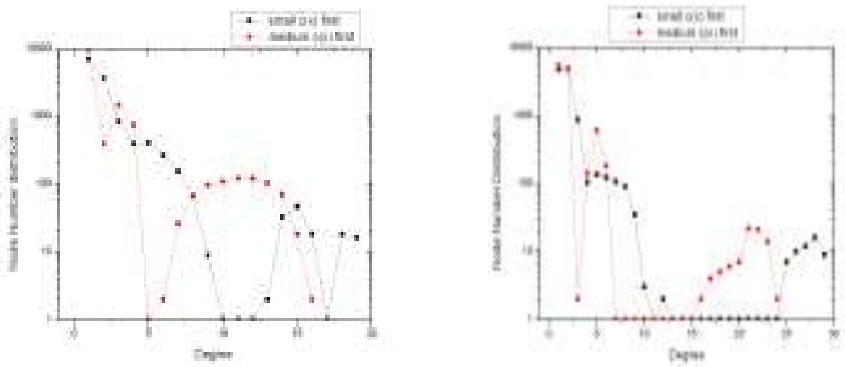
	δ(*rld*)	<*depth*>	max(*depth*)
α=1 (β=0)	0.02857	9.2038	13
α=0(β=1)	0.29796	5.3381	11

Table 1 shows the comparison of the metrics. When α =1 small δ(*rld*) with large <*depth*> gained, this result is consistent to preferential function *h(x)* definition in which α is the weight of *rld*. On the other hand, when α =0 ,<*depth*> is small and δ(*rld*) is large. In fact the δ(*rld*) 0.29796 in case of α =0 is almost 10 times as δ(*rld*) 0.02857 in case of α =1.And <*depth*> value 9.2038 in case of α =1 is almost twice as 5.3381 in case of α =0.

The simulation results show the coefficient α and β in fitness function could do determine the preference of the two performance metrics. The fitness function is valid in constructing optimized overlay network with fitness based preferential random walk.

5.3. Join sequencing

In practice the node's join sequencing plays import roles on load balancing when the node's capacity *c(x)* is not uniformly distributed. Intuitively, the larger *c(x)* nodes join first would benefit the network's load balancing



(a) Initial c(x) with reciprocal law distribution. *(b) Initial* c(x) *with power-law distribution.*

Figure 4 . Degree distribution on different join sequencing: medium *c(x)* nodes join first, small *c(x)* nodes join first.

Figure 4 shows the overlay network nodes' degree distributions when nodes' join sequencings are different. The two different join sequencings are: medium capacity nodes with *c(x)* 9Mpbs join first; small capacity nodes with *c(x)* 3Mbps join first as figure 2. By comparing $\triangle$<*rld*> and variance δ(*rld*) showed in table 2, the way that larger *c(x)* nodes join first achieve better performance than smaller *c(x)* nodes join first.

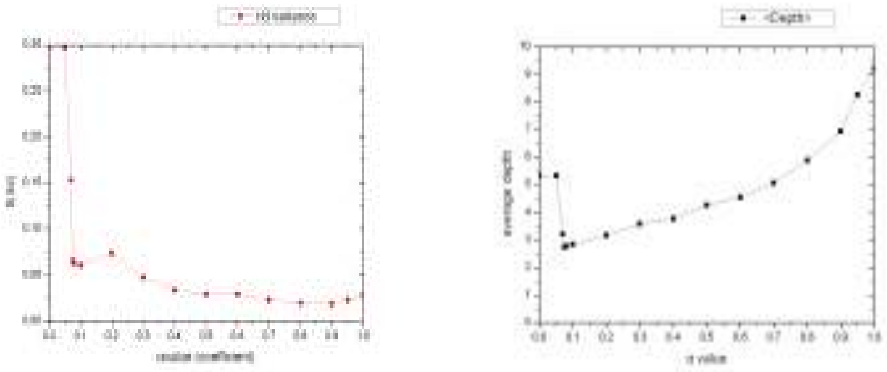
Table 2. Comparing medium *c(x)* join first to small *c(x)* join first

Join sequencing	E<*rld*>	<*rld*>	$\triangle$<*rld*>	δ(*rld*)
medium *c(x)* first	0.1592	0.1635	0.0043	0.0321
Small *c(x)* first	0.1592	0.2132	0.0540	0.1079

5.4. The effect of α on δ(*rld*) and <*depth*>

In order to find how the coefficient α of *h(x)* affect on δ(*rld*) and <*depth*>, we change α from 0.0 to 1.0 to simulate the overlay network construction. Figure 5(a) shows the simulation result when initial *c(x)* distribution law is reciprocal. The δ(*rld*) decreases when α increases in most cases, but the trend is not always decline, when α=0.075, δ(*rld*) decreases suddenly from 0.153 to 0.065,and then increases lightly when 0.075 < α < 0.2 .With α increase from 0.2 to 0.9, δ(*rld*) decreases slowly and increases slightly when α>0.9. The global minimum value occurs at α=0.9 with δ(*rld*) =0.0188, which is the optimized result on load balancing.

Figure 5(b) shows how the <*depth*> changes when α increases. <*depth*> is not always drop down with α decreases. The least value of <*depth*> occurs at α = 0.075 with <depth> = 2.74 which is the global minimum value, not occurs at α=0 when *depth* is the only preferential factor during preferential random walk.



(a) $\delta(rld)$ vary with α (b) <*depth*> vary with α

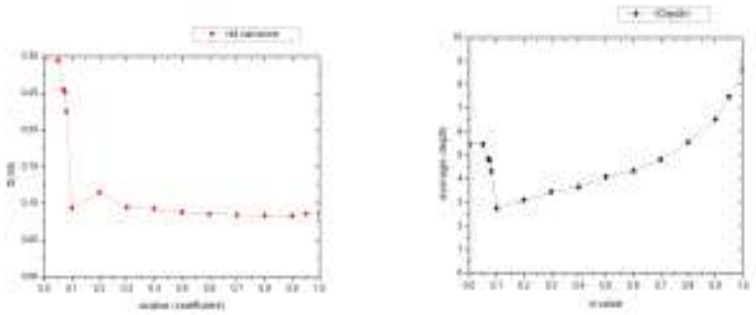
Figure 5 . δ(rld) and <depth> vary with α coefficient, N = 12900, walk length = 12. c(x) initial distribution of reciprocal law.

From the two curves, we find at α = 0.075 the global minimum <*depth*> and local minimum δ(*rld*) (not global minimum) could be gained. That is to say, when α = 0.075, we could gain the minimum <*depth*> with nearly optimized load balancing performance. This is important and useful to us for constructing optimized overly. It tells us when changing coefficient α, we could get suitable α to achieve optimized performance between load balancing and overlay delay. Table 3 shows the performance details nearby α = 0.075.

Table 3. Performance details around α = 0.075 of reciprocal distribution

α	0.05	0.07	0.075	0.1	0.2
δ(*rld*)	0.297	0.153	0.065	0.66	0.073
<*depth*>	5.33	3.21	2.74	2.84	3.16

Figure 6 shows how δ(*rld*) and <*depth*> changes with α when initial *c(x)* distribution is power-law. We find the two curves have the similar shape as figure 5 shows. The results indicate that the optimized result could be gained at some α near to 0.075, which is one common disciplinary. The reason for the disciplinarian would be exploited in our future work.



(a) $\delta(rld)$ vary with α (b) <*depth*> vary with α

Figure 6 . δ(rld) and <depth> vary with α coefficient, N = 11200, walk length = 12. c(x) initial distribution of power law

6. Conclusions

In this paper, we propose one scheme to construct optimized overlay multicast network with lower overlay delay and load balancing. The fitness function considering both overlay delay and load balancing with α as weighted coefficient is defined to implement preferential random walk and construct optimized overlay network. Simulations and experiments show that he fitness function is valid and the optimized overlay network could be gained. We also find the local and global optimized performance results occur at some middle α value between 0 and 1, which is not consist with our intuitions that optimized result with single metric should occur at the boundary of α (0 or 1.0).

7. References

[1]S.Banerjee,B. Bhattachargee, and C.Kommareddy. Scalable application layer multicast. In ACM SIGCOMM,August 2002.

[2]D.A.Tran,K.A.Hua,and T.T.Do. Zigzag: An efficient peer-to-peer scheme for media streaming.In IEEE INFOCOM,June 2003

[3] B.Zhang, S,Jamin,and L.Zhang. Host multicast:A framework for delivering multicast to end users. In IEEE INFOCOM ,June 2002

[4] I.Stoica,R.Morris, D.Karger, M.F.Kaashoek,and H.Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet application. In ACM SIGCOMOM,August 2001.

[5] A.Rowstron and P.Druschel. Pastry:Scalable, decentralized object location and routing for large-scale peer-to-peer systems. In IFIP/ACM International Conference on distributed System Platforms,November 2001.

[6] X.Zhang, Q.Zhang, Z.Zhang, G.Song,and W.Zhu. A construction of locality-aware overlay network: moverlay and its performance. IEEE Journal on Selected Areas in Communications, January 2004

[7] A-M. Kermarrec, L. Massoulie, and A.J.Ganesh. Probabilistic reliable dissemination in large-scale systems. IEEE Transactions on Paralled and Distributed systems, 14(3):248-258,2003.

[8] L. Guo, S. Chen, S. Ren, X. Chen, and S. Jiang, "PROP: a scalable and reliable P2P assisted proxy streaming system," in Proc. ICDCS'04,Tokyo, Japan, Mar. 2004.

[9]S. Shi and J. Turner, "Routing in overlay multicast networks," in Proc. INFOCOM'02, New York, Jun. 2002.

[10]X,Zhang, J.Liu, B.Li,and T.Yum. Data-driven overlay streaming: Design, implementation, and experience. In IEEE INFOCOM,Mar.2005

[11] R. Lling, B. Monien, and F. Ramme. A study of dynamic load balancing algorithms, 1991.

[12] R. Els and B. Monien. Load balancing of unit size tokens and expansion properties of graphs. In Proceedings of the fifteenth annual ACMsymposium on Parallel algorithms and architectures, pages 266– 273. ACM Press, 2003.

[13]Jesse S. A. Bridgewater, P. Oscar Boykin, Vwani P. Roychowdhury. Balanced Overlay Networks (BON): Decentralized Load Balancing via Self-Organized Random Networks .CoRR cs.DC/0411046: (2004).

[14]P.L.Krapivsky,S.Redner, F.Leyvraz, Connectivity of growing random networks, Phys.Rev. Lett. 85, 4629(2000)

[15]C.Gkantsidis,M.Mihail,and A.Saberi. Random walks in peer-to-peer netwroks.In IEEE INFOCOM,March 2004.

[16] R.Motwani and P.Raghavan. randomized Algorithms. Cambridge University Press,1995

[17]Aldous Fill. reversible Markov chains and random walks on graphs. http://stat-www.berkeley.edu/users/aldous/RWG/book.html

[18]N.Metropolis,A.rosenbluth,M.Rosenbluth,A.Teller, and E.Teller.Equation of state calculations by fast computing machines.J.Chenm. Phys,21:1087– 1092 ,1953

[19]W.Hastings.Monte carlo sampling methods using markov chains and heir applications. Biometrika, 57:97-109,1970.

Authors

Xuan Zhang received his B.Sc. degree from BUAA in 1995 and MS. degree from Tsinghua University in 2000. He is now a Ph.D. candidate of Electronics engineering and instructor in Tsinghua university. His research interests are network multimedia, multicast in next generation internet. He designed and implemented the nationwide native multicast based videoconferencing system on CERNET backbone covering all the provincial network center

Xing Li received his B.Sc. degree in radio electronics from Tsinghua University in 1982, and his MS. AND Ph.D. degrees in electrical engineering from Drexel University, Philadelphia in 1985 and 1989, respectively. He is now a professor of China Education and Research Network (CERNET) Center. He holds memberships of the following: communication and Expert Committee of the China national "863" High-Tech Project, Technical Board of CERNET, China Communication Institute, IEEE signal Processing Society, IEEE Computer Society, ISOC, Board of Executive Council of APNIC, Board of Steering Committee of APIA, Sigma Xi, and he is a vice chairman of APNG.