Hierarchical Multicast Tree Algorithms for Application Layer Mesh Networks*

Weijia Jia[1], Wanqing Tu[1], and Jie Wu[2]
[1]Deprtment of Computer Science, City University of Hong Kong,
83 Tat Chee Ave. Hong Kong, China
Email: itjia@cityu.edu.hk
[2]Department of Computer Science and Engineering
Florida Atlantic University, Boca Raton, F133431, USA

Abstract. This paper proposes a set of novel multicast algorithms for *m*-D mesh overlay networks that can achieve shorter multicast delay and less resource consumptions. In contrast to previous approaches, our algorithms partition the group members into clusters in the lower layer, seeking an *optimal* core (root) to guarantee the minimum routing delay for each cluster and building a shared tree within each cluster to minimize the number of links used. In the upper layer, a shared tree is then constructed using our algorithms to implement the inter-cluster routing. The extended simulation results indicate that the application layer multicast that is constructed by our algorithms is efficient in terms of routing delay and link utilizations as compared with other well-known existing multicast solutions.

1 Introduction

Multicast function was originally implemented in the network layer [1]. In recent years, the *application layer multicast* is considered as an alternative multicast function in the overlay network (i.e. the application layer) by many researchers [2-9] for the following attractive features: 1) no requirement for multicast support in the network layer of OSI reference model; 2) no need to allocate a global group id, such as IP multicast address; and 3) data is sent via unicast which enable flow control, congestion control and reliable delivery services that are available for the unicast can also be employed in the application layer multicast.

Generally, the overlay topologies for the application layer multicast fall into two categories: (1) Topologies consisting of a single tree [3,10-11]; (2) Abstract

*This work is supported by Strategy Grant of City University of Hong Kong under nos 7001709 and 7001587 and partially by the National Grand Fundamental Research 973 Program of China under Grant No.2003CB317003.

coordinate spaces obtained from *m*-D Cartesian coordinates on an *m*-torus [5, 12-13]. Such abstract coordinate space is a mesh from which members are assigned the logical addresses. A drawback of using a single tree is that the failure of a single application may cause a partition of the topology. The advantage of building the overlay mesh network is that the next-hop routing information can be encoded in the logical addresses for the good choice of address space and topology. It shows that the robust communications of the application layer multicast built in the mesh overlay network.

Many well-known multicast schemes based on the mesh network have been presented. Double-Channel XY Multicast Wormhole Routing (DCXY) [14] uses an extension of the XY routing algorithm to set up the routing scheme. Dual-Path Multicast Routing (DPM) [15] is developed for the *2*-D mesh. It assigns a label l for each node in the mesh and partitions the group into two subgroups (i.e. g_h and g_l) such that they are composed of the members with their l greater (g_h) or less (g_l) than the label of the source respectively. The routing paths are constructed through connecting the nodes covered by g_h in the ascending order of the l value and the nodes covered by g_l in the descending order of the l value. CAN-based multicast [5] is developed for the P2P applications that utilize the CAN (Content-Addressable Network) [16] configuration. CAN-based multicast is scalable especially when multiple sources coexist. However, only flooding approach is used for propagating the multicast messages which compromises the efficiency in terms of multicast delay and consumes a large number of network links. We will give the performance comparisons of these well-known multicast solutions with our multicast scheme in Section 3.

Our motivation is to design an application layer multicast scheme in *m*-D mesh overlay networks that can achieve shorter multicast delay and less resource consumptions. The network is partitioned into clusters in terms of some regular mesh area (the issue is omitted due to space limit). After group members are initially scattered into different clusters, a tree is built to connect the cluster members within each cluster. The connection among different clusters is done through hooking the tree roots. To construct such architecture, a set of novel algorithms based on the *m*-D mesh networks are presented: (1) *cluster formation algorithm* that partitions the group members with the "closeness" relationship in terms of *static delay distance* into different clusters; (2) o*ptimal core selection algorithm* that can seek the *optimal* core (i.e. root) for a shortest path cluster tree using the minimum sum of *static delay distances* to all cluster members as the metric; (3) *weighted path tree generation algorithm* that may maximize the link usage (i.e., using the minimum number of links) for creating the shortest path tree to reliably route the multicast message and (4) *multicast routing algorithm* that efficiently dispatches the multicast packets in the group based on the architecture constructed by above three algorithms. Our solution is suitable for both logical address *m*-torus and *m*-D (abstract or physical) mesh networks. To set up such shortest path tree, we apply a heuristic approach to reduce the number of links used so as to utilize the resource effectively. To avoid confusion, we wish to point out that we do not seek the *optimal* multicast tree; instead, we seek the *optimal* core for a cluster of members based on the total *static delay distance*.

The paper is structured into four sections: Section 2 discusses the algorithms for cluster formation, seeking of the *optimal* core(s) for a cluster of nodes, multicast tree

generation and routing. Performance results are demonstrated in Section 3 and we conclude the paper with some discussions in the final section.

2 Algorithms for Multicast Architecture and Routing

Denote the multicast group with n members as $G=\{u_0,\ldots,u_i,\ldots,u_{n-1}\}$, $i\in [0,n-1]$. Suppose the group members are mapped into an m-D mesh network by some P2P scheme. Each member u_i can be identified by m coordinates: $(U_{i,0},\ldots,U_{i,j},\ldots,U_{i,(m-1)})$, where $0 \le U_{i,j} \le k_j-1$ and $0 \le j \le m-1$. End hosts $u_i=(U_{i,0},\ldots,U_{i,j},\ldots,U_{i,(m-1)})$ and $u_{i'}=(U_{i',0},\ldots,U_{i',j},\ldots,U_{i',(m-1)})$ ($i'\in [0,n-1], i'\neq i$) are neighbors if and only if $U_{i,j} = U_{i',j}$ for all j, except $U_{i,j'} = U_{i',j'} \pm 1$ along only one dimension j'. Thus, in the m-D mesh, an end host may have m to $2m$ neighbors. We also define the *Euclid distance* of two nodes in the mesh as their *static delay distance*. In a 2-D mesh, the *static delay distance* of two nodes (X_0,Y_0) and (X_1,Y_1) is $|X_1-X_0|+|Y_1-Y_0|$. The sum of *static delay distances* from (X_0,Y_0) to other nodes (X_i,Y_i) is $f(X_0,Y_0)=\sum_{i=1}^{n-1}|X_i-X_0|+|Y_i-Y_0|$.

2.1 Cluster Formation Algorithm

In our application layer multicast scheme, the group members are initially split into several clusters by some management nodes (called Rendezvous Points – RP). The cluster size is normally set as

$$S=(k,3k-1) \quad (1)$$

The expression $(k, 3k-1)$ represents a random constant between k and $3k-1$. Like NICE, k is a constant, and in our simulation, we also use k=3. The definition of cluster size is for the same reason as the one of NICE that is to avoid the frequent cluster splitting and merging (see [4]). Define the state of the end host that has not been assigned into any cluster as *unassigned*. We describe the cluster formation as follows. The RP initially selects the *left lowest end host* (say u) among all *unassigned* members. The *left lowest end host* is the end host who occupies the mesh node that has the minimum coordinates along m dimensions among all nodes occupied by the *unassigned* group members. The cluster member selection is in the dimension order around u by using the following algorithm.

Alg-1: Cluster Formation
Input: *Unassigned group member set* $G'=\{u_0,\ldots,u_i,\ldots u_{n-1}\}, i\in[0,n-1]$ and the RP;
// n is the set size that initially equals to the group size
Output: Cluster set $CS=\{\}$;
1. While $G'\neq\Phi$ do {
2. the RP selects the *left lowest end host* u in G' and removes u from G';
3. for $j=0$ to $m-1$ do { //m is the dimension number of mesh overlay
4. The RP selects *unassigned* closest member in the j-th dimension into the cluster and removes it from G';
5. For $j'=0$ to $j-1$ do {

---

6. The RP selects the closest *unassigned* member in the sub-mesh $k_j' \times k_j$ into the cluster and removes it from *G'*;}
7. The RP selects the closest *unassigned* member in the sub-mesh $k_0 \times \cdots \times k_j$ into the cluster and removes it from *G'*;
8. If (the cluster size equals to *S*) {*j=m-1*;}}}

Fig. 1 shows a 2-D mesh. In this mesh, the initial *left lower end host* is (0,0). According to steps *3-4*, the RP firstly selects the end host in (0,1) into the cluster. Because *j=0*, steps *5-7* are neglected. Then, the RP selects the end host in (1,0) into the cluster by steps *3-4*. Based on steps *5-7*, the next selected cluster member is the one in (1,1). The cluster formation guarantees that each cluster contains the closest group members in terms of *static delay distance*. According to our research results in [18], the scheme that assigns closed members into the same cluster will improve the scalability and efficiency of application layer multicast.

2.2 Optimal Core Selection Algorithm

Each cluster will have a cluster core. The core is the root of the tree in the cluster. The following theorem gives the sufficient and necessary conditions to select a cluster core in each cluster that is *optimal* in terms of the minimum sum of *static delay distance*s to all other cluster members.

Theorem 1*: Let u be the cluster member that occupies the node $(U_0,\ldots,U_j,\ldots,U_{m-1})$ in a m-D mesh network and $n_{>j}$, $n_{<j}$, and $n_{=j}$ be the number of cluster members with the j-th coordinates larger than, less than, and equal to U_j respectively. Then u is the optimal core if and only if the following m inequalities hold simultaneously:*

$$| n_{<j} - n_{>j} | \le n_{=j}, \quad j=0, 1, \ldots, m-1. \tag{2}$$

Proof (➔): Suppose $u = (U_0,\ldots,U_j,\ldots,U_{m-1})$ is an *optimal core*, then for any member *u'* in the mesh, there exists $f(u) \leqslant f(u')$. To achieve (5), we first consider a node $u'=(U_0,\ldots,U_j+1,\ldots,U_{m-1})$ and its multicast static delay distance $f(u')$. Given any member $u_i = (U_{i,0},\ldots,U_{i,j},\ldots,U_{i,(m-1)})$ and $U_j \leqslant U_{i,j}$, the distance from u_i to the end host *u* is one unit longer than the distance from u_i to *u'*. Similarly, it can be seen that for any member $u_i = (U_{i,0},\ldots,U_{i,j},\ldots,U_{i,(m-1)})$ and $U_{i,j} \leqslant U_j$, the distance from u_i to node *u* is one unit shorter than the distance from u_i to *u'*. Because there exist $(n_{>Uj} + n_{=Uj})$ members whose *j*-th coordinates are larger than or equal to U_j, and $n_{<Uj}$ cluster members whose *j*-th coordinates are less than U_j, we have

$$0 \le f(u') - f(u) = \sum_{i=0}^{n'} (d(u',u_i) - d(u,u_i)) = n_{>U_j} + n_{=U_j} - n_{<U_j}$$

$$\rightarrow n_{<U_j} - n_{>U_j} \le n_{=U_j}$$

By comparing $f((U_0,\ldots,U_j-1,\ldots,U_{m-1}))$ with $f(u)$ in the same way as above, we can achieve the inequality of (2).

(➔): It is easy to demonstrate that if (2) is violated, then u cannot be the *optimal* core. Assume $n_{<Uj} - n_{>Uj} > n_{=Uj}$, then $n_{>Uj} > n_{<Uj} + n_{=Uj}$. This means that the number of end hosts with the *j*-th coordinates greater than U_j is more than the other two cases.

Thus the distance from u to these end hosts is larger than some other end hosts, a desired contradiction. ■

The *optimal core selection algorithm* in the m-D mesh network is given below

---

Alg-2: Optimal Core Selection in m-D Mesh Networks

Input: Cluster member set $C=\{c_0=(C_{0,0}, C_{0,1}, \ldots, C_{0,(m-1)}), c_1=(C_{1,0}, C_{1,1}, \ldots, C_{1,(m-1)}), \ldots, c_{(n'-1)}=(C_{(n'-1),0}, C_{(n'-1),1}, \ldots, C_{(n'-1),(m-1)})\}$; // n' is the cluster size

Output: *optimal core* $c^*=(C_0^*, C_1^*, \ldots, C_{(m-1)}^*) \in C$;

1. Initiate $\{a_{(Cj)min}, \ldots, a_{(Cj)t}, \ldots, a_{(Cj)max}\}=\{0, \ldots, 0, \ldots, 0\}$; // $a_{(Cj)t}$ records the number of cluster members whose j-th coordinates equal to $(C_j)_t$, where $(C_j)_{min} \leqslant (C_j)_t \leqslant (C_j)_{max}$ and $0 \leq j \leq m-1$
2. For $k = 0$ to $n'-1$ do
3. If (the j-th coordinate of $c_k == (C_j)_{td}$) $\{a_{(Cj)t} = a_{(Cj)t} + 1;\}$
4. For $i=0$ to $n'-1$ do {
5. For $j=0$ to $m-1$ do {
6. If $(|\sum_{l=(C_j)_{\min}}^{C_{i,j}} a_l - \sum_{l=C_{i,j}}^{(C_j)_{\max}} a_l| \leq a_{(C_{l,J})})$ $\{C_j^*=C_{i,j}; j=j+1;\}$
7. Else $\{j=m-1; i=i+1;\}\}\}$
8. $c^*=(C_0^*, \ldots, C_j^*, \ldots, C_{(m-1)}^*)$.

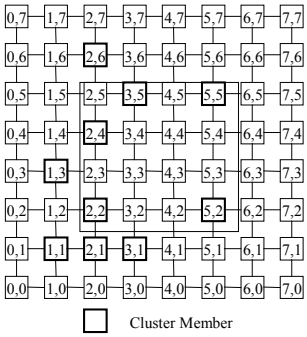


Fig. 1. Selecting the *optimal* core in a *2*-D mesh.

In *Alg-2*, steps *1-3* can be executed in time $O(n)$. Steps *4-7* can be improved using binary searching algorithm that yields an $O(ln(n))$ complexity. But for brevity of discussion, we keep the linear search algorithm here. The algorithm may select multiple *optimal* cores. Only one of them will be used at random as the current core and other cores can be the back-up cores for fault-tolerance. Fig. 1 illustrates the *optimal core selection* in a *2*-D mesh. It is known that the core should be in the area $[1,1]\times[5,6]$. It can be checked that the optimal core's x coordinate must be 2 while y coordinate could be 2 or 3 for $f((2,2)) = f((2, 3)) = 26$. Node (2, 2) is the member and is preferred to (2, 3).

2.3 Weighted Path Tree Generation Algorithm

To multicast the packets in each cluster, a tree using the cluster core as the root is established in each cluster. Because several multicast groups may exist in the network, multicast traffic has to compete with other traffic. It is anticipated that the tree should maximize the sharing of link utilization within the cluster so that the rest of the links may be used for other traffic. Our approach is to connect all members such that (1) the branch on the tree between two adjacent members is the shortest path in the cluster, (2) under the condition (1), the total number of links on the tree should be also minimized.

Table 1. The weights marked '*' belong to the cluster members

Y=6	0	1*	0	0	0
Y=5	0	3	2*	1	1*
Y=4	0	4*	2	1	1
Y=3	1*	5	2	1	1
Y=2	2	10*	4	2	2*
Y=1	1*	3*	1*	0	0
	X=1	X=2	X=3	X=4	X=5

Before the discussion of the *algorithm*, we first define the following terminologies (using a *2*-D cluster as the model):

1. *Shortest path area nodes* (*SPAN*): For any two nodes (X_0,Y_0) and (X_1,Y_1), let $X_{min}=\min\{X_0,X_1\}$, $X_{max}=\max\{X_0,X_1\}$, $Y_{min}=\min\{Y_0,Y_1\}$ and $Y_{max}=\min\{Y_0,Y_1\}$. X_{min}, X_{max}, Y_{min} and Y_{max} uniquely define a *rectangle area* $[X_0,Y_0]\times[X_1,Y_1]$. Each node (X,Y) in $[X_0,Y_0]\times[X_1,Y_1]$ is on one of the shortest paths between $[X_0,Y_0]\times(X_0,Y_0)$and (X_1,Y_1) and is called the s*hortest path area* (*SPAN*) nodes between (X_0,Y_0) and (X_1,Y_1).
2. *SPAN nodes of a cluster member*: When the tree is built in the cluster with the size of n', we call all nodes in the *SPAN* area from the core (i.e. the root of the tree) (X^*,Y^*) to a cluster member $c_i(i\in[0,n'-1])$ as the *SPAN* nodes of c_i. We take Fig. 1 as an example. Assume that the core is in the node (2,2). All nodes in $[2,2]\times[5,5]$ are the *SPAN* nodes of this cluster member.
3. *Node Weight*: A node may be the *SPAN* node of several cluster members. If a node is the *SPAN* node of *k* cluster members, this node is assigned the weight of *k*. Table 1 gives the weights of all nodes in Fig. 1. Take the non-member node (2,5) as an example. Its weight 3 means that 3 cluster members may pass through node (2,5) to (2,2) by the shortest paths. Apparently, the weight of (2,2) is 10.
4. *Path Weight*: Given a shortest path, the path weight is the sum of all on-path node weights. For example, the weight of path <(2,2),(2,3), …,(2,5),…,(5,5)> is 26.

Let the cluster with *n'* members be $C=\{c_0=(C_{0,0},\ldots,C_{0,(m-1)}),c_1=(C_{1,0},\ldots,C_{1,(m-1)}),\ldots,c_{(n'-1)}=(C_{(n'-1),0},\ldots,C_{(n'-1),(m-1)})\}$ and the cluster core be $c^*=(c_0^*,\cdots,c_{m-1}^*)$. We sort the cluster members in a non-decreasing order of the distances from $c^*$to them, thus $d(c^*,c_i)\leqslant d(c^*,c_j)$ where i j. The main idea of the *weighted path tree generation algorithm* can be sketched as follows. Assign a weight for each node in the *rectangle area* $[c_0^*,\cdots,c_{m-1}^*]\times[c_{i,0},\cdots,c_{i,(m-1)}]$ as described before. After knowing the weight of

each node, the RP computes the weight of each shortest overlay path. The *weighted path tree generation algorithm* is shown below:

Alg-3: Weighted Path Tree Generation
Input: Cluster member $CM=\{c_0=(C_{0,0},\ldots,C_{0,(m-1)}),\ldots,c_i=(C_{i,0},\ldots,C_{i,(m-1)}),\ldots,c_{(n'-1)}=(C_{(n'-1),0},\ldots,C_{(n'-1),(m-1)})\}$, $i \in [0,n'-1]$ and the *optimal* core $c^*=(C_0^*,\ldots,C_{(m-1)}^*)$;

Output: Tree *T;*

1. $T=\{\}$;
2. For any node $c_i=(C_{i,0},\ldots,C_{i,,(m-1)})$ with $((C_j)_{min} \le (C_j) \le (C_j)_{max})$, initialize its weight $W_{c_i}=0$;
3. For $i'=0$ to $n'-1$ do
   If (c_i is a *SPAN* node of $c_{i'}=(C_{i',0},\ldots,C_{i,(m-1)})$) $\{W_{c_i}=W_{c_i}+1;\}$
4. For $i=0$ to $n'-1$ do
   Select the shortest path $P=<(C_0^*,\ldots,C_{(m-1)}^*),\ldots,(C_{i,0},\ldots,C_{i,(m-1)})>$ with the maximum weight and add P to T;

2.4 Multicast Routing Algorithm

To build a tree for each cluster, the *weighted path tree generation algorithm* is employed to construct a tree connecting all the cluster roots for the inter-cluster routing. Then, the *optimal core selection algorithm* is used to select the root of this tree. At last, the following multicast routing is designed to routing the packets among all group members.

Alg-4: Multicast routing for group G:

1. Source s sends its multicast messages to its cluster core c, c then forwards them to the roots r of all other trees;
2. c routes the multicast packets to its own cluster members along the cluster tree;
3. At the same time, all cluster cores, upon receiving the multicast messages, transmit them along the cluster trees to all cluster members within the clusters.

---

3 Performance Evaluations

3.1 Simulation Model

This section evaluates our multicast algorithms with the simulation developed in *ns-2* [17] and run by a group of SUN SPARC-20 workstations. In this simulation, six multicast routing algorithms for 2-D meshes are used for the performance testing and comparison: *Double-Channel XY Multicast Wormhole Routing* (*DCXY*) [14], *Dual-Path Multicast Routing* (*DPM*) [15], *RCWP*, *OcxyP*, *RcxyP* and our multicast scheme named as *OCWP*. *RCWP* is the multicast scheme that randomly selects the cluster core for each cluster but constructs the tree by the *weighted path tree generation algorithm*; *OCWP* is the multicast scheme that selects the cluster core by the *optimal cluster core selection algorithm* and constructs the forwarding paths by the *weighted*

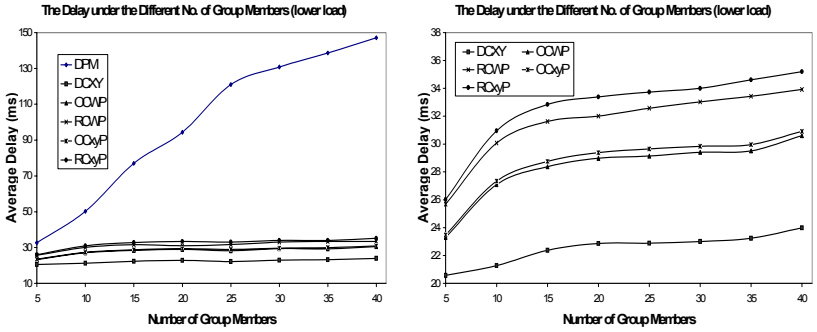
path tree generation algorithm; *OcxyP* selects the cluster core by the *optimal cluster core selection algorithm* but constructs the forwarding paths by using the *XY routing algorithm*; *RcxyP* randomly selects the cluster core and constructs the multicast paths by using the *XY routing algorithm*. The network topology used in the simulation is a 32×32 *2*-D mesh. The bandwidth of each link is 10Mbps. During the simulation, 20,000 multicast packets are randomly generated as a Poisson process and the average size of the packets is 1200 bytes so that the average time to transmit a packet on the defined link is about 1ms. The following two metrics are employed to evaluate these multicast schemes:

- *Average multicast delay*: Define the message multicast delay at a node as the sum of the routing delay, queuing delay and transmission delay. The *average multicast delay* AD is computed by

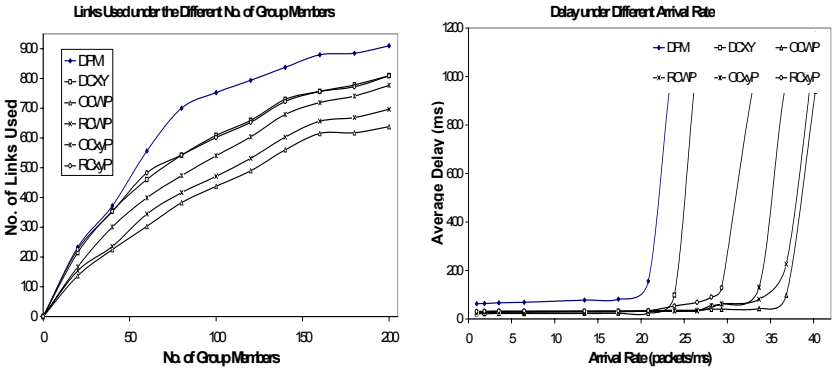
$$AD = (\sum_{i=0}^{n-1} d(s,u_i)) / n \tag{3}$$

  where $d(s,u_i)$ is the packet delay from the source s to the member u_i and n is the group size.
- *Number of link used*: It refers to the total number of links used in G in order to multicast the messages to all group members.

3.2 Simulation Observations on Regular Mesh Multicasting



(a) (b)



(c) (d)

Fig. 2 Simulation results for *DCXY*, *DPM*, *RCWP*, *OcxyP*, *RcxyP* and our *OCWP*.

The average delay metric under the light load of network is shown in Fig. 2 (a) and (b). The link usage for different algorithms is shown in Fig. 2 (c). It can be seen that the average delay increases with the increase of the network load (Fig. 2 (d)). From these simulation results, we have the following observations:

1. Under the lower load circumstance, the delay is mainly related to the distance from the source to the group members (Fig. 2 (a)). Because the *DCXY* approach always transmits multicast packets to group members along the shortest paths from the source to the group members, it achieves the best delay performance among the other systems when the network is lightly loaded. When *DPM* approach is applied, the delay increases rapidly as the number of group members increases. This indicates that DPM does not scale well (Fig. 2 (b)). When traffic is low, *OCWP* achieves the second best delay performance to *DCXY* but it scales well as the traffic increases (Fig. 2 (d)).
2. Fig. 2 (c) shows the average number of links used by these routing approaches. In general the number of links will be increased with the number of the group members. The figure shows that for the same number of group members, *OCWP* makes use of the minimum number of links for transmitting the multicast packets whereas *DPM* uses the maximum. The shared tree routing approach (such as *RCxyP*) uses almost the same number of links as *DCXY*.
3. Fig. 2 (d) shows that the delay increases as the packet arrival-rate increases. The system saturation points for *DPM, DCXY, RCxyP, OCxyP, RCWP* and *OCWP* are about 21.5, 24, 29.5, 34, 36.5 and 37.5 packets/ms respectively. Our algorithm achieves the maximum throughput. It reveals that under the same condition, *OCWP* obtains the best balance over the performance parameters, i.e., the less resource a system consumes, the higher the throughput and the shorter the end-to-end delay under the high traffic load.

4 Conclusions and Future Work

The *cluster formation*, *optimal core selection* and *weighted path tree generation* algorithms are suitable for multicast communication on (abstract) mesh networks. It is proved that the core selection algorithm is *optimal* in terms of the minimum sum of *static delay distances* from the core to all the members in the cluster. The multicast tree formulated by our tree generation algorithm can effectively utilize the links with the shorter average delay. As compared with other multicast schemes, our algorithms can select a suitable core, and construct an efficient tree in terms of balancing the less resource a system consumes, the higher the throughput and the shorter multicast delay under the high traffic load. We anticipate that the issues discussed may be applied to ad-hoc network routing where the nodes can move and an *optimal* core may be re-selected or re-positioned.

References

[1] S. Deering and D. Cheriton, "Multicast Routing in Datagram Internetworks and Extended LANs", ACM Transactions on Computer-Systems, pp. 85-110, Vol. 8, No. 2, May, 1990.

[2] H. Chu, S. Rao, S. Seshan, and H. Zhang, "A case for end system multicast", Proc. of ACM SIGMETRICS 2000, pp. 1-12, June 17-21, 2000, Santa Clara, California, USA.

[3] P. FRANCIS, "Yoid: extending the internet multicast architecture", available at http://www.aciri.org/yoid/docs/index.html, April, 2000.

[4] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast", Proc. of ACM SIGCOMM, pp. 205-217, August 19-23, 2002, Pittsburgh, Pennsylvania, USA.

[5] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, "Application-level multicast using content-addressable networks", Proc. Of The *3*rd International Workshop on Network Group Communication, pp. 14-29, November 7-9, 2001, London, UK.

[6] J. Jannotti, D. K. Gifford, K. L. Johnson, M. Frans Kaashoek, and J. W. O'Toole Jr., "Overcast: reliable multicasting with an overlay network", Proc. of The *4*th Usenix Symposium on Operating Systems Design and Implementation, October 22-25, 2000, Paradise Point Resort, San Diego, California, USA.

[7] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel, "ALMI: An application level multicast infrastructure", Proc. of The 3rd USENIX Symposium on Internet Technologies and Systems, March 26-28, 2001, Cathedral Hill Hotel, San Francisco, USA.

[8] Y. Chu, S. G. Rao, S. Seshan, and H. Zhang, "Enabling conferencing applications on the Internet using an overlay multicast architecture", Proc. of ACM SIGCOMM 2001, pp. 55-67, August 27-31, 2001, San Diego, California, USA.

[9] B. Zhang, S. Jamin, and L. Zhang, "Host multicast: a framework for delivering multicast to end users", Proc. of IEEE INFOCOM 2002, pp. 1366-1375, June 23-27, 2002, New York, USA.

[10] H. Deshpande, M. Bawa, and H. Garcia-Molina, "Streaming live media over a peer-to-peer network", Stanford Univ. Comput. Sci. Dept., Stanford, CA, June 2001.

[11] D.Pendarakis, S.Shi, D.Verma, and M. Waldvogel, "ALMI:An application level multicast infrastructure", *Proc. 3*rd Usenix Symp. Internet Technologies and Systems, San Francisco, CA, pp.49-60, March 2001.

[12] I.Stocia, R.Morris, D.Karger, M.F.Kaashoek, and H.Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications", ACM SIGCOMM 2001, San Diego, CA, pp.160-172, August 2001.

[13] B.Y.Zhao, J.Kubiatowicz, and A.Joseph, "Tapestry: An infrastructure for fault-tolerant wide-area location and routing", Univ. California, Berkeley, CA, Apr. 2001.

[14] X. Lin, P. K. McKinley, and L. M. Ni, "Deadlock-free multicast wormhole routing in *2*-D mesh multicomputers", IEEE Trans. On Parallel And Distributed Systems, Vol. 5, pp.793-804, 1994.

[15] X. Lin, P. K. McKinley and A. H. Esfahanian, "Adaptive multicast wormhole routing in *2*-D mesh multicomputers", *Proc.* of Parallel Architectures And Languages Europe 93, pp.228-241,1993.

[16] S. Ratnasamy, P. Francis, M. Handley, R.Karp, and S.Shenker, "A scalable content-addressable network", ACM SIGCOM 2001, August 27-31, 2001, San Diego, CA, USA.

[17] UC Berkeley, LBL, USC/ISI, and Xerox PARC, "*ns* Notes and Documentation", October 20, 1999.

[18] W. Tu and W. Jia, "A scalable and efficient end host multicast for Peer-to-Peer Systems", *Proc.* of IEEE Globecom 2004, pp. 967-971, November 29-December 3, 2004, Dallas, Texas, USA.