This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE CCNC 2006 proceedings.

Dual Parent Multicast Graph for Failure Resilient Peer-to-Peer Multimedia Streaming

XuanTung Hoang and Younghee Lee
Information and Communications University,
119 Munji-Ro, Yuseong-gu, Daejeon 305-732, Korea
Tel. +82-42-866-6251
Email: {tung_hx, yhlee}@icu.ac.kr

***Abstract*— Failure resiliency is one of major concerns in designing peer-to-peer systems for continuous multimedia streaming since peers can leave or fail without notices. Building resilient multicast networks based on parent multiplicity is an approach to combat network failures and improve stream continuity. In this paper, we model the problem of generating resilient multicast graphs with parent multiplicity, called Dual Parent Multicast Graph (DPMG), as two sub-problems: constructing a regular graph on a set of labeled vertices; and labeling edges of that graph. Our proposed algorithms to solve the two sub-problems produce a DPMG with good resiliency property especially for locally and burstly occurred failures in peer-to-peer networks.**

I. Introduction

Multimedia streaming is an important service in the current Internet. For the purpose of media multicasting, the peer-to-peer approach appears to be promising with ease of deployment, flexibility and scalability [2]–[4]. Peer-to-peer streaming systems, by nature, are dynamic since peers can freely leave or fail without notice. Guaranteeing continuous media streams to receiving peers in such a dynamically changing environment is, thus, a non-trivial task.

For continuous peer-to-peer data dissemination, the traditional multicast tree is insufficient. Although tree topology is the minimum connected graph implying lowest maintenance cost, multicast tree is very vulnerable to failures. Leave or failure of only one peer disconnects all its descendants from the source. This causes service interruption on a large number of peers.

To solve the vulnerability problem of multicast distribution tree, several works based on parent multiplicity have been proposed [5], [11], [16]. The idea of parent multiplicity is that peers select several parent peers instead of only one parent like in the tree configuration. This provides peers backup paths to download data streams when the active path is failed. It also allows peers to utilize path diversity for improving throughput and combating network loss.

In those researches, parents are randomly selected from either the whole set of peers or from a local peer set, such as uncle peer set, cousin peer set or grandparent peer set [5]. On one hand, although selecting parents randomly from the whole peer set guarantees good resiliency, it is, in fact, costly and hard to implement. Such global parent multiplicity strategy requires global knowledge of all peers or a random walk mechanism to sample the network [13], [15]. Global knowledge, e.g. a server tracking status of all peers, does not scale to large systems and is the single point of failure. Network sampling mechanisms based on random walks require relatively long time before returning the sample peer set. Also, with global parent multiplicity, alternative overlay paths to the source can greatly differ in length. This increases the switching time from active path to a backup path and magnifies buffering space at clients. Selecting parents from local peer sets, or local parent multiplicity, on the other hand, is easier to implement and can provide competitive resiliency when failures are uniformly distributed among all peers [5]. However, choosing parent peers locally can be sensitive to failures that happen locally on the peer-to-peer multicast network.

In this paper, we aim at systematically constructing resilient multicast graph with local parent multiplicity, called DPMG (Dual Parent Multicast Graph). We model the problem of generating DPMGs for multimedia streaming applications as two sub-problems: constructing a regular triangle-free graph on a set of labeled vertices; and labeling edges of that graph. The proposed method we apply to solve the two sub-problems systematically generates DPMGs that are robust to failures occurred locally on the topology of the peer-to-peer multicast network. The proposed DPMG can be directly applied to active-backup streaming applications like PRM [16] and can be extended to Multiple Parent Multicast Graph (MPMG) to integrate with path diversity and advanced coding techniques like FEC [9] or MDC [12].

This paper is organized as follows. In the next section, we discuss related works and relate them with the work in this paper. For recursively building DPMG, in section III, we define and solve the problem of constructing a regular triangle-free graph on labeled vertices and the problem of labeling all edges of that triangle-free graph. Section IV is for resiliency evaluation of DPMG. Finally, we conclude our paper in section V.

II. Related Work

Failure resiliency has been an active research area for overlay multicast. Essentially, failures can be solved with network redundancy. Narada [2] follows mesh-first approach. It forms

[0]This work is supported in part by the Korea Science and Engineering Foundation under Grant No. R01-2003-000-10562-0.

a random mesh to obtain reliable communication among peers before constructing a good distribution tree on the mesh. Peers periodically probe each other to establish better links and drop low quality links. BitTorrent [15], Bullet [8], [17] and SCAMP [13] construct random meshes by connecting to a random subset of peers. BitTorrent uses a centralized server, called ”tracker”, to store a list of participating peers. It returned a random subset from the list to a peer requesting to have neighbors. Bullet uses a broadcasting scheme on an overlay tree to collect and distribute random subsets. In SCAMP, peers acquaint lists of neighbors, or views, via gossiping. Multiple copies of a join request originated from a peer independently walk on the network of existing peers before ending up at a set of neighbors for the joining node.

TMesh [1] and PRM [16] obtain network redundancy by constructing an overlay tree then adding backup links to the tree. Besides forwarding data to their children, peers in PRM forward data probabilistically to random selected peers. Thus, each peer can has multiple parents sending data to it, a tree parent and a random parent. This operation mode can be categorized as active-backup operation. If the tree parent fails, a peer can temporarily use the random parent while recovering the failed connection or finding another one.

SplitStream [7] and CoopNet [10] utilize path-diversity for guaranteeing continuity of multimedia streaming. While CoopNet employs a central server to manage overlay network topology, SplitStream distributedly build the streaming network atop DHT-based overlays like Pastry [14]. Both SplitStream and CoopNet construct disjoint trees and stream MDC-coded media streams on those trees independently. K-DAGs [11] applies parent multiplicity for construction of multiple paths. Media streams after being coded with FEC or MDC [12], are sent over the multi-path structure of k-DAGs to each peer for improving stream quality and combating network loss. The idea of DPMG presented in this paper is in fact similar to k-DAGs. However, we specify which peers are preferred to be parents for improving resiliency. Thus it can be applied to k-DAGs to find parents.

The work presented in [5] analyses resiliency of augmented overlay trees when failures are independent and uniformly distributed. The tree augmentation strategies considered in that paper is identical to parent multiplicity scheme. Thus, the analysis of [5] is applicable to DPMG under the uniform and independent failure model. This paper considers failure resiliency of DPMG in another model where failures occurs locally on the logical structure of overlay multicast graph. This failure model is reasonable for dependent failure models, for example, where peers leave the streaming session mainly because of services interruption cause by a small numbers of other peers.

III. Dual Parent Multicast Graph

Since our target application is continuous multimedia streaming in a failure-prone environment, we expect the DPMG to have the following properties:

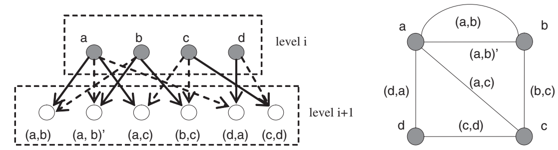


Fig. 1. Two adjacent level of DPMG and the corresponding genGraph representation

```
ALGORITHM TriangleFreeGenGraph
INPUT:
    - n vertices {0,1,2, ..., n – 1}
    - Integer k, 0 < k ≤ n/4
OUTPUT: 2k-regular triangle-free graph G
BEGIN
   G := empty graph
   for j:=0 to k – 1
      for i:= 0 to n – 1
         Add edge (i, i + 2j + 1 mod n) to G
      endfor
   endfor
END
```

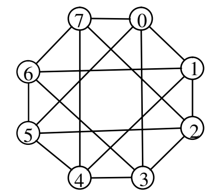


Fig. 2. Algorithm for construction of triangle-free genGraphs and an example output genGraph

- *Parent multiplicity*: Each peer that does not directly connect to the source can download the stream provided by the source via m different parent peers. In this paper, we only consider the case that m equals 2. Similar approach can be applied for arbitrary m.
- *Symmetry*: Overlay paths from a peer to the source via different parents are of the same length. This property is expected for continuous media streaming applications in both streaming systems with active-backup configuration and systems with path-diversity. In active-backup streaming systems, symmetric parent multiplicity implies smallest recovery time when failures occur at the active path and the stream is switched to the backup path. In streaming systems with path diversity, concurrently downloading coded video data via paths of the same length mitigates buffer space requirement and eases the synchronization mechanism.
- *Bounded load*: Peers should not be overloaded by too many children

The above desired properties imply a graph with multiple levels. At the top of the graph, level 0, is the streaming source. Peers that are j-hop away from the source form level j. Each peer at level $j+1$, $j > 0$, takes 2 peers at level j as its parents. We call the ratio between the number of peers at level $j+1$ and level j expansion factor k. Expansion factor k is a fixed parameter of a DPMG except for several (2 or 3) initial levels. For simplicity, we assume that peers at the same level are homogenous. That is they have the same computational power and forwarding capability. Since each peer can be represented by a vertex in the graph model, we use the terms ”peer” and ”vertex” interchangeably.

A. GenGraph Representation

DPMGs described above can be constructed level by level. Consider two successive levels i and $i+1$. Since each peer in

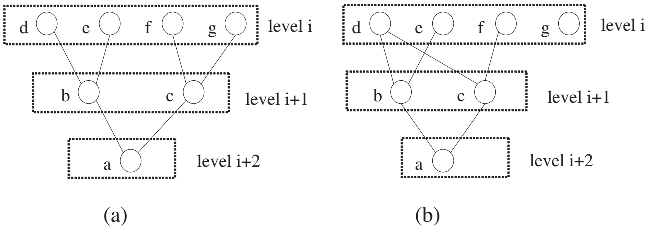


Fig. 3. (a) Peer a has 4 grandparents (expected configuration); (b) Peer a has 3 grandparents and is less reliable than (a)

level $i+1$ has m parents that are at level i, a peer at level $i+1$ can be represented by a set of peers in level i. In other words, all peers at level $i+1$ can be represented by a hypergraph G_i spanning all level-i peers. For the case that m equals 2, G_i becomes a graph and each edge of G_i is correspondent to a peer in the next level. For the ease of exposition, we term that graph G_i *generating graph* or *genGraph*. The reason for the name is that given a $genGraph$ G_i of level i, we can generate the level $i+1$ from G_i. Figure 1 shows an example genGraph G_i and corresponding levels i and $i+i$.

B. GenGraph Construction

Using the concept of $genGraphs$, we can view the problem of building level $i+1$ as the generation of a genGraph G_i. With regard to robustness against failures, the question is how to generate a genGraph G_i so that the number of disconnected peers at the next level (level $i+1$) caused by failures in the current level (level i) is minimized. This kind of inter-level effect, in fact, depends on couples of parameters of the genGraph. Before presenting the relationship between characteristics of genGraphs and the effect at the lower level caused by failures at a higher level, we review the definitions of subgraphs and induced subgraphs in graph theory.

Definition 1 (Subgraphs): A graph $G'(V',E')$ is a subgraph of graph $G(V,E)$ iff $V' \subseteq V$ and $E' \subseteq E$.

Definition 2 (Induced subgraphs): $G'(V',E')$ is induced subgraph of $G(V,E)$ iff G' is a subgraph of G and E' are all those edges in G for which both endpoints are in V'. The induced subgraph $G'(V',E')$ of $G(V,E)$ can be denoted by $G[V']$

GenGraph is actually a useful representation for understanding the effect of failures at the current level to the next level. The relationship between failures in two successive levels can be modeled by induced subgraph relation as follows:

Theorem 1: Given a level i with peer set P_i, genGraph G_i spanning P_i , and a set F_i of offline peers (leaved, failed or disconnected peers), $F_i \subseteq Pi$. The peers at level $i+1$ that become disconnected because of F_i are corresponding to edge set of the induced subgraph $G_i[F_i]$ of G_i.

For high failure resiliency of DPMG, as stated in the above theorem, a desired genGraph should contain induced subgraphs with as few edges as possible. For overall performance of DPMG, however, the genGraph of a level should have as many edges as possible to highly utilize forwarding capacity of peers and to shorten path lengths from peers to the source. That conflict can be compromised by constructing regular graphs with maximum number of edges while guaranteeing graph properties for good resiliency, such as large girth, small clique number and small number of cliques. That graph construction problem is non-trivial for some properties. For a practical algorithm, we just attempt to construct genGraphs that are *triangle-free*. In term of practicality, triangle-free property is possible in relatively many graphs and guarantees reasonably high expanding factor k, $k = \frac{|P_{i+1}|}{|P_i|}$. In terms of resiliency, triangle-free graphs have no cliques of size greater than or equal to 3 and thus, subgraphs induced any f vertices contain less than $\frac{f(f-1)}{2}$ edges.

Figure 2 presents an algorithm that generate a triangle-free genGraph with expanding factor k on n vertices and an example output graph of the algorithm for $n = 8$ and $k = 2$. The algorithm simply constructs a ring lattice as follows. All n vertices are continuously labeled with integers from 0 to $n-1$ and arranged them in order on a circular axis. Between any two vertices whose labels locate $2j-1$ units apart from each other on the label scale, for all integer j from 1 to k, there is an edge connecting them. Note that the algorithm is applicable only when $4k \leq n$. At some initial levels where the number of peers n is still less than $4k$, we relax the requirement of triangle-freedom or accept a smaller expansion factor k when constructing the DPMG.

C. Labeling Edges of GenGraphs

From a genGraph G_i, we need to generate the next level of DPMG by assigning a labeled vertex to each edges of G_i. The output vertex set of those assignments will be used as input for generating the next levels genGraph. Assigning each edge of G_i to a labeled vertex is equivalent to the problem of labeling edges of G_i. The importance of edge-labeling in a genGraph is illustrated in figure 3. Since peer a has four grandparents in figure 3(a) and three in figure 3(b), figure 3(a) shows better resiliency than 3(b). Note that peers in a level are corresponding to edges in genGraph of the upper level. The conditions for "3-grandpa peers" like peer a in figure 3(b) to occur is that 2 endpoints of an edge (peer b and c) in genGraph G_{i+1} are two edges that share a common endpoint (peer d) in genGraph G_i. With a carefully designed labeling scheme for genGraph G_i we can minimized the number of 3-grandpa peers in level $i+2$. Our labeling scheme for genGraph on n vertices with expansion factor k is defined as a function $f_{n,k}(i,j)$ mapping a pair of vertex labels to an edge label as follows.

For even k, $0 < k < n/4$:

$$f_{n,k}(i,j) = f_{n,k}(j,i) = \begin{cases} n\lambda+2i & \text{if } j=(i+2\lambda+1) \bmod n, \ \lambda \text{ is even}, 0\leq\lambda<k \\ n(\lambda-1)+((2i+n-(n \bmod 2)) \bmod 2n) & \text{if } j=(i+2\lambda+1) \bmod n, \ \lambda \text{ is odd}, 0\leq\lambda<k \\ undefined & otherwise \end{cases}$$

$$
\begin{array}{r}0\\1\\2\\3\\4\\5\\6\\7\\8\\9\\10\\11\end{array}
\begin{pmatrix}
* & 0 & * & 13 & * & * & * & * & * & 7 & * & 22\\
0 & * & 2 & * & 15 & * & * & * & * & * & 9 & *\\
* & 2 & * & 4 & * & 17 & * & * & * & * & * & 11\\
13 & * & 4 & * & 6 & * & 19 & * & * & * & * & *\\
* & 15 & * & 6 & * & 8 & * & 21 & * & * & * & *\\
* & * & 17 & * & 8 & * & 10 & * & 23 & * & * & *\\
* & * & * & 19 & * & 10 & * & 12 & * & 1 & * & *\\
* & * & * & * & 21 & * & 12 & * & 14 & * & 3 & *\\
* & * & * & * & * & 23 & * & 14 & * & 16 & * & 5\\
7 & * & * & * & * & * & 1 & * & 16 & * & 18 & *\\
* & 9 & * & * & * & * & * & 3 & * & 18 & * & 20\\
22 & * & 11 & * & * & * & * & * & 5 & * & 20 & *
\end{pmatrix}
$$

(a)

$$
\begin{array}{r}0\\1\\2\\3\\4\\5\\6\\7\\8\\9\\10\\11\end{array}
\begin{pmatrix}
* & 0 & * & 13 & * & 24 & * & 33 & * & 7 & * & 22\\
0 & * & 2 & * & 15 & * & 26 & * & 31 & * & 9 & *\\
* & 2 & * & 4 & * & 17 & * & 28 & * & 29 & * & 11\\
13 & * & 4 & * & 6 & * & 19 & * & 30 & * & 27 & *\\
* & 15 & * & 6 & * & 8 & * & 21 & * & 32 & * & 25\\
24 & * & 17 & * & 8 & * & 10 & * & 23 & * & 34 & *\\
* & 26 & * & 19 & * & 10 & * & 12 & * & 1 & * & 35\\
33 & * & 28 & * & 21 & * & 12 & * & 14 & * & 3 & *\\
* & 31 & * & 30 & * & 23 & * & 14 & * & 16 & * & 5\\
7 & * & 29 & * & 32 & * & 1 & * & 16 & * & 18 & *\\
* & 9 & * & 27 & * & 34 & * & 3 & * & 18 & * & 20\\
22 & * & 11 & * & 25 & * & 35 & * & 5 & * & 20 & *
\end{pmatrix}
$$

(b)

Fig. 4. Edge-labeling for example genGraphs: (a) $n = 12$, $k = 2$; (b) $n = 12$, $k = 3$

For odd k, $0 < k < n/4$:

$$
f_{n,k}(i,j) = f_{n,k}(j,i) =
\begin{cases}
n\lambda+2i & \text{if } j=(i+2\lambda+1) \bmod n,\\
 & \lambda \text{ is even}, 0\le\lambda<k-1\\
n(\lambda-1)+ & \text{if } j=(i+2\lambda+1) \bmod n,\\
((2i+n-(n \bmod 2)) \bmod 2n) & \lambda \text{ is odd}, 0\le\lambda<k-1\\
n\lambda+2i & \text{if } j=(i+2\lambda+1) \bmod n,\\
 & \lambda=k-1, 0\le i\le\lceil\frac{n}{2}\rceil-1\\
2kn-1-(n\lambda+2i) & \text{if } j=(i+2\lambda+1) \bmod n,\\
 & \lambda=k-1, \lceil\frac{n}{2}\rceil\le i\le n-1\\
undefined & otherwise
\end{cases}
$$

For ease of explanation, we term the distance between two endpoints of an edge on the circular label scale the length of the edge. For example, edges $(0, 3)$, $(5, 8)$ and $(11, 2)$ have length 3 in a genGraph G_i on 12 vertices. Note that with our genGraph construction algorithm described in section III.B, every edge has odd length. Our labeling scheme labels all edges of length $2\lambda + 1$ with even integers from $n\lambda$ to $n(\lambda + 2) - 1$ if λ is even, and with odd integers from $n(\lambda - 1) + 1$ to $n(\lambda + 1) - 1$ if λ is odd. For the case that λ is even, even integers from $n\lambda$ to $n(\lambda + 2) - 1$ are sequentially assigned to edges $(i, i+2\lambda+1)$ starting from $i = 0$. A similar assignment starting from $i = \lfloor \frac{n}{2} \rfloor$ is applied when λ is odd.

The above rule is for all possible λ except for $\lambda = k - 1$ and k is odd. In this case, we have n remaining unlabeled edges with n unassigned integers from $n(k-1)$ to $nk-1$. All even integers among those unassigned numbers are assigned sequentially to the first edges. Remaining integers are assigned to remaining edges in decreasing order.

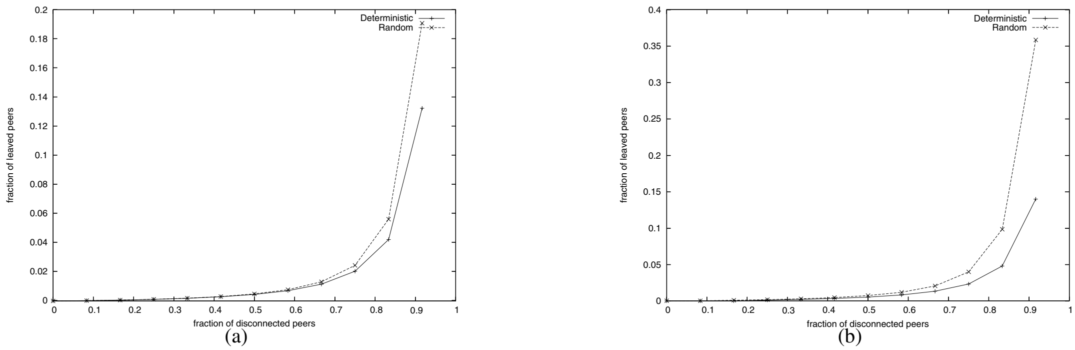
For illustration, we apply our labeling scheme to two example genGraphs: $n = 12$, $k = 2$ and $n = 12$, $k = 3$. Adjacent matrices of those graphs are shown in figure 4(a) and 4(b). In those matrices, column and row indices are vertex labels, and entries corresponding to edges of the genGraph are filled with edge labels. Entries that represent no edges are set to null and depicted by stars in the figures. Arrows depict the order of labeling.

IV. Evaluation of Failure Resiliency

Investigating the robustness of DPMGs against peer failures that occur locally, we assume α percent of random peers at a particular level i_0 leaves and evaluate the fraction D of disconnected peers in all lower levels. DPMGs constructed deterministically with our genGraph construction algorithm and edge-labeling algorithm are compared to random DPMGs in terms of resiliency. Random DPMGs are built by uniformly selecting from the upper level two parents for each peer. For fair comparisons, a deterministic DPMG is compared with a random DPMG that has the same general structure. That is the two DPMGs have the same expansion factor k and have the same number of peers at the top level i_0. The expansion factor k in our simulations, is set to 3, there are 12 peers at the top level and the depth of DPMGs are 5. Those setups form a network with more than 4000 nodes.

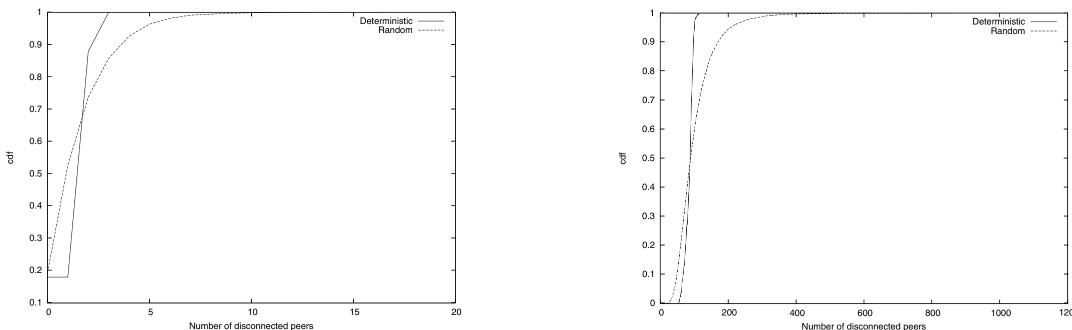
Comparing deterministic and random DPMGs, figure 5(a) and 5(b) show fraction of disconnected peers as a function of fraction of leaved peers at the top level i_0 for the average case and the 90-percentile case respectively. When there are just few peers failed at a level (α is small), on average, both random and deterministic DPMGs cause similar fractions of disconnected peers. When α increases, deterministic DPMG has better resiliency than random ones. And when all peers at a level fail, the whole network is corrupted for both random and deterministic DPMGs. However, as long as α is less than 1, deterministic DPMGs are always better than random DPMGs.

Deterministic DPMGs also provide statistically good resiliency. Figure 6(a) and 6(b) show cumulative probability functions of the number of disconnected peers for small α (α = 0.25) and for large α (α=0.75). The cdf in the case of deterministic DPMGs always increases sharply. It reveals that with the deterministic method the number of disconnected peers caused by locally-occurred failures is densely distributed around the expectation. Thus, it guarantees the worst case



(a) (b)

Fig. 5. Average value (a) and 90-percentile (b) of the fraction of disconnected peers D as a function of leaved peers fraction α



(a) $\alpha = 0.25$ (b) $\alpha = 0.75$

Fig. 6. Cdf of the number of disconnected peers

where a plenty of peers are disconnected due to just few peer failures never happens.

V. Conclusion

We have presented DPMG, a method for selecting multiple parents per peer in a resilient peer-to-peer system. This method can be used to construct overlay multicast networks for continuous streaming applications. The approach used in DPMG is a deterministic construction of multicast graphs level by level. We have shown that our approach improves resiliency when network failures are locally occurred on the logical structure of the peer-to-peer network.

References

[1] W. Wang, D. Helder, S. Jamin, and L. Zhang, *Overlay Optimizations for End-host Multicast*. In Proc. NGC, Boston, MA, USA, Oct. 2002.

[2] Y. Chu, S. Rao, and H. Zhang. *A Case for End-System Multicast*. In Proc. ACM SIGMETRICS, Santa Clara, CA, USA, June 2000.

[3] Y. Chawathe. *Scattercast: An Architecture for Internet Broadcast Distribution as an Infrastructure Service*. PhD thesis, University of California, Berkeley, USA, Dec. 2000.

[4] B. Zhang, S. Jamin, and L. Zhang. *Host Multicast: A Framework for Delivering Multicast to End Users*. In Proc. IEEE INFOCOM, New York, NJ, USA, June 2002.

[5] Jeremy I. Silber, Sambit Sahu, Jatinder Singh and Zhen Liu. *Augmenting Overlay Trees for Failure Resiliency*. Globecom 2004. June 2004.

[6] I. Stoica, R. Morris, D. Karger, M. Kasshoek, and H. Balakrishnan. *Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications*. In Proc. ACM SIGCOMM, San Diego, CA, USA, Aug. 2001.

[7] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, *SplitStream: High-bandwidth multicast in a cooperative environment*. In Proceedings of SOSP'03, New York, USA, October 2003.

[8] D. Kostic, A. Rodriguez, J. Albrecht, and A. Vahdat, *Bullet: High Bandwidth Data Dissemination Using an Overlay Mesh*, in Proc. of ACM SOSP, 2003.

[9] T. Nguyen and A. Zakhor, *Distributed Video Streaming with Forward Error Correction*, in Proc. Packet Video Workshop, Pittsburgh, USA, April 2002.

[10] V. Padmanabhan, H. Wang, P. Chou, and K. Sripanidkulchai. *Distributing Streaming Media Content Using Cooperative Networking*. In Proceedings of NOSSDAV'02, Miami Beach, FL, May 2002.

[11] M. Bansal and A. Zakhor, *Path Diversity for Overlay Multicast Streaming*, Packet Video Workshop, Irvine, CA, Dec. 2004.

[12] Goyal, V.K. *Multiple description coding: compression meets the network*. Signal Processing Magazine, IEEE, Vol.18, Iss.5, Sep 2001 Pages:74-93

[13] A.J.Ganesh, A.-M Kermarrec, and L. Massoulie. *SCAMP: Peer-to-peer lightweight membership service for largescale group communication*. In Proceedings of the 3rd International workshop on Networked Group Communication, 2001.

[14] A. Rowstron and P. Druschel, *Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems*. IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), Heidelberg, Germany, pages 329-350, November, 2001.

[15] B. Cohen. *Incentives build robustness in BitTorrent*. In Proceedings of the First Workshop on the Economics of Peer-to-Peer Systems, Berkeley, CA, June 2003.

[16] S. Banerjee, S. Lee, B. Bhattacharjee, and A. Srinivasan. *Resilient Multicast using Overlays*. In SIGMETRICS, 2003.

[17] Dejan Kostic, Adolfo Rodriguez, Jeannie Albrecht, Abhijeet Bhirud, and Amin Vahdat, *Using Random Subsets to Build Scalable Network Services*, Proceedings of the USENIX Symposium on Internet Technologies and Systems, March 2003.