Improving Reliability for Application-layer Multicast Overlays

Ye Tian, *Member, IEEE,* Hong Shen, and Kam-Wing Ng

Abstract—Reliability of tree-like multicast overlays caused by nodes' abrupt failures is considered as one of the major problems for the Internet application-layer media streaming service [1]. In this paper, we address this problem by designing a distributed and light-weighted protocol named the instantaneous reliability oriented protocol (*IRP*). Unlike most of existing empirical solutions, we first define the overlay reliability problem formally, and propose a protocol containing a node joining algorithm (*IRP-Join*), a node preemption algorithm (*IRP-Preempt*), and a node switching algorithm (*IRP-Switch*) for reactively constructing and repairing the overlay, as well as proactively maintaining the overlay. With the formal problem presentation, we set up a paradigm for solving the overlay reliability problem by theoretically proving the effectiveness of our algorithms. Moreover, by comparing IRP with existing solutions via simulation-based experiments and real-world deployment, we show that IRP achieves a better reliability, while incurs fewer structural adjustments on the multicast overlay, thus providing a superior overall performance.

Index Terms—Reliability, multicast, algorithm/protocol design and analysis.

1 Introduction

With the wide deployments of broad band technologies and due to the insufficient infrastructure support of IP-Multicast, tremendous amount of work has been carried out on building application-layer multicast systems for providing end users the media streaming service. According to their overlay structures, existing works could be classified as tree-like systems and mesh-like systems. The former ones include ESM[2], NICE[3], SCRIBE[4], and ZigZag[5]. In these systems, nodes are organized in a spanning tree, and the media data is pushed from parent node to child nodes. For fully utilizing nodes' bandwidths, systems containing multiple trees are also designed, such as CoopNet[6], SplitStream[7], and Chunkyspread[8]. On the other hand, data driven based mesh-like overlays have been proposed recently for media multicasting, examples include CoolStreaming[9] and PRIME[10], and many successful commercial applications such as PPLive[11] also adopts this mesh-like design. Compared with the tree-like design, the mesh-like overlay is more resilient against node failures, but it incurs additional control overhead with the data driven technique. In some hybrid systems such as Bullet[12] and mTreebone[13], a tree-like overlay is used as the backbone network for media streaming, while data-driven mesh links are used for enhancing the system's failure resilience.

- *Y. Tian and H. Shen are with the Anhui Province Key Laboratory on High Performance Computing, School of Computer Science and Technology, University of Science and Technology of China, Hefei, Anhui 230026, China. H. Shen is also with School of Computer Science, University of Adelaide, Australia.*
  E-mail: {yetian, hongshen}@ustc.edu.cn.
- *K.-W. Ng is with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong, China. E-mail: kwng@cse.cuhk.edu.hk.*

In this paper, we focus on a typical tree-like application-layer multicast system like the one in [2], and address its reliability issue. The reliability problem arises because when a node on a multicast tree fails, especially when it fails abruptly without notifications, all the nodes receiving media directly and indirectly from it will lose their streaming services. In media streaming, it is unacceptable if such incident happens frequently, hence a well designed protocol is required for avoiding the service interruptions experienced by users caused by node failures.

In recent years, different approaches have been proposed for the tree-like multicast overlay's reliability problem. In particular, the *Min-Depth* scheme widely adopted [3] [6] [5] and studied [14] tries to construct a stable overlay by building a compact multicast tree on node joining events, and the *Preempt-Degree* scheme [15] makes the same efforts on node failures. The recently proposed *ROST* algorithm [16] exploits an age-bandwidth tradeoff by switching parent-children pairs on the overlay for enhancing its failure resilience. However, it is noticed that all these existing approaches are empirical, and their effectiveness are testified mainly with experiments. Moreover, there is a lack of theoretical work on formally understanding and presenting the overlay reliability problem, and theoretically verifying the effectiveness of the solutions. Based on these observations, in this study we will answer two questions: 1) what is overlay reliability? and 2) how to improve the overlay's reliability? We first formally define the overlay reliability problem for the tree-like application-layer multicast overlay, then present a distributed and light-weighted protocol named the instantaneous reliability oriented protocol (IRP) for improving the overlay's reliability. We also theoretically prove that the protocol is effective with the formal presentation of the overlay

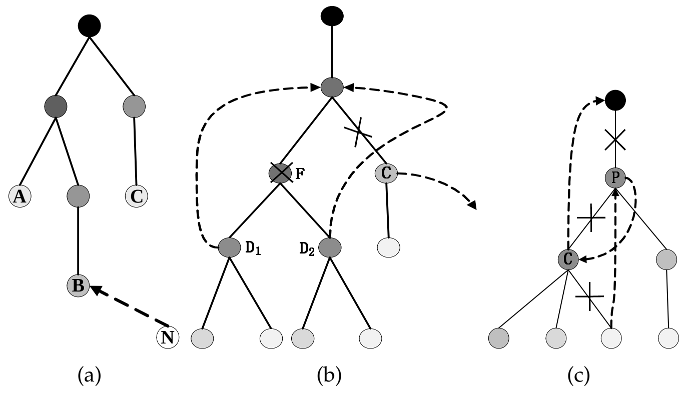


Fig. 1. Examples demonstrating (a)IRP-Join, (b)IRP-Preempt, and (c)IRP-Switch, where darker node has stayed longer on the overlay.

reliability problem.

Our proposed IRP protocol is composed of three algorithms, namely IRP-Join, IRP-Preempt, and IRP-Switch. The algorithms of IRP-Join and IRP-Preempt are used to handle the node joining and node failure events[1] respectively, while IRP-Switch proactively adjusts the overlay structure periodically. For example, in Fig. 1(a), when a new node N joins, it runs IRP-Join to evaluate potential parent nodes (i.e., A, B, and C) for their fitness in providing a reliable streaming service, and selects the best one (i.e., B) to connect to. Fig. 1(b) demonstrates IRP-Preempt: when a node F fails, its child nodes D_1 and D_2 (together with their descendent nodes) run IRP-Preempt to rejoin the overlay. In the figure, D_2 rejoins by preempting a connected node C from its parent node and replaces its position, while the preempted node C rejoins the overlay by following the same procedure in IRP-Preempt. Fig. 1(c) shows how IRP-Switch works, where a pair of parent-child nodes, P and C, switch their positions according to certain criteria for improving the overlay's reliability. In the figure, C replaces P by connecting directly to the source node, and P connects to C as a child node. Obviously, whether in IRP-Join, IRP-Preempt, or IRP-Switch, a node needs to make critical decisions such as which node to connect to, preempt, or make a switch with. One of the most significant contribution in this paper is that we propose a number of metrics assisting nodes to make these decisions, and design a mechanism for them to obtain and propagate the metrics in distributed and inexpensively ways.

Finally, through the experiments based on simulation and PlanetLab deployment , we show that IRP has a superior performance regarding overlay reliability compared with existing solutions, and its reliability is achieved with fewer adjustments on the overlay structure; we also find that each algorithm of the protocol works better than its existing solution counterpart. Other observations from the experimental studies are: the proactive node switching algorithm could be configured as optional and be executed infrequently; the overlays under IRP have properties of small service latencies and overlay stretches; IRP could be applied on overlays containing multiple trees directly with good performance.

1. By "node failure", we mean not only the physical failure of the node, but also the traffic congestion at the node that causes it unable to provide the required QoS.

The remainder of this paper is organized as follows: Section 2 reviews and discusses related work; Section 3 formally defines the problem of overlay reliability; Section 4 proposes the IRP protocol, and the effectiveness of its three algorithms are analyzed in Section 5; Section 6 discusses the experiment setup and analyzes the experimental results from simulation and real-world deployment; Finally, Section 7 concludes the paper.

2 Related Work

Existing solutions on improving the tree-like application-layer multicast overlay's reliability could be classified into two types: proactive ones and reactive ones. In the reactive solutions, adjustments on the overlay's structure are made only on events of node joining and failure, while in the proactive ones, the multicast overlay is periodically adjusted proactively.

We first look at the reactive solution. In a multicast overlay, when a new node joins, it relies on a bootstrap mechanism for locating its parent. In detail, after contacting a bootstrap node for obtaining a number of online nodes called neighbors, the new joining node selects one among them with spare outgoing bandwidth as its parent. Such a bootstrap mechanism is usually implemented with random sampling techniques such as RanSub[17] and RandPeer[18]. For parent selection, many systems such as NICE[3], CoopNet[6], and ZigZag[5] apply a *Min-Depth* scheme, where the new joining node selects the neighbor of the minimum depth as its parent. Moveover, in [14], a systematic comparison is conducted among a number of node joining schemes, and it is found that Min-Depth outperforms other schemes such as selecting the oldest node or random selection.

When a node on the multicast overlay fails, all its child nodes will lose their connections and need to reconnect to the overlay. One possible solution is that these disconnected nodes rejoin as new nodes [2]. However, this scheme will increase the overlay's depth and degrade its reliability. In [15], a preemption operation is discussed where the disconnected node follows certain criteria to preempt (i.e., replace) an online node even though this node is connected, and the preempted node rejoins the overlay by following the same procedure. Moreover, [15] reports that the *Preempt-Degree* scheme, in which a node with a larger degree (i.e. outgoing bandwidth) preempts a node with a smaller one, outperforms other node preemption schemes such as Preempt-Age (comparing nodes' ages) or No-Preempt. In [19], it is shown that by actively estimating the nodes' lifetime model, stable overlays could be constructed and maintained with reactive approaches on node joining and failure events.

On the other hand, the recently proposed *ROST*[16] algorithm belongs to the proactive approach, where a tradeoff between a node's bandwidth and age is exploited. In ROST, each node calculates the product of its age and outgoing bandwidth, when a node finds that its product is larger than the product of its parent, and it has a larger out-going bandwidth, it makes a switch with its parent node. It is shown that by making this proactive adjustment periodically, nodes with larger bandwidths will get moved gradually to higher positions, and the overlay is of better fault resilience than the bandwidth-optimized overlays and the time-optimized overlays defined in [16].

Recently, N. Magharei et al. [20] showed that the mesh-like overlay is more stable by comparing a typical mesh-like overlay with a typical multiple-tree overlay with the *interior disjoint policy* engaged.

Constructing a reliable multicast tree topology is also extensively studied in IP-Multicast. However, in general the nodes in IP-Multicast are routers that fail infrequently, so the major concern for reliability is to reduce the service disruptions caused by failures of the physical links/nodes on the multicast tree. In most cases, the reliability problem is usually studied together with the goals of satisfying delay constraints and minimizing the costs. For example, [21] took reliability as a QoS requirement in setting up paths between source and destination. [22], [23], and [24] addressed the problem of dynamically rearranging the IP-multicast topology for enabling node joining and leaving during the multicast session while preserving the streaming quality. However, as IP-multicast and application-layer multicast have completely different rationales and architectures[1], existing approaches for improving the IP-Multicast topology's reliability can not be applied under the context of application-layer multicasting.

Our work differs from existing works in that in our work, we formally present the overlay reliability problem, and propose a new solution with theoretical proofs for its effectiveness. To our best knowledge, this is the first solution with its effectiveness theoretically verified. Moveover, our protocol does not require any detailed knowledge on nodes' lifetime model, and has a superior performance compared with existing solutions.

3 The Overlay Reliability Problem

In this subsection, we formally define the overlay reliability problem and present some related concepts. Table 1 lists the denotations used throughout this paper.

3.1 Definition and denotation

Unlike IP-multicast, in an application-layer multicast system, a node is a self-interest agent, which may subscribe and withdraw the streaming service at any time. A node's "lifetime" is defined as the time between its joining and leaving of the multicast overlay. For the application-layer multicasting, when the user behind a node on the overlay quits, the node fails, and all its descendant nodes will lose their streaming services. In this work, we consider node failure as the only reason causing loss of streaming service, but ignore the link failure, as a link fails much more infrequently than an application-layer node. For example, the mean time between physical link failures is in tens of days[25], while an application-layer node's lifetime is only in minutes or hours[26][27]. Moreover, application-layer nodes can use techniques such as multihoming to improve the reliability of the unicast path between them.

TABLE 1
Denotation

Denotation	Meaning
$U^N(n,t)$	node n's node unreliability metric at time t
$U^P(n,t)$	node n's path unreliability metric at time t
$U^O(T,t)$	overlay T's overlay unreliability metric at time t
$U^O_{in}(T(n),t)$	subtree $T(n)$'s internal overlay unreliability metric at time t
$par(n)$	parent node of node n
$T(n)$	subtree rooted at node n
$\Phi(n)$	num. of node n's descendant nodes

For understanding the overlay reliability problem, first of all a metric is required to describe a node's reliability status, and based on this metric the overlay reliability problem could be defined and solved. The most direct way is to use the node's failure probability as the metric. If we assume that all the nodes have an i.i.d lifetime distribution, with the cumulative distribution function (CDF) expressed as $F(x) = \Pr\{l < x\}$, then for a node which has joined the overlay for time x (x is referred to as the node's age), the probability that it fails after time y is $U(x,y) = \Pr\{l < x + y | l \geq x\}$. However, $U(x,y)$ is not suitable to be used directly as a metric for two reasons: First, $U(x,y)$ depends on two variables, x and y, although the age x is deterministic for a specific node, y could take any positive value. When used as a metric, how to determine y is problematic. More importantly, the calculation of $U(x,y)$ requires the knowledge of node's lifetime model $F(x)$, but in real-world application-layer multicasting, it is infeasible for an individual node to know $F(x)$.

If we rewrite y in $U(x,y)$ as Δt, and apply the Bayes' theorem, then we have

$$\begin{aligned}\lim_{\Delta t \to 0} \frac{U(x,\Delta t)}{\Delta t} &= \lim_{\Delta t \to 0} \frac{\Pr(l < x + \Delta t, l \geq x)}{\Pr(l \geq x)\Delta t} \\ &= \lim_{\Delta t \to 0} \frac{F(x+\Delta t) - F(x)}{(1 - F(x))\Delta t} \\ &= \frac{f(x)}{1 - F(x)} = h(x)\end{aligned}$$

where $f(x)$ is the probability density function (pdf), $f(x) = \frac{dF(x)}{dx}$. One can see that $h(x)$ actually is the hazard rate widely used in reliability analysis[28]. In other words, $h(x)$ describes the instantaneous rate of failures for nodes at age x. Unlike $U(x,y)$, $h(x)$ depends only on a node's current age x, making it more suitable to be used as a metric. Moreover, we will see in Section 6 that $h(x)$ could be well estimated without knowing $F(x)$ under the context of application-layer multicasting. Finally, we stress that here we do not intend to obtain the accurate value of the probability or rate of node failure, but to find a suitable metric for formally describing the overlay reliability problem. This metric could be used to assist individual node in making proper decisions when constructing and adjusting the overlay in distributed and inexpensive ways.

Based on the above arguments, for an individual online node n, we use the term *node unreliability metric* for describing its reliability status at time t, which is defined as its instantaneous hazard rate, expressed as

$$U^N(n,t) = h(x) = \frac{f(x)}{1-F(x)}$$

where x is node n's age at time t. For the root node, its node unreliability metric is defined as 0 all the time.

We consider a path denoted as $\{n_0, n_1, ..., n_{s-1}, n_s\}$ on a multicast tree, where n_0 is the root node and n_s is the last node along the path (but n_s is not necessarily a leaf node). For the nodes $n_1, n_2, ..., n_s$ with their ages $x_1, x_2, ..., x_s$ at time t respectively, the probability that none of them fails could be expressed as $S(x_1, x_2, ..., x_s) = \prod_{i=1}^{s}(1-F(x_i))$. We define the *path unreliability metric* for the node n_s at time t as the instantaneous failure rate of any node on the path[2], which could be calculated as

$$\begin{aligned} U^P(n_s,t) &= \frac{\sum_{i=1}^{s}\frac{\partial(1-S(x_1,x_2,...,x_s))}{\partial x_i}}{S(x_1,x_2,...,x_s)} \\ &= \sum_{i=1}^{s}\frac{f(x_i)}{1-F(x_i)} \\ &= \sum_{i=1}^{s}U^N(n_i,t) \end{aligned}$$

We can see that the path unreliability metric actually is the sum of the node unreliability metrics for all the nodes on the path, which conforms to the concept of the instantaneous failure rate.

For a node n on the overlay, a failure of any of its ancestors will cause an interruption on its streaming service, and we can express the rate of such failures as $U^P(par(n), t)$, where $par(n)$ stands for n's parent node. For the entire overlay, we define its *overlay unreliability metric* at time t as the sum of the instantaneous ancestor failure rates for all the nodes on the overlay, i.e., for an overlay T, its overlay unreliability metric at time t is

$$U^O(T,t) = \sum_{\forall v \in T} U^P(par(n),t)$$

2. Generally speaking, the path unreliability metric should be a property associated with a path instead of a node, however, on a multicast tree, as a path can be uniquely determined by the end node of the path, we consider that the path unreliability metric is a property of the end node for simplicity.

With the above definitions, one can see that for a particular node, its individual node unreliability metric will be considered as many times as its number of descendant nodes when calculating the overlay's overlay unreliability metric, therefore the expression of $U^O(T,t)$ could be rephrased as

$$U^O(T,t) = \sum_{\forall n \in T}(U^N(n,t) \cdot \Phi(n))$$

here $\Phi(n)$ is the number of n's descendant nodes on the overlay T.

We may also consider how a part of the multicast tree contributes to $U^O(T,t)$ of the entire overlay. Specifically, for a non-leaf node n_r, we consider the subtree $T(n_r)$ containing all n_r's descendant nodes with n_r as its root. The overlay unreliability metric could be re-expressed by separating $T(n_r)$'s contribution as

$$\begin{aligned} U^O(T,t) &= U^O(T-T(n_r),t) \\ &+ U^P(par(n_r),t)\cdot(\Phi(n_r)+1) + U^O_{in}(T(n_r),t) \end{aligned}$$

Here $U^O(T-T(n_r),t)$ is the overlay unreliability metric for the part of the overlay without $T(n_r)$ (which is also a multicast tree), and $U^O_{in}(T(n_r),t)$ is the *internal overlay unreliability metric* for the subtree $T(n_r)$, which is defined as the overlay unreliability metric of a hypothetic overlay containing only the root node and the subtree $T(n_r)$, where n_r is the only child node of the root.

3.2 The problem

As application-layer multicasting is concerned in this paper, we consider the fact that nodes' lifetimes are heavy-tailed, which is widely observed in recent studies on real-world applications [26] [27]. For heavy-tailed lifetime, it is well known that "the longer a node stays, the more reliable it is". With our denotations, this knowledge could be expressed as: $U^N(n,t) \geq U^N(n,t+\Delta t)$, for $\Delta t \geq 0$. From the definition of the overlay unreliability metric, we also have: $U^O(T,t) \geq U^O(T,t+\Delta t)$, for $\Delta t \geq 0$, given that the overlay structure has not been changed during the interval $[t, t+\Delta t)$.

For an application-layer multicast overlay, events of node joining and failure will force the overlay to reactively change its structure. Under the heavy-tailed node lifetime model, if an optimal overlay regarding reliability is constructed on these events, the overlay's unreliability metric will continue to decrease all the time until the next node joining or failure event occurs, therefore we may choose to construct an overlay with the minimum overlay unreliability metric at these opportunities. On the other hand, even if the overlay is constructed optimal at these events, we can not ensure that it will continue to be optimal all the time. If some proactive overlay maintenance mechanism similar to ROST is introduced, we have chances to adjust the overlay structure for better reliability.

However, one important constraint in this problem is the feasibility of the operations performed for changing the overlay. For example, on reactive or proactive opportunities, if we tear down all the edges of the multicast tree and re-construct the overlay completely, clearly an optimal overlay could be obtained, however, this is impractical under application-layer media streaming as all the nodes will lose their services. In this paper, we only consider the feasible operations which have been applied in previous works. Concretely, on node joining events, we only consider to join the new node as a leaf node, as in [14]; on node failures, we only consider the preemption operations performed in [15]; while for proactive overlay maintenance, we only consider to switch the parent-child node pairs as in [16].

Summarizing all these discussions, we can formally define the overlay reliability problem studied in this paper as follows: *given the feasible operation constraints of node joining, node preemption, and node switching above mentioned, we seek to minimize the overlay unreliability metric (i.e. $(U^O(T,t))$) at each proactive or reactive overlay structure modifying opportunity*. In addition, for practical application-layer multicasting, the solution must have the following properties:

- Distributed protocol: In an application-layer multicast system, it is impractical for an individual node to have an up-to-date knowledge of the entire overlay, therefore a distributed solution is required for the overlay reliability problem.
- Few adjustments on the overlay structure: To achieve a reliable overlay against abrupt node failures, sometimes we need to make adjustments such as parent re-selection on the overlay's structure. However, such adjustments themselves will cause disturbances on the on-going streaming services. In our solution, we wish to make as few adjustments as possible.
- Light overhead: To preserve reliability of a multicast overlay, nodes need to exchange some information for making their decisions, thus additional control overhead will be incurred. We require that our solution should be inexpensive regarding this overhead.

As time is not revelent to the unreliability metrics, for the remainder of this paper, we will omit t. For example, we will simply use $U^O(T)$ to denote the overlay T's current overlay unreliability metric under consideration.

4 The Instantaneous Reliability oriented Protocol (IRP)

In this section, we propose a protocol for improving the tree-like application-layer multicast overlay's reliability. We name our proposal as instantaneous reliability oriented protocol (*IRP*) for the reason that the protocol is based on the concept of instantaneous hazard rate. Before describing the protocol, first we assume that a random sampling service (e.g., RanSub[17] and RandPeer[18]) is available, where each node on the overlay keeps a random subset of online nodes as its neighbors. As we have discussed, a new joining node uses its neighbor nodes to locate its parent; in addition, during node preemption, a disconnected node or a preempted node also selects its potential parent among its neighbors.

4.1 Node reliability index (NRI), path reliability index (PRI), and descendant number

In IRP, we require that each node on the multicast overlay maintains three values: 1) its node reliability index (NRI); 2) its path reliability index (PRI); and 3) the number of its descendant nodes. For a node n's NRI, referred to as $NRI(n)$, it only depends on node n's age and reflects its individual node reliability metric. The NRI of the root node is defined as 0. A detailed NRI setting could be found in Section 6.1. A node's PRI is the sum of the NRIs for all its ancestor nodes, and from NRI's property, it is easy to see that a node's PRI also reflects its path reliability metric.

It is inexpensive for a node to obtain and update its PRI, as only its ancestor nodes are involved. Here we use a scheme called *lazy update*. In lazy update, a node requests and keeps the ages for all its ancestor nodes when connecting to a new parent, and calculates its PRI by figuring out the NRIs for all its ancestors. For updating the NRIs, the node only needs to add the elapsing time to the previously obtained ages of its ancestor nodes, and updates their NRIs locally. We can see that in lazy update, a node only needs to communicate with each of its parent once, therefore the cost is small. Please note that a similar approach could be applied for a node to obtain and update its depth.

For obtaining the descendant node number, a node may query all its child nodes for their descendant numbers, and sums them up to get its own descendant number. Here we also apply a lazy update scheme: when a node finds some changes among its children, for example, when it has a new child node or one of its child nodes has left, it updates its descendant number, and reports to its parent. On receiving a new descendant number report from its child node, a node checks whether or not there is a change for its own descendant number, if yes, it reports the new number to its parent. This procedure repeats until the root node is reached. However, in this scheme, it is possible for a node at a high position to receive reports frequently, as it has a large number of descendant nodes. To avoid overloading these high position nodes, two strategies could be applied: first, a node is allowed to send only one report within a certain period; second, a node checks the difference between its current descendant number and the number in its last report when there is some change, and reports to its parent node only when the difference exceeds a certain threshold.

With each node keeping its NRI, PRI, and descendant number, we describe our solution for improving the multicast overlay's reliability. Our solution is composed of three algorithms, namely the instantaneous reliability oriented node joining algorithm (*IRP-Join*), the instantaneous reliability oriented node preemption algorithm

Algorithm 1 *IRP-Join(n_j)*

```
1.  $n_j$ contacts the bootstrap node for its neighboring node set $R$;
2.  For each eligible node $n_i \in R$, $n_j$ queries for $n_i$'s PRI;
3.  $n_j$ connects to the node $n_i^*$ with the minimum PRI as its parent;
4.  Return.
```

Algorithm 2 *IRP-Preempt(n_d)*

```
1.  $n_d$ ranks all its neighboring nodes in an ascending order according to PRIs;
2.  For the $i$ th node $n_i$
3.      If $n_i$ could support one more children
4.          $n_d$ connects to $n_i$ as its parent;
5.      Else
6.          For each children node $n_{ic}$ of $n_i$
7.              If $\Phi(n_d) > \Phi(n_{ic})$
8.                  $n_d$ preempts $n_{ic}$, and connects to $n_i$ as its parent;
9.                  IRP-Preempt($n_{ic}$);
10.                 Return.
```

(*IRP-Preempt*), and the instantaneous reliability oriented node switching algorithm (*IRP-Switch*).

4.2 The IRP-Join algorithm

The IRP-Join algorithm is very simple. In this algorithm, after obtaining its neighbors from the bootstrap node, a new joining node selects an eligible neighbor node (a node which can support one more child) with the minimum PRI as its parent. If there is no eligible node, the new joining node contacts the bootstrap node again for renewing its neighbor set. A formal description of IRP-Join could be found in Algorithm 1.

4.3 The IRP-Preempt algorithm

In our proposed IRP-Preempt algorithm, a disconnected node rejoins the overlay by trying to connect to one of its neighbors as its new parent. Similar to IRP-Join, the disconnected node first tries on the neighbor with the minimum PRI as its potential parent, if the neighbor could support one more child, the disconnected node successfully rejoins the overlay; otherwise, the disconnected node tries to preempt one of the neighbor's current children by comparing their descendant numbers: if the disconnected node has more descendants than the child node, the preemption succeeds, and the preempted node follows the same procedure to rejoin the overlay. If none of the child nodes could be preempted, the disconnected node tries on the next neighbor with the second minimum PRI. The disconnected node tries on all its neighbors until it successfully reconnects to the overlay. If it fails with all its neighbors, it contacts the bootstrap node for renewing its neighbor set, and tries again. A formal description of IRP-Preempt could be found in Algorithm 2.

In node preemptions, when a node has been preempted, its former parent will still send media data to it, until it could receive a full-rate streaming from its new parent, in this way we avoid service interruptions

Algorithm 3 *IRP-Switch(n_p,n_c)*

```
1.  On timeout
2.      If ($n_c$>$n_p$)
3.          $n_p$ calculates $c1$ as in Eq. (1) and $n_c$ calculates $c2$ as in Eq. (2);
4.          If $c1<c2$
5.              $n_c$ connects to $n_p$' parent as its parent;
6.              all $n_p$'s children nodes except for $n_c$ connect to $n_c$ as their parent;
7.              $n_p$ connects to $n_c$ as its parent node;
8.              $d_p$ of $n_c$'s former children with the fewest descendants connect to $n_p$
as their parent;
9.  Restart the timer.
```

caused by preemptions. In case that more than one nodes try to preempt the same online node simultaneously, we let the online node to make the decision. Concretely, the online node will get preempted by the node with the largest number of descendant nodes, and rejects the other nodes' preemption requests.

4.4 The IRP-Switch algorithm

In this subsection, we describe the proactive overlay maintaining algorithm named IRP-Switch. In IRP-Switch, for a parent node n_p and a child node n_c, with their degrees as d_p and d_c respectively, if $d_p < d_c$, then the parent node n_p calculates a value as

$$c1 = (NRI(n_c) - NRI(n_p)) \cdot (\Phi(n_p) - \Phi(n_c)) \quad (1)$$

For the child node n_c, as it has $(d_c - d_p)$ more child nodes than n_p, it selects $(d_c - d_p)$ of its child nodes with the largest numbers of descendant nodes, and calculates a value as

$$c2 = PRI(n_p) \cdot (\sum_{i=1}^{d_c - d_p} (\Phi(n_{c,i}) + 1)) \quad (2)$$

where $\Phi(n_{c,i})$ is the number of the descendant nodes for n_c's ith child node with most descendants. In IRP-Switch, n_p and n_c calculate and compare $c1$ and $c2$ periodically, and get switched when $c1 < c2$. In detail, n_c takes n_p's position by connecting to n_p's parent and being connected by n_p's child nodes except for itself, while n_p becomes n_c's child; moreover, as n_c has $(d_c - d_p)$ more child nodes than n_p, it keeps $(d_c - d_p)$ of its former children with the most descendants as its current children, but its other d_p child nodes connect to n_p as their new parent. A formal description of the algorithm could be found in Algorithm 3.

In IRP-Switch, we could also use the same method as in IRP-Preempt for avoiding service interruptions, where a node receives media data from its original parent until it could receive a full-rate streaming from its new parent. For the case that more than one child nodes try to switch a same parent simultaneously, the parent node will make the decision. Concretely, the parent node compares the difference of $(c2 - c1)$ for each requesting child node, and only gets switched with the one of the maximum difference.

4.5 Cheating robustness

In previous discussions, we assumed that all the nodes are honest. However, as a self-interest agent, a node on the application-layer multicast overlay may have incentives to cheat. There are at least two possible cheating schemes under our protocol. In one scheme, a node cheats on its individual information to influence other nodes to make decisions which are beneficial to itself. For example, a disconnected node may pretend to have a large number of descendants in order to perform a successful preemption. In the other scheme, a decision-making node may cheat if the future overlay adjustment is not beneficial to itself. For example, a parent node may reject switch requests from all its child nodes, even though some are eligible to make a switch. Recently many mechanisms are designed for safeguarding self-managing networks against cheating behaviors, such as the referee system proposed in [16] and the SybilGuard system [29], and these solutions could be integrated in our protocol without major modifications. However, as this issue is not closely related to the focus of this paper, we do not discuss it any further, but assume that a powerful anti-cheating mechanism is available, and nodes are honest in our study.

4.6 Incorporating other QoS concerns

Although we focus on improving the application-layer multicast overlay's reliability, IRP is able to work with other QoS concerns, such as the service latency and the media resolution. A simple way for incorporating these concerns is to set QoS constraints, where future overlay adjustments are allowed only when these constraints are satisfied. For example, a node may only consider the nodes with a better resolution than a threshold as its potential parent in IRP-Preempt. Moreover, we will see in Section 6.6 that our protocol preserves good properties for the overlay regarding the service latency and overlay stretch, although they are not our design objectives.

5 Analysis

In the previous section, we have described our proposed IRP protocol with the IRP-Join, IRP-Preempt, and IRP-Switch algorithms for improving the application-layer multicast overlay's reliability. In this section, we show that these algorithms are effective under the consideration of the overlay unreliability metric defined in Section 3. Moreover, by proving their effectiveness, we wish to set up a paradigm for designing and testifying future solutions of the overlay reliability problem.

Before presenting the proofs, first we make an assumption on node reliability index (NRI): *for a node n, $NRI(n)$ is proportional to its node unreliability metric $U^N(n)$*. With this assumption, it is very easy to see that $PRI(n)$ is also proportional to node n's path unreliability metric $U^P(n)$. In Section 6.1, we will show how to statistically satisfy this assumption by exploiting recent studies on real-world Internet media streaming.

We prove the effectiveness of the IRP-Join algorithm in the following.

Theorem 1: Suppose a new joining node n_j has the knowledge of the entire overlay. By running the IRP-Join algorithm, the multicast overlay formed after n_j's join is of the minimum overlay unreliability metric.

Proof: As n_j has newly joined, we assume that its age $x_j \to 0$, therefore $U^N(n_j) \to \infty$. First we consider the case that n_j is placed at a non-leaf position, which means that there are at least one node as n_j's descendant. According to our discussion in Section 3, $U^N(n_j)$ will be counted into the overlay unreliability metric at least once, hence $U^O(T) \to \infty$. Next we consider the case that n_j joins as a leaf node. Suppose n_j connects to a node n_p as its parent, and the path from the root node n_0 to n_p is $\{n_0, n_1, ..., n_p\}$. Before n_j joins, let the overlay's overlay unreliability metric be $U^O(T_{prev})$. After n_j has joined, nodes $n_1, ..., n_p$ each has one more descendant node, i.e., n_j, therefore the overlay unreliability metric becomes

$$U^O(T_{after}) = U^O(T_{prev}) + \sum_{i=1}^{p} U^N(n_i)$$

The difference on the overlay unreliability metrics after and before n_j's join, $\sum_{i=1}^{p} U^N(n_i)$, actually is node n_p's path unreliability metric $U^P(n_p)$. As a node's PRI is proportional to its path unreliability metric, and n_j connects to the node with the minimum PRI in IRP-Join, clearly $U^O(T_{after})$ is minimized. □

Following a similar idea to that in the proof of Theorem 1, we can see that with partial knowledge of the entire overlay for n_j, the overlay formed after n_j's joining is of the minimum overlay unreliability metric among all its possible choices.

For proving the effectiveness of the IRP-Preempt algorithm, we first assume that after a node n_d becomes disconnected due to its parent's failure, it temporarily connects to a leaf node n_l with the maximum PRI among all the nodes on the overlay. We refer to this hypothetic overlay as T_1. For IRP-Preempt, we have the following result.

Theorem 2: By running the recursive IRP-Preempt algorithm, the multicast overlay's overlay unreliability metric is decreased after each preemption of the algorithm.

Proof: Suppose for the first preemption of the algorithm, the disconnected node n_d has preempted node n_1 by connecting to n_1's parent. According to our assumption on n_l, and the property that a node's PRI is proportional to its path unreliability metric, clearly $U^P(par(n_1)) \leq U^P(n_l)$. Before the preemption, T_1's overlay unreliability metric could be expressed as

$$\begin{aligned} U^O(T_1) \quad = \quad & U^O(T_1 - T(n_d) - T(n_1)) \\ & + U^P(n_l) \cdot (\Phi(n_d) + 1) + U^O_{in}(T(n_d)) \\ & + U^P(par(n_1)) \cdot (\Phi(n_1) + 1) + U^O_{in}(T(n_1)) \end{aligned}$$

After the preemption, the overlay unreliability metric for the formed overlay (denoted as T_2) is

$$\begin{aligned} U^O(T_2) &= U^O(T_2 - T(n_d) - T(n_1)) \\ &\quad + U^P(n_l) \cdot (\Phi(n_1) + 1) + U^O_{in}(T(n_1)) \\ &\quad + U^P(par(n_d)) \cdot (\Phi(n_d) + 1) + U^O_{in}(T(n_d)) \end{aligned}$$

Here it is assumed that the preempted node n_1 also temporarily connects to n_l. Please note that $T_1 - T(n_d) - T(n_1) = T_2 - T(n_d) - T(n_1)$ as the remaining part of the overlay is unchanged, and $par(n_1)$ in $U^O(T_1)$ is the same node as $par(n_d)$ in $U^O(T_2)$. As $\Phi(n_d) > \Phi(n_1)$ according to IRP-Preempt, clearly $U^O(T_2) < U^O(T_1)$. Similarly, it could be proved that $U^O(T_3) < U^O(T_2)$ for the second preemption, and $U^O(T_4) < U^O(T_3)$ for the third preemption, $\cdots$. For the last node preempted, it connects to a leaf node in its neighbor set with a path unreliability metric no greater than n_l, and the algorithm returns. □

Finally, for the IRP-Switch algorithm, we prove its effectiveness with the following theorem.

Theorem 3: By running the IRP-Switch algorithm, after each switch operation, the multicast overlay's overlay unreliability metric is decreased.

Proof: Suppose in the IRP-Switch algorithm, a parent node n_p and one of its child node n_c, with d_p and d_c as their degrees, get switched, then we consider the changes on the overlay's structure and their corresponding influences on the overlay unreliability metric. For the switch there are three changes: First, node n_p and node n_c's positions get switched; Second, for n_p's former child nodes other than n_c, they have a new parent node, n_c; Finally, for the $(d_c - d_p)$ child nodes with most descendant nodes, n_p is no longer their ancestor node. We consider the influences caused by each of these changes on the overlay unreliability metric: First, for the switch of n_p and n_c's positions, the difference on the overlay's overlay unreliability metrics after and before the switch is $(U^N(n_c) - U^N(n_p))$; Second, for n_p's former child nodes (except for n_c) and their descendants, the difference is $(U^N(n_c) - U^N(n_p)) \cdot (\Phi(n_p) - \Phi(n_c) - 1)$; Finally, for n_c's $(d_c - d_p)$ child nodes and their descendants, the difference is $-U^N(n_p) \cdot (\sum_{i=1}^{d_c - d_p} (\Phi(n_{c,i}) + 1))$, where $n_{c,i}$ is the ith node in n_c's $(d_c - d_p)$ child nodes with the largest numbers of descendants. Clearly for all these changes, the total difference on the overlay's overlay unreliability metrics after and before the switch is

$$\begin{aligned} \Delta U^O(T) &= (U^N(n_c) - U^N(n_p)) \cdot (\Phi(n_p) - \Phi(n_c)) \\ &- U^N(n_p) \cdot (\sum_{i=1}^{d_c - d_p} (\Phi(n_{c,i}) + 1)) \end{aligned}$$

As a node's NRI is proportional to its node unreliability metric, it is easy to see that $\Delta U^O(T)$ actually is proportional to $(c2 - c1)$ in the IRP-Switch algorithm. As in the algorithm, a pair of parent-child nodes get switched when $c2 < c1$, therefore the overlay's overlay unreliability metric is decreased after each switch operation of the algorithm. □

6 Performance Evaluation

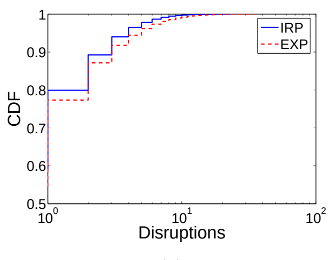
6.1 Simulation setup

We study the performance of our solution with with simulation experiments, and we also deploy an application-layer multicast overlay on the PlanetLab[30] testbed for examining our proposal under the real-world networking envrionment.

For simulation experiments, an event-driven simulator is developed using C++, where the proposed IRP protocol and the existing solutions of Min-Depth[14], Preempt-Degree[15], and ROST[16] are implemented. Here we do not consider the naive approaches used for demonstrating the merits of Min-Depth, Preempt-Degree, and ROST in their original works. The simulated multicast overlay is deployed upon an artificial Internet topology containing more than $20,000$ routers generated by GT-ITM[31] using its transit-stub model. Link delays between two transit nodes, one transit node and one stub node, and two stub nodes on the underlying topology are chosen uniformly in the ranges of $[15, 25]$ms, $[5, 10]$ms, and $[2, 4]$ms, respectively.

We use synthetic traces containing $20,000$ nodes with their joining and leaving events to drive the simulator. In the synthetic traces, nodes are assumed to join the overlay with a Poisson process, and their lifetimes follow the Lognormal distributions reported in [27]. For each node in the trace, a random outgoing bandwidth is assigned according to the measurement results in [19]. We choose the streaming media's playback rate as 500 Kbps. The root node's out degree is set as 20. In our simulation, each node keeps 50 neighbors. Our simulated overlay starts with only the root node, and nodes join and leave the overlay according to the trace. The simulated overlay has approximately $1,000$ online nodes under its stable state, but the experiment includes the phases when the population grows and diminishes at the beginning and at the end of the simulation.

There is one factor undetermined for our simulation, that is, how to set a node's NRI? For setting NRI, we exploit the fact that the nodes' lifetimes are heavy-tailed, which is widely observed in recent studies on real-world media streaming applications. In particular, K. Sripanidkulchai et al.[26] show that the streaming sessions on the Akamai network exhibit Pareto-like heavy-tailed behaviors; while E. Veloso et al.[27] report that the Internet streaming sessions' lengths could be modeled with Lognormal heavy-tailed distributions very well. As empirical results show that above a certain size, the two distributions are statistically undistinguishable [32], here we use the simplest heavy-tailed distribution, i.e., the Pareto distribution, for deriving NRI: If we let the CDF of the nodes' lifetimes be a Pareto distribution as $F(x) = 1 - \alpha\beta^\alpha x^{-\alpha-1}$, then the hazard rate is $h(x) = (\alpha + 1)/x$. We set a node n's $NRI(n)$ as $1/x$ in our experiments, where x is n's current age. One can see that under the Pareto lifetime, $NRI(n)$ is strictly proportional to $U^N(n)$, which satisfies our assumption in Section 5. More



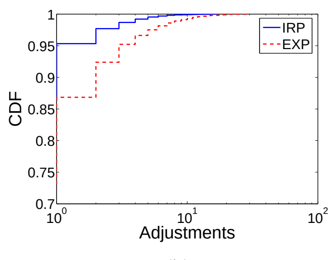


Fig. 2. CDFs of (a) disruptions and (b) adjustments for different protocols

importantly, by setting $NRI(n) = 1/x$, a node's NRI only depends on its age, while any detailed information on the nodes' lifetime model is not required. Please note that although the Pareto distribution is used for deriving our NRI setting, we use the Lognormal distributions reported in [27] in our experiments. Finally, the setting of NRI is not a rigid part of the IRP protocol, but could be configured by the system administrator.

For making a comprehensive comparison between our solution and the existing approaches of Min-Depth, Preempt-Degree, and ROST, we focus on the following three metrics in our simulation study:

- *Disruptions* - Average number of ancestor failures experienced by a node during its lifetime;
- *Adjustments* - Average number of times a node changes its parent node caused by preemptions and switches during its lifetime;
- *Overhead* - Average number of messages sent by a node for updating related information such as PRIs and descendant numbers during its lifetime.

Clearly, disruptions reflect the reliability of the overlay, while adjustments and overhead reveal the cost for achieving this reliability.

6.2 Overall performance comparison

We first make an overall performance study and comparison between IRP and the existing solutions. In this experiment, both the proactive and the reactive algorithms proposed by us or in early works are applied, that is, we compare the IRP protocol integrating the algorithms of IRP-Join, IRP-Preempt, and IRP-Switch, with the combination of the existing solutions, i.e. "Min-Depth + Preempt-Degree + ROST", which we refer to as *existing protocol* (denoted as *EXP* for short). Note that Min-Depth, Preempt-Degree and ROST are proposed in different works, and their combination is supposed to achieve a better reliability than applying each of them individually. For the nodes' lifetimes, we use the Lognormal distributions reported in [27] with parameters of ($\mu = 5.19$, $\sigma = 1.44$) (denoted as $LN(5.19, 1.44)$) and ($\mu = 5.74$, $\sigma = 2.01$) (denoted as $LN(5.74, 2.01)$) observed from different types of media streaming, and we also explore some other Lognormal distributions regarding diverse features in heavy-tailing. In the experiment, we set the interval for executing the proactive algorithms of IRP-Switch and ROST as 400 seconds. The experiment results are listed in Table 2. For each entry of the table, we list the mean value and the standard deviation (in brackets) summarized from five simulation executions. We also plot the CDFs of the nodes' disruptions and adjustments under difference protocols with the trace of $LN(5.19, 1.44)$ in Fig. 2.

By carefully examining the simulation outputs of the two protocols under four difference traces, we have the following findings. First of all, it is observed from Table 2 and Fig. 2(a) that by executing IRP, there are fewer ancestor failures, indicating a better overlay reliability than applying EXP. Second, we can see from Table 2 and Fig. 2(b) that IRP makes much fewer adjustments on the overlays' structures than EXP, although it maintains better reliability. We explain this by the fact that a better overlay reliability and fewer adjustments are correlated, as it is less likely for a node on a reliable overlay to encounter parent failures, therefore there are fewer node preemptions. Finally, we find that in general, the overhead incurred by both protocols is in a few messages per node's lifetime, which is not significant, and IRP generally has a heavier overhead, with the only exception under the trace of $LN(3.58, 2.8)$. Note that this observation does not follow our intuition, as in IRP, a node needs to know its PRI and descendant number, while in EXP, only the depth needs to be obtained. Intuitively, IRP should have twice the overhead of EXP. We explain this observation with the lazy update scheme applied: Recall that in the lazy update, overhead is only incurred when there are changes on the overlay's structure. As IRP makes fewer overlay adjustments, it is less frequent for a node under IRP to send messages to its parent or child nodes for updating PRIs and descendant numbers.

In summary, we conclude that there are two merits for our proposed IRP protocol compared with the existing solutions: first, the protocol improves the overlay reliability non-trivially; second, it achieves better reliability with much fewer structure adjustments. In addition, we note that the superiority of IRP is observed under a number of traces with great diversity, suggesting that our setting of NRI is appropriate as long as the nodes' lifetimes are heavy-tailed.

6.3 Detailed performance comparison

In this subsection, we make a detailed comparison on node joining, node preemption and node switching schemes. We use the same simulation settings as in Section 6.2, and use $LN(5.19, 1.44)$ and $LN(5.74, 2.01)$ for generating the traces.

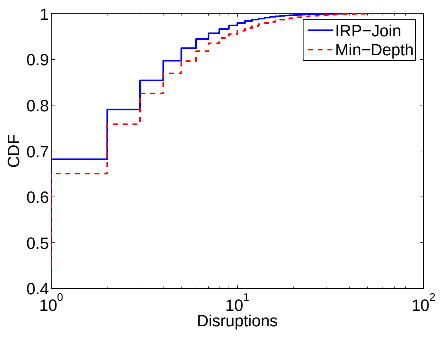
First, for a fair comparison on the effectiveness of the Min-Depth scheme preferred in [14] and our proposed IRP-Join algorithm, we use the No-Preempt scheme for handling disconnected nodes, where a disconnected node rejoins the overlay by connecting to the leaf node

TABLE 2
Overall Performance Comparison

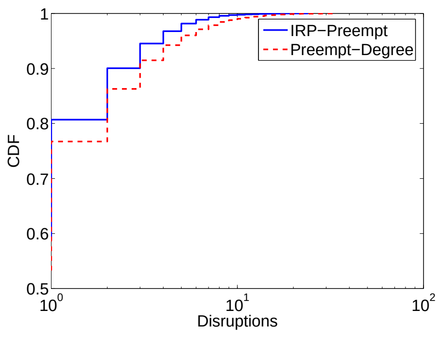
	LN(5.19, 1.44)		LN(5.74, 2.01)		LN(3.58, 2.8)		LN(6.78, 1.2)	
	IRP	EXP	IRP	EXP	IRP	EXP	IRP	EXP
Disruptions	0.85(0.07)	1.06(0.06)	0.42(0.03)	0.51(0.06)	0.15(0.01)	0.18(0.02)	1.30(0.05)	1.43(0.09)
Adjustments	0.24(0.02)	0.68(0.03)	0.22(0.02)	0.36(0.02)	0.08(0.01)	0.15(0.01)	0.46(0.02)	0.94(0.03)
Overhead	3.83(0.25)	3.78(0.15)	2.36(0.11)	2.33(0.15)	1.43(0.03)	1.54(0.05)	5.37(0.16)	4.61(0.17)

TABLE 3
Detailed Performance Comparison

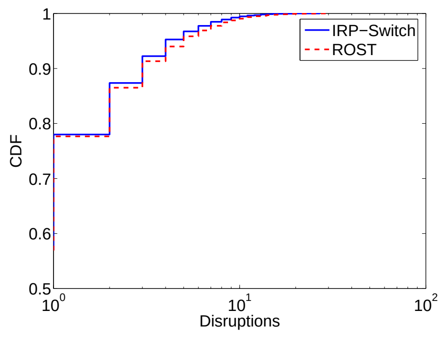
	LN(5.19,1.44)		LN(5.74, 2.01)	
	IRP-Join	Min-depth	IRP-Join	Min-depth
Disruptions	1.84(0.54)	1.92(0.10)	0.87(0.16)	1.04(0.14)
Adjustments	0.00(0.00)	0.00(0.00)	0.00(0.00)	0.00(0.00)
Overhead	2.63(0.10)	2.91(0.10)	1.85(0.16)	2.02(0.14)
	IRP-Preempt	Preempt-Degree	IRP-Preempt	Preempt-Degree
Disruptions	0.87(0.07)	1.03(0.09)	0.44(0.03)	0.53(0.08)
Adjustments	0.11(0.01)	0.56(0.03)	0.06(0.01)	0.30(0.02)
Overhead	4.04(0.24)	3.84(0.19)	2.50(0.10)	2.48(0.19)
	IRP-Switch	ROST	IRP-Switch	ROST
Disruptions	1.12(0.12)	1.08(0.09)	0.56(0.04)	0.69(0.11)
Adjustments	0.41(0.05)	0.46(0.03)	0.29(0.01)	0.30(0.06)
Overhead	2.32(0.17)	2.10(0.10)	1.65(0.06)	1.69(0.11)



(a)



(b)



(c)

Fig. 3. CDFs of disruptions under different (a) node joining schemes, (b) node preemption schemes, and (c) node switch schemes

of the minimum depth, and we do not execute the algorithms of ROST and IRP-Switch. The simulation outputs are presented in Table 3, and we also plot the CDFs of the nodes' disruptions in Fig. 3(a) under the trace of $LN(5.19, 1.44)$. From the experiment results we can see that nodes under the IRP-Join algorithm encounter fewer ancestor failures than under the Min-Depth scheme, and consequently there is a lighter overhead for IRP-Join. Our observation indicates that a node's NRI in IRP is a better metric than its depth for a new joining node in selecting its parent.

For fairly comparing the effectiveness of the Preempt-Degree scheme in [15] and our proposed IRP-Preempt algorithm, we use Min-Depth for handling new joining nodes and we also do not run the algorithms of ROST and IRP-Switch. The simulation outputs are presented in Table 3 and Fig. 3(b). It could be observed that although our proposed IRP-Preempt algorithm incurs a heavier overhead, it achieves a better overlay reliability than Preempt-Degree. Moreover, it makes much fewer adjustments on the overlay's structure. The experiment result shows that a node's descendant number is a better metric than its degree in making node preemption decisions.

Finally, we consider our proactive algorithm of IRP-Switch and the ROST algorithm proposed in [16]. For fairness, we use Min-Depth and No-Preempt for handling new joining nodes and disconnected nodes respectively, and execute the proactive node switching algorithms every 20 seconds. The simulation results are presented in Table 3 and Fig. 3(c). From the experiment outputs we could see that the IRP-Switch algorithm has a similar performance as ROST on preserving the overlay's reliability, but it triggers fewer adjustments. The reason why there is no obvious superiority for IRP-Switch over ROST is that in both algorithms, the adjustments on the overlay's structure are restricted in switching parent-child pairs along an overlay path, and both algorithms will perform similar node switches. And in the next subsection, we will see that the proactive node switching is not very necessary.

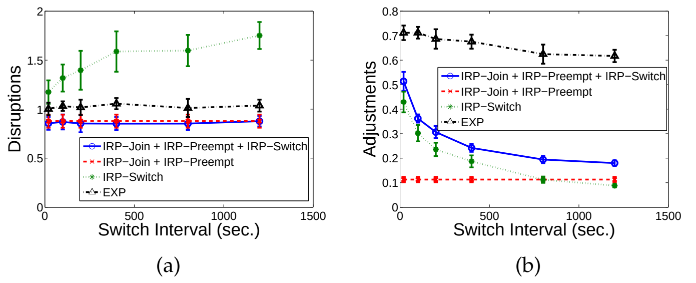


Fig. 4. (a) Disruptions and (b) adjustments for different algorithm combinations under constant churn

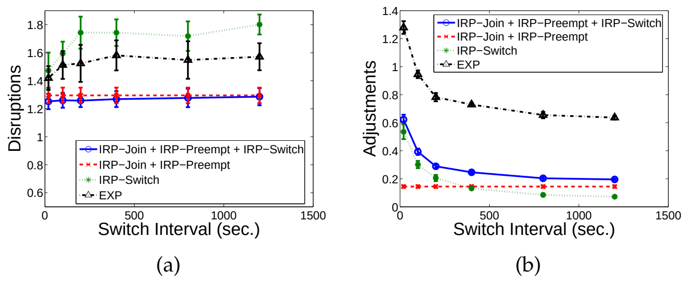


Fig. 5. (a) Disruptions and (b) adjustments for different algorithm combinations under flash crowds

In summary, by conducting overall and detailed comparisons between IRP and the existing solutions, it is observed that our proposed algorithms show superior performances than their counterparts. As in either protocol, the incurred network overhead is not significant. In the following discussion, we do not consider it any further, but focus on disruptions and adjustments.

6.4 Proactive vs. reactive approaches

If we examine the results obtained under the same trace in the previous experiments, it is interesting to find that the preemption algorithm has the greatest contribution for the overlay's reliability. As the node joining algorithm does not make any adjustments, it is a natural question to ask whether or not a proactive algorithm such as IRP-Switch is necessary?

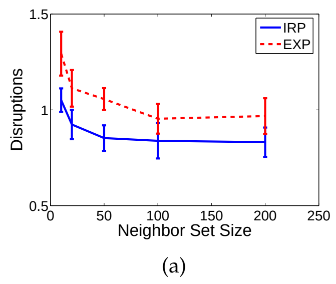
For answering this question, we compare three algorithm combinations when: 1) all the algorithms are applied, i.e. "IRP-Join + IRP-Preempt + IRP-Switch"; 2) only the reactive algorithms are applied, i.e. "IRP-Join + IRP-Preempt"; and 3) only the proactive algorithm of IRP-Switch is applied. As IRP-Switch is executed periodically, we also study the performances with different execution intervals varying from 20 seconds to 1,200 seconds. The $LN(5.19, 1.44)$ traces are used in this experiment. We plot the disruptions and adjustments for each algorithm combination in Fig. 4, and the performances of EXP are also presented for comparison. From the experiment results we could find that, first of all there are only limited improvements on the overlay's reliability by applying the proactive IRP-Switch algorithm alone, but executing it frequently (at a cost of adjustments) helps; Second, the combination of the reactive algorithms "IRP-Join + IRP-Preempt" achieves very good reliability, and makes very few adjustments, in particular, it obviously outperforms EXP in both disruptions and adjustments; Finally, combining all the proactive and reactive algorithms leads to the best reliability, but frequently executing the IRP-Switch algorithm cannot improve the overlay's reliability greatly while introduces many adjustments. Based on these observations, we conclude that the proactive algorithm does not play an important role in improving the overlay's reliability under constant churn.

However, in Internet streaming, sometimes users are requesting media in flash crowds, which means a large number of users subscribe a streaming service within a very short time. Intuitively, under this highly dynamic user behaviors, reactive adjustments on the overlay's structure should be less effective while the proactive solution should work better. For examining this point, we simulate flash crowds in our experiment. With the trace of flash crowding behaviors, again we compare the performances of "IRP-Join + IRP-Preempt + IRP-Switch", "IRP-Join + IRP-Preempt", IRP-Switch and EXP, and plot the results in Fig. 5. It is interesting to find that under flash crowds, the performances of the algorithm combinations with and without IRP-Switch exhibit similar characteristics as under constant churn in Fig. 4, where IRP-Switch does not contribute greatly to the overlay's reliability. We believe the reason for the good performance of the reactive algorithms is that they are executed on events of node joining and failure, and timings of these events themselves reflect the dynamics of the nodes' behaviors, therefore the combination of "IRP-Join + IRP-Preempt" could be able to adjust the overlay dynamically by nature.

Summarizing all the observations, we conclude that very good reliability could be expected by applying only the reactive algorithms of IRP-Join and IRP-Preempt, while the proactive algorithm of IRP-Switch could be configured as optional and executed infrequently, for avoiding frequent overlay structure adjustments and reducing the protocol overhead.

6.5 Sensitivity to neighbor set size

In our proposed IRP-Join and IRP-Preempt algorithms, nodes need to select potential parents among their neighboring nodes. As the overhead incurred for obtaining and updating neighbors is related to the neighbor set size, it is necessary for us to examine its influence on the performance of our proposed solution. In this experiment we also use the settings as in Section 6.2, but vary the neighbor set size from 10 nodes to 200 nodes, and study the disruptions and adjustments for the IRP protocol under the traces of $LN(5.19, 1.44)$. We also study EXP in this experiment, as in Min-Depth and Preempt-Degree, nodes also rely on their neighbor sets



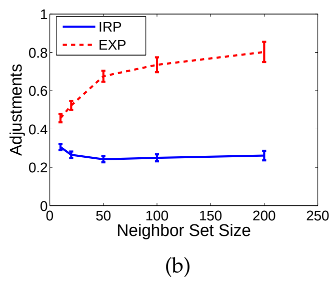


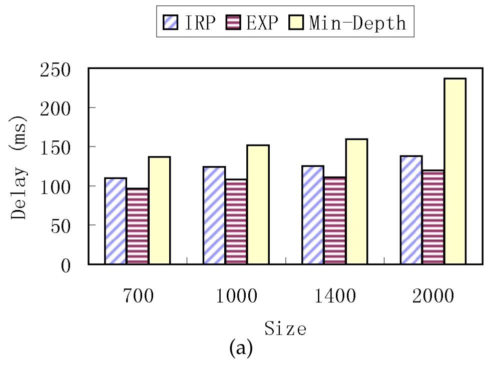
Fig. 6. (a) Disruptions and (b) adjustments for different protocols with varying neighbor set size

for joining and rejoining the overlay. The experiment results are plotted in Fig. 6. From Fig. 6(a) it is observed that when a node has more neighbors, reliability of the overlays under the two protocols are improved. However, we find that under IRP, keeping 50 neighbors leads to a good enough reliability, and it is unnecessary for nodes to keep a very large neighbor set. It is very interesting to observe from Fig. 6(b) that under EXP, the value of adjustments per node is increasing with a larger neighbor set, but there is no obvious change under IRP. We explain the increasing adjustments under EXP with the fact that there are more preemptions when nodes have more choices with a larger neighbor set. However, under IRP, as there are fewer parent failures, the consequent preemptions are fewer. Balancing the two factors of a larger neighbor set and fewer parent failures, the overlay adjustments under IRP are not changed significantly.

6.6 Latency and overlay stretch

Although we focus on improving the application-layer multicast overlay's reliability, it is also important to examine the quality of the service experienced by end users in terms of end-to-end delays. In this subsection, we study the nodes' service delays and *overlay stretch* for the overlays under different protocols. Overlay stretch is defined as the ratio between the sum of the service delays for all the nodes on the overlay and the sum of the delays if these nodes receive the streaming from the source with direct unicast channels on the underlying network.

As mentioned in Section 6.1, we deploy all the overlay nodes including the root at stub nodes on the underlying network topology generated by GT-ITM. We study the simulated overlays in their stable states under our proposed IRP and EXP combining all the existing solutions. The size of the stable state overlays in our simulation varies from 700 nodes to 2,000 nodes. Note that for the schemes of Min-Depth and Preempt-Degree, their only design objective is to decrease the overlay depth, and for the ROST algorithm, it also needs to decrease the depth of the multicast tree after each switch. Therefore, EXP should have a smaller latency and overlay stretch than IRP. We also study the overlays under the Min-Depth scheme but without any preemptions and switches for comparison. The experiment results are presented in Fig. 7. From the figures, we could find that as expected, EXP forms an overlay with the smallest average delay and overlay stretch, but the performance of our proposed IRP is also very satisfactory, while the overlay under Min-Depth without any preemptions and proactive switches has a poor performance.



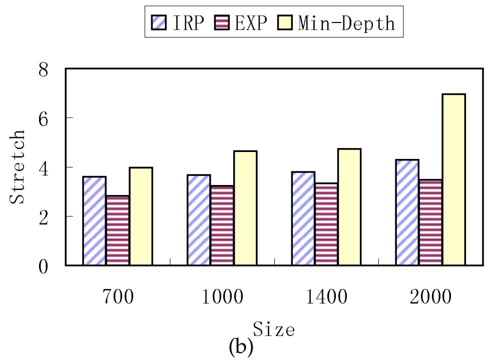
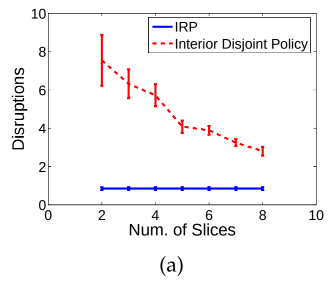


Fig. 7. (a) Average delays and (b) overlay stretches for overlays under different protocols with varying overlay size



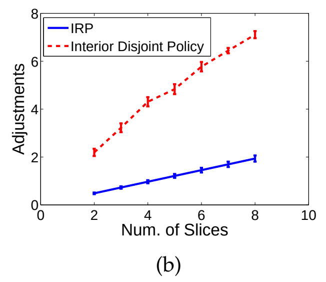


Fig. 8. (a) Disruptions and (b) adjustments for multiple-tree overlays applying IRP and Interior Disjoint Policy

6.7 Multiple-tree overlay

In previous experiments, we focus on the multicast overlay containing one single tree. However, overlays containing multiple trees for the Internet media streaming emerge in recent years. In this subsection, we focus our study on overlays with multiple trees. Generally, in a multiple-tree overlay, the streaming media is divided into *slices*, where each slice is a description of the Multiple Description Coded (MDC) media content, and for propagating each slice, a multicast tree is constructed. In the multiple-tree overlay, each client joins all the multicast trees for receiving a full-rate streaming service. When constructing the multiple-tree overlay, the *interior disjoint policy* could be applied, where each client joins only one multicast tree with all its outgoing bandwidth as an *interior node*, and joins other multicast trees as *leaf nodes*. For example, SplitStream[7] applies the interior disjoint policy in its design, but ChunkSpread[8] does't. Under the interior disjoint policy, each client contributes its bandwidth in one multicast tree for forwarding one slice, therefore a node in the tree of the multiple-tree overlay where it is a interior node will have a bigger degree than in the single-tree overlay. When the interior disjoint policy is not used, we could view each multicast tree in the multiple-tree overlay as an independent tree, and apply the IRP protocol for improving the overlay's reliability by improving each multicast tree's reliability individually.

We conduct a comparison between two kinds of multiple-tree overlays with and without the interior disjoint policy. For the overlay without the policy, each multicast tree is constructed and maintained by IRP independently, while for the overlay applying the interior disjoint policy, we use a protocol similar to the one in [20] for building its multicast trees. We study the disruptions and the adjustments for different overlays in our simulation. In a multiple-tree overlay, an ancestor failure in one multicast tree will only cause a client to lose $1/S$ of its streaming service, where S is the number of the slices, therefore we count an ancestor failure in one multicast tree as $1/S$ disruption in the experiment. We consider situations when the content is coded into different numbers of slices, and present the experiment results in Fig. 8. From the figures, it is observed that by running IRP on each multicast tree without the interior disjoint policy, we could have a more stable overlay than applying the policy; meanwhile, IRP incurs much fewer adjustments. In fact, the interior disjoint policy helps build a reliable overlay only when the number of the slices is large, but its reliability is at a cost of frequent overlay structure adjustments, as indicated by the trends in Fig. 8(a) and (b), and moreover, the network and media-codec overheads increase with the slice number. In conclusion, our findings in this experiment suggest that the proposed IRP protocol could be applied directly on the multiple-tree multicast overlay for improving its overall reliability.

6.8 Deployment and evaluation on PlanetLab

Previous experiments are carried out with simulation. To testify the effectiveness of our proposal under real-world networking environment, we have deployed a multicast overlay on the PlanetLab testbed [30]. The deployed overlay contains about 100 PlanetLab nodes under stable state, and there are 400 overlay nodes joining and leaving the overlay according to the $LN(5.74, 2.01)$ lifetime model. For multicasting, the source node located at "planetlab1.ie.cuhk.edu.hk" sends a fix-sized UDP packet every two seconds to its child nodes, and on receiving the data, a node forwards it to its children immediately using UDP, but nodes do not buffer the data or retransmit if the it is lost. The data multicasting lasts for more than four hours, and after the experiment for each node ever joined the overlay we calculate its *data reception ratio*, which is defined as the ratio between the total amount of the data it has practically received and all the data it is supposed to receive (i.e., the data sent out by the source node during its online time). Obviously, the data reception ratio reflects how reliable the multicast overlay provides a streaming service to the node, as any disruption of the service will result in data loss. We do not consider the factor of the traffic congestion as PlanetLab nodes are generally well connected to the Internet. We examine the overlays under the IRP and the EXP protocol on PlanetLab for comparison. Fig. 9 shows the relationship between a node's lifetime and its data reception ratio for nodes under the two protocols, and Fig. 10 plots the CDFs of the data reception ratios. From the two figures one can see clearly that nodes under IRP have higher data reception ratios than nodes under EXP, and this observation indicates that IRP is more effective in improving the multicast overlay's reliability than EXP under the real-world networking environment.

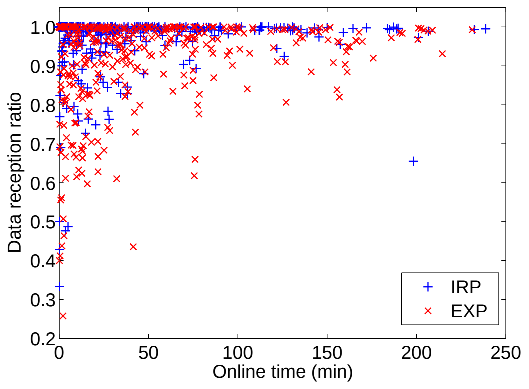


Fig. 9. Relationship between lifetime and data reception ratio

7 Conclusion

In this paper, we address the reliability issue for the tree-like application-layer multicast overlays. We first formally presented the overlay reliability problem, and designed a distributed and light-weighted protocol named instantaneous reliability oriented protocol (IRP) for constructing, repairing, and maintaining reliable multicast overlays proactively as well as reactively. Concretely, our protocol is composed of three algorithms, i.e., IRP-Join, IRP-Preempt, and IRP-Switch, for node joining,

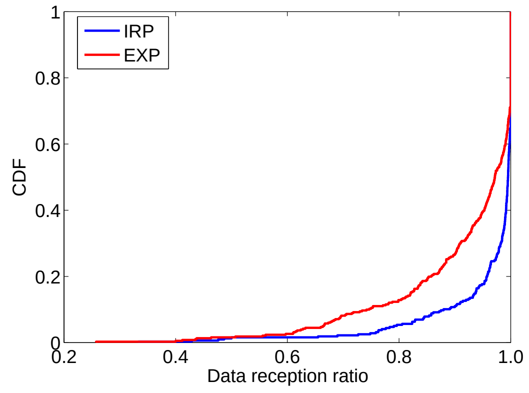


Fig. 10. CDFs of data reception ratios under IRP and EXP

node preemption, and node switching respectively. With the formal presentation of the problem, we theoretically proved that these algorithms are effective. To our best knowledge, this is the first effort for formally understanding and theoretically solving the application-layer multicast overlay's reliability problem. Finally, through experiments based on simulation and PlanetLab deployment, we studied the performance of our solution, and compared the entire protocol and each of its components with its existing solution counterpart. The experiment results indicate that IRP could improve the multicast overlay's reliability non-trivially with fewer adjustments on the overlay's structure. We also explored the necessity of the proactive node switch algorithm, studied the issues of service latency and overlay stretch, and discussed the application of our protocol on multicast overlays containing multiple trees.

Acknowledgment

The authors would like to thank the anonymous reviewers for their constructive comments. This work is supported in part by the National Natural Science Foundation of China under grants 60533020 and 60772034, and the Chinese Academy of Sciences "100 Talents" Project.

References

[1] J. Liu, S. G. Rao, B. Li, and H. Zhang, "Opportunities and challenges of peer-to-peer Internet video broadcast," *Proceedings of the IEEE*, vol. 96, no. 1, pp. 11–24, 2008.

[2] A. G. Y. Chu, T. S. E. Ng, S. G. Rao, K. Sripanidkulchai, J. Zhan, and H. Zhang, "Early experience with an internet broadcast system based on overlay multicast," in *Proc. of USENIX'04*, Boston, MA, USA, Jun. 2004.

[3] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," in *Proc. of ACM SIGCOMM'02*, Pittsburgh, PA, USA, Aug. 2002.

[4] M. Castro, P. Druschel, A. M. Kermarrec, and A. I. T. Rowstron, "Scribe: a large-scale and decentralized application-level multicast infrastructure," *IEEE J. on Sel. Areas in Commun.*, vol. 20, no. 8, pp. 1489 – 1499, 2002.

[5] D. A. Tran, K. Hua, and T. Do, "A peer-to-peer architecture for media streaming," *IEEE J. on Sel. Areas in Commun.*, vol. 22, no. 1, pp. 121 – 133, 2004.

[6] V. N. Padmanabhan, H. J. Wang, and P. A. Chou, "Resilient peer-to-peer streaming," in *Proc. of IEEE ICNP'03*, Atlanta, GA, USA, Nov. 2003.

[7] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: High-bandwidth multicast in a co-operative environment," in *Proc. of SOSP'03*, Lake Bolton, New York, USA, Oct. 2003.

[8] V. Venkataraman, K. Yoshida, and P. Francis, "Chunkyspread: Heterogeneous unstructured end system multicast," in *Proc. of IEEE ICNP'06*, Santa Barbara, CA, USA, Nov. 2006.

[9] X. Zhang, J. Liu, B. Li, and T. P. Yum, "CoolStreaming/DONet: A data-driven overlay network for efficient live media streaming," in *Proc. of IEEE INFOCOM'05*, Miami, FL, USA, Mar. 2005.

[10] N. Magharei and R. Rejaie, "PRIME: Peer-to-peer receiver-driven mesh-based streaming," in *Proc. of IEEE INFOCOM'07*, Anchorage, AK, USA, May 2007.

[11] PPLive, http://www.pplive.com.

[12] D. Kostic, A. Rodriguez, J. Albrecht, and A. Vahdat, "Bullet: High bandwidth data dissemination using an overlay mesh," in *Proc. of SOSP'03*, Lake Bolton, New York, USA, Oct. 2003.

[13] F. Wang, Y. Xiong, and J. Liu, "mTreebone: A hybrid tree/mesh overlay for application-layer live video multicast," in *Proc. of ICDCS'07*, Toronto, Ontario, Canada, Jun. 2007.

[14] K. Sripanidkulchai, A. Ganjam, B. Maggs, and H. Zhang, "The feasibility of supporting large-scale live streaming applications with dynamic application end-points," in *Proc. of ACM SIGCOMM'04*, Portland, OR, USA, Sep. 2004.

[15] M. Bishop, S. Rao, and K. Sripanidkulchai, "Considering priority in overlay multicast protocols under heterogeneous environments," in *Proc. of IEEE INFOCOM'06*, Barcelona, Spain, Apr. 2006.

[16] G. Tan and S. A. Jarvis, "Improving the fault resilience of overlay multicast for media streaming," *IEEE Trans. on Paral. and Distr. Systems*, vol. 18, no. 6, pp. 721 – 734, 2007.

[17] D. Kostic, A. Rodriguez, J. Albrecht, A. Bhirud, and A. Vahdat, "Using random subsets to build scalable network services," in *Proc. of USENIX USITS'03*, Seattle, WA, USA, Mar. 2003.

[18] J. Liang and K. Nahrstedt, "RandPeer: Membership management for qos sensitive peer-to-peer applications," in *Proc. of IEEE INFOCOM'06*, Barcelona, Spain, Apr. 2006.

[19] Y. Tian, D. Wu, G. Sun, and K.-W. Ng, "Improving stability for peer-to-peer multicast overlays by active measurements," *Journal of System Architecture*, vol. 54, no. 1-2, pp. 305 – 323, 2008.

[20] N. Magharei, R. Rejaie, and Y. Guo, "Mesh or multiple-tree: A comparative study of live p2p streaming approaches," in *Proc. of IEEE INFOCOM'07*, Anchorage, AK, USA, May 2007.

[21] A. Chakrabarti and G. Manimaran, "Reliability constrained routing in qos networks," *IEEE/ACM Trans. on Networking*, vol. 13, no. 3, pp. 662 – 675, 2005.

[22] S. Raghavan, G. Manimaran, and C. S. R. Murthy, "A rearrangeable algorithm for the construction of delay-constrained dynamic multicast trees," *IEEE/ACM Trans. on Networking*, vol. 7, no. 4, pp. 514 – 529, 1999.

[23] F. Bauer and A. Varma, "ARIES: A rearrangeable inexpensive edgebased on-line Steiner algorithm," *IEEE J. on Sel. Areas in Commun.*, vol. 15, no. 3, pp. 514 – 529, 1997.

[24] S. Hong, H. Lee, and B. H. Park, "An efficient multicast routing algorithm for delay-sensitive applications with dynamic membership," in *Proc. of IEEE INFOCOM'98*, San Francisco, CA, USA, Mar. 1998.

[25] G. Iannaccone, C. Chuah, R. Mortier, S. Bhattacharyya, and C. Diot, "Analysis of link failures in an IP backbone," in *Proc. of Internet Measurement Workshop, IMW'02*, Marseille, France, Nov. 2002.

[26] K. Sripanidkulchai, B. M. Maggs, and H. Zhang, "An analysis of live streaming workloads on the internet," in *Proc. of IMC'04*, Taormina, Italy, Oct. 2004.

[27] E. Veloso, V. Almeida, W. Meira, A. Bestavros, and S. Jin, "A hierarchical characterization of a live streaming media workload," *IEEE/ACM Trans. on Networking*, vol. 14, no. 1, pp. 133 – 146, 2006.

[28] K. S. Trivedi, *Probability and Statistics with Reliability, Queuing, and Computer Science Applications*, 2nd ed. New York, NY: John Wiley and Sons, 2002.

[29] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman, "SybilGuard: Defending against sybil attacks via social networks," in *Proc. of ACM SIGCOMM'06*, Pisa, Italy, Sep. 2006.

[30] L. Peterson, T. Anderson, D. Culler, and T. Roscoe, "A blueprint for introducing disruptive technology into the Internet," *SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 1, pp. 59 – 64, 2003.

[31] E. Zegura, K. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *Proc. of IEEE INFOCOM'96*, San Francisco, CA, USA, Mar. 1996.

[32] A. B. Downey, "Lognormal and pareto distributions in the Internet," *Computer Commun.*, vol. 28, no. 7, pp. 790 – 801, 2004.

Ye Tian received the B.E. degree in electronic engineering and the M.S. degree in computer science from the University of Science and Technology of China in 2001 and 2004 respectively. He received the Ph.D. degree in computer science from The Chinese University of Hong Kong in 2007. Currently he is an associate professor in the School of Computer Science and Technology at the University of Science and Technology of China. His research interests include computer networks and distributed systems. He is a member of IEEE.

Hong Shen is Professor (Chair) of Computer Science at University of Adelaide, Australia, and also a specially-appointed professor at University of Science and Technology of China. He received the B.E. degree from Beijing University of Science & Technology, M.E. degree from University of Science & Technology of China, Ph.Lic. and Ph.D. degrees from Abo Akademi University, Finland, all in Computer Science. He was Professor and Chair of the Computer Networks Laboratory in Japan Advanced Institute of Science and Technology (JAIST) during 2001-2006, and Professor (Chair) of Compute Science at Griffith University, Australia, where he taught 9 years since 1992. With main research interests in parallel and distributed computing, algorithms, data mining, high performance networks and multimedia systems, he has published more than 200 papers including over 100 papers in international journals such as a variety of IEEE/ACM transactions. Prof. Shen received many awards/honours including 1991 National Education Commission Science and Technology Progress Award, 1992 Chinese Academy of Sciences Natural Sciences Award, and 2005 Chinese Academy of Sciences "Hundred Talents". He serves on editorial roles for several journals.

Kam-Wing Ng is currently a professor with the Department of Computer Science and Engineering at The Chinese University of Hong Kong. His research interests include computer networks and grid computing.