Improving Resiliency of Overlay Networks for Streaming Applications

Wenjie Wang, Ye Du, Sugih Jamin
{wenjiew,duye,jamin}@eecs.umich.edu

Abstract

Early deployment of peer-to-peer (P2P) streaming network demonstrates its potential to deliver quality media streams to a large audience. However, P2P streaming is vulnerable to node failures and malicious attacks because of its high bandwidth demand and stringent timing requirement. In this paper, we investigated several heuristics to improve an overlay's resilience to failures and attacks. We formalized the overlay connectivity resilience as a graph theoretic problem and proved that disjoint paths with variable lengths are effective in increasing overlays' connectivity resilience. We proposed a distributed heuristic called "MPath" that enables each peer to improve its connectivity without global knowledge or coordination. We designed the *MPath* heuristic to work in conjunction with existing overlay improvement mechanisms. Such integration not only speeds up overlay convergence, but also significantly cuts down *MPath*'s maintenance overhead. Our experiments showed that *MPath* reduced the number of disconnected components in an overlay by an order of magnitude when 20% of overlay nodes failed.

1 Introduction

Peer-to-peer (P2P) networks, especially P2P file sharing networks, have been quickly adopted by large Internet communities in the past few years. Recently, the emergence of P2P streaming service, such as conference broadcasting [1] and Internet TV [2], demonstrates its potential to deliver high quality media streams to a large audience. In fact, there were instances of Internet deployments of end-host multicast for video/audio broadcasting. The technical programs of ACM SIGCOMM '02 [3] and SIGCOMM '04 [4] were broadcast live online with end-host multicast protocols. There are also large scale audio service experiment with end-host multicast that involves over 10,000 participants [5].

Compared to other widely used file-sharing networks, P2P video streaming applications place the following unique requirements on the quality of service provided by the underlying overlay networks.

- Reliability: Video streaming usually requires high bandwidth. A large amount of data loss could render the entire stream useless [6].
- Time Constraint: Streaming applications require their data to be delivered in a timely fashion. Data that miss the deadline cannot be played back.
- Resource Discovery: In P2P streaming, there will be only a limited number of servers with sufficient CPU power and adequate network capacity to serve a large number of peers. A peer—usually a notebook or an office desktop—must quickly locate one or more peers with enough free capacity to serve the media.
- Membership Dynamics: The membership of an overlay network is highly dynamic. The departure of peers should not interrupt the media playback of the peers they were serving.

These QoS requirements and characteristics make providing scalable, satisfying P2P streaming service more challenging than providing traditional client-server unicast delivery.

In this chapter, we focus on improving the resiliency of overlay networks to random failures and targeted attacks. We call resiliency to random failures and targeted attacks *connectivity resilience*. An overlay that is resilient to connectivity failures tends to remain connected even if a large number of nodes have failed. Nodes may become unavailable due to membership churn, node or link failures, or targeted

attacks. A disconnected segment in an overlay prevents all peers inside that segment from receiving data from nodes in other segments. A disconnected overlay also makes application layer features such as routing recovery [7] and buffering less capable of recovering data lost during the overlay recovery time.

In an overlay network, one common reason for a peer to suffer poor QoS is that it loses its connection to one or more of its upstream peers due to either group dynamics or node failures. In order to reduce the probability of such connectivity failures, we conducted both empirical and theoretical analyses to study factors that can affect the connectivity probability between peers. We found the following guidelines that can be used to improve the connectivity resilience of overlays.

- A peer can increase its chances of remaining connected to data sources either by reducing its distance to the data sources or by adding more disjoint paths. We prove that the latter is more effective.
- When adding disjoint paths to data sources, variable length paths are more effective than paths with similar lengths in improving the chances of keeping overlay nodes connected.
- Irregular graphs are more tolerant to node failures than regular graphs. Random links can improve the resilience of an overlay to connectivity failures.
- Data sources with high node degrees increase the chances that overlay peers can form disjoint paths.

Based on the above observations, we propose a distributed heuristic we call "MPath" that can effectively improve the connectivity of the whole overlay network. The basic idea of *MPath* is to provide peers in an overlay with *good* alternative paths to the data sources. *MPath* also enables individual peers to improve their own connectivity as well as the overall overlay connectivity without global knowledge or coordination.

MPath can also be used on streaming overlays with low degree nodes. Due to the limited number of downstream peers one host can support, overlay links have to be added carefully to avoid overloading the hosts with slow Internet connections. The degree constraints limit the number of disjoint paths a peer can find. Due to the limited node degree each peer has, streaming overlays tend to have long distances among peers. This represents an opportunity for *MPath* to select disjoint paths with variable lengths.

We design the *MPath* heuristic so that it can be integrated with overlay improvement procedure used by existing overlay protocols. Performance evaluation shows that our heuristic greatly improves the connectivity resilience of overlay protocols. More importantly, it does not severely impact the performance of overlays in terms of end-to-end latencies. Our simulations confirm that MPath improves the resilience of overlays to connectivity failures by 18%, and reduces the chance of disconnected network segments by almost 10 times. In exchange for the increased resilience, MPath increases average end-to-end latencies by less than 5%.

The rest of the chapter is structured as follows: We first formalize our overlay resilience problem and present the general theoretical analysis. We investigate and derive several guidelines and heuristics for overlay resilience improvement in Section 3. We present our *MPath* heuristic in Section 4 and report the performance evaluation result in Section 5. We summarize this chapter in Section 7.

2 Problem Analysis and Formalization

2.1 Assumptions

k-connected is widely used as a metric to indicate a graph's fault-tolerance. A graph is k-connected (k-node connected, to be exact) if the graph remains connected even after any $k-1$ nodes are removed. A k-regular graph, a graph in which all nodes have degree k, can be $k-1$ connected if carefully designed. However, it is not possible for a graph to be k-connected if any of its nodes has degree limit less than k. In this chapter, we do not assume nodes in an overlay to have the same degree limit. We assign nodes (v) in our overlay network with different out-degree limit $\hat{D}(v)$. The degree limit of a node is based on the speed of its Internet connection. For instance, a host with a 56 Kbps modem connection will have a lower degree limit than a host with a 10 Mbps connection.

The bandwidth requirements of streaming video and the limited available bandwidth of Internet hosts make it hard to construct a k-connected overlay without saturating the bandwidth of peers with slow con-

nections. For the same reason, existing approaches [8][9] that build "regular" graphs with k connectivity as their metric may not be applicable to the fault-tolerance study of overlay networks.

2.2 Performance Metric and Failure Model

Instead of k-connectivity, we define the largest connected component size (LCCS) [10] as our primary fault-tolerance metric. An overlay may be partitioned into several connected components if a large number of its nodes or edges fail. The LCCS is defined as the size of the connected component with the largest number of nodes. This metric is more practical than k-connectivity because it focuses on keeping the "core" of an overlay connected, instead of wasting efforts to maintain connectivity for nodes at the edge of the overlay.

The second metric we employ is called disconnected segments. A disconnected segment is a partitioned component consisting of two or more nodes. When a node loses its upstream nodes, it can reconnect to other nodes. Nodes in a disconnected segment may not be aware that the network has been partitioned. The perceived QoS of these nodes will suffer if they cannot reconnect to nodes in other segments quickly. With a large number of disconnected segments, it is likely that more overlay nodes may perceive bad QoS.

The LCCS and the number of disconnected segments can vary significantly with the removal of different sets of nodes. It is necessary to collect the statistics based on several different node failure models. We define the following four failure models:

- Random node failure: Nodes are removed one by one uniformly at random.
- Preferential node attack: Nodes are removed one by one preferentially; the probability of a node being removed is directly proportional to its degree.
- Random edge failure: Edges are removed one by one uniformly at random.
- Preferential edge attack: Edges are removed one by one preferentially; the probability of an edge being removed is directly proportional to the product of its two end-point node degrees.

Since a node failure can be transformed to an edge failure by converting a graph to its dual graph [11], in this chapter, we only present the results obtained from node failure and preferential node attack.

2.3 Formalization of Connectivity Resilience Problem

To improve the connectivity of an overlay, we first need to understand the factors affecting its connectivity, and among these factors, which one dominates the overlay's resilience in the face of failures. To answer this question, we formalize the connectivity resilience problem in terms of a graph theoretic problem. The formalization provides us the variables and the key elements that we should focus on.

To simplify our analysis, we assume the existence of two groups of nodes,[1] S and H. The first group (S) of nodes represents the data sources in the group. They connect to the Internet with fast connections. The other group H are the clients, with varying Internet connection speeds. For each node v in the overlay, we define *in-degree* ($D(v)$) and *out-degree* ($\hat{D}(v)$). The out-degree of a node is limited by the node's connection bandwidth. We set a global maximum out-degree of k. We do not impose a limit on a node's in-degree, i.e., a node can connect to any number of hosts to receive data. However, for the whole overlay, the total of all nodes' out-degrees should be equal to the sum of their in-degrees, i.e.,

$$\sum_{v \in V} \hat{D}(v) = \sum_{v \in V} D(v)$$

We assume a uniform failure probability p for all nodes except the data sources S. We formalize the reachability resilience problem as follows.

Definition 1 *Given a set of nodes $V = (S \bigcup H)$, for any node $v \in S \bigcup H$, its out-degree limit is $\hat{D}(v) \leq \mathcal{K}$, where $\mathcal{K}$ is a constant. A directed graph $G = (V, E)$ is $m_{(\mathcal{K},S)}$-connected if G remains connected to S with any node set M removed from V ($|M| = m$ and $M \bigcap S = \varnothing$). G is connected to S if there is at least one directed path from v to node $s \in S$ for any $v \in V - M - S$.*

Definition 2 *Given a set of nodes $V = (S \bigcup H)$, for any node $v \in S \bigcup H$, its out-degree limit is $\hat{D}(v) \leq \mathcal{K}$, where $\mathcal{K}$ is a constant. A directed graph $G = (V, E)$ is $m_{(\mathcal{K},S,P)}$-connected if G remains connected to S with*

[1] In our discussion, when referring to overlays, we use *node* in place of *host*, and *distance* in place of *latency* to be consistent with standard graph theory vocabulary.

probability P when any node set M is removed from V ($|M| = m$ and $M \bigcap S = \varnothing$). G is connected to S with probability P if the probability that there is at least one directed path from v ($v \in V - M - S$) to node $s \in S$ is at least P.

Definition 1 defines a directed graph that remains connected if a set of nodes are removed according to our failure models. Definition 2 defines a directed graph that remains connected with a high probability after the node removal. If the requirement in definition 2 is met, the CCS of the graph should be no less than $|V| * P$ on average.

We first need to understand the complexity to construct a $m_{(\mathcal{K},S)}$-connected or $m_{(\mathcal{K},S,P)}$-connected graph. The $m_{(\mathcal{K},S)}$-connected problem is similar to the *minimum k-connected spanning subgraph* [12] and *minimum k-outconnected spanning subgraph* [13] problems in graph theory. The minimum k-connected spanning subgraph problem has been proven to be NP-hard, and there is a $2k$-approximation algorithm [14][15]. There are efforts in the literature that provide several approximation algorithms for the k-connected spanning subgraph problem with uniform weights on the edges of the graph [16][17]. However, these approximate solutions do not assume any node degree constraint. They assume global knowledge and global cooperation that are not feasible in large distributed systems. It is also difficult to integrate these theoretical algorithms with overlay construction and improvement procedures. As for definition 2, the introduction of the additional probability requirement makes it even more complex than the k-connected spanning subgraph problem.

In this chapter, instead of studying the connectivity problem on the whole graph, which leads us to a NP-complete problem [18], we analyze the problem from the point of view of an individual peer. Intuitively, if all peers in the overlay are highly connected with the others, the probability that the whole graph being partitioned will be low. We provide a general theorem for the reachability probability introduced in definition 2.

2.3.1 Connectivity Probability: General Case

Assume a node v that is h hops from a random data source $s \in S$. With uniform node failure probability p, there are two main factors that affect the probability that v remains connected to s: the number of paths and the length of these paths. In this section, we intend to answer this question: If there are t paths from v to s with lengths $h_1, h_2, ..., h_t$, what is the probability that v will remain connected to s given the uniform node failure probability of p?

We call node v *reachable* from data source s if there is a path between v and s in the overlay. The failure of any node in a path from v to s makes v unreachable from s via that path.

We use $P(path(v,s))$ to represent the probability that v is reachable from s, in which $path(v,s)$ stands for the paths between v and s. We also use $path(v,s,h)$ to represent all the paths between v and s with length h. For example, if v and s are directly connected, this direct path is in $path(v,s,1)$.

For any v in V, the length of paths from v to s varies from 1 to $N-1$, where $N = |V|$. For each path from v to s, its failure probability is determined by the failure possibility of the nodes it traverses. We use $P'(\mathcal{A})$ to stand for the failure probability of path $\mathcal{A}$, and $P(\mathcal{A})$ to represent the probability that path $\mathcal{A}$ remains connected. Clearly, $P(\mathcal{A}) + P'(\mathcal{A}) = 1$. Since we assume the failure of an individual node is independent from the failure of other nodes, the failure of one path ($\mathcal{A}$) from v to s is not mutually exclusive of another ($\mathcal{B}$). In fact, if path $\mathcal{A}$ and $\mathcal{B}$ are node-disjoint paths, their failures are independent of each other. Therefore, the probability that both of them fail would be $P'(\mathcal{A}) \times P'(\mathcal{B})$. If path $\mathcal{A}$ and $\mathcal{B}$ are not node disjoint, the probability that both of them fail would be larger than $P'(\mathcal{A}) \times P'(\mathcal{B})$.

Based on the above observation, we have the following theorem. The proof is available in Appendix A.

Theorem 3 $\forall v \in V, v \neq s$, $P(path(v,s)) \leq 1 - \prod_{h=1}^{N-1} \prod_{u \in V-\{s,v\}} (1 - P(path(v,u,1)) \times P(path(u,s,h-1)))$.

Although the above theorem provides an upper bound on the probability that v is still reachable from s, it does not tell which parameter dominates this reachability probability. However, it is clear that the number of paths from v to s and the lengths of these paths are the two factors controlling the reachability from v to s. In the next section, we study the above problem from a different angle, such that it can provide us with effective heuristics to improve the connectivity probability. Basically, we answer the following two questions:

- Which approach is more likely to increase a node's reachability probability, shortening the existing paths or adding more paths?

- For a node with k paths to a data source, which of the following cases will result in better LCCS for a graph, paths with highly variable lengths or comparable lengths?

3 Resilience Improvement Heuristics

Our goal is not to calculate the reachability probability among peers, but to find specific guidelines to improve overlay connectivity. We obtain these resilience improvement guidelines using both theoretical analysis and empirical experiments. For theoretical analysis, we simplify our assumptions from previous sections, and investigate what kinds of alternative path peers should use. With empirical experiments, we compare graphs with different degree and connectivity characteristics to observe which kinds of graph are more resilient to connectivity failures.

We have observed the following guidelines that can help improve the resilience of overlays to connectivity failures.

- Add more disjoint paths to the data source.

- Increase variance in the length of these disjoint paths.

- Build irregular overlays or random overlays.

- Build overlays with diversified node degree and random links.

3.1 Shorter Paths or More Paths

To simplify the calculation of reachability probability from v to s, we only consider node-disjoint paths from v to s. We assume there are t node-disjoint paths from v to s with lengths $h_1, h_2, ..., h_t$.

Obviously the smaller h is, the more likely v will remain reachable from s, and the larger t is, the more likely v will remain connected to s. We call these the "shorter path" heuristic and "more paths" heuristic respectively. For P2P streaming overlays, it is difficult to keep all h_i's small because of the limited resources contributed by each node. Similarly, t can not be arbitrarily large either. The key question is: Should a peer focus on maximizing t or minimizing h?

If there is only a single path $\mathcal{A}$ from v to s with length h, the probability that v is reachable from s is $(1-p)^h$. If there are t node-disjoint paths from v to s, the probability is $\bar{P} = 1 - \prod_{i=1}^{t}(1-(1-p)^{h_i})$, where h_i represents the length of each path $\mathcal{A}_i$ from v to s.

Now we recalculate the reachability probability $\bar{P}$ shown above after we make the following changes to $path(v,s)$. We call them *heuristic C1* and *heuristic C2* respectively.

- *C1*: reducing the length of all paths in $path(v,s)$ by 1, or

- *C2*: adding a new disjoint path to $path(v,s)$.

Since it is infeasible for us to make meaningful observations with $O(t)$ variables in $\bar{P}$, we assume that all paths in $path(v,s)$ have the same length, i.e., $h = h_1 = h_2 = ... = h_t$. With this assumption, we have $\bar{P} = 1-(1-(1-p)^h)^t)$. We will discuss the effect of various path lengths in the next section.

With *heuristic C1*, the new reachability probability after reducing the length of all paths by 1 is $\bar{P}_{c1} = 1-(1-(1-p)^{h-1})^t$, with the corresponding increase in reachability probability of

$$\begin{aligned}\Delta_{c1} &= \bar{P}_{c1} - \bar{P} \\ &= (1-(1-p)^h)^t - (1-(1-p)^{h-1})^t. \quad (1)\end{aligned}$$

As for *heuristic C2*, the new reachability probability after adding a new disjoint path is $\bar{P}_{c1} = 1-(1-(1-p)^h)^{t+1}$, with the corresponding increase in reachability probability of

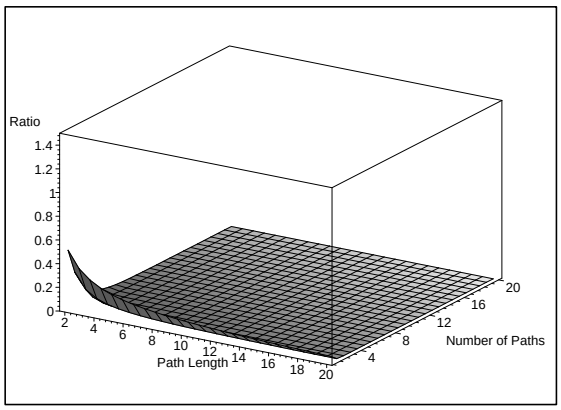
$$\begin{aligned}\Delta_{c2} &= \bar{P}_{c2} - \bar{P} \\ &= (1-(1-p)^h)^t - (1-(1-p)^h)^{t+1} \\ &= (1-(1-p)^h)^t(1-p)^h. \quad (2)\end{aligned}$$

Now, we compare Δ_{c1} and Δ_{c2}.

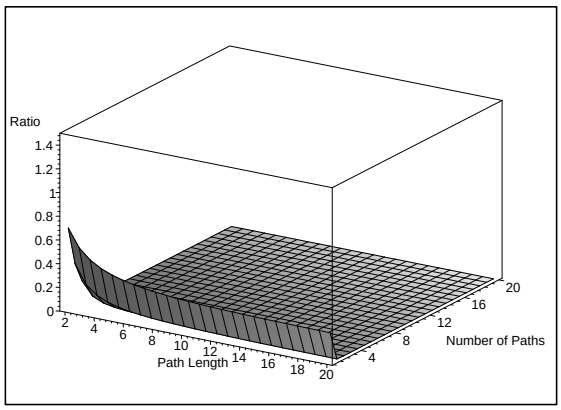
$$\begin{aligned}R = \frac{\Delta_{c1}}{\Delta_{c2}} &= \frac{1}{(1-p)^h}(1-\frac{1-(1-p)^{h-1}}{1-(1-p)^h})^t \\ &= \frac{1}{(1-p)^h}(\frac{(1-p)^{h-1}-(1-p)^h}{1-(1-p)^h})^t \\ &= \frac{1}{\hat{p}^h}(\frac{\hat{p}^{h-1}-\hat{p}^h}{1-\hat{p}^h})^t \quad (3)\end{aligned}$$

In Equation 3, we replace $(1-p)$ with $\hat{p}$.

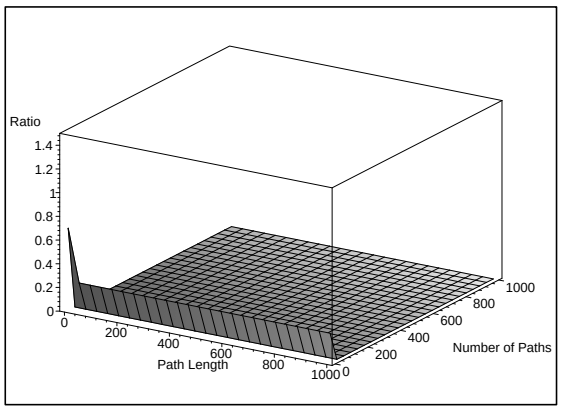
There are three variables in Equation 3, $\hat{p}$, h, and t. Fig. 1 shows their contribution to the value of $\frac{\Delta_{c1}}{\Delta_{c2}}$.



a. Node failure probability p = 0.01



b. Node failure probability p = 0.20



c. Node failure probability p = 0.20, 1000 paths.

Figure 1: Value of R: with two or more disjoint paths, adding a new disjoint path always contributes more to a peer's reachability probability. The flat plane with ratio value close to 0 indicates that Δ_{c1} (improvement contributed by "shorter paths") is negligibly small compared with Δ_{c2} (improvement contributed by "more paths"). The only exception is the case where there is only one path from v to s (the front left corner of the figures), which means when there is only one short path between v and s, shortening the path path can effectively increase the reachability probability, but the amount increased is still more than 30% lower than that of the heuristic "more paths".

If the value of Equation 3 is less than 1, *heuristic C2* contributes more to the improvement of v's reachability probability.

The flat plane with ratio value close to 0 in Fig. 1.a to Fig. 1.c indicates that Δ_{c1} (improvement contributed by "shorter paths") is negligibly small compared with Δ_{c2} (improvement contributed by "more paths"). It is clear that with node failure probabilities of 0.01 and 0.2, heuristic C2 always contributes significantly more to v's reachability. The only exception is the case where there is only one path from v to s (the front left corner of the figures). This means that with one path from v to s, if the path length is short, shortening the path is relatively effective in increasing the reachability probability, but the amount increased is still more than 30% lower than that of the "more paths" heuristic.

Figure 1.a and 1.b only plot the results for 20 disjoint paths with a maximum path length of 20. However, as shown in Figure 1.c, the conclusion we arrived at is applicable to longer path lengths and larger number of disjoint paths. Meanwhile, in overlay networks, h and t will not be arbitrary values. For example, in an overlay with about 1000 nodes with average node degree of 10, t should be less than 10. Then h would be less than 100. Figure 1.c indicates that even though v can reduce each of its 1000 disjoint paths to s by one hop, in terms of reachability probability, it is still far less effective than adding one more disjoint path.

For streaming overlays, the node degree limits constrain both the number of disjoint paths and the length of these disjoint paths a peer can find, especially if the data sources have low degrees. If peers close to the data sources reach their node limit, it is difficult to shorten the lengths of all paths. However, the limitation on the number of disjoint paths can be eased by having high degree data sources and high degree peers near the sources.

We can see that adding additional disjoint path among peers is an effective way to increase the connectivity among peers. Our next question is: When a peer selects a disjoint path, what kind of path should it select? Does the length of the disjoint path matter?

3.2 Lengths of Paths

In the previous section we concluded that creating more disjoint paths is more helpful in increasing the connectivity among peers, with the assumption that all the disjoint paths have the same path length. In this section, we study the effect of paths lengths. Particularly, given t disjoint paths from node v to s, do we want these paths to have similar path lengths or do we want them to have variable path lengths?

We continue to use h_i to represent the length of path $\mathcal{A}_i$ from v to s and $\hat{p}$ for $(1-p)$. With t disjoint paths from v to s, the probability that v is reachable from s is $\bar{P} = 1 - \prod_{i=1}^{t}(1-\hat{p}^{h_i})$. We assume $t > 1$.

Our first step is to study the trend of $\bar{P}$, i.e., how $\bar{P}$ changes with different sets of h_i values. To make a fair comparison among different groups of disjoint paths, we assume the sum of the length of disjoint paths to be a constant T ($\sum_{i=1}^{t} h_i = T$).

We first study the trend of function f.

$$f = 1 - \bar{P} = \prod_{i=1}^{t}(1-\hat{p}^{h_i})$$

Since $\sum_{i=1}^{t} h_i = T$, we have $\prod_{i=1}^{t} \hat{p}^{h_i} = \hat{p}^T$.

Letting $x_i = \hat{p}^{h_i}$ and $C = \hat{p}^T$, function f can be expressed in the following equation,

$$f = (1-x_1)(1-x_2)\cdots(1-x_t)$$

where $x_1 x_2 \cdots x_t = C$ and $0 < x_1, x_2, \cdots, x_t < 1$.

Given that $x_1 x_2 ... x_t = C$, we can get

$$\begin{aligned} f &= (1-x_1)(1-x_2)...(1-x_t) \\ &= (1-x_1)(1-x_2)...(1-\frac{C}{x_1 x_2 ... x_{t-1}}) \end{aligned}$$

Then we have

$$\frac{\partial f}{\partial x_i} = \prod_{j \neq i}(1-x_j)[\frac{x_t}{x_i} - 1]$$

and

$$\frac{\partial^2 f}{\partial^2 x_i} = \prod_{j \neq i}(1-x_j)\frac{-x_t}{x_i^2}$$

It is obvious that when $x_1 = x_2 = ... = x_n$, $\forall i, \frac{\partial f}{\partial x_i} = 0$ and $\frac{\partial^2 f}{\partial^2 x_i} < 0$, which means when $x_1 = x_2 = ... = x_t$, the function f will achieve its maximum value. With $0 < x_1, x_2, \cdots, x_t < 1$, $\frac{\partial^2 f}{\partial^2 x_i}$ is always less than 0 and function f has only one extreme value.

Since function $f = 1 - \bar{P}$, then $\bar{P}$ has only one minimal value in the range of $0 < x_1, x_2, \cdots, x_t < 1$. $\bar{P}$ is monotonically increasing when $x_1, x_2, \cdots, x_t$

get further away from the minimal point where $x_1 = x_2 = ... = x_n$.

Based on the observation of function $\bar{P}$, we know that the reachability probability from v to s will be high if we have different lengths of disjoint paths from v to s. In other words, a peer can increase its connectivity by adding more disjoint paths with different lengths.

3.3 k-Regular Graph vs. Irregular Graph

We have studied, from the point of view of a single node, how to improve its connectivity to data sources. However, we have not investigated, in general, what characteristics make a graph resilient to failures. Such macro-level knowledge can provide us with general guidelines for overlay construction. In this section, we look at the resilience of regular graphs and irregular graphs constructed by overlay protocols.

We already know from existing efforts that the connectivity of a graph depends on both the node degree distribution and the way nodes are interconnected. Here, we first assume peers are randomly interconnected such that we can focus on the effect of degree distribution. We investigate the effect of different node interconnects in the next section.

To study the impact of the regularity on overlay connectivity, we create a 4-regular graph and compare it with TMesh, which builds an irregular overlay or unstructured overlay. Note that a 4-regular graph uses 42% more links than that of TMesh. We also build a 6-regular graph, which uses about 2.1 times the number of links as TMesh. Then we compare the resilience of these two regular graphs with TMesh. The results for random node removal and preferential node removal are shown in Figs. 2 and 3 respectively. The ratios reported on the y-axis are normalized to the group size after node removal. For example, in this experiment, we use a group of 1000 nodes, after removing 200 nodes, the LCCS for the ideal case where the overlay remains connected is 800 nodes. We report the ratio of LCCS to group size as 1 (800/800) in the figures[2].

Intuitively the more links an overlay uses, the more tolerant it is to failures because the extra links tend to keep the overlay connected. However, from Figs. 2 and 3, it is obvious that the 4-regular graph shows very little resilience to node failures. Random node removal and preferential node removal remove similar sets of nodes in a regular graph since all nodes in a regular graph share the same node degree (the node degree in TMesh varies from one to ten). The figures tell us that even though a regular graph uses a larger number of links, its connectivity resilience is not as good as an irregular graph in terms of LCCS.

Even with 6-regular graph, the graph starts to fall apart rapidly after 7% of nodes fail or leave. The LCCS of a 6-regular graph gets worse than that of TMesh after 8% nodes are randomly removed. Considering that the average node degree of TMesh is only 2.8, nodes in the TMesh overlay are more tightly connected with each other than these regular graphs.

The performance metric we adopt—LCCS—is quite different from the connectivity metric used in fault-tolerance study of graph theory. For instance, a 4-regular graph is 4-connected, which means that removing any set of 3 nodes leaves the graph connected. The TMesh overlay we built is not even a 2-connected graph. However, it is more tolerant to node failures if our objective is to achieve high LCCS, as opposed to keeping all nodes connected under all circumstances. A regular graph reduces to several partitioned components with similar sizes after a number of node failures.

In practice, building regular graphs on the Internet is not realistic because of varied CPU power and connection speed of Internet hosts. Fortunately, it turned out irregular graphs are more suitable for overlay streaming on the Internet.

3.4 Effect of Node Degree and Node Interconnection

One key difference between TMesh and 4-regular graphs is their node degree distributions. Nodes in the TMesh overlay have node degree varying from one to ten. In this section, we study the role of node degree distribution and the effect of different interconnection schemes on the failure resilience of overlays.

It has been shown that in power-law graphs, node degree distribution has limited correlation with connectivity [19]. We need to confirm whether this conclusion can also be applied to small-degree graphs like the streaming overlay. More importantly, we are interested in whether graphs with certain node degree distributions are more likely to tolerate connectivity failures.

[2]The details of the experimental setup are available in Section 5.1

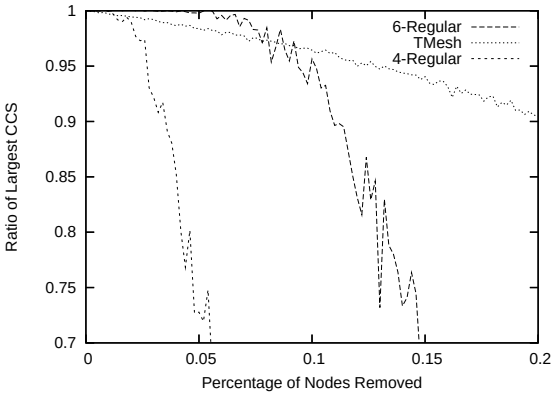


Figure 2: Ratio of the largest connected component size to graph size after uniform random node removal

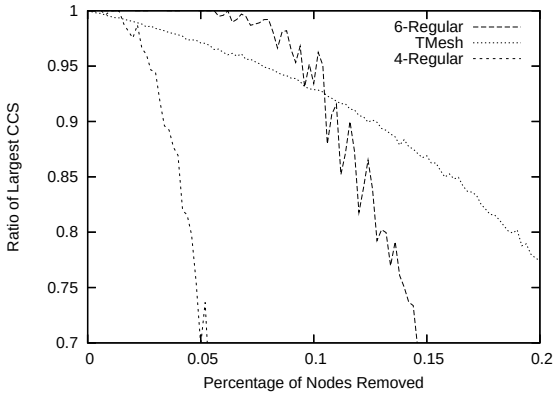


Figure 3: Ratio of the largest connected component size to graph size after preferential node removal

We construct two *regular* graphs that use the same number of links as the 4-regular graph but with different node degree distribution. We name them "two-degree" and "three-degree" graphs. We construct these two graphs as follows.

- two-degree: We first assign a degree limit of two to all nodes and connect them in a circle. We then uniformly pick k nodes at random and increase their node degree limit to 10 so that the sum of all nodes' degrees is the same as that of the 4-regular graph. Links are then randomly added to the graph so that all nodes reach their designated degree limit.
- three-degree: Similar to "two-degree" graph, all nodes are first connected in a circle. Then k' nodes are randomly selected to have degree limit of 6 and another k' nodes are randomly selected to have degree limit of 8.

Fig. 6 summarizes the node degree distribution of these graphs. Figs. 4 and 5 show their resilience to node failures. It is clear that with the same number of links as the 4-regular graph, two-degree and three-degree graphs have much improved resilience to failures. We can observe the trend that graphs with higher diversities in node degrees are more resilient to node failures.

We notice that high degree nodes in these two regular graphs do not effectively improve the resilience of these graphs to node failures. Even though two-degree and three-degree graphs have a significantly larger number of high degree nodes than that of TMesh (Fig. 6), their resilience to connectivity failure is not significantly better. We believe that this is caused by the way high degree nodes are interconnected. To confirm this speculation, we construct "two-degree" and "three-degree" graphs in a way we call *high-degree node clustered*. Instead of randomly picking k nodes to increase their degree limit, we pick k consecutive nodes along the initial circle and increase their degree. We show the results for preferential node removal in Fig. 7. Comparing Figs. 5 and 7, we can see that having high degree nodes clustered together significantly lowers the resilience of a graph to connectivity failures. This means that for a content provider, it is better to distribute their high bandwidth servers to different places in streaming overlays. Keeping these servers closely connected together reduces the reliability of the services in the face of malicious attacks.

Considering the way we constructed the new "two-degree" and "three-degree" graph, we conclude that by increasing the diversity in node degrees and randomizing the link selection of a graph, we can improve a graph's connectivity resilience. Adding randomized links can connect two nodes that are far from each other together. Intuitively, these links increase the number of disjoint path from a node to a data source. This is consistent with the "more paths" heuristic we have from the theoretical analysis in Section 3.2.

However, randomized link selection normally degrades the end-to-end latencies, which may result in

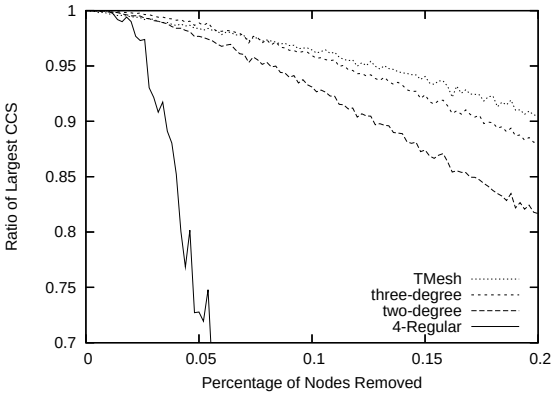


Figure 4: Ratio of the largest connected component size after uniformly random node removal

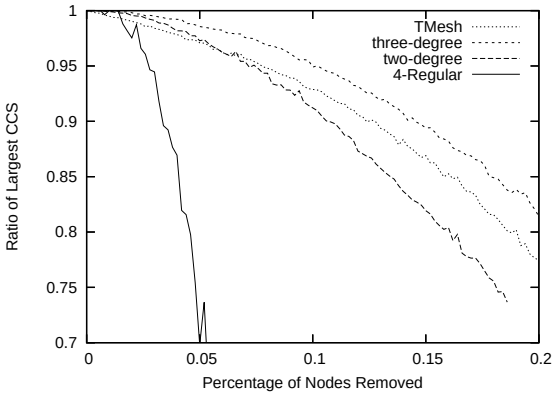


Figure 5: Ratio of the largest connected component size after preferential node removal

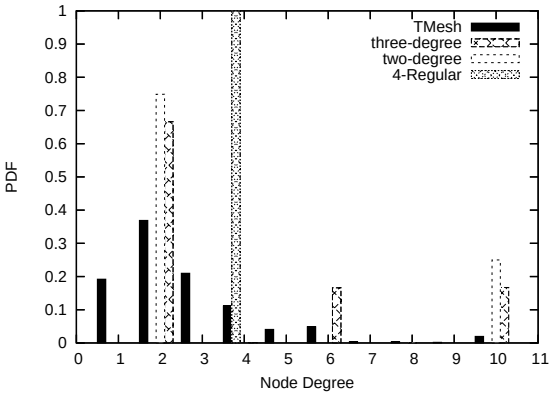


Figure 6: Node degree distribution of 4-Regular, two-degree and three-degree graphs, and TMesh.

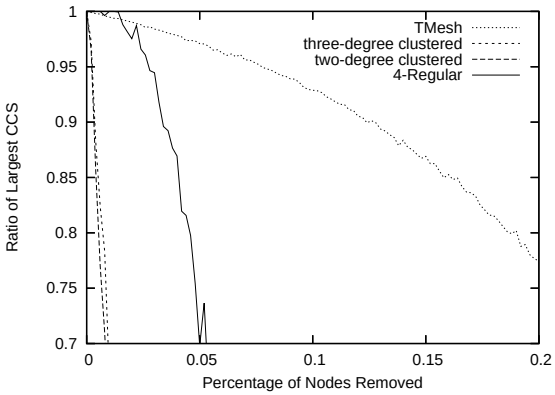


Figure 7: Ratio of LCCS after preferential node removal for overlays with high degree nodes clustered

inefficient overlay networks and cause high overhead. In the next section, we design an overlay improvement heuristic that can effectively improve the connectivity resilience of overlays without compromising their efficiency.

4 Overlay Link Selection with MPath

We have observed the following guidelines that can help improve the resilience of overlays to connectivity failures: add more disjoint paths to the data source with different lengths and build irregular overlays or random overlays with diversified node degrees and random links.

We now need a heuristic that can enable a peer in an overlay to independently select its downstream or upstream peers such that the overlay can achieve good connectivity resilience overall. To design such a heuristic, we should take into account how to generate random graphs, how to calculate node-disjoint paths, and how to increase the variance in the length of disjoint paths in a distributed fashion.

Most unstructured overlay networks, such as $e*$ [20], HMTP [21], Narada [22], and TMesh [23], build irregular graphs. For our heuristic design, we focus on increasing the number of disjoint paths with various lengths among peers in these irregular overlays.

4.1 Calculation of Disjoint Paths

With root-path information [24], the calculation of node-disjoint paths is straightforward. A peer A sends a query to peer B for B's root-path. By comparing B's root-path with its own root-path, A knows whether adding a new link to B will create a node-disjoint path to the root. For overlay protocols that do not maintain root-path information, such as Narada, routing tables can be used to calculate node-disjoint paths. For example, peers in Narada maintain path vector routing tables. Peer A can access B's routing table and decide whether B is a candidate for a node-disjoint path.

4.2 Variance in Disjoint Path Length

As stated earlier, it is preferable to have a greater variance in the length of disjoint paths. To increase the variance in path lengths, a peer should connect to other peers that meet the following criterion: candidate peers create paths to the data source with high δ value, where δ is the difference in the length of the paths. For example, a peer p with root-path length of r, would prefer peers that have root-path of length $r - \delta$ and $r + \delta$ where δ should be maximized.

It is not feasible for a peer to query all other peers for various disjoint paths, but it is possible for a peer to randomly query a certain number of peers before finding a qualified peer. The searching procedure can be slow. There are several ways to speed up the process. First, a peer can query the data source or peers near the data source for their connected neighbors, and check whether these neighbors qualify for node-disjoint paths. This approach does not scale well because it adds extra workload to peers near the data source. Second, an overlay can distribute the connectivity information of all its peers to the whole group. Peer A broadcasts its root-path or connectivity information such that others peers know whether peer A can provide them with node disjoint paths. This approach generates high overhead.

We take a third approach in our heuristic. Since most overlay networks have their own overlay improvement procedure, it would be very efficient to integrate the disjoint path search into the overlay improvement process. For example, in Narada, a peer randomly picks other peers and calculates the "utility" metric based on the routing information. If the "utility" is above a threshold, a new link is added. In TMesh, extra shortcut links are added based on either relative delay penalty (RDP) or end-to-end delay gain. We only need to conduct one additional path check during these link selection procedures. We check whether the peer will provide a node-disjoint path, and if so, what is the difference in the distance to the data source. If the difference δ in the path length is high, this link should be added. We call this link selection heuristic "MPath". The integration of MPath with the overlay improvement procedure significantly reduces the overhead incurred for overlay resilience.

In the guidelines presented in previous sections, three out of the four guidelines are related to randomizing the properties of the overlay built. It is possible that adding random links into the overlay will help the fault-tolerance of the overlay. We investigate the effect of random links in our performance evaluation.

Table 1: Hosts with different bandwidths

Connection speed	Max degree	% of hosts
below 100 Kbps	2	10%
100 Kbps - 2 Mbps	4	30%
2 Mbps - 10 Mbps	6	40%
above 10 Mbps	10	20%

5 Performance Evaluation

The evaluation of heuristic MPath consists of two parts. We first demonstrate that MPath can effectively improve the connectivity resilience of overlay networks. Then, we show that the use of MPath does not compromise the performance of the overlay network. Before we delve into the details of our evaluation, we discuss our experiment setup. This setup also applies to the experiments we conducted in previous sections.

5.1 Experiment Setup

In our experiment, we employ a transit-stub topology of 4,400 nodes generated by the GT-ITM topology generator [25], in which we randomly pick 1000 nodes to participate in the overlay network. We use a realistic node degree setting rather than uniform degree distribution. Saroiu *et al.* [26] studied the speed of Internet connections of hosts in P2P file-sharing networks. They found that in the Gnutella network, about 10% of hosts had connection speed less than 100 Kbps, 30% between 100 Kbps and 2 Mbps, 40% between 2Mbps and 10 Mbps, and the remaining 20% had connection speed higher than 10 Mbps. In our experiments, we assume a similar bandwidth distribution among hosts in our overlays. We simulate the network capacity of a host by appropriately limiting its maximum node degree (see Table 1). For instance, a host with connection speed less than 100 Kbps would have a maximum node degree of two. Note that the maximum degree is not set proportional to its connection speed because having higher node degree or serving more clients will consume increasingly more CPU cycles, which are important for applications with high data rate. We limit the maximum node degree to ten.

In the experiments, we first constructed an overlay using a given overlay protocol. Then a certain number of nodes or edges are removed from the constructed overlay according to the failure scenario used. For each overlay scheme, we repeat our experiment 100 times for a given failure scenario and a certain percent of failed nodes or edges.

5.2 Heuristic MPath

In addition to the MPath heuristic we presented in the last section, we add a variant of MPath named "MPath Greedy". With *MPath Greedy*, instead of integrating the node-disjoint path searching with the overlay improvement process, a peer completely focuses on adding disjoint paths to other peers. With the original MPath heuristic, we limit each peer to have at most three disjoint paths to the data source. *MPath Greedy* has no limits on the number of disjoint paths each peer can have, but it obeys the same node degree limits as the original *MPath* heuristic. To make the comparison fair, we also force *MPath Greedy* to use the same number of links as the original MPath.

We apply the *MPath* and *MPath Greedy* heuristics to the TMesh protocol and present resilience result in Figs. 8 and 9. We can see that both MPath versions greatly improve the resilience of TMesh to connectivity failures. With 20% of high degree nodes removed, the original TMesh overlay is broken into pieces where the largest piece consists of less than 78% of the nodes. With the MPath heuristic integrated, the ratio is increased to 96%.

In Fig. 10, we show the number of disconnected segments of TMesh overlay after preferential node removal. Remember that a disconnected segment is a partitioned component with two or more nodes and it is more detrimental to the recovery of the overlay than a disconnected node. Fig. 10 shows that MPath can significantly reduce the number of disconnected segments. With 20% of high degree nodes removed, the average number of disconnected segments is less than 4, almost 10 times less than that of the original TMesh.

Fig. 11 shows the average end-to-end latency of the overlays built by the MPath heuristic. We vary the group size from 50 nodes to 1000 nodes. The metric we use is Average Relative Delay Penalty (ARDP) defined in TMesh [23]. RDP is the ratio of the latency $D'_{i,j}$ between a node pair i and j on the overlay to the latency $D_{i,j}$ between them on the physical network. ARDP is then the average RDP between all node pairs in the overlay. We can see that MPath performs almost as well as the original TMesh.

In Fig. 8 to Fig. 10, we include the HMTP overlay

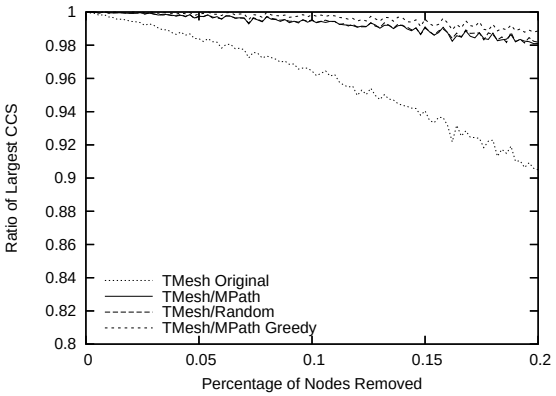


Figure 8: Ratio of the LCCS to group size after uniformly random node removal

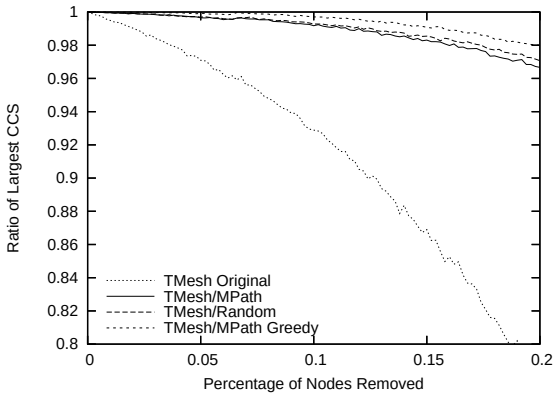


Figure 9: Ratio of the LCCS to group size after preferential node removal

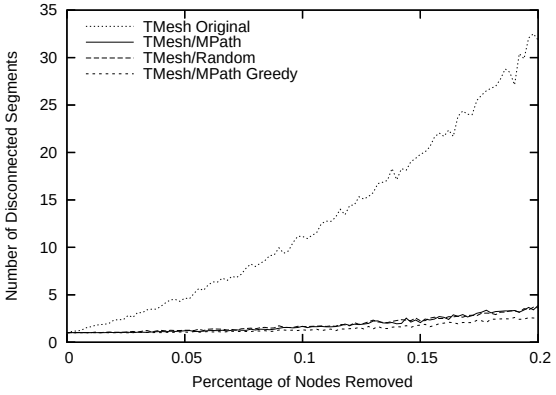


Figure 10: Number of disconnected segments after preferential node removal

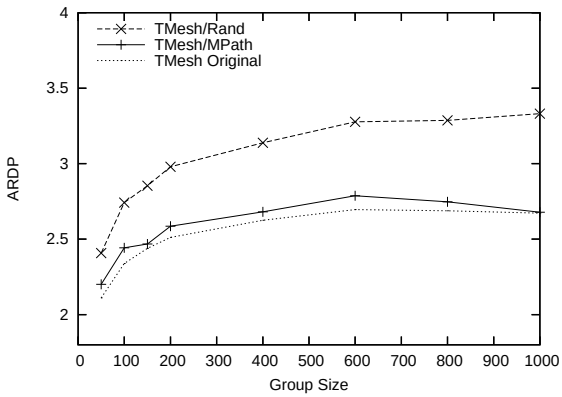


Figure 11: Average end-to-end latency performance of MPath heuristic

with random extra links (HMTP/Rand)[3]. We can see that HMTP with random extra links shows good resilience to node failures. However, random links introduce higher end-to-end latency. With the same number of links, HMTP with random extra links perform 11% worse than that of the MPath heuristic (Fig. 11).

5.3 Effect of Random Links

The good performance of random links in terms of connectivity resiliency agrees with the observation we had during our study on the effect of path lengths and node degree distributions. That is, random links tend to create paths with various lengths and nodes with random degrees.

To confirm this conjecture, we show the path length distribution and node degree distribution of the random graph in Figs. 12 and 13. For path length distribution, we first find all disjoint paths of a node to the data source, then calculate the difference between the length of these disjoint paths and the shortest path to the data source. We report the CDF of these length difference in Fig. 12. From Figs. 12 and 13, we can clearly see the similarity in the path length distribution and node degree distribution in the overlay with random links and MPath links, which explains the good connectivity resilience contributed by adding extra random links.

6 Related Work

Fault-tolerant networking has been widely studied and investigated in graph theory and network practice. A large amount of research effort has been devoted to creating fault-tolerant networks in the context of parallel computing [27][8][28] [29][30] and data communication networks [31][32][33] [34][35][36][37]. Most of these efforts are based on some graph-theoretic models of fault-tolerance [38]. The basic idea is to add redundancy to the network—which is viewed as a graph—such that after the removal of a certain number of nodes, the remaining graph is still connected. Another approach used in parallel computing is to utilize the remaining part of the network to simulate the whole system so that failure can be concealed [27][8]. These approaches normally cause significant degradation in system performance.

The unique characteristics of overlay networks, including limited network resources and group dynamics, make their fault-tolerance improvement solutions different from that of parallel computing or communication networks. The limited network resource of each individual peer constrains its node degree, i.e., the number of peers it can support. The fault-tolerance techniques adopted must not overload overlay nodes that come with various hardware and software configurations. Compared to CPUs in a parallel computer or routers on the Internet, a node in an overlay network has a much higher failure probability. The churn in overlay networks may make the approaches that result in instability and high network overhead [27][8] inapplicable.

Meanwhile, the unique characteristics of overlays provide more flexibility in connectivity resilience improvement. Since links in overlay networks are actually virtual links (TCP connections or UDP sessions), new overlay links can be added promptly with fairly low cost. This is usually not possible for parallel computers or communication networks. An overlay can be easily adjusted over time based on the dynamics of the overlay network. For example, a node in an overlay can monitor the number of alternative paths in the overlay to the media source. If the number is too small, it can start to add new links to other nodes in the group to ensure the existence of an alternative path in case the current path to the sender fails.

Recent deployment of overlay multicast has attracted efforts to improve fault-tolerance of overlay networks. Dual-Tree [39], kTree [40], RITA[41], and RON [7] are typical examples. The basic ideas of Dual-Tree and kTree are similar. Multiple tree structures are constructed to provide redundancy in the connectivity among peers. kTree constructs multiple minimum spanning trees, which incur high overhead. Dual-Tree creates an extra tree structure among a subset of the peers, which may be suitable for IP multicast networks, but not sufficient for overlay streaming networks due to the high failure probability of overlay nodes. In RITA, a peer initiates new links when its perceived QoS drops below a threshold. With multiple node failures that partition the network, a peer may not be able to recover from such failures. Resilient overlay network (RON) implements an overlay in which nodes monitor the quality of the overlay links among themselves. In case the link between two nodes fails or becomes congested, RON's

[3]The TMesh overlay we used here is based on HMTP protocol. The difference between TMesh and HMTP/Rand is that TMesh employs extra links that are selected to reduce end-to-end latencies

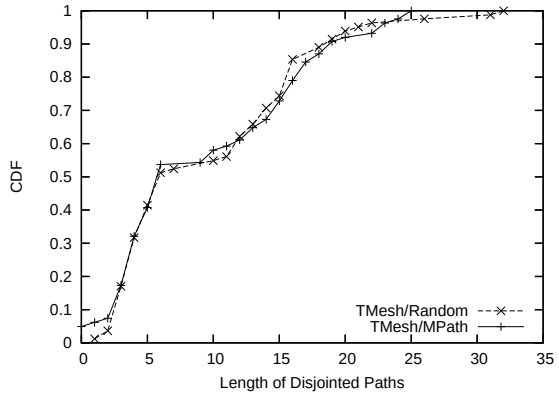


Figure 12: The distribution of path length difference of overlays with random and MPath heuristics

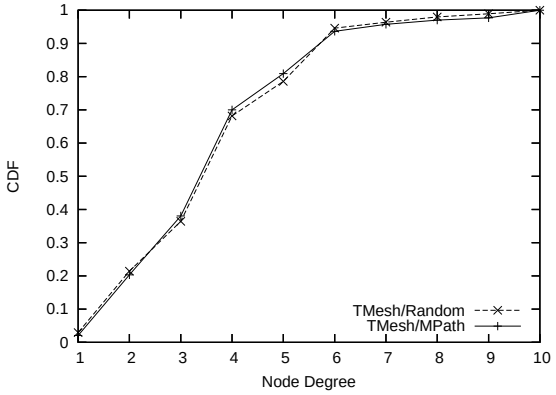


Figure 13: Node degree distribution of overlays with random and MPath heuristics

routing algorithm will find an alternative path for these two nodes. Compared to kTree and Dual-Tree, one apparent benefit of alternative paths is that it does not require additional overhead to maintain the extra tree structures.

The low degree limit in streaming overlays distinguishes overlay networks from networks with power-law properties. Studies have shown that power-law graphs are vulnerable to attacks on high degree nodes [19]. This is not necessarily the case in overlay networks. Low degree graphs present quite different properties from power-law networks in terms of degree distribution and fault-tolerance.

For streaming networks, where there are usually limited numbers of data sources per overlay, it is possible that the data sources will be under denial of service (DoS) attack. With the data source brought offline, it is not as important any more to keep the remaining part of the overlay connected. There are existing efforts on protecting the data sources from DoS attacks [42][43]. This topic, however, is not the focus of this chapter.

7 Conclusion and Future Work

In this chapter, we have investigated a heuristic to improve the connectivity resilience of streaming overlays. It is important for streaming overlays to maintain connectivity in case of membership churn, node failures, and malicious attacks. We employed both theoretical analysis and empirical methods to study the factors affecting resiliency of overlays in the face of connectivity failures. Our study shows that adding disjoint paths always helps to increase the chances of keeping nodes connected. Paths created with varied lengths can further improve these chances. Overlays with randomly added links and diversified node degrees also tend to be more robust to failures. Based on these observations, we designed the heuristic MPath, which effectively increases the resilience of streaming overlays. Furthermore, we integrated the MPath heuristic with the overlay improvement procedure such that it does not generate much overhead.

In addition to building overlays that are resilient to failures, for streaming overlays with a single or limited number of sources, protecting sources from DDoS attacks becomes an important and challenging issue. With the data sources brought offline, keeping the overlay connected does not help improve the streaming quality. Designing an overlay that can protect the data sources from DDoS attacks will be an interesting topic to explore in future work.

A Appendix

For a given graph $G = (V, E)$, assuming data source $S \in V(G)$, we will analyze the upper bound of the probability P that v is reachable from S if each node $v \in V$ has a failure probability of p. Note that $\forall d_1, d_2 \in V(G)$, $Path(S, d_1)$ and $Path(S, d_2)$ are correlated but not mutually exclusive. This observation makes the analysis of P's upper bound possible.

Definition 4 *For $\forall D \in V(G), D \neq S, Path(S,D)$ means there is a simple path between S and D. Except node D, all nodes on Path(S,D) have a failure probability of p. The probability of $Path(S,D)$ equals the probably that S is reachable from D.*

Definition 5 *For $\forall D \in V(G), D \neq S, Path(S,D,i)$ means that there is a simple path between S and D with length i.*

Definition 6 *For $\forall D \in V(G), D \neq S, R \in N(S), CP(S,R,1,R,D,i-1)$ indicates that there is a simple path between S and D with length i. The next node on the path is R.*

Note that we only consider simple paths in the above definitions. This is because the existence of simple path (S,D) is the necessary condition for S to be reachable from D. To consider the case where S is reachable from D, considering the existence of a simple path is sufficient.

Lemma 7 *For $\forall D \in V(G), D \neq S$,*

$$P(Path(S,D) \leq 1 - \prod_{i=1}^{N-1} P(\overline{Path(S,D,i)}).$$

Proof:

$$\begin{aligned}
& P(Path(S,D)) = 1 - P(\prod_{i=1}^{N-1} \overline{Path(S,D,i)}) \\
= \;& 1 - P(\overline{Path(S,D,N-1)} | \prod_{i=1}^{N-2} \overline{Path(S,D,i)}) \\
& \times P(\prod_{i=1}^{N-2} \overline{Path(S,D,i)}) \\
\leq \;& 1 - P(\overline{Path(S,D,N-1)}) \times P(\prod_{i=1}^{N-2} \overline{Path(S,D,i)}) \\
& \cdots\cdots \\
\leq \;& 1 - \prod_{i=1}^{N-1} P(\overline{Path(S,D,i)})
\end{aligned}$$

■

Lemma 8 *For $\forall D \in V(G), D \neq S, R \in V(G) - \{S,D\}$,*

$$\begin{aligned}
& P(\prod_{R \in V(G)-\{S,D\}} \overline{CP(S,R,1,R,D,i-1)}) \\
\geq \;& \prod_{R \in V(G)-\{S,D\}} P(\overline{CP(S,R,1,R,D,i-1)}).
\end{aligned}$$

Proof:

$$\begin{aligned}
& P(\prod_{R \in V(G)-\{S,D\}} \overline{CP(S,R,1,R,D,i-1)}) \\
= \;& P(\overline{CP(S,v,1,v,D,i-1)} | \\
& \prod_{R \in V(G)-\{S,D,v\}} \overline{CP(S,R,1,R,D,i-1)}) \\
& \times P(\prod_{R \in V(G)-\{S,D,v\}} \overline{CP(S,R,1,R,D,i-1)}) \\
\geq \;& P(\overline{CP(S,v,1,v,D,i-1)}) \\
& \times P(\prod_{R \in V(G)-\{S,D,v\}} \overline{CP(S,R,1,R,D,i-1)}) \\
\cdots \;& \cdots \\
\geq \;& \prod_{R \in V(G)-\{S,D\}} P(\overline{CP(S,R,1,R,D,i-1)})
\end{aligned}$$

■

Theorem 9 *For $\forall D \in V(G), D \neq S$,*

$$P(Path(S,D)) \leq 1 - \prod_{i=1}^{N-1} \prod_{R \in V(G)-\{S,D\}} (1 - P(Path(S,R,1)) \times P(Path(R,D,i-1))).$$

Proof:

$$\begin{aligned}
& P(Path(S,D) \leq 1 - \prod_{i=1}^{N-1} P(\overline{Path(S,D,i)}) \\
= \;& 1 - \prod_{i=1}^{N-1} P(\prod_{R \in V(G)-\{S,D\}} \overline{CP(S,R,1,R,D,i-1)}) \\
\leq \;& 1 - \prod_{i=1}^{N-1} \prod_{R \in V(G)-\{S,D\}} P(\overline{CP(S,R,1,R,D,i-1)}) \\
= \;& 1 - \prod_{i=1}^{N-1} \prod_{R \in V(G)-\{S,D\}} (1 - P(CP(S,R,1,R,D,i-1)) \\
= \;& 1 - \prod_{i=1}^{N-1} \prod_{R \in V(G)-\{S,D\}} (1 - P(Path(S,R,1) \\
& \times P(Path(R,D,i-1)))
\end{aligned}$$

■

References

[1] Y. Chu, S. Rao, S. Seshan, and H. Zhang. Enabling Conferencing Applications on the Internet using an

Overlay Multicast Architecture. In *Proc. of ACM SIGCOMM '01*, San Diego, CA, USA, Aug. 2001.

[2] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum. CoolStreaming/DONet: A Data-driven Overlay Network for Live Media Streaming. In *Proc. of IEEE INFOCOM '05*, Miami, FL, USA, Mar. 2005.

[3] ACM SIGCOMM 2002 Conference. http://www.acm.org/sigcomm/sigcomm2002/.

[4] ACM SIGCOMM 2004 Webcast Statistics. Available from http://warriors.eecs.umich.edu/tmesh/sigcommstat/.

[5] 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, Aug. 2004.

[6] S. Shenker. Fundamental Design Issues for the Future Internet. *IEEE Journal on Selected Areas in Communication*, 13(7), Sep. 1995.

[7] D. Andersen, H. Balakrishnan, M. Kaashoek, and R. Morris. Resilient Overlay Networks. In *Proc. of ACM Symposium on Operating Systems Principles*, Banff, Canada, Oct. 2001.

[8] F.T. Leighton, B. M. Maggs, and R. K. Sitaraman. On the Fault Tolerance of Some Popular Bounded-Degree Networks. *SIAM Journal on Computing*, 27(5):1303–1333, 1998.

[9] J.W. Mao and C.B. Yang. Shortest Path Routing and Fault-Tolerant Routing on de Bruijn Networks. *Networks*, 35(3):207–215, Apr. 2000.

[10] Steve S. Skiena. *The Algorithm Design Manual*. Springer; 1 Edition, 1998.

[11] F. Harary. *Graph Theory*. Addison Wesley Publishing, 1995.

[12] A. Czumaj and A. Lingas. On Approximability of the Minimum-Cost k-Connected Spanning Subgraph Problem. In *Proc. of ACM-SIAM Symposium on Discrete Algorithms (A Conference on Theoretical and Experimental Analysis of Discrete Algorithms)*, Baltimore, MD, USA, Jan. 1999.

[13] V. Auletta, Y. Dinitz, Z. Nutov, and D. Parente. A 2-Approximation Algorithm for Finding an Optimum 3-Vertex-Connected Spanning Subgraph. *J. Algorithms*, 32(1):21–30, 1999.

[14] J. Cheriyan and R. Thurimella. Approximating Minimum-Size k-Connected Spanning Subgraphs via Matching. *IEEE Symposium on Foundations of Computer Science*, 30(2):528–560, 2000.

[15] G. Kortsarz and Z. Nutov. Approximating Node Connectivity Problems via Set Covers. In *Proc. of the Third International Workshop on Approximation Algorithms for Combinatorial Optimization*, Saarbrcken, Germany, Sep. 2000.

[16] J. Cheriyan, T. Jordn, and Zeev Nutov. On Rooted Node-Connectivity Problems. *Algorithmica*, 30(3):353–375, 2001.

[17] H. N. Gabow. A Representation for Crossing Set Families with Applications to Submodular Flow Problems. In *Proc. of the Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, Austin, TX, USA, Jan. 1993.

[18] D. Karger. Randomization in Graph Optimization Problems: A Survey. *Optima*, 58:1–11, 1998.

[19] R. Albert, H. Jeong, and A.-L. Barabasi. Attack and Error Tolerance of Complex Networks. *Nature*, 406, 2000.

[20] W. Wang, C. Jin, and S. Jamin. Network Overlay Construction under Limited End-to-End Reachability. In *Proc. of IEEE INFOCOM '05*, Miami, FL, USA, Mar. 2005.

[21] B. Zhang, S. Jamin, and L. Zhang. Host Multicast: A Framework Delivering Multicast To End Users. In *Proc. of IEEE INFOCOM '02*, New York, NY, USA, Jun. 2002.

[22] Y. Chu, S. Rao, and H. Zhang. A Case For End System Multicast. In *Proc. of ACM SIGMETRICS '00*, Santa Clara, CA, USA, Jun. 2000.

[23] W. Wang, D. Helder, S. Jamin, and L. Zhang. Overlay Optimizations for End-host Multicast. In *Proc. of the Fourth International Workshop on Networked Group Communications*, Boston, MA, USA, Oct. 2002.

[24] P. Francis. Yoid: Extending the Internet Multicast Architecture. *Unrefereed report*, Apr. 2000. http://www.aciri.org/yoid.

[25] K. Calvert, M. Doar, and E. Zegura. Modelling Internet Topology. In *IEEE Communications Magazine*, Jun. 1997.

[26] S. Saroiu, P.K. Gummadi, and S.D. Gribble. A Measurement Study of Peer-to-Peer File Sharing Systems. In *Proc. of Multimedia Computing and Networking 2002 (MMCN '02)*, San Jose, CA, USA, Jan. 2002.

[27] J. Bruck, R. Cypher, and D. Soroker. Tolerating Faults in Hypercubes Using Subcube Partitioning. *IEEE Transactions on Computers*, 41(5):599–605, May 1992.

[28] B.A. Izadi and F. zgner. Enhanced Cluster k-Ary n-Cube, A Fault-Tolerant Multiprocessor. *IEEE Trans. Computers*, 52(11):1443–1453, 2003.

[29] L. Zhang. Fault Tolerant Networks with Small Degree. In *Proc. of ACM Symposium on Parallel Algorithms and Architectures*, Barcelona, Spain, Jun. 2000.

[30] C.T. Ho and L. J. Stockmeyer. A New Approach to Fault-Tolerant Wormhole Routing for Mesh-Connected Parallel Computers. *IEEE Trans. Computers*, 53(4):427–439, 2004.

[31] N.A. Nordbotten, M.E. Gmez, J. Flich, P. Lpez, A. Robles, T. Skeie, O. Lysne, and J. Duato. A Fully Adaptive Fault-Tolerant Routing Methodology Based on Intermediate Nodes. In *Proc. of IFIP International Conference on Network and Parallel Computing*, Wuhan, China, Oct. 2004.

[32] E. Ayanoglu, C. Gitlin, and J. Mazo. Diversity Coding for Transparent Self-healing and Fault-tolerant Communication Networks. *IEEE Transactions on Communications*, 41(11):1677–1686, Nov. 1993.

[33] Z. Ye, S. V. Krishnamurthy, and S. K. Tripathi. A Routing Framework for Providing Robustness to Node Failures in Mobile Ad Hoc Networks. *Ad Hoc Networks Journal*, 2(1):87–107, 2004.

[34] M.D. Schroeder, A.D. Birrell, M. Burrows, H. Murray, R. M. Needham, and T. L. Rodeheffer. Autonet: A High-speed, Self-Configuring Local Area Network Using Point-to-Point Links. *IEEE Journal on Selected Areas in Communications*, 9(8), Oct. 1991.

[35] R. Casado, A. Bermdez, F. J. Quiles, J. L. Snchez, and J. Duato. Performance Evaluation of Dynamic Reconfiguration in High-Speed Local Area Networks. In *Proc. of the Sixth International Symposium on High-Performance Computer Architecture*, Toulouse, France, Jan. 2000.

[36] O. Lysne and J. Duato. Fast Dynamic Reconfiguration in Irregular Networks. In *Proc. of International Conference on Parallel Processing*, Toronto, Canada, Aug. 2000.

[37] D. Avresky, N. Natchev, and V. Shurbanov. Dynamic Reconfiguration in High-Speed Computer Clusters. In *Proc. of the 3rd IEEE International Conference on Cluster Computing*, Newport Beach, CA, USA, Oct. 2001.

[38] J.P. Hayes. A Graph Model for Fault-tolerant Computing Systems. *IEEE Transactions on Computers*, C-25(9):875–884, 1976.

[39] A. Fei, J. Cui, M. Gerla, and D. Cavendish. A "Dual-Tree" Scheme for Fault-Tolerant Multicast. In *Proc. of International Conference on Communications*, Helsinki, Finland, Jun. 2001.

[40] A. Young, J. Chen, Z. Ma, A. Krishnamurthy, L. Peterson, and R. Y. Wang. Overlay Mesh Construction Using Interleaved Spanning Trees. In *Proc. of IEEE INFOCOM '04*, Hong Kong, China, Mar. 2004.

[41] Z. Xu, C. Tang, S. Banerjee, and S. Lee. RITA: Receiver Initiated Just-in-Time Tree Adaptation for Rich Media Distribution. In *Proc. of the Int'l Workshop on Network and Operating Systems Support for Digital Audio and Video'03*, Monterey, CA, USA, Jun. 2003.

[42] A. Keromytis, V. Misra, and D. Rubenstein. SOS: Secure Overlay Services. In *Proc. of ACM SIGCOMM '02*, Pittsburgh, PA, USA, Aug. 2002.

[43] W. Wang, Y. Xiong, Q. Zhang, and S. Jamin. Ripple-Stream: Safeguarding P2P Streaming Against DoS Attacks. In *Proc. of 2006 International Conference on Multimedia and Expo*, Toronto, Canada, Jul. 2006.