# Timely, Reliable, and Cost-effective Internet Transport Service using Dissemination Graphs

Amy Babay, Emily Wagner, Michael Dinitz, Yair Amir
Johns Hopkins University — {babay, ewagne14, mdinitz, yairamir}@cs.jhu.edu
LTN Global Communications — {ewagner, yairamir}@ltnglobal.com

***Abstract*—Emerging applications such as remote manipulation and remote robotic surgery require communication that is both timely and reliable, but the Internet natively supports only communication that is either completely reliable with no timeliness guarantees (e.g. TCP) or timely with best-effort reliability (e.g. UDP). We present an overlay transport service that can provide highly reliable communication while meeting stringent timeliness guarantees (e.g. 130ms round-trip latency across the US) over the Internet. To enable routing schemes that can support the necessary timeliness and reliability, we introduce *dissemination graphs*, providing a unified framework for specifying routing schemes ranging from a single path, to multiple disjoint paths, to arbitrary graphs. We conduct an extensive analysis of real-world network data, finding that a routing approach using two disjoint paths performs well in most cases, and that cases where two disjoint paths do not perform well typically involve problems around a source or destination. Based on this analysis, we develop a timely dissemination-graph-based routing method that can add targeted redundancy in problematic areas of the network. This approach can cover over 99% of the performance gap between a traditional single-path approach and an optimal (but prohibitively expensive) scheme, while two dynamic disjoint paths cover about 70% of this gap, and two static disjoint paths cover about 45%. This performance improvement is obtained at a cost increase of about 2% over two disjoint paths.**

## I. Introduction

The Internet natively supports end-to-end reliable communication (e.g. using TCP) or best-effort timely communication (e.g. using UDP). However, emerging applications such as remote manipulation and remote robotic surgery bring severe constraints on timeliness, while still requiring high reliability. Human perception requires feedback to be received within about 130ms to be perceived as natural. This 130ms includes both the time for the command to reach the destination and for the feedback to be returned, translating to a latency requirement of 65ms each way. Supporting such applications on a continent-wide scale is demanding: the network propagation delay across North America, for example, is about 35-40ms. Our work develops an overlay transport service to support these applications over the Internet.

In recent years, overlay network architectures have been developed to support applications that require both timeliness and reliability, such as VoIP [1] and live television [2]. A live TV service, supporting interviews from remote studios, requires a one-way latency bound of about 200ms with 99.999% of packets delivered on time. A global overlay network with 10-20 well-situated overlay nodes can support such a service by using the 160-165ms available after accounting for a 35-40ms propagation delay to allow some buffering and hop-by-hop recovery on overlay links. We are involved with a commercial service provider (LTN Global Communications) that uses this approach to support the TV and media industries [3].

In contrast, for the demanding applications we target, there is almost no flexibility to allow for recovery or buffering. Moreover, while techniques such as redundant sending along a single path or network coding can improve reliability, the combination of bursty loss on the Internet and the strict timeliness constraints of the target applications reduces their effectiveness. Thus, a different approach is needed.

For applications with such strict timeliness and reliability requirements, flooding on the overlay topology can provide an optimally reliable solution. In this approach, each packet is sent on all possible paths, so it has the highest possible probability of reaching its destination on time. However, overlay flooding is very expensive. Since ISPs charge for each packet sent, the cost of a redundant dissemination scheme corresponds to the number of overlay links a packet is sent on. Since flooding requires each packet to be sent on every link, it incurs an extremely high cost.

A less expensive approach is to send on multiple disjoint paths. For example, sending on two disjoint paths costs slightly more than twice the cost of the single best path and allows a packet to reach its destination as long as it is successfully transmitted along one of the two paths. Most existing systems that send data redundantly over more than a single path to improve reliability use disjoint paths (e.g. [4], [5]).

Disjoint paths offer a coarse-grained tradeoff between cost and reliability, as adding paths provides higher reliability at a higher cost. However, this approach uniformly invests resources along the paths from a source to a destination. Investing fewer resources in more reliable parts of the network and more resources in less reliable parts of the network can improve the cost-reliability tradeoff. By considering loss and latency characteristics of network links, we aim to provide close to optimal reliability at a reasonable cost.

We present a new approach that transports packets in a timely, reliable, and cost-effective manner by constructing *dissemination graphs* based on network characteristics, application latency and reliability requirements, and cost. A dissemination graph is a connected subgraph of the overlay network topology that connects a source and destination. In our approach, each packet from the source to the destination is sent over all the links in the dissemination graph.

1063-6927/17 $31.00 © 2017 IEEE
DOI 10.1109/ICDCS.2017.63

Ideally, we would calculate the cheapest dissemination graph that meets the application's reliability and latency constraints. However, the problem of finding such a dissemination graph is NP-hard. While we can make computing optimal dissemination graphs tractable for certain topologies, the calculation is too slow to effectively adapt to changing network conditions.

Therefore, our approach is to analyze real-world network data, examine the types of problems that occur in the field, and develop methods to construct and deploy dissemination graphs that can provide the necessary reliability and timeliness during such problems. A key finding of this analysis is that a routing approach using two disjoint paths performs well in most cases, and that cases where two disjoint paths do not perform well typically involve problems around a source or destination. The grounding in real-world data and focus on applications with extremely strict timeliness and reliability requirements separates our approach from the few previous works that have considered redundant dissemination schemes beyond disjoint paths to improve performance in overlay routing (e.g. [6]).

Based on the types of problems we observe in the collected data, we develop a timely dissemination-graph-based routing method that precomputes a limited number of dissemination graphs that address the most common types of problems we observed and switches between them as network conditions change. Specifically, the approach uses two disjoint paths under normal conditions, and switches to use dissemination graphs that add targeted redundancy around a source or destination when problems are detected in that region. We show that this approach can cover over 99% of the performance gap between a traditional single-path approach and an optimal (but prohibitively expensive) scheme, compared with about 70% for two dynamic disjoint paths or about 45% for two static disjoint paths. This performance improvement is obtained at a cost increase of about 2% over two disjoint paths.

The primary contributions of this work are:

1) The invention of dissemination graphs, providing a unified framework for specifying routing schemes ranging from a single path, to multiple disjoint paths, to arbitrary graphs.
2) An extensive analysis of real-world network data, finding that a routing approach using two disjoint paths performs well in most cases, and that cases where two disjoint paths do not perform well typically involve problems around a source or destination.
3) A dissemination-graph-based routing approach that employs targeted redundancy based on current network conditions to provide close to the optimal reliability possible under strict timeliness constraints at a reasonable cost.

We have implemented dissemination-graph-based routing in the Spines open-source overlay messaging framework [7] to create a complete dissemination-graph-based transport service over the Internet.

## II. Related Work

### A. Overlay Routing and Recovery

Our dissemination-graph-based transport service builds on existing work on overlay networks. Many previous works have observed inefficiencies in Internet routing and employed overlays to make use of alternative paths with better performance characteristics. For example, the Detour framework uses an overlay to experiment with alternative routing protocols based on performance metrics [8], RON recovers from problems on a direct Internet path by sending packets through an intermediate node (selected based on loss or latency measurements) [9], and the analysis of one-hop source routing in [10] shows that many Internet path failures can be overcome using the simple approach of sending through a single randomly selected intermediate node when a problem is detected. However, these approaches were not designed to meet strict latency deadlines. In contrast, the work in [1] presents an overlay routing protocol specifically designed to support real-time communication for VoIP. That work introduces an *effective latency* metric that considers both the loss and latency characteristics of overlay links, with the goal of selecting a path with the highest probability of delivering packets within a specific timeliness requirement.

In addition to bypassing problems on a given Internet path via overlay routing, overlays have also been used to improve reliability and latency by enabling fast recovery of lost messages over relatively short overlay hops. Fully reliable hop-by-hop protocols (e.g. [11]) can improve latency compared with end-to-end protocols but cannot support timeliness guarantees. OverQoS [12] combines at most one recovery attempt per lost packet with forward error correction (FEC) to provide a statistical bound on the loss rate experienced by an overlay link. We use a family of recovery protocols specifically designed to support applications with strict latency requirements [13], [1], [2], as described in Section III-C. Because of our applications' high reliability requirements and low tolerance for interruptions, rerouting on a single path after problems are detected is not sufficient, even when combined with recovery protocols or FEC, as the applications' strict timeliness requirements reduce the number of successful recoveries that can be performed and the effectiveness of FEC.

### B. Multipath Routing and Redundant Dissemination

Existing work has shown the benefits of redundant dissemination over multiple edge- or node-disjoint paths in the context of overlay networks (e.g. [14], [15], [16]) and wireless networks (e.g. [17], [4]). Redundant dissemination is used to improve performance (e.g. [14], [15], [17], [4]), as well as to improve security or resilience to attacks (e.g. [4], [16]). In Section VII, we show that in the context of performance, disjoint paths provide a substantial improvement in reliability compared to a single path, but more sophisticated dissemination graphs can provide considerably better performance for a similar cost.

While most existing works using redundant dissemination consider only disjoint paths, [6] proposes routing over non-disjoint paths in order to satisfy application reliability constraints in the presence of geographically correlated failures, while minimizing cost and considering latency constraints. While our goals are similar, the extremely demanding latency and reliability requirements of our target applications require a different approach that can react quickly to changing network conditions. The path-set computation of [6] employs several heuristics to reduce running time in practice, but still computes an optimal path-set, which can be highly computationally intensive when many paths need to be considered. Because we aim to provide close to optimal reliability, it is likely that many paths will need to be considered, making the computation too expensive for timely reactions. Overlay flooding is used for extreme resilience in [16], but this approach is only cost-effective for a small subset of critical traffic and is too expensive for our applications.

Other work combines the use of multiple paths with forward error correction (FEC) or multiple description coding (MDC). For example, [18] sends video encoded via multiple state encoding over multiple paths, and PDF [5] uses FEC while sending packets over both the direct Internet path and a maximally link-disjoint backup path. SplitStream [19] distributes content over multiple multicast trees that do not share interior nodes to improve load balancing as well as resilience, and suggests combining this approach with MDC or FEC to further improve reliability. While such schemes could be used with dissemination graphs, we choose to use fully redundant dissemination to avoid the need for application-specific encoding (as in MDC), and to avoid introducing additional latency for redundant encoded packets (as in FEC), as this may be significant given our strict timeliness constraints.

### C. *Theory of Reliable Network Design*

Without considering latency constraints or recovery, the problem of calculating the reliability of communication between a source and destination over a given dissemination graph can be formulated as the classical two-terminal reliability problem. This problem and the related all-terminal network reliability problem have been extensively studied and shown to be #P-hard (e.g. [20], [21], [22], [23], [24]). Moreover, only a few works consider any form of latency constraints (e.g. [25], which considers a hop-count constraint), and none of the theoretical models we are aware of incorporate recovery protocols, which can have a significant impact on reliability.

Because of the hardness of calculating reliability, prior work on designing reliable networks or graphs has used heuristics or other approximate approaches (e.g. [26], [27], [28], [29]). These approaches generally cannot provide guarantees on how far the solution may be from the optimal, and because they do not consider latency constraints or recovery, they are not directly applicable to our problem. We take a different approach of examining real network data to design a practical solution that can address most common problems.

## III. A Practical Overlay Framework for Timely, Reliable Transport

Our overall approach combines a resilient overlay architecture, dissemination-graph-based routing, and hop-by-hop recovery to support applications with extremely demanding timeliness and reliability requirements.

We use an overlay network to implement routing strategies based on dissemination graphs. Our experience shows that a relatively small number of nodes (i.e. tens) are sufficient to provide excellent global coverage, making it feasible for each node to maintain global state regarding the status of all the overlay links in the system.

We envision this approach being used by a service provider that is able to make available sufficient access bandwidth to meet application demands, so limiting the amount of data sent on each link to minimize congestion is not a concern. However, in order to provide a practical Internet transport service, the approach must be cost effective. Therefore, a key goal is to achieve the required reliability while maintaining a reasonable overall cost. ISPs charge for bandwidth based on the total amount of data sent, so the cost of a dissemination scheme corresponds to the number of times it requires each packet to be sent. Since each overlay link a packet traverses requires sending the packet on that link, the cost of using a particular dissemination graph corresponds to the number of overlay links it includes.

### A. *Resilient Overlay Architecture*

A well-constructed overlay network can quickly route around problems on a particular Internet path, support effective redundant dissemination methods, and enable fast packet recovery. Rerouting and redundant dissemination at the overlay level avoid problems on the current Internet path between a source and destination by making use of alternative paths. However, for this to be effective, the overlay topology should be built with sufficient redundancy, so that multiple paths are available at the overlay level, and redundancy in the overlay topology should match physical redundancy in the underlying network, so that a single underlying problem does not affect several overlay links.

These goals are achieved by constructing a resilient overlay topology as described in [30], [16]: overlay nodes are situated in well-provisioned data centers, where ISPs have invested in multiple independent fiber connections; overlay links are selected to follow the underlying network as much as possible (based on available ISP backbone maps); and overlay links are kept short to increase routing predictability. In this way, we can have high confidence that overlay links do not overlap in the underlying network and that disjoint paths in the overlay topology are largely disjoint in the underlying network as well.

In addition to improving routing predictability, short overlay links also enable fast packet recovery. Recovering a packet on an overlay link requires at least two propagation delays across the link (one to request the lost packet and one to retransmit it), so ensuring that the propagation delay across each link is low enables lost packets to be recovered while still meeting strict latency constraints.

### B. *Dissemination-Graph-Based Routing*

Our dissemination-graph-based routing approaches send each packet over a connected subgraph of the overlay topology that connects the packet's source and destination (i.e. a dissemination graph). These approaches use source-based routing to specify the dissemination graphs: the complete set of overlay links on which each packet should be sent is determined by its source and stamped on the packet at the time it is sent. A bitmask with one bit per link in the overlay topology compactly represents the graph. While the bitmask must be included in the header of each packet, this small overhead is not a limiting factor for the approach: we currently use exactly 64 bits (one word) to represent all the overlay links in a globe-spanning topology and expect 128 bits (two words) to be sufficient to represent most overlay topologies of the size needed to support our target applications (although any size bitmask is supported).

This flexible approach provides a unified routing framework that makes it simple to specify arbitrary graphs. Each source specifies exactly which links each of its packets should be sent over, and intermediate nodes simply forward each packet on all of their outgoing links that are included in the dissemination graph for that packet.[1] The specified links may form a single path, multiple node-disjoint paths, or more complex dissemination graphs, but intermediate nodes do not need any additional logic to handle these different types of graphs.

Moreover, source-based routing eliminates the possibility of packets being dropped due to inconsistent network views across the overlay nodes as routing re-stabilizes after a change is detected: each node honors the bitmask stamped on the packet, so it follows exactly the path decided at the time it is sent, even if network conditions change while it is in flight. However, because the decision is made at the source, this approach can increase the time required to respond to certain network problems when a single path is used, as information about the problem must propagate to the source before routing can be updated. For disjoint paths or more complex dissemination graphs, it is not clear to us how to implement an effective non-source-based routing scheme.

### C. *Hop-by-Hop Recovery*

We augment dissemination-graph-based routing with hop-by-hop recovery to improve performance. While our target applications' strict timeliness requirements limit the number of recoveries that can be performed successfully for a given packet, at least one recovery on one overlay link is generally possible. As described in Section III-A, our overlay links are designed to be short, typically with a propagation delay of about 10ms across the link. This makes it feasible to use, for example, 20-25ms to recover a lost packet on an overlay link, while meeting a 65ms delivery deadline on the scale of a continent with 35-40ms end-to-end propagation delay.

[1]Note that duplicates are suppressed: a node only forwards the first copy it receives of a given packet.

We consider a family of recovery protocols based on the real-time recovery protocol of [1] and a later generalization [2]. These protocols are designed to operate within timeliness constraints and therefore are not 100% reliable: intermediate nodes can discard packets once their delivery deadline has passed, since recovery will not be useful after that point. The basic real-time recovery protocol allows a given packet to be requested and retransmitted at most once per overlay link.

## IV. Foundational Approaches to Dissemination Graph Construction

Building on the novel idea of dissemination-graph-based routing and our novel overlay framework for deploying such routing schemes over the Internet, we investigate several foundational approaches to constructing dissemination graphs. These approaches range from a single path, to disjoint paths, to arbitrary graphs, and can all be specified using our framework. Together, these approaches present a range of trade-offs between reliability, cost, simplicity, and feasibility.

### A. *Dynamic Single Path*

When a dynamic single path is used, each packet is sent on the shortest path from its source to its destination, as determined by its source at the time the packet is sent. We consider a link-weight metric based on the *expected latency* metric of [1], which takes into account both the loss rate and the latency on each overlay link, with the goal of selecting the path that is most likely to reach the destination within the time constraint. Specifically, we calculate the expected latency of a link as:

$$(1-p)\cdot T+(p-2p^2+p^3)\cdot(3T+\Delta)+(2p^2-p^3)\cdot T_{max} \quad (1)$$

Here, $p$ is the current loss rate of the link, $T$ is the current latency of the link, $3T+\Delta$ is the time to recover a lost packet (a constant $\Delta$ to detect the loss, plus three propagation delays for the original send, request, and retransmission), and $T_{max}$ is the maximum allowed latency, used as a penalty for packets that are lost and cannot be successfully recovered by the deadline.

This is the cheapest approach considered: the cost is just the number of overlay links in the single best path (note that the best path in terms of expected latency may not have the fewest number of overlay links, and therefore may not be the path with the lowest cost). If there is any problem on the selected path that cannot be masked by the recovery protocol being used, losses will be visible to the application at least until a new path is selected. While subsecond rerouting is possible, the time needed to react to problems can still result in interruptions of 100-200ms, which are not acceptable for the most demanding applications.

### B. Static Two Node-Disjoint Paths

To avoid disruptions due to problems that occur on a single path, multiple paths may be used simultaneously. When two static node-disjoint paths are used, each packet is sent over a dissemination graph consisting of two node-disjoint paths, where the paths are chosen based on their normal-case latencies.[2] The paths are not recomputed when loss rates or link latencies change, making this approach very simple to implement, as no link monitoring or rerouting is required. This approach masks the failure of any one path, at approximately twice the cost of using the single best path. However, because the paths are static, if both of the selected paths experience a problem, the application will be affected until the problem resolves.

### C. Dynamic Two Node-Disjoint Paths

When two dynamic node-disjoint paths are used, each packet is sent over a dissemination graph consisting of two node-disjoint paths chosen based on their expected latencies (considering both loss and latency, according to Equation 1), as calculated at the packet's source at the time it is sent.

Like static two node-disjoint paths, this approach costs about twice as much as using the single best path, but it fixes the potential problem of continuing to use two failed or problematic paths when alternative good paths are available. The application will only experience a problem if both paths are affected before rerouting occurs, or if no unaffected path exists.

### D. Overlay Flooding

When overlay flooding is used, each packet is sent over every link in the overlay topology. Overlay flooding is extremely expensive, but provides optimal timeliness and reliability: if there is any possible path that can transport a packet from its source to its destination within its deadline, it will be delivered on time.

### E. Time-Constrained Flooding

Time-constrained flooding is a novel approach that preserves the optimality of flooding at a lower cost. In time-constrained flooding, a packet is sent on every overlay link that can improve the probability that it reaches its destination on time, providing optimal reliability. Time-constrained flooding improves on the cost of overlay flooding by not sending packets to nodes from which they cannot reach their destination within the time allowed.

The time-constrained flooding dissemination graph between a source and destination for a given latency constraint is constructed as follows:

1) Run Dijkstra's algorithm from the source to mark each node with its distance (in terms of network latency) from the source.
2) Reverse all edges and run Dijkstra's algorithm from the destination to mark each node with its distance from the destination.
3) Iterate over each edge in the graph: let $d_s$ be the distance from the source to the head vertex of the edge, $d_t$ be the distance from the tail vertex of the edge to the destination, and $l$ be the latency of the edge. If $d_s + d_t + l \leq$ latency constraint, include the edge.
4) Remove unnecessary loops by checking whether each included node is on at least one path from the source to the destination that does not include any cycles and removing any nodes (and their adjacent edges) that are not on such a path.

The time-constrained flooding dissemination graph for a given source and destination can be computed once based on the normal-case latencies of the overlay links and does not need to be recomputed as network conditions change. While time-constrained flooding is optimal in terms of reliability, it does not consider the cost of the dissemination graph beyond removing edges that do not improve reliability and therefore is still likely to be too expensive for practical use.

### F. Optimal Dissemination Graphs

The ideal approach would be to determine the cheapest dissemination graph that meets the application's reliability and latency constraints (or alternatively, the most reliable dissemination graph that can be constructed within a given budget). However, without considering latency constraints (which only make the problem harder), calculating the reliability of a graph is exactly the two-terminal reliability problem [21], which has been shown to be #P-hard [20]. Based on the hardness of two-terminal reliability, we can show that calculating optimal dissemination graphs is also at least NP-hard (formal problem definitions and hardness proofs appear in the Appendix).

While we can find optimal dissemination graphs using exhaustive search, such computations are too slow to permit fast reactions to network problems, taking on the order of tens of seconds to complete for overlay topologies of the size we consider.

## V. Data Collection and Analysis Tool

Each foundational approach discussed above presents a different set of trade-offs between reliability, cost, simplicity, and feasibility. To determine which factors are most important in practice, we collect and analyze real network data to learn about the types of network problems that occur in the field and how each approach performs during such problems.

### A. Data Collection Environment and Method

We collected data over a period of several months on a global overlay network that we have access to through LTN Global Communications. We use an overlay topology based on the one used by LTN, including overlay nodes in twelve data centers spanning East Asia, North America, and Europe, as shown in Figure 1.

[2]Ideally, we would like to minimize the latency of the longer of the two paths, but there is not a known efficient method to compute such paths. Instead, we minimize the sum of the latencies of the two paths, which can be done efficiently, for example using Suurballe's algorithm [31].

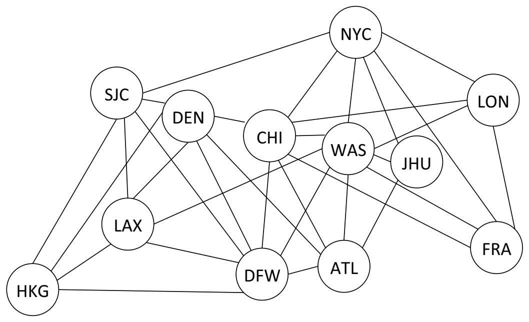

Fig. 1. Global overlay topology spanning East Asia, North America, and Europe. Each circle represents an overlay node located in a data center.

To determine the network's loss and latency characteristics, we collected fine-grained data by sending a message on each overlay link in the topology every 10ms. The granularity may be improved (e.g. to every 1ms) with an increased bandwidth allowance and improved logging infrastructure: the ability to store logs as we recorded data over a long period of time and processes' ability to keep up with logging data from multiple neighbors were limiting factors in our data collection.

Each node logged every message it received, including sequence numbers and timestamps that allow us to calculate loss and latency. Loss rates and round-trip latencies can be calculated directly. To determine approximate one-way latencies, we assume that during periods with no network problems, the latency is symmetric between each pair of nodes (i.e. each one-way latency is equal to half the round-trip time). We then use these one-way latencies determined during stable periods to calculate clock skew and appropriately adjust the one-way latencies during problematic periods. This approach is more accurate than simply using half the round-trip latency, as we find that network problems that result in increased latency on an overlay link often occur in only one direction.

## B. Flow Modeling

To analyze the performance that different overlay routing approaches would have achieved during our data collection periods, we developed the Playback Network Simulator, an overlay simulation tool that uses the per-link data we collected to model the performance of a flow through the network from a source to a destination using any dissemination-graph-based overlay routing protocol and any recovery protocol in the family we consider (i.e. recoveries in the style of [1], [2]). A complete description of the Playback Network Simulator is available in [32].

For a given time period, source-destination pair, sending rate, overlay routing protocol, and recovery protocol, the Playback Network Simulator simulates the end-to-end loss rate and latency based on the network conditions at that time. For each simulated packet in the flow, it calculates whether it would have reached the destination using the given protocols, and if so, what its latency would have been. The simulated packet is propagated across the network according to its dissemination graph. For each link the packet traverses, the simulator calculates the latency and loss rate of that link by averaging over the collected data for that link in a sliding *modeling window* centered at the time that packet reaches the link. Based on the loss rate, the simulator randomizes to determine whether the packet is successfully transmitted. If the first attempt to transmit the packet across a link is unsuccessful, the simulator performs further randomizations to determine when that loss is detected, whether a request for retransmission is successful, and whether a retransmission of the packet is received, based on the specific recovery protocol used. If the packet is successfully transmitted, the latency to traverse the link is calculated as the average one-way link latency at that time plus any time needed to perform recovery.

In modeling dynamic reroutes, we assume that a routing update is issued as soon as the average latency or loss rate measured over a sliding *problem detection window* crosses a certain latency-increase threshold or loss threshold. The size of the problem detection window affects how quickly we can respond to changes in the network: short problem detection windows allow for fast rerouting when problems occur, but may cause instability by rerouting in response to small changes in the network (our experience shows windows on the order of a few hundred milliseconds to be practical). Once an update is generated, we assume that it takes 65ms to propagate to the source. This is a conservative estimate, since the maximum latency between two nodes in the North American portion of the overlay is about 50ms, and in many cases the delay will be considerably shorter.

Note that all modeling parameters, including the modeling window, the problem detection window, and the delay for updates to propagate to the source, can be changed, and the same data can be reanalyzed with different parameters. The only parameter that cannot be changed after data collection is the collection interval itself (which was 10ms for our data). In our evaluation, we use a modeling window of 100ms (which corresponds to 10 packets in our data) and a problem detection window of 200ms, with a 5% loss threshold and 25% latency-increase threshold.

## C. Network Fault Pattern Analysis

We evaluated several initial dissemination-graph construction approaches to determine how they would perform on our collected data and which types of problems they successfully address. The approaches we considered were: dynamic single path, static two paths, dynamic two paths, and time-constrained flooding. Each was evaluated using no recovery protocol and using the real-time recovery protocol of [1]. The performance of each approach was evaluated for sixteen flows across North America. These flows include all transcontinental source-destination combinations of four cities on the East coast of the US (NYC, JHU, WAS, ATL) and two cities on the West coast of the US (SJC, LAX). A full analysis of the results appears in Section VII; here we only provide the intuition leading to our new method.

Overall, we find that two dynamic node-disjoint paths perform quite well, covering close to 70% of the performance

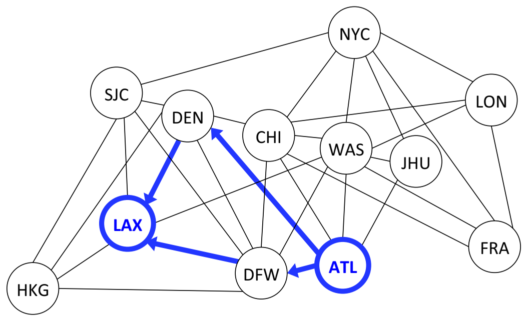

Fig. 2. Static two node-disjoint paths from Atlanta, Georgia to Los Angeles, California (4 edges).

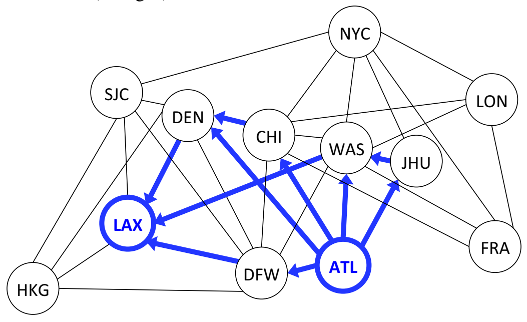

Fig. 3. Source-problem graph from Atlanta, Georgia to Los Angeles, California (10 edges).

gap between a single-path approach and the optimal reliability of time-constrained flooding. Examining the data, we observed that most instances in which two node-disjoint paths did not achieve 100% reliability for a particular flow involved problems on links connected to the source or destination of that flow. We classified each interval in which two node-disjoint paths experienced problems and found that just over 3% of problems involved packets that were dropped or late due to problems on links *not* connected to the source or destination. Therefore, to close the performance gap between two disjoint paths and the optimal time-constrained flooding, we focus on problems involving the source or destination of a particular packet flow, as we find that such problems account for the vast majority of that gap.

## VI. Dissemination-Graph-Based Transport Service using Targeted Redundancy

Based on the analysis described above, we design a new approach with the goal of achieving reliability close to that of time-constrained flooding (which is optimal), at a cost similar to that of two disjoint paths. Our approach is to use a dissemination graph consisting of two disjoint paths for each source-destination flow, except when a network problem is detected at the source or destination of the flow.

Fast reactions to network problems require both quick detection of problems and fast selection of graphs that can address those problems. Because calculating optimal dissemination graphs for arbitrary conditions is computationally intensive, our approach to enabling fast graph selection is to precompute a limited number of dissemination graphs that can

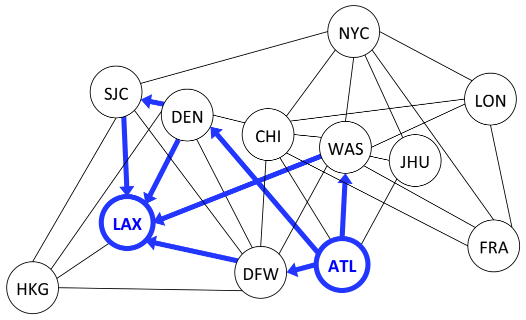

Fig. 4. Destination-problem dissemination graph from Atlanta, Georgia to Los Angeles, California (8 edges).

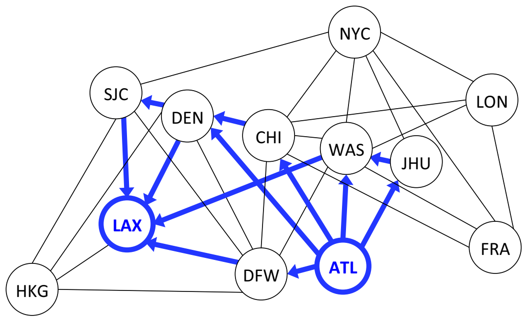

Fig. 5. Robust source-destination-problem dissemination graph from Atlanta, Georgia to Los Angeles, California (12 edges).

address most common problems, changing the problem of computing a dissemination graph to a much simpler problem of classifying network events into a few broad categories. Using this approach, rerouting only requires selecting a new dissemination graph from the precomputed set. Based on the findings that two disjoint paths avoid many common problems and that problems that cannot be avoided using two disjoint paths generally involve a flow's source or destination, each source computes four dissemination graphs for each of its possible destinations:

1) Static two node-disjoint paths dissemination graph
2) Source-problem dissemination graph
3) Destination-problem dissemination graph
4) Robust source-destination-problem dissemination graph

**Two Static Node-Disjoint Paths.** The two static node-disjoint paths are calculated based on normal-case latencies, as described in Section IV-B.

**Source-Problem and Destination-Problem Graphs.** The source-problem and destination-problem graphs aim to maximize the number of ways out of the source or into the destination, respectively. For destination-problem graphs, we consider all overlay nodes directly connected to the destination, eliminating any that cannot be used by the source to reach the destination within the time constraint. We then find a tree that connects the source to all of these nodes.[3] The complete

[3]If the destination has many direct neighbors, the set of nodes to include can be pruned, for example, by eliminating the nodes on the highest-latency paths, or furthest from the destination (since recoveries are least likely to succeed in that case).

| **Routing Approach** | **Availability (%)** | **Unavailability (seconds per flow per week)** | **Reliability (%)** | **Reliability (packets lost/late per million)** |
|---|---|---|---|---|
| **Time-Constrained Flooding** | 99.995887% | 24.88 | 99.999854% | 1.46 |
| **Targeted Redundancy (via Dissemination Graphs)** | 99.995886% | 24.88 | 99.999848% | 1.52 |
| **Dynamic Two Disjoint Paths** | 99.995866% | 25.00 | 99.998913% | 10.87 |
| **Static Two Disjoint Paths** | 99.995521% | 27.09 | 99.998453% | 15.47 |
| **Redundant Single Path** | 99.995764% | 25.62 | 99.998535% | 14.65 |
| **Single Path** | 99.994206% | 35.04 | 99.997605% | 23.95 |

TABLE I
AGGREGATE AVAILABILITY AND RELIABILITY WITH 65MS LATENCY CONSTRAINT, OVER FOUR WEEKS AND SIXTEEN TRANSCONTINENTAL FLOWS.

destination-problem dissemination graph consists of this tree, plus the edges connecting these nodes to the destination.

There are several possible methods for computing the tree that connects the source to the neighbors of the destination. The simplest approach is to use the shortest-path tree, as it is easy to compute and ensures the lowest possible latencies. The shortest-path tree may be a good practical choice for certain applications or large topologies, but it does not provide cost guarantees.

We use minimal cost Steiner shallow-light trees [33], which provide the lowest cost trees that ensure that the path from the source to each neighbor node is short enough to allow it to reach the destination on time. While this is an NP-hard problem, exact calculations of these graphs are feasible for our topology (and are likely to be feasible for many practical topologies, since they only need to be performed once, offline). Moreover, exact solutions can be found in time exponential only in the number of target neighbor nodes, which is likely to be small in practice [34]. However, if exact calculations are not practical, such trees can also be approximated (e.g. [33], [35]).

The same approaches can be used for source-problem graphs by simply reversing the edges of the graph and treating the source as the destination.

**Robust Source-Destination-Problem Graphs.** The robust source-destination-problem graphs are computed by combining the source-problem and destination-problem graphs and performing an additional check to ensure that the graph includes at least two disjoint paths through the network.

**Quick Problem Detection System.** Fast rerouting is accomplished using a quick detection system in which each overlay node monitors each of its links, flooding an update to all of the other overlay nodes whenever it detects that a new problem has started on one of its links or that an existing problem has resolved. When the number of problematic incoming links for a given node exceeds a threshold, each source will switch to using a destination-problem dissemination graph for that destination. Similarly, if a node detects problems on a threshold number of its outgoing links, it will switch to using source-problem graphs. The source-destination-problem graphs are used when there are problems at both the source and destination. If a source-problem or destination-problem graph is selected for a given flow, but a problem is detected on another link of that dissemination graph (not at the source or destination), the robust source-destination-problem graph will also be used.

This approach is scalable to large numbers of simultaneous packet flows, as it requires only a small monitoring overhead per overlay link. All simultaneous flows between a particular source-destination pair can use the same dissemination graph, so no per-flow monitoring is needed.

## VII. EVALUATION

To assess the performance of our dissemination-graph-based routing approach that adds targeted redundancy during problems involving a flow's source or destination, we use our Playback Network Simulator, as described in Section V-B. We analyze how the targeted redundancy approach would perform over the four weeks of data we collected over several months (from July to October 2016) and compare it to the initial dissemination graph construction approaches we considered in Section V-C. We consider the same sixteen transcontinental flows as in Section V-C, modeling a sending rate of one packet per millisecond for each flow. This rate corresponds well to the applications we target (described in Section VIII).

All results consider a 65ms one-way latency deadline, since we aim to support highly interactive remote manipulation applications. All results discussed in this section use the recovery protocol of [1]; results without recoveries show a similar pattern but with somewhat lower overall reliabilities. For the single-path approach, we additionally consider a *redundant single path* scheme, in which each message is originally transmitted twice, separated by 0.5ms.

### *A. Overall Performance*

Table I presents overall reliability and availability results for each of the dissemination-graph-construction approaches we consider, aggregated over all sixteen flows across the US and over all four weeks of data collection. We say that a flow is *unavailable* if the loss rate on that flow exceeds 50% over a period of at least one second. As seen in Table I, over the course of a week, the average unavailability for a flow using a single-path approach is about 35 seconds, or about 25-27 seconds using any of the other approaches. This translates to an overall availability of about 99.994% for a single path and 99.996% for the other approaches.

For the time that a flow is available, we calculate its *reliability*, or the percentage of packets delivered within their latency deadline. From Table I, we see that both time-constrained flooding and the targeted redundancy approach reach nearly 99.9999%. This translates to about 1.5 packets per million that do not arrive within their deadline using these approaches, compared to 10-24 packets per million that do not arrive

| **Routing Approach** | **Week 1 2016-07-12** | **Week 2 2016-08-08** | **Week 3 2016-09-01** | **Week 4 2016-10-13** | **Overall** | **Scaled Cost** |
|---|---|---|---|---|---|---|
| **Time-Constrained Flooding** | 100.00% | 100.00% | 100.00% | 100.00% | 100.00% | 15.75 |
| **Targeted Redundancy (via Dissemination Graphs)** | 99.05% | 99.73% | 98.53% | 99.94% | 99.81% | 2.098 |
| **Dynamic Two Disjoint Paths** | 75.63% | 67.73% | 94.75% | 69.69% | 69.65% | 2.059 |
| **Static Two Disjoint Paths** | 37.89% | 43.18% | -175.13% | 51.63% | 44.58% | 2.059 |
| **Redundant Single Path** | 67.06% | 47.72% | 43.12% | 58.00% | 54.59% | 2.000 |
| **Single Path** | 0.00% | 0.00% | 0.00% | 0.00% | 0.00% | 1.000 |

TABLE II
PERCENT OF THE BENEFIT OF TIME-CONSTRAINED FLOODING OBTAINED BY EACH APPROACH AND SCALED COST (BASELINE IS SINGLE-PATH).

on time for the other approaches, representing an order of magnitude improvement.

## B. Comparison of Approaches

As time-constrained flooding sends packets over every link that can possibly improve reliability, it provides an upper bound on the performance that any dissemination graph or path can achieve (using the same recovery protocol). Since single-path approaches are commonly deployed today, we use our single-path approach as a baseline. We consider the performance gap between time-constrained flooding and a single path as the scale for measuring the performance of other approaches.

Specifically, for each week, we calculate this performance gap as the difference between the total number of packets delivered by their 65ms deadline using the single-path approach and using time-constrained flooding. Across all sixteen flows, with each flow sending at a rate of one packet per millisecond, time-constrained flooding would deliver 50,849 more packets on time than a single path in Week 1, 425,127 more packets in Week 2, 19,331 more packets in Week 3, and 735,055 more packets in Week 4, for a total of 1,230,362 more packets delivered on-time across all four weeks (out of over 38.7 billion total packets). Table II shows what percent of this performance gap each approach covers, aggregated over all sixteen flows for each of the four weeks we consider.

These results show that our dissemination graph approach with targeted redundancy achieves nearly optimal reliability, covering 99.81% of the gap between time-constrained flooding and single-path routing (for a total of 1,227,979 additional packets delivered on time compared to single-path routing). While two disjoint paths offer a substantial improvement over a single path, they do not reach the same level of reliability, covering about 70% of that gap if dynamic reroutes are used and about 45% if the paths are static.

In addition, the "Scaled Cost" column of Table II shows that our approach is able to provide this performance improvement at a cost increase of about 2% compared with two disjoint paths, with two disjoint paths costing about 3% more than twice the cost of the single best path. The source-problem and destination-problem graphs used by our approach may include about twice as many edges as a graph consisting of two disjoint paths (as shown in Figures 3 and 4), and the more expensive source-destination-problem graphs can include three times as many edges (as in Figure 5). However, the more expensive graphs are used infrequently: in our analysis over four weeks, our targeted redundancy approach uses the static two node-disjoint paths dissemination graph 98.11% of the time, the source-problem graph 0.82% of the time, the destination-problem graph 0.68% of the time, and the source-destination-problem graph 0.39% of the time. Therefore, the approach incurs a small total overhead compared with two disjoint paths. In contrast, time-constrained flooding has an overhead of more than 7 times the cost of two disjoint paths (or over 15 times the cost of a single path), making it too expensive for practical use.

While the overall pattern of results is fairly consistent across Weeks 1, 2, and 4, Week 3 shows a somewhat different pattern. This is because it was an exceptionally reliable week, with even the single-path approach providing over 99.9997% on-time delivery across all flows over the week, and many of the losses during that week occurred during a disconnection that even time-constrained flooding could not avoid. During this period, our approach still outperforms two disjoint paths, but two disjoint paths are sufficient to close nearly 95% of the gap between a single path and the optimal reliability.

The Week 3 results also illustrate the drawback of using two static disjoint paths: such an approach will continue to use the same paths even if they suffer a complete disconnection, which can cause it to perform much worse than even a single-path approach that is able to route around problems. As an example, on September 8, 2016, the New York node's incoming links from both Washington and Johns Hopkins were completely disconnected, which would cause the Los Angeles to New York flow to experience two outages of about 15 seconds each using two static disjoint paths, while other approaches show no serious service interruptions (leading to the -175.13% figure for static two disjoint paths in Week 3 in Table II). This effect contributes to the redundant single-path approach consistently outperforming two static disjoint paths, although we note that our independent loss modeling also provides the best-case simulated performance for redundant sending (as it assumes that the loss probability of the second copy of a packet is independent of the outcome of the first packet).

## C. Case Study

While our dissemination graph approach performs comparably to two disjoint paths during periods when the network is largely loss-free, it can provide a dramatic improvement during problematic periods. Figures 6 and 7 show one such period, occurring on August 15, 2016 (during Week 2). In these figures, blue dots represent packets delivered within the 65ms latency constraint, while red dots represent packets delivered after 65ms or dropped (dropped packets are shown

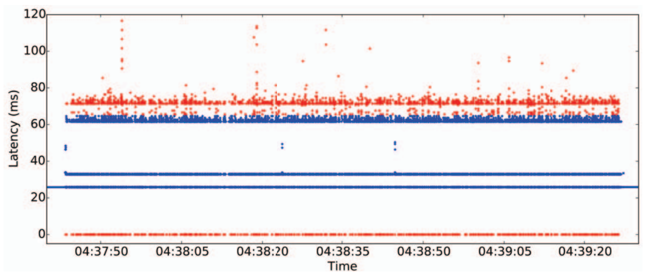

Fig. 6. Packets received and dropped over a 110-second interval from Atlanta to Los Angeles, using two node-disjoint paths (3982 lost or late packets, 20 packets with latency over 120ms not shown).

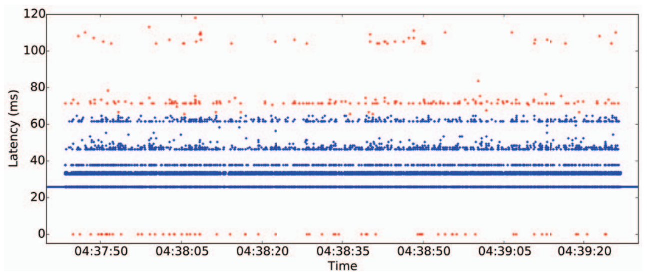

Fig. 7. Packets received and dropped over a 110-second interval from Atlanta to Los Angeles, using our novel dissemination-graph-based approach to add targeted redundancy at the destination (299 lost or late packets).

at 0ms latency). During this 110-second interval, all of the Los Angeles node's incoming links were experiencing 34-72% loss. Using two disjoint paths plus recovery effectively reduces the rate of dropped or late packets between Atlanta and Los Angeles to just under 4%, but our approach provides more than an order of magnitude improvement, reducing the rate of packets not delivered within 65ms to about 0.3%.

This improvement stems from the fact that the destination-problem dissemination graph from Atlanta to Los Angeles provides additional ways into the Los Angeles node, and additional opportunities for lost packets to be recovered. This can be seen in the distinct "stripes" in Figures 6 and 7. Using two paths (Figure 6), one of the paths used at that time generally has a latency of about 26ms (lowest blue stripe), while the other has a latency of about 32ms (second blue stripe). Packets that are initially lost but subsequently recovered on the shorter path generally arrive with a latency of about 60ms (third blue stripe), while packets recovered on the longer path often arrive too late, with latencies around 70ms (highest red stripe). Figure 7 illustrates the impact of the destination-problem graph, including several additional blue stripes that represent the additional paths and recovery opportunities available between the source and destination.

## VIII. Use Cases

Our dissemination-graph-based transport service is designed to support applications with demanding combinations of timeliness and reliability requirements. A major use case for this service is to support remote robotic surgery or other remote manipulation tasks. These applications require high reliability and have extremely stringent timeliness constraints: in order for interaction to feel natural, the round-trip delay between performing an action and receiving a response (e.g. video, haptic feedback) must be less than about 130ms (65ms each way). Position updates for manipulating robots are commonly sent at a frequency of 1000 Hz, matching our evaluation using 1 packet per millisecond well.

Collaborating with robotics researchers, we have demonstrated an initial proof-of-concept for this application, remotely manipulating a robotic arm capable of performing robotic ultrasound. Using the LTN infrastructure and our Spines overlay messaging framework, we have shown that we can manipulate a robot located in a hospital at the Technical University of Munich, Germany from Johns Hopkins University in Baltimore, Maryland over the Internet with a one-way network latency of about 50ms.

Another use case for the service is to support high-value video feeds. Such feeds carry high-quality video (e.g. professional sporting events) to a few sites from which the video can ultimately be distributed to a large number of endpoints. Because any error in the original transmission can be propagated to millions of viewers, these high-value feeds require extremely high reliability (beyond normal broadcast quality). As timeliness constraints for these feeds are less strict than those for remote manipulation, our service can employ both redundant dissemination graphs and multiple recovery attempts for lost packets [2] to achieve even higher reliability.

In both cases, these are high-value applications, where it is reasonable to pay the additional cost of redundant dissemination (about twice the cost of a single path), although the high overhead of an approach like time-constrained flooding would be impractical.

## IX. Conclusion

We have presented *dissemination graphs*, providing a unified framework for specifying routing schemes based on paths, as well as more complex graphs. Based on an extensive analysis of real-world network data, we designed a dissemination-graph-based routing approach that employs targeted redundancy to invest resources in problematic areas of the network. We demonstrated that this approach can cost-effectively cover over 99% of the performance gap between a traditional single-path approach and an optimal but impractical scheme.

## Acknowledgment

We thank Jonathan Stanton for his help in deploying our data collection program on the LTN Global Communications cloud infrastructure. We thank Thomas Tantillo for his help with the Spines overlay messaging framework. We thank Bernhard Fuerst, Ashkan Khakzar, and Nassir Navab for discussions regarding the network needs of remote robotic manipulation. This work was supported in part by NSF grant 1535887 and by DARPA grant N660001-1-2-4014. Its contents are solely the responsibility of the authors and do not represent the official view of the NSF or DARPA.

## References

[1] Y. Amir, C. Danilov, S. Goose, D. Hedqvist, and A. Terzis, "An overlay architecture for high-quality VoIP streams," *IEEE Transactions on Multimedia*, vol. 8, no. 6, pp. 1250–1262, Dec 2006.

[2] Y. Amir, J. Stanton, J. Lane, and J. Schultz, "System and method for recovery of packets in overlay networks," U.S. Patent 8 437 267, May, 2013.

[3] LTN Global Communications, "LTN Global Communications," http://www.ltnglobal.com, retrieved April 7, 2015.

[4] P. Papadimitratos and Z. J. Haas, "Secure message transmission in mobile ad hoc networks," *Ad Hoc Networks*, vol. 1, no. 1, pp. 193 – 209, 2003. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1570870503000180

[5] T. Nguyen and A. Zakhor, "Path diversity with forward error correction (PDF) system for packet switched networks," in *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, March 2003, pp. 663–672 vol.1.

[6] K. Karenos, D. Pendarakis, V. Kalogeraki, H. Yang, and Z. Liu, "Overlay routing under geographically correlated failures in distributed event-based systems," in *On the Move to Meaningful Internet Systems*, 2010, pp. 764–784.

[7] Johns Hopkins Distributed Systems and Networks Lab, "The Spines messaging system," http://www.spines.org, retrieved April 7, 2015.

[8] S. Savage, T. Anderson, A. Aggarwal, D. Becker, N. Cardwell, A. Collins, E. Hoffman, J. Snell, A. Vahdat, G. Voelker, and J. Zahorjan, "Detour: informed internet routing and transport," *IEEE Micro*, vol. 19, no. 1, pp. 50–59, Jan 1999.

[9] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris, "Resilient overlay networks," in *Proceedings of the Eighteenth ACM Symposium on Operating Systems Principles (SOSP)*, 2001, pp. 131–145. [Online]. Available: http://doi.acm.org/10.1145/502034.502048

[10] K. P. Gummadi, H. V. Madhyastha, S. D. Gribble, H. M. Levy, and D. Wetherall, "Improving the reliability of internet paths with one-hop source routing." in *Proceedings of the 6th Usenix Symposium on Operating Systems Design and Implementation (OSDI)*, 2004, pp. 183–198.

[11] Y. Amir and C. Danilov, "Reliable communication in overlay networks," in *Proceedings of the IEEE International Conference on Dependable Systems and Networks*, June 2003, pp. 511–520.

[12] L. Subramanian, I. Stoica, H. Balakrishnan, and R. H. Katz, "OverQoS: An overlay based architecture for enhancing internet QoS," in *Proceedings of the 1st Symposium on Networked Systems Design and Implementation (NSDI)*, 2004, pp. 71–84.

[13] Y. Amir, C. Danilov, S. Goose, D. Hedqvist, and A. Terzis, "1-800-OVERLAYS: Using overlay networks to improve VoIP quality," in *Proceedings of the International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*, 2005, pp. 51–56. [Online]. Available: http://doi.acm.org/10.1145/1065983.1065997

[14] A. C. Snoeren, K. Conley, and D. K. Gifford, "Mesh-based content routing using XML," in *Proceedings of the 18th ACM Symposium on Operating Systems Principles (SOSP)*, 2001, pp. 160–173. [Online]. Available: http://doi.acm.org/10.1145/502034.502050

[15] D. G. Andersen, A. C. Snoeren, and H. Balakrishnan, "Best-path vs. multi-path overlay routing," in *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement (IMC)*, 2003, pp. 91–100. [Online]. Available: http://doi.acm.org/10.1145/948205.948218

[16] D. Obenshain, T. Tantillo, A. Babay, J. Schultz, A. Newell, M. E. Hoque, Y. Amir, and C. Nita-Rotaru, "Practical intrusion-tolerant networks," in *Proceedings of the 36th International Conference on Distributed Computing Systems (ICDCS)*, June 2016, pp. 45–56.

[17] P. Papadimitratos, Z. J. Haas, and E. G. Sirer, "Path set selection in mobile ad hoc networks," in *Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc)*, 2002, pp. 1–11. [Online]. Available: http://doi.acm.org/10.1145/513800.513802

[18] J. G. Apostolopoulos, "Reliable video communication over lossy packet networks using multiple state encoding and path diversity," in *Proc. SPIE, Visual Communications and Image Processing*, vol. 4310, 2001, pp. 392–409. [Online]. Available: http://dx.doi.org/10.1117/12.411817

[19] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: High-bandwidth multicast in cooperative environments," in *Proc. 19th ACM Symposium on Operating Systems Principles (SOSP)*, 2003, pp. 298–313. [Online]. Available: http://doi.acm.org/10.1145/945445.945474

[20] L. Valiant, "The complexity of enumeration and reliability problems," *SIAM Journal on Computing*, vol. 8, no. 3, pp. 410–421, 1979.

[21] C. J. Colbourn, *The Combinatorics of Network Reliability*. New York, NY, USA: Oxford University Press, Inc., 1987.

[22] J. Provan and M. Ball, "The complexity of counting cuts and of computing the probability that a graph is connected," *SIAM Journal on Computing*, vol. 12, no. 4, pp. 777–788, 1983.

[23] M. Jerrum, "On the complexity of evaluating multivariate polynomials," Ph.D. dissertation, University of Edinburgh, 1981.

[24] T. Farley, "Network reliability and resilience," Ph.D. dissertation, Arizona State University, 2009.

[25] H. Cancela, F. Robledo, G. Rubino, and P. Sartor, "Monte carlo estimation of diameter-constrained network reliability conditioned by pathsets and cutsets," *Computer Communications*, vol. 36, no. 6, pp. 611 – 620, 2013, reliable Network-based Services. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0140366412002861

[26] K. Aggarwal, Y. Chopra, and J. Bajwa, "Topological layout of links for optimizing the s-t reliability in a computer communication system," *Microelectronics Reliability*, vol. 22, no. 3, pp. 341 – 345, 1982. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0026271482900063

[27] J. Barrera, H. Cancela, and E. Moreno, "Topological optimization of reliable networks under dependent failures," *Operations Research Letters*, vol. 43, no. 2, pp. 132 – 136, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167637714001771

[28] B. Elshqeirat, S. Soh, M. Lazarescu, and S. Rai, "Dynamic programming for minimal cost topology with two terminal reliability constraint," in *2013 19th Asia-Pacific Conference on Communications (APCC)*, Aug 2013, pp. 740–745.

[29] B. Elshqeirat, "Optimizing reliable network topology design using dynamic programming," Ph.D. dissertation, Curtin University, 2015.

[30] A. Babay, C. Danilov, J. Lane, M. Miskin-Amir, D. Obenshain, J. Schultz, J. Stanton, T. Tantillo, and Y. Amir, "Structured overlay networks for a new generation of internet services," in *Proceedings of the 37th International Conference on Distributed Computing Systems (ICDCS), Vision track*, June 2017.

[31] J. W. Suurballe, "Disjoint paths in a network," *Networks*, vol. 4, no. 2, pp. 125–145, 1974. [Online]. Available: http://dx.doi.org/10.1002/net.3230040204

[32] E. Wagner, "The Playback network simulator: Overlay performance simulations with captured data," Masters Project, Johns Hopkins University, December 2016.

[33] M. V. Marathe, R. Ravi, R. Sundaram, S. Ravi, D. J. Rosenkrantz, and H. B. Hunt, "Bicriteria network design problems," *Journal of Algorithms*, vol. 28, no. 1, pp. 142 – 171, 1998. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0196677498909300

[34] L. Guo, K. Liao, and H. Shen, "On the shallow-light steiner tree problem," in *15th International Conference on Parallel and Distributed Computing, Applications and Technologies*, Dec 2014, pp. 56–60.

[35] M. T. Hajiaghayi, G. Kortsarz, and M. R. Salavatipour, "Approximating buy-at-bulk and shallow-light k-steiner trees," *Algorithmica*, vol. 53, no. 1, pp. 89–103, 2009. [Online]. Available: http://dx.doi.org/10.1007/s00453-007-9013-x

## Appendix

### A. Definitions

Formally, given a graph $G = (V, E)$, an assignment of probabilities $p : E \to [0, 1]$ to each edge, and two distinguished nodes $s, t \in V$, we define the $(s, t)$-reliability $rel(G, s, t)$ to be the probability that $s$ and $t$ are in the same connected component of the graph obtained by retaining each edge $e$ with probability $p(e)$ (independently), and hence removing it with probability $1 - p(e)$. Given a subset $E' \subseteq E$, let $G[E'] = (V, E')$ be the subgraph using only edges in $E'$.

In the MINIMUM COST DISSEMINATION GRAPH problem (Min-DG), we are given as input a graph $G = (V, E)$, an assignment of probabilities $p : E \to [0, 1]$ to each edge, two distinguished nodes $s, t \in V$, and a threshold $r \in [0, 1]$. A subgraph $E' \subseteq E$ is *feasible* if $rel(G[E'], s, t) \geq r$, and otherwise is *infeasible*. Our goal is to return a feasible subgraph while minimizing the number of edges in the subgraph. If there are no feasible solutions, then we must return INFEASIBLE. Note that this does not take into account the latency requirement between $s$ and $t$, just whether they are connected (equivalently, the latency requirement is arbitrarily large).

In the MAXIMUM RELIABILITY DISSEMINATION GRAPH problem (Max-DG), we are given as input a graph $G = (V, E)$, an assignment of probabilities $p : E \rightarrow [0, 1]$ to each edge, two distinguished nodes $s, t \in V$, and a value $c \in \mathbb{N}$. A subgraph $E' \subseteq E$ is *feasible* if $|E'| \leq c$, and otherwise is *infeasible*. Our goal is to return a feasible subgraph while maximizing $rel(G[E'], s, t)$. If there are no feasible solutions, then we must return INFEASIBLE.

As these are optimization questions, we will be concerned not just with solving them, but also with *approximating* them. For Min-DG, we say that an algorithm is an $\alpha$-approximation if on every input, the number of edges in the subgraph returned by the algorithm is at most $\alpha$ times the size of the optimal feasible solution (if there is no feasible solution, then even an approximation algorithm must return INFEASIBLE). The value $\alpha$ is known as the *approximation ratio*.

Given a graph $G = (V, E)$, probabilities $p : E \rightarrow [0, 1]$, and distinguished nodes $s, t \in V$, the 2-TERMINAL RELIABILITY problem is to compute $rel(G, s, t)$. We define the associated decision question 2-TERMINAL RELIABILITY (DECISION) to be the language consisting of tuples $(G = (V, E), p : E \rightarrow [0, 1], s, t, r)$ where $rel(G, s, t) \geq r$.

A classical result of Valiant [20] is that 2-TERMINAL RELIABILITY is #P-Hard. This obviously implies that 2-TERMINAL RELIABILITY (DECISION) is NP-hard.

## B. Hardness Results

We first show that Min-DG is not just NP-hard: it is NP-hard to even approximate.

**Theorem 1.** *Unless* **P** = **NP**, *there is no polynomial-time $\alpha$-approximation for* MINIMUM COST DISSEMINATION GRAPH *for any $\alpha$.*

*Proof.* We prove this by a gap reduction from 2-TERMINAL RELIABILITY (DECISION). Our reduction is trivial: given an instance $(G, p, s, t, r)$ of 2-TERMINAL RELIABILITY (DECISION), we simply reinterpret it as an instance of Min-DG. Let $ALG$ be an $\alpha$-approximation for Min-DG. If $rel(G, s, t) < r$ then by definition $ALG$ must return INFEASIBLE. On the other hand, if $rel(G, s, t) \geq r$, then $ALG$ will return a subgraph $E'$ where $rel(G[E'], s, t) \geq r$ (and where $|E'|$ is at most $\alpha$ times the optimum, but the size bound makes no difference). Hence $ALG$ lets us decide in polynomial time whether $rel(G, s, t) < r$ or $rel(G, s, t) \geq r$, and so since 2-TERMINAL RELIABILITY (DECISION) is NP-hard, $ALG$ can only exist if **P** = **NP**. □

For the Max-DG problem we need a (slightly) less trivial reduction, and can only prove hardness (not hardness of approximation). This is intuitively because it might be the case that $rel(G, s, t)$ can be approximated extremely well even though exactly computing it is #P-hard (this is still an important open question).

**Theorem 2.** *Unless* **P** = **NP**, *there is no polynomial-time algorithm for* MAXIMUM RELIABILITY DISSEMINATION GRAPH.

*Proof.* As before, we prove this by a reduction from 2-TERMINAL RELIABILITY (DECISION). Given an instance $(G, p, s, t, r)$ of 2-TERMINAL RELIABILITY (DECISION), we will create an instance $(\hat{G} = (\hat{V}, \hat{E}), \hat{p}, \hat{s}, \hat{t}, B)$ of Max-DG as follows. Let $P = \{y_1, y_2, y_{|E|+1}\}$ be a set of $|E| + 1$ new nodes, let $\hat{s}$ and $\hat{t}$ be two extra new nodes, and let $\hat{V} = V \cup P \cup \{\hat{s}, \hat{t}\}$. We will define a path from $\hat{s}$ to $\hat{t}$ by using the nodes in $P$, creating an edge set $E_P = \{\{\hat{s}, y_1\}, \{y_{|E|+1}, \hat{t}\}\} \cup \{\{y_i, y_{i+1}\} : i \in [|E|]\}$. Our final edge set $\hat{E} = E \cup E_P \cup \{\{\hat{s}, s\}, \{t, \hat{t}\}\}$. Our probability function is defined as

$$\hat{p}(e) = \begin{cases} p(e) & \text{if } e \in E, \\ r - \epsilon & \text{if } e = \{\hat{s}, y_1\}, \\ 1 & \text{otherwise,} \end{cases}$$

where $\epsilon > 0$ is a small value which we will define later. Finally, let $B = |E| + 2$.

It is easy to see that this reduction can be computed in polynomial time. Before we analyze the reliability, note that the edges in $E_P$ give $rel(\hat{G}[E_P], s, t) = r - \epsilon$, while any set $E' \subseteq \hat{E}$ with $E_P \setminus E' \neq \emptyset$ has $rel(\hat{G}[E'], s, t) = rel(\hat{G}[E' \cap E], s, t)$. In other words, if the entire path $E_P$ is not in a subgraph, then that subgraph has reliability determined just by the original edges from $E$ that are in it. Since the budget is $|E| + 2$, this means that any optimal solution to Max-DG is either the set $E' = E \cup \{\{\hat{s}, s\}, \{\hat{t}, t\}\}$ or is the set $E_P$. Any other edge set is suboptimal.

Suppose that $rel(G, s, t) \geq r$. Then $E'$ has reliability $rel(\hat{G}[E'], \hat{s}, \hat{t}) = rel(G[E], s, t) = rel(G, s, t) \geq r$. On the other hand $E_P$ has reliability strictly less than $r$. Hence if $rel(G, s, t) \geq r$, any algorithm for Max-DG must return exactly the set $E'$.

Now suppose that $rel(G, s, t) < r$. It is easy to see that if we set $\epsilon$ to be small enough (say, less than $(\min_{e \in E} p(e))^{n^3}$), then $rel(G, s, t) < r - \epsilon$. This is because, since the reliability can be expressed as a polynomial in the probability values $\{p(e)\}$, there are only a finite number of different values that it could possibly be. Thus in this case, any algorithm which solves Max-DG must return $E_P$ as a solution.

Thus the existence of an exact algorithm for Max-DG which runs in polynomial time would imply that **P** = **NP**, since it would allow us to solve 2-TERMINAL RELIABILITY (DECISION). □