CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 2015; **27**:3614–3632
Published online 6 August 2014 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cpe.3356

BLOR: An efficient bandwidth and latency sensitive overlay routing approach for flash data dissemination

Xiaoyong Li, Yijie Wang*,†, Yongquan Fu, Xiaoling Li and Weidong Sun

Science and Technology on Parallel and Distributed Processing Laboratory, College of Computer, National University of Defense Technology, Changsha, China

SUMMARY

The problem of flash data dissemination refers to transmitting time-critical data to a large group of distributed receivers in a timely manner, which widely exists in many mission-critical applications and Web services. However, existing approaches for flash data dissemination fail to ensure the timely and efficient transmission, because of the unpredictability of the dissemination process. Overlay routing has been widely used as an efficient routing primitive for providing better end-to-end routing quality by detouring inefficient routing paths in the real networks. To improve the predictability of the flash data dissemination process, we propose a bandwidth and latency sensitive overlay routing approach named BLOR, by optimizing the overlay routing and avoiding inefficient paths in flash data dissemination. BLOR tries to select optimal routing paths in terms of network latency, bandwidth capacity, and available bandwidth in nature, which has never been studied before. Additionally, a location-aware unstructured overlay topology construction algorithm, an unbiased top-k dominance model, and an efficient semi-distributed information management strategy are proposed to assist the routing optimization of BLOR. Extensive experiments have been conducted to verify the effectiveness and efficiency of the proposals with real-world data sets. Copyright © 2014 John Wiley & Sons, Ltd.

Received 24 October 2013; Revised 7 July 2014; Accepted 12 July 2014

KEY WORDS: data dissemination; overlay routing; topology construction; detour routing; path selection; performance evaluation

1. INTRODUCTION

The problem of rapid data dissemination in complex heterogeneous network environments, such as peer-to-peer (P2P) networks [1–6], distributed systems [7], streaming systems [8, 9], wireless networks [10, 11], and the clouds [12, 13], is now recognized as a key element in many real applications and Web services such as the acquisition of timely weather information, emergency responses (e.g., earthquake early warning‡), information battlefields environment, news dissemination, complex distributed queries [14], and traffic-condition broadcasts. In all these scenarios, the real data usually need to be sent to a large number of users as quickly as possible. This specialized form of dissemination is termed as *flash data dissemination* [15], which transmits dynamically created and time-critical data to a group of users in a timely manner. Thus, flash data dissemination often requires high throughput and low latency. As a result, high delays or low throughput significantly reduces the usefulness of flash data dissemination.

As known to us, the overlay network creates a virtual topology on top of the physical network, which has become an effective alternative to IP multicast for efficient point to multi-point commu-

*Correspondence to: Yijie Wang, Science and Technology on Parallel and Distributed Processing Laboratory, College of Computer, National University of Defense Technology, Changsha, China.
†E-mail: wangyijie@nudt.edu.cn
‡Shake-cast, http://www.anss.org.

nication across the Internet or wide-area P2P systems. The overlay-based data dissemination has been frequently studied since the last decade, because of its ease of deployment, self-organization, high scalability, and robustness [16]. Therefore, a number of overlay-based solutions, protocols, and systems have been proposed to optimize the data dissemination [17–23].

A classic solution to optimize the data dissemination with overlays is the *Content Delivery Networks* (CDN) (e.g., Akamai[§] and Coral [17]), which is promoted by the commercial companies. The CDNs replicate content and place it at proximity locations to the clients that have high bandwidth. Although CDNs can provide a scalable and cost-effective mechanism for accelerating data dissemination, they require substantial storage and network resources and primarily focus on the dissemination of large files such as multimedia video files. Furthermore, their performances will be seriously degraded if the number of the nodes grows too fast [24]. Besides CDNs, a series of systems and protocols are proposed to improve data dissemination, such as BitTorrent[¶], Bullet [18], OCEANSTORE [20], PROMISE [21], and Splitstream [22]). In addition, many publish/subscribe systems, including the content-based or topic-based systems, are also widely used to disseminate data in a decoupled manner. The classic systems include Scribe [25], Vitis [26], and PeerChatter [27], all of which use a gossip-based overlay or structured overlay to efficiently maintain all node connections. Unfortunately, all the aforementioned systems are exclusively designed to establish an efficient system for large file sharing or content transmission, by building overlay multicast trees or overlay meshes, but not to address the problem of flash data dissemination.

On the contrary, in this paper, we are only interested in the *time-critical* content delivery and study the overlay routing problem for flash data dissemination, in which the overall dissemination time to all the peers is minimized. That is, given a sender and a large set of interested receivers spread across the whole network, the goal is to minimize the time of delivering small-sized or medium-sized data to all the receivers. To efficiently address this problem, lots of efforts have been conducted to improve data dissemination by optimizing the overlay routing [4, 16, 27–36] in the last decade. Although all these studies have shown the feasibility of improving data dissemination by optimizing the routing paths, all of them are either limited by the scalability or unsuitable for our flash data dissemination.

Generally, to reduce the total time for flash data dissemination, the routing efficiency and the transmission efficiency are both need to be improved. On one hand, the routing efficiency determines the transmission delay in the network and can also improve the data location. On the other hand, there exist many overlapping logical paths, which can easily result in poor transmission or link congestion. If we can detour routing with some optimized intermediate nodes, then the transient failures or congestion on default IP routing paths can be bypassed. To the best of our knowledge, CREW [15] is the most efficient system that is exclusively designed for flash data dissemination and is currently deployed in many real applications. Nevertheless, it does not consider the overlay routing problem in its protocol design. We have experimentally investigated that the performance can be further improved with the overlay routing technique.

Motivated by these considerations, in this paper, we propose an efficient overlay routing approach called *bandwidth and latency sensitive overlay routing (BLOR)* to optimize flash data dissemination, which enables rapid dissemination in large-scale real networks. BLOR first optimizes the overlay routing based on an unstructured overlay, so as to improve the routing efficiency, especially for reducing the dissemination latency. Furthermore, BLOR detours the overlay routing by selecting the predominant candidate nodes for path switching, in order to improve the transmission efficiency. Specifically, BLOR tries to detour routing by optimizing the latency, bandwidth capacity, and available bandwidth altogether in nature, which has never been studied before, to the best of our knowledge. Extensive experimental results with real-world data sets demonstrate that our approach can significantly improve flash data dissemination.

In summary, our contributions are as follows:

[§]http://www.akamai.com.
[¶]http://www.bittorrent.com/.

DOI: 10.1002/cpe

- An algorithm called *Location-aware unstructured overlay topology construction (Lautc)* is proposed, in order to support the efficient overlay routing;
- A bandwidth and latency sensitive overlay routing approach BLOR is proposed to improve flash data dissemination based on the overlay constructed with Lautc;
- An unbiased top-k dominance model is proposed to optimize the candidate nodes selection for path switching;
- A semi-distributed storage structure is proposed to efficiently manage all the collected information and facilitate the implementation of the overlay routing;
- Extensive experiments with real data sets are conducted to demonstrate the performance of the proposals.

This paper extends an earlier published conference paper [37] in several substantial ways. First, we provide more details about the background and the related work. Second, we formally define the top-k dominance model, which makes it more concise and clear. Third, some optimization techniques are integrated into the approach implementation to achieve better performance. Fourth, we propose a practical semi-distributed storage structure to manage the collected information based on the clustering with delays. Finally, we reconduct extensive experiments to evaluate the performance of our proposals.

The remainder of the paper is organized as follows. In the next section, we discuss the related work. Section 3 provides an overview of our proposed approach. Section 4 and Section 5 describe Lautc and BLOR in detail, respectively. This is followed by the experimental evaluation of our proposals from various perspectives in Section 6. Finally, we conclude the paper with a summary of our future work in Section 7.

2. RELATED WORK

In P2P research community, many studies [1–3] have proved that making use of diverse dissemination paths can effectively improve the performance of network applications. Detour routing by path switching, as an effective way to optimize the data dissemination, can avoid the inefficient paths by routing through the detour nodes. To the best of our knowledge, this paper is the first work to improve the overlay routing by considering the end-to-end latency, bandwidth capacity, and available bandwidth altogether for flash data dissemination. Next, we will give a brief overview of the existing overlay routing techniques.

Detour routing assumes that there exist many available routing paths between two peer nodes and adaptively change the routing paths according to the actual application requirements. To overcome the path failure and performance bottlenecks by path switching has received considerable attention recently. In [28], the authors first propose the idea of solving the routing problem by detouring. Inspired by this, the authors in [29] propose to increase the path diversity with alternate paths. Nevertheless, the aforementioned studies only indirectly impact the performance and cannot meet the quality of service (QoS) requirements in real applications. Cha *et al.* [30] propose to place some detour nodes to optimize the dissemination, which is difficult to be achieved in P2P networks.

All the aforementioned approaches try to select the optimized routing paths from some candidate paths. As the growth of the network size, optimizing routing with detour nodes needs a large amount of measurement and computation overhead, thus the scalability of the approaches is strictly limited. Therefore, Gummadi *et al.* [31] use a set of randomly selected nodes as the detours to reduce the overhead. For instance, it randomly picks some potential nodes as the candidate nodes and then chooses the best paths through these nodes to forward data. The deficiency of the approach is that some better detour nodes may be ruled out, because of the random selection of the detour nodes.

Moreover, many other studies use overlay topology information to optimize the path selection. The authors in [33] choose paths according to the failure rate of the overlay links, which needs a large amount of resources to obtain the information. In [1], the authors select the paths according to the complete probing of the end-to-end paths and underlying path loss rate, thus the scalability is obviously limited. Nakao *et al.* [35] make use of trace-route to obtain IP layer routing paths and delay information and, based on this, to assess the overlaps of overlay and underlying physical net-

DOI: 10.1002/cpe

work paths. Furthermore, it selects less number of overlapping paths as the candidate dissemination paths, which is only applicable to the small-scale overlays because of its poor scalability.

All the aforementioned studies do not take bandwidth information into consideration for detour routing. Recently, Zhu *et al.* [38] propose to use available bandwidth for the overlay routing, and each node probes the bandwidth to a large number of other nodes, which limited the scalability. While in this study, each node only needs to probe to a small number of nodes. In [36], the authors make use of the link-state-like protocol to examine the bandwidth, loss rate, and other link attributes of the network paths, but the cost is too high to support large network applications. Moreover, Lee *et al.* [4] propose an efficient approach called bandwidth-aware routing overlay network (BARON) to improve overlay routing. Although BARON can optimize the data dissemination by selecting better routing paths among the global paths according to the bandwidth capacity and available bandwidth, not only it requires a large amount of storage and detection overheads but also increases the network transmission delays, because of the lack of consideration for latency. In summary, the aforementioned algorithms are all unsuitable for our flash data dissemination.

3. OUR DESIGN

3.1. Overview

Although the overlay can improve the data dissemination, simply using the overlay may be not enough to provide reliable QoS, because of the mismatch between the overlay and the underlying physical network topology. That is, it is difficult to provide great transmission performance and scalability, because of the lack of information about underlying physical network. Moreover, the transmission performance of flash data dissemination may be greatly affected by the inefficient paths or the bottleneck paths, which may be caused by the transient failures or the congestion of the links, as well as the overlapping of multiple paths.

Generally, to efficiently process the overlay-based flash data dissemination, we mainly face two challenges in the process of flash data dissemination:

1. *Routing efficiency during the data location*. Data location process may incur high delays due to the mismatch of the overlays and underlays, and the routing efficiency directly determines the transmission delay over the network.
2. *Routing optimizations during the data transmission*. Because there may be a large number of candidate routing paths in the overlay, to maximize the end-to-end bandwidth by selecting the optimal routing nodes becomes a challenging problem.

As we know, the efficiency of data location is greatly determined by the degree of the match between the overlay topology and the underlying physical topology. Thus, if we can construct the overlay and make it greatly reflect the underlying physical network, then the efficiency of data location can be greatly improved. Additionally, many recent works [1–3] show that routing through intermediate overlay nodes instead of using the default IP routes can effectively avoid inefficient data transmission paths, when end-to-end quality of service routing is not provided in the network infrastructure.

Therefore, to solve the first problem, we propose the algorithm Lautc to construct a location-aware unstructured topology. Lautc tries to construct the unstructured overlays that match the underlying physical network to minimize the lookup delays as far as possible. Moreover, to address the second problem, we do the detour routing with BLOR by considering the latency, bandwidth capacity, and available bandwidth altogether to avoid the poor paths. BLOR is based on the overlay that is built with Lautc and improves the overlay routing with a location-aware path selection strategy in nature.

3.2. Location-aware overlay construction

Our goal is to design and implement a novel overlay topology construction method to support overlay routing in real world heterogeneous networks. However, in large-scale P2P network environments, the nodes and the links are often highly dynamic, thus we need to construct an unstructured P2P overlay to adapt to the dynamic network environments.

DOI: 10.1002/cpe

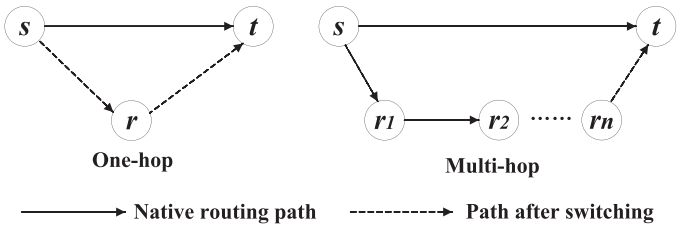


Figure 1. One-hop and multi-hop path switching.

As a representative protocol for constructing unstructured P2P overlay, *Gnutella*[‖] finds data by flooding searching with a limited *time-to-live* (*TTL*). Specifically, each node multicasts the search requests to all its neighbors and reduces the *TTL*, until the desired data is found or the *TTL* value reaches 0. Each node in Gnutella adopts a random neighbor selection strategy, which greatly affects the data location and transmission. If nodes can obtain some knowledge about the physical location of others when choosing neighbors, the routing latency can be significantly reduced in flash data dissemination.

Therefore, we propose to construct a location-aware overlay with Lautc in BLOR, in order to be aware of the underlying physical networks to reduce the delays when looking up the required data. To this end, we choose the nearby neighbors according to the estimated delays obtained with network coordinates computing method for the overlay construction.

Additionally, the shortest paths in the overlay routing may be in the following scenario. A message first passes through some short hops to reach a nearby node, and the node connects to a distant node, while the distant node connects to the node near to the target [39]. Therefore, it is better to choose both nearby and remote nodes as neighbors for each node when we construct the overlay topology

The main idea of Lautc includes three aspects: (1) getting the delays between nodes with network coordinates, (2) sampling the neighbors with a random walk process, and (3) choosing neighbors with an adaptive strategy to build the unstructured overlay topology. Network coordinate approaches [40–46] have been proved to be an efficient way for estimating delays between nodes without complete measurements. Therefore, we try to adopt a certain network coordinate approach in Lautc to estimate the delays between nodes, in order to save the expensive measurement overhead.

3.3. Overlay routing optimization

In the overlay, multiple logical connections between end-to-end nodes may share the same physical network links. Once the shared links become the bottlenecks or interrupted, the logic paths built on these links will be inefficient or interrupted. At this point, they cannot provide connectivity, and the invalid redundant connections consume more network resources.

As for the flash data dissemination, content needs to be sent to the targets as soon as possible. Because the overlapping of paths or the bottleneck links may significantly decrease the performance of data transmission, we do the path switching to avoid the poor performance paths to improve the efficiency. The main idea behind BLOR is as shown in Figure 1, which describes one-hop and multi-hop path switching cases [4].

However, path switching cannot be random, as changing the routing paths will probably increase the routing latency. Additionally, the new routing paths cannot be guaranteed to be better than the default ones. Therefore, in order to maximize the performance of flash data dissemination, not only we need to consider the bandwidth capacity and available bandwidth of the paths, but also, we need to minimize the routing hops to reduce the total transmission delays.

The primary principle of BLOR is to select optimized intermediate nodes for overlay routing to increase the throughput, as well as to reduce the delay as much as possible.

[‖]http://www.gnutellaforums.com/.

DOI: 10.1002/cpe

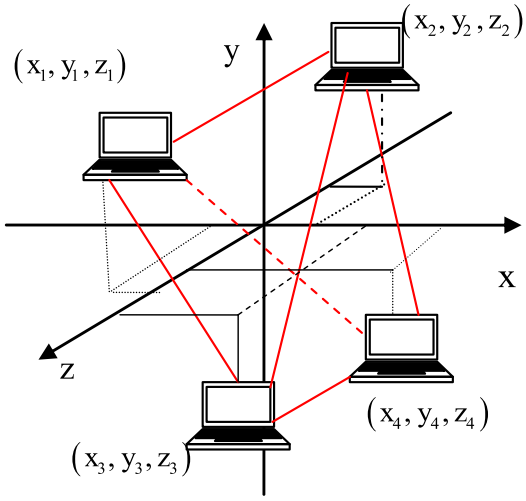


Figure 2. An example of the network coordinate model.

4. OVERLAY CONSTRUCTION WITH LAUTC

The main idea of Lautc is to use the network coordinates of nodes to obtain the delays between the nodes, combined with a random walk sampling process and the corresponding neighbor selection strategy to build the unstructured overlay topology. Network coordinate approaches have been proved to be an efficient way to estimate the minimal delays between hosts without direct measurements. There exist many approaches [40–46] that can be adopted to obtain the network coordinates in Lautc. The basic network coordinate model of the nodes in the Internet or P2P is shown in Figure 2. In the overlay construction with Lautc, we adopt PIC method [44] to obtain the network coordinates, because of its practicability and popularity.

The PIC is proved to be a practical coordinate-based approach for estimating distances in the Internet. Generally, PIC first assigns a point in a multi-dimensional Euclidean space to each node and uses the distance between two points in the space as an estimate of the network distance between the nodes. Then nodes can compute their coordinates in the Euclidean space when they join the system. Thus, given the coordinates for two nodes, any node can predict the distance between them [44]. Note that the distance here means the latency (i.e., round-trip delay or network hops).

4.1. The Lautc algorithm

For each new participating node, Lautc constructs the overlay according to the following three steps: (1) computing its own network coordinate with PIC method [44], (2) obtaining other nodes' information by parallel executing multiple random walks sampling, and (3) selecting nodes as its neighbors and building the relationship.

In Lautc, each node selects the k nearest nodes and randomly picks k' remote nodes as its neighbors. Each node maintains a list of remote nodes and adopts *least recent used* (*LRU*) replacing strategy to limit the size of storage. When a node forwards the searching or publishing messages, the *LRU* node of the list will be replaced by the source node.

The number of neighbors in Lautc is not clearly defined, we try to adaptively choose the number of neighbors for each node according to the capability of the node. Generally, the more powerful nodes have larger probability of interconnection, thus the number of their neighbors increases. On the contrary, the nodes with low capability usually have less number of neighbors.

In summary, as for each new participating node m, Lautc executes the following steps:

1. Node m obtains the landmark nodes from the bootstrap node and calculates its coordinate according to the network coordinate computing method PIC [44];

DOI: 10.1002/cpe

2. Node m obtains a set L of existing nodes in the system from the bootstrap node and randomly selects l nodes from L as the start nodes for sampling;
3. Parallel executing l-way random walks with a certain length by traversing the network from m, then all the traversed nodes return the information including IPs and the network coordinates to m;
4. Node m receives the information and calculates the delays with the coordinates to obtain the set of delays S_{delay};
5. Node m selects k nodes (denoted as S_{near}) with the smallest delays in S_{delay}, then randomly chooses k' nodes from the remaining nodes (denoted as S_{rand}), and adds them to its neighbor list, that is, $m.neighbor\ List = S_{\text{near}} \cup S_{\text{rand}}$;
6. Node m establishes the relationships with the neighbors and stores the related information.

The main motivation of selecting k nearest nodes as neighbors is to reduce the average delay of routing, while selecting k' neighbors is to enhance the system stability and improve the searching. In order to place more emphasis on the routing efficiency of flash data dissemination, we select more close nodes as neighbors, and the rate of k and k' is set to 2:1 in our experiments. The number of neighbors in the overlay that is constructed by Lautc is determined by each node's processing capability. The more powerful nodes would have the more number of the neighbors.

4.2. Other issues in Lautc

To gradually establish the system, two other issues need to be solved in the topology construction with Lautc, including the bootstrap issues and the topology maintaining issue such as processing for the departure and failure of the nodes.

4.2.1. Bootstraps in Lautc. The bootstrap issue is mainly derived from two aspects: the bootstrap of the network coordinates computing process and the bootstrap of the topology construction process. We first select some landmarks and calculate the coordinates of the landmarks. Then, the new participating node calculates its own coordinate according to the landmarks and makes itself as a landmark of the other nodes for the coordinates computation with the PIC method [44]. Then, each node constantly updates its coordinate by exchanging information with other nodes to gradually establish a stable network coordinate system.

In the bootstrapping of topology construction process, the system first establishes some nodes and starts these bootstrapping nodes as the entrances of the new joining nodes. Thus, the new participating node can find the entrance of the system and gradually establish the topology construction process by continuously adding neighbors with the corresponding steps of the Lautc algorithm. Note that the initial entrances should be published to notify the subsequent joining nodes.

4.2.2. Topology maintaining. Because of the dynamic of real P2P networks, the nodes may depart the system or fail at any moment, thus we need to update the topology when this case occurs. In Lautc, each node periodically detects its neighbors by probing to obtain the status of the neighbors. Once a failed probe is found, then it will probe the neighbor again. If it probes n times and all the probes are unsuccessful, then the neighbor will be regarded to be failed or left. Thus, the node will be removed from the neighbor list. Note that n is a threshold value, which is set by the users.

Note that because the nodes in the system may frequently leave or fail, the number of the neighbors for each node may be small. Thus, to improve the connectivity and robustness of the unstructured P2P overlay, we need to adjust the neighbor relationships of these nodes. Therefore, if the number of the neighbors is smaller than l, which is also set by the user or customer, then we will initiate a new request for neighbors. Thus, we can combine the new neighbors found by sampling with the existing neighbors to reestablish the neighbor relationship.

DOI: 10.1002/cpe

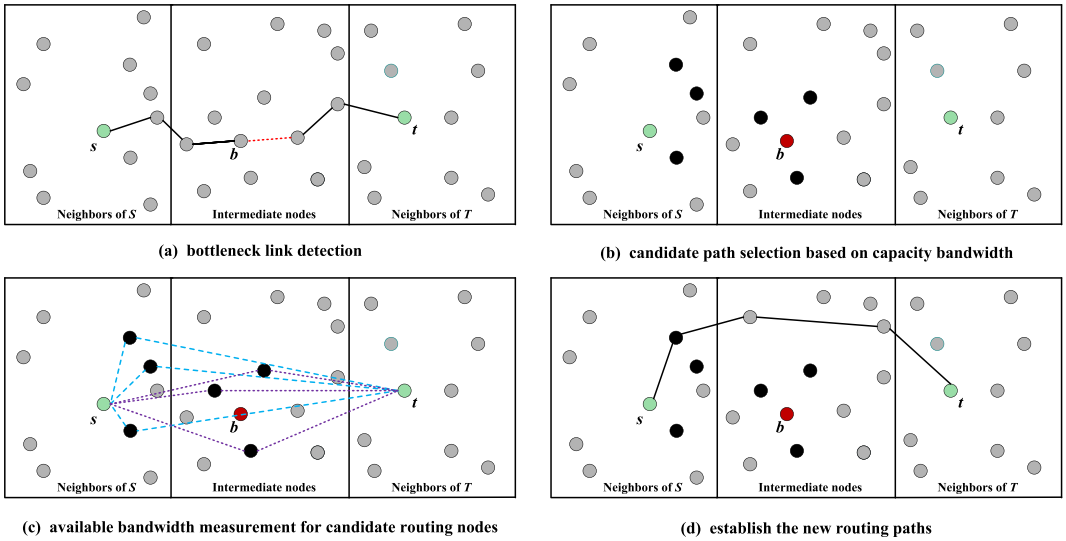


Figure 3. Path switching with BLOR.

5. ROUTING OPTIMIZATION WITH BLOR

5.1. *The framework of BLOR*

The BLOR optimizes the routing over a location-aware unstructured overlay constructed with Lautc, which takes full account of the latency. Thus, the efficiency of data location is much better than the traditional unstructured overlay.

The bandwidth capacity is the maximum amount of data per time unit that the link or path has available, when there is no competing traffic. If a path consists of several links, the link with the minimum transmission rate determines the capacity of the path. Available bandwidth is the maximum amount of data per time unit that a link or path can provide, given the current utilization. Note that utilization is the aggregate capacity currently being consumed on a link or path. Available bandwidth is the amount of usable bandwidth without affecting cross-traffic, and available bandwidth along a path is the minimum available bandwidth of all traversed links. Bandwidth capacity and available bandwidth are both important parameters of the network paths, which have been proved to be the key factors for affecting the data dissemination [4, 36, 38]. Thus, besides delays, we try to select the routing paths according to the bandwidth capacity and available bandwidth, in order to avoid the bottleneck link or the poor performance end-to-end paths.

To summarize, BLOR mainly includes the following five processes:

1. Measuring, collecting, and storing the information of bandwidth capacity between the nodes;
2. Detecting the bottleneck link and locating the bottleneck node b (Figure 3(a));
3. The source node s and the bottleneck node b pick the nodes that meets Eq. (1) from neighbors to obtain a candidate node set S_{cand} (Figure 3(b));
4. Node s probes the available bandwidth between s and each node from S_{cand}, as well as the available bandwidth to target node t (Figure 3(c));
5. Selecting the best detour node according to the bandwidth capacity and available bandwidth for the new routing paths (Figure 3(d)).

In the second step, many available tools [39, 47, 48] can be used to detect the bottleneck links, and we use pathneck [39] method to detect the bottleneck link. When the bottleneck node b is detected, we obtain the bandwidth capacity of the paths between the candidate nodes and from the candidate nodes to the target node. In order to ensure that the performance of the detour path is better than the default path, we introduce an improvement factor, which satisfies the following constraint:

$$\min(cap_{sc}, cap_{ct}) > \alpha \cdot cap_{st} \tag{1}$$

DOI: 10.1002/cpe

where the cap_{ab} is the bandwidth capacity between a and b, while s, c, and t correspond to the source node, the candidate node, and the target node, respectively. The value of α in Eq. (1) reflects the demanding requirements of the bandwidth capacity of the switching path, and the value of α is determined by the actual needs of the applications.

Accordingly, we select the set of the nodes that meet Eq. (1) as the candidate node set $S_{\text{cand\_bottle}}$. Additionally, we use the same way to select some qualified neighbors of node s as the candidates (denoted as $S_{\text{cand\_send}}$). Thus, we can obtain the final candidate set as $S_{\text{cand}} = S_{\text{cand\_bottle}} \cup S_{\text{cand\_send}}$. Besides, we can use available bandwidth measurement tools, such as Pathload [49], pathChirp [50], and Spruce [51] to measure the available bandwidth Avail_sc between the source node s and each candidate node c in S_{cand}, as well as Avail_ct between each candidate node c in S_{cand} and the target t. Thus, the available bandwidth of the new dissemination path from the source to the target satisfies

$$Avail_{st} = \min(Avail_{sc}, Avail_{ct}) \tag{2}$$

After obtaining all the available bandwidth information for all the candidate nodes, we can select the nodes with the predominant performance as the detour nodes and reselect new paths for data dissemination. Additionally, we select the candidate nodes from the neighbors of the bottleneck node to keep the original routing paths as far as possible, which reduces the delay caused by the path switching. However, there also exist two challenging problems that need to be solved: (1) how can we find the real outstanding nodes from a large number of candidates as the detour nodes for the overlay routing and (2) how can we manage the information efficiently, especially for the bandwidth capacity. In the next two subsections, we will introduce the novel corresponding strategies to address the above two problems.

5.2. *Ranking the candidates*

Bandwidth capacity and available bandwidth are both important parameters of the network paths, especially for our flash data dissemination. Thus, we apply a top-k dominance model to rank the candidate nodes, aiming at balancing the multiple factors and selecting the real excellent candidate nodes for path switching.

First, we introduce several elementary definitions related with top-k dominance model, including the concepts of *dominate*, *directly dominate*, *dominance weight*, and *dominance layer*.

Definition 1
(dominate) Given two objects a and b in $\mathbb{R}^d$, a dominates b (denoted as $a \prec b$), if a is not worse than b on all attributes and is better than b on at least one attribute according to the given preference of attribute values.

Definition 2
(directly dominate) Given a data set D and objects $a, b, c \in D$, if $a \prec b$, and not exist any object c, *s.t.* $c \prec b$ and $a \prec c$, then we define that object a directly dominates b (denoted as $a \lessdot b$).

Without loss of generality, we assume that lower is better for all the attributes throughout this paper. Thus, as shown in Figure 4, we can easily find that $C \prec \{G, H, I\}$ and $D \prec \{C, F, G, I, H\}$, while $C \lessdot G$ and $D \lessdot \{C, F\}$, according to the aforementioned two definitions.

Definition 3
(dominance weight) Given a data set D, if object $a \lessdot b$ and b is directly dominated by m objects, then the dominance weight $w_{a,b}$ of a to b is $1/m$ or 0 otherwise.

For example, from Figure 4, we can see that $\{G, F\} \lessdot H$, thus $w_{G,H}$=$w_{F,H}$=1/2. Similarly, $G \lessdot I$, and $w_{G,I} = 1$.

Definition 4
(dominance layer) If an object a is not dominated by other objects, then object a is at the 1-layer. If object a is not at the $1, \cdots, (k-1)$-layer ($k > 1$), and it is only dominated by the objects at the $1, \cdots, (k-1)$-layer, then the object a is at the k-layer.

DOI: 10.1002/cpe

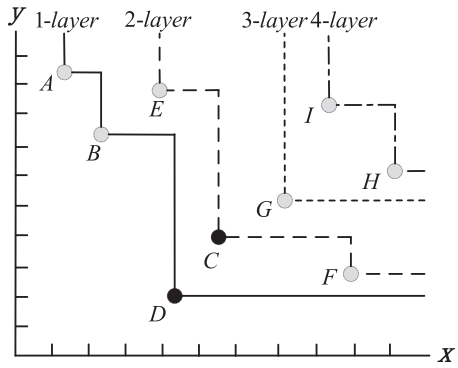


Figure 4. Example of a top-2 dominance data set.

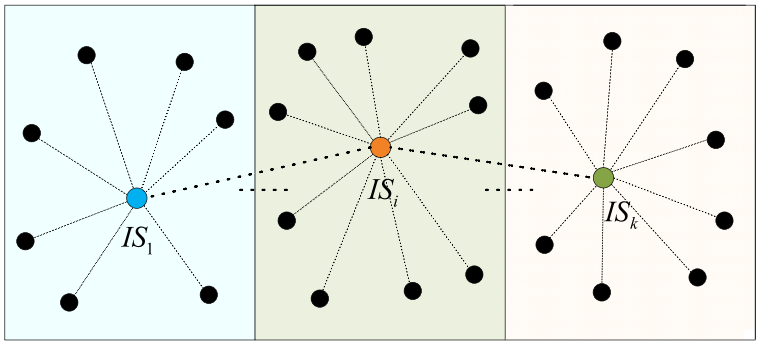


Figure 5. The semi-distributed storage structure.

From the aforementioned definition of dominance layer, we can alleviate the process of finding the direct dominance relation, as one object a dominates another object b, then a will be in the smaller layer than b.

Definition 5
(ranking score) Given a data set D, the ranking score value $f(a)$ of an object a is defined as $f(a) = \sum_{e \in D, a \prec e} (w_{a,e} + f(e))$.

Example 1
As shown in Figure 4, our goal is to find out the top-2 dominance points with the top-2 largest ranking score values.

From Figure 4, we divide all the points in Figure 4 into four layers, and we can easily find the following direct dominance relations: $\{B, G\} \prec I$, $\{F, G\} \prec H$, $C \prec G$, $B \prec E$, $D \prec C$, and $D \prec F$.

According to Definition 5, we know the ranking score value of G is $f(G) = f(I) + f(H) = 1/2 + 1/2 = 1$, $f(C) = 1 + 1 = 2$, and $f(F) = 1/2$. Similarly, $f(D) = f(F) + f(C) + w(D, F) + w(D, C) = 2 + 1/2 + 1 + 1 = 9/2$, and $f(B) = f(E) + f(I) + w(B, E) + w(B, I) = 0 + 0 + 1 + 1/2 = 3/2$. Thus, the top-2 dominance points are D and C in the example.

Therefore, according to the aforementioned top-k dominance model, we can easily rank all the candidate nodes as the detour nodes for path switching, if we change x and y into the values of the bandwidth capacity and available bandwidth of the nodes. Moreover, assume that larger coordinate values are preferred on each dimension.

In addition, one another problem also needs to be considered. When choosing a node as a detour for a specific path, the protocol should further consider if this node has not already been selected as a detour node for another path of the same flash data dissemination, as the overload for the node may degrade the routing performance. Therefore, to avoid this problem, we try to label the detour nodes. Specifically, if one node becomes a detour node, we use a variable (e.g., WORKING = TRUE) to

DOI: 10.1002/cpe

denote it. When the dissemination task is finished, we reset the variable. Therefore, when we choose the candidate nodes as the detour nodes, we additionally check the status of the nodes and choose the best nodes that do not work as the detour nodes to be the relay nodes.

5.3. The information management

The related information in our approach includes the network coordinates of the nodes, the bandwidth capacity, and available bandwidth. The coordinates of the nodes are individually maintained by the nodes themselves, while the available bandwidth information is obtained by the real measurements if needed. Thus, the management of them is not complicated, and in this paper, we only focus on the management of the bandwidth capacity. Generally, the management for the information of bandwidth capacity includes the measuring, collecting, and storage for the bandwidth capacity information.

Different from the available bandwidth measurement, the bandwidth capacity measurement needs much more time and communication overhead. There exist several tools for the measurement of bandwidth capacity, such as pathrate [52], CapProbe [53], bprobe [54], and nettimer [55]. In our proposed approach, we adopt pathrate, because of its high accuracy and popularity. Generally, the bandwidth capacity between two nodes remains unchanged unless the routing paths change. Thus, we only need to update it when the routing paths change, which can be detected by trace-route.

The storage of bandwidth capacity is the foremost part of the information management. Obviously, it is not practical that we store all the information in a centralized node, because of the consideration of communication bottleneck and a single point of failure. On the contrary, if we adopt a fully distributed management manner, then each node will access other remote nodes to obtain information, which may affect the applications on the nodes, and the access efficiency is relatively poor. Therefore, we adopt a semi-distributed architecture to manage the bandwidth information, which is similar to the work in [4].

In addition, in order to reduce the accessing overhead, we divide the *information servers* (*IS*) that are used to manage the information into many clusters according to the delays, which are calculated with the network coordinates obtained by the aforementioned approaches. The specific network structure for storage in our approach is shown in Figure 5.

From Figure 5, we can see that each peer node P is managed by a certain information server IS_i, which is the nearest server to P. Thus, P can obtain or store the information from or to the server IS_i more efficiently than the other servers. Assume that there exist k IS that are deployed in the system, the specific framework for the bandwidth capacity information management can be described as follows:

1. The new participating node P_{new} probes to the k servers to obtain the coordinates of the servers;
2. P_{new} calculates the delays to the servers with the network coordinates of the nodes;
3. P_{new} selects the server IS_i with the smallest delays to manage the information;
4. P_{new} measures the bandwidth capacity information to the other nodes and stores them to IS_i;
5. Each node periodically detects the change of the routing path by trace-route method for each T time interval;
6. Each node remeasures the bandwidth capacity information to the nodes and updates the information in the corresponding ISs if the routing paths change.

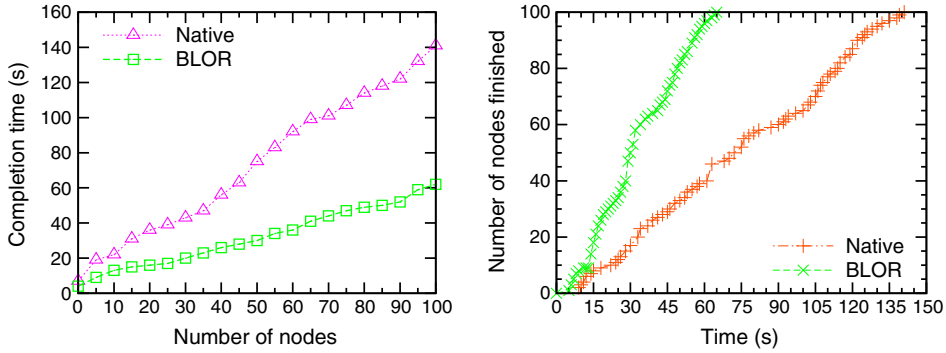
Note that when a node leaves the system, we do not need to notify the *IS* to delete the related information but periodically update the internal information by the *IS*. If we find that the redundant information exists in the system for a long time, then it will be directly deleted.

6. EXPERIMENTAL EVALUATION

6.1. Experimental environments

In this section, we will report extensive experiments to validate our proposals. The experiments mainly consist of two parts: (1) we test the efficiency of data dissemination, with or without BLOR

DOI: 10.1002/cpe



(a) Complete time with 2M file (b) Coverage speed with 2M file

Figure 6. Performance of BLOR and the native method.

optimization, and (2) we analyze and test BLOR from the perspectives of the latency and bandwidth optimizations.

Specifically, in order to ensure the authenticity of the simulation results, we set up a testbed with PeerSim[**] for all the experiments in this paper, which is a classic simulator for P2P network. The project collects the delays between about 2000 DNS by King [56], thus we can obtain the pairwise delays directly. To model the dynamic of the underlying Internet, we adopt BRITE[††] to generate Internet router topologies with 3000 routers.

For the first evaluation part, we employ the network with end-to-end available bandwidth ranging from 800 Kbps to 4 Mbps. The available bandwidth of bottleneck link is set to 150 kbps, and the ratio of the bottleneck links is set to 1%. The bandwidth capacities of all the paths are generated by BRITE, from 100 Mbps to 1 Gbps. Additionally, we generated networks with varying packet loss rates, from 1% to 20%. The latency between nodes is always heterogeneous, as dictated by the router backbone generated by BRITE.

For the second experimental part, we adopt the data sets collected by HP Labs in the PlanetLab[‡‡]. The data sets include the bandwidth capacity and latency information. We sample 200 nodes from 1740 nodes for the simulation and analyze the network paths.

Additionally, we have tested the Lautc algorithm by simulation in the second experiment part. The structure of the PeerSim mainly consists of three parts: the overlay construction, the protocol, and the event generator.

- *Overlay construction* is the core of the simulation, and we construct the overlay based on the delay information;
- *Protocol* implementation is based on events generated by the P2P protocols;
- *Event generator* is mainly used to generate various events, such as nodes joining and data location.

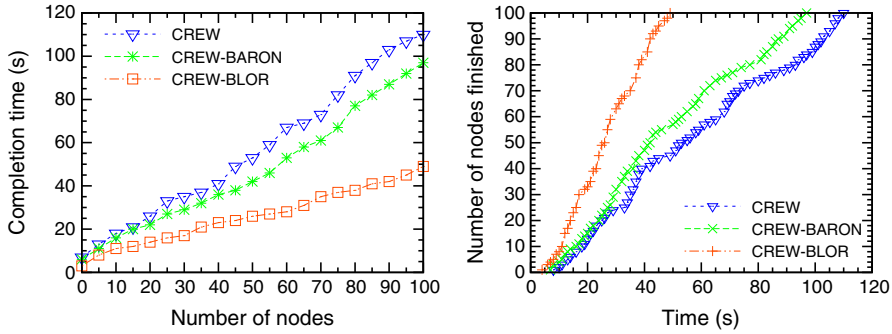
6.2. *Performance metrics*

To evaluate the efficiency of flash data dissemination, we introduce two metrics, that is, *complete time* and *coverage speed* in the experiments, which are proposed in [15]. Then, to evaluate the performance of Lautc, we compare the *latency stretches* of different topologies. Finally, to analyze the performance of the real network paths after optimization, we introduce two other metrics: *optimizable path ratio* and *optimizable pair ratio*.

- *Complete time:* The total time from the start at the sender until all the targets receive all the contents.

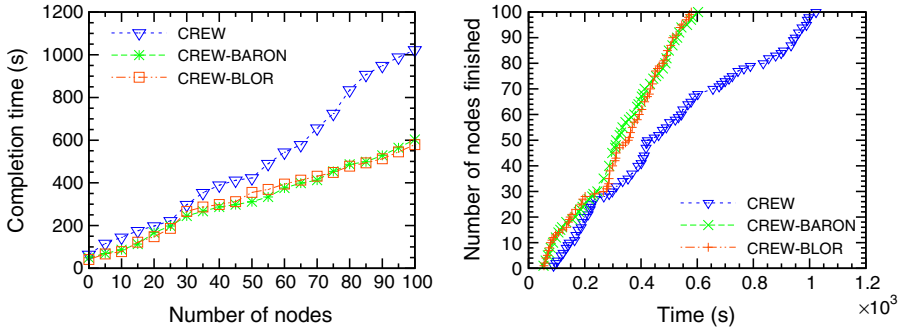
[**]http://peersim.sourceforge.net.
[††]http://www.cs.bu.edu/brite.
[‡‡]http://www.planet-lab.org.

DOI: 10.1002/cpe



(a) Complete time with 2M file (b) Coverage speed with 2M file

Figure 7. Comparison of CREW, BARON, and BLOR.



(a) Complete time with 20M file (b) Coverage speed with 20M file

Figure 8. Comparison of CREW, BARON, and BLOR.

- *Coverage speed:* It indicates how many nodes receive all the content at a certain point of time.
- *Latency stretch:* The ratio of the delays between logical nodes and the delays between nodes in physical network.
- *Optimizable path ratio:* The ratio of the paths that can be optimized among all the paths.
- *Optimizable pair ratio:* The ratio of all the node pairs whose paths can be optimized among all the node pairs.

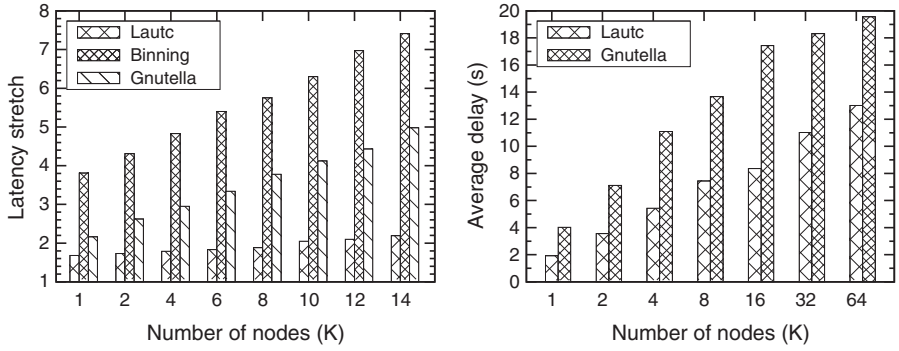
6.3. Evaluation of flash data dissemination

In this section, we test how fast that BLOR can disseminate information to a set of receivers over spread across a wide-area network.

First, we evaluate the time to disseminate a 2 MB file among an increasing number of recipients and compare the efficiency of BLOR with the native routing method. Note that the native routing method means the routing method with the native IP routes (i.e., the default routing protocol in the physical network). We ran each experiment five times and plot the average value of the five runs, and all the results of BLOR are with only 1 hop. As shown in Figure 6(a), when the number of recipients is greater than five, BLOR disseminates much faster than the native routing method and for 100 nodes, BLOR is more than twice as fast as the native routing method. Figure 6(b) shows that the coverage speed of BLOR is larger at any given point of time, which indicates that nodes in BLOR obtain the content much faster than the native method.

Second, we implement the state-of-the-art flash data dissemination method (*concurrent random expanding walkers* (CREW) [15] in PeerSim. CREW is a smart gossip protocol designed to support both content and network heterogeneity and deal with transmission failures without sacrificing dissemination speed. Furthermore, we implement BLOR and BARON [4] based on the CREW protocol. Note that BARON utilizes both capacity and available bandwidth information to quickly

DOI: 10.1002/cpe



(a) Lautc *vs*. Binning for latency stretch (b) Lautc *vs*. Gnutella for average delay

Figure 9. Evaluation of Lautc.

locate alternative overlay paths that provide larger bandwidth than the default paths. In addition, to evaluate the effects of different file sizes, we have tested the dissemination files with the size of 2 and 20 MB, and the experimental results are described in Figures 7 and 8.

From Figures 7 and 8, we can find that BLOR and BARON both can improve the data dissemination. Moreover, for the small-sized file dissemination, BLOR can significantly reduce the transmission time, whereas for larger-sized file, BLOR is very close to BARON. This is probably a consequence of the main transmission time for large files, while the latency for data location and forwarding can be relatively ignored, which indicates that our approach is more useful in flash data dissemination.

6.4. Bandwidth and latency optimization

6.4.1. Evaluation of Lautc. We first compare the latency stretch of the topologies constructed with Lautc, Binning [57], and Gnutella. As shown in Figure 9(a), the average latency stretch is less than 2 in Lautc and remains unchanged as the increase of nodes. Comparatively, the average latency stretch of Binning and Gnutella are about 5 and 3 respectively, which both grow fast as the increase of the node number. Thus, we can conclude that the performance of Lautc is much better than the others. Because the performance of Binning is too worse than the others, in the following tests, we only focus on the comparison between Lautc and Gnutella.

For the data location simulation tests, the simulator carries out 2000 data location requests. From Figure 9(b), we can see that the average data location time of Lautc is 55% times lower than Gnutella. Figure 10 shows the CDF of the ratio of Lautc and Gnutella for the average data location time. In the P2P simulation environments, 2000 and 5000 nodes are produced by BRITE and PeerSim, respectively. Experimental results show that the time of Lautc is much less than Gnutella, as the messages can select close nodes to route, which reduces the routing hops and the total routing delays.

6.4.2. Optimization with BLOR. First, we analyze the status of the real network paths. As shown in Figure 11, the optimizable path ratio is nearly 28%. In addition, it can also be seen from Figure 11 that the optimizable pair ratio accounts nearly 90%, and the paths with 1.5 times gain of bandwidth capacity accounted for nearly 21%. Therefore, we believe that in a real network environment, there exist a large number of paths that can be further optimized.

Second, we analyze the improvement of bandwidth capacity with three path selection strategies: (1) the maximum bandwidth capacity (*MAX-BW*), (2) the minimum delay (*MIN-RTT*), and (3) the average bandwidth capacity of all the paths that meet the constraint of Eq. (1) (*AVG*). For these three strategies, the optimized bandwidth capacity values with different α and the corresponding delays are shown in Figure 12. From Figure 12(a), we can see that *MAX-BW* gets the largest gain, while *AVG* and *MIN-RTT* are more or less the same. Figure 12(b) shows that the *MAX-BW* increases the largest delay, mainly because of the path switching, which increases the hops of routing. However,

DOI: 10.1002/cpe

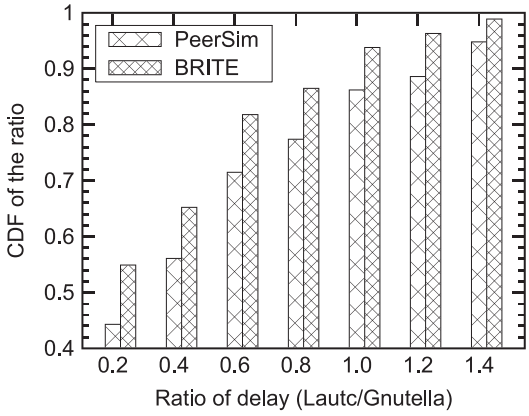


Figure 10. The delay ratio between Lautc and Gnutella.

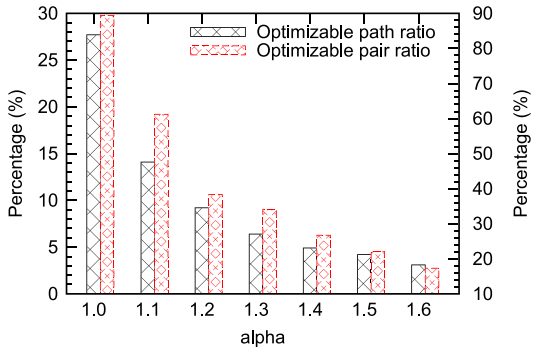
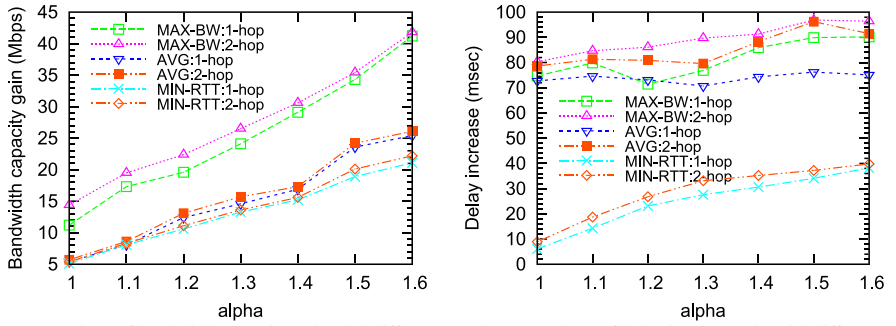


Figure 11. The optimizable path ratio and pair ratio for different α.



(a) Comparison of bandwidth capacity gain with different α (b) Comparison of delay increase gain with different α

Figure 12. Evaluation of path selection strategies.

if we choose *MIN-RTT*, it obtains the minimum bandwidth capacity gains. Therefore, we need to take bandwidth capacity and latency into account for detour routing.

Third, we test the performance of BLOR approach. In the experiment, we periodically introduce background traffic per 5 min on the routing paths. Therefore, the available bandwidth of each path changes constantly. We totally execute 300 times, and obtain the results of using the default routing method and BLOR method for the available bandwidth. From Figure 13, we can see that BLOR can avoid congestion paths or inefficient paths, which significantly improves the efficiency of flash data dissemination. Furthermore, in order to evaluate the effects of the hops, we also examine the performance of BLOR with different hops. As the results of 100 running times shown in Figure 14, the gain of available bandwidth is obvious, which is more limited for larger hops. The larger the

DOI: 10.1002/cpe

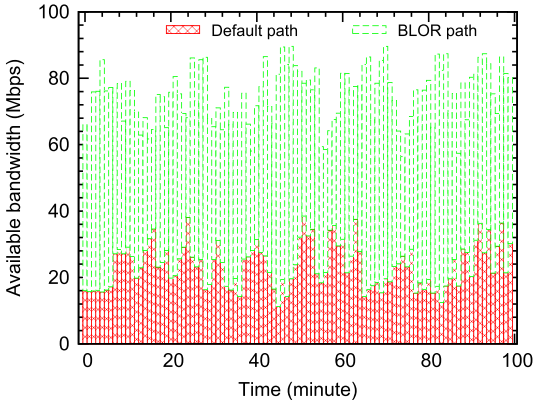


Figure 13. The gain of the available bandwidth.

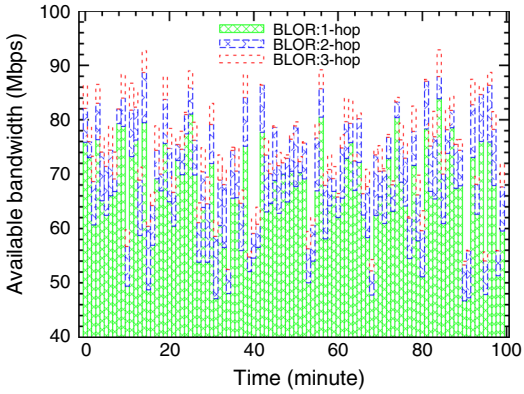


Figure 14. The effects on the hops.

number of hops is, the larger the delays will be; thus, we need to balance the gains and losses. In our experience, we find that one hop or two is enough, for small data with one-hop and two-hop for data with larger size.

7. CONCLUSION

This paper presents BLOR, a novel efficient bandwidth and latency sensitive overlay routing approach for flash data dissemination. BLOR selects paths by optimizing latency, bandwidth capacity, and available bandwidth altogether to solve the problems of inefficient routing and data transmission in the overlay-based flash data dissemination. Extensive experiments have verified that the performance of flash data dissemination can be significantly improved with BLOR. Therefore, we can conclude that BLOR is a practical and efficient overlay routing approach for flash data dissemination.

We are currently investigating some extensions of our work. First, we focus on further optimizing flash data dissemination by establishing an elaborate dissemination topology, such as tree, mesh, or their combination. Second, we plan to further investigate the benefits of our approaches when considering actual communication patterns in real networks and study the effects of churns on the detour routing performance. Furthermore, we implement a real flash data dissemination system with our proposed approach and transplant it to the PlanetLab for further evaluation and application.

DOI: 10.1002/cpe

ACKNOWLEDGEMENTS

The authors would like to thank all the editors and reviewers for their detailed reviews and constructive comments, which have significantly improved the quality of this paper. This work was supported by the National Grand Fundamental Research 973 Program of China (grant no. 2011CB302601), the National Natural Science Foundation of China (grant no. 61379052), the National High Technology Research and Development 863 Program of China (grant no. 2013AA01A213), the Natural Science Foundation for Distinguished Young Scholars of Hunan Province (grant no. 14JJ1026), and the Specialized Research Fund for the Doctoral Program of Higher Education (grant no. 20124307110015).

REFERENCES

1. Tao S, Xu K, Xu Y, Fei T, Gao L, Guerin R, Kurose J, Towsley D, Zhang ZL. Exploring the performance benefits of end-to-end path switching. *Proceedings the 12th IEEE International Conference on Network Protocols (ICNP)*, IEEE, Kuala Lumpur, Malaysia, 2004; 304–315.
2. Tao S, Xu K, Estep A, Gao TFL, Guerin R, Kurose J, Towsley D, Zhang ZL. Improving VoIP quality through path switching. *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, Miami, USA, 2005; 2268–2278.
3. Akella A, Maggs B, Seshan S, Shaikh A, Sitaraman R. A measurement-based analysis of multihoming. *Proceedings of the ACM SIGCOMM Conference*, Karlsruhe, Germany, 2003; 353–364.
4. Lee S, Banerjee S, Sharma P, Yalagandula P, Basu S. Bandwidth-aware routing in overlay networks. *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, Phoenix, USA, 2008; 1732–1740.
5. Caminero A, Rana O, Caminero B, Carrión C. Network-aware heuristics for inter-domain meta-scheduling in grids. *Journal of Computer and System Sciences* 2011; **77**(2):262–281.
6. Lin W, Dou W, Xu Z, Chen J. A QoS-aware service discovery method for elastic cloud computing in an unstructured peer-to-peer network. *Concurrency and Computation: Practice and Experience* 2013; **25**(13):1843–1860.
7. Wang Y, Li S. Research and performance evaluation of data replication technology in distributed storage systems. *Computers & Mathematics with Applications* 2006; **51**(11):1625–1632.
8. Pallickara S, Fox G. Enabling hierarchical dissemination of streams in content distribution networks. *Concurrency and Computation: Practice and Experience* 2012; **24**(14):1594–1606.
9. Li X, Wang Y, Li X, Wang Y. Parallelizing skyline queries over uncertain data streams with sliding window partitioning and grid index. *Knowledge and Information Systems* 2014. DOI: 10.1007/s10115-013-0725-8.
10. Miranda H, Leggio S, Rodrigues L, Raatikainen K. An algorithm for dissemination and retrieval of information in wireless ad hoc networks. *Concurrency and Computation: Practice and Experience* 2009; **21**(7):889–904.
11. Lee V, Liu K. Scheduling time-critical requests for multiple data objects in on-demand broadcast. *Concurrency and Computation: Practice and Experience* 2010; **22**(15):2124–2143.
12. Alicherry M, Lakshman TV. Network aware resource allocation in distributed clouds. *Proceedings of the IEEE INFOCOM*, Orlando, USA, 2012; 963–971.
13. Li X, Wang Y, Li X, Wang Y. Parallel skyline queries over uncertain data streams in cloud computing environments. *International Journal of Web and Grid Services* 2014; **10**(1):24–53.
14. Wang Y, Li X, Li X, Wang Y. A survey of queries over uncertain data. *Knowledge and Information Systems* 2013; **37**(3):485–530.
15. Deshpande M, Xing B, Lazardis I, Hore B, Venkatasubramanian N, Mehrotra S. CREW: a gossip-based flash dissemination system. *Proceedings the 26th IEEE International Conference on Distributed Computing Systems (ICDCS)*, Lisboa, Portugal, 2006; 45–52.
16. Andersen DG, Balakrishnan H, Kaashoek MF, Morris R. Resilient overlay networks. *Proceedings the ACM Symposium on Operating Systems Principles (SOSP)*, New York, USA, 2001; 131–145.
17. Freedman MJ, Freudenthal E, Mazieres D. Democratizing content publication with coral. *Proceedings the 1st Conference on Symposium on Networked Systems Design and Implementation (NSDI)*, San Francisco, USA, 2004; 18–18.
18. Kostić D, Rodriguez A, Albrecht J, Vahdat A. Bullet: high bandwidth data dissemination using an overlay mesh. *Proceedings the Nineteenth ACM Symposium on Operating Systems Principles (SOSP)*, New York, USA, 2003; 282–297.
19. Macedo DF, Correia LHA, dos Santos AL, Loureiro AAF, Nogueira JMS. A rule-based adaptive routing protocol for continuous data dissemination in WSNs. *Journal of Parallel and Distributed Computing (JPDC)* 2006; **66**(4): 542–555.
20. Kubiatowicz J, Bindel D, Chen Y, Czerwinski S, Eaton P, Geels D, Gummadi R, Rhea S, Weatherspoon H, Weimer W, Wells C, Zhao B. Oceanstore: an architecture for global-scale persistent storage. *ACM Sigplan Notices* 2000; **35**(11):190–201.
21. Hefeeda M, Habib A, Botev B, Xu D, Bhargava B. PROMISE: peer-to-peer media streaming using collectcast. *Proceedings of the Eleventh ACM International Conference on Multimedia*, Berkeley, USA, 2003; 45–54.
22. Castro M, Druschel P, Kermarrec AM, Nandi A, Rowstron A, Singh A. SplitStream: high-bandwidth multicast in cooperative environments. *ACM SIGOPS Operating Systems Review* 2003; **37**(5):298–313.

23. Fernandess Y, Malkhi D. On collaborative content distribution using multi-message gossip. *Journal of Parallel and Distributed Computing (JPDC)* 2007; **67**(12):1232–1239.
24. Wu CJ, Li CY, Yang KH, Ho JM, Chen MS. Time-critical data dissemination in cooperative peer-to-peer systems. *Proceedings of IEEE Global Communications, 2009*, Honolulu, USA, 2009; 1–6.
25. Castro M, Druschel P, Kermarrec AM, Rowstron AIT. SCRIBE: a large-scale and decentralized application-level multicast infrastructure. *IEEE Journal on Selected Areas in Communications* 2002; **20**(8):1489–1499.
26. Rahimian F, Girdzijauskas S, Payberah AH, Haridi S. Vitis: a gossip-based hybrid overlay for Internet-scale publish/subscribe enabling rendezvous routing in unstructured overlay networks. *IPDPS*, Anchorage, USA, 2011; 746–757.
27. Zheng Z, Wang Y, Ma X. Peerchatter: a peer-to-peer architecture for data distribution over social networks. *INFORMATION-An Internal Interdisciplinary Journal* 2012; **15**(1):259–266.
28. Savage S, Collins A, Hoffman E, Snell J, Anderson T. The end-to-end effects of Internet path selection. *ACM SIGCOMM Computer Communication Review* 1999; **29**(4):289–299.
29. Opos JM, Ramabhadran S, Terry A, Pasquale J, Snoeren AC, Vahdat A. A performance analysis of indirect routing. *Proceedings of the IEEE International Parallel & Distributed Processing Symposium (IPDPS)*, Long Beach, USA, 2007; 1–10.
30. Cha M, Moon S, Park CD, Shaikh A. Placing relay nodes for intra-domain path diversity. *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, Barcelona, Spain, 2006; 1–12.
31. Gummadi KP, Madhyastha HV, Gribble SD, Levy HM, Wetherall D. Improving the reliability of Internet paths with one-hop source routing. *Proceedings of the 6th Conference on Symposium on Operating Systems Design & Implementation (OSDI)*, San Francisco, USA, 2004; 1–15.
32. Fei T, Tao S, Gao L, Guerin R. How to select a good alternate path in large peer-to-peer systems. *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, Barcelona, Spain, 2006; 1106–1118.
33. Cui W, Stoica I, Katz RH. Backup path allocation based on a correlated link failure probability model in overlay networks. *Proceedings of the IEEE International Conference on Network Procotols (ICNP'02)*, Paris, France, 2002; 236–245.
34. Tang C, McKinley PK. A distributed multipath computation framework for overlay network applications. *Technical Report MSU-CSE-04-18*, Michigan State University, 2004.
35. Nakao A, Peterson L, Bavier A. A routing underlay for overlay networks. *Proceedings of the ACM SIGCOMM*, Karlsruhe, Germany, 2003; 11–18.
36. Jain M, Dovrolis C. Path selection using available bandwidth estimation in overlay-based video streaming. *Computer Networks* 2008; **52**(12):2411–2418.
37. Li X, Wang Y, Fu Y, Li X, Sun W. BLOR: bandwidth and latency sensitive overlay routing for flash data dissemination. *Proceedings of the IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW)*, Shanghai, China, 2012; 53–64.
38. Zhu Y, Dovrolis C, Ammar M. Proactive and reactive bandwidth-driven overlay routing: a simulation study. *Computer Networks* 2006; **50**(6):742–762.
39. Hu N, Li L, Mao Z, Steenkiste P, Wang J. Locating Internet bottlenecks: algorithms, measurements, and implications. *Proceedings of the ACM SIGCOMM, 2004*, Portland, USA, 2004; 41–54.
40. Ng TS, Zhang H. Predicting Internet network distance with coordinates-based approaches. *Proceedings IEEE International Conference on Computer Communications (INFOCOM)*, New York, USA, 2002; 170–179.
41. Lim H, Hou JC, Choi CH. Constructing Internet coordinate system based on delay measurement. *Proceedings of the 3rd ACM SIGCOMM Workshop on Internet Measurement (IMC)*, Karlsruhe, Germany, 2003; 129–142.
42. Dabek F, Cox R, Kaashoek F, Morris R. Vivaldi: a decentralized network coordinate system. *ACM SIGCOMM Computer Communication Review* 2004; **34**(4):15–26.
43. Zhang R, Hu C, Lin X, Fahmy S. A hierarchical approach to Internet distance prediction. *Proceedings of the 26th International Conference on Distributed Computing Systems (ICDCS)*, Lisboa, Portugal, 2006; 73–80.
44. Costa M, Castro M, Rowstron R, Key P. PIC: practical Internet coordinates for distance estimation. *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS)*, Tokyo, Japan, 2004; 178–187.
45. Li X, Wang Y, Fu Y, Sun W. PNDP: predicting accurate network distance based on clustering. *Information–An International Interdisciplinary Journal* 2012; **15**(1):267–274.
46. Fu Y, Wang Y, Biersack E. HybridNN: an accurate and scalable network location service based on the inframetric model. *Future Generation Computer Systems* 2013; **29**(6):1485–1504.
47. Katti S, Katabi D, Blake C, Kohler E, Strauss J. MultiQ: automated detection of multiple bottleneck capacities along a path. *Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement (IMC)*, Taormina, Italy, 2004; 245–250.
48. Ribeiro VJ, Riedi RH, Baraniuk RG. Locating available bandwidth bottlenecks. *IEEE Internet Computing* 2004; **8**(5):34–41.
49. Jain M, Dovrolis C. Pathload: a measurement tool for end-to-end available bandwidth. *Proceedings of the Passive and Active Measurements Workshop (PAM)*, Fort Collins, USA, 2002; 14–25.
50. Ribeiro V, Riedi R, Baraniuk R, Navratil J, Cottrell L. pathChirp: efficient available bandwidth estimation for network paths. *Proceedings of the Passive and Active Measurements Workshop (PAM)*, San Diego, USA, 2003; 1–11.
51. Strauss J, Katabi D, Kaashoek F. A measurement study of available bandwidth estimation tools. *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement (IMC)*, Karlsruhe, Germany, 2003; 39–44.
52. Dovrolis C, Ramanathan P, Moore D. Packet-dispersion techniques and a capacity-estimation methodology. *IEEE/ACM Transactions on Networking* 2004; **12**(6):963–977.

DOI: 10.1002/cpe

53. Kapoor R, Chen LJ, Lao L, Gerla M, Sanadidi MY. CapProbe: a simple and accurate capacity estimation technique. *ACM SIGCOMM Computer Communication Review* 2004; **34**(4):67–78.
54. Carter RL, Crovella ME. Measuring bottleneck link speed in packet-switched networks. *Performance Evaluation* 1996; **27**:297–318.
55. Lai K, Baker M. Measuring link bandwidths using a deterministic model of packet delay. *ACM SIGCOMM Computer Communication Review* 2000; **30**(4):283–294.
56. Gummadi KP, Saroiu S, Gribble SD. King: estimating latency between arbitrary Internet end hosts. *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurement (IMC)*, Warsaw, Poland, 2002; 5–18.
57. Ratnasamy S, Handley M, Karp R, Shenker S. Topologically-aware overlay construction and server selection. *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, New York, USA, 2002; 1190–1119.

DOI: 10.1002/cpe