Dynamic overlay routing based on available bandwidth estimation: A simulation study*

Yong Zhu, Constantinos Dovrolis, Mostafa Ammar

College of Computing

Georgia Institute of Technology

Abstract

Dynamic overlay routing has been proposed as a way to enhance the reliability and performance of IP networks. The major premise is that overlay routing can bypass congestion, transient outages, or suboptimal paths, by forwarding traffic through one or more intermediate overlay nodes. In this paper, we perform an extensive simulation study to investigate the performance of dynamic overlay routing. In particular, we leverage recent work on available bandwidth (avail-bw) estimation, and focus on overlay routing that selects paths based on avail-bw measurements between adjacent overlay nodes. First, we compare two overlay routing algorithms, reactive and proactive, with shortest-path native routing. We show that reactive routing has significant benefits in terms of throughput and path stability, while proactive routing is better in providing flows with a larger safety margin ("headroom"), and propose a hybrid routing scheme that combines the best features of the previous two algorithms. We then examine the effect of several factors, including network load, traffic variability, link-state staleness, number of overlay hops, measurement errors, and native sharing effects. Some of our results are rather

*This work was supported by the NSF CAREER award ANIR-0347374, and by a Georgia Tech Broadband Institute (GTBI) grant.

surprising. For instance, we show that a significant measurement error, even up to 100% of the actual avail-bw value, has a negligible impact on the efficiency of overlay routing.

1 Introduction

Overlay networks have been the subject of significant research and practical interest recently [1, 2, 3, 4, 5, 6, 7]. The initial motivation for overlay networks was mainly due to the following three shortcomings of the IP routing infrastructure (referred to as the *native network*). First, to deal with the slow fault recovery and routing convergence of BGP [8], overlay networks can bypass broken paths by rerouting traffic through intermediate overlay nodes. The detection of broken paths by overlay nodes can be quickly performed through active probing. Second, the IP routing model is basically a "one-size-fits-all" service, providing the same route independent of performance requirements. Instead, overlay networks can offer different routes to the same destination, depending on the performance metric (*e.g.,* delay, throughput, loss rate) that each application cares for [1]. Third, the fact that interdomain IP routing is largely determined by ISP commercial policies often results in suboptimal paths [9]. Overlay networks can provide better end-to-end performance by routing through intermediate overlay nodes, essentially forcing the flow of traffic in end-to-end paths that would otherwise not be allowed by ISP policies.

Over the last few years much has been learnt about overlay networks. To name some major steps, the Resilient Overlay Network (RON) was the first wide-scale overlay implementation and testbed, over which several measurement studies have been performed [1]. Those studies showed the fault recovery and performance benefits of overlay routing [1, 10, 11]. Another research thread focused on enhanced services that can be provided by overlay networks, such as multicasting [2, 12], end-to-end QoS [6, 13], secure overlay services [14], and content delivery [4]. Overlay path selection algorithms, focused on QoS-aware routing, have been studied in [13]. The impact of the overlay topology on the resulting routing performance was studied in [15], suggesting that knowledge of the native network topology can significantly benefit the overlay construction.

Overlay networks rely heavily on active probing, raising questions about their scalability and long-

term viability. For instance, Nakao et al. argue that independent probing by various overlay networks is untenable, and that a "routing underlay" service is needed that will be shared by different overlays [16]. The high cost of overlay network probing was the motivation for the tomography-based monitoring scheme reported in [17]. More recently, a comparison between overlay networks and multihoming has been reported in [18, 19], suggesting that multihoming may be capable to offer almost the same performance benefits with overlay networks, but in a much simpler and more cost-effective way. Furthermore, an ongoing debate focuses on the "selfishness" of overlay routing, and on the potential performance inefficiency and instability that it can cause [20, 21, 22, 23, 24]. It is clear that there is still much to be learnt about overlay networks, and that the key debates on the scalability, efficiency, and stability of overlay networks have to be addressed before their wider-scale deployment.

In this paper, we focus on an aspect of dynamic overlay networks that has been largely unexplored previously, namely, *the use of available bandwidth (avail-bw) measurements in the path selection process*. Previous work on overlay routing assumed that the only information that can be measured or inferred about the underlying native network is related to delays, loss rate, and sometimes TCP throughput. The problem with these metrics is that they are not direct indicators of the traffic load in a path: delays can be dominated by propagation latencies (which do not depend on network load), losses occur after congestion has already taken place, while measurements of TCP throughput can be highly intrusive and they can be affected by a number of factors (such as flow size, advertised window, or TCP stack). The avail-bw, on the other hand, directly represents the additional traffic rate that a path can carry before it gets saturated. Consequently, an overlay node can route a traffic flow (or an aggregation of many flows) to a path only if the maximum throughput of that flow is lower than the avail-bw of the path. The use of avail-bw in overlay routing has recently become possible, based on recent advances in avail-bw measurement techniques and tools [25, 26, 27, 28, 29]. Obviously, if an application has additional requirements on the end-to-end delay or loss rate, then those requirements can be jointly considered with avail-bw in the path selection process.

This paper presents an extensive simulation study of dynamic overlay routing based on avail-bw estimation. We first focus on two algorithms that represent two different and general approaches: *proactive*

and *reactive* routing. The former attempts to always route a flow in the path that provides the maximum avail-bw, so that the flow can avoid transient congestion due to cross traffic (and overlay traffic) fluctuations. The latter reroutes a flow only when the flow cannot meet its throughput requirement in the current path, and there is another path that can provide higher avail-bw. The routing algorithms are compared in terms of efficiency, stability, and safety margin (or headroom). We show that reactive routing has significant benefits in terms of throughput and stability, while proactive routing is better in providing flows with a wider safety margin. We then propose a hybrid routing scheme that combines the best features of the previous two algorithms. We also examine the effect of several factors, including network load, traffic variability, link-state staleness, number of overlay hops, measurement errors, and native sharing effects. Some of our results are rather surprising. For example, we show that a significant measurement error, even up to 100% of the actual avail-bw value, has a negligible impact on the efficiency of overlay routing. Also, we show that a naive overlay routing algorithm that ignores native sharing between overlay paths performs equally well with an algorithm that has a complete view of the native topology and of the avail-bw in each native link. We note that the main contribution of this paper is not a novel routing algorithm or a new avail-bw measurement technique, but an investigation of the applicability of avail-bw estimation in dynamic overlay routing.

The rest of this paper is organized as follows. Section 2 presents the model of dynamic overlay routing that we consider, and the two routing algorithms that we compare. Section 3 describes the simulator, states some simplifying assumptions, and formalizes the three main performance metrics we use. Section 4 is the main body of the paper, comparing the two routing schemes, proposing a hybrid algorithm, and examining the effect of various factors. We conclude in Section 5.

2 Dynamic overlay routing

2.1 Overlay routing model

We consider two layers of network infrastructure: the native network and a virtual overlay network. The native network includes end-systems, routers, links, and the associated routing functionality, and it

provides best-effort datagram delivery between its nodes. The overlay network is formed by a subset of the native layer nodes (routers and/or end-systems) interconnected through overlay links to provide enhanced services. Overlay links are virtual in the sense that they are IP tunnels over the native network, *i.e.*, overlay packets are encapsulated in IP datagrams and sent from one overlay node to another through the native network. Figure 1 shows an example of an overlay network constructed over a native network. Note that since overlay links are virtual, the overlay network topology can be a full mesh allowing maximum flexibility in choosing overlay routes.[1]

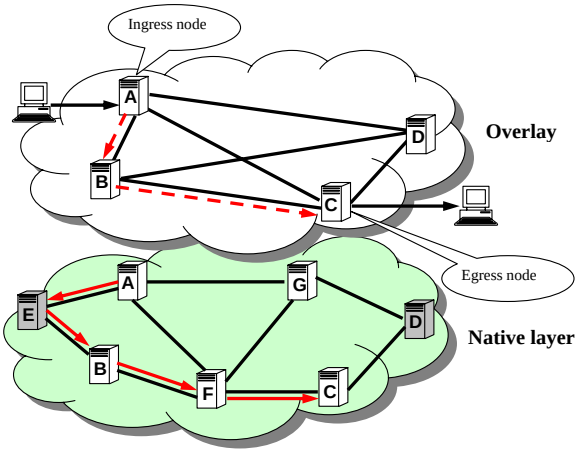


Figure 1. Overlay and native network layers.

An important service that overlay networks provide is *dynamic path selection* based on specified performance objectives. The performance of a path can be a function of the delay, loss rate, and/or avail-bw in the path, among other metrics. Additionally, different traffic classes can be associated with a different path performance metric. An overlay flow arrives at an *ingress node*, destined to a certain *egress node*. Upon the flow's arrival, the ingress node determines the best overlay path to the egress node based, ideally, on the current state and performance of the overlay links (referred to as *overlay link-state*). The chosen overlay path information is then included in the header of each packet (*source routing*), and the packet is forwarded to the corresponding sequence of overlay nodes. To provide resilience to network failures and load variations, the ingress node of an active overlay flow checks for a better path at the end

[1]Overlay networks with hundreds of nodes may require a sparser connectivity, or some form of hierarchical routing, to deal with scalability problems in the link-state measurement and dissemination process.

of every *path update period* P_u, during the lifetime of the flow. If a better path is found, the flow can be switched to that path. The previous path update events and the corresponding time scales are illustrated in Figure 2.

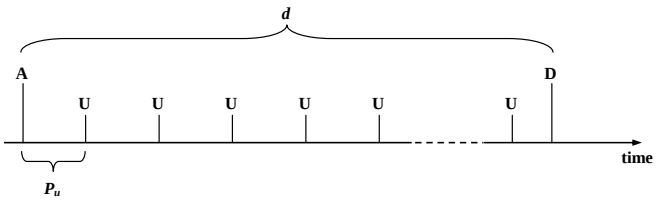


Figure 2. Overlay flow events and the related time scales. A, U **and** D **are the flow arrival, path update, and flow departure events, respectively.** d **is the flow duration and** P_u **is the path update period.**

To perform dynamic path selection, the overlay nodes need to perform *link-state measurement* and *link-state dissemination*. The overlay link-state is the input to the overlay routing algorithm. The state of an overlay link can be represented by a collection of performance metrics, such as delay, loss rate, availability, or capacity. In this work, we focus exclusively on avail-bw, leveraging recent advances in relate [25, 26, 27, 28, 29]. Of course it is possible to further limit the path selection algorithms with additional constraints on the path delay or loss rate, for example.

The avail-bw, also known as residual capacity, of a native link is defined as the capacity of the link minus its average traffic load. The avail-bw of an overlay link (or native path), on the other hand, is the minimum avail-bw among all native links that comprise that overlay link (or native path). Unlike the avail-bw of a native link, which can be easily measured passively by the corresponding router, the avail-bw of overlay links cannot be estimated passively by overlay nodes. Instead, the avail-bw of an overlay link has to be measured through active end-to-end probing techniques performed by the overlay nodes. Recent developments in end-to-end avail-bw estimation provided us with tools and techniques that can estimate the avail-bw of a network path. These techniques are based on special probing packet streams that can identify in a non-intrusive way the maximum rate that will not cause congestion in a path. The latency of the existing measurement techniques varies from a few tens of milliseconds to tens of seconds, depending on whether the tools run continuously in the background or whether they

run in a "measure-once-and-terminate" mode. Their accuracy depends on the traffic burstiness and the number of bottleneck links in the path, and relative measurement errors in the range of 10-30% should be expected [25].

Each overlay node measures the avail-bw of the paths to its adjacent overlay nodes. Periodically, the link-state information that is generated from these measurements is disseminated to all other overlay nodes. The link-state database of an overlay node is refreshed upon receiving new link-state information. Note that the link-state measurement and dissemination are performed independent of any flow-related events. There are three important time scales involved in the avail-bw measurement and dissemination process: the *measurement delay* (D_m), the *link-state refresh period* (P_r) and the *dissemination delay* (D_d). The measurement delay D_m is the time needed to generate a new avail-bw estimate. The link-state refresh period P_r (or simply, refresh period) is the time interval between consecutive updates of the local avail-bw link state. Note that P_r cannot be less than D_m, but it could be larger to reduce the link-state dissemination overhead. The end of a link-state refresh period is determined by the end of the last measurement period. The dissemination delay D_d^{ij} refers to the time needed for the new link-state generated by the i'th overlay node to reach the j'th overlay node. We assume that D_m and P_r are constant, while D_d^{ij} varies randomly for each pair (i, j) of overlay nodes. The overlay link-state measurement and dissemination events and time scales are shown in Figure 3.

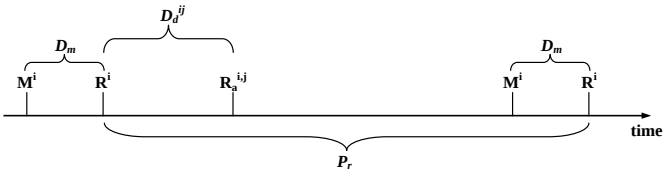


Figure 3. Time scales for the measurement and dissemination of overlay link-state at the i'th overlay node. M^i represents the start of an avail-bw measurement for all the egress overlay links of the i'th overlay node. R^i is a link-state refresh event, and it takes place at the end of the last avail-bw measurement. $R_a^{i,j}$ represents the arrival of the new link-state from overlay node i to node j.

2.2 Overlay routing algorithms

We model the overlay topology as a directed graph $G = (V, L)$ whose vertices and links represent the set of overlay nodes and overlay links, respectively. The avail-bw of each overlay link $l = (u, v) \in L$ is denoted by $b(l)$. An overlay path p is a sequence of one or more overlay links and its avail-bw $b(p)$ is defined as $b(p) = \min_{l \in p} b(l)$.

We use the overlay flow as the basic traffic unit for overlay routing, meaning that all packets of a flow are sent via the same path determined for that flow. Each overlay flow is modelled by four parameters $f = (v_i, v_e, d, r)$; $v_i, v_e \in V$ are the ingress and egress overlay nodes of the flow, and d is the flow duration. The last parameter r is the flow's *maximum throughput limit (max-rate limit)*, and it represents the maximum throughput that the flow can achieve. For instance, the throughput of a flow may be limited by its ingress or egress access capacity, the throughput of a streaming flow may be limited by the rate of the best-quality encoder, and the throughput of a TCP flow may be limited by the size of end-host socket buffers. Due to limited network resources, a flow's actual throughput can be lower than its max-rate-limit r. We therefore use the symbol a to represent the current value of the *achieved throughput* of a flow ($a \leq r$).

When we compare the path that a flow is currently routed on with another path we need to take into account the load that the flow already imposes on the former. To do so, we introduce another metric referred to as *headroom*. For a flow f, the headroom $h(f, l)$ at an overlay link l is defined as

$$h(f,l) = \begin{cases} b(l) + a & \text{if } f \text{ is routed on } l \\ b(l) & \text{otherwise} \end{cases} \tag{1}$$

Similar to avail-bw, the headroom of a path can be defined as the minimum headroom among all links along that path, *i.e.*, for an overlay path p,

$$h(f,p) = \min_{l \in p} h(f,l) \tag{2}$$

Note that the headroom $h(f,p)$ of path p is equal to the avail-bw $b(p)$ if flow f is *not* routed on p; otherwise, the headroom is larger than the avail-bw by the flow's achieved throughput a.

In this paper, we first consider two overlay path selection schemes: *proactive overlay routing* and *reactive overlay routing*. In both schemes, a flow will be initially routed on the path that provides the maximum headroom. With the proactive algorithm, the flow is switched to the path that appears to have the maximum headroom at the end of each path update period (see Figure 2). Note that due to potential staleness in the link-state information, that path may not actually be the best choice. With the reactive algorithm, on the other hand, the flow stays at its current path if it has achieved its max-rate limit r ("satisfied flow"). Otherwise, the flow is "unsatisfied" and it is routed on the path with the maximum headroom; that path may be the same with the previously used path.

The intuition behind proactive routing is that the maximum headroom path can provide a flow with *a wider safety margin* to avoid transient congestion due to traffic load variations, measurement errors, and stale link-state. The intuition behind reactive routing is that a flow should stay at its current path if it is already satisfied, leading to fewer path changes and *more stable overlay routing*.

The path selection algorithm for the proactive and reactive schemes is based on the shortest-widest routing algorithm of [30]. The pseudo-code for both reactive and proactive overlay routing is given in Table 1. Even though the algorithmic difference between the two routing schemes is minor, Section 4 shows that it can result in very different performance.

INPUT:
    $f = (v_i, v_e, d, r)$: overlay flow under consideration;
    $P = \{p_i\}$: set of alternative paths from v_i to v_e;
    a: achieved throughput of f (zero for new flow);
OUTPUT:
    Selected path p';

if ((Proactive-Routing) OR (Reactive-Routing AND $a < r$))
    Update headroom $h(f,p)$ for all $p \in P$;
    $p' = argmax_{p_i \in P} h(f, p_i)$;
    Route f on path p';

Table 1. The pseudo-code for both reactive and proactive overlay routing.

3 Simulation model and performance metrics

3.1 Simulation model

We have implemented a flow-level discrete-event simulator for dynamic overlay routing. The native network topology is based on the core US topology of four large ISPs (Sprint, ATT, Level3 and Verio), estimated from the measurements of the Rocketfuel project [31] (see Figure 4). These four ISPs are tier-1 providers and so they are interconnected in a full mesh, with three inter-ISP connections per ISP pair. The inter-ISP links connect routers that are located in the same city. We assume that the native-layer routes are based on the shortest path algorithm, and that they do not change with time (at least for the time scales of overlay routing that we are interested in).

The overlay network consists of 18 overlay nodes located in major US cities. Each overlay node is connected with an overlay access link to one of the four ISPs at the corresponding router that is located in the same city. The overlay nodes are interconnected in a full-mesh topology.

There are three types of native links: intra-ISP links, inter-ISP links and overlay access links. In our simulation, the capacity of these three link types is uniformly distributed in the range of $[500, 1500]$, $[400, 600]$ and $[8000, 12000]$Mbps, respectively. Note that the most likely bottlenecks are the inter-ISP links, while the overlay access links are the least likely bottlenecks.

Overlay flows are generated according to a Poisson process with average arrival rate F_a.[2] The flow duration is exponentially distributed with mean F_d. The selection of the source and destination nodes for the overlay flows follows a randomly generated (non-uniform) traffic matrix. The flow max-rate limit follows an exponential distribution with mean F_r.

We also simulate some non-overlay traffic, referred to as *cross traffic*. The cross traffic causes random load fluctuations in the native network. Specifically, the cross traffic at each native link is modelled as a fluid process with variable rate. The rate change events take place based on a Poisson process, independent of the rate changes at other links. The average time period between rate variations is F_c.

[2]The Poisson flow arrival model is reasonable, as long as there are no correlations on bursts in the overlay flow arrival process. The Poisson model has been previously validated for application session arrivals [32].

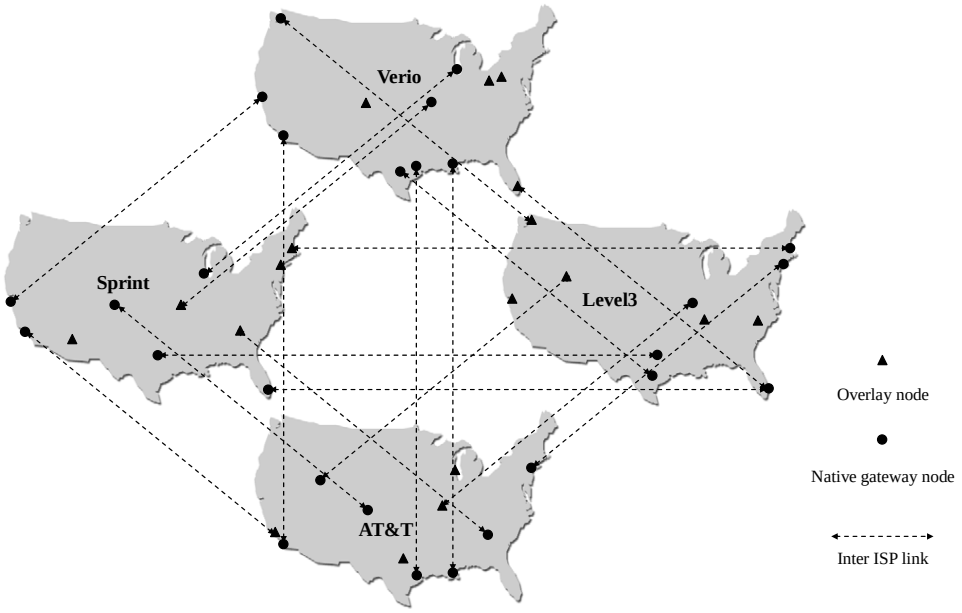


Figure 4. A sketch of the native network topology (also showing the location of the overlay nodes).

The rate of the cross traffic after a rate change event is chosen randomly as $\min(b, x \cdot C)$, where b is the avail-bw of the link, x is uniformly distributed in $[0, 1]$, and C is the link capacity. Since the cross traffic rate is at most b, this traffic can cause load variations but not congestion.

Our simulator does not capture bandwidth sharing in saturated links or congestion control by overlay flows. Consequently, if a new flow arrives at a saturated link, then the new flow will obtain zero throughput while the existing flows will maintain their previous throughput. The subtle interactions between congestion control and dynamic overlay routing are outside the scope of this paper. Also, an unsatisfied flow can only increase its throughput, as a result of additional avail-bw in its path, at a path update event.

Table 2 shows the set of important parameters and their default values in our simulation study.

Each simulation result is obtained by running the simulator until 30,000 overlay flows have been serviced. Furthermore, to avoid the effect of transient simulation effects, we start to collect data after the first 10,000 overlay flows.

Table 2. Major simulation parameters and their default values

Overlay flow and cross traffic parameters		
Flow arrival rate	F_a	10.0 flows/sec (average)
Flow duration	F_d	50sec (average)
Max-rate limit	F_r	20Mbps (average)
Cross traffic rate change period	F_c	20sec (average)
Native network parameters		
Number of native nodes		275
Number of native links		1164
Intra-ISP link capacity		$[500, 1500]$Mbps
Inter-ISP link capacity		$[400, 600]$Mbps
Overlay access link capacity		$[8000, 12000]$Mbps
Overlay routing parameters		
Link-state measurement delay	D_m	0.1sec
Link-state refresh period	P_r	0.5sec
Link-state dissemination delay	D_d	$[0, 0.2]$sec
Path update period	P_u	1.0sec

3.2 Performance metrics

We evaluate overlay routing based on three important aspects: *efficiency*, *stability*, and *safety margin*. Efficiency refers to the ability of overlay routing to achieve higher throughput than native routing, by avoiding saturated links. Stability refers to the frequency with which overlay flows switch between different paths. The safety margin represents the robustness of overlay routing in the presence of cross traffic fluctuations, measurement errors, and stale link-state information.

Specifically, to quantify the efficiency of a routing scheme we use the *normalized average throughput* T. This is defined as the total amount of data sent by completed overlay flows, normalized by the amount of data that would have been sent if each of these flows had achieved its max-rate limit,

$$T = \frac{\sum_{i=1}^{k} \int a_i(t)dt}{\sum_{i=1}^{k} r_i \cdot d_i} \leq 1 \tag{3}$$

where k is the number of completed flows, a_i and r_i are the achieved throughput and the max-rate limit of the i'th flow, respectively, and d_i is the duration of the i'th flow. Notice that, given the limited network capacity resources, it may be infeasible to have T=100% for a given overlay load. Consequently, under

the same offered load, a higher value of T reflects a more efficient overlay routing scheme.

To quantify the stability of a routing scheme we use the *path switching ratio* S. Suppose that an overlay flow i experienced u_i path update events during its lifetime, and that c_i among these updates were path changes. The ratio $\frac{c_i}{u_i} \in [0, 1]$ reflects the relative frequency with which flow i switched between paths: if it is one the flow switched paths with every path update, while if it is zero the flow never switched paths. The *path switching ratio* S is the weighted average of the previous ratio across all completed overlay flows, with weights proportional to the flow durations,

$$S = \sum_{i=1}^{k} \left(\frac{u_i}{\sum_{j=1}^{k} u_j} \cdot \frac{c_i}{u_i} \right) = \frac{\sum_{i=1}^{k} c_i}{\sum_{i=1}^{k} u_i} \tag{4}$$

A higher value of S indicates that flows switch paths more frequently and so the network is less stable.

To quantify the safety margin of a routing scheme we use the *normalized average headroom* H. As we did for normalized throughput, we normalize the headroom of each flow by its max-rate limit. Instead of measuring the headroom of a flow as a continuous function of time however, we use Poisson sampling to estimate the time-average of the normalized per-flow headroom. Consider the j'th overlay flow at a sampling instant i, and let h_{ij} and r_{ij} be the headroom and max-rate limit of that flow, respectively. The flow's relative headroom is h_{ij}/r_{ij}. The weighted average of the relative headroom of all active flows at the i'th sampling instant, weighted by the max-rate limit of each flow, is

$$H_i = \sum_{j} \frac{r_{ij}}{\sum_{j'} r_{ij'}} \cdot \frac{h_{ij}}{r_{ij}} = \frac{\sum_j h_{ij}}{\sum_j r_{ij}}$$

Taking the corresponding weighted average across all sampling instants i, we get that the *normalized average headroom* is

$$H = \sum_{i} \frac{\sum_j r_{ij}}{\sum_{i'} \sum_{j'} r_{i'j'}} \cdot H_i = \frac{\sum_i \sum_j h_{ij}}{\sum_i \sum_j r_{ij}} \tag{5}$$

In the simulations of Section 4, the average sampling period for the calculation of H is 0.5 seconds. Note that H, as opposed to T, can be larger than 100%. Also, a larger value of H does not necessarily mean a higher value of T. In Section 4.4, however, we show how larger headroom can lead to better

overlay flow performance in the presence of traffic spikes causing congestion events.

4 Simulation study

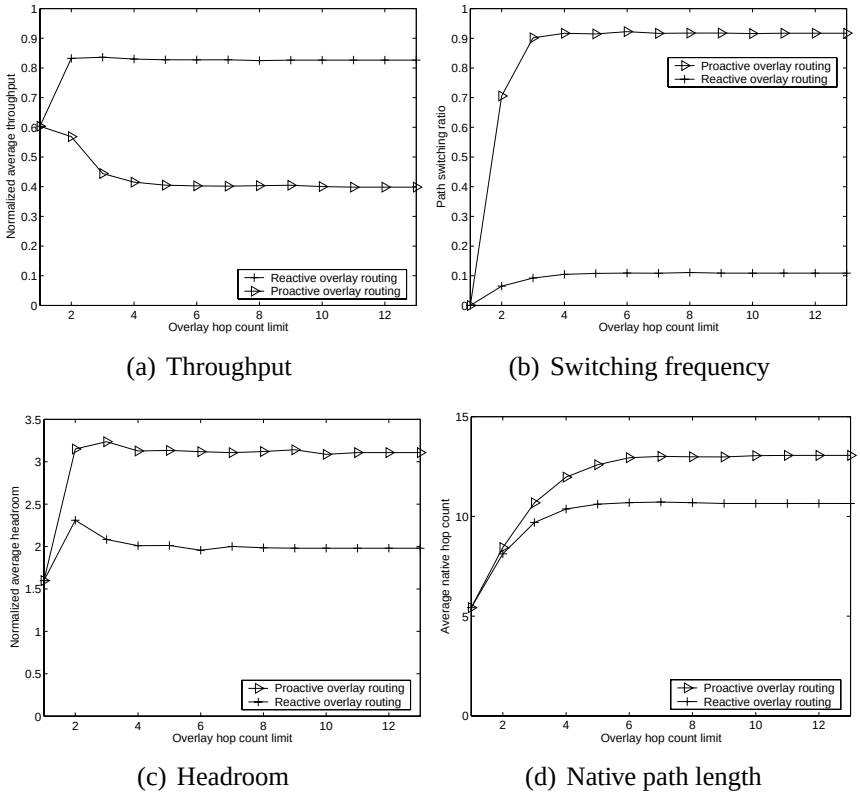
In this section, we first evaluate and compare the efficiency, stability, and safety margin of proactive and reactive overlay routing under various network conditions. Based on the results of this comparison, we propose a hybrid algorithm that combines the best features of reactive and proactive routing. Finally, we examine the effect of several important factors on the performance of the hybrid algorithm.

4.1 Maximum overlay hop count

A major advantage of overlay routing is its ability to utilize several alternate paths instead of the single path that is provided by IP routing. The number of such alternate paths increases with the number of overlay nodes an end-to-end path can traverse. We refer to the *overlay hop count* as the number of hops (or overlay links) that an end-to-end path traverses. In practice, the overlay hop count would be bounded by a maximum value H_{max}. The practical necessity for this limit is related to source routing: the intermediate overlay nodes need to be encoded in the header of each packet, and there is a limited number of bits for doing so. H_{max}=1 means that the overlay path is the same with the native-layer path, while H_{max}=2 means that the overlay path can traverse at most one intermediate overlay node.

In this simulation, we increase the maximum overlay hop count H_{max} from 1 to 13, and compare the performance of reactive and proactive overlay routing. The performance of native routing is also shown, as $H_{max} = 1$. Figure 5(a) shows that the average throughput T of reactive routing improves significantly when we increase H_{max} from one to two hops. The increase for larger values of H_{max} is negligible, meaning that longer overlay paths are rarely needed to avoid congestion. This shows that using a single intermediate overlay node with reactive routing is sufficient to obtain most throughput gain compared to native routing, and that this gain can be substantial. On the other hand, proactive routing performs worse as we increase H_{max}. One reason for this behavior is shown in Figure 5(d), which shows the *average native hop count* as a function of H_{max}.[3] As we would expect, the chosen paths in the native network

[3] Another major reason is given in Section 4.2.



(a) Throughput (b) Switching frequency

(c) Headroom (d) Native path length

Figure 5. Effect of maximum overlay hop count H_{max}.

tend to be longer as we increase H_{max}. Also, the paths used by proactive routing are significantly longer than the paths used by reactive routing, because the former always attempts to choose the path with the maximum headroom. As a result, the proactive algorithm uses more network resources than the reactive algorithm, decreasing the network's avail-bw and causing a lower value of T.

In terms of stability, increasing H_{max} causes the following two effects: 1) more alternate paths are considered by each flow and so there is a higher frequency of path switching, and 2) more native links are affected by the previous path changes, causing further variations in the avail-bw distribution and triggering even more path switching. Indeed, as Figure 5(b) shows, a higher value of H_{max} causes more frequent path switching. Note that proactive routing experiences significant instability, while reactive routing maintains a low path switching ratio across the range of of H_{max}.

Although proactive routing performs worse than reactive in terms of efficiency and stability, it does have the advantage of providing overlay flows with a higher average headroom, as shown in Figure 5(c).

The increased headroom can act as a wider safety margin in the presence of traffic fluctuations, as will be shown in Section 4.4. Note that the maximum headroom is obtained (by both algorithms) when H_{max}=2; longer overlay paths can cause larger consumption of network capacity.

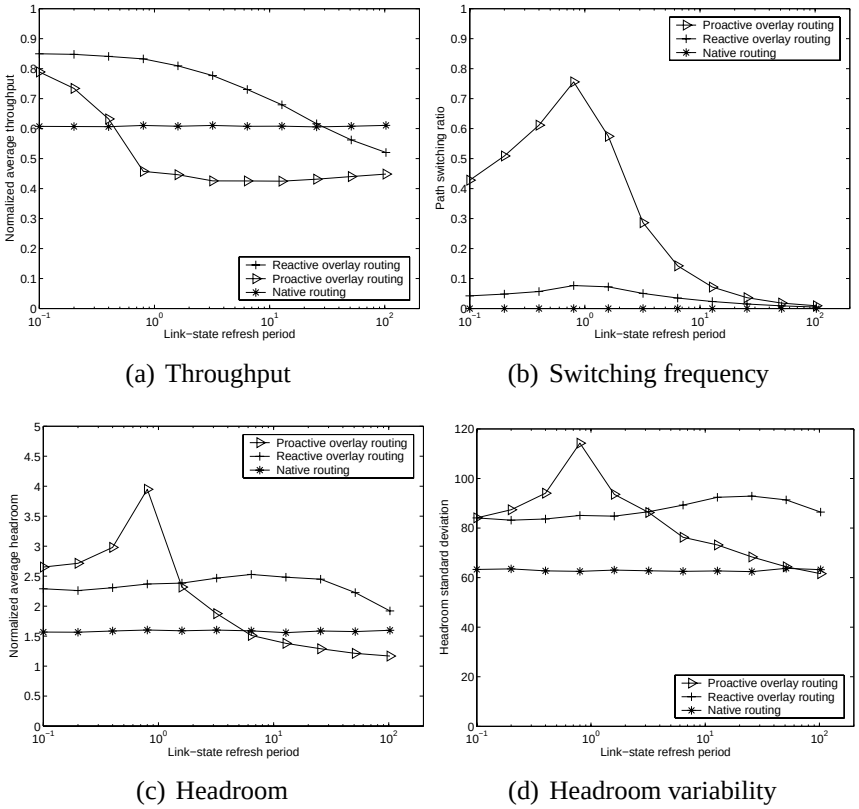
The previous results show that with at most one intermediate overlay node, reactive overlay routing can achieve significantly improved efficiency and headroom over native routing and maintain good stability. For proactive routing, limiting the maximum overlay hop count to two is even more critical in terms of efficiency and stability. Consequently, in the rest of the paper we will set H_{max}=2 for both algorithms. The practical implication of this limit is that a single node identifier in the packet header would be enough to encode the overlay route.

4.2 Link-state refresh period

Recall that the link-state refresh period P_r is the time length between successive updates of the overlay avail-bw link-state. A higher value of P_r increases the staleness of overlay routing information, but also decreases the link-state dissemination overhead.

Figure 6 shows the performance of proactive, reactive, and native routing as we vary P_r from 100msec to 100sec. Note that P_r cannot be lower than the measurement delay D_m, which is set to 100msec in our simulations. Even though P_r would not be more than a few seconds in practice, we simulate a wider range for illustration purposes.

In terms of average throughput, Figure 6(a) shows that, as we would expect, the efficiency of both reactive and proactive routing drops as P_r increases. Interestingly, however, the reactive algorithm is much more robust to stale link-state than the proactive algorithm. The former can achieve better throughput than native routing as long as P_r is less than about 10 seconds, while the latter does worse than native routing if P_r exceeds 400msec. The reason for this major difference between reactive and proactive routing is that the latter relies much more heavily on avail-bw information, because it considers switching even the satisfied flows. Consequently, a higher value of P_r, with its increased link-state staleness, causes a dramatic throughput loss for the proactive algorithm. The corresponding throughput loss for the reactive algorithm is negligible when P_r is between 100ms-1000ms, which is probably a



(a) Throughput (b) Switching frequency

(c) Headroom (d) Headroom variability

Figure 6. Effect of link-state refresh period P_r.

reasonable range for P_r in terms of dissemination overhead.

In terms of stability, Figure 6(b) shows that the path switching ratio curves can be split into two regions. S increases as P_r approaches 1sec, which is the flow update period P_u. Then, for longer values of P_r, S decreases. The reason for this behavior is as follows. As P_r increases from 100ms to 1sec, the link-state becomes increasingly more stale. This increasing staleness means that a higher fraction of the path switching decisions were based on inaccurate information and they were incorrect (meaning that the flow did not move to the maximum headroom path or it did not avoid congestion). Consequently, further path switching is required to correct the previous routing decisions. On the other hand, when P_r becomes larger than P_u (1 second) the link-state information changes less often than the frequency with which overlay flows examine whether to switch paths. Consequently, many of the path update events see the same link-state information, and so the corresponding flows decide that they should stay at the same path. This tends to decrease the metric S as P_r increases.

Figure 6(c) shows the average headroom curves for different routing algorithms. Comparing with the average throughput curves of Figure 6(a), note that larger average headroom does not necessarily mean higher average throughput. Also, the average headroom for the proactive algorithm is maximized not with the most uptodate link-state information (i.e., P_r=100msec), but when $P_r \approx P_u$=1sec. The reason for this counterintuitive behavior is related to the variability of the headroom. Figure 6(d) shows the standard deviation of the distribution of per-flow headroom samples. In the proactive algorithm, the variability of the headroom increases with P_r until $P_r \approx P_u$, and then decreases, following the same trend with the path switching frequency S. The reason is that more frequent path switching causes a wider dispersion of the headroom that each flow observes. The increased variability generates high headroom values with larger probability, increasing the average headroom H.[4] Similarly, in the reactive algorithm the average headroom follows the same trend with the headroom standard deviation.

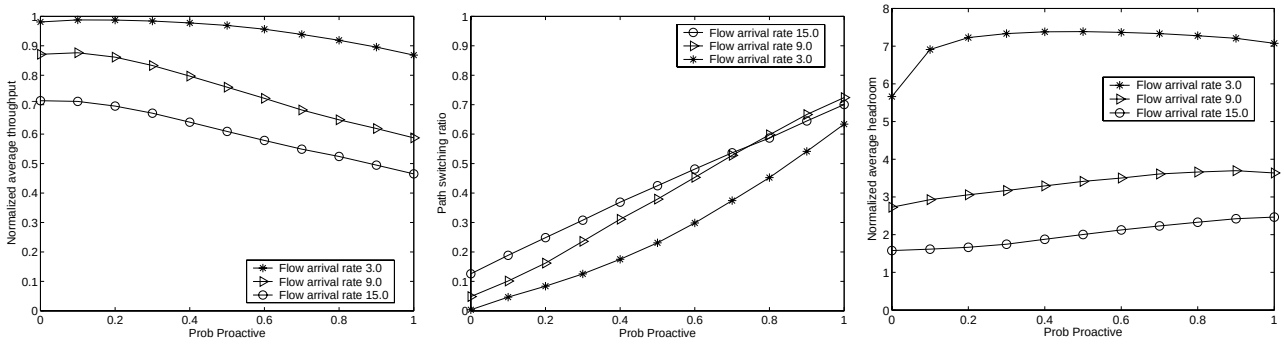
A related observation in Figure 6(d) is that the reactive algorithm has the additional benefit, over the proactive algorithm, that it results in lower headroom variability in the range $P_r < P_u$=1sec, which is probably the most practical range for the selection of P_r. With the proactive algorithm, flows switch paths much more frequently, causing significant network load variations and headroom variability.

4.3 Hybrid routing and probability of proactive switching

The previous simulation results showed that the proactive algorithm performs worse than the reactive algorithm in terms of throughput and stability. Furthermore, those results showed two major reasons for the difference between the two algorithms: first, proactive routing tends to use longer native paths and thus consumes more network resources (especially when $H_{max} > 2$), and second, proactive routing is much more sensitive to stale link-state information. On the other hand, proactive routing performs better than reactive in terms of average headroom (when P_r is less than P_u), providing overlay flows with a wider safety margin. Section 4.4. shows that this wider safety margin can improve the performance of overlay flows in the presence of random congestion events.

[4]Lower values, on the other hand, have a weaker impact on the average headroom H because they are always positive and relatively small.

Given the previous trade-off, we propose a simple heuristic that combines the previous two algorithms in a probabilistic manner. We refer to this algorithm as *hybrid routing*. At each path update event, the hybrid algorithm performs proactive path switching with a probability p_p (referred to as *probability of proactive switching*); otherwise, the algorithm performs reactive path switching. The intuition behind this algorithm is to maintain the good throughput and stability properties of reactive routing (i.e., to use a low p_p), but at the same time to occasionally switch the flow to a path with higher headroom, even if it is satisfied, in order to improve its safety margin.



(a) Throughput (b) Switching frequency (c) Headroom

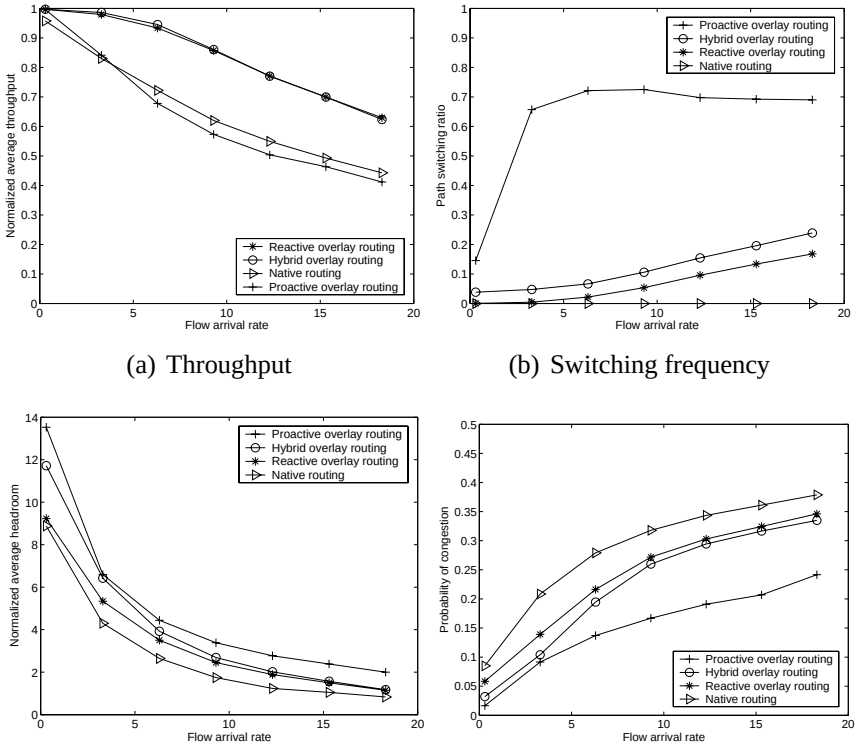
Figure 7. Effect of probability of proactive switching p_p.

Figure 7 shows the performance of hybrid routing for different values of p_p. The three curves in each graph represent cases of low (F_a=3.0), medium (F_a=9.0), and high (F_a=15.0) overlay traffic load. Note that p_p=0 corresponds to reactive routing and p_p=1 corresponds to proactive routing. In terms of efficiency, p_p=0.1 is almost as good as pure reactive routing. In terms of stability, the smaller p_p the lower S will be. The difference between p_p=0.1 and p_p=1 is substantial, however, and so the former value gives a major stability gain compared to proactive routing. In terms of headroom, on the other hand, the choice of p_p makes a significant difference only in low load conditions; in heavy load conditions the headroom is quite limited even with the proactive algorithm. In low load conditions, p_p=0.1 gives us most of the headroom gain of proactive routing.

To summarize, the simulation results indicate that the probability of proactive switching p_p can be close to 10%, resulting in almost the same efficiency and stability with the reactive algorithm, but also increased headroom especially in lower load conditions. In the rest of the paper, we set p_p=0.1.

4.4 Traffic load and flow arrival rate

The overlay traffic load is determined by the flow arrival rate, flow duration and max-rate limit. Increasing one of these parameters while keeping the others fixed increases the offered traffic load. In this experiment, we vary the flow arrival rate F_a and compare the performance of proactive, reactive, and hybrid overlay routing, as well as native shortest-path routing, under increasing load. These simulations provide further evidence that the hybrid algorithm combines the best features of reactive and proactive routing.



(a) Throughput (b) Switching frequency

(c) Headroom (d) Probability of congestion

Figure 8. Effect of overlay traffic load F_a.

Figure 8(a) shows the normalized throughput for the four routing algorithms. Under very light load, where native links are rarely saturated, both native routing and overlay routing satisfy almost all flows. In higher loads, reactive routing manages to avoid saturated paths, as long as there is at least one non-congested path. Under heavy-load conditions, it still performs much better than native routing. The hybrid algorithm offers essentially the same throughput as reactive routing. Proactive routing, on the

other hand, performs even worse than native routing, mostly due to its high sensitivity on stale link-state information.

In terms of stability, Figure 8(b) shows that the path switching ratio of reactive routing remains almost zero at low traffic loads and increases slowly as more flows become unsatisfied in heavy loads. Proactive routing, on the other hand, introduces significant instability, with about 70% of the flow update events causing path switching, even before the network becomes congested. The path switching ratio with hybrid routing is slightly higher than with reactive routing, but still much lower than proactive routing.

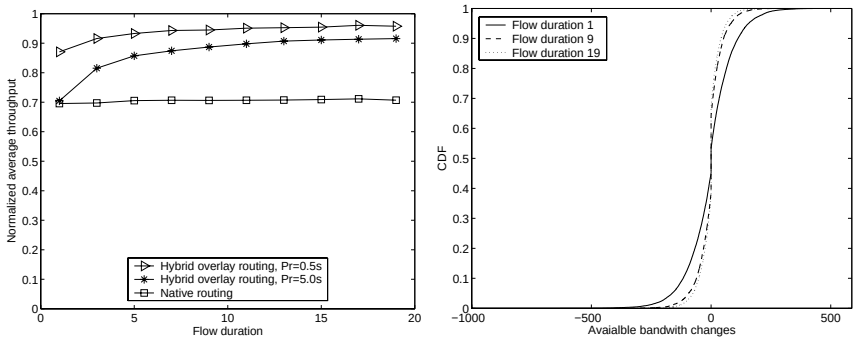
In terms of average headroom, Figure 8(c) shows that proactive routing achieves consistently higher average headroom than both reactive routing and native routing. The hybrid algorithm provides almost the same headroom with proactive routing in lower load conditions, when there is still significant headroom. Recall that the intuition behind the headroom metric is to quantify the safety margin that overlay flows have in the presence of random traffic variations. To understand why larger headroom can lead to better performance for overlay flows, we examine the frequency with which "traffic spikes" can lead to saturated links with each routing algorithm. Specifically, we generate instantaneous traffic spikes based on a Poisson process. At each event of the Poisson process, we randomly select a native link of an overlay flow and examine whether a traffic spike with magnitude 10% of the existing traffic on the link would cause congestion (*i.e.,* the total offered load on the link exceeds the capacity). If that is the case, we count that event as a "congestion incident". The process is repeated for every overlay flow at each Poisson event. Figure 8(c) shows the probability of congestion incidents for the four different routing algorithms, as a function of the flow arrival rate F_a. Note that the probability of congestion is significantly lower with the proactive algorithm, as a result of the higher headroom that that algorithm provides. The reactive and the hybrid algorithms have a higher likelihood of encountering congestion, especially in heavy load conditions, but they still do clearly better than native routing.

4.5 Effect of traffic variability

The performance of overlay routing depends on the variability of the underlying traffic, because the best path for the next update period is predicted based on the load of each path at the end of the last

update period. Under the same aggregate load, we would expect that higher traffic variability will lead to worse overlay routing performance in terms of all three performance metrics. In this section, we examine two factors that affect the variability of the underlying traffic: the overlay flow duration F_d and the cross traffic variation period F_c.

Let us first focus on the average overlay flow duration F_d. In this simulation, we increase F_d from 1 to 19 seconds and reduce the flow arrival rate F_a proportionally so that the aggregate offered load remains constant. We also remove the non-overlay cross traffic, so that we focus exclusively on the variability introduced by overlay traffic. Figure 9(a) shows the throughput of hybrid routing under two refreshing periods: P_r=0.5sec and P_r=5.0sec; the latter represents a rather extreme case of stale link-state. For reference, we also show the results with native shortest-path routing.



(a) Throughput

(b) CDF of avail-bw variations during 0.5s intervals

Figure 9. Effect of overlay flow duration F_d.

Since the offered load remains constant, the native routing can achieve the same throughput independent of the flow duration. In contrast, the throughput of hybrid routing increases with F_d. For short flows, more flow arrival/departure events take place within each measurement/refresh period, making the overlay traffic more variable and causing greater staleness in the link-state routing information. As flows become longer, the throughput increases because the resulting traffic variability decreases, and so even outdated link-state information is reasonably accurate. To further illustrate this point, Figure 9(b) shows the cumulative distribution functions (CDF) of the avail-bw variations across successive 0.5 second intervals for three different flow durations. Notice that shorter flows cause more significant traffic

variability than longer flows.

Another observation from Figure 9(a) is that when the flow duration decreases, hybrid overlay routing needs a shorter refresh period P_r to maintain the same throughput. More generally, a shorter refresh period is required when the traffic variability increases. In selecting P_r, the objective should be that the refresh period is short enough so that the avail-bw does not change significantly during successive link-state updates. Similar observations hold for the path switching ratio and the average headroom (not shown here).

Let us now focus on the variability of the non-overlay cross traffic. As mentioned in Section 3, the cross traffic has an average rate variation period F_c=20sec. In this simulation experiment, we vary F_c from 0.01sec to 100sec. Obviously a higher value of F_c would make the cross traffic less variable.

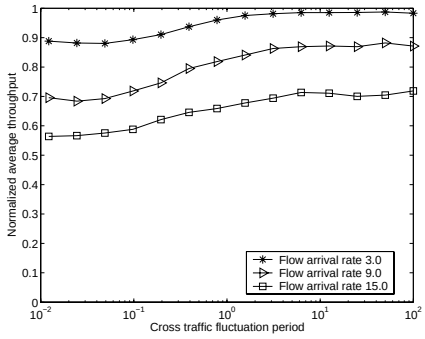


Figure 10. Throughput.

Figure 10 shows the average throughput with hybrid overlay routing under three load conditions. Note that the average throughput is not so sensitive to F_c, as long as the latter is larger than the flow update period P_u (1sec). This is because when $P_u < F_c$, the avail-bw of overlay paths does not change significantly between successive path update events. With more frequent cross traffic variations, there is a reduction in the average throughput. The reduction is not major with the hybrid (or the reactive) algorithm however, because the latter is quite robust to stale link-state. Similar observations hold for the path switching ratio and the average headroom (not shown here).

4.6 Measurement errors

The literature on avail-bw measurement techniques reports estimation errors that vary from $\pm$10% to $\pm$30% [25, 26, 27, 28, 29]. It seems unlikely at this point that these measurement techniques can be improved so dramatically that the estimation error will become almost zero. Since the overlay routing algorithms that we study rely on avail-bw estimation, we need to examine the effect of avail-bw measurement errors on the performance of overlay routing.

Let us denote the real avail-bw value at an overlay link by v, the measured value by m, and the error factor by $e \in [0, 1]$. We consider the following three error models:

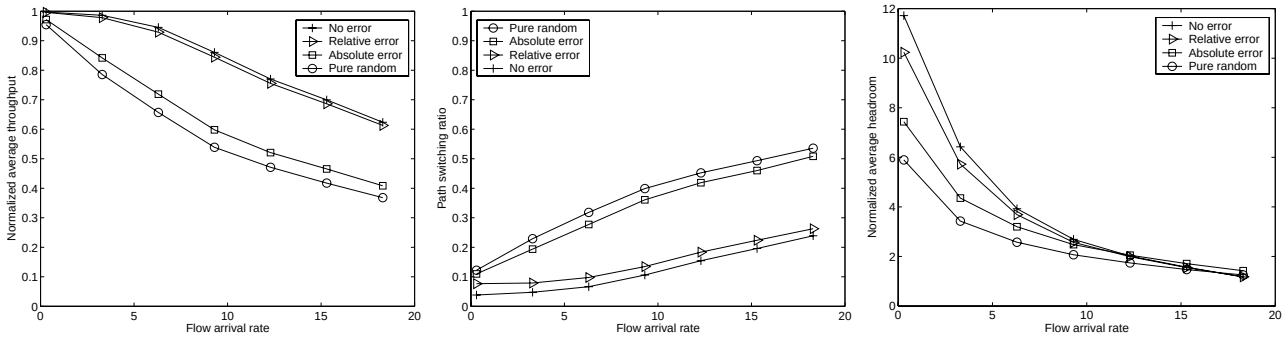
Relative error: The measurement m is symmetrically and uniformly distributed in a range around v, with the error being proportional to v. That is, $m \in [(1 - e) \cdot v, (1 + e) \cdot v]$

Absolute error: The measurement m is symmetrically and uniformly distributed in a range around v, with the error being proportional to the overlay link capacity C. That is, $m \in [\max(0, v - e \cdot C), v + e \cdot C]$.

Random estimate: The measurement m varies randomly anywhere between zero and the overlay link capacity C. C represents here an upper bound on the avail-bw of that link. In other words, instead of actually measuring avail-bw, the routing algorithms use a random estimate of the avail-bw in each overlay link, limited by the corresponding link capacity.

We note that the avail-bw estimation literature reports that the measurement errors typically follow the relative error model. The reason we examine the absolute error and random estimate models is to show that different error models can affect overlay routing in very different ways.

Figure 11(a) compares the throughput of hybrid overlay routing with no error, 100% relative error ($e = 1$), 100% absolute error, and random estimate. Surprisingly, we observe that *relative avail-bw estimation errors, even up to 100%, have very small impact on the performance of overlay routing.* On the other hand, absolute errors or random estimates cause a significant drop in the resulting average throughput. Similar trends appear in the stability and headroom results (see Figures 11(b) and 11(c)).



(a) Throughput (b) Switching frequency (c) Headroom

Figure 11. Effect of measurement error.

What is the reason that even large relative errors have no significant impact on the efficiency of overlay routing? The answer is actually simple if we consider that overlay routing decisions do not depend on the actual (absolute) avail-bw, but on the ranking of different paths in terms of avail-bw. After analyzing several simulations, we found that an overlay flow can be found in one of two main conditions. First, one of the paths that the flow can choose from has significantly higher avail-bw than the rest. Second, several of the paths that the flow can choose from have similar avail-bw. In the former, a relative error would affect all candidate paths, but the path with the maximum avail-bw would still be reported as the best with high probability. For example, if two paths have avail-bw 1000 and 100, a relative error of $\pm 100\%$ would result in a uniformly distributed value within [0,2000] and [0,200], respectively; the probability that the first path would be reported as better than the second is 95%. In the latter, the measurement error can affect the selection of the best path, but because the flow chooses among paths with similar avail-bw values the path selection does not matter significantly. Note that the previous reasoning would not hold true in the case of absolute errors or random estimate.

In summary, the previous experiments showed that relative errors in the avail-bw estimation process (which are common and probably unavoidable) will not have significant impact on the performance of hybrid and reactive overlay routing. This implies that even relatively simple measurement techniques, which produce a “ballpark” estimate of the avail-bw, may be useful in overlay routing applications.

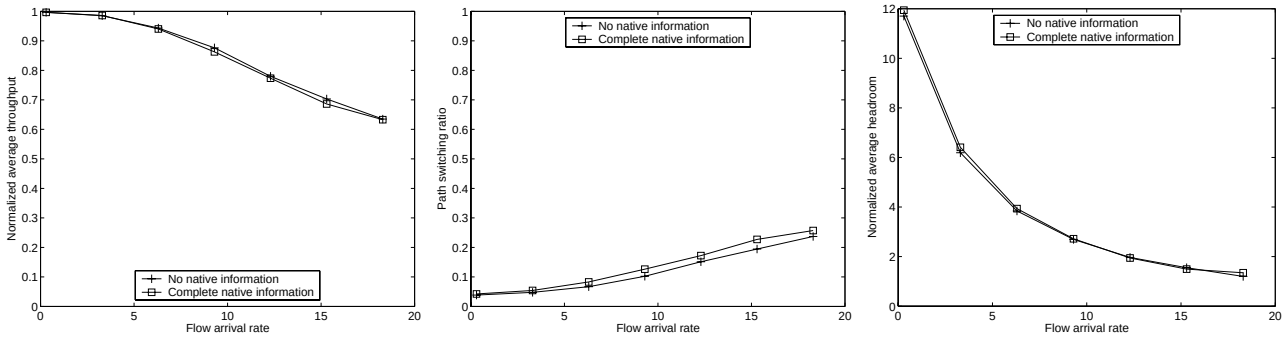
4.7 Native layer link sharing

Two overlay links (or paths) may share one or more native links. Such "native sharing" effects may not be visible to the overlay network, which typically has information only for the overlay links and their avail-bw. Furthermore, as will be shown next, native sharing can cause errors in the selection of the maximum headroom path.

To see the effect of native sharing on overlay routing, we compare the following two different models, in terms of the amount of knowledge the overlay network has about the native network:

- *No-information:* The overlay network has absolutely no knowledge about the native topology or about the avail-bw of native links. This implies that when native sharing occurs, the overlay nodes may judge incorrectly which path provides the maximum headroom. On the other hand, this mode of operation would be the simplest to implement, as it does not require any sophisticated probing techniques for the detection of native sharing.

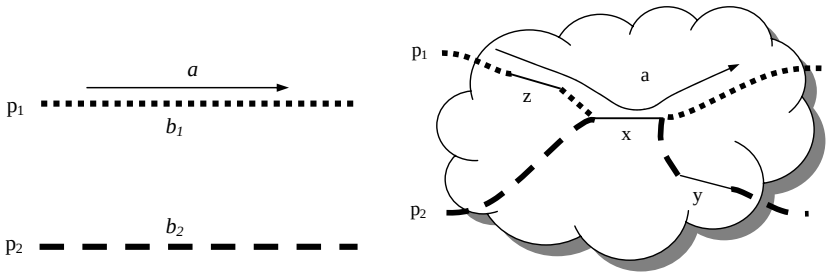
- *Complete-information:* This is an ideal case, in which the overlay nodes have accurate information about both the native layer topology and the avail-bw of each native link. This information would enable an overlay node to correctly determine the maximum headroom path even when native sharing takes place. Note that, at least so far, there are no measurement techniques that can estimate the avail-bw of each native link in a network path.



(a) Throughput (b) Switching frequency (c) Headroom

Figure 12. Hybrid routing performance with and without information about the native network.

Figure 12 shows the performance of hybrid overlay routing, as a function of the offered load, with the previous two modes of operation. Perhaps surprisingly, notice that the "no-information" mode performs almost identically to the "complete-information" mode. This means that overlay routing can be equally effective even if it ignores the subtle effects of native sharing. What is the reason for this counter-intuitive effect?



(a) Overlay topology (b) Native sharing

Figure 13. Native sharing between two overlay paths.

To explain, we will use the following model. Suppose that we have two overlay paths p_1 and p_2, as shown in Figure 13(a). The flow under consideration f is currently routed on p_1, and its throughput is a. The avail-bw in the two paths is b_1 and b_2, respectively.

Ignoring any potential native sharing ("no-information" model), the overlay routing algorithm will estimate the headroom of the two paths as follows:

$$h'(f, p_1) = b_1 + a \quad h'(f, p_2) = b_2$$

In the "complete-information" model, the headroom of the two paths will be denoted by $h''(f, p_1)$ and $h''(f, p_2)$, respectively. Based on the type of native sharing between p_1 and p_2, we have the following three cases:

1. The paths p_1 and p_2 do not share any native link: In this case, $h''(f, p_1) = b_1 + a$, and $h''(f, p_2) = b_2$. Therefore, the complete-information model provides the same headroom estimates as the no-information model, and so there is no difference in the performance of overlay routing between the two models.

2. The paths p_1 and p_2 share a native link x, but x is not the bottleneck of p_2, *i.e.*, the avail-bw in p_2 is limited by a native link y and it is not affected by flow f. In this case, we still have that $h''(f, p_1) = b_1 + a$, and $h''(f, p_2) = b_2$. Again, the two native information models would lead to the same routing decision.

3. The paths p_1 and p_2 share a native link x, and x is the bottleneck of path p_2 (shown in Figure 13(b)). This case can be further analyzed as two different sub-cases.

   (a) x is also the bottleneck of p_1. Then, $h''(f, p_1) = b_x + a$ and $h''(f, p_2) = \min(b_y, b_x + a)$, where y is the native link with the second lowest available bandwidth along path p_2, and b_x and b_y is the avail-bw in links x and y respectively. Therefore, $h''(f, p_1) \geq h''(f, p_2)$. In this case, the complete-information model would report that the maximum headroom path is p_1. Notice however that this is also what the no-information model would report, because $h'(f, p_1) = b_x + a > h'(f, p_2) = b_x$. Therefore, the overlay routing algorithm would choose the same path with both native information models.

   (b) x is not the bottleneck of p_1. Then, $h''(f, p_1) = b_z + a$ and $h''(f, p_2) = \min(b_y, b_x + a)$, where z is the bottleneck of p_1. In the no-information model, the headroom of each path would be estimated as: $h'(f, p_1) = b_z + a$ and $h'(f, p_2) = b_x$. In this case, it is easy to verify that the no-information and complete-information models would disagree in their path selection only if the following inequality holds:

$$b_y > b_z + a > b_x > b_z \tag{6}$$

   In this case, the complete-information model would cause a path change, while the no-information model would not.

The previous analysis shows that a naive overlay routing algorithm that ignores any native sharing effects would incorrectly choose the current path over another path with larger avail-bw only in a very specific scenario of native sharing. In that scenario, the two paths share at least one link x which is the bottleneck

of p_2 but not of p_1, and the avail-bw of the three involved links x, y, and z satisfy (6). Apparently, this scenario does not happen often in our simulations, which explains why the simulation results for the no-information and complete-information models are so close. Note that the slightly higher path switching ratio of the complete-information model in Figure 12(b) is due to the few cases in which (6) is true.

In summary, native sharing effects can affect, in principle, the performance of overlay routing. Fortunately, however, an overlay routing algorithm that ignores native sharing would rarely choose a different path even if it had complete information about the native topology and the avail-bw of each native link.

5 Conclusions

This paper presented a simulation study of dynamic overlay routing. Given that most previous work focused on delay-driven path selection, we focused instead on avail-bw based overlay routing algorithms leveraging the recently developed measurements techniques for end-to-end avail-bw. We considered two main approaches on overlay routing, proactive and reactive, as well as a number of factors that can affect the performance of these routing algorithms.

The main conclusions of this study follow:

- Reactive overlay routing performs better in terms of efficiency than native or proactive overlay routing. The efficiency gain compared to native routing can be substantial, especially if the network is not very lightly loaded. Also, reactive routing is much much stable than proactive routing.

- Proactive overlay routing performs better in terms of headroom (safety margin) than native and reactive overlay routing.

- A single intermediate overlay node is sufficient for reactive routing to achieve its throughput and headroom gain over native routing. For proactive routing, limiting the maximum overlay hop count H_{max} to 2 is even more critical in terms of efficiency and stability.

- The reactive algorithm is quite robust to stale link-state information, and it performs better than native routing even when the link-state refresh period P_r is a few seconds. The proactive algorithm,

on the other hand, is very sensitive to link-state staleness, and P_r should be as short as possible.

- A hybrid algorithm that acts reactively in about 90% of the time and proactively in about 10% of the time, can achieve a good compromise between high throughput, stability and safety margin, combining the best features of reactive and proactive routing.

- Overlay routing performs better with longer overlay flows, because the latter create lower traffic variability. Cross traffic variations can also decrease the performance of overlay routing, especially when these variations are significant in lower time scales than the path update period P_u.

- Relative errors in the avail-bw estimation process (which are common and probably unavoidable) will have negligible impact on the efficiency of hybrid overlay routing. Absolute or random errors, on the other hand, can have a significant impact.

- Even though native sharing effects can affect the performance of hybrid overlay routing, ignoring native sharing performs almost equally well with having complete information about the native network.

Finally, in this work we chose to ignore bandwidth sharing issues in congested links and congestion responsive transport mechanisms. Congestion control adds a feedback loop between the overlay nodes and the network that may interact with the overlay routing feedback loop, causing effects that are currently largely unexplored [33]. These interactions will be the subject of future work.

References

[1] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris. Resilient overlay networks. In *Proc. ACM SOSP*, pages 131–145, 2001.

[2] J. Touch. Dynamic Internet overlay deployment and management using the X-Bone. *Computer Networks*, 36(2-3):117–135, 2001.

[3] H. Eriksson. MBONE: the multicast backbone. *Communications of the ACM*, 37(8):54–60, 1994.

[4] J. Byers, J. Considine, M. Mitzenmacher, and S. Rost. Informed content delivery across adaptive overlay networks. In *Proc. ACM SIGCOMM*, pages 47–60, August 2002.

[5] J. Jannotti, D. K. Gifford, K. L. Johnson, M. F. Kaashoek, and J. W. O'Toole, Jr. Overcast: Reliable multicasting with an overlay network. In *Proc. Symposium on Operating System Design and Implementation (OSDI)*, pages 197–212, October 2000.

[6] L. Subramanian, I. Stoica, H. Balakrishnan, and R. Katz. OverQoS: Offering QoS using overlays. In *First Workshop on Hop Topics in Networks (HotNets-I)*, pages 11–16, October 2002.

[7] S. Tao, K. Xu, Y. Xu, T. Fei, L. Gao, R. Guerin, J. Kurose, D. Towsley, and Z.-L. Zhang. Exploring the performance benefits of end-to-end path switching. In *Proc. IEEE ICNP*, pages 304–315, 2004.

[8] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian. Delayed Internet routing convergence. In *Proc. ACM SIGCOMM*, pages 175–187, 2000.

[9] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. E. Anderson. The end-to-end effects of Internet path selection. In *Proc. ACM SIGCOMM*, pages 289–299, 1999.

[10] D. G. Andersen, A. C. Snoeren, and H. Balakrishnan. Best-path vs. multi-path overlay routing. In *Proc. ACM SIGCOMM conference on Internet measurement*, pages 91–100. ACM Press, 2003.

[11] N. Feamster, D. G. Andersen, H. Balakrishnan, and M. F. Kaashoek. Measuring the effects of Internet path faults on reactive routing. In *Proc. ACM SIGMETRICS*, pages 126–137. ACM Press, 2003.

[12] Y. hua Chu, S. G. Rao, and H. Zhang. A case for end system multicast. In *Proc. ACM SIGMETRICS*, pages 1–12. ACM Press, 2000.

[13] Z. Li and P. Mohapatra. QRON: QoS-aware routing in overlay networks. *IEEE Journal of Selected Areas in Communications*, 22(1):29–40, 2004.

[14] A. D. Keromytis, V. Misra, and D. Rubenstein. SOS: Secure overlay services. In *Proc. ACM SIGCOMM*, pages 61–72, 2002.

[15] Z. Li and P. Mohapatra. The impact of topology on overlay routing service. In *Proc. IEEE INFOCOM*, pages 408–418, 2004.

[16] A. Nakao, L. Peterson, and A. Bavier. A routing underlay for overlay networks. In *Proc. ACM SIGCOMM*, pages 11–18. ACM Press, 2003.

[17] Y. Chen, D. Bindel, H. Song, and R. H. Katz. An algebraic approach to practical and scalable overlay network monitoring. In *Proc. ACM SIGCOMM*, pages 55–66. ACM Press, 2004.

[18] A. Akella, B. Maggs, S. Seshan, A. Shaikh, and R. Sitaraman. A measurement-based analysis of multihoming. In *Proc. ACM SIGCOMM*, pages 353–364. ACM Press, 2003.

[19] A. Akella, J. Pang, B. Maggs, S. Seshan, and A. Shaikh. A comparison of overlay routing and multihoming route control. In *Proc. ACM SIGCOMM*, pages 93–106. ACM Press, 2004.

[20] T. Roughgarden and E. Tardos. How bad is selfish routing? *Journal of ACM*, 49(2):236–259, 2002.

[21] A. Akella, S. Chawla, and S. Seshan. Mechanisms for Internet routing: A study. Technical Report CMU-CS-02-163, Carnegie Mellon University, 2002.

[22] L. Qiu, Y. R. Yang, Y. Zhang, and S. Shenker. On selfish routing in Internet-like environments. In *Proc. ACM SIGCOMM*, pages 151–162, 2003.

[23] M. Seshadri and R. H. Katz. Dynamics of simultaneous overlay network routing. Technical Report UCB//CSD-03-1291, University of California, Berkeley, 2003.

[24] H. Xie, L. Qiu, Y. R. Yang, and Y. Zhang. On self adaptive routing in dynamic environments. In *Proc. IEEE ICNP*, pages 12–23, 2004.

[25] M. Jain and C. Dovrolis. End-to-end available bandwidth: measurement methodology, dynamics, and relation with tcp throughput. *IEEE/ACM Transaction on Networking*, 11(4):537–549, 2003.

[26] N. Hu and P. Steenkiste. Evaluation and characterization of available bandwidth probing techniques. *IEEE Journal of Selected Areas in Communications*, 21(6):879–894, 2003.

[27] V. J. Ribeiro, R. H. Riedi, R. G. Baraniuk, J. Navratil, and L. Cottrell. pathChirp: Efficient available bandwidth estimation for network paths. In *Proc. Passive and Active Measurements (PAM) workshop*, pages 13–24, 2003.

[28] J. Strauss, D. Katabi, and F. Kaashoek. A measurement study of available bandwidth estimation tools. In *Proc. ACM SIGCOMM Conference on Internet Measurement*, pages 39–44, 2003.

[29] A. Akella, S. Seshan, and A. Shaikh. An empirical evaluation of wide-area Internet bottlenecks. In *Proc. ACM SIGCOMM Conference on Internet Measurement*, pages 101–114, 2003.

[30] Z. Wang and J. Crowcroft. Quality-of-service routing for supporting multimedia applications. *IEEE Journal of Selected Areas in Communications*, 14(7):1228–1234, 1996.

[31] N. Spring, R. Mahajan, and D. Wetherall. Measuring isp topologies with rocketfuel. In *SIGCOMM '02: Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 133–145, New York, NY, USA, 2002. ACM Press.

[32] V. Paxson and S.Floyd. Wide Area Traffic: The Failure of Poisson Modeling. *IEEE/ACM Transactions on Networking*, 3(3):226–244, June 1995.

[33] E. J. Anderson and T. E. Anderson. On the stability of adaptive routing in the presence of congestion control. In *Proc. IEEE INFOCOM*, pages 948–958, 2003.