# Routing Stability in Congested Networks: Experimentation and Analysis

Aman Shaikh
Anujan Varma
Computer Engineering Department
University of California
Santa Cruz, CA 95064
{aman,varma}@cse.ucsc.edu

Lampros Kalampoukas
Rohit Dube
High Speed Networks Research
Bell Laboratories
Holmdel, NJ 07733
lampros@ieee.org, rohitd@dnrc.bell-labs.com

## ABSTRACT
Loss of the routing protocol messages due to network congestion can cause peering session failures in routers, leading to route flaps and routing instabilities. We study the effects of traffic overload on routing protocols by quantifying the stability and robustness properties of two common Internet routing protocols, OSPF and BGP, when the routing control traffic is not isolated from data traffic. We develop analytical models to quantify the effect of congestion on the robustness of OSPF and BGP as a function of the traffic overload factor, queueing delays, and packet sizes. We perform extensive measurements in an experimental network of routers to validate the analytical results. Subsequently we use the analytical framework to investigate the effect of factors that are difficult to incorporate into an experimental setup, such as a wide range of link propagation delays and packet dropping policies. Our results show that increased queueing and propagation delays adversely affect BGP's resilience to congestion, in spite of its use of a reliable transport protocol. Our findings demonstrate the importance of selective treatment of routing protocol messages from other traffic, by using scheduling and utilizing buffer management policies in the routers, to achieve stable and robust network operation.

## 1. INTRODUCTION
Routing protocols used in the Internet today exchange various control packets to disseminate routing information and to determine the liveliness of peering sessions established between pairs of routers. These control packets usually share resources such as bandwidth and buffer space with data traffic, and therefore, are subject to loss. Congestion in the network can hinder the propagation of routing information or peering refresh requests if the routing protocol messages are not isolated from data traffic. Anecdotal evidence of congestion at public exchange points due to excessive data traffic, denial of service attacks (e.g., SMURF) and system misconfiguration (e.g., pointing defaults) has been reported [1]. The objective of this paper is to analyze and quantify the effect of congestion on the stability of two common Internet routing protocols — OSPF and BGP — when the routing protocol messages are not isolated from data traffic.

A number of previous studies have dealt with various aspects of routing behavior in the Internet [3, 7, 12, 13, 17]. These studies used experimentation to measure and document various aspects of the routing dynamics, and they identify several stability-related problems in today's Internet. Some of these studies have also tried to identify the root causes of these problems, attempted to analyze them, and suggested remedies for them. Chinoy [3] analyzed the dynamics of routing information by collecting traces of the routing traffic sent over the NSFNET backbone network for a 12-hour period. He found that most of the routing fluctuations in the NSFNET originated at the edge of the network, and had cycle intervals of a few minutes. Paxson [17] used the "traceroute" utility at 37 Internet sites to analyze the routing behavior for pathological conditions, routing stability and routing symmetry. Govindan and Reddy [7] used a year's worth of inter–domain routing traces collected in 1994–95 and analyzed the Internet inter–domain topology, its routing stability behavior, and the effect of growth on these two characteristics. One of the key findings of their study was that the stability behavior of Internet routes has degraded with growth. Labovitz et al. [12] collected data from the BGP routing messages generated by border routers at five of the Internet's public exchange points during a nine-month period in 1996. They found that the volume of BGP routing message was several orders of magnitude more than expected and a majority of the information was redundant and pathological. In a subsequent paper [13], they identified reasons behind many of those unexpected routing messages and described remedies for them. Labovitz et al. [11, 12] have also shown that significant correlation exists between network usage and instability observed for BGP. Varadhan et al. [21] showed that there are routing policies that can cause BGP routes to oscillate and never converge to a stable configuration. Govindan et al. [6] subsequently described an architecture for coordinating routing policies and avoiding routing instabilities that can arise due to conflicting policies. Their approach essentially involves a static analysis of policies to verify that they do not contain conflicts that can lead to oscillations in BGP. Griffin and Wilfong [8] explored the worst–case complexity of performing such a static analysis of BGP routing policies. They developed an abstract model of BGP and defined a number of conditions that can be checked to make sure that BGP routes do not oscillate. For each of these conditions, they showed that the complexity of statically checking it is either NP–complete or NP–hard. Recently, Savage et al. [19] found that in several cases an alternate routing path exists which is superior with respect to performance metrics like round–trip time, loss rate and bandwidth to the actual path taken by packets. This is not surprising since routing path selection in several deployed routing protocols is based on shortest-path-first (SPF) or nearest-exit computations that use constant performance metrics.

Most of the existing studies on routing behavior assume reliable and loss-free delivery of routing control messages. Losses of the routing protocol messages due to congestion or errors may result in peering session failures, which in turn may lead to route flaps and routing instabilities. In our study we consider two routing protocols, OSPF and BGP, because of their wide deployment and the important role they play in today's Internet infrastructure. We have performed extensive experimentation and developed analytical models to capture the stability and robustness properties of OSPF and BGP in congested network environments as a function of the traffic overload factor, queueing delays, and packet sizes. We then used our analytical framework to evaluate the effect of factors such as link propagation delays and packet dropping policies. In this paper, we present several graphs showing the dynamics of BGP for various round trip times (RTTs) which can be used as a guideline for traffic engineering. To our knowledge, this is is the first attempt to systematically analyze the dynamics of routing protocols and validate the results with extensive experiments.

The organization of the paper is as follows: in Section 2 we describe the network setup and the methodology used in our experiments. Section 3 introduces the analytical models that are subsequently used to evaluate the stability and robustness properties of OSPF and BGP. In particular, in Sections 3.1 and 3.2 we develop closed-form analytical expressions for the route flap and link recovery times for OSPF, whereas in Sections 3.3 and 3.4 we introduce an analytical framework to evaluate the same properties for BGP. The framework for BGP takes into account the dynamics of TCP that is used for reliable transmission of BGP messages. In Section 4 we discuss our experimental results and compare them to those obtained using the analysis. Finally, in Section 5 we provide a summary and suggest future directions for our work.

## 2. EXPERIMENTAL SETUP

The network configuration used in our experiments consists of three routers. The routers are interconnected using ATM OC-3c (155 Mbits/sec) links to form the topology shown in Figure 1. One CBR PVC (Permanent VC) is established between each pair of routers to transport data and routing traffic. The routers use the *Classical IP over ATM* [14] framework for transporting IP packets over an ATM network. The VCI/VPI assignments are also shown in Figure 1. The transmission rate of the VCs, in all the experiments, is set to 10 Mbits/sec.

Data traffic is generated by a traffic generator connected to router HR1 through the 10.4.4.0/24 Ethernet segment. A Smartbits SMB2000 traffic generator/analyzer system by Netcom Systems was used for traffic generation.The exact transmission rate is set so that the desired traffic overload level is achieved on the egress ATM VCs at router HR1. IP datagrams are transported over the ATM links using AAL5 encapsulation. The Ethernet generation and transmission process takes into account discrepancies due to the various overheads associated with Ethernet frames, AAL5 PDUs, and ATM cells in order to achieve the specified overload level. The destination IP address for all generated traffic is set to 192.168.64.1 which corresponds to the workstation shown at the right in Figure 1.

The use of three routers allows for two distinct forwarding paths to be formed for forwarding traffic originated by the Ethernet traffic generator (10.4.4.2) and destined to the 192.168.64.1 workstation. When multiple forwarding paths to a destination node exist, routing protocols select the one with the minimum cost. In our configuration under normal conditions, the routing protocols select the two-hop path (10.4.4.2 → HR1 → HR2 → 192.168.64.1) as the shortest path to forward the generated traffic. When a link failure is inferred by the routing protocols, traffic is diverted to and forwarded through the three-hop path (10.4.4.2 → HR1 → HR0 → HR2 → 192.168.64.1). Throughout this paper we will refer to the shortest path as the primary routing path, and to the alternate path as the secondary one. Similarly, we will refer to the egress link from router HR1 towards HR2 that belongs to the primary path as the primary link, and to the one that overlaps with the secondary path as the secondary link.

Some recently introduced commercial routers [10] employ powerful and flexible mechanisms for buffer management, queueing and scheduling. These include dynamic pushout, static buffer allocation, and intelligent scheduling mechanisms (priority-based and hierarchical). However, most of the deployed routers do not provide such a fine level of service differentiation, but instead make extensive use of the FIFO queueing and scheduling mechanisms. This does not provide any isolation or protection of routing protocol messages from data traffic. Since our interest in this paper is to study the behavior of routing protocols in congested network environments, the line interfaces in our experimental setup were configured to operate in a strict FIFO mode.

Route flaps (link-down events as seen and reported by the routing protocol), are induced by operating the line interfaces at a sustained overloaded state. In all our experiments, the links connecting the routers are always operational at the physical layer and continue forwarding packets that have already been queued awaiting transmission even after the routing protocol reports a link-down event. It is the routing protocols that infer a link failure due to losses experienced in the transmission of "keepalive" or "hello" messages that are intended to determine the state of the underlying transmission link. Subsequently, the routers withdraw

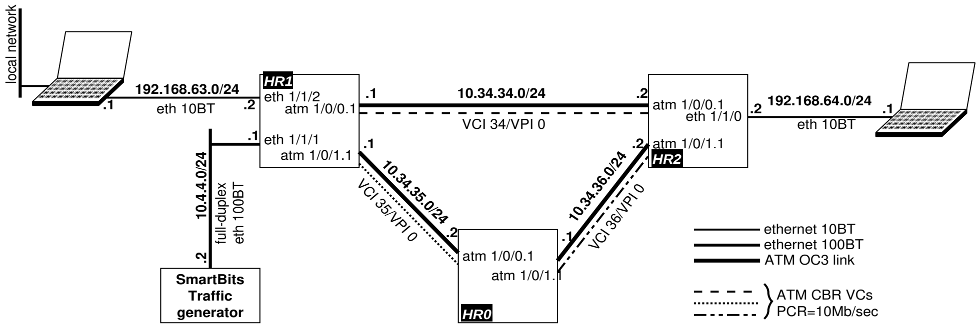



**Figure 1: Network topology used in the experiments.**

any routes in the forwarding table associated with the failed link. Any packets that cannot be forwarded as a result of the IP lookup failure are dropped. Since data packets and routing messages share a single queue, the probability for a packet/message loss to occur is approximately equal to the overload factor.

It is important to mention here that the ATM interfaces used in our experiments make use of the Drop-from-Front policy: the packet queued at the front of the queue is dropped when a packet arrives at a full queue to make space for the new packet. In our study however, the type of dropping policy used does not alter the fundamental routing protocol dynamics: with both Drop-from-Front and Drop-Tail the probability of dropping a packet is approximately equal to the overload factor. The basic difference between the two dropping policies, however, lies in the average queueing delay experienced by packets that are eventually forwarded: the Drop-from-Front policy leads to smaller queueing delays since packets queued in front of a given packet might eventually be dropped. Furthermore, for a given buffer size and transmission rate, the queueing delay experienced by packets in a system using the Drop-from-Front policy is inversely proportional to the traffic overload factor. Thus, the queueing delay for packets that are eventually transmitted decreases as congestion grows. It is also important to mention here that even though packets are transmitted as cells over the ATM links, the dropping process at the routers is "packet–aware", and since all ATM links in our setup are point–to–point, no cell loss occurs at the ATM layer.

The simple configuration of Figure 1 allowed us to monitor the routing protocol events easily, perform all relevant measurements and interpret them, while still producing the effects encountered in larger systems. Results and events reported at run-time by the routing protocols are collected by the 192.168.63.1 workstation connected to the Ethernet management port of router HR1. Notice that in our setup only the ATM egress links (interfaces 1/0/0.1 and 1/0/1.1) connected to router HR1 are overloaded. Therefore, it suffices to monitor the state of the routing protocols at this router. We developed a set of Expect and Tcl/Tk scripts and C programs to collect and post-process observed data.

## 2.1 Experimentation Methodology

In our experiments we study and analyze the behavior of two routing protocols, OSPF and BGP, in congested environments. Both protocols are widely used and constitute the cornerstone of today's Internet infrastructure. OSPF is one of the most widely deployed intra-domain routing protocols, while BGP is the de facto standard for inter-domain routing protocols. The routing platform uses a 4.3 BSD-like TCP/IP stack, over which BGP establishes peerings with neighboring routers.

For each experiment we study the effect of network congestion on routing protocols for overload factors ranging from 25% to 400%. More precisely the traffic overload factor $f$ and the packet drop probability $p$ are defined as:

$$f = \frac{r' - r}{r}, \quad p = \frac{r' - r}{r'},$$

respectively; where $r$ is the transmission rate of the corresponding VC and $r'$ the rate of generated traffic. Furthermore, the effect of queueing delay is also evaluated: the buffer space, depending on the experiment, is set to either 4 Mbytes or 16 Mbytes. As mentioned above, the exact queueing delay depends not only on the transmission rate and the buffer size, but also on the dropping policy in effect. Detailed analytical and experimental results regarding the effect of buffer size and the expected queueing delay, for each dropping policy, are presented in Sections 3 and 4. We also investigate the effect of the size of data packets on the routing protocol dynamics. The relative size of the routing packets with respect to the data packets may introduce a bias in the dropping probabilities of the routing packets. Thus, the loss rate of routing packets may vary with different data packet sizes for the same overload factor. In order to see if data packet size has any substantial effect on the dynamics of routing protocols, we considered three possible sizes for data packets: 64, 256, and 1500 bytes. We found that routing protocol dynamics were similar for these three data packet sizes. For that reason and also due to space constraints, we only present results for 256 bytes. Results for 64 and 1500 bytes can be found in [20].

In our experiments we study the sensitivity of routing protocols to two parameters: the overload factor (level of congestion) and queueing delay. The platform used in the experiments supports static buffer threshold on a per physical interface basis. Additionally, the use of ATM and specifically the availability of CBR VCs, allowed us to shape egress traffic to arbitrary low rates. As a result, since only one VC was configured for each physical interface, it was rather simple to control the maximum queueing delay experienced by transmitted packets. Furthermore, reducing the VC transmission rate also reduces the amount of traffic that needs to be generated to bring the given IP interface to the desired overload level. This greatly reduced the requirements and minimized the cost of the traffic generation equipment needed for the experiments. In subsequent sections we illustrate that the robustness of routing protocols to congestion is mainly determined by the loss rate and the queueing delay and is independent of the actual link transmission rate.

Two quantities are used to characterize the behavior and robustness of routing protocols in congested networks: the time it takes for a routing flap to occur once the desired level of traffic overload is applied and the time that it takes for link adjacency to be re-established once a failure has occurred. Throughout the paper we refer to the first quantity as route flap time and we denote it as U2D (Up-to-Down); and to the second as adjacency recovery time, and we denote it as D2U (Down-to-Up).

To achieve a statistically accurate picture, each experiment is repeated between 10 and 16 times (the number of samples collected for each experiment was primarily determined by the amount of time required to collect these sample points). Confidence intervals for 95% and 99.5% confidence levels are computed and presented for the U2D and D2U quantities.

It is important to clarify here that the route flap time is independent of the existence of alternate forwarding paths. However, this is not true for the adjacency recovery time. In our experiments we have considered two possible cases: (i) a single forwarding path exists between routers HR1 and HR2 that consists of the primary link only, and (ii) two forwarding paths are available between routers HR1 and HR2 (through either the primary or the secondary link). In the first case, a static route is also installed in the HR1 router so that after a route flap occurs, the primary link is still used to forward traffic to the destination node. This configuration provides valuable information regarding the adjacency recovery time on a link that remains overloaded possibly due to the existence of traffic that remains unaffected by the route flap on the specific interface. In the second case, no static routes are installed. Therefore, after a route flap on the primary link occurs, packets are forwarded to the destination through the secondary path. While the secondary path is in effect, the transmission queue associated with the primary link drains, and eventually peering between the routers connected with the primarily link is re-established. At that time, the traffic reverts to the primary link. Therefore, in case (i) the duration of the D2U time depends on the traffic overload on the primary link, while in case (ii), it depends on the time required to drain the transmission queue associated with the primary link and the ability of the routing protocol to rapidly modify the corresponding entry in IP forwarding table in router HR1 and send traffic over the primary link again. When presenting the results, we refer to the experiments performed in case (i) above as *2-node*, and to those in case (ii) as *3-node* experiments.

# 3. ANALYTICAL MODELS

In this section, we develop analytical models for estimating the duration of the U2D (flap time) and D2U (adjacency recovery) cycles for both OSPF and BGP. Before proceeding further with a detailed discussion, we state two assumptions that are common to all our analytical models:

1. The overload factor remains constant. This assumption matches the experimental setup and makes it easy to compare results from the analytical models and the experiments.

2. Every packet has the same probability $p$ of being dropped irrespective of its size or its source. This probability $p$ depends only on the overload factor. Moreover, we also assume that the decision of dropping a particular packet is made independently for each packet. This assumption allows us to use the theory of Markovian processes for analyzing the dynamics of the routing protocols. Although verifying the validity of such an assumption is difficult, the results from our analytical models closely match those from experiments, suggesting that the assumption is reasonable in practice.

## 3.1 Route Flap for OSPF (U2D)

The behavior of OSPF during a U2D cycle can be modeled using an *absorbing* Markov chain. In order to refresh and maintain the OSPF adjacency with router HR2, router HR1 transmits a "hello" packet every $t_{HI}$ seconds known as "HelloInterval". We will refer to the timer associated with the hello interval as $T_{HI}$. The router on the other side, HR2, declares HR1 down if it does not receive a hello packet from HR1 within $t_{RDI}$ seconds which is known as "RouterDeadInterval". We will denote the router dead interval timer as $T_{RDI}$. Every time HR2 receives a hello packet from router HR1, it resets a timer that is scheduled to expire $t_{RDI}$ seconds later. If the timer does expire, HR2 declares that the link adjacency is down. Furthermore, routes learned over the failed interface are withdrawn from the routing and the forwarding tables. The default values for $t_{HI}$ and $t_{RDI}$ are set to 10 seconds and 40 seconds, respectively [16]. This means that the adjacency between HR1 and HR2 goes down if four consecutive hello packets are lost. An implicit assumption here is that hello packets are sent exactly every $t_{HI}$ seconds, and that both the ends have their clocks perfectly synchronized. A slight jitter in the time at which router HR1 sends hello packets may result in the adjacency going down even if four consecutive packets are not dropped. This may occur, for example, if three consecutive packets have been dropped at HR1, and HR2 receives the fourth packet a little after $t_{RDI}$ seconds with respect to the last received hello packet. In this case, the dead interval timer, that was set to $t_{RDI}$ at router HR2, will elapse before arrival of the fourth hello packet, thereby triggering a flap after only three consecutive drops. Normally, the routers have a jittering component built into timers whose expiry results in packets being sent out on the network to avoid synchronization problems that occur in large networks [4]. To avoid this problem, HR1 jitters $t_{HI}$ by picking a random value from a uniform distribution over an interval (10 - $\delta$, 10 + $\delta$). On our platform, the value of $\delta$ is set to 10% of the hello interval ($t_{HI}$). Note that only the timer associated with the transmission of hello packets, $T_{HI}$, is jittered. The $T_{RDI}$ timer is always reset to a fixed value. As for $\delta$ we assume that it is set to a value that guarantees the transmission of at least three hello packets within an interval of $t_{RDI}$ seconds. With this assumption, we can model the behavior of OSPF during a U2D cycle with the transition diagram shown in Figure 2. Our two-router system can be in one of the following five

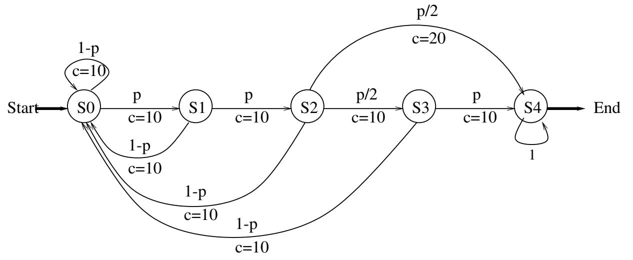



**Figure 2: Markov chain model for a U2D cycle of OSPF.**

states at any given time:

$S_0$: The last hello packet sent by HR1 made it to HR2. Our two-router system is in this state at the beginning of a U2D cycle.

$S_1$: The last hello packet sent by HR1 did not reach HR2, but the previous packet did.

$S_2$: The last two hello packets sent by HR1 did not arrive at HR2, but the packet before these two did.

$S_3$: The last three hello packets sent by HR1 did not reach HR2.

$S_4$: The $T_{RDI}$ timer at HR2 expires. The adjacency between HR1 and HR2 is down. This marks the end of the U2D cycle. Note that the number of consecutive hello packets dropped can be either three or four.

Each state in Figure 2 represents an event in the OSPF state machine, in this case the transmission of a hello packet. Also the diagram in Figure 2 shows the transition probabilities along with the cost associated with each transition, denoted by $c$. The cost of a transition is the time delay before a new event occur or equivalently the time to the next hello packet transmission. The next state depends on the outcome of the action trigerred by an event, i.e. whether the transmission of a hello packet was successful or not. As shown in the figure, the system starts from state $S_0$. The cycle ends when it reaches the absorbing state $S_4$. All the transitions in the model are synchronized to the events related to the transmission of hello packets from HR1 to HR2. The system moves towards the absorbing state $S_4$ from its present state if the next hello packet is dropped. Packets are dropped with probability $p$. Each time a hello packet makes it to HR2, the system falls back to the starting state $S_0$.

Transitions out of state $S_2$ require some more explanation. Recall that the system is in state $S_2$ if the last two hello packets have been dropped. From $S_2$, it may take either one or two more hello packet drops for the link to be declared down depending on whether the transmission of the fourth hello packet with respect to the last successfully transmitted one is greater or less than 40 seconds. In the first case the system transitions immediately to the terminating state $S_4$ while in the second it transitions to $S_3$. According to the assumption we made earlier, the time interval between two successive hello packets is picked from a uniform distribution over (10 - $\delta$, 10 + $\delta$). The variable that represents the time at which the fourth hello packet is sent relative to the last successful one is a sum of four independently chosen uniformly distributed random values. Therefore, the time at which the fourth hello packet gets transmitted is also a random variable with a distribution whose mean is equal to 40 seconds. Moreover, the probability of the time at which the fourth packet gets transmitted being less than 40 seconds turns out to be $1/2$. This lets us assign a probability of $p/2$ for each of the transitions $S_2 \rightarrow S_4$ and $S_3 \rightarrow S_4$.

Another point worth noting in Figure 2 is the cost of transitions. As we mentioned earlier, every transition in the figure corresponds to a hello packet transmission attempted by HR1. This occurs every (10 - $\delta$, 10 + $\delta$) seconds. However, with the exception of $S_2 \rightarrow S_4$, we have assigned a fixed 10 seconds cost to each transition. Assigning a constant cost equal to $t_{HI}$ to each transition simplifies the analysis. Our model already takes care of the main effect caused by the jitter component by providing two transition paths from $S_2$ to $S_4$. Note that transition $S_2 \rightarrow S_4$ has a cost of 20 seconds ($t_{RDI} - 2 \times t_{HI}$), which is the residual content of timer $T_{RDI}$ after two unsuccessful attempts. This also means that a path from $S_0$ to $S_4$ with no repetitions of states will have a total cost of 40 seconds, which is equal to $t_{RDI}$.

Having described the model, we turn to estimating the expected time of a U2D cycle or equivalently the expected time of moving from the starting state $S_0$ to the terminating state $S_4$. Using the theory of absorbing Markov chains [5], we can calculate the expected duration of a U2D cycle for OSPF as a function of $p$. A closed form expression for the expected flap time as a function of $p$ is given below:

$$E[U2D](p) = \frac{20}{p^3 + p^4} + \frac{20}{p^2 + p^3} + \frac{2(5p + 10)}{p + p^2} + \frac{10}{1 + p} \quad (1)$$

Table 1 presents the expected flap time $E[U2D](p)$ values as computed from Eq. (1) for five overload factors for which we carried out experimental measurements.

The model assumes that the system lies in state $S_0$ at the beginning of a U2D cycle. While this is true for the 2–node experiments, a minor correction to the model is required to make it applicable to 3–node configurations. The difference in the 3–node experiments is that after the forwarding path in HR1 reverts from the secondary link to the primary one, packet losses will occur only after the buffer overflows. Therefore, for the 3–node configurations, we have to add the queue fill-up time to the expected flap time given by Eq. (1) when computing the actual expected flap time. The time it takes to fill the queue in turn depends on the overload factor as well as the queue capacity. Table 1 shows the queue fill–up time for the two buffer sizes used in our experiments and for various overload factors. Table 1 also includes expected flap time values for the parameters used in the two 3–node configurations. These values are obtained by adding the queue fill–up time to the expected values derived from the model (which is also equal to $E[U2D](p)$ values of the 2–node configuration).

## 3.2 Adjacency Recovery for OSPF (D2U)

The behavior of OSPF during a D2U cycle differs for 2–node and 3–node experiments. In both the cases, the OSPF

| Overload Factor (%) | $p$ | 2–node 4 MB buffer $E[U2D](p)$ (seconds) | 3–node 4 MB buffer Queue Fill–Up Time (seconds) | | 3–node 16 MB buffer Queue Fill–Up Time (seconds) | |
|---|---|---|---|---|---|---|
| | | | Queue Fill–Up Time (seconds) | $E[U2D](p)$ (seconds) | Queue Fill–Up Time (seconds) | $E[U2D](p)$ (seconds) |
| 25 | 0.20 | 2600.00 | 17.21 | 2617.21 | 68.83 | 2668.83 |
| 50 | 0.33 | 600.00 | 8.47 | 608.47 | 33.87 | 633.87 |
| 100 | 0.50 | 200.00 | 4.20 | 204.20 | 16.80 | 216.80 |
| 200 | 0.67 | 97.50 | 2.09 | 99.59 | 8.36 | 105.86 |
| 400 | 0.80 | 64.06 | 1.04 | 65.10 | 4.16 | 68.22 |

**Table 1: Expected value of flap time (U2D) for OSPF.**

| Overload Factor (%) | $p$ | Expected Number of Hello packets | $E[D2U](p)$ (seconds) |
|---|---|---|---|
| 25 | 0.20 | 1.25 | 12.5 |
| 50 | 0.33 | 1.5 | 15 |
| 100 | 0.50 | 2.0 | 20 |
| 200 | 0.67 | 3.0 | 30 |
| 400 | 0.80 | 5.0 | 50 |

**Table 2: Expected value of recovery time (D2U) for OSPF in 2-node experiments.**

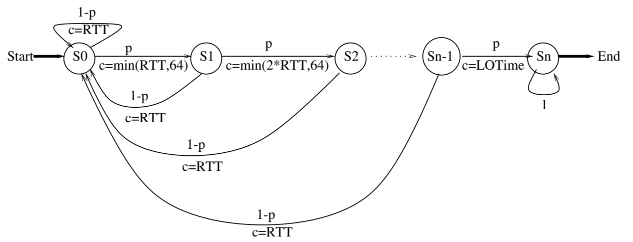



**Figure 3: Markov chain model for a U2D cycle of BGP.**

adjacency between HR1 and HR2 comes up as soon as one hello packet is able to reach HR2[1]. For the 2–node experiments, since the HR1→HR2 link is overloaded, the hello packets may get dropped even during a D2U cycle. Since the probability of a packet being dropped at HR1 is $p$, the probability of a hello packet reaching HR2 is $(1-p)$. Hence, the expected number of hello packet transmission attempts before the route comes up is equal to $\frac{1}{(1-p)}$. Table 2 lists the expected duration of a D2U cycle ($E[D2U](p)$) for various overload factors.

## 3.3 Route Flap for BGP (U2D)

In this section we introduce the framework that is used to analyze the behavior of BGP. Notice that there is a fundamental difference between the approach we describe here for modeling BGP and that used for OSPF: messages generated by BGP are transmitted over TCP that guarantees reliable delivery. Therefore, in order to accurately model the packet transmission and the dropping process we have to take into account interactions between BGP and TCP.

[1]In reality, the adjacency establishment also involves link state database synchronization between the routers, but in our experimental setup the adjacency gets established from HR1's point of view as soon its first hello packet reaches HR2.

BGP peers periodically exchange keepalive and route update messages. In our experimental setup route update messages are infrequent. Therefore, in the rest of the discussion we focus exclusively on keepalive messages. The purpose of the keepalives is to reaffirm the liveliness of the peer and to determine the operational status of the underlying communication path. Each one of the BGP peers associates a timer, known as "KeepaliveTimer" and denoted by $T_{KT}$, with every active session. The expiration of the $T_{KT}$ timer triggers the transmission of a new keepalive message. In the absence of route update messages, this timer fires every $t_{KT}$ seconds. Let us assume now that router HR1 is peering with router HR2, and consider the messages flowing from HR1 to HR2: router HR1 transmits a keepalive message every $t_{KT}$ seconds. Router HR2 defines a window of duration $t_{HT}$, known as "HoldTime", during which it expects to receive at least one keepalive message from HR1. The corresponding timer is known as "HoldTimer" and will be denoted as $T_{HT}$. In the routing platform used in our experiments $t_{KT}$ and $t_{HT}$ were set to 60 and 180 seconds respectively. Notice that the keepalive messages are transmitted over TCP. Therefore, ordered delivery is guaranteed. That is, a keepalive message that arrives at HR2 will be presented to BGP if and only if all messages sent before it have already been successfully delivered to BGP. Therefore, for the adjacency to be refreshed, HR2 must receive at least one keepalive message within $t_{HT}$ seconds from the arrival of the previous message. The property of ordered delivery is important because it allows us to focus on one keepalive message at a time when modeling the flap time. This is because if TCP fails to deliver a given keepalive within $t_{HT}$, we are guaranteed that no other keepalive messages generated at the source BGP peer will get to the destination BGP peer either.

TCP uses retransmissions to achieve reliable packet delivery. The retransmission interval, known as RTO, is a function of the current RTT estimate and the RTT standard deviation. Furthermore, a backoff factor of two is applied to RTO for every unsuccessful retransmission attempt of a given packet.

The overall dynamics of the system are quite complex. To simplify the analysis we make the following assumptions:

1. The residual time in the $T_{HT}$ timer in router HR2 is equal to $t_{HT}$ at the time the first transmission attempt of the next keepalive message at HR1 is made.

2. The initial RTT for a TCP session is equal to the actual queueing delay plus link propagation delay, and the standard deviation of the measured RTT is zero.

| Overload Factor (%) | Drop–from–Front RTT (seconds) | $n$ | LOTime (seconds) | Expected Time (seconds) | Drop–Tail Expected Time (seconds) |
|---|---|---|---|---|---|
| 50 | 2.67 | 7 | 33.33 | 11118.67 | 5500.00 |
| 100 | 2.0 | 7 | 54.0 | 1076.0 | 948.00 |
| 200 | 1.33 | 8 | 32.0 | 476.49 | 390.44 |
| 400 | 0.80 | 9 | 14.4 | 304.95 | 260.25 |

**Table 3: Expected flap time (U2D) for BGP. The values of RTT, $n$ and LOTime are same for all the overload factors for Drop–Tail, and they are 4 seconds, 6 and 54 seconds respectively.**

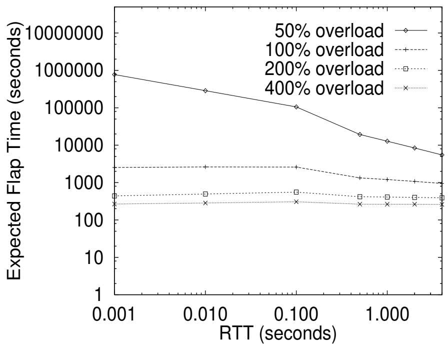



**Figure 4: Expected flap time (U2D) for BGP versus RTT for various overload factors: $t_{HT} = 180$ seconds.**

The above assumptions result in overestimating the number of retransmission attempts by TCP before an adjacency is declared down [20]. Therefore, it is anticipated that our model will slightly overestimate the flap time. The experimental results presented in Section 4.3 confirm this.

The state transition diagram used to compute the expected flap time (U2D) for BGP is shown in Figure 3. Each state in the diagram represents the transmission of a keepalive message. In case of a successful transmission, the state transition cost is equal to RTT, which is the time needed for TCP to receive back an acknowledgment. In case of a packet loss, the transition cost grows as indicated in the diagram, as the retransmission interval increases due to the backoff factor that is applied to the current RTO value. With a 4.3 BSD–like TCP implementation in our routers, the RTO value cannot exceed 64 seconds [22]. Also, it is important to observe that the number of states in the diagram depends on the number of retransmissions that can be attempted by TCP and is consequently dependent on the actual RTT of the corresponding peering session. The number of states $n$ is the maximum integer that satisfies the following condition:

$$t_{HT} >= \left( \sum_{i=0}^{n-2} c_{i,i+1} \right) \tag{2}$$

The residual time, denoted as LOTime (left over time), can be computed as:

$$\text{LOTime} = t_{HT} - \left( \sum_{i=0}^{n-2} c_{i,i+1} \right) \tag{3}$$

The expected flap time (U2D) for BGP can then be obtained by solving the Markov chain that corresponds to the RTT of interest. As an example, Eq. (4) shows the expected value of BGP flap time as a function of $p$ for RTT value of 1 second.

$$E[U2D](p) = \frac{1}{p^8}[1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6 + 64p^7 + 52p^8] \tag{4}$$

Table 3 shows the expected flap time for BGP as a function of the traffic overload for both Drop-from-Front and Drop-Tail policies. In this particular scenario, the buffer size was set to 4 MBytes and the link speed to 10 Mbits/sec. Therefore, the queueing delay with Drop-Tail policy is 4 seconds. The queueing delay with Drop-from-Front depends on the overload factor and is equal to $4 * (1 - p)$, where $p$ is the packet drop probability. In this specific case the link propagation delay is negligible and is ignored. It is important to observe in Table 3 that BGP becomes significantly more robust when the Drop-from-Front policy is in effect.

Figure 4 illustrates the effect of RTT on BGP's flap time for a number of different overload factors. As can be seen from this figure, increasing RTTs lead to smaller flap times. As explained earlier, this behavior is expected since the number of TCP retransmission attempts within a $T_{HT}$ window decreases as the RTT increases.

## 3.4 Adjacency Recovery for BGP (D2U)

In this section we develop a model to capture the dynamics of the BGP adjacency recovery process. Modeling this process requires modeling two components: the TCP connection establishment and the BGP session establishment. Notice that the establishment of the TCP connection precedes that of the BGP session.

BGP session establishment is bidirectional in nature. It is possible that both BGP peers initiate and establish a BGP session with each other simultaneously. BGP implementations are equipped with connection collision detection and resolution mechanisms [18]. In uncongested networks it does not matter in whose favor the collision is resolved. However, in our setup the traffic that causes congestion flows only in one direction. Therefore, the cost of establishing a TCP connection in the congested direction is higher than that in the opposite direction.

TCP uses a three-way handshake to establish a connection. First, the client-end of the connection sends a SYN segment to the server requesting that a connection be opened. Next, the server responds with a SYNACK segment that acknowledges the SYN segment received from the client and request from the client to establish the other half of the connec-

| Overload Factor (%) | Drop–from–Front | | | | | | | Drop–Tail |
|---|---|---|---|---|---|---|---|---|
| | RTT (seconds) | Initial RTO (seconds) | Number of $R_i$s ($n$) | $a1$ | $a2$ | LOTime (seconds) | Expected Time (seconds) | Expected Time (seconds) |
| 50 | 2.67 | 8.01 | 4 | 1 | 3 | 59.99 | 126.85 | 107.49 |
| 100 | 2.0 | 6.00 | 5 | 1 | 3 | 26.00 | 171.98 | 189.19 |
| 200 | 1.33 | 3.99 | 5 | 2 | 4 | 56.00 | 351.10 | 452.42 |
| 400 | 0.80 | 2.40 | 6 | 2 | 4 | 41.60 | 788.88 | 1452.03 |

**Table 4: Expected adjacency recovery time (D2U) for BGP connection when congestion is from the client to the server of the TCP connection (the values of RTT, initial RTO, number of $R_i$s ($n$) and LOTime are same for all overload factors for Drop–Tail and they are 4 seconds, 12 seconds, 4 and 32 seconds respectively).**

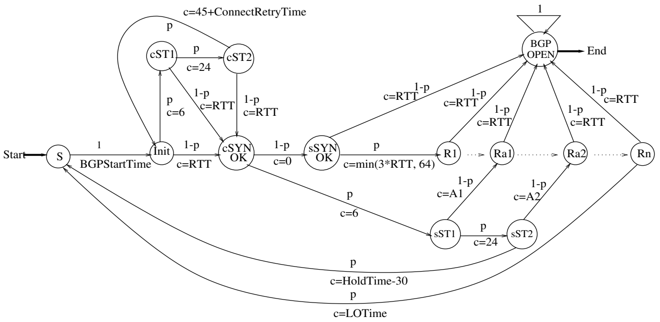



**Figure 5: Markov chain model for a D2U cycle of BGP when congestion is from the client to the server of the TCP connection.**

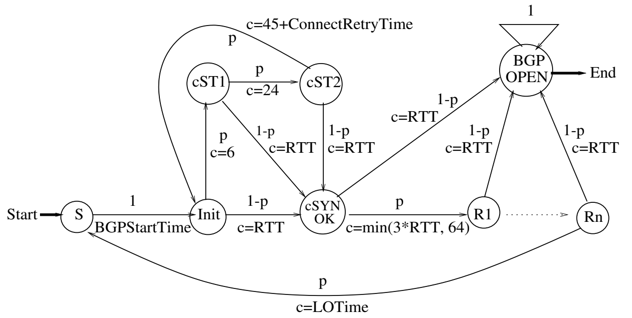



**Figure 6: Markov chain model for a D2U cycle of BGP when congestion is from the server to the client of the TCP connection.**

tion. Finally, the client responds with an ACK segments that acknowledges the SYNACK segment received from the server. If the congestion is in the client-to-server direction, our framework must model the process of successfully transmitting two segments over the congested interface: the SYN and the ACK (to the SYNACK segment) since these two segments are transmitted in the congested direction. Otherwise, only the successful transmission of the SYNACK segment needs to be modeled.

We have developed models to capture the adjacency recovery times for both cases: the first case represents a lower bound and the second an upper bound on the expected recovery time. Once the TCP connection is established, the only parameter that remains to be modeled is the time needed for the BGP "open" message to be successfully delivered to its peer.

Figure 5 shows the state diagram that models the BGP adjacency recovery time when congestion is in the direction from the TCP client to the server. Similarly, the state diagram shown in Figure 6 models the case where congestion is in the opposite direction.

In Figure 5, states Init, cST1, cST2 and cSYNOK model the transmission of the SYN segments. In Figure 6, the same states model the transmission of the SYNACK segment. Notice that TCP attempts to (re)transmit a SYN or a SYNACK segment at most three times: at $t$ = 0, 6, and 30 seconds [22]. If all three transmissions fail, the TCP connection establishment is aborted and the state machine eventually returns to the Init state.

In Figure 5, states sSYNOK, sST1 and sST2 model the events associated with the attempted transmission of the ACK segment that is sent in response to the receipt of the SYNACK segment from the server. The values of parameters A1, A2, and LOTime shown in this figure are given by Eq. (5), (6), and (7), respectively.

$$A1 = \left( \sum_{i=0}^{a1-1} min(3 * RTT * 2^i, 64) \right) - 6 \tag{5}$$

$$A2 = \left( \sum_{i=0}^{a2-1} min(3 * RTT * 2^i, 64) \right) - 30 \tag{6}$$

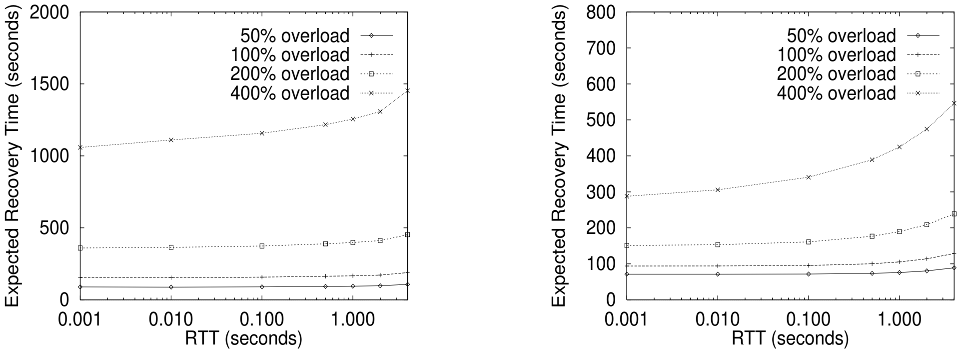
$$\text{LOTime} = t_{HT} - \left( \sum_{i=0}^{n-1} min(3 * RTT * 2^i, 64) \right) \tag{7}$$

In both Figures 5 and 6 the remaining states capture the events associated with the transmission of the BGP open message. As was the case with the BGP flap time, the exact structure of the state machines depends on the network delays.

Table 4 presents the expected adjacency recovery time with BGP, for the model shown in Figure 5 for a range of overload factors and for both Drop-from-Front and Drop-Tail policies. Notice that with Drop-from-Front the RTT depends on the overload factor. Table 5 presents the adjacency recovery time when congestion is in the direction from the server to the client end of the TCP connection. Figure 7(a) and (b) illustrate the effect of RTT on the adjacency recovery time for BGP in the case that congestion is in the direction from the client to the server and in the opposite direction

| Overload Factor (%) | Drop–from–Front | | | | | Drop–Tail |
|---|---|---|---|---|---|---|
| | RTT (seconds) | Initial RTO (seconds) | Number of $R_i$s ($n$) | LOTime (seconds) | Expected Time (seconds) | Expected Time (seconds) |
| 50 | 2.67 | 8.01 | 4 | 59.99 | 83.52 | 88.91 |
| 100 | 2.0 | 6.00 | 5 | 26.00 | 114.01 | 128.76 |
| 200 | 1.33 | 3.99 | 5 | 56.00 | 201.15 | 239.01 |
| 400 | 0.80 | 2.40 | 6 | 41.60 | 419.74 | 546.39 |

**Table 5: Expected adjacency recovery time (D2U) for BGP connection when congestion is from the server to the client of the TCP connection (the values of RTT, initial RTO, number of $R_i$s ($n$) and LOTime are same for all the overload factors for Drop–Tail and they are 4 seconds, 12 seconds, 4 and 32 seconds respectively).**




(a) Congestion in TCP client to server direction (b) Congestion in TCP server to client direction

**Figure 7: Expected recovery time (D2U) for BGP versus RTT for different overload factors. HoldTime = 180 seconds, ConnectRetryTime = 120 seconds and BGPStartTime = 60 seconds.**

respectively. The interested reader is referred to [20] for a detailed discussion on the operation of TCP, BGP and their interaction, as well as for a detailed description of the state machine.

# 4. EXPERIMENTAL RESULTS

In this section, we present experimental results based on data and measurements collected using the network configuration described in Section 2. We performed experiments for both OSPF and BGP. The experiments attempt to identify the effect of parameters such as packet size, buffer size (or equivalently queueing delay) and network topology on the stability of the routing protocol. In the experiments we use 64, 256, and 1500 byte packets. Due to space constraints, we present here only the results for the 256-byte packets. The results for 64- and 1500-byte packets are not fundamentally different from those corresponding to 256 bytes [20]. We study the effect of queueing delay on OSPF by setting the buffer size to either 4 MBytes or 16 MBytes.

For each experiment, the number of samples obtained varies between 10 and 16. In our plots we show the mean as well as the 95% and 99.5% confidence intervals of the collected sample points for the flap time (U2D) and the adjacency recovery time (D2U). Each of these time is then compared to the expected values obtained using the analytical models given in Section 3. Also, each set of results is presented in both linear and logarithmic scale. To avoid measurement errors at the beginning of each experiment, the first measurement is always ignored. We should note here that due to large variations in some sets of measurements, the low end of the 99.5% confidence intervals may become negative. In our plots, negative values are always clamped to 1.0.

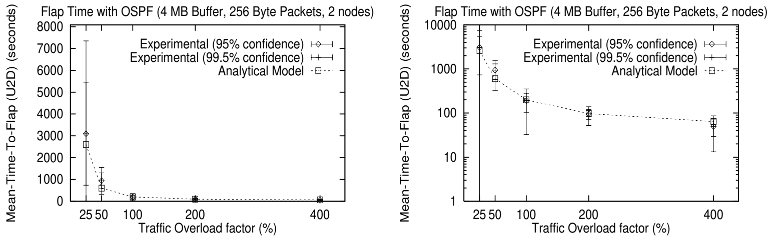
## 4.1 OSPF experiments: the 2–node case

The buffer size for the 2–node OSPF experiments was set to 4 MBytes. The flap time (U2D) and adjacency recovery time (D2U) for 256 byte packets are shown in Figures 8 and 9, respectively. The results presented in these plots demonstrate that there is a close match between the analytically derived values in Sections 3.1 and 3.2 and the experimental ones. This validates the models used for obtaining expected values for the flap and adjacency recovery times. OSPF's behavior with 64 and 1500 byte packets [20] is similar to that observed with 256 byte packets.

## 4.2 OSPF experiments: the 3–node case

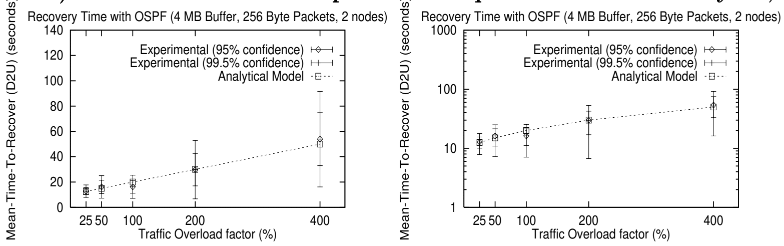
In our 3–node experiments with OSPF, we investigate the effect of buffer size, and consequently the queueing delay, on its stability properties. In particular, we performed the experiments with buffer sizes of 4 MBytes and 16 MBytes. In these experiments the data packet size was always fixed at 256 bytes.

The main difference between the 3–node and the 2–node case is that, in the former case, traffic is physically diverted from the primary to the secondary link when a flap occurs. Notice that the flap time (U2D) is determined mainly by the routing message drop process. The buffer size contributes only a fixed component to the U2D time, to account for the time it take for the buffer to fill after an adjacency recovery from the previous flap has been completed. This is important since the analytical model is applicable only after the dropping probability for outgoing packets becomes $p$. This occurs only when the buffer becomes full. Once the buffer is full, the buffer size has no bearing on exactly when a route flap occurs. Section 3.1 presented the expected flap time




(a) linear scale (b) log scale

**Figure 8: Flap Time (U2D) for 2–node OSPF experiments: packet size = 256 bytes, buffer size = 4 MB.**




(a) linear scale (b) log scale

**Figure 9: Recovery Time (D2U) for 2–node OSPF experiments: packet size = 256 bytes, buffer size = 4 MB.**

taking into account the buffer fill time. However, since the buffer fill time is orders of magnitude smaller than the expected flap time, the flap time for the 3–node configuration is not expected to differ significantly from that observed in the 2–node case. This is verified by Figures 8 and 10. Due to space constraints, we have omitted the 3-node U2D results for 16 MBytes buffer size which do not differ significantly from their 4 MByte counterparts [20]. Figure 10 also verifies the accuracy of our analytical model and demonstrates the excellent match between the analytical results and the measured values from experiments.

The behavior during a D2U cycle in the 3–node case is very different from that observed in the 2–node case. The reason is that the primary link (HR1→HR2) does not remain overloaded: once the route flaps, data traffic is diverted over to the secondary link (HR1→HR0). This allows for the adjacency to recover almost immediately since the very first hello packet queued after the flap makes it to HR2. The exact time at which the hello packet reaches HR2 depends on the number of packets ahead of it at the primary interface at the time it gets queued. Therefore, the buffer size affects recovery time. However, the adjacency recovery time in the 3–node case is relatively small compared to the flap time or to the adjacency recovery time in the 2–node case. The effect of buffer size on the adjacency recovery time can be seen from Figures 11 and 12.

## 4.3 BGP experiments

We have performed only 2–node experiments for BGP. The buffer size in these experiments was set to 4 MBytes. As before, we consider three packets sizes: 64, 256, and 1500 bytes, but present results for only 256 bytes. These results are shown in Figures 13 and 14. Regarding flap times (U2D), we can see that the expected values are higher than the mean values in most of the cases. This confirms our expectation that analytically calculated values should overestimate the actual flap times as explained in Section 3.3.

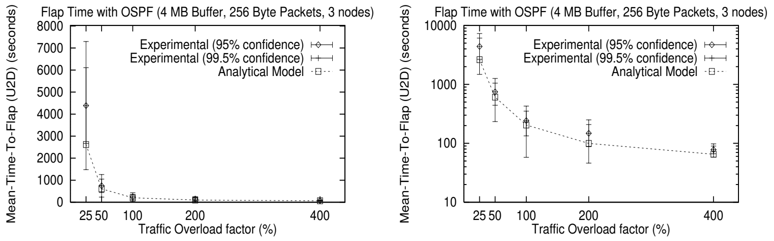
The graphs for adjacency recovery time (D2U) show the expected values for both HR1 and HR2 initiated connections. Both the theoretical values are lower than the mean values in most of the cases. One reason is that our models assume that connections initiated by both the ends do not interact with each other at all (see Section 3.4). This is not true in reality. In our routers this interaction proved to be "destructive" more often than expected, as we found out through a detailed investigation of the traces generated by the TCP and BGP state machines of the routers. We observed that the connection establishment process failed several times because of interference caused by the connection initiated by the other side. It is also important to note here that the BGP state machine implementation on our routing platform deviates slightly from that described in the BGP specification [18], especially in the part dealing with recovery from failed connection establishment. This leads to a higher than expected mean recovery time in most cases.

The results obtained for 64 and 1500 byte packets demonstrated similar behavior [20]. As with OSPF, the packet size does not alter the fundamental BGP dynamics.

# 5. CONCLUSION

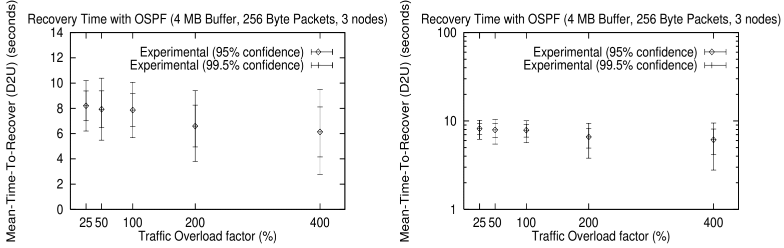
In this paper we studied the stability properties of routing protocols in congested networks. We have analyzed two widely deployed protocols: OSPF and BGP. These protocols were chosen due to the important role they play in the Internet today. We have used both analysis and experimentation to obtain in-depth understanding of their dynamics and to quantify their tolerance to traffic overload. In addition to traffic overload, the effects of several other factors — such as packet sizes, buffer size, dropping policy and link propagation delays — were also studied.

We observe that OSPF's behavior depends only on the traffic overload factor and is insensitive to the packet size distribution, the buffer size, or the packet dropping policy in effect. This significantly simplified our analytical model and



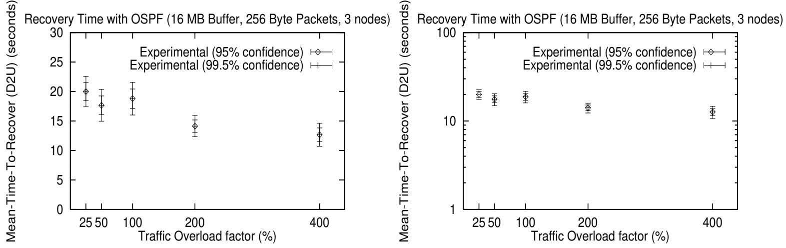

(a) linear scale (b) log scale

**Figure 10: Flap Time (U2D) for 3–node OSPF experiments: packet size = 256 bytes, buffer size = 4 MB.**



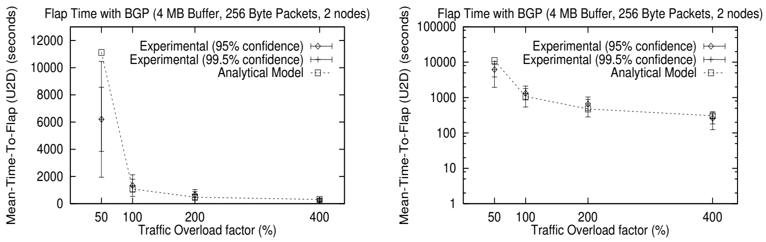

(a) linear scale (b) log scale

**Figure 11: Recovery Time (D2U) for 3–node OSPF experiments: packet size = 256 bytes, buffer size = 4 MB.**



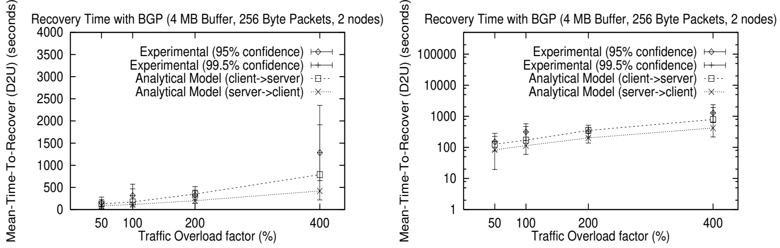

(a) linear scale (b) log scale

**Figure 12: Recovery Time (D2U) for 3–node OSPF experiments: packet size = 256 bytes, buffer size = 16 MB.**




(a) linear scale (b) log scale

**Figure 13: Flap Time (U2D) for 2–node BGP experiments: packet size = 256 bytes, buffer size = 4 MB.**




(a) linear scale (b) log scale

**Figure 14: Recovery Time (D2U) for 2–node BGP experiments: packet size = 256 bytes, buffer size = 4 MB.**

enables us to derive closed-form expression that accurately captures its stability properties.

Deriving an accurate analytical model for BGP proved to be more challenging because the exact form of the analytical model depends on the round-trip time between peering nodes. This is because BGP uses TCP for reliable packet transmission. Using the analytical models, we computed the expected flap and link recovery times for a range of RTTs and we showed that the resilience of BGP to congestion decreases with increasing queueing and link propagation delays. Furthermore, this suggested that packet dropping and buffer management policies that reduce the queueing delay of routing protocol messages improve the robustness of BGP.

We also performed extensive measurements in an experimental network of routers to validate the results from our analytical models. The data collected match the analytical results well, thereby validating the accuracy of our models. Our analysis and experimental results clearly demonstrate the need to isolate routing protocol messages from data traffic by employing a combination of scheduling, queueing, and buffering mechanisms in order to improve the stability of routing protocols.

We plan to extend the work in this paper by making the traffic and loss models more realistic. We also plan to tie these results with actual measurements in the Internet. Labovitz et al. [11, 12] have shown that there is a significant correlation between the measured BGP instability and network usage. It will be interesting to see what fraction of these instabilities is due to congestion and routing packet losses. We also plan to extend our models to include other deployed routing protocols such as IS-IS, and emerging signaling protocols such as LDP (Label Distribution Protocol) [2] as a part of the future work.

## 6. ACKNOWLEDGMENTS

This research is supported by Cisco Systems, Lucent Technologies, and the University of California MICRO program. We thank the anonymous reviewers for their valuable comments.

## 7. REFERENCES

[1] North American Network Operators Group (NANOG), mailing list archives, http://www.nanog.org.

[2] L. Andersson, P. Doolan, N. Feldman, A. Fredette, and B. Thomas. LDP Specification. *Internet Draft*, Oct. 1999.

[3] B. Chinoy. Dynamics of Internet Routing Information. In *Proceedings of the ACM SIGCOMM '93*, Sept. 1993.

[4] S. Floyd and V. Jacobson. The Synchronization of Periodic Routing Messages. *IEEE/ACM Transactions on Networking*, 2(2):122–136, Apr. 1994.

[5] R. Goodman. *Introduction to Stochastic Models*. The Benjamin/Cummings Publishing Company, Inc., Menlo Park, California, 1988.

[6] R. Govindan, C. Alaettinoglu, G. Eddy, D. Kessens, S. Kumar, and W.-S. Lee. An Architecture for Stable, Analyzable Internet Routing. *IEEE Network*, 13(1):29–35, 1999.

[7] R. Govindan and A. Reddy. An Analysis of Internet Inter–Domain Topology and Route Stability. In *Proceedings of the IEEE INFOCOM '97*, 1997.

[8] T. G. Griffin and G. Wilfong. An Analysis of BGP Convergence Properties. In *Proceedings of the ACM SIGCOMM '99*, Sept. 1999.

[9] B. Halabi. *Internet Routing Architectures*. Cisco-Press, 1997.

[10] V. P. Kumar, T. V. Lakshman, and D. Stiliadis. Beyond Best Effort: Routers Architectures for the Differentiated Services of Tomorrows Internet. *IEEE Communications Magazine*, 36(5):152–64, 1998.

[11] C. Labovitz, A. Ahuja, and F. Jahanian. Experimental Study of Internet Stability and Backbone Failures. In *Proceedings of the 29th Annual International Symposium on Fault-Tolerant Computing*, June 1999.

[12] C. Labovitz, G. R. Malan, and F. Jahanian. Internet Routing Instability. In *Proceedings of the ACM SIGCOMM '97*, Sept. 1997.

[13] C. Labovitz, G. R. Malan, and F. Jahanian. Origins of Internet Routing Instability. In *Proceedings of the IEEE INFOCOM '99*, 1999.

[14] M. Lauback. Classical IP and ARP over ATM. *Request for Comments (RFC): 1577*, January 1994.

[15] J. T. Moy. *OSPF : Anatomy of an Internet Routing Protocol*. Addison–Wesley Publishing Company, Reading, Massachusetts, Jan. 1998.

[16] J. T. Moy. OSPF Version 2. *Request for Comments (RFC): 2328*, April 1998.

[17] V. Paxson. End–to–End Routing Behavior in the Internet. In *Proceedings of the ACM SIGCOMM '96*, Aug. 1996.

[18] Y. Rekhter and T. Li. A Border Gateway Protocol 4 (BGP-4). *Request for Comments (RFC): 1771*, March 1995.

[19] S. Savage, A. Collins, and E. Hoffman. The End–to–End Effects of Internet Path Selection. In *Proceedings of the ACM SIGCOMM '99*, Sept. 1999.

[20] A. Shaikh. Analysis of Routing Protocol Stability in Congested Networks. MS Thesis, University of California, Santa Cruz, June 2000.

[21] K. Varadhan, R. Govindan, and D. Estrin. Persistent Route Oscillations in Inter–Domain Routing. ISI technical report 96–631, USC/Information Sciences Institute, 1996.

[22] G. R. Wright and W. R. Stevens. *TCP/IP Illustrated Volume 2: The Implementation*. Addison–Wesley Publishing Company, Jan. 1995.