
Interactions of Intelligent Route Control with TCP Congestion Control

Ruomei Gao,¹ Dana Blair,² Constantine Dovrolis,¹ Monique Morrow², and Ellen Zegura¹

¹ College of Computing, Georgia Institute of Technology

² Cisco Systems Inc.

Summary. Intelligent Route Control (IRC) technologies allow multihomed networks to dynamically select egress links based on performance measurements. TCP congestion control, on the other hand, dynamically adjusts the send-window of a connection based on the current path's available bandwidth. Little is known about the complex interactions between IRC and TCP congestion control. In this paper, we consider a simple dual-feedback model in which both controllers react to packet losses, either by switching to a better path (IRC) or by reducing the offered load (TCP congestion control). We first explain that the IRC-TCP interactions can be synergistic as long as IRC operates on larger timescales than TCP ("separation of timescales"). We then examine the impact of sudden RTT changes on TCP, the behavior of congestion control upon path changes, the effect of IRC measurement delays, and the conditions under which IRC is beneficial under two path impairment models: short-term outages and random packet losses.

1 Introduction

We consider a model in which the egress TCP traffic of a stub network \mathcal{S} is controlled by an IRC-capable multihomed edge router. In particular, we focus on the aggregate TCP traffic $T(\mathcal{S}, \mathcal{D})$ from \mathcal{S} towards a destination network \mathcal{D} . The traffic $T(\mathcal{S}, \mathcal{D})$ is subject to two closed-loop controllers: TCP congestion control at the flow level and IRC path control for the entire aggregate (see Figure 1).

TCP interprets packet losses as indications of congestion and adjusts the congestion window of each flow based on the well-known TCP congestion control algorithms. TCP uses ACKs and the Retransmission TimeOut (RTO) to close the feedback loop, and so the TCP *reaction timescale*, i.e., the amount of time it takes to detect a packet loss and decrease the congestion window, is roughly in the order of one Round-Trip Time (RTT).

IRC, on the other hand, interprets packet losses as indication of path impairment (not necessarily congestion). It controls the egress link, and thus the forwarding path, of the TCP flows from \mathcal{S} to \mathcal{D} . IRC can switch to another

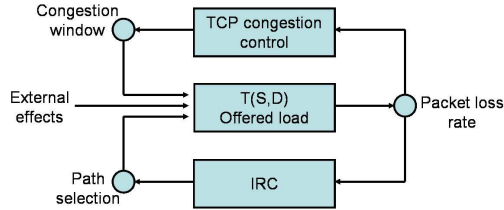


Fig. 1. The dual-feedback model.

path at the end of each *routing period* (of duration T_r), after estimating the performance of all egress paths through active or passive monitoring [4]. A typical value for T_r in current IRC products is a few tens of seconds [3].

The control action of both TCP and IRC can affect the packet loss rate that $T(S, D)$ experiences. For instance, if $T(S, D)$ saturates the current path causing some packet drops, then TCP reduces the window of the affected connections decreasing the offered load. Or, if the current path is subject to short-term outages, then IRC can switch $T(S, D)$ to another path that does not suffer from such impairments. Note that the packet loss rate that $T(S, D)$ experiences is not determined only by the two controllers, but also by external effects (e.g., congestion due to other traffic in the current path, faulty equipment, routing instabilities).

In this paper, we first discuss the “separation of timescales” principle, which states that the two controllers, TCP and IRC, should operate with significantly different reaction timescales so that they do not compete with each other. We then focus on the impact of sudden RTT changes, as a result of path switching, on TCP’s throughput. An RTT decrease can cause packet reordering and throughput reduction due to Fast-Retransmit. A large RTT increase, on the other hand, can also cause throughput loss due to the expiration of the Retransmission Timeout. We also examine the impact of sudden available bandwidth changes, as a result of path switching, on TCP’s throughput. The important point here is that TCP may not be able to quickly capture the additional throughput that is available in the path that IRC has switched to. This is something that the IRC path selection process needs to take into account to avoid unnecessary switching decisions.

We next study the performance of IRC when it relies on a realistic active measurement process, and under two path impairment models: short-term outages and random losses. We conclude that an IRC system that always switches to a path with lower loss rate may fail to provide the maximum possible TCP throughput to its users. This point indicates that IRC systems can benefit from using predicted throughput as the main path switching criterion. Last, we examine whether IRC is beneficial overall, depending on the underlying path impairment. Our analysis shows that IRC is synergistic to TCP for outages that last more than the measurement timescale and when the loss rate is significant.

The structure of the paper is as follows. We first discuss the "separation of timescales" issue (§2). Then, we investigate the impact of sudden RTT changes on TCP throughput (§3). Next (§4), we examine how TCP congestion control reacts to path changes, depending on the available bandwidth difference between the two paths. The impact of measurement latency on the IRC path selection process is the focus of §5. Finally, in §6, we put everything together and examine the conditions under which IRC is beneficial for TCP traffic.

2 Separation of timescales

In the rest of this paper, we consider a simple instance of the dual-feedback model in which there are only two paths from S to D . One path is referred to as "primary", while the other is referred to as "backup" and it is only used as long as IRC detects a path impairment at the primary. Further, we model each of the primary and backup paths as a bottleneck link of capacity C_p and C_b , respectively. By definition, $C_p \geq C_b$. The RTTs of the two paths are RTT_p and RTT_b , respectively. The following ns-2 simulations use TCP NewReno with Selective and Delayed ACKs.

In the dual-feedback model described earlier, the role of IRC can be either *synergistic* or *antagonistic* to TCP. Synergistic interaction means that the throughput of $T(S, D)$ with IRC is larger than the throughput without IRC, otherwise, the interactions are antagonistic. Antagonistic interactions can happen when IRC reacts to packet losses that TCP is designed to handle. For example, IRC can be antagonistic to TCP when it reacts to packet losses that are caused when $T(S, D)$ saturates its current path, and the new path chosen by IRC provides a lower throughput than the original path. In this case, IRC should stay the course and let TCP react to these packet losses. On the other hand, synergistic interaction takes place when IRC reacts to random packet losses, congestion induced by traffic other than $T(S, D)$, path outages, etc. Congestion control on the flows of $T(S, D)$ would not be able to avoid such externally-imposed losses.

To achieve synergistic interaction, it is important to follow the *separation of timescales* principle. This means that the two controllers should operate with significantly different reaction timescales so that they do not compete with each other. Given that TCP's reaction timescale can be anywhere from few milliseconds up to almost a second (the RTT range for most TCP connections in the Internet), IRC's routing period T_r should be larger than that. On the other hand, however, it is desirable to keep T_r as low as possible so that IRC can provide fast recovery from short-term impairments [4]. Given this trade-off, we propose that T_r is more than one second and less than 3-5 seconds. With such separation of timescales, TCP gets the opportunity to react to sporadic packet losses first. If TCP does not manage to eliminate the losses during a *measurement period* of $T_m (< T_r)$ seconds, which takes place

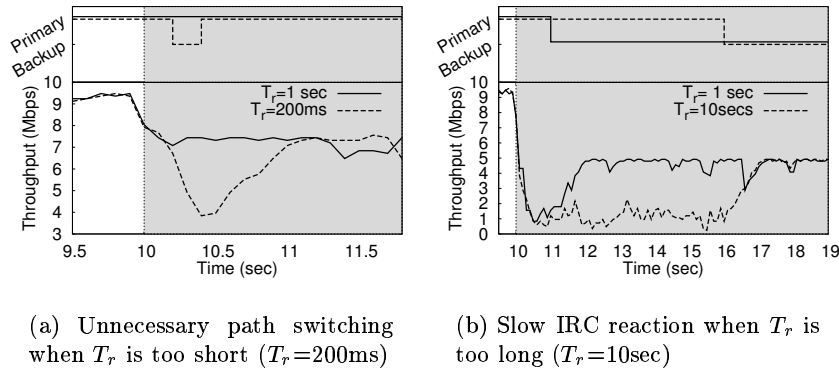


Fig. 2. The effect of the IRC routing period T_r .

at the end of the routing period, IRC gets activated and switches $T(S, D)$ to another path. IRC can be synergistic to TCP as long as the new path provides higher TCP throughput than the current (lossy) path.

In Figure 2, we illustrate what can happen when T_r is either too short or too long. In Figure 2(a), we compare the aggregate throughput of four TCP flows in $T(S, D)$ when T_r is set to 1 second (recommended value) with $T_r=200$ msec. The latter is close to TCP's reaction time in this path, given that the RTT is about 60 msec. In both cases we set the measurement period to $T_m=200\text{msec}$. At $t=10\text{sec}$, we introduce a congestion event at the primary path caused by CBR cross traffic. Without IRC, or with $T_r=1\text{sec}$, the cross traffic causes some congestive losses at the primary path, TCP decreases its offered load, but in this case, the throughput of $T(S, D)$ in the primary path is still higher than in the backup path. When T_r is set to 200msec, IRC is quick to detect packet losses and it switches the traffic to the backup path before TCP can adjust its offered load. Note that TCP's throughput is penalized by this unnecessary path switching.

On the other hand, a very long routing period is not desirable either. As shown in Figure 2(b), when the primary path suffers from random packet drops with 10% loss rate, IRC should switch the traffic as soon as possible because TCP cannot avoid this path impairment by decreasing its offered load. For $T_r=10\text{sec}$, it takes much longer to detect the impairment and react to it, compared to $T_r=1\text{sec}$, causing a significant throughput penalty.

Following the previous guidelines, in the rest of the paper we set the routing period to $T_r=1\text{sec}$. The IRC measurement period T_m covers the last 400msec of the routing period.

3 TCP and RTT changes

In this section, we focus on the impact of sudden RTT changes, due to IRC path switching, on TCP performance. Intuitively, an abrupt RTT decrease can cause packet reordering and the activation of Fast-Retransmit, while a significant RTT increase can cause the expiration of the Retransmission TimeOut (RTO). In this the next section, we consider an *ideal IRC* system that can detect impairments at the primary path without latency or error. The effect of IRC measurement delays and errors is studied in §5.

Suppose, without loss of generality, that the RTT of the primary path is lower, i.e., $RTT_p < RTT_b$. Note that the reverse path, from \mathcal{D} to \mathcal{S} , is the same in both paths (IRC cannot control the incoming traffic), and so the RTT difference is equal to the difference of the One-Way Delays (OWD) in the forward path from \mathcal{S} to \mathcal{D} . Let OWD_p and OWD_b be the OWDs in the primary and backup paths, respectively. In the following simulations, we limit the advertised window of the TCP flows that form $T(S, D)$ so that the traffic aggregate cannot saturate either path. Consequently, any reduction in the TCP throughput is due to RTT changes rather than congestion control.

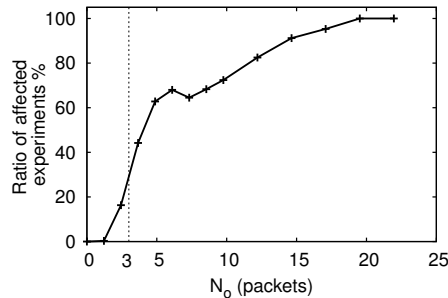


Fig. 3. Switching to a lower RTT path can cause Fast-Retransmit.

Switching to lower RTT path: First, consider the case that IRC switches the traffic from the backup to the primary path, i.e., all TCP flows in $T(S, D)$ experience a sudden RTT decrease. The first few packets sent through the primary path will arrive at the destination Out-Of-Order (OOO), because they reach the receiver before the last few packets sent through the backup path. For each OOO packet, the TCP receiver sends a Duplicate-ACK (DUPACK) to the sender. Typically, three consecutive DUPACKs trigger a Fast-Retransmit, and the congestion window is reduced by 50%. This reduction can cause a reduction in the TCP throughput, as a result of the IRC path change.

The number of OOO packets (N_o) depends on the OWD difference between the two paths. Specifically, a first-order estimate of the number of OOO packets in a TCP connection after the path change is $N_o = K(OWD_b - OWD_p)$,

where K is the throughput (in packets per second, pps) of the connection just before the path change. We expect that Fast-Retransmit will take place when $N_o \geq 3$. Since K varies significantly, however, Fast-Retransmit may occur for even lower OWD differences. So, to avoid throughput loss due to Fast-Retransmit, IRC can attempt to avoid switching to a path of lower RTT when the OWD difference is more than approximately $3/K$ seconds. For example, if the target per-connection throughput is 1Mbps and the packet size is 1500 bytes, then $K \approx 83$ pps, and Fast-Retransmit will probably happen if the OWD difference is more than about 35msec.

To simulate such path changes, we fixed RTT_p to 40msec and varied RTT_b from 40msec to 200msec. For each value of RTT_b , we conducted 1000 independent simulations of path switching from the backup to the primary path, where $T(S, D)$ consists of four TCP connections. In each simulation, we examined whether the path switching was followed by a throughput decrease due to Fast-Retransmit. The results are shown in Figure 3. The y-axis is the percentage of simulations in which we observed a throughput reduction, and the x-axis is an estimate of N_o . Note that Fast-Retransmit starts even before we reach the threshold of three packets. *The threshold $N_o=3$ is a reasonable rule of thumb, however, as it results in a throughput decrease in more than 30% of the simulations.*

Switching to higher RTT path: We next consider the case that IRC switches the traffic from the primary to the backup path, i.e., all TCP flows in $T(S, D)$ experience a sudden RTT increase. Such events can cause the expiration of the RTO timer even though there was no packet loss. The RTO expiration is followed by a reduction of the congestion window to one segment and by Slow-Start. The RTO timer is determined by a running-average of the observed RTTs, also taking into account the variability of the measured RTTs (see [9] for details). The RTO timer has a fixed OS-specific lower bound, which is often set to 200msec or 1sec [8].

When IRC switches to the backup path, the RTO of the ongoing connections is either based on the fixed minimum-RTO (say 200 msec) or the RTO of the primary path RTT_p . If RTT_b is larger than the minimum of these two values, then the path switching event will be followed by the expiration of the RTO timer for all ongoing TCP flows. This event can cause major throughput loss, and IRC systems should avoid it, if possible. *A practical guideline is to avoid switching to a path in which the RTT is larger by 200 msec or more.* We have conducted similar simulations as in the previous paragraphs, and the results (not presented here) confirm this analysis.

4 IRC and TCP congestion control

TCP congestion control is designed to adjust the send-window based on the available bandwidth (avail-bw) in the connection's path. However, the adjustment is gradual, hence a TCP connection will not be able to just "jump" to

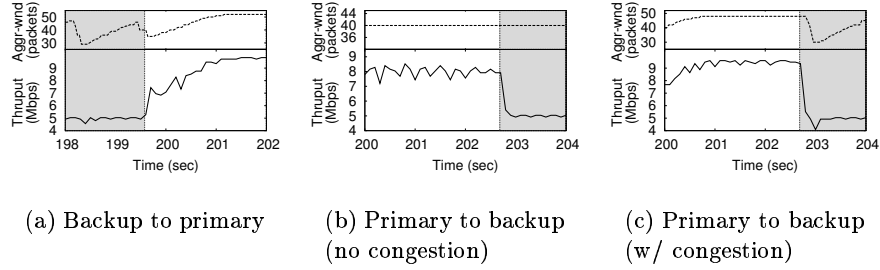


Fig. 4. Congestion window and throughput variations upon path changes.

the appropriate sending rate after IRC has switched to a path with lower or higher avail-bw. In this section, we focus on the behavior of congestion control upon IRC path changes. We consider two cases, depending on whether IRC switches to a path with higher avail-bw (backup to primary transition) or to a path with lower avail-bw (primary to backup transition).

Switching to higher avail-bw path: When a TCP flow moves from the backup to the primary path, its throughput will change from $R_b = W_b/RTT_b$ to $R_p = W_b/RTT_p$, where W_b is the send-window before the path change. Given that the primary path provides higher avail-bw than the backup path, the connection can now increase its window. This window increase process will typically be linear, assuming the connection was in congestion avoidance before the path change.

We show such an event in the simulation of Figure 4(a). The top part of the plot shows the aggregate window of four TCP flows, while the bottom part shows their aggregate throughput. The shaded part of the graph covers the time period when IRC uses the backup path, while the rest covers the primary path period. Note the rather slow increase of the aggregate window and throughput after the path change. The rate of this increase depends on the throughput difference $C_p - R_p$, the number of ongoing flows, and their RTT. With N connections in congestion avoidance, and with a packet size L , $T(S, D)$ will utilize the additional capacity in $RTT_p \frac{C_p - R_p}{LN}$ seconds, if the window of each connection increases by one packet per RTT (i.e., ignoring the effect of delayed ACKs).

Switching to lower avail-bw path: Suppose that IRC has detected an impairment at the primary path and it switches $T(S, D)$ to the backup path. We need to consider two cases, depending on whether $T(S, D)$ will cause congestion in the backup path or not.

In the first case, $T(S, D)$ does not cause congestion at the backup path. This will happen if the offered load in $T(S, D)$ is lower than the avail-bw in the backup path, or the buffer space at the bottleneck link of the backup path is sufficiently large to hold the excess packets. This scenario is shown in

Figure 4(b). Here, we have four connections that are limited by the advertised window when they use the primary path ($R_p=8\text{Mbps}$), and that saturate the backup path ($C_b=5\text{Mbps}$) without causing congestion. After the path change, the throughput of $T(S, D)$ drops immediately to C_b without causing packet drops, because the bottleneck link has sufficiently large buffers. TCP congestion control is not activated after such path changes, simply because there is no congestion when the traffic is switched to the lower avail-bw path.

In the second case, $T(S, D)$ does cause congestion at the backup path. In that case, one or more TCP connections experience packet loss after the path switching event, and there is a time period in which the aggregate throughput is less than C_b . Figure 4(c) shows an example. Note that the transition to the backup path is followed by a reduction in the aggregate congestion window and in the throughput at the backup path. Again, the duration of this transient effect depends on the number of ongoing flows, their RTT, and C_b .

The key point of this section is that IRC path changes can move a TCP aggregate to a path with different (higher or lower) avail-bw. It should be expected that the switched traffic will need some time to adapt to this avail-bw. Consequently, IRC will need to wait for the completion of such transient throughput variations before considering switching to another path.

5 IRC measurement latency

In the previous two sections, we considered an *ideal-IRC* model that can instantaneously detect the start and end of an impairment period in the primary path. In practice, measurements take some time and they are error-prone because they rely on sampling and inference. In this section, we compare the ideal-IRC model with a *practical-IRC* model that relies on ping-like active probing to detect packet losses in the primary and backup paths. Specifically, our practical-IRC model has a routing period T_r of 1 second, a measurement period T_m of 400 msec (covering the last 40% of the routing period), and it generates one probing packet (64 bytes) per 10msec to detect packet losses in each path.

We examine two types of impairment in the primary path: outages and random packet losses. In both types, the primary path alternates between “good” periods of average duration T_g and “bad” periods of average duration T_b . Both durations are exponentially distributed. During an outage, all packets sent to the primary path are lost. With random losses, packets are dropped with probability p . In the following simulations, $T_g=100$ sec and $T_b=50$ sec, unless noted otherwise.

In the simulations presented in this section, the traffic in $T(S, D)$ consists of non-persistent TCP flows. Specifically, 100 users generate transfers with Pareto distributed sizes (shape parameter = 1.8), and so the aggregate traffic is Long-Range Dependent. After each transfer, a user stays idle for an exponentially long “thinking time”; the user then returns to the network with a

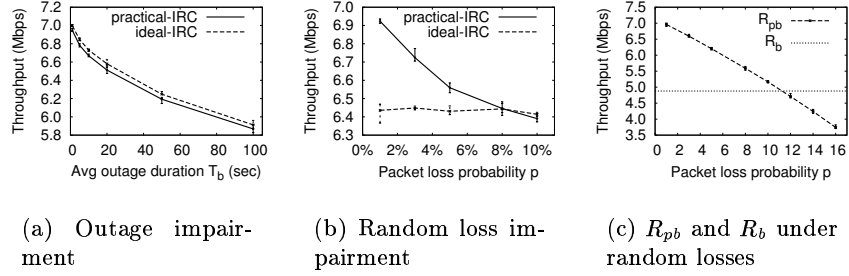


Fig. 5. Comparison of ideal-IRC with practical-IRC.

new transfer. The average throughput of the traffic aggregate in the primary path during the good periods is $R_{pg} \approx 7\text{Mbps}$, while the average throughput in the backup path is $R_b \approx 5\text{Mbps}$.

Outage impairments: Figure 5(a) compares ideal-IRC with practical-IRC in the case of outages for various values of the average outage duration T_b . As expected, ideal-IRC does better than practical-IRC for all values of T_b , but the absolute throughput difference is not significant. To better understand the difference between the two models, we need to consider the measurement latency that is introduced by practical-IRC to detect an impairment.

Specifically, T_{mb} is the latency to detect the impairment of the primary path and switch to the backup path, while T_{mg} is the latency to detect the restoration of the primary path and switch to that path. R_{pg} is the average throughput of $T(S, D)$ on the primary path during “good” periods, R_{pb} is the average throughput on the primary path during “bad” periods (which is zero in the case of outages), and R_b is the average throughput in the backup path.

Thus, the average throughput of $T(S, D)$ with ideal-IRC is

$$R_I = \frac{(R_b T_b + R_{pg} T_g)}{(T_b + T_g)} \quad (1)$$

and the average throughput with practical-IRC is

$$R_P = \frac{R_b(T_b - T_{mb} + T_{mg}) + R_{pg}(T_g - T_{mg}) + R_{pb}T_{mb}}{T_b + T_g} \quad (2)$$

Ideal-IRC performs better than practical-IRC, i.e., $R_I > R_P$, if and only if

$$\frac{R_{pb} - R_b}{R_{pg} - R_b} < \frac{T_{mg}}{T_{mb}} \quad (3)$$

Given that $R_{pg} - R_b$ and T_{mg}/T_{mb} are positive, ideal-IRC is better if $R_{pb} < R_b$, independent of the measurement latencies T_{mb} and T_{mg} . In other words, the ideal-IRC model is better if the backup path gives higher throughput than

the primary during bad periods. For outages, since R_{pb} is zero, condition (3) is always true.

Random loss impairments: Figure 5(b) compares ideal-IRC with practical-IRC in the case of random packet losses with probability p . Note that, rather surprisingly, practical-IRC performs better than ideal-IRC when the loss rate is less than 10%. The reason becomes clear from the previous analysis. If $R_{pb} < R_b$, ideal-IRC performs better than practical-IRC independent of the measurement latencies. Figure 5(c) compares R_{pb} and R_b for different values of p . Note that the throughput in the primary path during “bad” periods is still higher than the throughput in the backup path, if the loss rate is less than about 12%.

An important conclusion from this analysis is that a lossless path is not necessarily better than a lossy path. Consequently, an IRC system that always switches to a path with lower loss rate may fail to provide the maximum possible TCP throughput to its users. An improved IRC system can use TCP throughput as the primary performance metric for path selection. That is, IRC should be able to measure the TCP throughput in the current path, estimate or predict the TCP throughput in other paths, and then switch based on throughput comparisons. The problem of predicting TCP throughput has been the focus of [6].

6 To IRC or not to IRC?

We can now focus on the following key question: Under which conditions is IRC beneficial to TCP traffic, in terms of the aggregate resulting throughput? Specifically, we compare the TCP throughput of the aggregate $T(S, D)$ with IRC control (practical-IRC) and without IRC control (“no-IRC”). The latter means that $T(S, D)$ stays at the primary path independent of whether that path is lossy or not.

Outage impairments: Figure 6(a) compares practical-IRC and no-IRC in the case of outages. We observe that the throughput of practical-IRC is always higher, indicating that IRC is always beneficial and synergistic to TCP when dealing with outages. The following analysis, however, reveals that under a certain condition IRC may not be beneficial even for outages. First, it is easy to see that the average throughput with the no-IRC model is

$$R_N = \frac{R_{pb}T_b + R_{pg}T_g}{T_b + T_g} \quad (4)$$

The no-IRC model performs better than the practical-IRC model, i.e., $R_N > R_P$, when

$$\frac{R_b - R_{pb}}{R_{pg} - R_b} < \frac{T_{mg}}{T_b - T_{mb}} \quad (5)$$

Note that $R_{pg} > R_b$ and $T_b > T_{mb}$. For outages, since $R_{pb} = 0$, (5) can be written as

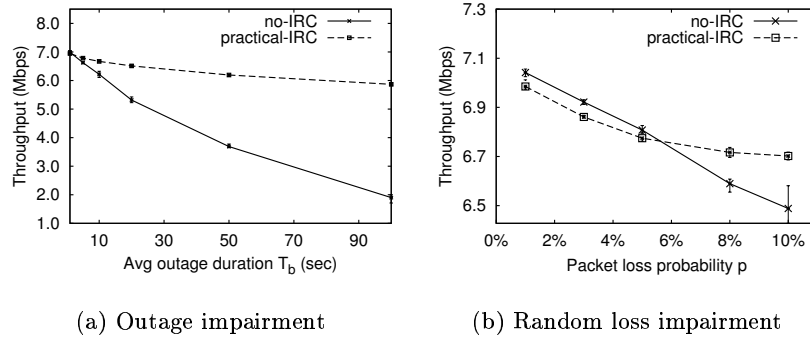


Fig. 6. Comparison of practical-IRC with static routing (no-IRC).

$$R_b(T_b - T_{mb}) < (R_{pg} - R_b)T_{mg} \quad (6)$$

This inequality indicates that if the throughput loss during the periods T_{mg} (right hand side) is larger than the throughput gain during the periods $T_b - T_{mb}$ (left hand side), IRC is not beneficial. This can be the case when the outage duration T_b is comparable to the measurement latency T_{mb} , for instance. Therefore, when (6) is true, it is better to keep the traffic at the primary path, instead of switching to the backup path.

Random loss impairments: Figure 6(b) shows the corresponding results for random losses. The capacity of the backup path is $C_b=6$ Mbps, in this case, to illustrate that practical-IRC can be better than no-IRC when the loss rate p is larger than a certain value. Note that for low values of p , less than 5% in these simulations, static routing appears slightly better. The analysis of the previous paragraph still applies, and condition (5) determines whether practical-IRC is better than no-IRC. Note that $R_{pb} > 0$ in this case.

In summary, the simulation and analytical results of this section show that IRC should not switch paths simply based on the detection of some packet losses in the primary path. Major losses or outages should trigger a path change if they persist for more than the measurement latency. For sporadic losses, on the other hand, the key question is whether the measured throughput in the primary path in the presence of such losses is lower than the predicted throughput in the backup path. If that is *not* the case, then IRC should stay in the current path.

7 Related work and summary

A number of papers (see [1] and references therein) explored the performance and availability benefits of IRC systems. The work by Tao et al. [10] evaluated IRC experimentally, also comparing it with overlay routing. Recently,

two papers investigated the risk of oscillations when different IRC systems get synchronized [4, 7]. [7] focused on exploring the conditions under which synchronization may occur, while [4] proposed randomized IRC path switching techniques to avoid synchronization. The interactions of TCP congestion control with adaptive routing have been studied from a more theoretical perspective in [2]. The most related paper to our work is [5], which focused on the stability of intradomain traffic engineering when the traffic is generated from persistent TCP connections.

In conclusion, the main points of this paper are: IRC can be synergistic to TCP if the measurement timescale is larger than TCP's typical RTT or RTO; IRC needs to examine the RTT difference between two candidate paths to avoid retransmissions and throughput loss; IRC should expect a throughput transient period after most path changes; switching paths based on loss rate comparisons can lead to throughput loss; IRC is synergistic to TCP for outages that last more than the measurement timescale and for significant loss rates.

References

1. A. Akella, B. Maggs, S. Seshan, A. Shaikh, and R. Sitaraman. A Measurement-Based Analysis of Multihoming. In *Proceedings of ACM SIGCOMM*, 2003.
2. E. J. Anderson and T. E. Anderson. On the Stability of Adaptive Routing in the Presence of Congestion Control. In *IEEE INFOCOM*, Apr. 2003.
3. Cisco Systems. Optimized Edge Routing. http://www.cisco.com/en/US/networksol/ns471/networking_solutions_package.html.
4. R. Gao, C. Dovrolis, and E. Zegura. Avoiding Oscillations due to Intelligent Route Control Systems. In *Proceedings of IEEE INFOCOM*, 2006.
5. J. He, M. Bresler, M. Chiang, and J. Rexford. Towards Multi-Layer Traffic Engineering: Optimization of Congestion Control and Routing. *IEEE Journal on Selected Areas in Communications*, 2007.
6. Q. He, C. Dovrolis, and M. Ammar. On the Predictability of Large Transfer TCP Throughput. In *Proceedings of ACM SIGCOMM*, 2005.
7. R. Keralapura, C. Chuah, N. Taft, and G. Iannaccone. Race Conditions in Coexisting Overlay Networks. *IEEE/ACM Transactions on Networking*, 2006.
8. A. Medina, M. Allman, and S. Floyd. Measuring the Evolution of Transport Protocols in the Internet. *SIGCOMM Comput. Commun. Rev.*, 35(2):37–52, 2005.
9. V. Paxson and M. Allman. RFC2988: Computing TCP's Retransmission Timer, 2000.
10. S. Tao, K. Xu, Y. Xu, T. Fei, L. Gao, R. Guerin, J. Kurose, D. Towsley, and Z. Zhang. Exploring the Performance Benefits of End-to-End Path Switching. In *Proceedings of IEEE ICNP*, 2004.