INTERDOMAIN TRAFFIC ENGINEERING FOR
MULTI-HOMED NETWORKS

A Thesis
Presented to
The Academic Faculty

by

Ruomei Gao

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
College of Computing

Georgia Institute of Technology
December 2007
INTERDOMAIN TRAFFIC ENGINEERING FOR
MULTI-HOMED NETWORKS

Approved by:

Dr. Constantine Dovrolis, Advisor
College of Computing
Georgia Institute of Technology

Dr. Ellen Zegura, Advisor
College of Computing
Georgia Institute of Technology

Dr. Mostafa Ammar
College of Computing
Georgia Institute of Technology

Dr. Nick Feamster
College of Computing
Georgia Institute of Technology

Dr. Henry Owen
School of Electrical and Computer
Engineering
Georgia Institute of Technology

Date Approved: 2007
To my parents,
for their unconditional love and support.

To my sister,
for her help and guidance.
ACKNOWLEDGEMENTS

I am in debt to many people around me, without whose help and support, I would not be able to reach this point of my Ph.D study.

First, I would like to express my greatest gratitude to my advisors Dr. Ellen Zegura and Dr. Constantine Dovrolis. Ellen led me into networking research, allowed me the freedom to explore, encouraged me at my down time, provided insightful advices on my projects. She always has been a resourceful, encouraging, and caring guide in my Ph.D years.

In my last few years of my Ph.D. study, I am fortunate enough to work with Constantine, and all of my thesis work is done with him. With Constantine, I experienced a totally different style of guidance. His exuberant enthusiasm always inspires my passion for our work, and his uncompromising pursue to quality research and rigorous details always pushes me forward.

I would also like to thank my committee, Dr. Mostafa Ammar, Dr. Nick Feamster, and Dr. Henry Owen for their interests in the thesis and insightful comments. In particular, I would like to thank Mostafa for his advice on selecting the last piece of my thesis work, and Nick for providing me the access to his machines.

I am grateful to my mentors of my internships, Dr. Tim Griffin, Andre Broido, and Dr. kc claffy. They not only advised me on my research project during my internship, they also exposed me to a wider research community and enriched my Ph.D. experience. During my years in College of Computing, I am also fortunate to work with Dr. Milena Mihail and Dr. Chuanyi Ji. I learned from Dr. Milena Mihail helped me on networking research related topics on graph theories. Dr. Chuanyi Ji provided great help through her expertise on anomaly detection.
I wouldn’t have finished my long journey of Ph.D study without my fellow graduate students. Li Zou helped me through my early Ph.D. years. Shashidhar Merugu kindly provided me advices on writing and research. Qi He always generously offers me help, advices, and encouragement, during her years in Georgia Tech as well as after her graduation. Manish Jain has been a great source for help and encouragement, especially in the last year of my study. Amogh DhamDhere has been a great labmate and make the finishing year more endurable. I am also grateful to have the fellowship and friendship of Yang Chen, Jinliang Fan, Meng Guo, Hyewon Jun, Pradnya Karbhari, Abishek Kumar, Ravi Prasad, Srinivasan Seetharaman, Sridhar Srinivasan, Wenrui Zhao, Qi Zhao, Yong Zhu.

At last, I would like to thank my family, to whom this thesis is dedicated. They always support and encourage me. My special thanks goes to my sister. She takes on the responsibilities more than a elder sibling: she guides me, helps me, and pushes me forward. I am grateful for their love, patience and encouragement.
# TABLE OF CONTENTS

DEDICATION .................................................................................................................. iii

ACKNOWLEDGEMENTS ................................................................................................. iv

LIST OF TABLES ................................................................................................................ x

LIST OF FIGURES ............................................................................................................. xi

SUMMARY ........................................................................................................................... xiv

I  INTRODUCTION ........................................................................................................... 1

1.1 Traffic Engineering .................................................................................................... 1

   1.1.1 Interdomain Egress Traffic Engineering ......................................................... 4

   1.1.2 Interdomain Ingress Traffic Engineering ....................................................... 5

1.2 Challenges .................................................................................................................. 7

1.3 Contributions .............................................................................................................. 9

1.4 Thesis Organization .................................................................................................. 11

II  RELATED WORK ......................................................................................................... 12

2.1 Interdomain Egress Traffic Engineering ................................................................. 13

  2.1.1 Intelligent Route Control .................................................................................. 14

  2.1.2 Interaction between Routing Systems ............................................................. 15

  2.1.3 Traffic Engineering with Cooperations .......................................................... 18

  2.1.4 Other Approaches to interdomain TE ............................................................. 19

2.2 Interaction between TCP Congestion Control and Traffic Engineering .................. 19

2.3 Interdomain Ingress Traffic Engineering .................................................................. 20

  2.3.1 BGP Community Attributes ............................................................................ 22

  2.3.2 AS-Path Prepending ....................................................................................... 23

III  AVOIDING OSCILLATIONS DUE TO INTELLIGENT ROUTE CONTROL SYSTEMS 25

  3.1 Introduction ............................................................................................................. 25
3.2 IRC model .............................................................. 27
3.2.1 Network model ..................................................... 27
3.2.2 Traffic model ........................................................ 29
3.2.3 IRC processes ....................................................... 30
3.2.4 Measurement process ............................................. 31
3.2.5 Performance estimation .......................................... 32
3.2.6 Routing decision and path switching ......................... 33
3.3 IRC-induced traffic oscillations ................................. 34
3.3.1 Self-load effect .................................................... 34
3.3.2 Synchronization of IRC systems ............................... 36
3.4 Randomized IRC algorithms ....................................... 38
3.4.1 Deterministic Path Switching (DPS) ......................... 39
3.4.2 Fixed Switching Probability (FSP) ........................... 39
3.4.3 Adaptive Switching Probability (ASP) ....................... 39
3.4.4 Random Routing Period (RRP) ............................... 39
3.4.5 Hysteresis Routing Period (HRP) ............................ 40
3.5 Evaluation of IRC algorithms - stationary load ............. 40
3.5.1 Simulation setup .................................................. 41
3.5.2 Performance of FSP .............................................. 42
3.5.3 Parameterization of ASP, RRP and HRP ................. 46
3.5.4 Comparison of DPS, FSP, ASP, RRP and HRP ........... 47
3.5.5 Summary ........................................................... 47
3.6 Evaluation of IRC algorithms - nonstationary load ........ 47
3.6.1 Simulation setup .................................................. 49
3.6.2 Performance of FSP under nonstationary load ........... 51
3.6.3 Comparison of FSP, ASP, RRP and HRP under nonstationary load .................................................. 53
3.6.4 Summary ........................................................... 54
3.7 Heterogeneous IRC sources ....................................... 55
3.7.1 Heterogeneous IRC algorithms .................................. 55
3.7.2 Coexistence of deterministic and randomized path switching 55
3.8 Summary ..................................................................... 57

IV INTERACTIONS OF INTELLIGENT ROUTE CONTROL WITH TCP CONGESTION CONTROL ............................................ 58
4.1 Introduction/Model ..................................................... 59
4.2 Separation of timescales .............................................. 61
4.3 TCP and RTT changes ............................................... 63
  4.3.1 Switching to lower RTT path ................................. 64
  4.3.2 Switching to higher RTT path ............................... 66
4.4 IRC and TCP congestion control ................................. 67
  4.4.1 Switching to higher avail-bw path ......................... 67
  4.4.2 Switching to lower avail-bw path ......................... 68
4.5 IRC measurement latency ......................................... 70
  4.5.1 Outage impairments ............................................ 71
  4.5.2 Random loss impairments .................................... 72
4.6 To IRC or not to IRC? .............................................. 73
  4.6.1 Outage impairments ............................................ 74
  4.6.2 Random loss impairments .................................... 74
4.7 Summary ..................................................................... 76

V INTERDOMAIN INGRESS TRAFFIC ENGINEERING THROUGH OPTIMIZED AS-PATH PREPENDING ........................................ 77
5.1 Problem Statement and Formulation ............................ 78
  5.1.1 COP Problem Statement ...................................... 78
  5.1.2 Remarks .......................................................... 84
5.2 Optimal Padding Vector Algorithm ............................ 84
  5.2.1 Optimal Padding Vector (OPV) algorithm ................ 84
  5.2.2 Properties of the OPV algorithm ............................ 86
5.3 Estimation of Input Parameters .................................... 90
5.3.1 Selection of super-sources ........................................ 91
5.3.2 Estimation of AS-Path length matrix ............................ 91
5.4 Robust Padding Vector Algorithm ................................. 95
  5.4.1 Robust Padding Vector Algorithm .............................. 95
  5.4.2 Robustness evaluation ........................................ 97

VI CONTRIBUTIONS AND FUTURE WORK ............................ 102
  6.1 Summary of Contributions ...................................... 102
  6.2 Future Work ..................................................... 104

REFERENCES .......................................................... 108
LIST OF TABLES

1  Main notations. ................................................................. 78
2  Criteria for BGP best route selection. ................................. 81
3  Probability (%) of AS-Path length estimation error \(e\), conditioned on the estimated AS-Path length \(\hat{p}\) (both \(e\) and \(\hat{p}\) measured in AS hops). 94
LIST OF FIGURES

1 Internet traffic engineering: intradomain versus interdomain, ingress versus egress. ........................................... 2
2 Network model. ............................................................... 28
3 The timeline of an IRC routing period. ............................... 31
4 Path switching events (top and middle graphs) and loss rate (bottom graph) when using delay and loss rate versus available bandwidth measurements. ........................................... 35
5 Oscillations due to synchronization of different IRC systems. .... 37
6 RRP and HRP algorithms. .................................................. 40
7 Loss rate of FSP as a function of $P$ (four ingress links) with homogeneous flow rates. ........................................... 43
8 Loss rate of FSP as a function of $P$ (four ingress links), and flow rates follows Zipf distribution. ................................... 44
9 Switching frequency with FSP as a function of $P$ (four ingress links and homogeneous flow rate distribution). .............. 44
10 Ratio $F/P$. ................................................................. 45
11 Comparison of IRC algorithms– loss rate (four ingress links). ... 48
12 Comparison of IRC algorithms– switching frequency (four ingress links, homogeneous flow rates). .................................. 49
13 The pattern of periodic load changes between two ingress links. At $t1$ a block of DRC traffic moves from the right link to the left. At $t3$, $T_w$ time units later, an equally sized block moves back to the right link. 50
14 Loss rate of FSP for different values of $P$, $T_w \in [0, 5]$. ........... 51
15 Loss rate of FSP for different values of $P$, $T_w \in [0, 30]$. ........... 52
16 Loss rate of FSP for different values of $P$, $T_w \in [0, 400]$. .......... 52
17 Loss rate with FSP, ASP, RRP and HRP as we vary $T_w$. ............ 54
18 Loss rate when FSP and DPS traffic coexist (four ingress links, homogeneous flow rates). ........................................... 56
19 The dual-feedback model. .................................................. 59
20 The effect of the IRC routing period $T_r$: Unnecessary path switching when $T_r$ is too short ($T_r=200\text{ms}$) ....................... 63

xi
21 The effect of the IRC routing period $T_r$: Slow IRC reaction when $T_r$ is too long ($T_r = 10\text{sec}$) ................................................................. 64
22 Switching to a lower RTT path can cause Fast-Retransmit. ....... 65
23 Congestion window and throughput variations when traffic switch from the backup path to the primary path. ........................................... 67
24 Congestion window and throughput variations when traffic switch from the primary path to the backup path without induce congestion on the backup path. ................................................................. 69
25 Congestion window and throughput variations when traffic switch from the primary path to the backup path inducing congestion on the backup path. ................................................................. 69
26 Comparison of ideal-IRC with practical-IRC under outage impairment. 71
27 Comparison of ideal-IRC with practical-IRC under random loss impairment. ................................................................. 72
28 $R_{pb}$ and $R_b$ under random losses under random loss impairment ... 73
29 The performance of practical-IRC with static routing (no-IRC) under outage impairment. ................................................................. 75
30 The performance of practical-IRC with static routing (no-IRC) under random loss impairment. ................................................................. 75
31 High-level architecture of ingress interdomain traffic engineering. . . 79
32 Branching networks, labeled B in the diagram, in the upstream cloud $U_i$ ........................................................................................................ 80
33 Four estimation techniques for the AS-Path length. ...................... 92
34 Empirical probability distribution function of estimated AS-Path length. 94
35 The feasibility of uniform random distributed source load. Note that the CDF curves that are not visible are equal to 100% for all instances. 97
36 The feasibility of exponentially random distributed source load. Note that the CDF curves that are not visible are equal to 100% for all instances. ................................................................. 98
37 The feasibility of source loads with Pareto distribution. Note that the CDF curves that are not visible are equal to 100% for all instances. . 99
38 The robustness of uniform random distributed source load. Note that the CDF curves that are not visible are equal to 100% for all instances. 100
39  The robustness of exponentially random distributed source load. Note that the CDF curves that are not visible are equal to 100% for all instances. .................................................. 100
40  The robustness of source loads with Pareto distribution. Note that the CDF curves that are not visible are equal to 100% for all instances. .................................................. 101
41  Design of experiments on IRC emulation in the Internet. ............... 105
42  Experiment setup for IRC interaction study ....................................... 106
SUMMARY

Interdomain traffic engineering (TE) controls the flow of traffic between autonomous systems (ASes) to achieve performance goals under various resource constraints. Interdomain TE can be categorized into ingress TE and egress TE, which aim to control the ingress and egress traffic flow in a network, respectively. Most interdomain TE techniques are based on BGP, which was not designed to support performance-based routing. Hence even though some basic interdomain TE techniques are widely deployed, their overall effectiveness and impact on interdomain traffic are not well understood. Furthermore, systematic practices for deploying these techniques have yet to be developed.

In this thesis, we explore these open issues for both ingress and egress TE. We first focus on the AS-Path prepending technique in interdomain ingress TE. We design a polynomial algorithm that takes network settings as input and produces the optimal prepending at each ingress link. We also develop methods to measure the inputs of the optimal algorithm by leveraging widely available looking glass servers and evaluate the errors of such measurement. We further propose an algorithm, based on this optimal algorithm, that is robust to input errors.

We then focus on Intelligent Routing Control (IRC) systems often used at multi-homed networks for egress interdomain TE. To address the possible traffic oscillation problem caused by multiple IRC systems, we design a class of randomized IRC algorithms. Through simulations, we show that the proposed algorithms can effectively mitigate oscillations. We also show that IRC systems using randomized path switching algorithms perform better than those switching path deterministically, when both types of IRC systems co-exist.

xiv
To further understand the performance impact of IRC systems, we next focus on the performance of applications, such as TCP connections. We study the synergistic and antagonistic interactions between IRC and TCP connections, through a simple dual-feedback model. We first examine the impact of sudden RTT and available bandwidth changes in TCP connection. We then examine the effect of IRC measurement delays on closed loop traffic. We also show the conditions under which IRC is beneficial under various path impairment models.
CHAPTER I

INTRODUCTION

Since the commercialization of the Internet, it has grown rapidly, both in terms of the number of the networks connected to the Internet [32] and the volume of the traffic it carries [47]. Such growth puts on great pressure on existing network resources (e.g. the capacities of links and routers). The growing number of networks adds complexity and unpredictability to traffic flow and the increasing volume of traffic leads to high variance of traffic rate. An ultimate solution for such growth is to upgrade network facilities. However, upgrading network facilities, involving leasing or laying network fibers with higher capacity and purchasing more powerful routers, is a lengthy and expensive process. Thus, before upgrading network facilities, networks strive to keep up with the growth of Internet by means of traffic engineering.

1.1 Traffic Engineering

Traffic engineering (TE) refers to the design and implementation of controls that affect the flow of traffic. TE is used to achieve three major goals: load balancing, performance optimization, and cost reduction. Load balancing refers to the operation that moves traffic flow from highly utilized links to alternative links with lower utilization rate, thus it reduces “hot spots” in the network and avoids overloading links. Performance optimization can be achieved by routing traffic to an alternative path providing better performance than the original one, and it can also be achieved by redistributing traffic within one network so that the overall performance is improved. Similarly, cost reduction is done by routing traffic to a path with lower operation cost.
TE can be broadly divided into two types, based on the control level of the traffic flow: *intradomain* and *interdomain*. Figure 1 illustrates both types of TE, with intradomain TE at the lower part and interdomain TE at the upper part. In intradomain TE [9], the operator of an autonomous system (AS) controls the flow of traffic within a target network to achieve the three goals of TE, introduced earlier. The control of traffic flow is often done through changing the configuration of the network, such as changing the link weights for OSPF or IS-IS, or through tunneling traffic at the MAC layer or network layer, such as MultiProtocol Label Switching (MPLS). An example of intradomain TE is illustrated in Figure 1. In this example, the target network, shown as a gray network cloud, has a traffic demand from the top ingress point on the left to the bottom egress point on the right. The diagram shows two alternative paths between the ingress point and the egress point, labeled as the two dotted arrow curves. For the target AS, intradomain TE, concerning this traffic demand, is to route traffic on either of the paths to meet the performance or cost requirement.

![Image of TE types](image-url)

**Figure 1:** Internet traffic engineering: intradomain versus interdomain, ingress versus egress.
Interdomain TE, on the other hand, aims to control the ingress and egress points of traffic flows when entering or leaving the network. Depending on the relative direction of the traffic under control, i.e. inbound traffic or outbound traffic, interdomain traffic engineering can be categorized into interdomain *ingress* traffic engineering and interdomain *egress* traffic engineering. For inbound traffic, the target AS tries to affect the routing of the traffic of the source network and other upstream networks, so that traffic enters the target network through the desired ingress point. For outbound traffic, the target AS selects an egress point and routes traffic so that it is transferred to the destination through the selected egress point. In the example shown in Figure 1, given the source network, at the far left, and the destination network, at the far right, the traffic flow can enter the target AS at three of the ingress points and exit at two of the egress points. The target AS applies ingress TE at the ingress points to influence the routing of this traffic flow at the source network and the intermediate networks along the paths, so that the traffic flow enters from the desired ingress point. For egress TE, the target AS selects the best egress point and simply routes the traffic flow using the same techniques in intradomain TE.

Interdomain TE is inherently more challenging than intradomain TE because the target AS has very limited tool for interdomain TE, has no full control of traffic flow, and has little knowledge of AS-level topology and routing in other networks. This thesis seeks to gain deeper understanding of interdomain TE, explore its impact, both beneficial and adverse, and improve the efficiency and effectiveness of existing interdomain TE techniques In the following subsection, we will briefly introduce most-used techniques in interdomain TE, list the problems and challenges lies within, and propose our solutions for these problems.
1.1.1 Interdomain Egress Traffic Engineering

Interdomain egress TE is to select the egress points to achieve the three goals of interdomain TE: load balancing, performance improvement, and cost reduction. The techniques used in interdomain egress TE are similar to those in intradomain TE, except that besides changing network configuration and traffic tunneling, changing announcement in iBGP is also used.

Two issues have been major focus of egress selection in interdomain egress TE: performance and stability. In interdomain egress TE, path diversity provided by multiple egress points can be used to improve performance of data transferring. The target network tries to achieve optimal performance by selecting egress points for traffic to a single or multiple destinations. Stability issue arises from changing the flow of traffic. The change of the flow of traffic can cause routing and traffic instability[67, 69, 28]. In addition, the interactions among networks practicing interdomain egress TE or the interaction between interdomain egress TE operations and other routing systems, such as overlay networks, can also cause instability. Even an inappropriately configured egress TE system can cause traffic oscillation by itself alone[26].

In recent years, interdomain egress TE becomes more of a commonplace, especially after the prevalence of network multihoming[13]. Multihoming refers to networks subscribing to more than one transit providers. Initially, multihoming is used to guarantee connectivity to the Internet. Later, multihoming has also been used as means of performance enhancement. Egress point selection in interdomain egress TE for multihomed networks is equivalent to upstream provider selection. Intelligent Route Control (IRC) products are designed for such tasks. IRC systems monitor path performance through active or passive measurement, and choose the best path for route. Details of IRC will be explained in Chapter 3.
1.1.2 Interdomain Ingress Traffic Engineering

Compared to interdomain egress TE, interdomain ingress TE is even more difficult. A target AS can not directly control the flow of inbound traffic, so it tries to influence the routing of upstream ASes, most time without much knowledge of the upstream topology and routing policies. Hence, instead of dealing with the route selection problem, interdomain ingress TE addresses the route enforcement problem, i.e., the problem of how to make the upstream networks route traffic to the desired ingress point.

Most techniques in interdomain ingress TE techniques are based on the interdomain routing protocol: Border Gateway Protocol (BGP)[58]. When using BGP-based techniques, the target AS changes the BGP routing advertisement to influence the BGP routing decisions in the upstream ASes, thus affecting the ingress point of the traffic flow. Commonly used techniques include selective announcements, use of the Multi-Exit Discriminator (MED) attribute, and labeling routes with special BGP communities, AS-Path prepending.

The selective announcement technique is that a multihomed AS announces its network prefixes (or part of the network prefixes) to a subset of its upstream providers. If the announcements are well designed, selective announcement can effectively control inbound traffic. For example, the target AS has a /16 prefix and two upstream providers A and B. To distribute the traffic on the two upstream providers, the target AS splits the /16 prefix into two /17 prefixes, and announces one prefix to each of the upstream respectively. This way, traffic to half of the addresses enters through upstream A, and traffic to the other half enters through upstream B. The /16 prefix can be still announced to both upstreams, thus if one ingress link fails, the other upstream can be used as backup.

MED attribute can be used between two ASes having multiple peering points between them. When advertising routes to the same destination at multiple peering
points, the target AS can use a lower MED at the preferred peering point. The neighbor AS will accordingly deliver traffic through the preferred peering point. Using MED attribute in ingress TE is effective, but can only be used at peering points between the same pair of ASes.

BGP community attribute can also be used to indicate preference of traffic ingress point, but don’t have the limitation of multiple peering point between the same pair of ASes. Besides communities that label route preference, interdomain ingress TE can also be done through communities that restrict the propagation of routes. For instance, the target AS can advertise a route to ISP A with a community meaning "stop propagation after three AS hops", but advertise the same route without this community to ISP B. This way, networks three hops away through ISP A will route traffic to the ingress point connected to ISP B.

The basic idea of AS-Path prepending is simple. During a route selection procedure, a sequence of BGP attributes are compared for all candidate routes. Starting from the first attribute, if a tie is found, the next attribute is used to break the tie. So on and so forth, until the best route is selected. AS-Path length has high priority in the list, the second attribute in order, just after the Local Preference attribute. That is, if the values of Local Preference are the same for all candidate routes, the route with shortest AS-Path will be chosen as the best route. An AS can take advantage of the high ranking of AS-Path length to influence the routing of other networks for interdomain ingress TE. Instead of labeling or announcing preferred routes, AS-Path prepending technique tries to push traffic away from non-preferred routes, e.g., overloaded or expensive ingress links. At a non-preferred ingress point, the target AS appends several times, instead of appending the AS number once to the AS-Path as in normal BGP operations. The prepending makes the non-preferred routes longer, thus less likely to be selected as the best route for data transferring. AS-Path prepending technique is one of the most used techniques in interdomain ingress TE, for it is easy
to deploy and needs no special co-ordination from other networks.

Some techniques in interdomain ingress TE are not BGP-based. For instance, interdomain ingress TE can be done through DNS and NAT. When DNS is used for interdomain ingress TE, the target network announces a different prefix to each upstream. Through DNS, the target AS can dynamically associate a DNS name to IP addresses corresponding to any of the upstream providers. This way, DNS is able to control ingress traffic at the flow level. Similarly, NAT can translate a local address to the address associated with routers connecting to different upstream providers. Ingress TE with DNS or NAT requires complicated configurations of DNS or NAT.

1.2 Challenges

Interdomain TE, in terms of both operations and research, is less mature than intradomain TE. Multiple factors contribute to the immaturity of interdomain TE. First, the major tool of interdomain TE, BGP, is designed to express routing policies between ASes, and has few mechanisms to facilitate traffic engineering. Traffic is therefore routed according to the policy constraints, instead of performance requirements. Many interdomain TE techniques are “hacks” of BGP. These techniques can be hard to use, bring undesirable side-effect, and more important they are not always effective.

Next, unlike intradomain TE, in interdomain TE, an AS doesn’t have full control over the path the traffic traverses. This is especially relevant for ingress TE when one AS doesn’t have the cooperation of other ASes. For egress TE, the target AS only has full control of traffic flow in the next AS hop.

Furthermore, in interdomain TE, an AS also has no complete knowledge, sometimes little knowledge, of the AS-level topology and the policies between them. These obstacles directly affect the understanding of interdomain TE, the development of systematic practices, and the effectiveness of these techniques.
The implications of the limited knowledge in this problem area can be seen in many ways:

- When performing interdomain TE, one often assumes other interdomain traffic flows are stationary. However, multiple traffic flows under certain TE mechanisms can interact and lead to performance degradation. One example is the interaction between multiple IRC systems deployed in multihoming environments. When an IRC system chooses upstream connections based on over-estimated performance of other paths, its traffic can oscillate among these connections. Over-estimation of path performance will happen when an IRC system uses “self-load” unaware performance metrics, commonly used by existing IRC systems. Even when an IRC system takes “self-load” effect into consideration, oscillation can still occur when multiple IRC systems share bottleneck links, due to the interaction among IRC systems.

- Interdomain TE usually aims to improve the performance of aggregate traffic, not the performance of network applications which produce traffic that forms the aggregate traffic. Little understanding is acquired on the performance impact of interdomain TE on network applications. Frequent path switching induced by interdomain TE operations, especially techniques like IRC, can be disruptive for application traffic, especially, TCP traffic. TCP congestion control regulates data transfer according to observed RTTs and losses. IRC, as a commonly used interdomain TE technique, also makes path switching decision based on path performance metrics such as delay and loss rate. Both TCP congestion control and IRC tries to improve the performance by reacting to network conditions changes, such as congestions. However, the combined impact of the interactions of the two control systems is not well understood. Without considering this issue, IRC operation can attribute negatively to end-to-end performance.
• In interdomain ingress TE, the most commonly used technique is AS-Path prepending. With many unknowns, such as the BGP policies and the actual AS-level topology in the upstream ASes, the AS-Path prepending technique is often performed in an ad-hoc manner: The network operator keeps increasing the prepending until the traffic is diverted to other paths. The AS-Path prepending techniques is not always effective, because the topology or routing policies of upstream ASes. For multihomed networks, using AS-Path prepending to balance incoming traffic over all ingress links becomes an even more complicated problem.

1.3 Contributions

This thesis seeks to acquire a deeper understanding of interdomain TE and to improve the performance of existing techniques. More specifically, it addresses the issues mentioned earlier by proposing new algorithms, providing operational guidelines, and developing systematic operational approaches for existing interdomain TE techniques.

• Avoiding oscillations due to intelligent route control systems

IRC is a new interdomain TE mechanism that is gaining popularity. However, if IRC systems use self-load unaware performance metrics, or periodically perform deterministic path switching (which is often the case for current IRC products), traffic oscillation can occur. In this work, we identify the root causes of such oscillations, then use self-load aware metrics and various forms of randomization within the IRC algorithm to address these issues. We evaluate IRC algorithms under both stationary and dynamic traffic loads. We show through simulation results that the randomized algorithms are more efficient and more stable than the original algorithm in both cases, though the parameter settings must be optimized for different types of traffic loads. The experimental results also indicate that when the randomized algorithms co-exist with the original algorithm,
the IRC systems that use the randomized algorithms out-perform those that use the deterministic algorithm.

• **Investigating the interactions of TCP congestion control and IRC**

Both TCP congestion control and IRC respond to changes of network conditions, such as delay and losses, but the complex interactions of the two control systems is not well understood. In this work, we construct a simple dual-feedback model to investigate the interactions of TCP and IRC. Through analysis of the model and simulation results, we first explain that the IRC-TCP interactions can be synergistic when 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.

• **Interdomain ingress traffic engineering through optimized AS-Path prepending**

Despite wide deployment, the AS-Path prepending technique is still used in an ad-hoc manner. In this work, we show how to perform interdomain ingress TE in a systematic and algorithmic fashion using AS-Path prepending. We start with a model in which a target network has \( n \) large sources and \( m \) ingress links with capacity constraints. The problem is to provide the optimal prepending on each ingress link so that the load on the ingress links agrees with the capacity constraints. We propose the Optimal Prepending Vector algorithm (OPV) which takes the data rate of the \( n \) sources, the capacity of the \( m \) ingress links, the AS path length matrix, and a matrix describing the upstream tie breaking behavior as input. From this data, we provide the optimal prepending for all ingress links when such a solution is available. Next, our investigation on the
estimation of one of the inputs, the AS path length matrix, shows that all four methods under investigation provide non-negligible errors. Then, based on the OPV, we develop the robust prepending vector (RPV) algorithm, which is robust to the estimation error of the AS path length matrix and the unknown BGP tie breaking behavior in upstream ASes.

1.4 Thesis Organization

The rest of this thesis is organized as follows. Chapter 2 discusses existing work related to the topics in this thesis. Chapter 5 explores the limits and potential of AS-Path prepending technique in interdomain ingress TE. Chapter 3 and Chapter 4 focus on IRC as a mechanism for interdomain egress TE. Chapter 3 investigates the possible traffic oscillation caused by multiple IRC systems, and Chapter 4 studies the performance impact of IRC systems on network application through interactions between TCP congestion control and IRC. In Chapter 6, we summarize the research contributions and discuss the future directions suggested by this thesis.
CHAPTER II

RELATED WORK

This chapter provides an overview of the related work in the topics of interdomain TE, including mulithoming and IRC systems, the interaction of routing systems, AS-Path prepending technique in interdomain ingress TE, and also other related areas in interdomain TE.

Internet traffic engineering refers to the design and implementation of controls that affect the flow of traffic in a network, or internetwork, to meet performance objective[9]. Typical objectives include load balancing across links, achieving better performance, reducing cost for transit service, and combinations of them. In relatively stable conditions, in terms of both routing changes and load variations, TE can be instrumental in improving network efficiency and robustness.

From the perspective of the target traffic flow, TE can be divided into two categories: intradomain and interdomain. In intradomain TE, the operator of an Autonomous System (AS) controls the flow of traffic within that network. Earlier work on intradomain TE is well covered by [24, 23, 25, 39, 18, 22]. We refer readers to these papers and to references therein. Intradomain TE usually assumes that the ingress and egress links of interdomain traffic flows are given as inputs, in the form of a traffic matrix, and they are fairly stationary without big variance.

Interdomain TE, as introduced in the earlier chapter, depending on the direction of traffic relative to the target AS, is divided into ingress TE and egress TE In the following two sections, we are going to summarize related work on interdomain ingress and egress TE respectively.
2.1 Interdomain Egress Traffic Engineering

Interdomain egress TE addresses the egress route selection issue. Most research work on interdomain egress TE focuses on performance and stability of egress route selection algorithms. In this section, we will introduce relevant work on interdomain egress TE, then cover research work related to: intelligent route control, interaction between routing systems, and TE with cooperations.

Feamster et al. focused on egress interdomain TE [21], demonstrating a method to move traffic in a predictable fashion by tuning certain BGP policies. They also present a method to limit the influence of neighboring domains on the local path selection process through BGP policies. That work includes measurements (from the AT&T network) for the frequency and extent of AS-Path prepending: 32% of the routes have some form of prepending, with about 90% of the corresponding paths extended by 1-5 hops, with the maximum prepending 16 hops.

Uhlig et al. also focused on egress interdomain TE in [72]. They modeled the egress TE problem as follows: given m destination networks, a cost function for each of the p downstream providers, and the constraint that only n BGP filters can be configured, the objective is to find the best n filters for the m destinations and the p providers, starting from the default BGP configuration.

Uhlig and Bonaventure propose a systematic approach for interdomain egress TE [71]. In this approach, network operator can define the goal of TE as the objective functions. Optimization boxes deployed in the target AS take BGP routing information and the traffic statistics as input, and select the iBGP configuration changes to optimize the traffic flows. To follow up this work, an interdomain egress TE tool, Tweak-it is proposed [74]. Tweak-it takes the intradomain configuration, BGP routing policies and updates, and ingress traffic as the input, through a heuristic algorithm, produces solutions that requires the minimum BGP “tweakings”.
Teixeira et al, show through collected data and experiments that the default routing strategy, hot-potato routing (i.e. delivery traffic to the nearest egress point to save network resources within the target network), causes routing instability [69, 67]. To solve this problem, a new strategy for egress point selection, TIE (Tunable Interdomain Egress selection) is proposed [68]. In TIE routers select the egress points based on both the intradomain topology and goals of TE, thus TIE combines interdomain and intradomain TE together.

### 2.1.1 Intelligent Route Control

Multihoming capabilities have expanded tremendously with the development of “Intelligent Route Control” (IRC) products. IRC systems allow a stub network to automatically switch certain parts of their egress traffic from one provider to another, driven by cost and/or performance considerations. A number of vendors provide such systems (for a representative but incomplete list see [33, 8, 19, 46, 56, 57, 59, 64, 20]). Even though most commercial multihomed-IRC systems do not expose deep technical information about their internal operation, one of them (the ISMD device of Rether Networks) is described with significant detail in a research publication [29]. Another good description and evaluation of an operational multihoming-IRC system is given in [5]. These two publications, as well as several white papers and high-level descriptions from various vendors, allow us to understand the key points in the operation of existing IRC systems.

In the research domain, IRC systems have become the subject of thorough investigation only recently [3, 4, 27, 76, 66] (with the exception of the very early work reported in [48], which focuses on topological design and configuration of multihoming).

An experimental study, based on measurements from the Akamai content distribution network, shows that multihoming can lead to significant benefits in terms of
both reliability and performance for both ingress and egress traffic [3]. The results also show that up to four providers is typically enough to gain the full benefit of multihoming.

Experimental comparisons between IRC and routing overlays have been described in [4]. The measurement-based comparison of [4] suggests that IRC systems may be capable of offering almost the same performance as routing overlays, but in a much simpler and more cost-effective way.

Other experimental work that demonstrated similar benefits is reported in [66]. In [66], path switching schemes, i.e. multi-homing and overlay network, are investigated over a three-node testbed. The authors evaluate the performance improvement on end-to-end latency and loss rate.

In a more theoretical thread, the authors of [27] propose a series of “intelligent” routing schemes under the percentile-based charging method. The flexible nature of the routing schemes allow optimization under various constraints. On global effects of the proposed routing schemes, the simulation results show that the performance suffers negligible declination from the interaction between multiple customers and the intelligent routing customers co-exist well with single homing customers. This work also examined the equilibrium performance of competing IRC systems through simulations, and argued that a multihomed user can improve its own performance without adversely affecting other users. Two interesting pricing problems related to multihoming, namely the optimal set of ISPs that a network should subscribe to and how ISPs can react to that optimal subscription, have been recently investigated in [76].

2.1.2 Interaction between Routing Systems

When selecting egress links for interdomain egress TE, one network makes routing decision independently. When multiple networks conduct interdomain TE at the
same time, interaction between these TE operations may induce loss of performance, routing instability, or traffic oscillations. This problem motivates a series studies on the stability of interdomain egress route selection [78, 77, 80].

In [78], the authors study interdomain egress TE with a general model in which the egress route selection of some destinations is coordinated and the preference of candidate routes is rational and predetermined. Based on this model, the authors show instability and inefficiency can be caused by interdomain TE. The analysis results show that egress selection of interdomain egress TE is stable if no provider-customer loop exists, no valley route exist, and routing decision of customer reachable routes is independent to peer/provider reachable routes. Based on the previous work, the authors consider inbound traffic dependent interdomain egress TE [77]. The results show that when all networks use some forms of rational route selection algorithms, and interdomain egress TE is inbound-traffic-dependent, there exist networks having no stable route selection between them.

The TE model used by work above assume that the preference of candidate routes can be predetermined and will never change. This assumption doesn’t hold in the presence of performance-based routing mechanisms such as IRC systems.

Besides interdomain egress TE, interaction between other types of routing systems, for instance, TE and overlay routing, is also the topic of various studies [51, 40, 35, 36]. These studies take the game theory approach: model the interaction of routing systems as a game, and the routing systems are players of the game.

[51] and [40] investigate the interaction between overlay networks and intradomain TE. The former focuses on selfish routing in Internet-like environment. The stability and performance of selfish routing is evaluated against other four types of routing schemes on top of measured and generated topologies. Results show that selfish routing is able to achieve similar performance to the network optimal routing, with or without background complaint routing traffic (which is disputed by Roughgarden
and Tardos [61]). However, when under the traffic engineering through changing the OSPF link weight, the selfish routing can lead to big performance decline. This result is partly confirmed by further study of the interaction of overlay network and OSPF traffic engineering [40]. In this study, the interaction of overlay routing and traffic engineering is modeled as a two-player non-cooperative non-zero sum game. In a simple network, such a game has unique and global stability of Nash equilibrium point, however in a general network, a selfish overlay can cause huge cost increases and traffic oscillation.

[36] focuses on the interaction between overlay networks. The authors show that independent routing decision of multiple overlay networks can lead to routing and traffic oscillations. The root cause of such oscillations are partially overlapping routes and the periodic path probing process. Through an analytical model formulated in the paper, we can calculate the synchronization probability and the upper bound of the duration of traffic oscillation of two overlay networks.

Based on selfish routing, new strategy “overlay optimal routing” is introduced in [35]. In this new overlay routing scheme, the source can split traffic load on multiple path, thus overlay networks using this routing scheme are able to reach a Nash equilibrium and achieve better performance than selfish routing. The authors also point out that the Nash equilibrium reached by multiple players, also include non-overlay routing systems, may not be Pareto optimal, which means that no moves can be made to improve the performance of some players’ while maintaining the performance of others. To solve this problem, two pricing schemes are proposed so that the players can be guided to a the optimal equilibrium.

One common drawback of game-theory-based research on interaction of routing systems is that the model usually assume “fair” games between the players, which means every player gets a move in each round. This condition may not hold because different types of routing systems operate on different timescales. For example, OSPF
weight changes usually occur once or twice a day, and overlay network routing can change as frequently as once every few minutes.

2.1.3 Traffic Engineering with Cooperations

Interdomain TE has mostly been executed independently. Many propose to change this absolute independence and conduct interdomain TE in a cooperative fashion [44, 52, 79, 63]. The benefit of cooperative TE is three folds. First, limited information is exchanged between networks, which provide some reference for the operation of TE. Second, cooperative TE reduced the unpredictability of traffic flow. Last, when TE efforts are cooperative, the networks can eliminate the harmful interaction shown by work in last subsection.

Winick et al. are one of the earliest to propose coordinated TE between neighboring networks [79]. The authors argue that given appropriate network management tools and the capability of predicting traffic flow upon TE operations, neighboring networks are able to do efficient interdomain TE with only limited exchange of information.

Mahajan et al. first propose some building blocks toward cooperative interdomain TE [43]. Then propose a more complete framework toward a cooperative interdomain TE architecture Next [44]. The framework includes two parts: evaluation of routing choices and negotiation protocol between neighbor networks. Under the scenario of a pair of neighboring networks with multiple peering points, the authors discuss in details the negotiation procedure and the benefit of cooperative TE. The simulation results also show that the deceptive network doesn’t harm the truthful network.

Shrimali, Akella and Mutapcic propose an interdomain TE protocol in which networks use an iterative procedure to cooperatively optimize a social cost function [63]. This problem can be divided and optimized by ISPs individually in a distributed manner. Compared to earlier work, this new protocol doesn’t require the networks
exchange sensitive information. It also guarantee a Pareto optimal solution.

Quoitin et al. propose cooperation TE between the source and the destination networks through virtual peering [52]. A virtual peering is basically an IP tunnel connecting the border routers of two networks. A virtual peering is requested by the destination network, then the source and destination will negotiate the choice of the egress route at the source. One key contribution of virtual peering method is that it requires only the source and destination networks, not the intermediate networks.

Unavoidably, all proposed cooperative TE schemes either require exchange of routing information or require deploy new protocol or routing architecture, or both. Routing information is usually very sensitive, and networks often tightly guard such information, especially between competing ISPs. Deployment of new routing protocols or architecture, or just changes on existing protocols, is even more challenging. These difficulties may prevent the deployment of cooperative TE schemes.

2.1.4 Other Approaches to interdomain TE

Instead of adding extensions to BGP, Agarwal et al. proposed an Overlay Policy Control Architecture (OPCA) [1]. A major motivation behind OPCA is to support INTIE. OPCA is basically an overlay network running on top of BGP to facilitate policy exchanges between ASes. It consists of a set of policy agents deployed in the participating ASes that communicate through an overlay protocol to process external policy announcements and negotiate with remote ASes the selection of interdomain paths for ingress traffic.

2.2 Interaction between TCP Congestion Control and Traffic Engineering

TCP congestion control monitors path performance and regulates data delivery accordingly. Similarly, performance-based routing also monitors path performance, but
route traffic accordingly. The interaction between TCP and performance-based routing comes in when both systems react to the change of path performance. The lack of understanding of such interaction starts to draw attention recently [7, 30], our work presented in Chapter 4 also falls into this area.

Anderson and Anderson try to design a class of adaptive routing schemes that are stable when co-exist with congestion control [7]. The study identifies that when a adaptive routing scheme minimizes the maximum utilization of network links, which leaves the maximum headroom for the operation of congestion control, it will quickly converge to a optimal and stable solution. However, under unfair congestion control or in a overloaded network, this class of adaptive routing schemes will not achieve optimal performance. This paper defines a generic model for performance-based adaptive algorithms, which also include IRC, and it considers a wider ranges of congestion control mechanisms than TCP specific. Thus, it is not able to capture the performance impact for TCP connections, based on which most network applications are built.

He et al. look into the interaction between TCP congestion control and intradomain TE [30]. TCP congestion control, in this work, is modeled as a optimization function to maximize aggregate user utility by varying the source data rate. TE is modeled as a joint system with congestion control with the congestion control step and routing step alternating in a feedback loop. This joint system is proved be stable and optimal under elastic traffic by tuning the cost function of TE. The authors also propose new TE algorithms that adapts faster to traffic changes and more robust to large traffic variations.

2.3 Interdomain Ingress Traffic Engineering

Interdomain ingress TE is more challenging than other types of TE [55], for the goal is to influence the routing of upstream ASes of which the target AS has no knowledge
and control.

The feasibility of interdomain TE by a stub AS was examined in [70] (without considering a particular TE mechanism). This work suggest that, despite the important variability of interdomain flows, it would be useful for a stub AS to engineer its ingress traffic, aggregated at an appropriate level. In follow-up work, based on more recent measurements, the authors made more pessimistic conclusions regarding the feasibility of interdomain traffic engineering [73]. In the case of their UCL and PSC traces (both stub networks), they showed that there was significant temporal variability in the traffic carried by each major AS path. Furthermore, due to limited aggregation at the AS level, they argued that to effectively control ingress traffic one would need to affect the route selection in a large number of ASes.

The potential performance benefit of ingress TE of multihomed network is explored in [5]. Through a trace-driven experiments on a testbed, the authors show that ingress TE is able to achieve 20% - 25% performance improvement. The work also explored the delay and frequency of network performance measurement.

As we introduced in last chapter, four techniques are mostly used for BGP-based interdomain ingress TE: AS-Path prepending, selective announcement, using BGP MED attribute, and using special BGP community attribute.

Selective announcement was proposed as a scalable support for multihomed network ([10]. Deployment of selective announcement is straight forward, usually done by splitting of prefixes then announcing these prefixes on subset of upstream providers. Selective announcement is able to control traffic in finer granularity, which is hard to achieve in AS-Path prepending. The limitation of selective announcement is that the target AS needs a large enough prefix before the prefix splitting, else the splitted prefixes could be filtered by upstream for their small sizes[16].

The usage of BGP MED attribute is only limited to multiple peering points between the same pair of ASes. Thus it is rarely use by multihomed networks which
usually have only one connection to each of the upstream providers.

Quoitin et al. [53] evaluate the performance of two BGP-based techniques in interdomain ingress TE: AS-Path prepending and BGP community value. In this measurement work, one dual-homed network changed the prepending of routes announced to the upstream providers, then measures through TCP SYN packets the percentage of traffic/network accessing a server inside this dual-homed network via the two providers. The authors observe that AS-Path prepending tends to control the flow of traffic in a very coarse granularity, the marginal benefit of AS-Path prepending decreases quickly, and the effect of prepending is uncertain. Experiments on do not announce community shows community-based ingress TE is able to control traffic flow in finer granularity. However, community-based interdomain ingress TE also suffers from drawbacks like unpredictable and hard to configure. Through analysis of the results, the authors also point out that tie-breaking rules can be essential to ingress TE.

2.3.1 BGP Community Attributes

BGP appears to be flawed on supporting interdomain TE. Aiming for more effective interdomain TE, proposals ([12]) have been made on revising BGP to provide better support for interdomain TE. Introducing new BGP communities is one of the direction on modification of BGP. Compared with other interdomain ingress TE techniques, BGP community ([14, 62] is a more expressive tool, and has long been used to label routes [11]. The disadvantage of BGP community values is it requires cooperation of remote networks, that is, all networks follows the same interpretation of the community values.

The use of the BGP community attribute for interdomain ingress TE has been studied in depth in [54]. The authors presented various drawbacks of using that attribute: the requirement for a manually encoded filter for each supported community,
the fact that each AS must advertise the semantics of its own community values to peers, and the transitivity of that attribute. They proposed the use of a new standardized form of extended community, referred to as “redistribution community”, which supports the following actions (among others): the attached route should not be announced, or it should be announced only to specific BGP speakers, and that the attached route should be prepended a number of times when announced to specific peers.

Internet Draft [2] proposes a community, referred to as BGP PCC, that carries another community to be attached to a route, when the latter is sent to a distant AS. The PCC value, viewed as a tuple (AS-X, AS-Y, c), represents a request directed from the originating AS to AS-X to send community c to AS-Y when the associated route is sent from AS-X to AS-Y. The PCC community can be used for INITE, by allowing an originating AS to affect the route selection process in a remote AS.

2.3.2 AS-Path Prepending

AS-Path prepending has long been used as an interdomain ingress TE technique, but it is often practiced in a trial-and-error manner and its efficiency is not guaranteed. Not until recently, the lack of understanding of AS-Path prepending starts to draw much interest from the research community.

Besides the performance evaluation of AS-Path prepending in [53], later measurement work includes [41, 42]. In [41], the experiment set up is similar to [53], except that the effect of prepending is measured through looking glass servers and route servers. The prepending starts from zero, increases to five, one at a time in each update, then decreases back to zero in the same way. The prepending updates are scheduled once every two or three hours. The following work [42] expand the experiments from one local ISP to three RIPE sites. The new experiments set up also includes a control prefix to detect route updates not caused by prepending changes.
The more extended experiments shows that the effectiveness of AS-Path prepending heavily depends on the topological location of the target AS and its upstream providers. The results also reveals some hidden impact of AS-Path prepending on interdomain routing.

[15] proposes a framework for AS-Path prepending, involving passive/active measurement and traffic change prediction. The passive measurement extracts inbound traffic profile from NetFlow monitor data, the active measurement probes the effectiveness of AS-Path prepending on top senders, then the AS predicts the traffic change assuming prepending happens.

Wang et al. [75] conduct a more recent survey of the usage of AS-Path prepending, study the load balancing effectiveness of prepending, and propose two algorithms for this technique.
CHAPTER III

AVOIDING OSCILLATIONS DUE TO INTELLIGENT ROUTE CONTROL SYSTEMS

3.1 Introduction

In the previous chapter, we investigated AS-Path prepending as a interdomain ingress TE technique. From this chapter, we study interdomain egress TE through Intelligent Route Control (IRC).

Intelligent Route Control (IRC) systems are increasingly deployed in multihomed stub networks. IRC systems aim to optimize the cost and performance of outgoing traffic, based on measurement-driven dynamic path switching techniques. We first show that IRC systems can introduce sustained traffic oscillations under certain conditions, causing significant performance degradation instead of improvement. This happens, first, when IRC systems do not take into account the self-load effect, i.e., when they ignore that the performance of a path can change after additional traffic load switches to that path. Second, oscillations can take place when IRC systems follow deterministic switching rules. We then propose measurement methodologies and path switching algorithms that can effectively deal with the previous issues. The proposed IRC techniques use available bandwidth estimation to deal with the self-load effect, and they introduce a random component in the path switching decision or time scale. We evaluate the proposed techniques under a range of traffic conditions. When the underlying cross traffic is stationary, IRC systems should switch paths conservatively, only upon significant traffic bursts. On the other hand, if the cross traffic shows major variations in a time scale $T$, IRC systems improve performance only if
they can switch paths much faster than \( T \); otherwise, they end up causing oscillations or worse performance. We also show that the gradual deployment of randomized IRC systems, in the presence of traffic from deterministic IRC systems, would play a stabilizing role and it would be beneficial for networks that adopt the former.

In this chapter, we consider a multihomed destination ("sink") network \( D \) with \( m \) ingress links. \( D \) receives traffic from several source networks, including the multihomed networks \( S_1, S_2, \ldots, S_n \). The latter deploy IRC systems and they can reach \( D \) through some or all of its ingress links. We assume that a significant fraction of the total traffic destined to \( D \) originates from these IRC-capable networks. Even though this is not the case today, it is certainly a plausible scenario for the near future. Each IRC system uses measurements to monitor the performance of the paths to \( D \), and switches its traffic towards \( D \) to the path that offers the required performance level. The performance metrics that we focus on are loss rate and path switching frequency. An important point about our model is that the network measurements are not instantaneous; instead, as it always happens in practice, they take place over a time window and they consist of several samples. The IRC model is described in more detail in Section 3.2.

We first show that IRC systems can cause sustained traffic oscillations (Section 3.3). Interestingly, as a result of these oscillations, IRC systems can lead to a significant performance degradation, instead of improvement. We then identify two key factors that generate such oscillations. First, if IRC systems do not take into account the self-load effect, meaning that the performance of a path can drop significantly after we switch additional traffic to it, then IRC traffic can keep switching back and forth between the two paths. Second, oscillations can take place when different IRC systems get synchronized because there is significant overlap in their measurement time windows. When such synchronization occurs, independent IRC systems start behaving as a "herd", observing the same performance difference between two
or more paths and making the same sequence of path changes.

In the second part of this chapter, we propose measurement methodologies and path switching algorithms that can avoid the previous two oscillation factors (Section 3.4). First, we propose the use of available bandwidth measurements to avoid the self-load effect. Second, we introduce a random component in the path switching decision, or in the switching time scale, to avoid synchronization. We evaluate the proposed IRC techniques with simulations under diverse traffic conditions. As expected, the performance of IRC systems is intimately related to the variability of the background traffic. When the background traffic is stationary, IRC systems should switch paths conservatively, only upon major traffic fluctuations (Section 3.5). With nonstationary background traffic and congestion periods that last for a time scale $T_w$, IRC systems improve performance only if they can detect congestion and switch paths much faster than $T_w$; otherwise, they cause oscillations and hurt performance (Section 3.6). Finally, we investigate the gradual deployment of the proposed randomized IRC systems in the presence of traffic from deterministic IRC systems (Section 3.7). Interestingly, randomized IRC systems play a stabilizing role, reducing the overall loss rate. Furthermore, they are beneficial for the networks that adopt them, as the latter observe better performance than networks using deterministic IRC.

3.2 IRC model

In this section, we describe the model of an IRC system, as well as the network and traffic environment in which we assume that IRC systems operate.

3.2.1 Network model

We consider a single destination network $D$ that is multihomed, as well as a set of source networks that send traffic to $D$. We refer to the source networks that either have only one egress link, or that always use a specific egress link to reach $D$, as Default Route Control (DRC) sources. On the other hand, multihomed networks
that can dynamically switch between different egress paths to reach \( D \) are referred to as *Intelligent Route Control* (IRC) sources.

We assume that the underlying Internet routes to \( D \) are stable relative to the time scales of IRC path switching. Hence, the path from a DRC source to the destination network \( D \) will traverse a particular ingress link of \( D \), as determined by BGP. In contrast, the path from an IRC source to the destination network \( D \) will be determined by both BGP and by the choice of egress link at the source. When two sources use the same ingress link at \( D \), their paths intersect, minimally at the ingress link, and possibly further upstream as well. We model this path interaction as occurring only at the ingress link, rather than developing a more complex model for upstream path interaction. The justification for this assumption is twofold. First, to an approximation, we can consolidate the interaction over multiple upstream links as interaction over a single link. Second, for many enterprise stub networks today it is their access link to the Internet that is often the end-to-end path bottleneck.

![Network model](image)

**Figure 2:** Network model.

As illustrated in Figure 2, we assume that \( n \) IRC sources \( S_1, S_2, \ldots, S_n \) send traffic to destination \( D \) through \( m \) ingress links (DRC sources are not shown). The figure illustrates that, in general, some IRC sources may not be able to access all ingress
links of $D$ (source $S_n$ has this characteristic, for instance). Also, some IRC sources may always traverse the same destination ingress link, regardless of the selected source egress link, because of how the underlying Internet routes merge before reaching $D$ (source $S_1$ has this characteristic, for instance). From this point on, however, we assume that all IRC sources can traverse any of the $m$ ingress links at $D$.

### 3.2.2 Traffic model

We use the term *flow* to refer to the aggregation of all traffic from a single source network to $D$. When a flow's path is determined by IRC, the flow is an *IRC flow*; otherwise it is a *DRC flow*. We assume that network $D$ receives a significant fraction of traffic from IRC flows. This is a plausible assumption for the near future. We also assume that the *statistical characteristics* (including the average rate) of a flow remain constant over the time scales of interest. We do consider traffic burstiness, however, assuming that each flow follows the Fractional Gaussian Noise (FGN) model. Further, we assume that a flow's average rate does not change in response to changes in the path characteristics (e.g., congestion). This is a reasonable assumption as long as the arrival rate and size of new connections at the corresponding source network do not depend on network conditions (i.e., exogenous connection arrivals).

As previously mentioned, we model each traffic flow as an FGN fluid process [49]. The FGN process is self-similar and long-range dependent, and therefore it is considered an appropriate model for aggregate Internet traffic, especially in larger time scales where packet-level effects can be ignored. An FGN fluid process is determined by three parameters: the Hurst parameter that controls the degree of self-similarity, the average rate, and the variance over a given time scale. In the following, the Hurst parameter is set to 0.7, which is consistent with earlier measurement results [49]. For normalization purposes, we use the same coefficient of variation (CoV) for all flows. To pick a realistic CoV value, we fitted the FGN model to packet traces from an OC-3
university access link, at the time scale of 100 msec. The CoV in those traces varies between 0.09 to 0.16, and so we set the CoV of each flow so that the aggregate traffic at the ingress links of $D$ has a CoV that is around 0.1.

Regarding the average rate of IRC flows, we use two distributions: a constant value for all flows and the Zipf distribution. We expect that these two extreme distributions will give us the right insight on how IRC performance depends on the flow homogeneity. DRC flows have the same average rate regardless of the IRC flow rate distribution.

### 3.2.3 IRC processes

We model an IRC system as performing periodically a five-stage process. The five stages are: idle, measurement, performance estimation, routing decision, and path switching (see Figure 3). $T_r$ denotes the routing period, i.e., the time to complete a cycle through all five stages. An IRC system starts the cycle with the idle stage, which is optional. Then, in the measurement stage, the system collects performance samples for all candidate paths, using active probing or passive monitoring. Next, for each candidate path, the IRC system uses the previous measurements to estimate the performance of each path. In the routing decision stage, the IRC system determines whether it should switch to a different path, based on the specified performance objectives. Finally, if needed, the IRC system reroutes its outgoing traffic towards $D$ to the chosen egress link. That link will be used at least during the next routing period.

We assume that the performance estimation, routing decision, and path switching stages can be executed instantaneously, as they only involve simple calculations and local routing table changes. Thus, from a timing point of view, the routing period $T_r$ consists of the idle period and the measurement period $T_m$. For faster response to network congestion, the idle period can be minimized or even avoided. Hence,
we assume that the routing period is as short as possible, only bounded by the length of the measurement period, i.e., $T_r = T_m$. In the following simulations, we set the routing period to $T_r = 1$ second. Even though this is a short routing period compared to existing IRC systems, we believe that the requirements of interactive and transaction-based applications will gradually push network operators to reduce $T_r$ as much as possible.

![Diagram of IRC routing period]

**Figure 3:** The timeline of an IRC routing period.

### 3.2.4 Measurement process

A central component of an IRC system is the measurement process. In earlier work on multihoming and IRC, network measurement has been modeled as simple meter reading, i.e., an instantaneous and accurate action. That model however does not capture important characteristics of real network measurement, which can affect the stability and performance of IRC systems. Specifically, in practice network measurements take time, they are subject to estimation errors, and they often rely on sampling rather than continuous-time monitoring.

In our model, as illustrated in Figure 3, we assume that the performance of each path is estimated through periodic active probing. Each probing event collects a sample of the monitored path’s performance. In a realistic environment with bursty network traffic, multiple samples are necessary for reasonably accurate estimation. Also, we need to consider that each probing event takes some time $T_s$; for instance,
the time between sending a probing packet and receiving the corresponding ICMP response or acknowledgment. The path performance is estimated at the end of the measurement period, averaging the collected samples. In the following simulations the measurement period \( T_m \) consists of 10 samples, with a sampling period of 100msec.

### 3.2.5 Performance estimation

We consider three end-to-end metrics for evaluating the performance of a network path: queueing delay, loss rate, and available bandwidth. The queueing delay of a path is the difference between the measured Round-Trip Time (RTT) and the minimum observed RTT; the latter is typically due to propagation and transmission delays. The loss rate of a path is the fraction of lost probing packets. Finally, the available bandwidth of a path is defined as the minimum residual capacity among all links in the path. Most IRC systems today use the first two metrics, or straightforward variations of these metrics. Measurement techniques for available bandwidth have been developed only recently and it seems that they are not used by commercial IRC systems yet [34].

In the following results, we do not simulate the measurement process with individual probing packets. Instead, the previous three metrics are estimated as follows.

First, the simulated queueing delay \( d \) at an ingress link of capacity \( c \) is given by the following non-decreasing function of the instantaneous offered load \( r \):

\[
d = \begin{cases} 
\frac{1}{c - r}, & (r \leq c - \epsilon) \\
 d_{\text{max}}, & (r > c - \epsilon)
\end{cases}
\] \hspace{1cm} (1)

where \( \epsilon \) is a small positive constant. Note that the upper bound \( d_{\text{max}} \) models a finite buffer size (\( d_{\text{max}} = 1/\epsilon \) for continuity).

Second, the simulated loss rate \( l \) at an ingress link of capacity \( c \) is determined by the fluid model:

\[
l = \begin{cases} 
\frac{r - c}{c}, & (r \geq c) \\
0, & (r < c)
\end{cases}
\] \hspace{1cm} (2)
In other words, we assume that the link does not drop traffic unless it is saturated.

Third, the simulated available bandwidth \( A \) of an ingress link is given by

\[
A = \begin{cases} 
  c - r, & (r \leq c) \\
  0, & (r > c)
\end{cases}
\] (3)

Note that when comparing the currently used path \( p \) with another path \( p' \) in terms of available bandwidth, an IRC system has to consider the offered load \( r_f \) of its flow. Specifically, \( p' \) is better than \( p \) only if \( A_{p'} > A_p + r_f \).

### 3.2.6 Routing decision and path switching

How does an IRC system compare paths and determine whether to perform path switching? We consider two cases: first, that the IRC system can only measure delay and loss rate (the most common case today), and second, that it can also measure available bandwidth.

When measuring only queueing delay and loss rate, we consider the latter as more important in terms of network performance. So, the path with the lowest loss rate is considered the best. This is consistent with typical Service Level Agreements today, which consider losses as more detrimental than queueing delay. If there are several paths with the same loss rate (for instance, \( l=0 \)), the best path is the path with the minimum queueing delay. When the IRC system can also measure available bandwidth, the best path is that with the maximum available bandwidth. Note that if a path has some available bandwidth (\( A > 0 \)), then its loss rate \( l \) is zero (Equation 2). If all paths are saturated (i.e., \( A=0 \)), then the best path is chosen based on loss rate and/or queueing delay.

Taking into account that measurements are error-prone, we consider the performance of two paths as equal if the corresponding performance metrics are close, say within 10%. Specifically, if \( A \) and \( B \) are measurements over two paths \( p \) and \( p' \) respectively, we consider that \( p \) is better than \( p' \) if \( 0.9A > 1.1B \).
Knowing that there exists a better path than the currently used path does not mean that the IRC system should switch to the former. Specifically, we consider two path switching policies: choose-best and choose-good. With the former, the IRC system always switches to the best path. With the latter, the IRC system switches to the better path only if the current path is congested, i.e., if the loss rate is larger than zero.

With choose-best, an IRC system may switch paths even when there is no congestion in the current path, and so the path switching frequency increases. On the other hand, choose-best provides IRC traffic with a larger safety margin to random traffic variations and measurement errors. For this reason, we focus on the choose-best policy. The main conclusions of our study, however, are also valid for the choose-good policy when the ingress links of D are not overprovisioned.

3.3 IRC-induced traffic oscillations

In this Section, we discuss two conditions under which IRC systems can cause traffic oscillations. The first problem appears when using network measurements and performance metrics that do not take into account the load of the switched traffic ("self-load effect"). The second problem is the synchronization of different IRC systems when their measurement time windows overlap significantly.

3.3.1 Self-load effect

As explained in Section 3.2, the IRC routing decisions are based on estimates derived from path measurements. The problem, however, is that the performance of a path p can be significantly affected after the IRC system switches some traffic to or from that path. For example, consider an IRC system and two candidate paths p_1 and p_2. Initially, the IRC system routes its traffic over path p_1, and the measured loss rates are 1% for p_1 and zero for p_2. This does not necessarily mean that p_2 is better. After the IRC system switches its traffic to p_2, the loss rate at the latter can become larger
than 1%, due to the additional load that the IRC traffic imposes, and the loss rate at $p_1$ can become zero. Hence, the IRC system can start oscillating between the two paths. The same scenario can take place when the path switching decisions are based on queueing delay measurements. Note that it is not possible to predict the loss rate or queueing delay at a path after a load shift without an accurate characterization of the queueing behavior in that path, and such a characterization is quite difficult in practice.

With available bandwidth measurements, on the other hand, the previous problem can be avoided. The reason is that the available bandwidth shows how much additional traffic a path can carry before it is congested. For example, if $p_1$ has available bandwidth $A_1=2$Mbps and it carries an IRC flow of 5Mbps, while path $p_2$ has available bandwidth $A_2=4$Mbps, then the IRC system should prefer $p_1$ because if the flow is switched to $p_2$ it will certainly experience congestion.

![Figure 4: Path switching events (top and middle graphs) and loss rate (bottom graph) when using delay and loss rate versus available bandwidth measurements.](image)

Figure 4 shows the results of two simulations with the same configuration, except that one uses the loss-delay metric while the other uses the available bandwidth metric. Both experiments have four flows, one IRC flow and three DRC flows, and
two identical ingress links at the destination network $D$. The four flows have the same average rate, and the aggregate capacity of the ingress links is equal to twice the total load of the four flows. The experiments start with two flows on each link. In Figure 4, the top two plots show the path the IRC flow uses as a function of time. When the IRC flow switches from one path to the other, it is shown as a downward or upward step. The number of downward and upward steps is the number of switching events of the IRC flow during that time period. The bottom plot of Figure 4 shows the running average of the loss rate of the IRC flow, over a moving window of 30 seconds.

The path switching plots clearly show that when the IRC flow uses the delay-loss metric it experiences many more path switching events than using the available bandwidth metric. Also, the loss rate that the IRC flow experiences is significantly reduced when using the available bandwidth metric, because the flow stays at the best possible path.

3.3.2 Synchronization of IRC systems

Another issue with IRC systems is that different IRC flows can get synchronized, oscillating between two or more paths. The fundamental cause for the synchronization is the possible overlap of the measurement time windows of different IRC systems. To understand this effect, consider a simple example with two identical ingress links and two identical IRC flows, $f_1$ and $f_2$. To avoid the self-load problem, assume that the flows use the available bandwidth metric. Also, suppose that both IRC flows have equal routing and measurement periods with $T_r = T_m$ and that they take 10 available bandwidth samples in each measurement period. Let us assume that the timing between the two flows is such that the routing decision of $f_1$ occurs one sample period earlier than that of $f_2$. $f_1$ and $f_2$ start at the same path $p_1$, and after the first routing period $f_1$ detects greater available bandwidth on $p_2$ and switches to that path.
At that point, \( f_2 \) has already collected nine out of its 10 samples, and even though the available bandwidth in \( p_2 \) is now equal to that in \( p_1 \), it may also estimate that the average available bandwidth, across all 10 samples, is larger in \( p_2 \). So, \( f_2 \) will switch to \( p_2 \), where it will overlap with \( f_1 \) for 80% of the next measurement period. In the next routing period, \( f_1 \) and \( f_2 \) will move back to \( p_1 \) in the same fashion, hence producing a persistent oscillation between the two paths.

![Oscillations due to synchronization of different IRC systems.](image)

**Figure 5:** Oscillations due to synchronization of different IRC systems.

In practice, even a lower degree of overlap between the measurement periods of IRC systems can still cause synchronization. In Figure 5, we show the type of oscillation described above but under a less rigid configuration. In this simulation, the destination has two ingress links. Initially, there are 10 IRC flows and 10 DRC flows on each link. All flows have the same average rate and equal routing/measurement periods \( T_r = T_m = 1 \text{sec} \). The start times of the IRC flows (and thus their route decision events) are uniformly distributed over the length of a measurement period (one second). The graphs show the number of IRC flows on each path (top) and the traffic load (bottom) over time. Note the persistent path and traffic oscillations, as a result of IRC synchronization. The oscillation period is two seconds, which agrees
with the fact that IRC flows perform a routing decision in every second.

One may think that such oscillations can be avoided if IRC systems use the last measured sample, instead of an average across all samples, to estimate the performance of a path. This is not practical however, given that the performance of a path can vary significantly with time and the measurements are prone to errors. Consequently, the measurement period has to include several samples, and this implies that the measurements windows of different IRC systems can overlap in time causing synchronization. In the next section, we propose several path switching algorithms that can avoid synchronization through limited randomization.

### 3.4 Randomized IRC algorithms

The previous section identified two problems with IRC systems: the self-load effect, which can cause oscillations even with a single IRC flow, and the synchronization of different IRC flows due to a significant overlap in their measurement periods. The self-load effect can be avoided if the IRC system uses available bandwidth measurements. In this section, we focus on path switching algorithms that can avoid the synchronization problem.

In general, synchronization among a set of autonomous agents can be avoided with the introduction of a certain degree of randomness in the actions of these agents. In the context of IRC systems, this randomization can take several forms. First, we can add randomization in the path selection itself. This is not a good option, however, if there are only two or three paths to choose from. Second, we can add randomization in the path selection timing, i.e., in the length of the routing period \( T_r \). Third, we can add randomization in the path switching decision, i.e., on whether the path switching should be performed or not.
3.4.1 Deterministic Path Switching (DPS)

DPS is the basic path switching algorithm that we described in Section 3.2. It does not add any randomization in the path switching decision and it uses a fixed routing period \( T_r \). As shown in Section 3.3, DPS can lead to persistent oscillations when two or more IRC systems get synchronized. The following four algorithms are variations of DPS that include some form of randomization.

3.4.2 Fixed Switching Probability (FSP)

FSP switches to the best path with a probability \( P \) that we refer to as the switching probability. \( P \) controls the responsiveness of IRC flows to performance changes. When \( P = 0 \), the IRC flow behaves just like a DRC flow (static routing) and it does not respond to measurements. When \( P = 1 \), the IRC flow behaves as a DPS flow and it is susceptible to synchronization.

3.4.3 Adaptive Switching Probability (ASP)

ASP is a variation of FSP in which \( P \) adapts to the network conditions. The intuition is that \( P \) should be large when there is a major performance difference between the current path and the best path, while \( P \) should be much lower when the current path is almost as good as the best path. For simplicity, ASP only uses two \( P \) values: \( P_{hi} \) and \( P_{lo} \). The algorithm uses \( P_{lo} \) when the difference between the current path and the best measured path is below a certain threshold; otherwise it uses \( P_{hi} \). We set that threshold to \( 3\sigma \), where \( \sigma \) is the standard deviation of the corresponding performance metric in the current path.

3.4.4 Random Routing Period (RRP)

RRP adds randomization in the routing period \( T_r \) (see Figure 6). Specifically, \( T_r \) is uniformly distributed in a range \([T_m, T_M]\), in which \( T_M \) is the maximum possible routing period, while \( T_m \) is the measurement period (1 second). When \( T_M = T_m \),
RRP is the same with DPS. Note that when $T_r > T_m$, the measurement period covers the last $T_m$ time units of the routing period. A larger $T_M$ makes it less likely that measurement periods of different IRC systems will overlap, but it also decreases the responsiveness of the IRC system.

\[ \text{RRP algorithm} \]

\[ \text{HRP algorithm} \]

\textbf{Figure 6:} RRP and HRP algorithms.

### 3.4.5 Hysteresis Routing Period (HRP)

HRP is a variation of RRP in which a hysteresis period is added after each path switching event. Figure 6 illustrates the difference between the two algorithms. Contrary to RRP, which has a random routing period, HRP uses a fixed $T_r$, which is equal to the measurement period $T_m$. However, after each path switching event HRP inserts a random hysteresis period. The purpose of the hysteresis period is to break any synchronization that may result after the last path switching. At the same time, HRP allows quick response to congestion, because it keeps $T_r$ to its minimum. The length of the hysteresis period is uniformly distributed in a range $[0, T_H]$, where $T_r + T_H$ is the largest routing period.

### 3.5 Evaluation of IRC algorithms - stationary load

In the previous two sections, we illustrated that the basic path switching algorithm (DPS) can lead to synchronization and proposed four algorithms (FSP, ASP, RRP,
HRP) that rely on randomization to avoid such synchronization. In this section, we evaluate these algorithms in terms of loss rate and switching frequency under stationary load. By stationary load, we mean that the average rate of the DRC flows, i.e., the background traffic, on each ingress link of the destination $D$ remains time-invariant. We emphasize that stationarity does not mean constancy; on the contrary, both DRC and IRC traffic flows have highly variable rates, driven by the FGN model.

3.5.1 Simulation setup

The topology of the experiments is based on the network model of Section 3.2 in which $n=100$ sources send data to the destination network $D$ through $m$ ingress links. Among the $n$ sources, $n_I$ of them are IRC sources, while the remaining $n_D$ are DRC flows (background traffic). The ratio $n_I/n_D$ controls the fraction of IRC traffic relative to the statically routed background traffic. The traffic of each flow follows the FGN fluid model, as described in Section 3.2, with a rate change every 100 msec. For the DPS, FSP, ASP, and HRP algorithms, the routing period $T_r$ is set to one second, equal to the measurement period $T_m$. The start time of each flow is randomly chosen within the routing period. The first 50 seconds of simulation time are discarded to avoid any transient effects.

We use two metrics to evaluate the performance of IRC flows: loss rate and path switching frequency. Both metrics are defined over a time period. The loss rate of an IRC flow is defined as the fraction of the total traffic volume that is lost in that time period. The (path) switching frequency of an IRC flow is defined as the total number of path switching events during a time period divided by the length of that period. The loss rate evaluates the effectiveness of IRC in improving the performance of its traffic. The switching frequency, on the other hand, evaluates stability. The latter is important both for IRC flows (frequent path changes cause packet reordering and increased jitter) and for the underlying network performance (e.g., effectiveness of
traffic engineering). The reader should distinguish between these two continuous-time metrics that are calculated from the simulator and the three metrics (queueing delay, loss rate, available bandwidth) that an IRC flow estimates during a measurement period using sampling.

We examine the effect of four important factors, with two values per factor, giving us 16 different simulation configurations. First, the ratio of average IRC load to the aggregate traffic load: 50% and 90%. Second, the distribution of average rates for the IRC flows: homogeneous (i.e., the same for all flows) and Zipf (shape parameter=1). Third, the aggregate capacity of ingress links relative to the total offered load: 105% and 125%. And fourth, the number of ingress links \( m \): two and four. Note that we only examine what happens when the ratio of IRC traffic is significant (50% or more). If there is only a small fraction of IRC traffic, the synchronization effects that we examine are of minor significance for the aggregate traffic, but they can be important for the IRC flows. Also, we do not simulate greatly overprovisioned or underprovisioned links because such conditions lead to either zero loss rate and stable path selections (overprovisioning) or persistent losses at all paths (underprovisioning). In other words, IRC is not needed if there is plenty of capacity in the ingress links, and also it cannot avoid congestion if there is not enough aggregate capacity. It is the range in the middle that is interesting and important in practice, because that is the most cost-effective operating regime.

### 3.5.2 Performance of FSP

We first study the performance of FSP under different values of the switching probability \( P \). Figures 7 and 8 shows the loss rate with FSP as a function of \( P \), for the homogeneous and Zipf flow rate distributions and for \( m=4 \) links. Each plot shows four curves, for all combinations of capacity (105% and 125%) and IRC traffic ratio (50% and 90%). The curves show the average loss rate across all IRC flows, as well
Figure 7: Loss rate of FSP as a function of $P$ (four ingress links) with homogeneous flow rates.

as the 99% confidence intervals. We show more points at the low end of $P$ (below 0.1) to illustrate the trend in that range.

First note that as $P$ increases above 20%, the loss rate increases significantly. This is due to synchronization of IRC flows. The synchronization becomes more prevalent with higher $P$ because flows tend to switch paths more aggressively. With DPS ($P=1.0$, not shown here), the loss rate would be excessively large. Second, practically any small value of $P$, between 0.001 and 0.1 is sufficient to avoid synchronization and path switching (shown in Figure 9), and to result in the minimum possible loss rate. Thus, FSP is robust in the selection of $P$. Another interesting observation is that even without any path switching ($P=0$) the loss rate is close to minimum. This is because, with such stationary background traffic, IRC flows distribute themselves uniformly across the ingress links and then they rarely see that the performance is significantly better in another path. As will be shown in the next section, the situation is very different when we consider nonstationary traffic load and rapid congestion events.

Some more observations from Figure 7 and Figure 8 follow. First, as expected, larger capacity leads to lower loss rate. Second, reducing the fraction of IRC traffic
Figure 8: Loss rate of FSP as a function of $P$ (four ingress links), and flow rates follows Zipf distribution.

decreases the loss rate because the synchronization of IRC flows has lesser impact on the aggregate load at the ingress links. Third, the Zipf distribution results in a lower average loss rate across all flows, because it is mostly the few large flows that experience major losses when there is synchronization. This also explains why the loss rate confidence intervals are much wider with the Zipf distribution. Due to space constraints we do not present results for the case of two links; the results are similar.

Figure 9: Switching frequency with FSP as a function of $P$ (four ingress links and homogeneous flow rate distribution).
Figure 10: Ratio $F/P$.

Figure 9 shows the switching frequency $F$ of FSP in simulations with homogeneous flow rate distribution and four ingress links. The results are similar for other configurations. First, note that $F$ is almost zero when $P$ is less than 10%, validating that even a small switching probability can avoid synchronization. Second, $F$ increases significantly as the path switching probability increases beyond 10%. There are two reasons for this. First, increasing $P$ means that IRC flows will get the opportunity to switch paths more often, if they need to. So, if the traffic variations at the ingress links remained the same as we increase $P$, then we would observe that $F$ increases linearly with $P$. On the contrary, Figure 10 shows that the ratio $F/P$ is not constant, and that it increases with $P$. This means that as we increase $P$, not only IRC flows get the opportunity to switch paths more often, but they also need to switch paths more often. This is explained as follows: increasing $P$ causes further synchronization among IRC flows, which causes larger fluctuations in the traffic of the ingress links. These traffic fluctuations trigger further IRC path switching and even greater synchronization. In other words, aggressive path switching creates a positive feedback loop between the synchronization of IRC flows and the traffic fluctuations in the underlying bottleneck links.
3.5.3 Parameterization of ASP, RRP and HRP

We conducted a similar study for ASP, RRP and HRP. In this section, we summarize the simulation results that resulted in the best parameters for these three algorithms, while the next section compares all path switching algorithms.

The ASP algorithm depends on two probabilities, $P_{hi}$ and $P_{lo}$. We examined the following pairs of $(P_{hi}, P_{lo})$: (0.7, 0.3), (0.8, 0.2), (0.9, 0.1), (0.95, 0.05), and (1.0, 0.0). The simulation results show that the loss rate and switching frequency decrease as the difference $(P_{hi} - P_{lo})$ increases. When we reach the extreme pair (1.0, 0.0), the loss rate slightly increases under several configurations. We thus choose (0.95, 0.05) as the parameters for ASP under stationary load. This setting means that ASP switches to another path almost certainly when the current path is much worse, but it stays in the current path, almost always, otherwise.

For the RRP algorithm, the parameter $T_{M}$ controls the range of the maximum routing period. We examined the $T_{M}$ range from $T_{m}$ to $10T_{m}$. The simulations results show that the loss rate and switching frequency decrease rapidly as $T_{M}$ increases away from $T_{m}$, but then they flatten out as $T_{M}$ becomes larger than $4T_{m}$. The exact transition point depends on the simulation configuration. We set $T_{M} = 7T_{m}$, because the loss rate and switching frequency show diminishing returns for larger values, while the algorithm becomes less responsive to dynamic load changes as $T_{M}$ increases.

For the HRP algorithm, the parameter $T_{H}$ controls the range of the hysteresis period. We examined the $T_{H}$ range from 0 to $40T_{m}$. Similar to RRP, the loss rate and switching frequency decrease rapidly at the beginning, when $T_{H}$ is small. After about $5T_{m}$, however, the curves flatten out. We set $T_{H}$ to $20T_{m}$, as a trade-off between performance and responsiveness.
3.5.4 Comparison of DPS, FSP, ASP, RRP and HRP

In this section, we compare the performance of the five IRC algorithms we consider under stationary load conditions, using the parameters given previously.

Figures 11 and 12 show the loss rate and the switching frequency of the five IRC algorithms in the $m=4$ link configuration. The results with other configurations, not shown here, have similar trends. First, note that in terms of loss rate the four probabilistic path switching algorithms (FSP, ASP, RRP, HRP) do much better than deterministic path switching (DPS), as they decrease the loss rate by an order of magnitude or more. Second, the randomized algorithms perform similarly, with ASP and FSP being slightly better than RRP and HRP.

The switching frequency results show similar trends. Without randomization, IRC path switching can cause major synchronization and oscillations. The four randomized algorithms have a clear difference in terms of switching frequency, however. FSP is the most stable, ASP comes next, while RRP and HRP are the least stable.

3.5.5 Summary

We found that introducing some randomness in the IRC path switching process can avoid synchronization and dramatically improve performance and stability compared to deterministic path switching. We also observed that, under stationary load, optimal performance and stability result from very conservative path switching. Finally, the exact form of randomization, or its parameters, do not seem to matter significantly, especially in terms of the resulting loss rate.

3.6 Evaluation of IRC algorithms - nonstationary load

The simulation results of Section 3.5 examined the performance of IRC algorithms under stationary conditions, where the average DRC traffic load, i.e., the background traffic with which IRC flows share the links of $D$, remains constant. We also need
Figure 11: Comparison of IRC algorithms—loss rate (four ingress links).
Figure 12: Comparison of IRC algorithms– switching frequency (four ingress links, homogeneous flow rates).

to understand, however, the performance of IRC systems under dynamic network conditions in which the background traffic varies rapidly due to random events such as BGP rerouting, link/router failures, flash crowds, arrival/departure of major flows, etc. Instead of evaluating such dynamic conditions with simulations of individual load changes, we prefer instead to investigate the behavior of IRC in the presence of cyclostationary background traffic. The latter is a special form of nonstationary traffic in which the average rate varies periodically. The benefit of this approach is that it allows us to examine the performance of IRC as we vary the time scale in which congestion persists. In other words, the following evaluation resembles a frequency-domain analysis of IRC behavior, rather than a time-domain transient analysis.

3.6.1 Simulation setup

The simulations in this section refer to a two-link configuration with homogeneous DRC and IRC traffic flows. The total capacity is barely enough to carry the aggregate traffic load. The periodic pattern of the load changes at the two links is shown in Figure 13. Initially \( t_0 \), the two links are equally loaded with IRC and DRC traffic. IRC flows do not switch paths because the two links offer the same performance. At some time \( t_1 \), the DRC traffic of link-2 (at the right) moves to link-1. This causes
**Figure 13:** The pattern of periodic load changes between two ingress links. At $t_1$ a block of DRC traffic moves from the right link to the left. At $t_3$, $T_w$ time units later, an equally sized block moves back to the right link.

major congestion at the latter, and so the IRC flows of link-1 gradually move to link-2 ($t_2$). The delay $t_2 - t_1$ depends on the aggressiveness (or responsiveness) of the IRC path switching algorithm. For FSP, a higher value of $P$ will reduce the delay $t_2 - t_1$. If $P=1$ (DPS), $t_2 - t_1$ will be as short as one routing period $T_r$. Then, at some time $t_3 = t_1 + T_w$, half of the DRC traffic at link-1 moves to link-2. Again, this causes major congestion at the latter, and so half of the IRC traffic gradually moves to link-1 ($t_4$), getting us back to where we started at $t_6$. Note that the term “gradually” is only true here if $P$ is much lower than one. Otherwise, the IRC traffic can experience oscillations before reaching the load-balanced configuration shown at $t_4$. This pattern can repeat periodically, with a period of $2T_w$, if the events at $t_1$ and $t_3$ occur every $2T_w$ time units.

Even though the previous traffic pattern is very artificial, it produces a periodic load variation that allows us to examine the performance of various IRC algorithms as a function of the period $T_w$. In particular, we are interested in the relation between $T_w$, which is the time scale in which congestion emerges, and $T_m=1$ second, which is the minimum time scale in which an IRC flow can detect congestion and react to it.

We examined the performance of the five IRC algorithms (DPS, FSP, ASP, RRP, and HRP) as $T_w$ varies from 0.5 seconds to 400 seconds. This range is sufficient to show the three important modes in the behavior of IRC. Each simulation has 100 flows, half of which are IRC flows. The total capacity of the two identical links is set
to 105% of the aggregate traffic load. The rest of the parameters are as in Section 3.5.

3.6.2 Performance of FSP under nonstationary load

Due to space constraints, we present more detailed results only for FSP with $P$ set to 0.01, 0.05, 0.1, 0.5, and 1.0 (DPS). The other three randomized algorithms perform similarly. We identify three distinct ranges of $T_w$ in terms of the resulting loss rate and trends.

First, Figure 14 shows what happens when $T_w$ is less than 5 seconds. Recall that the routing period as well as the measurement period of FSP are set to 1 second, meaning that in this range of $T_w$ congestion emerges almost with the same frequency with which FSP can detect whether it should switch paths. Note that IRC does very poorly when $P$ is high, 0.5 or higher, and $T_w$ is less than 2-3 seconds. The reason is that in that range FSP tries to “catch its tail”, switching between paths with almost the same frequency with which the background traffic moves between these paths. On the contrary, IRC does best when it rarely switches paths, i.e., when $P$ is very low or zero. Hence, IRC techniques will not improve performance (they will actually hurt performance) if they detect and react to congestion in the same time scales in
Figure 15: Loss rate of FSP for different values of $P$, $T_w \in [0, 30]$.

Figure 16: Loss rate of FSP for different values of $P$, $T_w \in [0, 400]$. 
which congestion emerges.

Second, Figure 15 (the non-shaded part) shows what happens when \( T_w \) is larger than 5 seconds and lower than 30 seconds. In this case, congestion emerges much less frequently compared to the IRC time scale (one second), but nevertheless it does emerge periodically and it is significant. The results are quite different now. The loss rate can be significantly reduced with IRC compared to static routing or \( P=0 \). Also, an aggressive form of FSP with \( P=0.5 \) performs better than the conservative path switching probabilities 0.1 and 0.05. Deterministic path switching, on the other hand, does not perform better, because it still causes synchronization of IRC flows. Hence, when IRC techniques are fast enough to detect congestion when the latter is still at its onset, IRC can significantly improve performance. In that regime, IRC can also be more aggressive in terms of path switching compared to stationary load conditions.

Finally, Figure 16 (the non-shaded part) shows what happens when \( T_w \) is larger than 30 seconds and lower than 400 seconds. In this case, the background traffic goes through major fluctuations only rarely. Thus, this case is not very different than the stationary load evaluation of Section 3.5. Indeed, we see that as \( T_w \) increases, conservative path switching does better. Eventually, as \( T_w \) tends to infinity, the best choice for \( P \) becomes 0.01. Hence, when congestion is a rare event, IRC is still beneficial but it should be quite conservative in terms of path switching.

### 3.6.3 Comparison of FSP, ASP, RRP and HRP under nonstationary load

Here, we compare the performance of the four randomized IRC algorithms (FSP, ASP, RRP and HRP) under nonstationary traffic load. In this set of simulations, the parameters of the ASP, RRP and HRP algorithms are as in Section 3.5. For FSP, we set \( P=0.5 \).

The results are shown in Figure 17. As was the case with FSP, all IRC algorithms
Figure 17: Loss rate with FSP, ASP, RRP and HRP as we vary $T_w$.

perform poorly when $T_w$ is very short, below 5 seconds. Also, the algorithms do not show major differences when $T_w$ is very large, say above 200 seconds, which is consistent with our earlier comparisons in Section 3.5. The more interesting range is when $T_w$ falls between 10 to 100 seconds. In that case, the ASP and RRP algorithms perform best, HRP follows, and FSP comes last. We should note however that these comparisons are somewhat dependent on the parameterization of the four algorithms.

3.6.4 Summary

The simulations of this section revealed some interesting observations. First, the effectiveness of IRC techniques, in general, depends on the time scale in which such techniques can detect and react to congestion relative to the time scales in which congestion persists. If congestion appears rapidly and it only lasts for a few seconds, it would be very hard for IRC systems to avoid it given that they also need a few seconds to detect congestion through measurements. In that case, it may be better to stay at the same path and deal with congestion through other means. On the other hand, if congestion lasts for many seconds, we expect that IRC systems can be fast enough to detect it and switch to another path.
3.7 *Heterogeneous IRC sources*

In the previous sections, we showed that the four randomized path switching algorithms perform well when all IRC flows use the same algorithm. In this section, we investigate how these algorithms perform in a heterogeneous environment. We also investigate the coexistence of deterministic path switching with randomized path switching. This is a critical question for the gradual deployment of the latter.

3.7.1 *Heterogeneous IRC algorithms*

In practice, we expect that different IRC systems will be using different path switching techniques, with diverse forms and parameters of randomization. To examine such a heterogeneous environment, we simulated several configurations where different IRC flows use FSP, ASP, RRP and HRP, each algorithm adopted by the same number of flows.

The results are only summarized here due to space constraints. First, we observed that in such a heterogeneous environment the difference, in terms of loss rate, between the four randomized algorithms is further decreased (i.e., heterogeneity causes assimilation). The reason is that, since all IRC flows share the same bottleneck links, they will all observe, on the average, the same loss rate. On the other hand, the IRC flows still do much better than the deterministic path switching flows as the latter experience persistent oscillations. Hence, we expect that even if different IRC vendors adopt different path switching techniques, the resulting traffic will be stable as long as there is some degree of randomization in their switching techniques.

3.7.2 *Coexistence of deterministic and randomized path switching*

In practice, some networks may continue to use deterministic path switching techniques, while others gradually deploy randomized path switching. The critical question in such an environment is whether IRC users have the incentive to switch to randomized IRC or whether they will do better for themselves being more aggressive.
We investigate this question simulating a fraction of randomized IRC traffic with the rest of the traffic doing deterministic path switching (DPS). Figure 18 shows the resulting loss rate when the randomized IRC algorithm is FSP with $P=0.01$. The results with other algorithms show similar trends. We start from 5% FSP and 95% DPS traffic and gradually increase the fraction of FSP traffic to 95%.

![Figure 18: Loss rate when FSP and DPS traffic coexist (four ingress links, homogeneous flow rates).](image)

First, note the large gap between the loss rate of DPS and FSP flows. When the ratio of FSP to DPS traffic is 5:95, the loss rate of the former is only about 40% of the loss rate of the latter. This performance difference indicates that when FSP and DPS flows coexist at a network, the FSP algorithm gives a major advantage to its users over the DPS algorithm. We also observe that with the increase of FSP traffic, the loss rates for both FSP and DPS flows decrease. This improvement of the overall performance results from the decreased fraction of DPS traffic, which leads to reduced synchronization.

These results are encouraging for two reasons. First, the substantial difference between FSP and DPS shows that multihomed networks will have a strong incentive to use IRC systems with randomized rather than deterministic path switching. Second,
the decreasing loss rate as the fraction of FSP traffic increases suggests that the gradual deployment of randomized IRC systems will benefit all traffic sharing the same bottlenecks.

3.8 Summary

In this chapter, we investigate possible traffic oscillation caused by IRC systems. First, when evaluating path performance with self-load unaware metrics, such as delay and loss rate, even a single IRC system can lead to sustained traffic oscillation. Next, we show that even when self-load aware metrics are used, traffic oscillation can still be caused by significant overlapping of measurement periods. We hope that this negative result will motivate further research in the appropriate design of IRC systems, as well as in measurement studies at multihomed networks that use IRC.

We then show that some simple randomization techniques in the path switching algorithm, as well as the use of available bandwidth measurements, can be effective in avoiding IRC-induced traffic oscillations. We examine the randomized path switching algorithm under both stationary and dynamics traffic load environment. At last, we show when randomized and deterministic switching IRC systems co-exist, randomized switching IRC systems out-perform the deterministic switching IRC systems.
CHAPTER IV

INTERACTIONS OF INTELLIGENT ROUTE CONTROL WITH TCP CONGESTION CONTROL

In previous chapter, we discussed the interaction between multiple IRC systems. In this chapter, we investigate the impact of IRC caused by the interaction between TCP congestion control and IRC.

Intelligent Route Control (IRC) technologies allow a multihomed network to dynamically select the best egress route for any destination network 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 however about the complex interactions between IRC and TCP congestion control. In this paper, we consider a simple dual-feedback model with two paths (primary and backup) and with a single source-destination network pair. In this model, 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 will be synergistic as long as IRC operates in larger time scales than TCP’s congestion detection (“separation of time scales” principle). In this case, IRC provides a significant benefit to TCP traffic, as long as it reacts to persistent congestion, major random losses, network outages, and other impairments that cannot be alleviated by TCP’s congestion control. 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. The paper also shows the significance of TCP throughput
prediction techniques in the context of IRC path switching.

4.1 Introduction/Model

We consider a model in which the egress TCP traffic of a stub network $S$ is controlled by an IRC-capable multihomed edge router. In particular, we focus on the aggregate TCP traffic $T(S, D)$ from $S$ towards a destination network $D$. The traffic $T(S, D)$ is subject to two closed-loop controllers: TCP congestion control at the flow level and IRC path control for the entire aggregate (see Figure 19).

![Diagram of dual-feedback model]

**Figure 19:** The dual-feedback model.

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 $S$ to $D$. IRC can switch to another 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 [26]. A typical value for $T_r$ in current IRC products is a few tens of seconds [17].
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 (§4.2). Then, we investigate the impact of sudden RTT changes on TCP throughput (§4.3). Next (§4.4), 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 §4.5. Finally, in §4.6, we put everything together and examine the conditions under which IRC is beneficial for TCP traffic.

4.2 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 $\text{RTT}_p$ and $\text{RTT}_b$, 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 [26]. 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 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 Figures 20 and 20, we illustrate what can happen when $T_r$ is either too short or too long. In Figure 20, 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 $T_m=200$msec. At $t=10$sec, we introduce a congestion event at the primary path caused by CBR cross traffic. Without IRC, or with $T_r=1$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
Figure 20: The effect of the IRC routing period $T_r$: Unnecessary path switching when $T_r$ is too short ($T_r=200\text{ms}$)

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 21, 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.

4.3 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
Figure 21: The effect of the IRC routing period $T_r$: Slow IRC reaction when $T_r$ is too long ($T_r=10\text{sec}$)

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 §4.5.

Suppose, without loss of generality, that the RTT of the primary path is lower, i.e., $\text{RTT}_p < \text{RTT}_b$. Note that the reverse path, from $D$ to $S$, 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 $S$ to $D$. 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.

4.3.1 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
Figure 22: Switching to a lower RTT path can cause Fast-Retransmit. 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 $N_o = K(\text{OWD}_b - \text{OWD}_p)$, 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 22. 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-Retrasmits 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.*

### 4.3.2 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 [50] for details). The RTO timer has a fixed OS-specific lower bound, which is often set to 200msec or 1sec [45].

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.

4.4 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 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).

4.4.1 Switching to higher avail-bw path

When a TCP flow moves from the backup to the primary path, its throughput will change from $R_b = W_b/\text{RTT}_b$ to $R_p = W_b/\text{RTT}_p$, 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.

![Graph showing throughput and aggregate window variations](image)

**Figure 23:** Congestion window and throughput variations when traffic switch from the backup path to the primary path.
We show such an event in the simulation of Figure 23. 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 \( \text{RTT} \frac{C_b - R_p}{LN} \) seconds, if the window of each connection increases by one packet per RTT (i.e., ignoring the effect of delayed ACKs).

### 4.4.2 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 24. 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.
Figure 24: Congestion window and throughput variations when traffic switch from the primary path to the backup path without induce congestion on the backup path.

event, and there is a time period in which the aggregate throughput is less than $C_b$. Figure 25 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 $C_b$.

Figure 25: Congestion window and throughput variations when traffic switch from the primary path to the backup path inducing congestion on the backup path.

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.

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

Figure 26 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

$$R_I = \frac{(R_b T_b + R_{pg} T_g)}{(T_b + T_g)} \quad (4)$$
and the average throughput with practical-IRC is

\[
R_P = \frac{R_b(T_b - T_{mb} + T_{mg}) + R_{pg}(T_g - T_{mg}) + R_{pb}T_{mb}}{T_b + T_g}
\]

(5)

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

\[
\frac{R_{pb} - R_b}{R_{pg} - R_b} < \frac{T_{mg}}{T_{mb}}
\]

(6)

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 (6) is always true.

4.5.2 Random loss impairments

Figure 27 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 28 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 [31].

### 4.6 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.

4.6.1 Outage impairments

Figure 29 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

$$R_N = \frac{R_{pb}T_b + R_{pg}T_g}{T_b + T_g}$$

(7)

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

$$\frac{R_b - R_{pb}}{R_{pg} - R_b} < \frac{T_{mg}}{T_b - T_{mb}}$$

(8)

Note that $R_{pg} > R_b$ and $T_b > T_{mb}$. For outages, since $R_{pb} = 0$, (8) can be written as

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

(9)

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 (9) is true, it is better to keep the traffic at the primary path, instead of switching to the backup path.

4.6.2 Random loss impairments

Figure 30 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.
Figure 29: The performance of practical-IRC with static routing (no-IRC) under outage impairment.

Figure 30: The performance of practical-IRC with static routing (no-IRC) under random loss impairment.
The analysis of the previous paragraph still applies, and condition (8) determines whether practical-IRC is better than no-IRC. Note that $R_{ph} > 0$ 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.

4.7 Summary

In this chapter, we studied the interaction of TCP congestion control and IRC systems. We show that when the measurement timescale is larger than TCP’s typical operation timescale, that is, the timescale of RTT and RTO, the interaction between TCP and IRC can be synergistic. We also show that IRC needs to examine the RTT difference between two candidate paths to avoid retransmissions and throughput loss. When switching between paths with different capacities, traffic should expect a transient reduction period of throughput after the path switching.

We then investigate the effect of measurement latency on closed-loop traffic under two types of path impairment: outage and random losses. The results indicate loss rate is not a good indicator for path performance, and promote throughput prediction as the metric for path switching. Last, we evaluate the overall impact of the interaction between TCP and IRC. We show that IRC is synergistic to TCP for outages that last longer than the measurement timescale or random losses with significant loss rate.
CHAPTER V

INTERDOMAIN INGRESS TRAFFIC ENGINEERING
THROUGH OPTIMIZED AS-PATH PREPENDING

In Interdomain Ingress Traffic Engineering (INITE), a “target” Autonomous System (AS) aims to control the ingress link through which the traffic of one or more upstream source networks flows to the target network or to its customers. Currently, there are few methodologies for systematic INITE. In practice, ISPs often attempt to manipulate, mostly in a trial-and-error manner, the AS-Path length attribute of upstream routes through a simple technique known as prepending (or padding). In this paper, we focus on prepending and propose a polynomial-time algorithm (referred to as OPV) that determines the optimal padding for an upstream route at each ingress link of the target network. Specifically, given a set of “elephant” source networks for a particular customer of the target network, and a set of maximum load constraints on the ingress links of the latter, OPV determines the minimum padding at each ingress link so that the load constraints are met, when it is feasible to do so. OPV requires as input an AS-Path length estimate from each source to each ingress link. We describe how to estimate this matrix, leveraging the BGP Looking Glass Servers that are abundant today for monitoring interdomain routing. To deal with unavoidable inaccuracies in the AS-Path length estimates, and also to compensate for the generally unknown BGP tie-breaking process in upstream networks, we develop a robust variation (RPV) of the OPV algorithm. We show that RPV manages to identify a padding vector that meets the given maximum load constraints, when it is feasible to do so, even in the presence of inaccurate AS-Path lengths and unknown BGP tie-breaking behavior.
5.1 Problem Statement and Formulation

In this section, we first describe the COP problem more formally, state the formulation’s key assumptions, and make some remarks regarding related operational issues.

| $\mathcal{T}$ | target network | $D$ | destination network |
| $S$ | source network | $M$ | number of sources |
| $\mathcal{L}$ | ingress link | $N$ | number of ingress links |
| $I$ | instance of COP problem | $U$ | upstream cloud |
| $S$ | source load vector | $C$ | link maxload vector |
| $P$ | AS-Path length matrix | $Q$ | tie-breaking matrix |
| $A$ | padding vector | $P(A)$ | padded length matrix |
| $L(A)$ | link assignment vector | $R(A)$ | link load vector |

Table 1: Main notations.

5.1.1 COP Problem Statement

Consider a target network $\mathcal{T}$ that aims to do INITE through prepending (see Figure 31). INITE is performed separately for each major customer $\mathcal{D}$ of $\mathcal{T}$, (but $\mathcal{D}$ and $\mathcal{T}$ can be the same if the latter is a stub network, or if $\mathcal{T}$ performs INITE to all ingress traffic). In the following, $\mathcal{D}$ is an implied destination network, so it is not included in the notation. Suppose that $\mathcal{T}$ has $N$ ingress links that can receive traffic for $\mathcal{D}$. Each ingress link $\mathcal{L}_j$ originates a route for $\mathcal{D}$, and advertises that route to its upstream BGP peer.

The traffic that is destined to $\mathcal{D}$ may be produced by a potentially large number of upstream source networks. Instead of considering individual source networks, which may be impractical, we focus on super-source ASes. A super-source AS $S$ receives traffic destined to $\mathcal{D}$ from potentially several source networks. In the following, we refer to super-source ASes simply as “source networks”, with the understanding that they may not be where the traffic actually originates from. Suppose that $M$ is the number of source networks for the traffic to $\mathcal{D}$, and let $s_i$ be the average traffic load forwarded by source $S_i$. $S=\{s_i, i = 1 \ldots M\}$ is the source load vector. The
identification of source networks and the estimation of the load vector \( S \) are discussed in § 5.3.

**Figure 31:** High-level architecture of ingress interdomain traffic engineering.

The effectiveness of any AS-Path prepending technique, including ours, is limited by the use of local routing policies expressed through the Local Preference attribute. The reason is that the Local Preference attribute has a higher priority in the BGP path selection process than the AS-Path length (see Table 2). Consider a source network \( i \), and suppose that \( i \) maintains a distinct route to each ingress link \( \mathcal{L}_j \) of \( \mathcal{T} \). This can be achieved if \( \mathcal{T} \) advertises a unique *wayfinding prefix* at each link \( \mathcal{L}_j \) (we return to this point in § 5.3). The selected routes from source \( i \) to the \( N \) ingress links of \( \mathcal{T} \) form an AS path tree denoted by \( \mathcal{U}_i \), as illustrated in Figure 32. In the following, we assume that the branching nodes of this tree do not select their best routes to the \( N \) ingress links based on Local Preference. In other words, the candidate routes for \( \mathcal{T} \) at each branching node of \( \mathcal{U}_i \) are assigned the same Local Preference value. The design of automated ways for examining the previous assumption is an important task for future work.
Figure 32: Branching networks, labeled B in the diagram, in the upstream cloud \( \mathcal{U}_i \)

One of the key parameter is the BGP tie-breaking behavior in the upstream cloud. As shown in Table 2, when two routes have the same Local Preference and AS-Path length, an ordered-list of five other criteria is used to choose the best route. We do not attempt to estimate or infer all the parameters that affect those tie-breaking criteria; such a task would be extremely difficult and error-prone. We assume however that the tie-breaking behavior is determined by a matrix \( Q \). Specifically, suppose that the two routes to \( \mathcal{D} \) that originated from links \( \mathcal{L}_j \) and \( \mathcal{L}_k \) are received with the same AS-Path length by an AS \( \mathcal{R} \) in the upstream cloud \( \mathcal{U}_i \). Then, \( \mathcal{R} \) will choose \( \mathcal{L}_j \) if \( Q_{i,j} < Q_{i,k} \), and \( \mathcal{L}_k \) otherwise. Different columns of the same row of \( Q \) must be different, while their absolute values do not matter. Notice that \( Q \) is just a model of the BGP tie-breaking behavior; it is not related to a real BGP attribute. Furthermore, \( Q \) represents a \textit{globally consistent tie-breaking behavior}, i.e., we assume that a tie between two routes to \( \mathcal{D} \) with the same AS-Path length is broken in the same way in any AS in \( \mathcal{U}_i \). In the next section, when we study the OPV algorithm, we assume that \( Q \) is known. Then, in § 5.4, we relax that assumption and attempt to find a robust padding vector even when \( Q \) is completely unknown.
Table 2: Criteria for BGP best route selection.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Higher local preference</td>
</tr>
<tr>
<td>2.</td>
<td>Shorter AS path</td>
</tr>
<tr>
<td>3.</td>
<td>Lower origin type</td>
</tr>
<tr>
<td>4.</td>
<td>Lower MED value</td>
</tr>
<tr>
<td>5.</td>
<td>E-BGP routes preferred over I-BGP routes</td>
</tr>
<tr>
<td>6.</td>
<td>Lower IGP metric to next-hop</td>
</tr>
<tr>
<td>7.</td>
<td>Lower BGP router ID</td>
</tr>
</tbody>
</table>

Another key parameter is the AS-Path length matrix $P$. The element $P_{i,j}$ is the length of the shortest AS-Path from the source network $S_i$ to the ingress link $L_j$, where $P_{i,j}$ is a positive integer. Any ties in the length of the shortest AS-Path are broken based on the $Q$ matrix. Note that $P_{i,j}$ is the “unpadded” AS-Path length, i.e., it should be measured before the target network applies any prepending. Estimation techniques for the matrix $P$ are given in §5.3. For now we assume that $P$ is completely and accurately known.

The ingress link $L_j$ can increase the length of the AS-Path attribute by $a_j$, where $a_j$ is a non-negative integer, through prepending. The vector $A = \{a_j, j = 1 \ldots N\}$ is the padding vector. Given a padding vector $A$, the “padded” AS-Path length of the route to link $L_j$ is $P_{i,j}(A) = P_{i,j} + a_j$.

Based on the previous model, we can now prove the following fact.

**Lemma 5.1.1** Given a padded AS-Path length matrix $P(A)$ and a tie-breaking matrix $Q$, the traffic of a source network $S_i$ will enter the target network $T$ through the ingress link $L_j$, where

$$j = \arg \min_{1 \leq i \leq N} P_{i,j}(A)$$

(10)

If there is a tie in (10) between links $L_j$ and $L_k$, then $Q_{i,j} < Q_{i,k}$.

**Proof 5.1.1** In the simplest case, $S_i$ receives directly from $T$ the route to link $L_j$. Since that route has the minimum AS-Path length (or, in case of a tie, it is preferred), $S_i$ will select that route for reaching $T$ and the Lemma is proven.
Suppose now that $S_i$ does not directly receive the route to $L_j$. Let $R$ be an AS in $U_i$, residing in the shortest path from $S_i$ to $L_j$. Then, $P_{i,j}(A) = H_{i,r} + H_{r,j}(A)$, where $H_{i,r}$ is the length of the shortest AS-Path from $S_i$ to $R$, and $H_{r,j}(A)$ is the length of the shortest AS-Path from $R$ to $L_j$. $S_i$ can only receive the $L_j$ route if $R$ selects that route to reach $T$. To prove the Lemma, we need to show that $R$ cannot select another route to reach $T$.

Suppose that $R$ also receives a route to $T$ through link $L_k$, and selects that route. This means that $H_{r,k}(A) < H_{r,j}(A)$, where $H_{r,k}(A)$ is the length of the shortest AS-Path from $R$ to $L_k$. Thus, $H_{i,r} + H_{r,k}(A) < H_{i,r} + H_{r,j}(A) = P_{i,j}(A)$.

Case-1: $S_i$ receives the route to $L_k$ also through $R$. Then $P_{i,k}(A) = H_{i,r} + H_{r,k}(A)$, and so $P_{i,k}(A) < P_{i,j}(A)$. This contradicts the definition of $L_j$ in (10).

Case-2: $S_i$ receives the route to $L_k$ through another path. Then $P_{i,k}(A) \leq H_{i,r} + H_{r,k}(A)$, where the equality is broken in favor of that other path. Then, $P_{i,k}(A) \leq P_{i,j}(A)$, which again contradicts (10).

Based on the previous Lemma, we can now define the link assignment vector $L(A) = \{l_i(A), i = 1 \ldots M\}$, where $l_i(A)$ is the link $L_j$ that source $S_i$ selects based on (10). From the link assignment vector $L(A)$, the expected load at an ingress link $j$ is

$$r_j(A) = \sum_{k=1 \ldots M: l_k(A) = j} s_k \quad (11)$$

The vector $R(A) = \{r_j(A), j = 1 \ldots N\}$ is the link load vector.

The type of INITE that we consider is based on a set of maximum load constraints for each ingress link and each customer $D$. Specifically, let $c_j$ be the maximum traffic load (maxload) allowed at link $j$, with $C = \{c_j, j = 1 \ldots N\}$ being the corresponding link maxload vector. A link $j$ is overloaded if $r_j(A) > c_j$. When none of the $N$ ingress links is overloaded, we say that $A$ is acceptable.

The COP problem can be now stated as follows. **Given an instance $I = (S, C, P, Q)$, determined by the source load vector $S$, the link maxload vector $C$, the AS-Path length...**
matrix $P$, and the tie-breaking matrix $Q$, is there an acceptable padding vector $A$?

When this is the case, determine the optimal padding vector $A^*$, such that

$$\sum_{j=1}^{N} a_j^* \leq \sum_{j=1}^{N} a_j$$

(12)

across all acceptable padding vectors $A$. When there is an acceptable padding vector for a given instance $I$, we say that $I$ is feasible; otherwise $I$ is infeasible. The reasoning behind the previous optimality objective is to avoid unnecessary padding, given that excessive padding in practice sometimes triggers upstream route filtering.

Example 1 illustrates the COP problem.

---

Instance $I$:

$$S = \begin{bmatrix} 1 \\ 2 \\ 4 \end{bmatrix}, \quad C = [4,4], \quad P = \begin{bmatrix} 1 & 2 \\ 3 & 2 \\ 2 & 5 \end{bmatrix}, \quad Q = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}$$

**Without prepending:** $A=[0,0]$

$$L(A) = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}, \quad R(A) = [5,2].$$

Link $L_1$ is overloaded.

**After prepending:** $A=[2,0]$

$$P(A) = \begin{bmatrix} 3 & 2 \\ 5 & 2 \\ 4 & 5 \end{bmatrix}, \quad L(A) = \begin{bmatrix} 2 \\ 2 \\ 1 \end{bmatrix} \quad \Rightarrow \quad R = [4,3].$$

Acceptable padding vector.

---

**Example 1:** A simple example of COP problem.
5.1.2 Remarks

Together with the previously mentioned assumptions, we also make the following assumptions regarding operational issues that may be relevant in practice:

- The destination network $D$ is a customer of $T$ only, so there is no way that prepending in $T$ can shift the traffic away from $T$ to a different provider. For multihomed destinations, our techniques would still apply as long as $T$ is the primary provider, and any secondary providers of $D$ are used only as a backup. Moreover, with a small modification, our algorithm can also solve the case that $D$ is multihomed, which will be explained later in this paper.

- The time scales in which $T$ adjusts its padding vectors are relatively short compared to the time scales in which the routes from the source networks to $D$ change. Previous work has shown that BGP routes of major traffic sources tend to be stable for days or weeks [60].

- Some ISPs have an agreement with their peers that they will announce the same AS-Path to a certain destination through all their peering links. If that is the case, $T$ can group together all ingress links to each peer, and then apply the proposed algorithms at the level of link groups.

5.2 Optimal Padding Vector Algorithm

In this section, we first present the OPV algorithm, and then prove that it can determine the optimal padding vector $A^*$, when the given instance $I$ is feasible, in polynomial time.

5.2.1 Optimal Padding Vector (OPV) algorithm

The OPV algorithm takes as input an instance $I = (S, C, P, Q)$ of the COP problem. OPV examines the feasibility of a single padding vector $A^{(m)}$ in every iteration.
Algorithm 1 OPV \((I=(S, C, P, Q))\)

1: Compute \(\Phi\) from (13) and (14)
2: \(A^{(0)} = [0, \ldots, 0]\);
3: for \(m = 0\) to \(\Phi-1\) do
4: \hspace{1em} Compute \(L(A^{(m)})\) from (10)
5: \hspace{1em} Compute \(R(A^{(m)})\) from (11)
6: \hspace{1em} if \(A^{(m)}\): acceptable (i.e., \(\forall j, r_j(A^{(m)}) \leq c_j\)) then
7: \hspace{2em} return \(A^{(m)}\)
8: \hspace{1em} end if
9: \hspace{1em} \(A^{(m+1)} = A^{(m)}\)
10: \hspace{1em} Identify an overloaded link \(j\), i.e., \(r_j(A^{m}) > c_j\)
11: \hspace{1em} \(a_j^{(m+1)} = a_j^{(m)} + 1\)
12: end for
13: return \(I\): Infeasible

\(m\), starting from the zero padding vector. With each padding vector that is not acceptable, the algorithm identifies an overloaded link, and then increments the corresponding padding element. The exact selection of the overloaded link does not matter. The algorithm exits either when it has found an acceptable padding vector, or when it has completed \(\Phi\) iterations.

The bound \(\Phi\) is computed as follows:

\[
\phi_j = \max_{1 \leq s \leq M} \max_{1 \leq i \leq N} (P_{s,i} - P_{s,j})
\]  \hspace{1em} (13)

and

\[
\Phi = N + \sum_{1 \leq j \leq N} \phi_j - \min_{1 \leq j \leq N} \phi_j
\]  \hspace{1em} (14)

Note that \(\Phi\) can be computed in polynomial time, as shown in Lemma 5.2.3.

As mentioned earlier in the paper, a minor change can make the algorithm applicable on multihomed network \(D\) of \(T\). When \(D\) is multihomed to other ISPs with the shortest path \(\bar{p}_i\) to source \(S_i\), we change the bound \(\phi\) to be the minimum of Equation 13 and \(\bar{p}_i\). Thus, \(T\) will not “over-prepend” the path so that it pushes the traffic to other providers of \(D\).

We prove later in this section two important properties of OPV. First, when \(I\) is feasible, OPV reports the optimal padding vector \(A^*\), defined in (12). Second,
when $I$ is infeasible, OPV exits after $\Phi$ iterations reporting that indeed, there is no acceptable padding vector. Example 2 shows the iterations of OPV in the case of a feasible instance.

### 5.2.2 Properties of the OPV algorithm

The following lemma proves that, with a feasible instance, the optimal padding vector $A^*$ has at least one zero element.

**Lemma 5.2.1** Suppose that the instance $I$ is feasible, with optimal padding vector $A^*$. There exists a link $j$ with $a_j^* = 0$.

**Proof 5.2.1** Suppose $a_j^* > 0$ for all links $i$. Let $a_k^* = \min_{1 \leq i \leq N}(a_i^*)$. Construct a padding vector $A'$ such that $a_i' = a_i^* - a_k^*$. From (11), $R(A') = R(A^*)$. Since $A^*$ is acceptable, $A'$ is also acceptable. But $\sum a_i' < \sum a_i^*$, which contradicts that $A^*$ is the optimal padding vector. So, there must exist a link with $a_j^* = 0$.

A key property of OPV, proven next, is that, in a feasible instance, if an element $k$ of the padding vector reaches in some iteration $m$ the value that the corresponding optimal padding vector element has, i.e., if $a_k^{(m)} = a_k^*$, then the link $k$ will not be overloaded in any subsequent iteration, and so its padding element will remain at $a_k^*$.

**Lemma 5.2.2** Suppose that the instance $I$ is feasible, with optimal padding vector $A^*$. If the OPV padding vector in iteration $m$ is such that $a_j^{(m)} \leq a_j^*$ for all links $j$ and there exists a link $k$ such that $a_k^{(m)} = a_k^*$, then in all subsequent iterations $n$ ($n > m$), $a_k^{(n)} = a_k^{(m)} = a_k^*$.

**Proof 5.2.2** We prove the lemma by contradiction. Suppose that the padding of link $k$ is the first to exceed the corresponding value $a_k^*$ of $A^*$ after the $n$-th iteration. In other words, in iteration $n$ we have that $a_k^{(n)} = a_k^*$ and $a_j^{(n)} \leq a_j^*$ for all $j$, while in the next iteration $a_k^{(n+1)} > a_k^*$. 

86
Instance $I$: 

$$S = \begin{bmatrix} 4 \\ 3 \end{bmatrix}, \quad C = [2, 8, 2], \quad P = \begin{bmatrix} 1 & 2 & 2 \\ 1 & 2 & 2 \end{bmatrix}, \quad Q = \begin{bmatrix} 2 & 3 & 1 \\ 1 & 3 & 2 \end{bmatrix}$$

Bound $\Phi$: $N + \sum \phi_j - \min \phi_j = 3 + (1 + 0 + 0) - 0 = 4$

**Iteration 0:** $A^{(0)} = [0, 0, 0]$

Then $L(A^{(0)}) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, $R(A^{(0)}) = [7, 0, 0]$.

Link 1 is overloaded; increase $a_1$ by 1.

**Iteration 1:** $A^{(1)} = [1, 0, 0]$

Then $L(A^{(1)}) = \begin{bmatrix} 3 \\ 1 \end{bmatrix}$, $R(A^{(1)}) = [3, 0, 4]$.

Link 1 is overloaded; increase $a_1$ by 1.

**Iteration 2:** $A^{(2)} = [2, 0, 0]$,

Then $L(A^{(2)}) = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$, $R(A^{(2)}) = [0, 0, 7]$.

Link 3 is overloaded; increase $a_3$ by 1.

**Iteration 3:** $A^{(3)} = [2, 0, 1]$,

Then $L(A^{(3)}) = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$, $R(A^{(3)}) = [0, 7, 0]$.

No overloaded links.

**Return vector:** $A = [2, 0, 1]$.

**Example 2:** OPV goes through three iterations to find the optimal solution.
According to OPV, the padding of link $k$ can only be increased if that link is overloaded. So, in iteration $n$ there was at least one source $s$ such that $l_s(A^{(n)}) = k$, but $l_s(A^*) = h \neq k$. The fact that source $s$ preferred link $k$ over link $h$ in iteration $n$ means that

$$P_{s,k}(A^{(n)}) = P_{s,k} + a_{k}^{(n)} \leq P_{s,h} + a_{h}^{(n)}$$

where the case of equality is broken in favor of link $k$ through the $Q$ matrix, and so

$$P_{s,k} - P_{s,h} \leq a_{h}^{(n)} - a_{k}^{(n)}$$  \hspace{1cm} (15)

In the optimal vector $A^*$,

$$P_{s,k}(A^*) = P_{s,k} + a_{k}^* > P_{s,h} + a_{h}^*$$

and so

$$P_{s,k} - P_{s,h} > a_{h}^* - a_{k}^*$$  \hspace{1cm} (16)

From (15) and (16) we get that $a_{h}^* - a_{k}^* < a_{h}^{(n)} - a_{k}^{(n)}$. But $a_{k}^* = a_{k}^{(n)}$, and so $a_{h}^* < a_{h}^{(n)}$, which contradicts our earlier assumption.

The following lemma shows that, for a feasible instance, each optimal padding element is upper bounded by a certain function of the AS-Path length matrix that can be computed in polynomial time.

**Lemma 5.2.3** Suppose that the instance $I$ is feasible, with optimal padding vector $A^*$. Then, for any link $j$ the corresponding optimal padding is bounded by

$$a_j^* \leq \phi_j + 1$$  \hspace{1cm} (17)

where $\phi_j$ is given by (13).

**Proof 5.2.3** We prove the lemma by contradiction. Suppose that there exists a link $i$ with $a_i^* > \phi_i + 1$. Let us construct a new padding vector $A'$ such that $a_j' = a_j^*$ for all links $j \neq i$, and $a_i' = \phi_i + 1$. We prove next that $A'$ is also acceptable.
First, suppose that \( r_i(A') > r_i(A^*) \). This means that there exists a source \( s \) with 
\( l_s(A') = i \) and \( l_s(A^*) \neq i \). So, for all links \( j \neq i \)

\[
P_{s,i} + a'_i = P_{s,i} + \phi_i + 1 \leq P_{s,j} + a_j^*
\]

where all ties (equalities) are broken in favor of link \( i \), and so

\[
\phi_i + 1 - a_j^* \leq P_{s,j} - P_{s,i} \leq \phi_i
\]

which means that \( a_j^* \geq 1 \) for all links \( j \neq i \). We also assumed that \( a_i^* > \phi_i + 1 > 1 \),
and so \( a_j^* \geq 1 \) for all links \( j \) (including \( i \)). This contradicts Lemma 5.2.1.

From the previous contradiction, we have that \( r_i(A') = r_i(A^*) < c_i \), i.e., link \( i \)
is not overloaded with \( A' \). Any other link \( j \neq i \) cannot be overloaded with \( A' \) either,
because those links are not overloaded with \( A^* \), \( a'_j = a_j^* \) for all \( j \neq i \), and \( a_i^* < a_i^* \).
So, the padding vector \( A' \) is acceptable, which contradicts (12) because \( \sum a_j^* < \sum a_j^* \).
Consequently, \( a_j^* \leq \phi_j + 1 \) for all links \( j \).

The following theorem gives the main result for the OPV algorithm, for the case
of feasible instances.

**Theorem 5.2.4** Suppose that the instance \( I \) is feasible, with optimal padding vector \( A^* \). The OPV algorithm returns the vector \( A^* \) as its final outcome in polynomial time.

**Proof 5.2.4** Note that the OPV algorithm starts from the zero vector and it increments only one padding element in every iteration. Furthermore, Lemma 5.2.2 shows that if an element of the OPV padding vector has reached its corresponding value in \( A^* \), then it will not increase any further in subsequent iterations. Consequently, it is
certain that after \( 1 + \sum a_j^* \) iterations the OPV algorithm will terminate, reporting the
optimal padding vector \( A^* \) as its final outcome.
From Lemma 5.2.3, we get that the number of required iterations is \(1 + \sum_{j=1}^{N} a_j^* \leq (N + 1) + \sum_{j=1}^{N} \phi_j\). We also know from Lemma 5.2.1, however, that at least one element of \(A^*\) is zero. In the worst-case (maximum number of iterations), that element can be the link with the minimum \(\phi_j\) term. Thus, the required number of iterations becomes

\[
1 + \sum_{j=1}^{N} a_j^* \leq (N + 1) + \sum_{j=1}^{N} \phi_j - (\min_{1 \leq j \leq N} \phi_j + 1) = \Phi
\]  

(18)

The following theorem gives the main result for the OPV algorithm in the case of infeasible instances.

**Theorem 5.2.5** If instance \(I\) is infeasible, then the OPV algorithm detects that there is no acceptable padding vector in polynomial time.

**Proof 5.2.5** Note that the OPV algorithm fails to find an acceptable padding vector after \(\Phi\) iterations, it exits reporting that the instance is infeasible.

If \(I\) was feasible, then, from Theorem 5.2.4, OPV would have reported \(A^*\) in at most \(\Phi\) iterations. Since an acceptable padding vector has not been found after \(\Phi\) iterations, then it does not exist.

Example 2 demonstrates that the bound \(\Phi\) of (18) is actually tight. The value of \(\Phi\) in that example is 4.

### 5.3 Estimation of Input Parameters

In this section, we discuss the two key unknown inputs of the OPV algorithm, namely the set of super-source networks and the corresponding source load vector \(S\), and the length estimation matrix \(P\). As mentioned in § 5.1, we do not attempt to estimate the tie-breaking matrix \(Q\); the algorithm of the following section determines a robust padding vector independent of \(Q\). Also, the maxload vector \(C\) is supposed to be known given that it is chosen by the target network operator.
5.3.1 Selection of super-sources

We assume that the ingress routers of $\mathcal{T}$ collect statistics (with Cisco’s NetFlow for instance) of the arriving traffic, aggregated by source network; this is a common requirement for any type of traffic engineering. Because there may be too many source networks for a given customer $\mathcal{D}$, or because those networks may not be in upstream clouds, $\mathcal{T}$ can map one or more large sources of traffic to a single super-source $\mathcal{S}$. The requirement is that $\mathcal{S}$ has to be in the AS-Path from the actual source networks to any ingress link of $\mathcal{T}$.

To examine the former requirement, $\mathcal{T}$ can use AS-level topology maps, constructed with multiple vantage points as described in [65]. For example, the NetFlow data may show that several major sources of traffic for $\mathcal{D}$ belong to three stub ASes that are all customers of a regional or tier-2 provider AS-X, and that those three stub networks are not multi-homed (i.e., they can only reach $\mathcal{T}$ through AS-X). In that case, AS-X can be considered as a super-source $\mathcal{S}_i$. The corresponding source load element $s_i$ would be estimated as the sum of the load estimates of the actual sources, measured with NetFlow. The previous procedure can be applied to detect $M$ super-sources, capturing a large fraction of the traffic to $\mathcal{D}$.

5.3.2 Estimation of AS-Path length matrix

Recall that $P_{i,j}$ is the AS-Path length from source $\mathcal{S}_i$ to the ingress link $\mathcal{L}_j$. To estimate $P_{i,j}$, $\mathcal{T}$ has to originate a route for a \textit{wayfinding prefix} at each ingress link $\mathcal{L}_j$. A wayfinding prefix can be just the IP address of $\mathcal{L}_j$; if that is an unacceptably long prefix, the wayfinding prefix can cover instead a small part of the target network’s address space that includes $\mathcal{L}_j$ but no other ingress links. The same set of wayfinding prefixes can be used for estimating the matrix $P$ for all customers of $\mathcal{T}$.

To estimate $P_{i,j}$, the operator of $\mathcal{T}$ can use one of the four following techniques, in the given order. The first three techniques rely on \textit{Looking Glass Servers}. LGS’s are
abundant in the Internet today, providing a “peek” inside a network’s BGP routing tables. They are mostly used by ISPs for problem diagnosis and monitoring of inter-domain routing. The publicly available LGS’s that are listed in www.traceroute.org include 273 servers that in some cases provide access to multiple routers within an AS [37]. It is likely however that major ISPs have private access to even more LGS’s in remote networks, to accommodate synergistic problem diagnosis.

Figure 33: Four estimation techniques for the AS-Path length.

5.3.2.1 LGS in $S$

The source network $S$ may include an LGS. This is the ideal case, and it leads to the most accurate estimation. In the example of Figure 33, AS1 is a source network that deploys an LGS. $T$ can just query that LGS for the AS-Path to each wayfinding prefix.

5.3.2.2 LGS in a customer of $S$

Here, a customer of $S$ deploys an LGS. In that case, $T$ can query that LGS for each wayfinding prefix, and then remove from the returned AS-Paths the part that includes that customer AS. In Figure 33, AS4 is a customer of the source network AS5. AS4 deploys an LGS.
5.3.2.3  LGS in the path from $T$ to $S$

Another possibility is that $T$ can locate an LGS in a network in the reverse path, from $T$ to $S$. In Figure 33, AS7 is a source network and AS8 is a network deploying an LGS in the reverse path. In that case, $T$ can estimate the AS-Path length $P_{ij}$ as the sum of the AS-Path lengths from the LGS to $L_j$ and from the LGS to $S$. This technique assumes that the AS-Paths from the LGS to $S$ and from $S$ to the LGS have the same length.

5.3.2.4  Reverse path estimation

When none of the previous techniques is applicable, $T$ can just estimate the length of the AS-Path from $S$ to $L_j$ based on the length of the reverse path from $L_j$ to $S$. That reverse path is of course directly available from the border router at $L_j$. The problem with this technique is that it relies on the symmetry of the forward and reverse AS-Paths, which is often not the case in the Internet. On the positive side, the technique still produces a correct estimate if the two paths have the same length, even if those paths are different. This case is represented in the path between $T$ and AS11 in Figure 33.

AS-Path length distribution and estimation errors  The first two of the four previous techniques would give no estimation errors, because the corresponding LGS’s report directly the AS-Path from the source to the ingress links of the target network.

The third and fourth techniques, on the other hand, depend on the symmetry assumption, and they will probably suffer from estimation errors. To quantify these errors, we used 79 of the publicly available LGS’s listed at [37]. We estimated the AS-Path length from every LGS to each of the 78 other LGS’s using the previous two techniques: “LGS in the reverse path” (when applicable), and “reverse path estimation” (always applicable). Then, we compared each AS-Path length estimate
with the length of the actual AS-Path, as that was reported in the remote LGS.

![Estimated AS-Path Length (hops)](image)

**Figure 34:** Empirical probability distribution function of estimated AS-Path length.

Figure 34 shows the empirical probability density function for the estimated AS-Path lengths. Note that about 90-95% of the paths are up to 6 AS hops, including any prepending, while 60% of the paths are either 4 or 5 hops.

**Table 3:** Probability (%) of AS-Path length estimation error $e$, conditioned on the estimated AS-Path length $\hat{p}$ (both $e$ and $\hat{p}$ measured in AS hops).

<table>
<thead>
<tr>
<th>$\hat{p}$</th>
<th>-6</th>
<th>-5</th>
<th>-4</th>
<th>-3</th>
<th>-2</th>
<th>-1</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\hat{p}=3$</td>
<td>0.2</td>
<td>0.8</td>
<td>2.5</td>
<td>2.3</td>
<td>4.4</td>
<td>10.6</td>
<td>78.0</td>
<td>1.3</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$\hat{p}=4$</td>
<td>0.0</td>
<td>1.0</td>
<td>3.8</td>
<td>2.0</td>
<td>3.2</td>
<td>12.4</td>
<td>70.0</td>
<td>7.4</td>
<td>0.3</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$\hat{p}=5$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.4</td>
<td>0.3</td>
<td>12.4</td>
<td>64.1</td>
<td>11.8</td>
<td>3.8</td>
<td>0.1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$\hat{p}=6$</td>
<td>0.0</td>
<td>0.2</td>
<td>0.1</td>
<td>1.9</td>
<td>1.9</td>
<td>2.7</td>
<td>9.0</td>
<td>57.8</td>
<td>19.1</td>
<td>5.5</td>
<td>0.6</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$\hat{p}=7$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.5</td>
<td>1.0</td>
<td>2.0</td>
<td>12.1</td>
<td>45.5</td>
<td>21.2</td>
<td>11.1</td>
<td>6.6</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$\hat{p}=8$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.5</td>
<td>1.5</td>
<td>4.5</td>
<td>34.8</td>
<td>22.7</td>
<td>13.6</td>
<td>16.7</td>
<td>4.5</td>
<td>0.0</td>
</tr>
<tr>
<td>$\hat{p}=9$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>2.5</td>
<td>7.5</td>
<td>25.0</td>
<td>37.5</td>
<td>25.0</td>
<td>2.5</td>
</tr>
<tr>
<td>$\hat{p}=10$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>13.3</td>
<td>6.7</td>
<td>40.0</td>
<td>40.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$\hat{p}=11$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>25.0</td>
<td>50.0</td>
<td>25.0</td>
<td></td>
</tr>
<tr>
<td>$\hat{p}=12$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>100.0</td>
<td></td>
</tr>
</tbody>
</table>

The unconditional estimation error for the previous two estimation techniques is as follows: 60-65% of the estimates have zero error, 85-90% have an error of less than
±1 hop, and 95% have an error of less than ±2 hops. More importantly, Table III shows the conditional probability of an estimation error \( e \), given the estimated AS-Path length \( \hat{p} \). The estimation error \( e \) is measured as \( \hat{p} - p \), where \( p \) is the actual AS-Path length. The conditional probability that an estimate is error-free if \( \hat{p} \) is 3 or 4 hops is 78% and 70%, respectively, which is higher than the corresponding unconditional accuracy. Similarly, the conditional probability that an estimate has an error of less than ±1 hop if \( \hat{p} \) is 5 or 6 hops is 88% and 86%, respectively. We use this empirical error distribution, as well as the AS-Path length distribution of Figure 34, in the robustness study of the next section.

5.4 Robust Padding Vector Algorithm

The previous section showed that the estimate of the AS-Path length matrix \( P \) may include errors, and it gave us an empirical distribution for the estimation error probability. In this section, we first describe an algorithm that aims to determine a robust padding vector in the presence of these errors, and then evaluate the effectiveness of that algorithm with simulations.

5.4.1 Robust Padding Vector Algorithm

Recall that an instance \( I \) of the COP problem is defined by the set \( (S, C, P, Q) \). The OPT algorithm of § 5.2 was developed to find an optimal padding vector \( A^* \) for \( I \). In practice, we do not have a way to estimate \( Q \), and \( P \) may include estimation errors. To deal with these two issues, we develop the RPV (Robust Padding Vector) algorithm.

The basic idea in RPV is the following. Suppose that our original estimate of \( P \) is \( P_0 \). \( P_0 \), together with an arbitrary tie-breaking matrix \( Q \) and the given \( S \) and \( C \) vectors, form the instance \( I_0 \) that we start from. Another important instance is \( I_0^* \); this is the unknown actual instance that is based on the correct AS-Path length
matrix and the real tie-breaking behavior. Even though \( I_a \) is unknown, it belongs to the space \( \mathcal{I} \) of all possible instances that can be generated by applying a given error model to \( P_0 \), and by considering an arbitrary tie-breaking behaviors.

First, RPV generates a subset \( \mathcal{I}_X \) of \( \mathcal{I} \) that consists of \( X \) feasible instances. For each instance \( I_i \) in \( \mathcal{I}_X \), the corresponding optimal padding vector (computed with OPV) is \( A_i \). The set of padding vectors \( \mathcal{A} = \{ A_i, i = 1 \ldots X \} \) is our candidates for the desired robust solution. To generate padding vectors that differ significantly, we create the subset \( \mathcal{I}_X \) by applying the AS-Path length estimation error model only to the largest sources of \( I_0 \). Errors that correspond to small sources (relative to the rest of the sources) may not have an impact on the resulting padding vector.

Second, RPV generates a large subset \( \mathcal{I}_Y \) of \( \mathcal{I} \) that includes \( Y \) instances. The fraction of feasible instances in \( \mathcal{I}_Y \) is the feasibility index of \( \mathcal{I}_Y \). If \( Y \) is sufficiently large, the feasibility index estimates the probability that the actual instance \( I_a \) is feasible.

Third, for each candidate padding vector \( A_i \) in \( \mathcal{A} \), RPV measures the fraction of feasible instances in \( \mathcal{I}_Y \) for which \( A_i \) is acceptable. That fraction is the robustness index of \( A_i \). If \( Y \) is large, the robustness index estimates the probability that the corresponding padding vector \( A_i \) is acceptable for the actual instance \( I_a \), conditioned on the fact that the latter is feasible.

Eventually, RPV reports the padding vector \( \hat{A} \) with the maximum robustness index. The reason is that this vector maximizes the likelihood that it will be acceptable for \( I_a \). The higher \( Y \) is, the more samples we collect from the space of possible instances, and so the more reliable our robustness index will be. The robustness, on the other hand, can be increased if we increase \( X \).

\(^1\)We still assume that the real tie-breaking behavior can be captured by a matrix such as \( Q \).
Figure 35: The feasibility of uniform random distributed source load. Note that the CDF curves that are not visible are equal to 100\% for all instances.

5.4.2 Robustness evaluation

We evaluate the robustness of the padding vector that RPV reports using simulations. In the following, we consider a target network with \( N=5 \) ingress links that have the same maxload constraint (i.e., \( C_j=C \) for all \( j \)). The number of source networks is \( M=100 \). The source load distribution is the same for all sources, and it follows one of the following models: Uniform, Exponential, and Pareto (shape parameter: 1.7). The mean source load \( \bar{s} \) is the same in all distributions. The AS-Path length matrix \( P_0 \) is generated based on the empirical distribution shown in Figure 34. The error model that is applied to \( P_0 \) is based on the conditional estimation errors of Table III. The RPV algorithm uses \( X=100 \) and \( Y=10000 \) instances. \( I_X \) is formed by applying errors to the set of sources that generate, in total, at least 80\% of the aggregate load.

The feasibility index depends on the relation between the aggregate offered load \( M\bar{s} \) and the aggregate maxload constraint \( NC \). Given the source load vector \( S \), the following fraction \( \rho \) is our load metric,

\[
\rho = \frac{\sum_{i=1}^{M} s_i}{NC}
\]  

(19)
Figure 36: The feasibility of exponentially random distributed source load. Note that the CDF curves that are not visible are equal to 100% for all instances.

$\rho$ represents well the tightness of the given resource allocation problem for the homogeneous maxload constraints that we consider, and of course it should be less than 100% for any instance to be feasible. To achieve a certain load $\rho$ given an instance with a source load vector $S$, we calculate the required $C$ from (19).

Figures 38-40 show CDFs for the feasibility index and the robustness index of $\bar{A}$ for each of the previous three load distribution models, and for three load conditions ($\rho=0.5$, 0.6, and 0.7). Each CDF resulted from 1000 executions of the RPV algorithm with a different original instance $I_0$. The CDF curves that are not visible in the graphs are equal to 100% for all instances.

Note that even with the Uniform distribution for $S$, which is the most likely among the three to produce feasible instances, the feasibility index drops below 90% when $\rho$ is 0.7 (Fig 35). For the heavy-tailed Pareto sources, the feasibility is often below 90% even when $\rho=0.6$ (Fig 37). The feasibility could be even lower if we had simulated non-homogeneous maxload constraints across different links.

Figures 38-40 show that the robustness index of the padding vector that RPV reports is larger than 90% for all three load conditions, with both the Uniform and
**Figure 37:** The feasibility of source loads with Pareto distribution. Note that the CDF curves that are not visible are equal to 100% for all instances.

Exponential distributions. With the Pareto distribution, on the other hand, the robustness can be lower than 90% in 10-20% of the cases, but rarely lower than 80% (Fig 40). Note that a higher load $\rho$ does not necessarily mean a lower robustness index. The reason is that the latter is conditioned on feasible instances only. It can happen that even though the feasibility index is low, a large number of padding vectors are acceptable for the few feasible instances.

The overall conclusion from these results is that RPV can produce a robust padding vector $\tilde{A}$, in the sense that this vector will be acceptable for the actual instance $I_a$ with a probability of more than 80-90%, if the latter is feasible. For $I_a$ to be feasible with a high probability however, say more than 90%, the load metric $\rho$ should be below 50-70%, depending on the distribution of $S$, at least for the homogeneous maxload constraints that we simulated here.
Figure 38: The robustness of uniform random distributed source load. Note that the CDF curves that are not visible are equal to 100% for all instances.

Figure 39: The robustness of exponentially random distributed source load. Note that the CDF curves that are not visible are equal to 100% for all instances.
Figure 40: The robustness of source loads with Pareto distribution. Note that the CDF curves that are not visible are equal to 100% for all instances.
CHAPTER VI

CONTRIBUTIONS AND FUTURE WORK

Interdomain TE is essential for networks improving the performance of data transferring and reducing network operation cost. The current operation and understanding of interdomain TE, however, is still very preliminary and limited, compared to intradomain TE. This thesis focuses on some widely used interdomain TE techniques in both ingress and egress direction. More specifically, we explore the limitations and potential of AS-Path prepending technique in interdomain ingress TE, investigate possible traffic oscillation due to the interaction between multiple IRC systems, and study the impact of IRC systems to TCP-based network applications. In this chapter, we summarize the major contributions of this thesis, and point out several future research directions.

6.1 Summary of Contributions

- AS-Path prepending

  In this work, we made a first step to explore the use of AS-Path prepending in a more algorithmic framework, and to explore its potential and limitations.

  The main contribution of AS-Path prepending technique at the more theoretical level is to present a polynomial-time algorithm, optimal prepending vector algorithm (OPV), that can determine the optimal prepending vector given constraints on the maximum load of each ingress link. Even though OPV relies on accurate information of several parameters that can only be roughly estimated in practice, it is still important because it provides the best-case scenario for the effectiveness of prepending if all the required information was available. At
the more practical level, the contribution of the paper is to describe how to apply prepending in a robust manner, considering that the AS-Path length information may be subject to estimation errors and the tie-breaking behavior is unknown. Interestingly, our simulations show that it is possible to determine an acceptable padding vector in that case as well, as long as the maximum load constraints are not too tight.

- **Interaction between IRC systems**

In this paper, we first show that when self-load effect is not taken into account for IRC measurement mechanism, a single IRC system can cause sustained traffic oscillations. This result promotes the usage of self-load aware performance metrics such as available bandwidth for path evaluation in IRC systems. We also identify persistent overlapping of measurement periods as another cause of IRC induced traffic oscillations. We show that a new class of randomized path switching algorithms can effectively remove persistent overlapping of measurement periods, thus avoid IRC-induced traffic oscillations. Our algorithms point a new direction for the design of IRC systems.

We study the performance of IRC systems based on self-aware measurement and randomized path switching under both stationary and dynamic traffic environment. We show that the new IRC systems perform well when appropriately configured. Especially, randomized IRC systems over-perform deterministic IRC systems when they co-exist. These findings not only show the performance advantage of randomized IRC systems, but also provide strong incentives for gradual deployment of randomized IRC systems.

- **Interaction between TCP congestion control and IRC**

We further study the performance impact of IRC to network applications, exemplified by TCP connections.
In this piece of work, we first identify the importance of “separation of time scales” principle. That is, IRC is synergistic only when the IRC operation timescale is much larger than the TCP’s, which is typically several RTTs or the RTO of a connection. We then show the RTT and available bandwidth difference can cause transient performance loss, even when traffic is switched to a better path, i.e. a path with shorter RTT or larger available bandwidth. We also point out that path switching based on loss rate comparison may lead to throughput loss. Furthermore, our study on performance of IRC under various types of path impairment shows that IRC is more beneficial to TCP under severe impairment that last longer than the IRC operation timescale, such as path outage or very loss links, than short light impairment, such lossy links with low loss rate.

This work is the first investigating the impact of IRC on the performance of end applications. Even thought the study focuses on IRC, the results can extend to other interdomain egress TE techniques, and the guidelines on design and configuration of IRC systems also apply to these interdomain egress TE techniques as well.

6.2 Future Work

Following are a list of future research directions.

- Advanced study of performance impact of IRC on network applications

In this thesis, we take the first step investigating the performance impact of IRC on network applications. Our traffic models are limited to persistent TCP connection or closed loop TCP flows, and the impairment we consider only include random loss and outage. The next step is to advance this study by
including more traffic models representing different types of applications, and richer impairment models.

- **Experimental study of IRC systems**

So far, our work on IRC is based on simulations, at both flow level and packet level. Further study can be based on emulation in the Internet. Here we present a preliminary framework for an experimental study, as shown in Figure 41.

![Source network with IRC](image1)

![Proxies](image2)

![Destination network](image3)

**Figure 41:** Design of experiments on IRC emulation in the Internet.

The experiment consists of three elements: an IRC system at a multi-homed source network, several intermediate proxies, and a receiver at the destination network. The IRC system can be built on a Linux box, including the traffic generation module, the measurement module, and the path switching module. The functionality of the intermediate proxies is to forward traffic from the source network to the destination network, and they are selected in such way that the source network reaches them through different ISPs. Thus, by switching traffic across the proxies, the source network switches among the ISPs.

The setup of one IRC system, shown in Figure 41, will provide us a testbed for experiments on performance impact of IRC systems. For the study of interaction of multiple IRC systems, we will need a setup shown in Figure 42. The IRC
systems in source networks can be implemented in PlanetLab [38], and the proxies are implemented on Resilient Overlay Network (RON) servers [6], and the destination locates in Georgia Tech network.

**Figure 42:** Experiment setup for IRC interaction study

- **AS-Path prepping with dynamic inbound traffic demand**

  In this thesis, we consider AS-Path prepping of a single target AS and assume that the inbound traffic demand is stationary. However, stationary inbound traffic assumption doesn’t always hold in the dynamic Internet environment, especially when interdomain TE is used in neighboring or upstream networks. In future research, our AS-Path prepping work can be extended in the following ways.

  - Extending OPV for dynamic inbound traffic demand

    We develop OPV algorithm based on the assumption that inbound traffic from major source networks is stationary and use the source traffic vector as an input. The extension of OPV work should investigate the scenario when inbound traffic is non-stationary. We can approach OPV algorithm for non-stationary inbound traffic throught two possible ways. First, this algorithm can be designed in similar fashion to Robust Prepending Vector (RPV) algorithm. For RPV, we investigate possible estimation
errors of certain inputs and develop a random sampling technique for robust solutions. Second, we change the objective for the Constraint Optimal Prepending (COP) problem from meeting the capacity constraints of ingress links to maximize the minimum headroom of each ingress link. This way, the solution provides maximum headroom for traffic to expand.

- Interactions of AS-Path prepending among multiple networks

Our AS-Path prepending work considers AS-Path prepending of a single AS. However, when multiple ASes use AS-Path prepending at the same time, routing anomalies and traffic instabilities can be caused. Supportive clues can be found in measurement-based study on AS-Path prepending [42]. The results reveal that the routing changes caused by prepending can be very complex, especially when prepending is used by multiple networks in the upstream cloud. Moreover, the location of prepending can be critical to the effectiveness. The future work of AS-Path prepending should investigate the interaction of AS-Path prepending of multiple networks, including when OPV algorithm is used.
REFERENCES


108


